ADVANCED DATABASE MANAGEMENT SYSTEMS

MCS - 043

For

Masters In Computer Applications [MCA]

Dinesh Verma
S. Roy

Useful For

IGNOU, KSOU (Karnataka), Bihar University (Muzaffarpur), Nalanda University, Jamia Millia Islamia, Vardhman Mahaveer Open University (Kota), Uttarakhand Open University, Kurukshetra University, Seva Sadan's College of Education (Maharashtra), Lalit Narayan Mithila University, Andhra University, Pt. Sunderlal Sharma (Open) University (Bilaspur), Annamalai University, Bangalore University, Bharathiar University, Bharathidasan University, HP University, Centre for distance and open learning, Kakatiya University (Andhra Pradesh), KOU (Rajasthan), MPBOU (MP), MDU (Haryana), Punjab University, Tamilnadu Open University, Sri Padmavati Mahila Visvavidyalayam (Andhra Pradesh), Sri Venkateswara University (Andhra Pradesh), UCSDE (Kerala), University of Jammu, YCMOU, Rajasthan University, UPRTOU, Kalyani University, Banaras Hindu University (BHU) and all other Indian Universities.

GPH BOOK ®

GULLYBABA PUBLISHING HOUSE PVT LTD.
ISO 9001 & ISO 14001 CERTIFIED CO.

Published by:

GullyBaba Publishing House Pvt. Ltd.

Regd. Office:
2525/193, 1st Floor, Onkar Nagar-A,
Tri Nagar, Delhi-110035
(From Kanhaiya Nagar Metro Station Towards Old Bus Stand)
011-27387998, 27384836, 27385249
+919350849407

Branch Office:
1A/2A, 20, Hari Sadan,
Ansari Road, Daryaganj,
New Delhi-110002
Ph. 011-45794768

E-mail: hello@gullybaba.com, **Website**:GullyBaba.com, GPHbook.com

New Edition

Price: ₹129/-

ISBN: 978-93-81638-09-5

Disclaimer: Although the author and publisher have made every effort to ensure that the information in this book is correct, the author and publisher do not assume and hereby disclaim any liability to any party for any loss, damage, or disruption caused by errors or omissions, whether such errors or omissions result from negligence, accident, or any other cause.

If you find any kind of error, please let us know and get reward and or the new book free of cost.

The book is based on IGNOU syllabus. This is only a sample. The book/author/publisher does not impose any guarantee or claim for full marks or to be passed in exam. You are advised only to understand the contents with the help of this book and answer in your words.

All disputes with respect to this publication shall be subject to the jurisdiction of the Courts, Tribunals and Forums of New Delhi, India only.

Home Delivery of GPH Books

You can get GPH books by VPP/COD/Speed Post/Courier.
You can order books by Email/SMS/WhatsApp/Call.
For more details, visit gullybaba.com/faq-books.html
Our packaging department usually dispatches the books within 2 days after receiving your order and it takes nearly 5-6 days in postal/courier services to reach your destination.

Note: Selling this book on any online platform like Amazon, Flipkart, Shopclues, Rediff, etc. without prior written permission of the publisher is prohibited and hence any sales by the SELLER will be termed as ILLEGAL SALE of GPH Books which will attract strict legal action against the offender.

Preface

This book (MCS-043) is mainly targeted for the exam of 'Advanced Database Management Systems' for all Universities. This topic has been in demand and is the need of current scenario. Providing well researched material to the learners, with focus on examination and learning has been a constant pursuit for us. We at GPH, believe in the power of education and respect the growth and positive contribution that education and respect contribution that education can contribute to a society.

The unavailability of material in the market is a demotivating factor for the students and searching and collecting it, becomes a tough task for the student community. We, at GPH wish to strive for excellence as we perceive it not only as earning a degree but a step towards the growth and enrichment.

GPH Books are the pioneers in the field and an effort to provide a Strategy and a unique methodology so as to give you excellent performance in the examination. Your goal of attaining a higher grade and securing your future can be made accessible by the use of our published material. Your goal can thus be achievable and realistic! Hoping for your better, brighter and a promising and successful future!

Our website is www.gullybaba.com. It is a vital resource for your exams and can have manifold effect on our meticulous preparation. Now you can access us on the net thorough ***www.doeacconline.com, www.ignounline.com****, and* ***www.astrologyeverywhere.com.***

We greatefully acknowledge the significant contributions of our experts in bringing out this publication.

Dear Reader,

Welcome in the world of GullyBaba Publishing House (P) Ltd. Profound, in-depth study and research can guarantee you the most reliable, latest & accurate information on the subject. However, as the saying goes, nothing is perfect, but we still believe that there is always a scope for improvement. And we wish to be nothing less but aim for the best.

You, the reader can be our best guide in making this book more interesting and user friendly.

Your valuable suggestions are welcome!

Feedback about the book can be sent at **feedback@gullybaba.com.**
Publisher.

TOPICS COVERED

Block—1 Data Base Design and Implementation

Block—2 DBMS Advanced Features and Distributed Database

Block—3 Enhanced Database Models

Block—4 Emerging Trends and Example DBMS Architectures

Contents

Question Papers

GullyBaba Publishing House
2525/193, Ist Floor, Kanhaiya Nagar Metro Station,
Tota Ram Bazar, Onkar Nagar-A, Tri Nagar, Delhi-110035
Phone : 011-27387998, 9350849407, 9312235086

Chapter-1

DATABASE DESIGN

Q1. What is the need of EER model?

Ans. The ER modeling concepts which are sufficient for representing many database schemas for "traditional" database applications, which mainly include data-processing applications in business and industry. Since the late 1970s, however, newer applications of database technology have become commonplace; these include databases for engineering design and manufacturing (CAD/CAM), telecommunications, images and graphics, multimedia, data mining, data warehousing, geographic information systems (GIS), and databases for indexing the World Wide Web, among many other applications. These types of database have more complex requirements than do the more traditional applications. To represent these requirements as accurately and clearly as possible, designers of database applications must use additional semantic data modeling concepts.

Features that have been proposed for semantic data models, leading to the enhanced-ER or EER model. The EER(Enhance-ER) model includes all the modeling concepts of the ER model. In addition, it includes the concepts of subclass and superclass and the related concepts of specialization and generalization. Another concept included in EER model is that of category which is used to represent a collection of objects that is the union of objects of different entity types.

Q2. What is the use of EER diagram?

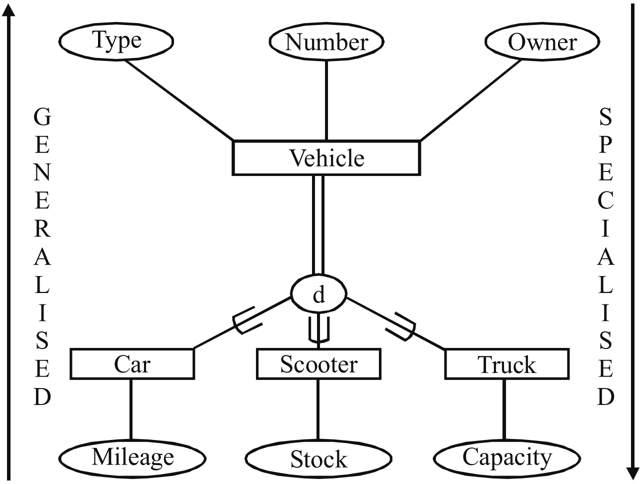
Ans. The EER diagrams are used to model advanced data model requiring inheritance, specialisation and generalisation.

Q3. Define Specialisation & Generalisation. Represent them in an example of diagram. Also discuss their constraints.

Ans. The process of defining the subclasses of an entity type is called **specialization**, where the entity type is called the super class of the specialization.

Generalisation is the reverse process of specialization; in other words, it is a process of suppressing the differences between several entity types, identifying their common features into a single super class. For example, the entity type

CAR and TRUCK can be generalized into entity type VEHICLE. Therefore, CAR and TRUCK can now be subclasses of the super class generalized class VEHICLE.



Generalisation and specialisation

Constraints and Characteristics of Specialisation and Generalisation

A super class may either have a single subclass or many subclasses in specialization. In case of only one subclass we do not use circle notation to show the relationship of subclass / super class. Sometimes in specialization, the subclass becomes the member of the super class after satisfying a condition on the value of some attributes of the super class. Such subclasses are called condition defined subclasses or predicate defined subclasses. For example, vehicle entity type has an attribute vehicle "type", as shown in the above figure.

Disjointness is also the constraints to a specialisation. It means that an entity can be a member of at most one of the subclasses of the specialisation. In an attribute-defined specialization the disjointness constraint means that an entity can be a member of a single sub-class only. In the above figure, the symbol 'd 'in circle stands for disjoint.

But if the real world entity is not disjoint their set of entities may overlap; that is an entity may be a member of more than one subclass of the specialization. This is represented by an (o) in the circle. For example, if we classify cars as luxury cars and cars then they will overlap.

In some cases, a single class has a similar relationship with more than one class. For example, the sub class 'car' may be owned by two different types of owners:
INDIVIDUAL or ORGANISATION. Both these types of owners are different classes thus such a situation can be modeled with the help of a Union (u).

Q4. What are the constraints used in EER diagrams?
Ans. The basic constraints used in EER diagrams are disjointness, overlapping and unions.

Q5. How is an EER diagram converted into a table?
Ans. For disjointness and union constraints the chances are that we create separate tables for the subclasses and no table for super class. For overlapping constraints it is advisable to have a table of super class. For such cases the tables of subclasses will have only those attributes that are not common to super class except for the primary key.

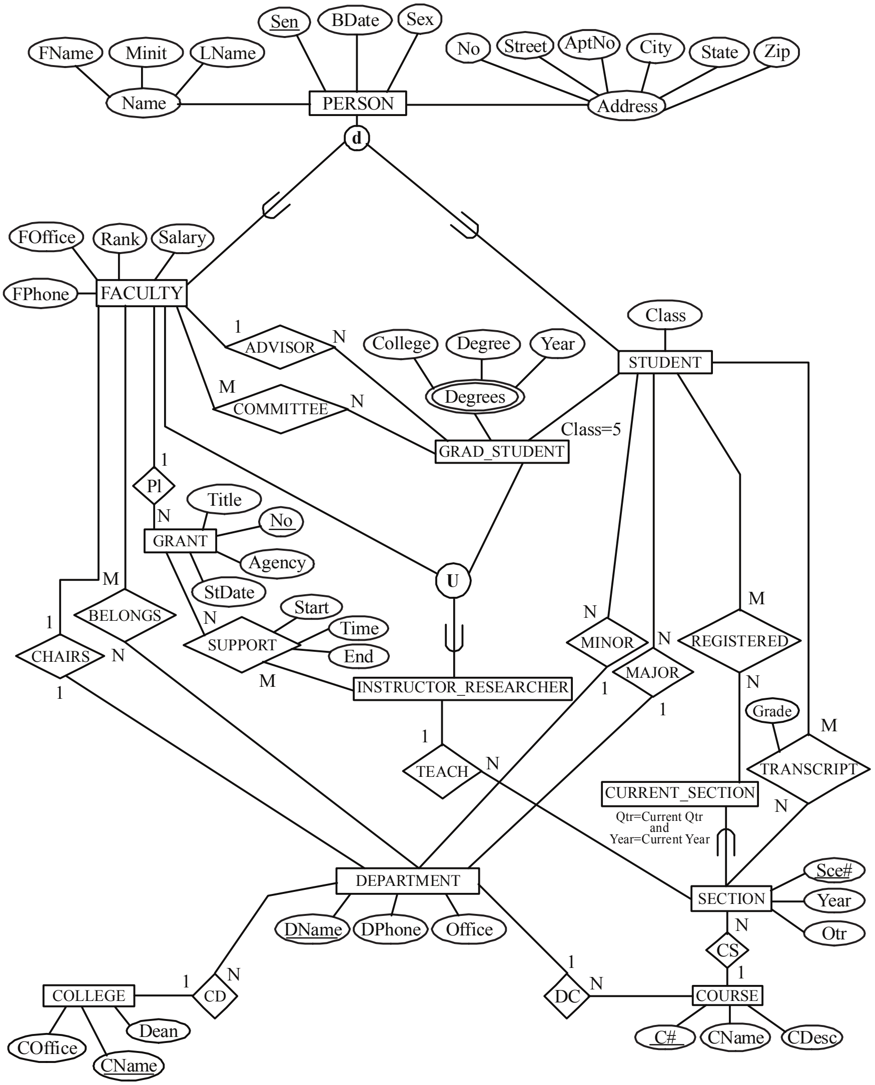
Q6. Explain the EER diagram with example of an institution.
Ans. The **INSTITUTE Database Example:** Consider a INSTITUTE database that keeps track students and their majors, transcripts and registration as well as of the institute's course offerings. The database also keeps track of the sponsored research projects of faculty and graduate students. A discussion of the requirements that led to this schema follows.
For each person, the database maintains information on the person's Name **Name**, social security number **Ssn**, address **Address**, sex **Sex**, and birth date **BDate**. Two subclasses of the PERSON entity type were identified: FACULTY and STUDENT. Specific attributes of FACULTY are rank **Rank** (assistant, associate, adjunct, research, visiting, etc.), office **FOffice**, office phone **FPhone**, and salary **Salary**, and all faculty members are related to the academic department(s) with which they are affiliated **BELONGS** (a faculty member can be associated with several departments, so the relationship is M:N). A specific attribute of STUDENT is [Class] (freshman = 1, sophomore = 2, … , graduate student = 5). Each student is also related to his or her major and minor departments, if known (**MAJOR** and **MINOR**), to the course sections he or she is currently attending **REGISTERED**, and to the courses completed **TRANSCRIPT**. Each transcript instance includes the grade the student received **Grade** in the course section.
GRAD_STUDENT is a subclass of STUDENT, with the defining predicate Class = 5. For each graduate student, we keep a list of previous degrees in a composite, multivalued attribute **Degrees**. We also relate the graduate student to a faculty advisor **ADVISOR** and to a thesis committee **COMMITTEE** if

one exists.

An academic department has the attributes name **DName**, telephone **DPhone**, and office number **Office** and is related to the faculty member who is its chairperson **CHAIRS** and to the college to which it belongs **CD**. Each college has attributes college name **CName**, office number **COffice**, and the name of its dean **Dean**.

A course has attributes course number **C#**; course name **Cname**, and course description **CDesc**. Several sections of each course are offered, with each section having the attributes section number **Sec#** and the year and quarter in which the section was offered (**Year** and **Qtr**). Section numbers uniquely identify each section. The sections being offered during the current semester are in a subclass CURRENT_SECTION of SECTION, with the defining predicate Qtr = CurrentQtr and Year = CurrentYear. Each section is related to the instructor who taught or is teaching it (**TEACH**, if that instructor is in the database).

The category INSTRUCTOR_RESEARCHER is a subset of the union of FACULTY and GRAD_STUDENT and includes all faculty, as well as graduate students who are supported by teaching or research. Finally, the entity type GRANT keeps track of research grants and contracts awarded to the university. Each grant has attributes grant title **Title**, grant number **No**, the awarding agency **Agency**, and the starting date **StDate**. A grant is related to one principal investigator **PI** and to all the researchers it supports **SUPPORT**. Each instance of support has an attributes the starting date of support **Start**, the ending date of the support (if known) **End**, and the percentage of time being spent on the project **Time** by the researcher being supported.



An EER conceptual schema for Institute offering Various courses

Q7. What are Multi valued Dependencies? When we can say that a constraint X is multi determined.

Ans. An MVD is a constraint due to multi-valued attributes. A relational must have at least 3 attributes out of which two should be Multi-valued.

Q8. Decompose the following into 4NF

EMP

ENAME	PNAME	DNAME
Krishna	X	Vijay
Krishna	Y	Manu
Krishna	X	Manu
Krishna	Y	Vijay

Ans.EMP_PROJECTS

ENAME	PNAME
Krishna	X
Krishna	Y

EMP_DEPENDENCIES

ENAME	DNAME
Krishna	Vijay
Krishna	Manu

Q9. Does the following relation satisfy MVDs with 4NF? Decompose the following relation into 5NF.

SUPPLY

SNAME	PARTNAME	PROJNAME
Krishna	Bolt	X
Krishna	Nut	Y
Radha	Bolt	Y
Vasudev	Nut	Z
Radha	Nail	X
Radha	Bolt	X
Krishna	Bolt	Y

Ans. No,

R1

SNAME	PARTNAME
Krishna	Bolt
Krishna	Nut
Radha	Bolt
Vasudev	Nut
Radha	Nail

R2

SNAME	PROJNAME
Krishna	X
Krishna	Y
Radha	Y
Vasudev	Z
Radha	X

R3

PARTNAME	PROJNAME
Bolt	X
Nut	Y
Bolt	Y
Nut	Z
Nail	X

Q10. Define 5NF. Explain why the following relation does not satisfy 5NF and decompose it to satisfy 5NF. What advantages are gained by this decomposition? [Dec06(s), Q1(i)]

Parent	**Child**	**Hobby**
P1	**C1**	**H1**
P1	**C1**	**H2**
P1	**C2**	**H3**
P2	**C3**	**H3**
P2	**C4**	**H3**
P3	**C4**	**H4**

Ans. 5NF: A Relation is said to be in the fifth normal form (5NF) if every join dependency is a consequence of its relation (candidate keys).
Alternatively, for each and every non-trivial join dependency * (R_1, R_2, R_3) each decomposed Relation R_1 is a super key of the main Relation.
5NF is also called project-Join Normal Form (PJNF).

Relation given as below:

Parent	Child	Hobby
P1	C1	H1
P1	C1	H2
P1	C2	H3
P2	C3	H3
P2	C4	H3
P3	C4	H4

Projections:

D1:

Parent	Child
P_1	C_1
P_1	C_2
P_2	C_3
P_2	C_4
P_3	C_4

D2:

Parent	Hobby
P_1	H_1
P_1	H_2
P_1	H_3
P_2	H_3
P_3	H_4

D3:

Parent	Hobby
P_1	H_1
P_1	H_2
P_1	H_3
P_2	H_3
P_3	H_4

$D1 \bowtie D2$

Parent	Child	Hobby
P1	C1	H1
P1	C1	H2
P1	C1	H3
P1	C2	H1
P1	C2	H1
P1	C2	H2
P1	C2	H3
P2	C3	H3
P2	C4	H3
P3	C4	H4

Spurious tuple (s)

D1⋈ D3

Parent	Child	Hobby
P1	C1	H1
P1	C1	H2
P1	C2	H3
P2	C3	H3
P2	C4	H3
P2	C4	H4
P3	C4	H3
P3	C4	H4

Spurious tuple(s)

D2⋈ D3

Parent	Child	Hobby
P1	C1	H1
P1	C1	H2
P1	C2	H3
P1	C3	H3
P2	C2	H3
P2	C3	H3
P3	C4	H4

Spurious tuple (s)

record missing: P2 C4 H3

But D1 = D2 = D3 = R

So *(D1, D2, D3) is join dependency

*((Parent, Child), (Child, Hobby), (Parent, Hobby)) is a join dependency.

The consequence of these join dependencies is that parent, child or Hobby is not relation key.

Decompose Relation to Satisfy 5NF:

R Decomposed in

R1 (Parent, Child),
R2 (Parent, Hobby), and
R3 (Child, Hobby)
Advantage of 5NF Decomposition
Relations R1 R2 and R3
are in 5NF
as it satisfy trivial join-dependency
So, no need to insert a value of Hobby when new child record is inserted.

Q11. When is a set of functional dependencies F minimal?

Ans. A set of functional dependencies F is minimal if it satisfies the following conditions:

- Every dependency in F has a single attribute for its right-hand side.
- We cannot replace any dependency $X \rightarrow A$ in F with Dependency $Y \rightarrow A$, where Y is a proper subset of X and still have a set of dependencies that is equivalent to F.
- We cannot remove any dependency from F and still have a set of dependencies that is equivalent to F.

Q12. Give the proof of transitive rule $\{X \rightarrow Y, Y \rightarrow Z\} | = X \rightarrow Z$.

Ans. 1. $X \rightarrow Y$ and 2. $Y \rightarrow Z$ both holds in a relation r. Then for any two tuples t1 and t2 in r such that t1[X] = t2[X], we must have 3. t1[Y] = t2[y], from assumption 1; hence we must also have 4. t1[Z] = t2[z], from 3 and assumption 2; hence $X \rightarrow Z$ must hold in r.

Q13. Define Inclusion Dependencies.

Ans. An inclusion dependency R.X < S.Y between two sets of attributes – X of a relation schema R, and Y of a relation schema S – is defined as the following constraint:

If r and s are the relation state of R and S respectively at any specific time then:

$_{x}(r(R)) \subseteq {}_{Y}(s(S))$

The subset relationship does not necessarily have to be a proper subset. Please note that the sets of attributes on which the inclusion dependency is specified viz., *X* of *R* and *Y* of *S* above, must have the same number of attributes. In addition, the domains for each corresponding pair of attributes in *X* and *Y* should be compatible.

Q14. What is the key idea behind DKNF?

Ans. The key idea behind DKNF is to specify the “ultimate normal form.”

Q15. Define Template Dependency.

Ans. The template dependencies are the more general and natural class of data dependencies that generalizes the concepts of JDs. A template dependency is representation of the statement that a relation is invariant under a certain tableau mapping. Therefore, it resembles a tableau. It consists of a number of hypotheses rows that define certain variables with a special row at the bottom, called the conclusion row. A relation r satisfies a template dependency, if and only if , a valuation (say ρ) that successfully maps the hypothesis rows to tuples in a relation r, finds a map for conclusion row to a tuple in r.

Q16. How is template dependency is different from a tableau ? Explain with example.

Ans. A template dependency is different from a tableau in the following two ways:

(1) A variable like (a, b, c etc.) in the conclusion row need not appear in any of the hypothesis row.

(2) Variables may not be necessarily restricted to a single column.

Let us show both the points above with the help of an example each.

Example : Consider the TD T on schema A B C in Figure below. It is a valid TD expression; variable c is not appearing in the hypothesis rows where the variables are c' and c". This TD has the variable of conclusion row on A and B in the hypothesis rows, but not on C, therefore, is called A B-partial.

T	(A	B	C)
a'		b	c"
a'		b	c'
a		b'	c"
a		b	c

Figure : A sample A B-Partial TD T

A TDT on schema R where every variable in the conclusion row appears in some hypothesis's row is termed as full. Consider a TD having w1, w2,..., wk as the hypothesis rows and w as the conclusion row, a TD is called S-partial, where S is the set defined as: $\left\{S \in R \mid w\left(S\right)\right\}$ appears in one of w1, w2, ..., wk). The TD is full if the S = R and strictly partial if $S \neq R$.

A TD in which each variable appears in exactly one column is called a typed TD, but if some variable appears in multiple columns then it is called an untyped

TD.

Another Example : Figure below shows an untyped TDT. This TD assumes that the domain of A is same as that of domain of B, otherwise such TD will not make any sense.

T	(A	B)
b		c
a		b
a		c

Figure : Untyped TD T

Let us now show the relationship of JD and MVD to the TD.

Another Example : Consider the MVD $A \rightarrow\rightarrow B$ over the relation schema A B C is equivalent to the TD T in Figure below. TDT indicates that if a relation has two tuples t1 and t2 that agree on A, it must also have a tuple t3 such that t3 (A B) = t1 (A B) and t3 (A C) = t2 (A C), which is just a way of stating that the relation satisfies $A \rightarrow\rightarrow B$

T	(A	B	C)
a		b	c'
a		b'	c
a		b	c

Figure : A TD T for MVD

However, please note that not every TD corresponds to a JD. This can be ascertained from the fact that their can an infinite number of different TDs over a given relation schema, whereas there is only a finite set of JDs over the same schema. Therefore, some of the TDs must not be equivalent to any JD.

Q17. List the various points which should be kept in mind while modeling the time in a temporal database system.

In a temporal database system you need to model time keeping the following points in mind:

- You need to define database as a sequence of time based data in chronological order.
- You need to resolve events that happen at the same time.
- A reference point of time may be defined to find time relative to it.
- Sometimes a calendar is used.

- The SQL support for temporal data includes:

(i) data types such as Date (dd, mm, yyyy,

(ii) TIME (hh:mm:ss), TIMEST AMP.

which specifies a unique sequence number, based on time, to identify sequence of events/activities, INTERVAL (time durations) and PERIOD (period frame reference point).

(iii) You can also define the concept of valid time for a data entity for example an assignment may be valid till a particular time.

Q18. Why is the DBMS most suitable for Information System implementation?

Ans. The following characteristics make DBMS a very valuable tool for information systems:

- Data independence
- Data sharing under secure environment
- Simple query mechanism
- Support for transaction and recovery
- Availability of tools for knowledge extraction.

Q19. Which of the information System Life Cycle stages are important for database design?

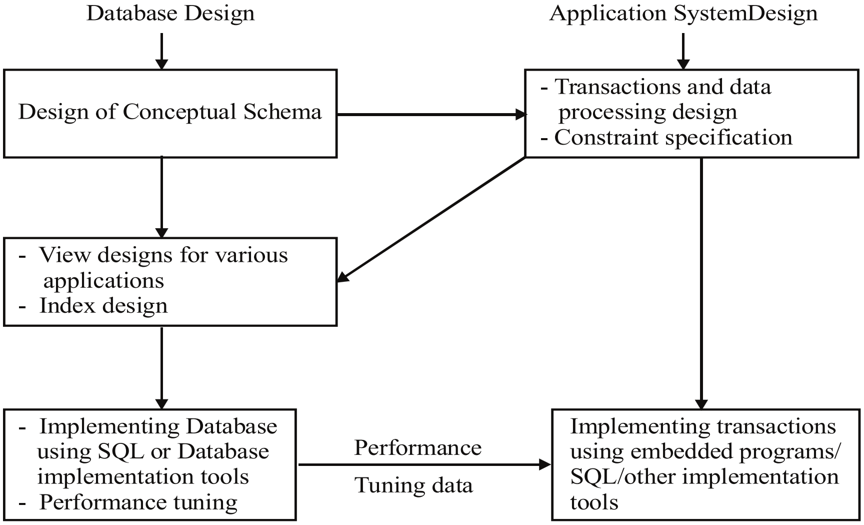
Ans. The stages where a lot of useful input is generated for database design are communication and modeling.

Chapter-2

DATABASE IMPLEMENTATION

Q1. Why database application design activity in an organisation is very important? Also draw and explain the diagramatic representation of database application design.

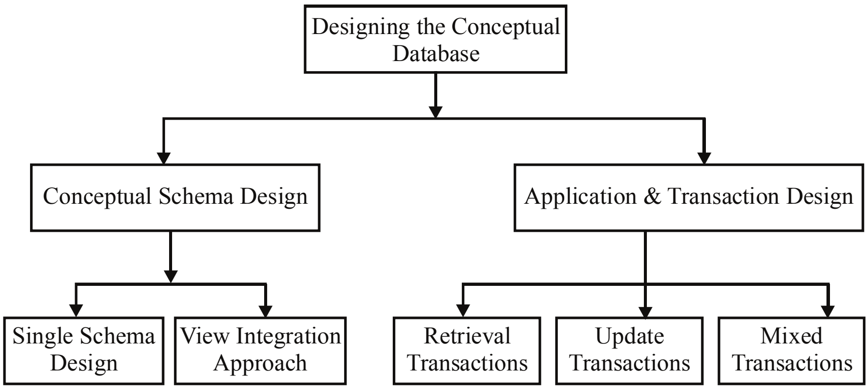
Ans. The database application design activity in an organisation is very important. This activity determines the overall quality of the information system. The database application design involves database design and application system design. The Figure shows the steps of these processes and the linkage for database application design and implementation.


Database Design
Application SystemDesign
Design of Conceptual Schema
- Transactions and data processing design
- Constraint specification
- View designs for various applications
- Index design
- Implementing Database using SQL or Database implementation tools
- Performance tuning
Performance
Tuning data
Implementing transactions using embedded programs/ SQL/other implementation tools

Database application system design and implementation

Q2. Explain the process of conceptual Database design.

Ans. The process of conceptual database design is given in Fig.



Conceptual Schema Design :

The following details are produced during the conceptual database design :

- Database structures and data types associated with each field.
- Keys and constraints.
- Data irrelationships such as referential constraints.
- Data dictionary.

The process of conceptual schema design requires primarily modification and conversion of E-R diagram into tables keeping the following points in consideration:

- Proper identification of entities, relationship, and attributes.
- Identification of key attributes.
- Identification of cardinality of relationship.
- Identifying weak entities, specialisation/generalisation etc.

Two common approaches for conceptual schema design :

a) Single Schema Design

b) View Integration

(i) Single Schema Design : This process results in the development of single schema.

(ii) View Integration : A commercial database may be very complex. It may be a good idea to model this database as per the viewpoints of various stakeholders, thus creating many views. A conceptual schema may be an integration of all such views. Integration involves the following steps :

- finding out the common entities and relationships of interest
- finding out if any conflicting requirements exist, such as name, type, constraints etc.
- modifying and manage the views.
- Refining the final model, if necessary

Q3. List the important considerations at logical & physical database design stages of database design.

Ans. Important considerations at this stage of database design are :

- The database views should be identified.
- The security considerations and constraints on the view should be identified and duly specified.
- The performance criteria for views may be identified.
- The physical database design which includes specific storage structure (e.g., clustered file organisation) and access paths (indexes etc.) may be identified.

Q4. Why is physical database design required?

Ans. A high-performance *transaction process systems* requires continuous operation of the system. Transaction performance, in terms of the average number of transactions per minute and maximum transaction response time, is critical. Hence, a careful physical database design that meets the organisation's transaction processing needs is a must.

Q5. Why do we choose a specific physical structure?

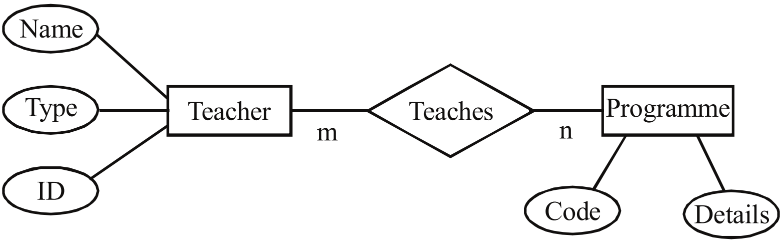
Ans. We do it so that the database application can achieve good performance. The physical database design process is restricted to choosing the most appropriate structure for the database files from among the options offered by that DBMS.

Q6. What are the steps for design of database system?

Ans. Following are the steps for design of database system:

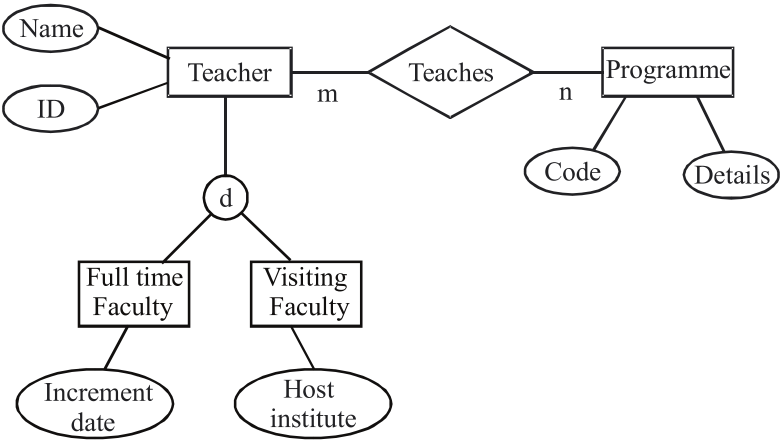
- Conceptual and Schema Design
- Logical and physical schema design
- Data processing and transaction design.

Q7. Consider the following EER diagram.



'Type' can be regular or visiting faculty. A visiting faculty members can teach only one programme. Make a suitable database application design for this.

Ans. The modified EER diagram is:



TEACHER (id, Name, Increment-date)
VISITING (id, Name, Host-institute)
TEACHES (id, code)
PROGRAMME (code, details)

Transaction/constraints: The **id** in TEACHES, if exists in id in visiting then COUNT on 'id' in teaches should not exceed 1.

Q8. List the criteria for physical design?

Ans. Following is the criteria list for physical data :

* Response time

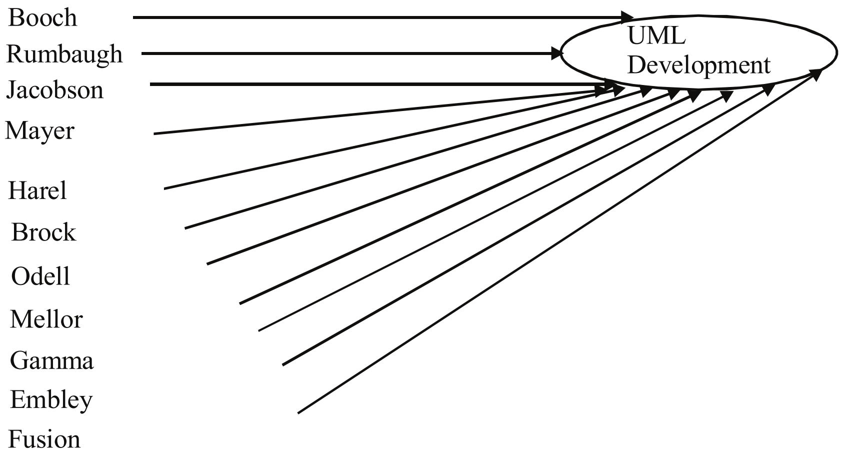
* Space utilization

* Transaction throughput.

Q9. What is UML? Describe its main features, also what are major building blocks of UML? Why is it widely used?

Ans. The Unified Modeling Language (UML) is used to express the construct and the relationships of complex systems. It was created in response to a request for proposal (RFP) from the Object Management Group (OMG). Earlier in the 1990s, different methodologies along with their own set of notations were introduced in the market. The three prime methods were OMT (Rumbaugh), Booch and OOSE (Jacobson). OMT was strong in analysis, Booch was strong in design, and Jacobson was strong in behavioral analysis. UML represents unification of the Booch, OMT and OOSE notations, as well as are the key points from other methodologies. The major contributors in this development shown in Figure below.

UML is an attempt to standardize the artifacts of analysis and design consisting of semantic models, syntactic notations and diagrams. The first draft (version 0.8) was introduced in October 1995. The next two versions, 0.9 in July 1996 and 0.91 in October 1996 were presented after taking input from Jacobson. Version 1.0 was presented to Object Management Group in September 1997. In November 1997, UML was adopted as standard modeling language by OMG. The current version while writing this material is UML 2.0.



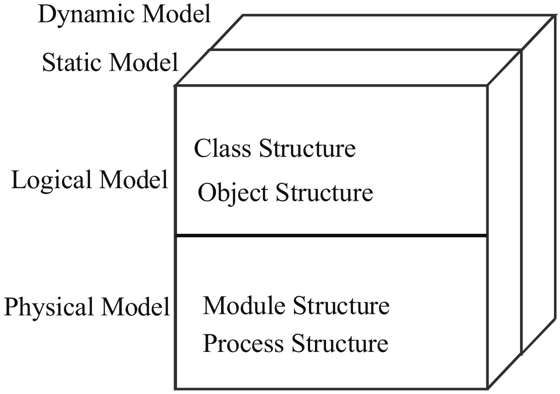
The Input for UML development

The major features of UML are:

- defined system structure for object modeling
- support for different model organization
- strong modeling for structure and behavior
- clear representation of concurrent activities

- support for object oriented patterns for design reuse.

The model for the object oriented development could be shown as in *Figure*. It could be classified as static/dynamic and logical/physical model.



The Model for Object oriented development

The Logical view of a system serves to describe the existence and meaning of the key abstractions and the mechanism that form the problem space, or that define the system architecture.

The Physical model describes the concrete software and hardware components of the system's context or implementation. UML could be used in visualizing, specifying, constructing and documenting object oriented systems. The major building blocks of UML are structural, behavioral, grouping, and annotational notations. Let us discuss these blocks, one by one.

(a) Structural Notations: These notations include static elements of a model. They are considered as **nouns** of the UML model which could be conceptual or physical. Their elements comprises class, interface, collaboration, use case, active class, component, and node. It also includes actors, signals, utilities, processes, threads, applications, documents, files, library, pages, etc.

(b) Behavioral Notations: These notations include dynamic elements of a model. Their elements comprises interaction, and state machine. It also includes classes, collaborations, and objects.

(c) Grouping Notations: These notations are the boxes into which a model can be decomposed. Their elements comprises of packages, frameworks, and subsystems.

(d) Annotational Notations: These notations may be applied to describe, illuminate, and remark about any element in the model. They are considered as explanatory of the UML. Their elements comprised of notes which could be used for constraints, comments and requirements.

UML is widely used as it is expressive enough, easy to use, unambiguous and is supported by suitable tools.

Q10. Describe the object modeling notations of UML.

Ans. A **system** is a collection of subsystems organised to accomplish a purpose and described by a set of models from different viewpoints.

A **model** is a semantically closed abstraction of a system which represents a complete and self-consistent simplification of reality, created in order to better understand the system.

A **view** is a projection into an organisation and structure of a system's model, focused on one aspect of that system.

A **diagram** is a graphical presentation of a set of elements.

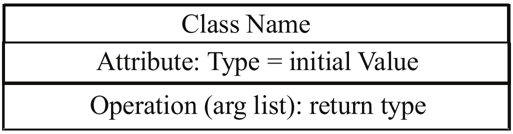
A **classifier** is a mechanism that describes structural and behavioral features. In UML the important classifiers are class, interface, data type, signals, components, nodes, use case, and subsystems.

A **class** is a description of a set of objects that share the same attribute, operations, relationships, and semantics. In UML, it is shown by a rectangle.

A **attribute** is a named property of a class that describes a range of values that instances of the property may hold. In UML, they are listed in the compartment just below that class name.

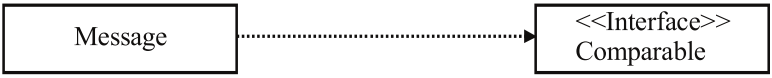
An **operation** is the implementation of a service that can be requested from any object of the class to affect behavior. In UML, they are listed in the compartment just below that class attribute.

The notation for class, attribute, and operations is shown as:


Class Name
Attribute: Type = initial Value
Operation (arg list): return type

Class with attributes and operations

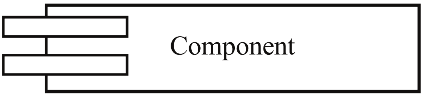
An **interface** is a collection of operations that are used to specify a service of a class or a component.



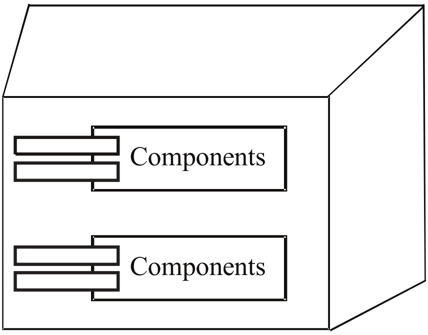
Realizing an interface

A **signal** is the specification of an asynchronous stimulus communicated between instances.

A **component** is a physical and replaceable part of the system that confirms to, and provides the realization of a set of interfaces. In UML, it is shown as a rectangle with tabs. The notation for component and interface is shown as:

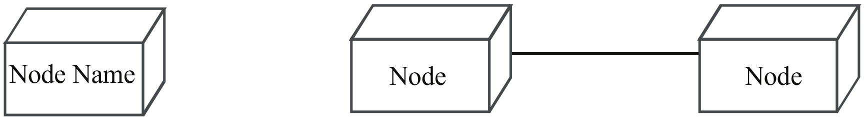


Component



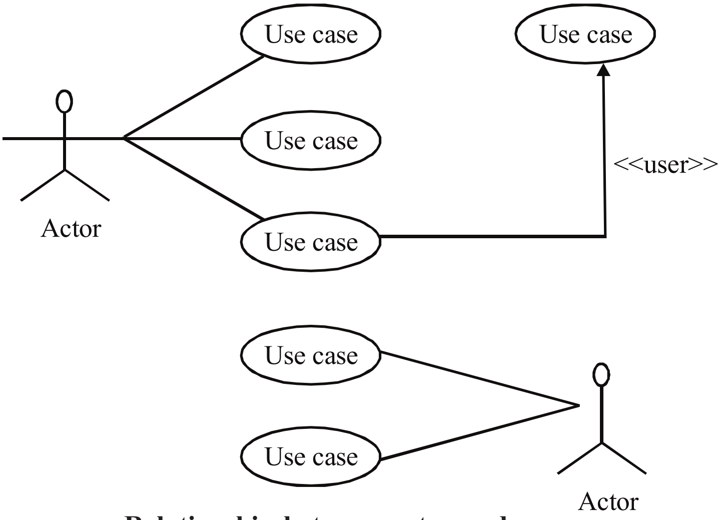
Components and Server

A **node** is a physical element that exists at runtime, and represents a computational resource generally having a large memory and often process capability. In UML, it is shown as a cube. The notation for node is shown as:



Node and relationship between nodes

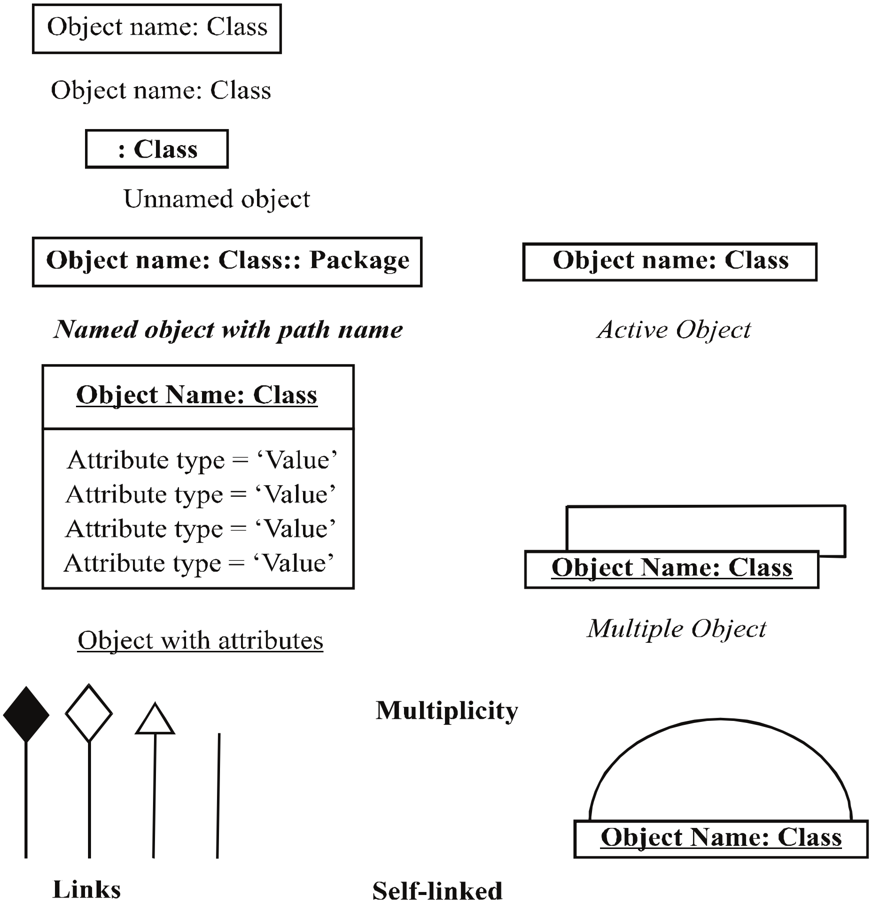
A **use case** is a description of a set of sequence of actions that a system performs to yield an observable result that is of a value to an actor. The user is called actor and the process is depicted by use case. The notation for use case is shown as:



Relationship between actor and use case

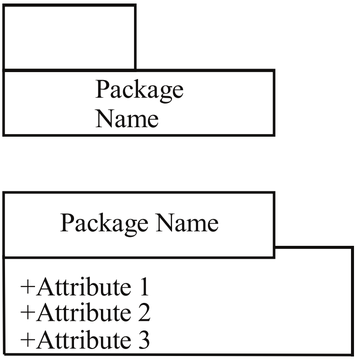
A **subsystem** is a grouping of elements of which some constitute a specification of the behavior offered by other contained elements.

An **object** is an instance of a class. The object could be shown as the instance of a class, with the path name along with the attributes. Multiple objects could be connected with links. The notation for unnamed and named objects, object with path name, active objects, object with attributes, multiple objects, and self linked object is shown as:



Different types of objects

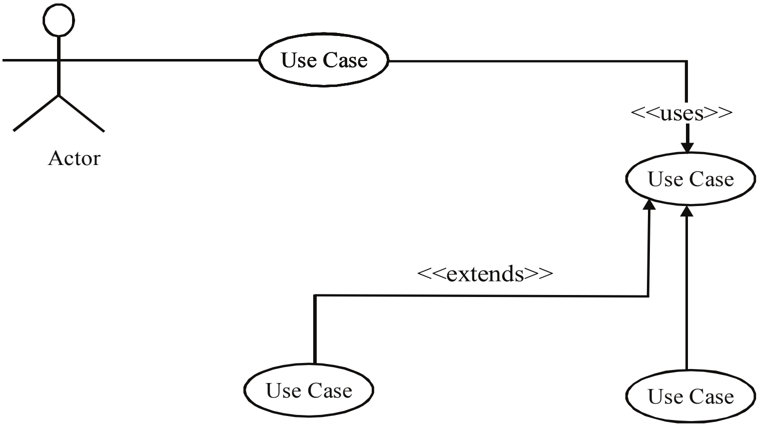
A **package** is a general purpose mechanism for organising elements into groups. It can also contain other packages. The notation for the package shown contains name and attributes as shown in *Figure*. Packages are used widely in a Java based development environment.



Package Diagram

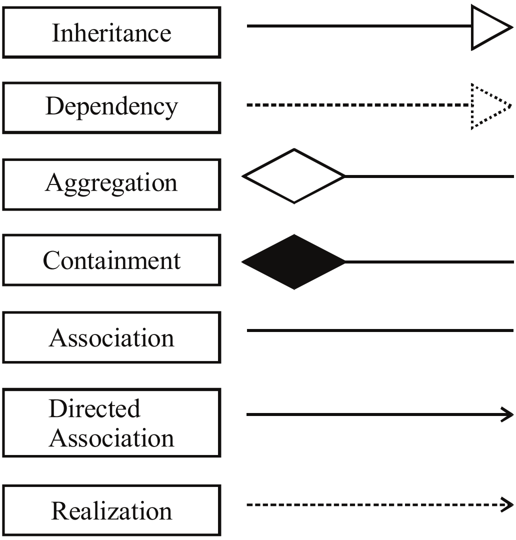
A **collaboration** is a society of classes, interfaces and other elements that work together to provide some cooperative behavior that is bigger than the sum of its parts.

A **relationship** is a connection among things. In object models, the common types of relationships are inheritance, dependency, aggregation, containment, association, realisation, and generalisation. The notation for relationship between use cases is shown as:



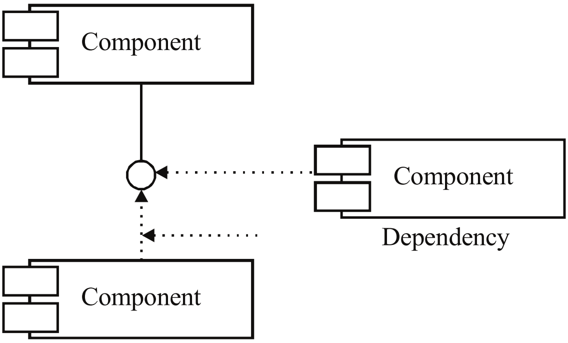
Relationship between different use case

Relationships exists in many forms. The notation for different form of relationship is shown as:

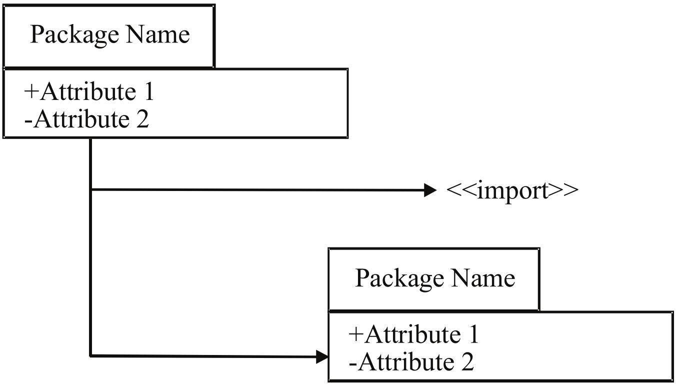


Common relationship types

A **dependency** is a relationship that states, that a change in specification of one thing may affect another thing, but not necessarily the reverse. In UML, it is shown as a dashed directed line. The notation for dependency between components and packages is shown as:



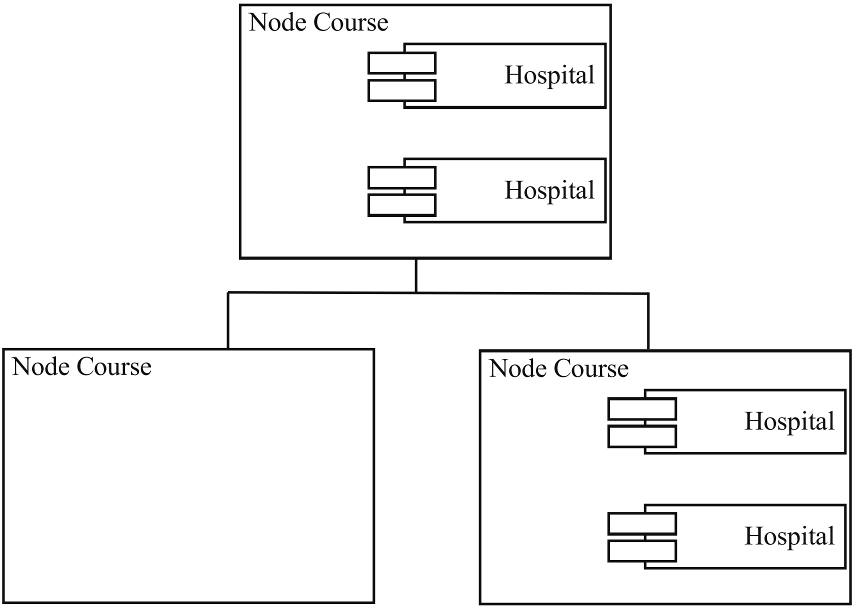
Dependency between components



Dependency between packages

A **generalization** is a relationship between a general thing and a specific kind of thing. It is also called "is-a-kind-of" relationship. Inheritance may be modeled using generalization. In UML, it is shown as a solid directed line with a large open arrow pointing to the parents.

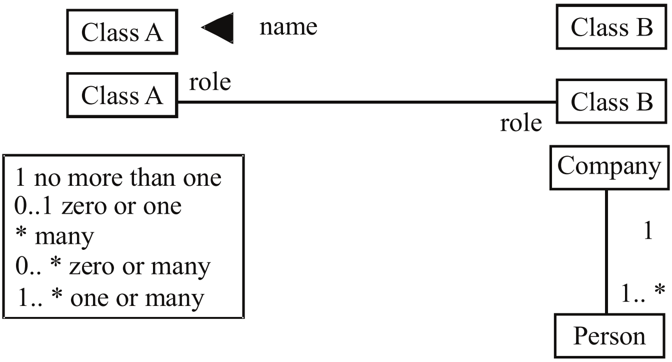
An **association** is a structural relationship that specifies that the objects of one thing are connected with the objects of another. In UML, it is shown as a solid line connecting same or different class. The notation for association between nodes is shown as:



Association between nodes

The four enhancements that apply to association are name, role, multiplicity, and aggregation. Each class participating in an association has a specific role which is specified at the rear end of the association.

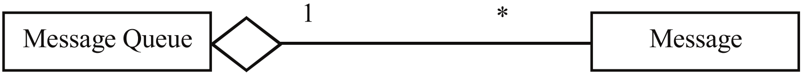
Multiplicity specifies how many objects may be concerned across an instance of an association which is written as a range of values (like 1..*). The notation for roles and multiplicity between classes is shown as:



1 no more than one
0..1 zero or one
* many
0.. * zero or many
1.. * one or many

Various roles and multiplicity defined with association

An **aggregation** is a structural relationship that specifies that one class represents a large thing which constitute of smaller things. It represents "has-a" relationship. In UML, it is shown as association with an open diamond at the large end. The notation for aggregation is shown as:



Aggregation

A **state** encompasses all the properties of the object along with the values of each of these properties.

An **instance** is a concrete manifestation of an abstraction to which a set of operations can be applied and which has a state that stores the effect of the operation.

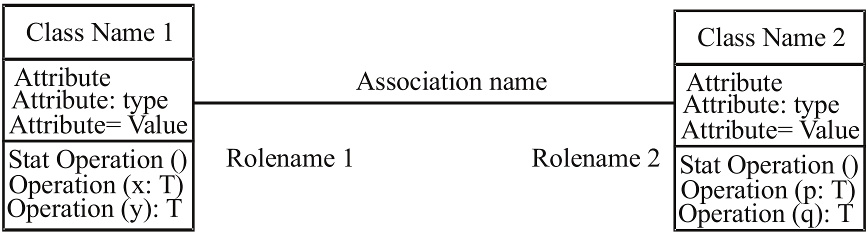
A **transition** is a relationship between two states indicating that an object in the first state will perform certain action and enter the second state when a specific event occurs and specific conditions are satisfied.

A **note** is a graphical symbol for rendering constraints or comments attached to an element or collection of elements.

Q11. What is a class diagram? Explain by drawing a class diagram for a class room scheduling system.

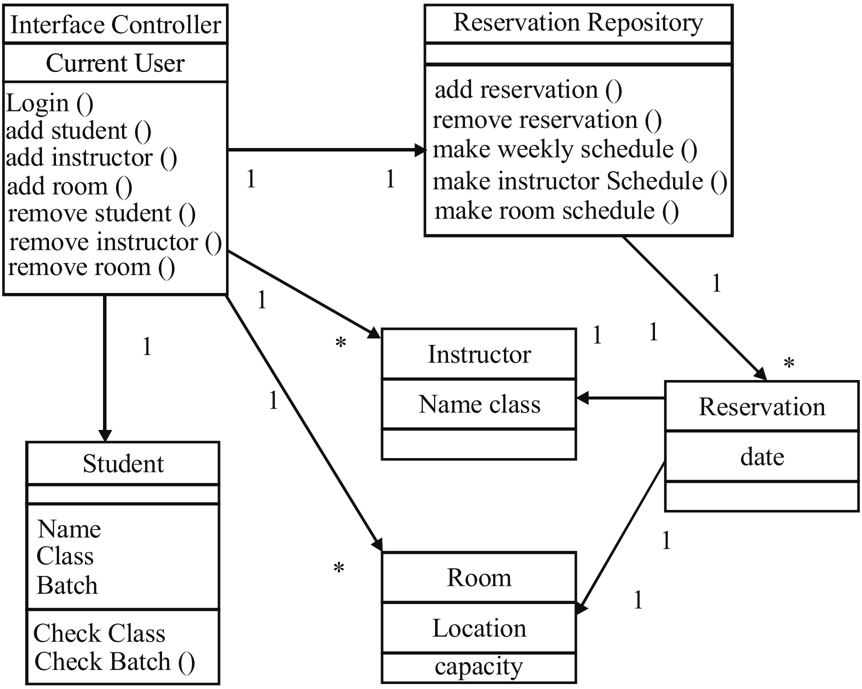
Ans. Class Diagram

A class diagram is used to support functional requirement of system. In a static design view, the class diagram is used to model the vocabulary of the system, simple collaboration, and logical schema. It contains sets of classes, interfaces, collaborations, dependency, generalization and association relationship. The notation for classes and the relationship between classes is shown as:



Relationship among classes

If in any college, there are limited classrooms that have to be allocated to different classes and instructors are fixed for all classes, then the class diagram for the allocation of classrooms and instructors is shown as:



Class diagram for a class room scheduling system

Q12. Explain different UML Behavioural Diagrams.

Ans. The UML stands for the Unified Modeling Language used to express the construct and the relationships of complex system. It was created in response to request for proposal (RFP) from the Object Management Group (OMG). UML is an attempt to standardize the aircrafts of analysis and design consisting of semantic models syntactic notations and diagrams.

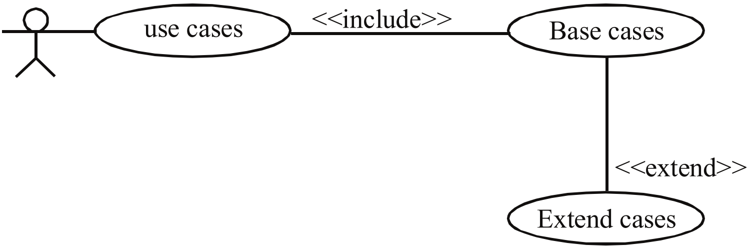
The major features of UML are:

- Defined system structure for object modeling.
- Support for different model organization.
- Strong modeling for structures and behaviour.
- Clear representation of concurrent activities.
- Support for object oriented patterns for design reuse.

UML could be used in visualizing, specifying constructing and documenting object oriented system. UML is build up by structural, behavioral, grouping and annotational notations. Behavioral Notations include dynamic elements of a model. Their elements comprise interaction and state machine and also classes, collaborations and objects.

Behavioral Diagrams : It is used for visualize, specify, construct and document dynamically a system. The interaction between objects indicate the flow of control among them is show by using these diagrams. The flow of control may includes simple, sequential thread through a system, as well as complex flows may involves branching, looking, recursion and concurrency. They could model time ordering or sequences of messages as correspondingly sequence diagram and collaboration diagram. The four main behavioral diagrams are: use case interaction, activity and statechart diagrams.

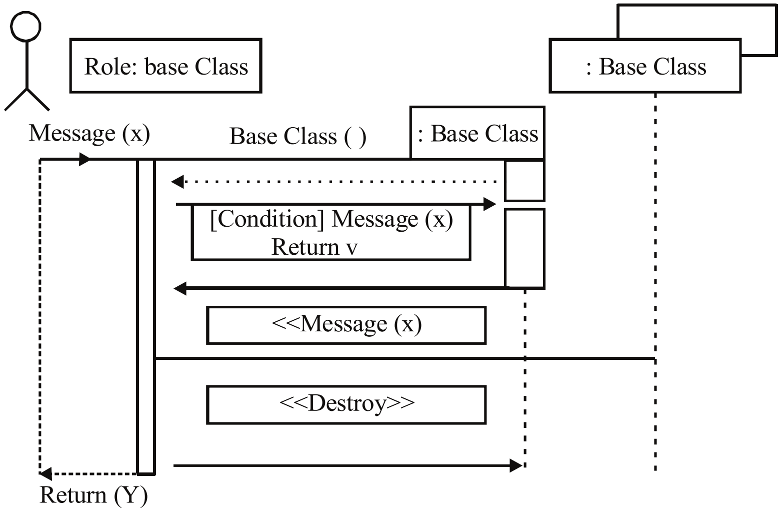
1. Use Case Diagram : It shows a set of use cases, actors and their relationships according to the context or requirement of system. It contains use cases, actors dependency, generalizations, association, relationships, roles, constraints, packages and instances. Such types of diagrams makes systems, a subsystems and classes approachable by presenting outside view of how the elements may be used in context.



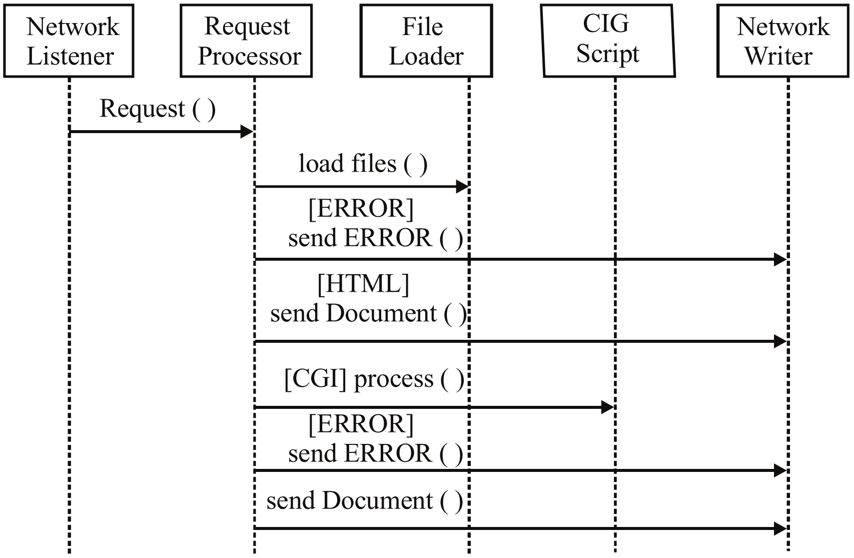
Relationship among use cases.

2. Interaction Diagram: It shows an interaction, consisting a set of objects and relationship, including the messages may be dispatched among them. It includes sequence diagrams and collaboration diagrams.

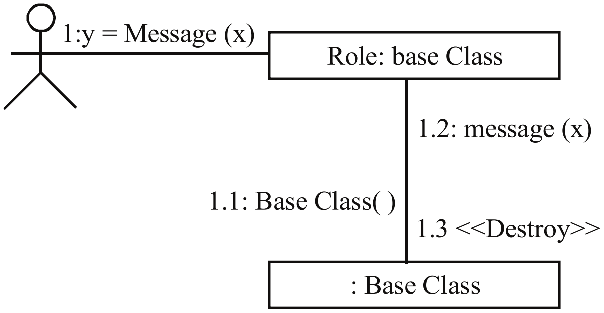
Sequence Diagrams : These are interaction diagrams emphasizes the time ordering of message. It is shown as a table that shows objects along x-axis and messages in increasing time, along the y-axis. It has global life line and the focus of control.



The sequence diagram for sending the document along the matter is shown as:

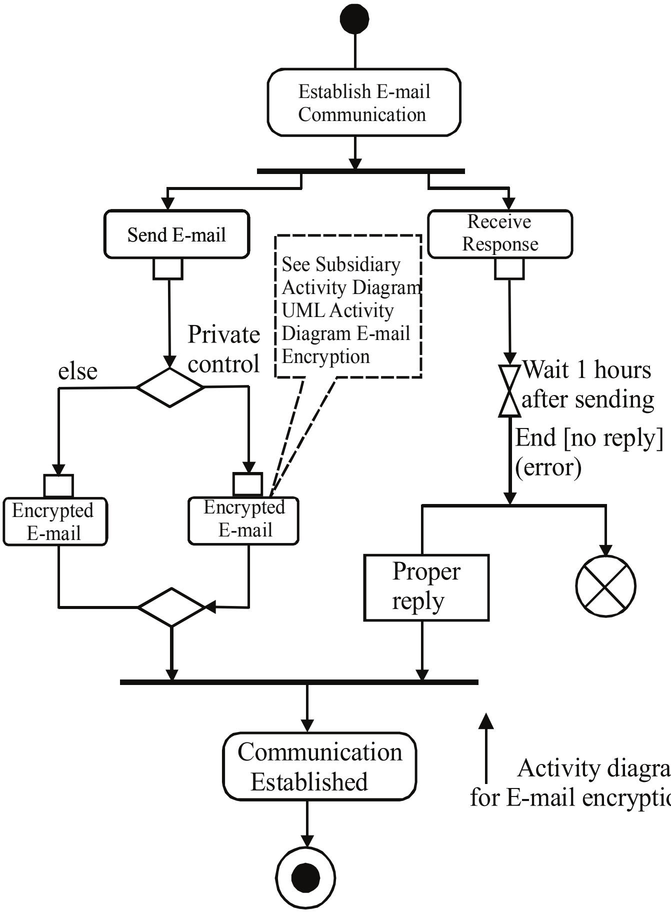


Collaboration Diagrams: These are the interaction diagrams that Emphasize the structural organization of an object that send & receive messages. There is a path in collaboration diagrams to indicate how one object is linked to another and sequence numbers to indicate time ordering of messages.



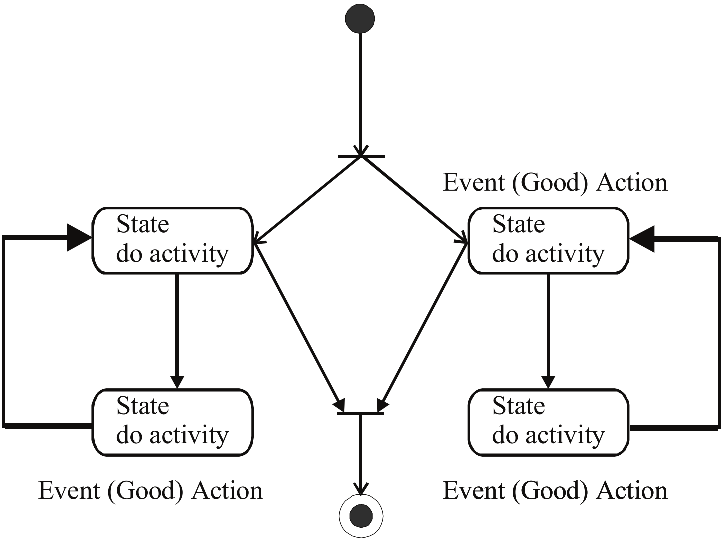
Collaboration Diagrams

Activity Diagrams : It shows the flow from one activity to another. An activity is an ongoing nonatomic execution with in a state machine. It results in some action which results in a change in state of the system or return of a value. It contains activity states, action states, transition states and objects.


Establish E-mail Communication
Send E-mail
Receive Response
See Subsidiary Activity Diagram UML Activity Diagram E-mail Encryption
Private control
else
Wait 1 hours after sending
End [no reply] (error)
Encrypted E-mail
Encrypted E-mail
Proper reply
Communication Established

Activity diagram for E-mail encryption

Statechart Diagram : It shows a state machine, Emphasizing the flow of control from one state to another. A state machine is a behaviour that specifies the sequence of states that an object goes through during life time in response to events together with its response to those events. A state is a situation or condition. It contains simple states, composite states, events, actions & transitions.



State diagram for multiple activities

Q13. Discuss the features of UML based design tools.

Ans. There are many tools used in the industry to develop information system. These tools provide the initial specification in UML that eventually leads to the database development. These tools provide support for conceptual, logical and physical database modeling design. Some of the features of these tools are :

- Support for Reverse Engineering : These tools allow the user to create a data model based on the already existing database structure.
- Forward Engineering and creation of database definitions : The tools can read the schema of the data models available in UML diagram - class diagram and create the database and the data storage model and generate appropriate DDL code for the same.
- Conceptual Design in UML notation : It allows modeling of database using UML notations.

Q14. Which of the UML diagrams help in database schema design the most and why?

Ans. The class diagram – it resembles ER diagram and also shows the possible transactions. Thus, it is useful for schema design as well as application design.

Q15. What are the various UML based design tools?

Ans. There are many open sources and proprietary UML based design tools available which can be easily searched through Internet.

Q16. What are the features of automated database design and implementation tools?

Ans. They should have facilities for:

- Drawing conceptual schema
- Mapping implementation
- Normalization
- Simple User interaction and displays
- Task analysis and
- Design verification.

Chapter-3

Important ER Diagrams

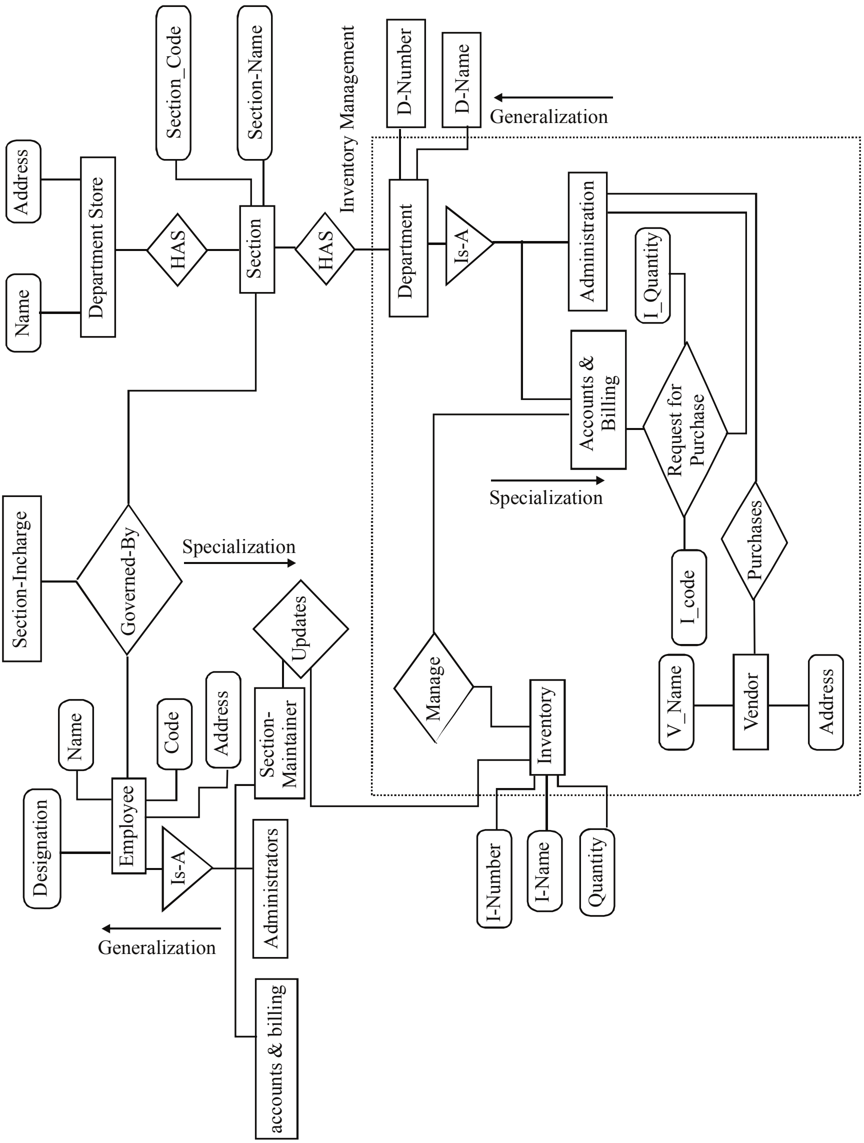
Q1. A departmental store consists of many item sections. A section is looked after by one section incharge. The store has three kinds of employees : accounts and billing; administrators; section maintainers. A request for purchase of items for the store is initiated by accounts and billing department which has the inventory details. Purchases are made by administration with proper quality checks from specified vendors. The section maintainers update the inventory placed at shelfs. Draw the E-R diagram for the store specifying aggregation, generalisation or specialisation hierarchy, if any. Translate your E-R diagram to relational schema and then design the tables in 3NF.

In E-R diagram, there are two Generalisations & specializations.

Ans. (1) Unionizing the three departments: Accounts & billing, Administration, and production into Department entity (Generalization) and Department entity can be specialized into accounts & billing, Administration & production (specialization).

(2) Unionizing Accountant, section incharge, Administrator into EMPLOYEE entity (Generalization) and EMPLOYEE entity is specialized into accountant, section incharge, administrator (Specialization).

In the following E-R diagram there is one Aggregation i.e. Inventory-management which is among Department, Vendor, Inventory, Purchased-From, PURCHASED-FROM, UPDATES, REQUEST-FOR-PURCHASE. An Inventory is maintained when, a request for purchase for the required item is made by accounts & billing department then the required items are purchased from vendors by administration department & then the inventory of the organization is updated by Section Maintainer.



The tables which are used in above E-R diagram :

(a) EMPLOYEE {Name, Code, Address, Designation}

(b) INVENTORY = {I-Code, I-Name, Quantity}

(c) VENDOR = {V-Name, Address, V-Code}

(d) DEPARTMENT = {D-Number, D-Name)

Here the attributes of tables are enclosed within curly braces and underlined attributes represent key to the relation.

The EMPLOYEE table is in 3NF because code is the prime attribute of EMPLOYEE and all other attributes are dependant on the code attribute i.e.

Code →| Name, Address, Designation

The INVENTORY table is in 3NF because I-Code is the prime attribute and all other attributes are dependant upon I-Code i.e.

I-Code →| I-Name Quantity

The DEPARTMENT table is also in 3NF because the D-Number is the prime attribute of DEPARTMENT and D-Name is dependant on D-Number i.e.

D-Number →| D-Name

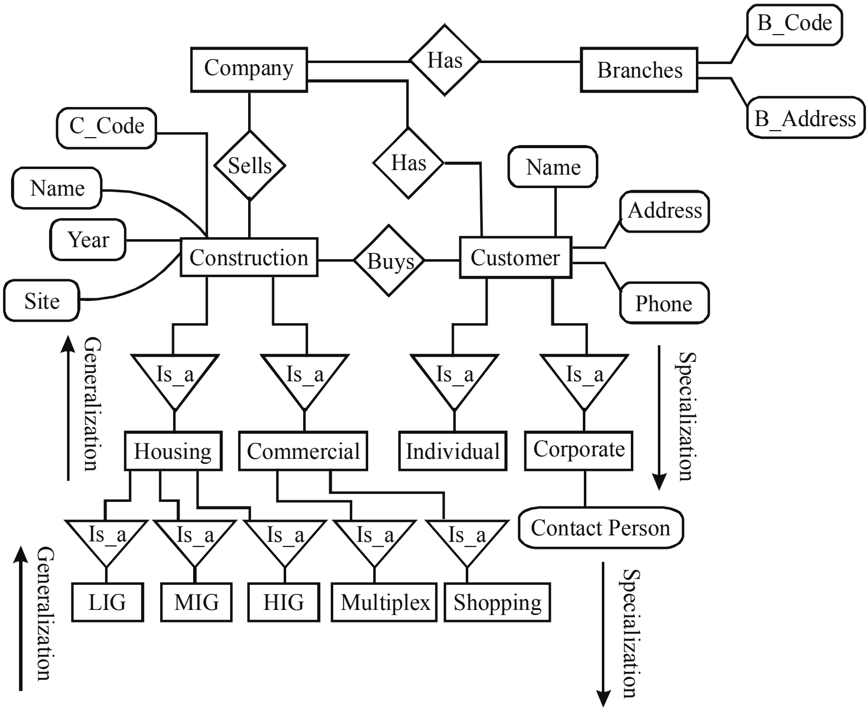
The VENDOR table is also in 3NF because V-Code is the prime attribute of VENDOR table and

V-Code →| V-Name Address

This implies all are in 3NF.

Q2. A construction company has many branches spread all over the country. The company has two types of constructions to offer: Housing and commercial. A housing company provides low income housing, medium style housing and high-end housing schemes, while on the commercial side it offers Multiplexes and shopping zones. The customers of the company may be individuals or corporate clients. Draw the E-R diagram for the company showing generalization or specialization hierarchies and aggregations, if any. Also create 3NF tables of your design.

Ans.



Generalization / specialization are ⇒ Unionizing Housing and commercial into CONSTRUCTION entity. Unionizing LIG, MIG and HIG entities into HOUSING Entity (Generalization). Unionizing Individual and Corporate into CUSTOMER entity. Unionizing Multiplex and Shopping into COMMERCIAL. Reverse with Specialization i.e. CONSTRUCTION is specialized into Housing and Commercial. CUSTOMER is specialized into Individual and Corporate. HOUSING is specialized into LIG, MIG and HIG. COMMERCIAL is specialized into Multiplex and Shopping.

If a table is in 3NF, we Leave it as it is, otherwise we break the table into more tables to bring it into 3NF.

Company (name, address, phone, contact person)
Construction (C_Code, year, site, project, name, area, type)
Specification (type, from, squmtr area, cost)
Branch (B_Code, B_Address)

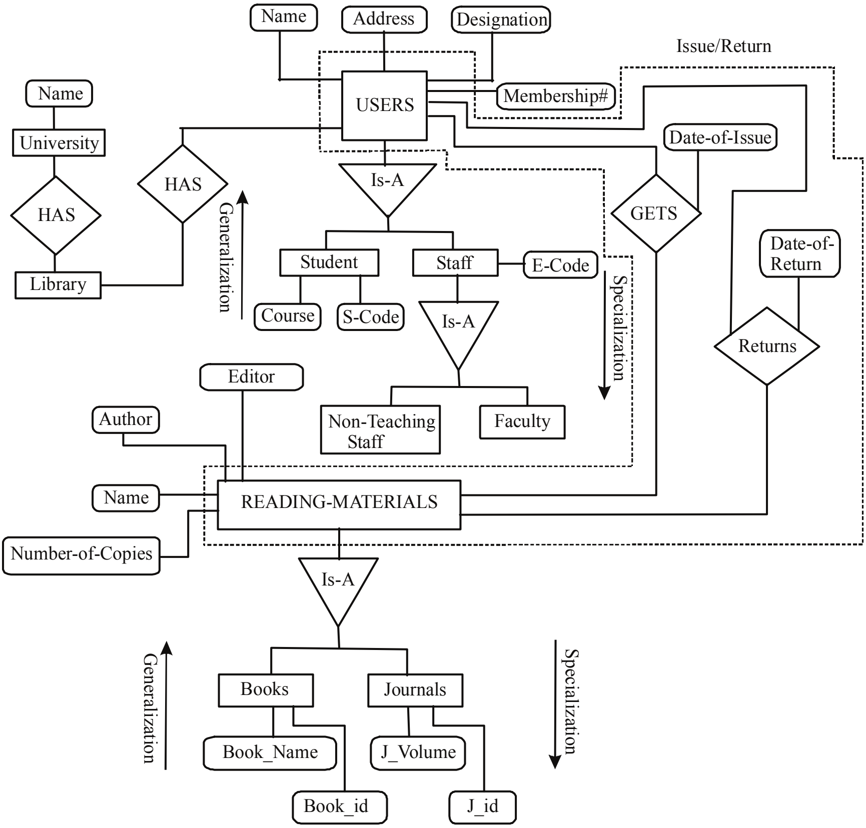
As all the tables are in 3 NF so no need to further breaking up.

Q3. The library is a university provides reading materials like books, journals, etc. to its users. The faculty, staff & students are its user list. Any user of the library has a membership card for using the library. The card is bar coded which helps in issue and return of reading materials. There is a maximum limit of the material to be issued. If issued material is not returned within 30 days from the data of issue, the user should be fined Rs. 1 per day till the day he/she returns the material. Draw the E-R Diagram for the library specifying aggregation, generalization, or specialization hierarchy, if any. Create 3NF tables of your design. Make suitable assumptions if any.

Ans. Assumption : We have taken Non-teaching staff and faculty into EMPLOYEE entity.

The first Generalization / specialization is ⇒ Unionizing Non-teaching staff and faculty into EMPLOYEE entity. Then unionizing EMPLOYEE & STUDENT entity into USER Entity (Generalization) USER entity can be specialized into STUDENT & EMPLOYEE & an EMPLOYEE can be a Non-Teaching staff or FACULTY (Specialization).

The second generalization/specialization is unionzing books and JOURNALS into READING MATERIALS entity (Generalizing). A READING MATERIAL can be specialized into Books & JOURNAL (Specialization). One point which is to be noted here is that generalization and speicalization are complementary to each other. Where there is generalization, there is specialization.



In above E-R diagram, there is one Aggregations, i.e.ISSUE/RETURN MATERIALS, GETS & RETURN. An Aggregation. ISSUE/RETURN exists whenever an USER GETS or RETURNS a READING-MATERIAL from and to LIBRARY.

If a table is in 3NF, we Leave it as it is, otherwise we break the table into more tables to bring it into 3NF.

(a) STUDENT = {Name, S-Code, Address, Designation, Course-Code, Membership#}
(b) EMPLOYEE = {Name, E-Code, Address, Designation, Membership#}
(c) READING-MATERIALS = {Name, Author, Edition, Number-of-copies#}
(d) UNIVERSITY = {Name, Address}
(e) BOOK = {Book_code, Issue_Date, Return_date, S_Code, Fine}

Note : in case of Employee taking books, S-code will act as E-Code

We check one-by-one whether tables used in E-R diagram are in 3NF.
The STUDENT table is in 3NF because all attributes of STUDENT table are dependant on the prime attribute s-code i.e.

S-Code → Name Address Designation course-code Membership#

The EMPLOYEE table is in 3NF because all non-prime attributes of this table are dependant on the prime attribute E-Code.

E-Code → Name Address Designation membership#

The READING - MATERIALS Table is also in 3NF because all non-prime attributes of this table are dependant on prime attribute i.e. Name Author i.e.

Name Author → EDITION, Number-of-copies.

This implies all are in 3NF.

Q4. A bank offers 3 types of accounts: personal account, saving account and debit-card account. It operates a number of branches in a city and one can have any number of accounts. Identify the entities and show their attributes. What relationship exists among entities? Draw the corresponding ER diagram.

Ans. ER Diagram for Bank:

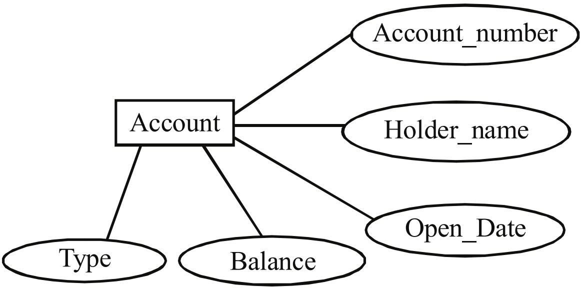
➢ A bank offers 3 types of Accounts:

Personal Account, Saving Account, Debit Card Account

∴ Create entity "Account"

Each Account has unique Account_number

So, Account is a strong entity.

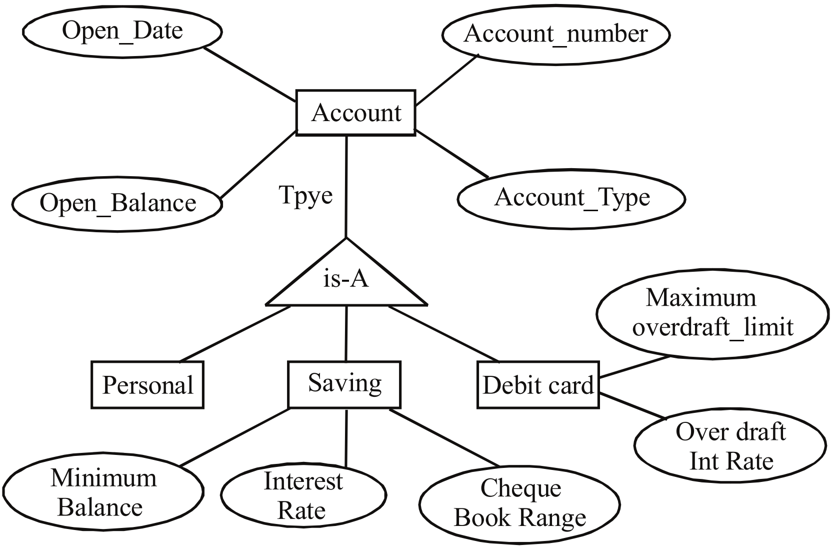


Each Account maintains following information

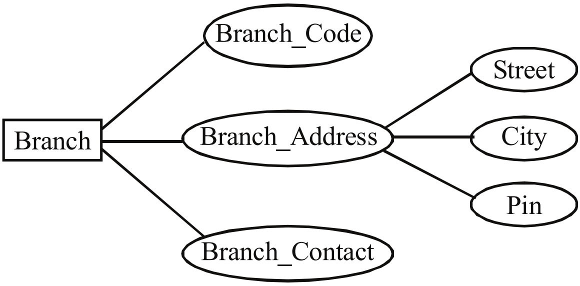
Account_number, Account Holder Name, Open Date, Balance

These three types of Accounts store distinct information as per its

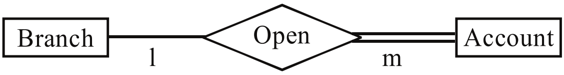
characteristics.
For e.g., Saving Account provides cheque facility, Interest at fixed Rate, Saving Account has maintained minimum balance as limit for amount withdrawal.
While Debit_Card Account allow withdrawal beyond balance but charge or debit interest on over_draft amount.
So, create separate Entity for each this account.



➢ Bank operates a number of branches in a city
It also keeps track of branch details.
It maintains branch information like branch_code, branch_address, branch_contact, branch_city
Branch_code is unique for each branch
So, branch is a strong entity.



Which Account opened in which Branch that Account_Branch relationship is shown as below:



A Branch can have many Accounts, while an Account is opened in one Branch only.

• Double line indicates Each Account Entity must have been opened in branch. (Total participation of "Account" in "Account_Open" Relationship

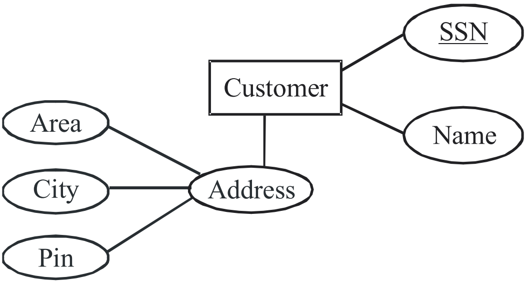
➢ System also keep track of Customer information

So, create Entity for customer

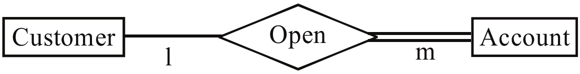
• It maintains SSM, Name, Address for each customer entity.

• SSN is unique for each customer

So, customer is strong entity



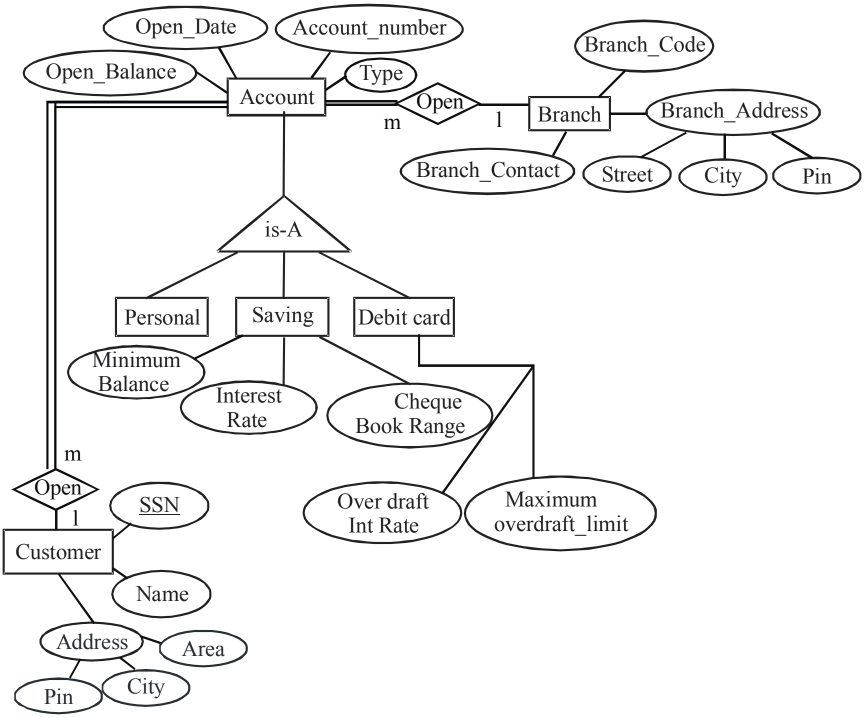
• A customer can open many Accounts, but an account can be opened by one customer only.



• 1-to-many relationship between Customer and Account Relation

• Each Account must participate in "Open" relationship.

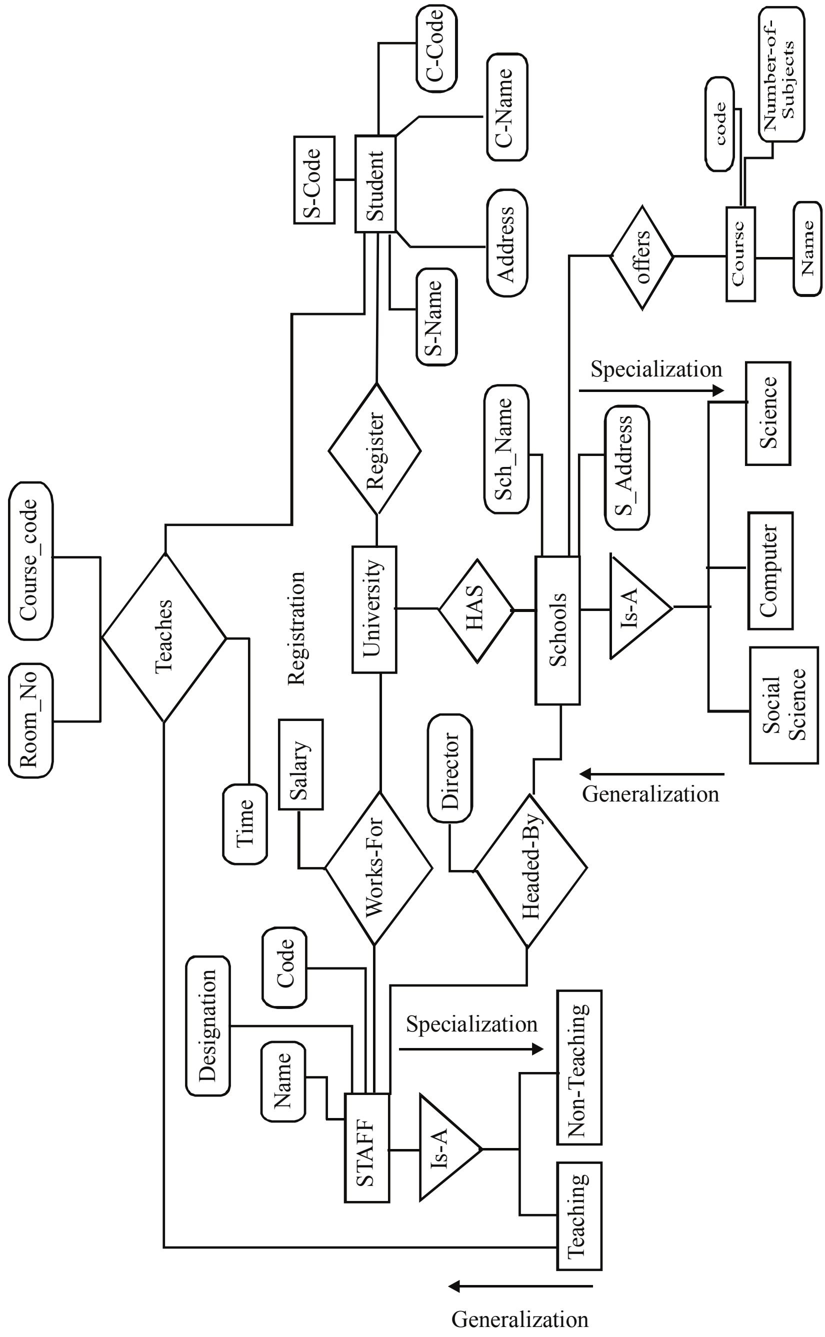
So, Total participation of "Account" in "Open" Relationship



Q5. A university has many Academic Units named schools. Each school is headed by a Director of School. The school has teaching and non-teaching staff. A school offers many courses. A course consists of many subjects. A subject is taught to the students who have registered for that subject in a class by a teacher. Draw the ER diagram for the university specifying aggregation, generalisation or specialisation hierarchy, if any. Create 3NF tables of your design. Make suitable assumptions, if any.

Ans.

E-R Diagram of University as given below:-


C-Code
C-Name
S-Code
Student
Address
S-Name
Register
Course_code
Room_No
Teaches
Registration
University
HAS
Schools
Sch_Name
S_Address
offers
Course
code
Number-of-Subjects
Name
Specialization
Science
Computer
Social Science
Is-A
Generalization
Time
Salary
Works-For
Director
Headed-By
Code
Designation
Name
STAFF
Is-A
Specialization
Non-Teaching
Teaching
Generalization

In above E-R diagram, we have shown Generalization, Specialization & Aggregation. Generalization & Specialization are containment relationship between a higher level entity set and one-or-more lower level entity sets.

Generalization is the result of taking the union of two-or-more disjoint entity sets to produce a higher level entity set.

Specialization is the result of taking a subset of a higher level entity set to form a lower level entity set.

Here in above E-R diagram, there are three Generalizations & Specializations.

The first Generalization & Specialization is:

Unionizing Teaching & Non-Teaching into Staff entity (Generalization) & Specializing Staff into Teaching & Non-Teaching.

The second Generalization & Specialization is ⇒ Unionizing Social Science, Computer and Science into school entity (Generalization) and specializing school entity into Social Science, Computer and Science Specialization.

Aggregation is the abstraction in which relationship sets (along with their entity sets) are treated as higher level entity sets, and can participate in relationships. In above E-R diagram there is one aggregation called Registration which is among UNIVERSITY, STUDENT & COURSE. An aggregation registration takes place whenever a student is registered by the university in a particular course.

The tables used in above E-R diagram:

(a) Staff = {Code, Name, Designation}

(b) School = {Sch-Name, Address}

(c) Student = {S-Name, S-Code, C-Code, C-Name, Address}

(d) Course = {C-Name, C-Code, No-of Subjects}

The staff table is in 3NF because the code is the attribute of Staff relation & all other attributes are dependent are dependant on this i.e.

Code Name, Designation Address

The school table is in 3NF because sch-Name is the PK of school relation and all other attributes are dependant on this

Sch-Name Address

The student table is not in 3NF because the S-Code is its PK and the attributes S-Name, Address and C-Code are dependant upon S-Code but C-name is dependant upon C-Code. This is transitive dependency.

S-Code S-N Address C-Code

but

C-Code C-Name

The course relation is in 3NF because C-Code is its PK and all other attributes are dependant on this i.e.

C-Code C-Name No. of Subject

The schedule relation is in 3NF because all attributes of this relation form a PK i.e. composite Primary key
This implies all relations are in 3NF.

Chapter-4

ADVANCED SQL AND SYSTEM CATALOGUE

Q1. What are Assertions? What is the utility of assertions? How are Assertions different from Views? Describe both Assertions and Views with the help of example. [June-07, Q1(c)]

Ans. Assertions are constraints that are normally of general nature. For example, the age of the student in a hypothetical University should not be more than 20 years of the minimum age of the teacher of that University should be 30 years. Such general constraints can be implemented with the help of an assertion statement. The syntax for creating assertion is:

Syntax :

```
CREATE ASSERTION <Name>

CHECK (<Condition>);
```

Thus, the assertion on age for the University as above can be implemented as:

```
CREATE ASSERTION age-constraint
CHECK (NOT EXISTS)(
SELECT *
FROM STUDENT s
WHERE s.age >25
OR s.age> (
SELECT MIN (f.age)
FROM FACULTY f
));
```

The assertion name helps in identifying the constraints specified by the assertion. These names can be used to modify or delete an assertion later. But how are these assertions enforced? The database management system is responsible for enforcing the assertion on to the database such that the constraints stated in assertion are not violated. Assertions are checked whenever a related relation changes.

An assertion for a university system that stores a database of faculty as :

FACULTY (code, name, age, basic salary, medical-allow, other benefits)

MEDICAL-CLAIM (code, data, amount, comment)

Assertion : The total medical claims made by a faulty member in the current financial year should not exceed his/her medical allowance.

```
CREATE ASSERTION med-claim
CHECK (NOT EXISTS (
        SELECT code, SUM (amount), MIN(medical-allow)
        FROM (FACULTY NATURAL JOIN MEDICAL-CLAIM)
        WHERE data >”31–01–2008”
        GROUP BY code
        HAVING MIN(medical-allow) < SUM(amount)
));
```

View Definition

Views are tables whose contents are taken or derived from other tables. View tables do not contain data. They just act as a window through which certain (not all) contents of a table can be displayed or seen. Also, one can manipulate the contents of the original table through such views.

For example, suppose we want that the clerk in your office should not have access to the details of the salaries of other employees, but he needs such information as EMP_NAME, DESIG and DEPT_NAME. Then we can create a view for this clerk. This view will contain only the required information and nothing more than that.

Similarly we can create a view for the Manager, containing the information that he needs. Hence, depending upon the requirement of different users we can create different views, all based upon one or more than one tables. View are created using the CREATE command. The syntax is as follows:

```
CREATE OR REPLACE VIEW <VIEW NAME> AS <SELECT STATEMENT>;
```

For example, to create a view for a clerk need to see a part of the information, we can design a view giving the following statement:

```
CREATE OR REPLACE VIEW clerk as
SELECT EMP_NAME, DESIG, DEPT_NAME
FROM EMPLOYEE, DEPARTMENT
WHERE EMPLOYEE.DEPT_CODE
= DEPARTMENT.DEPT_CODE;
```

Once created, a view can be treated just as you treat any other table. You can retrieve the records in a view using the SELECT command, delete the records using the DELETE command and update them using the UPDATE command. Views can also be joined to other views and / or table.

To view the records in the clerk view, the clerk user can use the following SELECT statement:

```
SELECT * FROM clerk;
```

Uses of View

Views are used for the following purposes:

(a) Views restrict access to a database. If you want users to see some but not all data of a table, you can create a view showing only that part of the table.

(b) Critical data in a table can be easily safe guarded using views.

(c) Views allow user to make simple queries to retrieve the required results. For example, users can retrieve information from multiple tables without knowing how to perform a join.

(d) Views allow the same data to be seen by different users in different ways.

Q2. Can views be used for Data Manipulations?

Ans. Views can be used during DML operations like INSERT, DELETE and UPDATE. When you perform DML operations, such modifications need to be passed to the underlying base table. However, this is not allowed on all the views.

Q3. ''Views can be used in an attempt to implement logical data independence." Comment on the above statement. Give justification and example in support of your answer Where can you use views for data modifications?

Ans. Logical data independencies indicate that the conceptual scheme can be changed without affecting the existing internal and conceptual levels. Logical data independence is achieved by providing external level/user view of the database.
The relations stored in the database are termed as base relations. Corresponding to these relations we derive a new relation. They are important because of
1. Security Concerns.
2. Forming more than one base Table.

In short a view in SQL terminology is a single table that is derived from other tables. These other tables could be base tables or previously defined views. A view does not necessarily exist in physical form, it is considered as a virtual table.

The application program or users see the database as described by their external views. The user is provided with the abstraction of physical data and user manipulates this abstraction. For example: We have two relations Employee personal and employee_salary. Now for an external user we can create view containing attributes from both the relations.
This view does not exist physically.
employee-personal (emp#, name, address)
employee-salary (emp#, post, salary)
we can define view:
employee-information (emp#, name, post, salary)
Example :- Suppose we have 3 relations as:-

EMPLOYEE:

FNAME	Minit	LName	SSN	BDatte	Address	Sex	Salary	Super SSN	DNo.

EMPLOYEE:

PNAME	PNUMBER	PLOCATION	DNUM

WORKS-ON

ESSN	PNO	HOURS

we have created a view WORKS-ON 1 on the basis on above 3 Relations The WORKS-ON 1 will look as:

WORKS-ON 1

FNAME	LNAME	PNAME	HOURS

The view implies logical data independence in the sense that if fields are added or deleted from the base relations and if there field name have been put into the view then the view need not be altered. The base relations will be altered & views will remain the same. This provides the high availability of data to the users. Here conceptual schema can be changed without changing the external schemas.
Thus, we can say that view implements logical data independence.

View Updation ⇒ Many commercial DBMSs disallow updates through a view unless the view is based on a single relation & includes the primary attributes of the relation.

The conditions under which most DBMSs determine whether an update is allowed through a view.

- Updates are allowed through a view defined using a simple query involving a single base relation & containing either the primary key of a candidate key of the base relation.
- Updates are not allowed through views involving multiple relations.
- Updates are not allowed through views involving aggregation or grouping operations.

Q4. Consider a constraint – the value of the age field of the student of a formal University should be between 17 years and 50 years. Would you like to write an assertion for this statement?

Ans. The constraint is a simple Range constraint and can easily be implemented with the help of declarative constraint statement. (CHECK CONSTRAINT statement can be used here). Therefore, there is no need to write an assertion for this constraint.

Q5. Can the view created in previous question be used to update subjectcode?

Ans. No, the view is using a GROUP BY clause. Thus, if we try to update the subjectcode. We cannot trace back a single tuple where such a change needs to take place.

Q6. Discuss the two strategies for implementing the views.

Ans. There are two strategies for implementing the views. These are :
- Query modification
- View materialisation

In the query modification strategy, any query that is made on the view is modified to include the view defining experssion. For example, consider the view STUDENT-PERFORMANCE. A query on this view may be : The teacher of the course MCS-013 wants to find the maximum and average marks in the course. The query for this in SQL will be :

```
SELECT MAX (smakrs), AVG(smarks)
FROM SUBJECT-PERFORMANCE
```

Since SUBJECT-PERFORMANCE is itself a view the query will be modified automatically as :

```
SELECT MAX (smarks), AVG (smarks)
FROM STUDENT s, MARKS m
WHERE s.enrolement-no=m.enrolement-no AND subjectcode="MCS-013";
```

However, this approach has a major disadvantage. For a large database system, if complex queries have to be repeatedly executed on a view, the query modification will have to be done each time, leading to inefficient utilisation of resources such as time and space.

The view materialisation strategy solves this problem by creating a temporary physical table for a view, thus, materialising it. However, this strategy is not useful in situations where many database updates are made on the tables, which are used for view creation, as it will require suitable updating of a temporary table each time the base table is updated.

Q7. What are the weaknesses of SQL? Can a host language help SQL in overcoming these weaknesses? Justify your answer with the help of examples.

Ans. SQL is both the data definition & data manipulation language of a number of relational database Management systems. It is based on tuple calculus. It resembles relational algebra in some places and tuple calculus in others

Weaknesses of SQL

1. SQL does not fully support some of the basic features of relational data model : The concept of domains, entity & referential integrity and hence the concept of primary key and foreign keys.

2. SQL is redundant in the sense that the same query may be expressed in more than one way.

3. Tests with number of implementations of SQL indicate a wide variation in response to time.
4. SQL is less orthogonal and powerful.

It is true that a host language can help SQL in overcoming its weaknesses. SQL only provides facilities to define and retrieve data interactively. To extend data manipulation operations of SQL. Example to separately process each tuple of a relation - it should be used with traditional high level language, which is called host language.

The use of database system in applications written in an HLL (High Level Language) requires that DML Statements be embedded in the host programs. All the statements and features that are available to an interactive user must be available to the application programmer using HLL. The DML statements are distinguished by means of a special symbol or are invoked by means of a subroutine call.

Example : The program prints some information about an employee, who earns highest salary.

```
#include<stdio.h>
#include<stdio.h>
VARCHAR user name [30];
VARCHAR pass word [10];
VARCHAR v_fname [15];
VARCHAR v_minit [1];
VARCHAR v_lname [15];
VARCHAR v_address [30];
char v_ssn [9];
float f_salary;
main ()
{
Strcpy (user name.arr, “Scott”);
Username.len=strlen (Username.arr);
Strcpy (password,arr,”TIGER”);
Password.len=Strlen (Password.arr);
EXEC SQL WHENEVER SQLERROR DO SQL_error();
EXEC SQL CONNECT : Username IDENTIFIED By : password;
EXEC SQL SELECT fname,minit, lname,address, salary
```

```
    INTO: v_fname, : v_minit,: v_lname,:v_address,:f_salary
    FROM EMPLOYEE
WHERE Salary = (Select max (salary) from employee);
Printf("Employee first name, middle initial, last name, address, salary\n");
Printf ("%s %s %s %s%f\n", v_fname.arr, v_minit,arr, v_name.arr,
v_address,arr, f_salary);
}
SQL-error ()
{
EXEC SQL WHENEVER SQL ERROR CONTINUE;
Print )"Error detected\n");
}
```

Q8. Why is the embedded SQL used? Describe an application that needs embedded language. How are static embedded SQL different from dynamic embedded SQL?

Ans. When an SQL statement is embedded in a source program then it is called embedded SQL. The DBMS precompiler accepts this source code as input and translates the SQL compiled. Basically, it is based on cursor concept that can range over the query result.

When a job having complex calculations is to be done at different time, then it is very easy to code the entire logic in a programming language & get the data through the embedded SQL. Now put the program in library function call from where it can be called as and when required.

There are two types of embedded SQL

(i) Static Embedded SQL :- This type of embedded SQL is added into the program at the time of compilation. It is fixed. It cannot be changed.

(ii) Dynamic Embedded SQL :- When an SQL statement is constructed during program execution then it is called dynamic embedded SQL. Generally it happens in case of accepting a user response.

Example :- Airline Reservation system requires such type of solution.

Statically Embedded DML statements are distinguished by means of special symbols. These statements are partially passed during the precompilation step to look for statements and variables from the host HLL. In statically embedded DML statements variables are optimized variables from the host HLL. In statically embedded DML statements variables are optimized in the precompilation step, making them domain compatible with the HLL. Whereas in Dynamically embedded DML statements are not precompiled nor optimized. We can use statically embedded DML where we can ascertain the data type and range of HLL variable before execution. In this situation we will not get data type mismatch errors. Some times we can not ascertain the data type and range of variables produced by the HLL at run time besides we have to use these variables in DML. In this situation we use dynamically embedded statements for example. Let we have VB environment. We know all the variables at design time for example these are item_code, quantity.

Example:1

```
Option explicit
Dim item_code, quantity
Item_code = text1.text
Quantity = Limit (text2. text)
rs. address ‘rs is opened recordset object.
rs. fields (“item_code”) = item_code
rs. fields (“quantity”) = quantity
rs. update.
```

Application of embedded SQL

- Converting set level operation into field level operations.
- Preparing general purpose interactive program, authorization control, access control, grant of privilege, rights during runtime and so on.

Q9. Relational database consists of the following relations about students in an institute :

STUDENT(ROLL NO#, STUDENT_NAME, ADDRESS, AGE)
DEPARTMENT (DEPT_NAME, COURSE#TEACHER)
TIMETABLE (COURSE#, SUBJECT_NAME, SEMESTER, TIME, ROOM#)
PERFORMANCE(ROLLNO#, COURSE#, GRADE)

Write a program in embedded SQL to get the names of all those students who have secured ‘A’ grade in a course offered by the Computer Science Department in the First semester.

EXEC SRL

Ans.

```
Declare c cursor for
Select student - name from student S,
Department D, Time Table T, performance P
Where D. course # = T. course # and P. course # =
            T.course # and S. Roll no. # = P. Roll no. # and
            Dept - name = ‘Computer Science’.
            Begin
                    Open c;
                    While (SQLCA . SQL stal ! = ‘02000’)
                    Loop
                            fetch, c into : Sn;
                            Print :Sn;
                    End loop ;
                    Close c ;
            End ;
    END-EXEC
```

Q10. Define Dynamic SQL. Is dynamic SQL is more powerful than embedded SQL? Explain with support of an example.

Ans. Dynamic SQL, unlike embedded SQL statements, are built at the run time and placed in a string in a host variable. The created SQL statements are then sent to the DBMS for processing. Dynamic SQL is generally slower than statically embedded SQL as they require complete processing including access plan generation during the run time.

However, they are more powerful than *embedded SQL* as they allow run time application logic. The basic advantage of using dynamic embedded SQL is that we need not compile and test a new program for a new query.

Let us explain the use of dynamic SQL with the help of an example:

Example : Write a dynamic SQL interface that allows a student to get and modify permissible details about him/her. The student may ask for subset of information also. Assume that the student database has the following relations.

STUDENT (enrolno, name, dob)

RESULT (enrolno, coursecode, marks)

In the table above, a student has access rights for accessing information on his/her enrolment number, but s/he cannot update the data. Assume that user names are enrolment number.

```
/*declarations in SQL*/
EXEC SQL BEGIN DECLARE SECTION;
        char inputfields (50);
        char tablename(10)
        char sqlquery ystring(200)
EXEC SQL END DECLARE SECTION;
        printf ("Enter the fields you want to see /n");
        scanf ("SELECT%s", inputfields);
        printf ("Enter the name of table STUDENT or RESULT");
        scanf ("FROM%s", tablename);
        sqlqueryystring ="SELECT" + inputfields + " "+
          "FROM" + tablename
          +"WHERE enrolno + :USER"
/* Plus is used as a symbol for concatenation operator; in some DBMS it may
be ||*/ /* Assumption: the user name is available in the host language variable
USER*/
EXEC SQL PREPARE sqlcommand FROM :sqlqueryystring;
EXEC SQL EXECUTE sqlcommand;
```

Please note the following points in the example above.

- The query can be entered completely as a string by the user or s/he can be suitably prompted.
- The query can be fabricated using a concatenation of strings. This is language dependent in the example and is not a portable feature in the present query.
- The query modification of the query is being done keeping security in mind.
- The query is perpared and executed using a suitable SQL EXEC commands.

Q11. What is SQLJ? What are the requirements of SQLJ? Briefly describe the working of SQLJ. Can SQLJ use dynamic SQL? If yes, then how? Otherwise specify what type of SQL it can use.

[June-07, Q4(c)]

Ans. In SQLJ, a preprocessor called SQLJ translator translates SQLJ source file to JAVA source file. The JAVA file compiled and run on the database. Use of SQL improves the productivity and manageability of JAVA Code as:

- The code becomes somewhat compact.
- No run-time SQL syntax errors as SQL statements are checked at compile time.
- It allows sharing of JAVA variable with SQL statements such sharing is not possible otherwise.

SQLJ cannot use dynamic SQL. It can only use simple embedded SQL. SQLJ provides a standard form in which SQL statements can be embedded in JAVA program. SQLJ statements always begin with a #sql keyword. These embedded SQL statements are of two categories– Declarations and Executable Statements.

Declarations have the following syntax:

```
            #sql <modifier> context_classname;
```

The executable statements have the following syntax:

```
            #sq; {SQL operation returning no output};
                        OR
#sql result ={SQL operation returning output};
```

Example:

Let us write a JAVA functions to print the student details of student table, for the student who have taken admission in 2008 and name are like ‘Shyam’. Assuming that the first two digits of the 9-digit enrolment number represents year, the required input conditions may be:

- The enrolno should be more than “08000000” and
- The name contains the sub string “Shyam”.

These input conditions will not be part of the Student display function, rather will be used in the main ( ) function that may be created separately by you. The following display function will accept the values as the parameters supplied by the main ( ):

```
Public void DISPSTUDENT (String enrolno, String name, int phone)
{
try {
if (name equals (“  “) )
name = “%”
if (enrolno equals (“ ”))
enrolno = “%”;
SelRowlter srows = null;
#sql srows = {SELECT Enrolno, name, phone
FROM STUDENT
WHERE enrolno > : enrolno AND name like :name
};
while (srows.next ( ) ) {
```

```
int enrolno = srows. enrolno ( );
String name = srows.name ( );

System.out.println (“Enrollment_No =” + enrolno);
System.out.println (“Name =”+name);
System.out.println (“phone=” +phone);
}
}Catch (Exception e) {
System.out.println (“error accessing database” +e.to_string);
}
}
```

Q12. Q1. A University decided to enhance the marks for the students by 2 in the subject MCS-013 in the table: RESULT (enrolno, coursecode,marks). Write a segment of embedded SQL program to do this processing.

Ans. /* The table is RESULT (enrolno.coursecode, marks). */

```
EXEC SQL BEGIN DECLARE SECTION;
char enrolno [10], coursecode[7]; /* grade is just one character */
int marks;
char SQLSTATE[6]
EXEC SQL END DECLARE SECTION;
/* The connection needs to be established with SQL */
/* program segment for the required function */
printf(“enter the course code for which 2 grace marks are to be added”)
scanf(“%s”, &coursecode);
EXEC SQL DECLARE CURSOR GRACE
SELECT enrolno, coursecode, marks
FROM RESULT
WHERE coursecode=:coursecode
FOR UPDATE OF marks;
EXEC SQL OPEN GRACE;
EXEC SQL FETCH FROM GRACE
INTO: enrolno, :coursecode, :marks;
WHILE(SQL CODE = = 0) {
EXEC SQL
UPDATE RESULT
SET marks = marks+2
WHERE CURRENT OF GRACE;
EXEC SQL FETCH FROM GRACE;
}
EXEC SQL CLOSE GRACE;
```

An alternative implementation in a commercial database management system may be:

```
DECLARE CURSOR grace IS
SELECT enrolno, coursecode, marks
FROM RESULT
WHERE coursecode = ‘MCS013’;
str_enrolno RESULT.enrolno%type;
str_coursecode RESULT.coursecode%type;
str_marks RESULT.marks%type;
BEGIN
OPEN grace;
IF GRACE %OPEN THEN
LOOP
FETCH grace INTO str_enrolno, str_coursecode, str_marks;
Exit when grace%NOTFOUND;
UPDATE student SET marks = str_marks+2;
INSERT INTO resultmcs-13 VALUES (str_enrolno, str_coursecode,
str_marks);
END LOOP;
COMMIT;
CLOSE grace;
ELSE
Dbms_output.put_line (‘Unable to open cursor’);
END IF;
END;
```

Q13. Can you embed dynamic SQL in JAVA?

Ans. No, at present JAVA cannot use dynamic SQL.

Q14. What is stored procedure?

Ans. Stored procedure is a compiled procedure in a host language that has been written for some purpose.

Q15. Consider the following relation of a student database:
Student (enrolno, name, age)
Result (enrolno, coursecode, marks)
Course (coursecode, c_name)
Passing marks in a subject are 50.
Write a trigger that will give 3 grace marks to the students having 47 marks, 2 grace marks to the students having 48 marks and 1 grace mark to the student having 49 marks. Another condition for triggering is that a student's age must exceed 30 to get grace marks.

Ans. Relations given are:
Student (enrolno, name, age)
Result (enrolno, coursecode, marks)
Course (coursecode, e_name)

Note:
Passing marks is 50
Trigger: It will give
3 grace marks to students having 47 marks.
2 grace marks to students having 48 marks.
1 grace mark to students having 49 marks.
If and only if age is above 30

```
CREATE TRIGGER GRACE_MARK
BEFORE INSERT ON Result
FOR EACH ROW
WHEN marks < 50
DECLARE
Stud_Age Student; student.age % type;
BEGIN
SELECT age into Stud_Age
FROM student
WHERE Student_enrolno = :NEW enrolno;
IF Stud_age > 30 Then
IF        : new.marks = 47 Then
          : new.marks = :new.marks + 3;
ELSE IF: new.marks = 48 Then
          : new.marks = :new.marks + 2;
ELSEIF : new.marks = 49 Then
          : new.marks = :new.marks + 1;
ENDIF
```

```
ENDIF
END;
```

Q16. Write a trigger that restricts updating of STUDENT table outside the normal working hours/holiday.

Ans. The trigger in some pseudo DBMS may be written as:

```
CREATE TRIGGER noupdatestudent
BEFORE INSERT OR UPDATE OR DELETE ON STUDENT
BEGIN
IF (DAY (SYSDATE) IN (‘SAT’, ‘SUN’)) OR
(HOURS (SYSDATE) NOT BETWEEN ’09:00’ AND 18:30’)
THEN
RAISE EXCEPTION AND OUTPUT (‘OPERATION NOT ALLOWED AT
THIS TIME/DAY’);
END IF;
END
```

Please note that we have used some hypothetical functions, syntax of which will be different in different RDBMS.

Q17. Discuss about the advanced features of SQL.

Ans. SQL Interfaces : SQL also has a good programming level interfaces. The SQL supports a library of functions for accessing a database. These functions are also called the Application Programming Interface (API) of SQL. The advantage of using an API is that it provides flexibility in accessing multiple databases in the same program irrespective of DBMS, while the disadvantage is that it requires more complex programming. The following are two common functions called interfaces:

SQL/CLI (SQL - call level interface) is an advanced form of Open Database Connectivity (ODBC).

Java Database Connectivity (JDBC) - allows object-oriented (JAVA to connect to the multiple databases.

SQL support for object-orientation : The latest SQL standard also supports the object-oriented features. It allows creation of abstract data types, nested relations, object identifies etc.

Interaction with Newer Technologies : SQL provides support for XML (eXtended Markup Language) and Online Analytical Processing (OLAP) for data warehousing technologies.

Q18. Define catalogue and data dictionary. How both are similar or different?

Ans. The *system catalogue* is a collection of tables and views that contain important information about a database. It is the place where a relational database management system stores schema metadata, such as information about tables and columns, and internal bookeeping information. A system catalogue is available for each database. Information in the system catalogue defines the structure of the database.

The information stored in a catalogue of an RDBMS includes :

- the relation names,
- attribute names,
- attribute domains (data types)
- descriptions of constraints (primary keys, secondary keys, foreign keys, NULL/NOT NULL, and other types of constraints),
- views, and
- storage structures and indexes (index name, attributes on which index is defined, type of index etc.)

Security and authorisation information is also kept in the catalogue, which describes:

- authorised user names and passwords,
- each user's privilege to access specific database relations and views,

the creator and owner of each relation. The privileges are granted using GRANT command.

The system catalogue can also be used to store some statistical and descriptive information about relations. Some such information can be:

- number of tuples in each relation,
- the different attribute values,
- storage and access methods used in relation.

All such information finds its use in query processing.
In relational DMBSs the catalogue is stored as relations.

The terms system catalogue and data dictionary have been used interchangeably in most situations.
Data dictionaries also include data on the secondary keys, indexes and views. The above could also be extended to the secondary key, index as well as view information by defining the secondary key, indexes and views. Data dictionaries

do not contain any actual data from the database, it contains only book-keeping information for managing it. **Without a data dictionary, however, a database management system cannot access data from the database.**

The Database Library is built on a *Data Dictionary*, which provides a complete description of record layouts and indexes of the database, for validation and efficient data access. The data dictionary can be used for automated database creation, including building tables, indexes and referential constraints, and granting access rights to individual users and groups. The database dictionary supports the concept of Attached Objects, which allows database records to include compressed BLOBs (Binary Large Objects) containing images, texts, sounds, video, documents, spreadsheets, or programmer-defined data types.

Q19. Discuss the disadvantages of data dictionary?

Ans. A DDS is a useful management tool, but it also has several disadvantages. It needs careful planning. We would need to define the exact requirements designing its contents, testing, implementation and evaluation. The cost of a DDS includes not only the initial price of its installation and any hardware requirements, but also the cost of collecting the information entering it into the DDS, keeping it up-to-date and enforcing standards. The use of a DDS requires management commitment, which is not easy to achieve, particularly where the benefits are intangible and long term.

Q20. What are the benefits and features of data dictionary?

Ans. A comprehensive data dictionary product will include:

- support for standard entity types (elements, records, files, reports, programs, systems, screens, users, terminals, etc.) and their various characteristics (e.g., for elements, the dictionary might maintain Business name, Business definition, name, Data type, Size, Format, Range(s), Validation criteria, etc.)
- support for user-designed entity types (this is often called the "extensibility" feature); this facility is often exploited in support of data modelling to record and cross-reference entities, relationships,data flows, data stores, processes, etc.
- the ability to distinguish between versions of entities (e.g., test and production) enforcement of in-house standards and convetions.
- comprehensive reporting facilities, including both "canned" reports and a reporting language for user-designed reports; typical reports include :
- detail reports of entities

- summary reports of entities
- component reports (e.g., record-element structures)
- cross-reference reports (e.g., element keyword indexes)
- where-used reports (e.g., element-record-program cross-references).
- a query facility, both for administration and casual users, which includes the ability to perform generic searches on business definitions, user descriptions, synonyms etc.
- language interfaces, to allow, for example, standard record layouts to be automatically incorporated into programs during the compile process.
- automated input facilities (e.g. to load record descriptions from a copy library).
- security features
- adequate performance tuning abilities
- support for DBMS administration, such as automatic generation of DDL (Data Definition Language).

Data Dictionary Benefits : The benefits of a fully utilised data dictionary are substantial. A data dictionary has the potential to :

* facilitate data sharing by

- enabling database classes to automatically handle multi-user coordination, buffer layouts, data validation, and performance optimisations,
- improving the ease of understanding of data definitions,
- ensuring that there is a single authoritative source of reference for all users.
- facilitiate application integration by identifying data redundancies,
- reduce development lead times by
    - simplifying documentation
    - automating programming activities.
- reduce maintenance effort by identifying the impact of change as it affects:
    - users,
    - database administrators,
    - programmers
- improve the quality of application software by enforcing standards in the development process.
- ensure application system longevity by maintaining documentation beyond project completions.
- data dictionary information created under one database system can easily be used to generate the same database layout on any of the other database systems BFC supports (Oracle, MS SQL Server, Access, DB2, Sybase, SQL, Anywhere,

etc.)

Q21. List the various types of views through which access to the data dictionary is allowed.

Ans. a) Views with the Prefix USER

b) Views with the Prefix ALL

c) Views with the Prefix DBA

Q22. What are the uses of Data Dictionary in Oracle?

Ans. Primary uses are :

- Oracle accesses the data dictionary to find information about users, schema objects, and storage structures.
- Oracle modifies the data dictionary every time that a data definition language (DLL) statement is issued.

Q23. What is the difference between active and passive Data Dictionary?

Ans. If a data dictionary system is used only by designers, users, and administrators, not by the DBMS software, it is called a passive data dictionary otherwise, it is called an active data dictionary or data directory. An active data dictionary is automatically updated as changes occur in the database. A passive data dictionary must be manually updated.

Q24. What are the approaches to implementing a distributed database catalogue?

Ans. a) Centralised: Keeps one master copy of the catalogue.

b) Partitioned: Partitions and replicates the catalogue as per the demands of the usage pattern.

c) Fully replicated: Keeps one copy of the catalogue at each site.

d) Centralised/partitioned: Combination of the above

Q25. How are Object Oriented catalogues implemented?

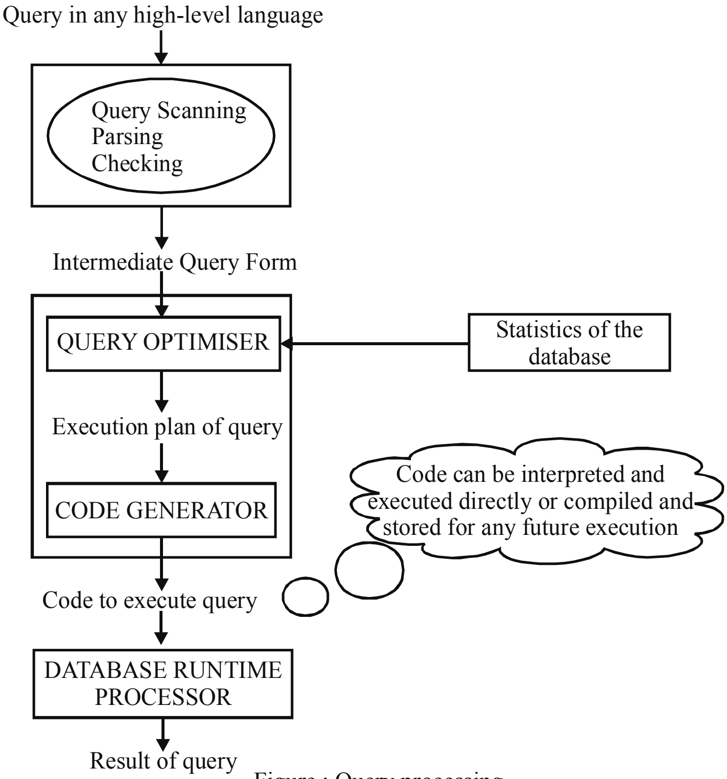
Ans. In addition to system catalogues that may be used in simple relational database system, an object oriented system catalogue needs to define complex user defined classes, inheritance hierarchy and complex inter-relationships.

Chapter-5

DBMS Advanced Features and Distributed Database

Q1. With the help of a block diagram describe the phases of Query processing. How do we optimize a Query under consideration? Does Query optimisation contribute to the measurement of Query cost? Support your answer with suitable explanation. [June-07, Q1(d)]

Ans.



Query in any high-level language
Query Scanning
Parsing
Checking
Intermediate Query Form
QUERY OPTIMISER
Statistics of the database
Execution plan of query
CODE GENERATOR
Code can be interpreted and executed directly or compiled and stored for any future execution
Code to execute query
DATABASE RUNTIME PROCESSOR
Result of query

Figure : Query processing

In the first step Scanning, Parsing, and Validating is done to translate the query into its internal form. This is then further translated into relational algebra (an intermediate query form). Parser checks syntax and verifies relations. The query then is optimized with a query plane, which then is compiled into a code that can be executed by the database runtime processor.

We can define query evaluation as the query-execution engine taking a query-evaluation plan, executing that plan, and retuning the answers to the query. The query processing involves the study of the following concepts:

- how to measure query costs?
- algorithms for evaluating relational algebraic operations.
- how to evaluate a complete expression using algorithms on individual operations?

A relational algebra expression may have many equivalent expressions. For example,

$\sigma_{(salary<5000)}$ $(\pi_{salary}$ (EMP)) is equivalent to $(\pi_{salary}$ $(\sigma_{salary<5000}$ <(EMP)).

Each relational algebraic operation can be evaluated using one of the several different algorithms. Correspondingly, a relational-algebraic expression can be evaluated in many ways.

An expression that specifies detailed evolution strategy is known as evaluation-plan, for example, you can use an index on salary to find employees with salary< 5000 or we can perform complete relation scan and discard employees with salary $\geq$ 5000. The basis of selection of any of the scheme will be the cost.

Query Optimisation: Amongst all equivalent plans choose the one with the lowest cost. Cost is estimated using statistical information from the database catalogue, for example, number of tuples in each relation, size of tuples, etc. Thus, in query optimisation we find an evaluation plan with the lowest cost. The cost estimation is made on the basic of heuristic rules.

Measure of Query Cost

Cost is generally measured as total elapsed time for answering the query. There are many factors that contribute to time cost. These are disk accesses, CPU time, or even network communication.

Typically disk access is the predominant cost as disk transfer is a very slow event and is also relatively easy to estimate. It is measured by taking into account the following activities:

Number of seeks × average-seek-cost
Number of blocks read × average-block-read-cost
Number of blocks written × average-block-written-cost.

Q2. What are the various methods adopted in select operation?

Ans. Selection operation can be performed in a number of ways such as:

File Scan	**Index Scan**
1. Linear Search	1.(a) Primary index, equality
	(b) hash key
2. Binary Search	2. Primary index, Comparison
	3. Equality on clustering index
	4. (a) Equality on search key of secondary index
	(b) Secondary index comparison

Q3. Define the algorithm for Block Nested-Loop Join for the worst-case scenario.

Ans.
```
For each block $B_r$ of r {
          For each block tuple $B_s$ of s {
                    For each tuple $t_i$ in $B_r$ {
                              For each tuple $t_i$ in Bs {
                    Test Pair ($t_i$, $s_i$) to see if they satisfy the join condition
                    If they do, add the joined tuple to result
                              };
                    };
          };
};
```

Q4. Give the method for calculating the cost of Hash-Join?

Ans. Cost of Hash-join.

(i) If recursive partitioning not required 3 (blocks of r + Blocks of s)

(ii) If recursive partitioning required then

2 (blocks of r + blocks of s (j log$_{m-1}$ (blocks of s) k - 1) + blocks of r + blocks of s.

Q5. Define Hybrid Merge-Join.

Ans. Hybrid Merge-Join is applicable when the join is equi-join or natural join and one relation is sorted.

Merge the sorted relation with leaf of B+tree.
(i) Sort the result on address of sorted relations tuples.
(ii) Scan unsorted relation and merge with previous result.

Q6. Define other operations?

Ans. There are many Other operations that are performed in database systems which are given below :

(a) Duplicate elimination : Duplicate elimination may be implemented by using hashing or sorting. On sorting, duplicates will be adjacent to each other thus, may be identified and deleted. An optimised method for duplicate elimination can be deletion of duplicates during generation as well as at intermediate merge steps in external sort-merge. Hashing is similar - duplicates will be clubbed together in the same bucket and therefore may be eliminated easily.

(b) Projection : It may be implemented by performing the projection process on each tuple, followed by duplicate elimination.

(c) Aggregate functions : Aggregate functions can be implemented in a manner similar to duplicate elimination.

d) Set Operations - can use either use a variant of merge-join after sorting, or a variant of hash-join.

QUERY EXPRESSIONS :

Q7. Discuss the methods for evaluating an entire expression.

Ans. Methods for evaluating an entire expression are :

- Materialisation
- Pipelining

Materialisation : Evaluate a relational algebraic expression from the bottom-up, by explicitly generating and storing the results of each operation in the expression.

Materialised evaluation is always possible though; the cost of writing/reading results to/from disk can be quite high.

Pipelining : Evaluate operations in a multi-threaded manner, (i.e. passes tuples output from one operation to the next parent operation as input) even as the first operation is being executed. In the previous expression tree, it does not store (materialise) results instead, it passes tuples directly to the join. Similarly, does not store result of join, and passes tuples directly to the projection. Thus, there is no need to store a temporary relation on a disk for each operation. Pipelining may not always be possible or easy for sort, hash-join.

One of the pipelining execution methods may involve a buffer filled by the result tuples of lower level operation while, records may be picked up from the buffer by the higher level operation.

Q8. For the following relations, write the SQL statement to find the details of the student(s) who have scored grade A in MCS-031. Also write its equivalent relationship algebraic query and form its query tree.
STUDENT (enrolno, name, age)
MARKS (enrolno, subjectcode, grade)

Ans. Relation given:

STUDENT (enrolno, name, age)
MARKS (enrolno, subjectcode, grade)

Query : To find detail of the student(s) who have scored grade A in MCS-031.

SQL:

```
SELECT *
FROM Student
WHERE enrolno in
            (SELECT enrolno
FROM MARKS
Where subject code = ‘MCS-031’ and grade = ‘A’);
```

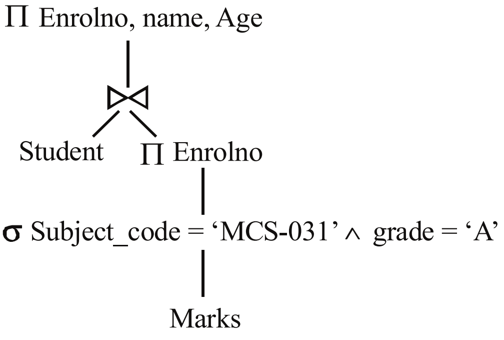
Relational Algebra :

$$R_1 \leftarrow \sigma \text{subjectcode = 'MCS-031' ^ grade = 'A'} \left(\text{MARKS}\right)$$

$$R_2 \leftarrow \prod \text{enrolno}\left(R_1\right)$$

$$\text{Result} \leftarrow \left(\sigma \text{student.enrolno=R}_2\text{.enrolno}\left(\text{R}_2\text{X Student}\right)\right)$$

Query tree:



Q9. How you define evaluation plan?

Ans. Evaluation plan defines exactly what algorithms are to be used for each operation and the manner in which the operations are coordinated.

Q10. How you define cost based optimization?

Ans. Cost based optimisation is

a) Generalizing logically equivalent expressions using equivalence rules

b) Annotating resultant expressions to get alternative query plans.

c) Choosing the cheapest plan based on estimated cost.

Q11. Consider the following relational database :

EMPLOYEE (emp_name, street, city)
WORKS (emp_name, comp_name, salary)
COMPANY (comp_name, city)
MANAGES (emp_name, manager_name)

For each of the following queries, give the relational algebraic expression and relational calculus expression :

(i) Find the names of all the employees who work for XYZ Bank Corporation.

$\pi_{\text{emp\_name}}(\sigma_{\text{comp-name = ' xyz Bank Corporation'}}\ (\text{WORKS}))$

{e [emp_name]|e ∈ works ^ e[emp-name] = ‘xyz Bank Corporation’}

(ii) Find the names of all the employees who live in the same city where the company for which they work is located.

$\pi_{\text{emp\_name}}\ (\sigma_{\text{EMPLOYEE.City=COMPANY.City}}$
(EMPLOYEE⋈WORKS ⋈ COMPANY))

{e[emp_name] |e ∈ EMPLOYEE ^ ∃ c,w (c ∈ COMPANY ^ w ∈ WORKS^ e[emp_name] = w [emp_name] ^ w [comp_name] = c [comp_name] ^ e [city] = c[city]

(iii) Find the names of those employees who earn more than every employee of the XYZ Bank Corporation.

$\pi_{\text{emp\_name}}\ (\sigma_{\text{salary>(Jmax(salary) (}\sigma\text{comp\_name = 'xyz Bank Corporation'}}\ (\text{WORKS})\ ))\ (\text{WORKS}))$

{e[emp_name] |e ∈ WORKS ^ ∃ w (W ∈ WORKS ^
w [comp_name] = ‘xyz Bank Corporation^ e[salary] > w [salary] }

(iv) Find the name, street and city of those employees who work for the XYZ Bank Corporation and earn more than Rs. 2,50,000 per annum.

$\pi_{EMPLOYEE.*}(\sigma_{comp\_name='xyz\ Bank\ Corporation' \wedge salary > 250000/12}(EMPLOYEE \bowtie WORKS)$

{e|e ∈ EMPLOYEE ^ ∃w (W ∈ WORKS ^ w [salary] > 250000/12 ^ w [comp_name = 'xyz Bank Corporation'
^ e[emp_name] = w[emp_name]}

(v) Find the names of managers who work in a bank located in Delhi.

$\pi_{manager\_name}(\sigma_{city='Delhi'}(MANAGES \bowtie_{manager\_name=emp\_name} WORKS \bowtie COMPANY))$

{e[manager_name] |e ∈ MANAGES ^ ∃ c, w (W ∈ WORKS ^ c ∈ company ^ e [manager_name] = w [emp_name] ^ w[comp_name] = c[comp_name] ^ c [city] = 'Delhi'}

Q12. Consider the relations .
GRADE (stud_id, subject_id, grade)
SUBJECT (subject_id, s_name, teacher)

(i). Write the relational algebraic query for the following :

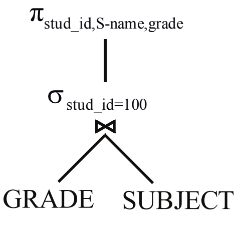
. List the student number, subject names and grades of the student whose id is 100.

. List the grades of all the students in the subjects taught by teacher "XYZ".

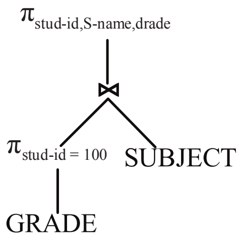
$\pi_{stud\_id,\ S\_name,\ grade}(\sigma_{stud\_id=100}(GRADE \bowtie SUBJECT))$

$\pi_{stud\_id,\ grade}(\sigma_{teacher='XYZ'}(GRADE \bowtie SUBJECT))$

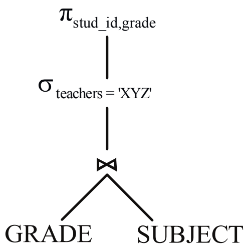
(ii). Convert the queries above into an optimized query graph.



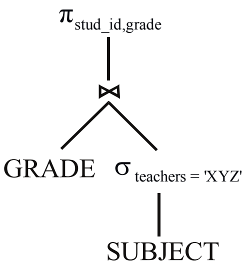
Initial Query graph



Optimized Query Graph



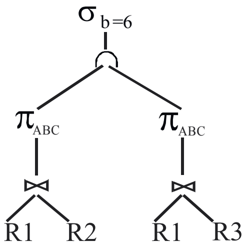
Initial Query graph



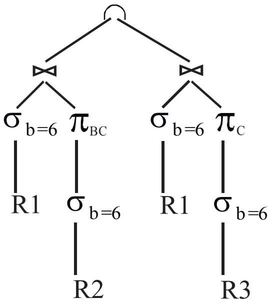
Optimized Query Graph

Q13. Optimize the following queries for the relation R1(A, B, C), R2(B, C, D) and R3(C, D, E)

(i)



Initial Query Tree

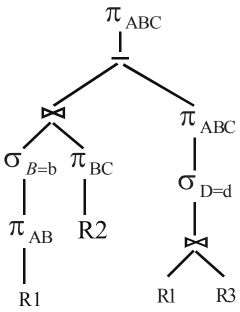


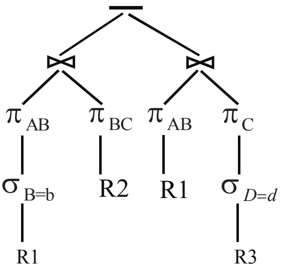
Optimized Query Tree

Optimized Query is:

$(\sigma_{B-b}(R1) \bowtie (\pi_{BC}(\sigma_{B-b}(R2))) \cap (\pi_{B-b}(R1) \bowtie \pi_{C}(\sigma_{B-b}(R3)))$

(ii)





Optimized Query is

: $(\pi_{AB}(\sigma_{B-b}(R1)) \bowtie \pi_{BC}(R2))-(\pi_{AB}(R1) \bowtie (\pi_{C}(\sigma_{D=d}(R3)))$

TRANSACTION MANAGEMENT & RECOVERY

Q14. What are the various transaction states ? Explain each state and state transition with the help of example(s).

Ans. Active ⇒ It is the initial state. The transaction stays in this state while it is executed.

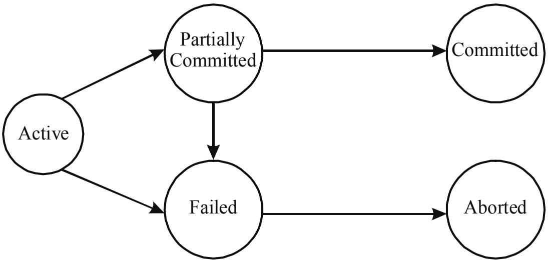
Partially Committed ⇒ A transaction that completes its execution successfully said to be committed, partially committed state comes after the final statement has been executed.

Failed ⇒ Failed state comes after the discovery that normal execution can no longer proceed.

Aborted ⇒ This state comes after the transaction has been rolled back & the database has been restarted to its state prior to the start of the transaction.

Committed ⇒This state comes after the successful completion of the transaction.

The state diagram corresponding to a transaction is shown in following figures:-



Example:

Consider the following transaction:

T

```
1. Sum:=0
2. Read(A)
3. Read(B)
4. Sum = A + B
5. Write(A)
6. Write(B)
```

When we issue the transaction *T* and it is ready for execution then it will be in the Active state. When it is in process (line 1 to 5) it will be in the Partially Committed state. If any operation fails at any point then it will be in the Failed state and followed by Aborted state. If the last operation is completed and successfully communicated to physical database then it will be in the Committed state.

Q15. What do you mean by transaction? What are the properties of a transaction? Violation of which property causes Lost Update problem? Justify your answer with suitable example. [June-07, Q1(e)]

Ans. A transaction may be defined as a collection of operations on the database that performs a single logical function in a database application or it should be an inseparable list of database operations or Logical Unit of Work (LUW). It should not violate any database integrity.

When a transaction processing system creates a transaction, it will ensure that the transaction has certain characteristics. These characteristics are known as the ACID properties.

ACID is an acronym for ATOMICITY, CONSISTENCY, ISOLATION and DURABILITY.

ACID Properties

Atomicity : The atomicity property identifies the transaction as atomic. An atomic transaction is either fully completed, or is not begun at all.

Consistency : A transaction enforces consistency in the system state, by ensuring that at the end of the execution of a transaction, the system is in a valid state. If the transaction is completed successfully, then all changes to the system will have been made properly, and the system will be in a valid state. If any error occurs in a transaction, then any changes already made will be automatically rolled back.

Isolation : When a transaction runs in isolation, it appears to be the only action that the system is carrying out at a time. If there are two transactions that are performing the same function and are running at the same time, transaction isolation will ensure that each transaction thinks that it has exclusive use of the system.

Durability : A transaction is durable in that once it has been successfully completed, all of the changes it has made to the system are permanent. There are safeguards that will prevent the loss of information, even in the case of system failure.

Violation of durability causes lost update problem.

Suppose the 2 transactions T_3 and T_4 run concurrently and they happen to be interleaved in the following way (assume the initial value of X as 10000).

T_3	T_4	Value of X	
		T_3	T_4
Read X		10000	
	Read X		10000
Sub 5000		5000	
	Add 3000		13000
Write X		5000	
	Write X		13000

After the execution of both the transactions the value X is 13000 while the semantically correct value should be 8000. The problem occurred as the update made by T_3 has been over written by T_4. The root cause of the problem was the fact that both the transactions had read the value of X as 10000. Thus one of the two update has been lost and we say that a lost update has occurred.

Q16. Why we use lock based protocol? What are the pitfalls of lock based protocol?

Ans. Lock based protocols are one of the major techniques to implement concurrent transaction execution. These protocols try to ensure serialisability. The pitfalls of such protocols can be:

- Overheads due to locking
- Locking may lead to deadlocks.
- We may either restrict concurrency – strict 2 PL or allow uncommitted data to be seen by other transactions.

Q17. How do you define transaction processing methods? Also list the

advanced transaction methods.

Ans. The transaction processing methods can be:

Batch-combining jobs in batches

Online-the effects are known immediately

Advanced methods may include real time transaction, weak consistency levels etc.

Q18. What is timestamp? What is the need of time stamping protocol?

Ans. A timestamp is a label that is allotted to a transacting when it starts. This label is normally generated as per some time unit so that transactions may be identified as per time order. They allow concurrency control but without letting transaction wait for a lock. Thus, problems such as deadlocks may be avoided.

Q19. How are insert and delete operations performed in a tree protocol?

Ans. The main problems of Insert and Delete are that it may result in inconsistent analysis when concurrent transactions are going on due to phantom records. One such solution to this problems may be locking the index while performing such an analysis.

Q20. Define multiple granularity?

Ans. Multiple granularity defines object at various levels like database, file, record, fields, etc.

Q21. Why does deadlock occur in concurrent execution of transactions where locking scheme is followed? How can you detect deadlock in database system? How is the problem of deadlock resolved? Explain with the help of an example.

Ans. Deadlock, is a situation of the system, when each transactions waits for some resources, which are hold by some of the resources.

This occurs in the system where locking scheme is allowed, because the locked item can not released by other transaction, but by the locked transaction itself.

Example:-

	T1	T2
	1. read-lock (Y)	3. read-lock (X);
Time	2. read-item (Y);	4. read-item (X);
		6. write-lock (Y);
	5. write-lock (X);	

Fig.(a)

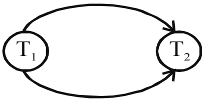


Fig. (b)
(a wait-for graph)

The Fig(a) show the schedule of transactions T_1 & T_2 where both are in deadlock state, as T_1 is waiting for X and T_2 is waiting for Y Neither T_1 nor T_2 nor any other transaction can access items X and Y_1 and X and Y are locked.

Detection of Deadlock ⇒ Deadlock detection algorithm examines the state of the system to decide whether deadlock has occurred. Such algorithms are invoked periodically. To implement this algorithm, the system must do the following things:

- Maintain Information about the current allocation of data items to transactions as well as any outstanding data item request.
- Provides an algorithm that uses this information to determine whether the system has entered a deadlock state.
- Recovers from the deadlock when the detection algorithm determines that deadlock exists.

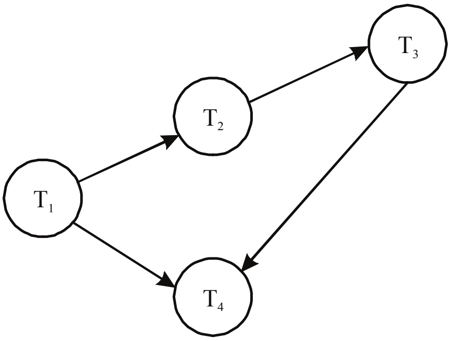
Wait -for-Graph ⇒ Wait-for-Graph is used by deadlock detection algorithm to decide whether deadlock occurs.

This graph describes the deadlock. This graph consists of a pair G = (V, E) where V is a set of vertices & E is a set of Edges. The set of vertices consists of all the transactions in the system. Each element in the set E of edges is an ordered pair T_i →| T_j. If $T_i \rightarrow T_j$ is in E, then there is a directed edge from transactions T_i to T_j implying that transactions T_i is waiting for transactions T_j to release a data item that it needs.

When transaction T_i requests a data item currently being held by transaction T_j, then the edge T_i →| T_j is inserted in the Wait-For-Graph. This edge is removed only when transaction T_j is no longer holding a data item needed by transaction T_i.

A deadlock exists in the system if & only if the wait-for-graph contains a cycle. Each transaction involved in the cycle is said to be deadlocked. To detect deadlocks, the system needs to maintain the wait-for-graph & periodically to invoke an algorithm that searches for cycle in the graph.

Following figure illustrates the concept in which a deadlock is detected through. Wait-for-Graph.



In the above figure, transaction T_1 is requesting an item held by transaction T_2. Transaction T_1 is again requesting an item which is held by transaction T_4. Transaction T_2 is requesting for an item which is held by transaction T_4 & transaction T_4 is requesting for an item which is held by transaction T_3.

$T_2 \rightarrow T_4 \rightarrow| T_3 \rightarrow| T_1$ is a cycle and transaction T_1 , T_2, T_3, T_4 are deadlocked. To resolve deadlock situations two actions are taken:

(i) To determine a transaction which has caused the deadlock situation and the cost of rolling back of such transaction should be minimum. Many factors may determine the cost of rolling back of a transaction.

→| How long the transaction has computed, & how much longer the transaction will compute before it completes its designed task.

→| How many data items the transaction has used.

→| How many transactions will be involved in the roll back.

(ii) Once we have decide that a particular transaction must be rolled back, we must determine how far this transaction should be rolled back : There are two choices :

- Total rollback means abort the transaction & restart it.
- Partial rollback means rollback the transaction only as far as necessary to break the deadlock.

The transaction must be capable to resume execution after a partial rollback.

Deadlock Avoidance :- In a deadlock avoidance scheme, case is taken to ensure that a circular chain of processes holding some resources & waiting for additional ones held by other transactions in the chain never occurs. The two phase locking protocol ensures serializability but does not ensure a deadlock free situation. There are various approaches for avoiding deadlocks. Some of them are as follows:

(i) One simplest Method for avoiding deadlock into lock all data items at the beginning of a transaction.

This has to be done in atomic manner otherwise there could be a deadlock situation again. Disadvantages of this approach are:

(a) Degree of concurrency is covered considerably.

(b) Locking all data items for the entire duration of a transaction makes there data items inaccessible to other concurrent transactions.

(ii) Another approach for the avoidance of Deadlock is to decide whether to wait or abort or roll back a transaction if the transaction has requested a data item that is locked in an incompatible mode by another transaction. The decision for abort or roll back or wait is controlled by time-stamp values. Aborted or rolled back transactions retain their time-stamp values & hence their seniority so that in subsequent sections they would eventually get a higher priority. We examine two such approaches which is as follows:

(a) Wait-Die :-

- If the requesting transaction is elder that the transaction that holds the lock on the requested data item the requesting transaction is allowed to wait.
- If the requesting transaction is younger that the transaction that holds the lock on the requested data item, the requesting transaction is aborted or rolled back.

(b) Wound-Wait :- this is just the opposite of wait-die scheme.

- If a younger transaction holds data-item requested by an older one, the younger transaction is the one that would be aborted and rolled back.
- If a young transaction requested a data item held by an older transaction the younger transaction is allowed to wait.

Q22. Define multi-version scheme?

Ans. Multi-version technique allows the creation of multi-version of each data items as it gets updated. It also records the timestamp of updating transaction for that data. This protocol tries to make available correct data as per transaction timestamp.

Q23. Define weak level of consistency in SQL?

Ans. Weak level of consistency in SQL allows transaction to be executed at a lower level. In such cases the programmer need to ensure that there is no concurrency related problems. SQL provides the following weak consistency levels. Read committed, and Repeatable Read.

Q24. Define dead locks? How can dead locks be avoided?

Ans. Deadlock is a situation when two or more transactions are waiting for each other to release locks held by others. A deadlock can be detected when a cycle is detected in the wait for graph. The basic procedure of deadlock avoidance may be to roll back certain transaction as per some rule such as wound-wait or allow the transaction a maximum time to finish after which it may be rolled back.

Q25. How is stable storage implemented?

Ans. It can only be implemented with more secondary storage, which are placed at different locations to take care of natural disasters or catastrophic events.

Q26. Compare and contrast the features of simple locking, intention mode locking and time stamping mechanisms from the viewpoint of transaction control under concurrent transitions.

Ans.

Simple Locking	**Intention Mode Locking**	**Time Stamping**
1. Transaction have to hold onto the lock on a data item atleast until it needed that item and may have to hold beyond the need of the data item	Intention mode provides greater flexibility and efficiency than Simple Locking	Does not allow any data item, transactions are ordered in the order they were submitted.
2. It has to modes exclusive and shared	It has five modes S, X, IS, IX SIX	Each data item has two stamps write time stamp and record time stamp.
3. It causes less overhead than IM Locking	It creates more overhead in maintenance of locks	Optimum overhead
4. Less costly in terms of time	Costly in terms of time as a transaction has to search a data item's previous locks and modes and it ascertains if a request for a lock can be granted	No time overhead
5. May cause deadlock.	Prone to deadlock	ensure serilizability but causes transaction to be rolled back.

Q27. Define recovery algorithms.

Ans. Recovery algorithms are techniques to ensure database consistency and transaction atomicity and durability despite failures. Recovery algorithms have two parts:

(1) Actions taken during normal transaction processing is to ensure that enough information exists to recover from failures,

(2) Actions taken after a failure to recover the database contents to a state that ensures atomicity, consistency and durability.

While modifying the database, without ensuring that the transaction will commit, amy leave the database in an inconsistent state. Let us consider an example transaction T_i that transfers Rs. 1000/- from account X to account Y; goal is either to perform all database modifications made by T_i or none at all. T_i modifies x by subtracting Rs. 1000/- and modifies Y by adding Rs. 1000/-. A failure may occur after one of these modifications has been made, but before all of them are made. To ensure consistency despite failures, we have several recovery mechanisms.

Q28. Describe four major threats to database security and integrity.

Ans. Four major threats to database security and integrity : Database security usually refers to protection from unauthorized access, whereas integrity refers to the avoidance of accidental loss of consistency. To protect the database we must take security & integrity measures at several levels :

(1) Physical and Human : In physical, the site or sites containing the computer systems must be physically secured against armed or surreptitious entry be intruders. In human, users must be authorized carefully to reduce the chance of any such user giving access to an intruder in exchange for a bribe or other favours.

(2) Operating system : No matter how secure the database system is, weakness in operating system security may serve as a means of unauthorized access to the database.

(3) Network : Since almost all database systems allow remote access through terminals or network, software level security within the network software is as important as physical security, both on the internet and in network private to an enterprise.

(4) Database system : Some database system users may be authorized to access only a limited portion of the database, other users may be allowed to issue queries, but may be forbidden to modify the data. It is the responsibility of the database system to ensure that these authorization restrictions are not violated.

Q29. What is a log file? What does it contain? How can a log file be used for recovery? Describe this with the help of an example that includes all recovery operations (UNDO, REDO, etc)

Ans. Log ⇒ contains the redundant data, which will be used in case of recovery. Log information is kept on stable storage. In case of a system crash losing volatile information. The modified data in buffers and log information in buffers will be lost. The transactions having commit marker will be redone and the transactions which have missing commit marker will be undone. It means that longer the system operates without a crash the longer it will take to recover from the crash. The recovery operation after a system crash will have to reprocess all transactions from the time of start up of the system. These problems can be avoided by using a means to propagate volatile information to stable storage at regular intervals. This scheme is called check pointing check pointing is used to limit the volume of log information to be processed in case of a system failure.

We will prefer log based recovery over check pointing where we have low traffic to database and it has high cost to checkpoint frequently such as in an EMI calculation application. Check pointing scheme will used where so many transactions are running per second and changing data. Such as in an online reservation system.

Immediate update, or UNDO/REDO, is another algorithm to support ABORT and machine failure scenarios.

- While a transaction runs, changes made by that transaction can be written to the database at any time. However, the original and the new data being written must both be stored in the log BEFORE storing it on the database disk.
- On a commit:
- All the updates which has not yet been recorded on the disk is first stored in the log file and then flushed to disk.
- The new data is then recorded in the database itself.
- On an abort, REDO all the changes which that transaction has made to the database disk using the log entries.
- On a system restart after a failure, REDO committed changes from log.

Example :

Using immediate update, and the transaction TRAN1 again, the process is:

Time	Action	LOG
t1	START	-
t2	read(A)	-
t3	write(10,B)	Was B == 6, now 10
t4	write(20,C)	Was C == 2, now 20
t5	COMMIT	COMMIT

Before		During		After
DISK	B = 6	B = 10		B = 10
A = 5	C = 2	A = 5	C = 2	A = 5 C = 20

If the DMBS fails and is restarted:

- The disks are physically or logically damaged then recovery from the log is impossible and instead a restore from a dump is needed.
- If the disks are OK then the database consistency must be maintained. Writes to the disk which was in progress at the time of the failure may have only been partially done.
- Parse the log file, and where a transaction has been ended with 'COMMIT' apply the 'new data' part of the log to the database.
- If a log entry for a transaction ends with anything other than COMMIT, apply the 'old data' part of the log to the database.
- flush the data to the disk, and then truncate the log to zero.

Q30. Suppose the write ahead Log Scheme is being used. Give the REDO and UNDO processes in the strategy of writing the partial update made by a transaction to the database, as well as in the strategy of delaying all writes to database till the commit. 5

Ans. In write-ahead log strategy the transaction updates the physical database and the modified record replaces the old record in the database on nonvolatile storage. The log information about the transaction modifications are written before the corresponding put(x) operation, initiated by the transaction, is performed. The write-ahead log strategy has the following requirements:

1. Before a transaction is allowed to modify the database, at lest the undo portion of the transaction log record is written to the stable storage.

2. A transaction is committed only after both the undo and the redo portion of the log are written to stable storage.

In the strategy of delaying all writes to database till the commit : the sequence of operations for transaction T and the actions performed by the database are shown below :

Time	Transaction Step	Log Operation	Database Operation
t_0	Start of T	Write(start Transaction T)	
t_1	get(X)		Read(X)
t_3	modify(X)		
t_4	put(X)	Write(record X modified, old value of X, new value of X)	
t_4			write(X)
t_5	get(Y)		Read(Y)
t_6	modify(Y)		
t_7	put(Y)	Write(record Y modified, old value of Y, new value of Y)	
t_8			write(Y)
t_9	Start Commit	Write(Commit Transaction);	
t_{10}	End of T		

The initiation of a transaction causes the start of the log of its activities; a start transaction along with the identification of the transaction is written out to the log. During the execution of the transaction, any output is preceded by a log output to indicate the modification being made to the database. This output to the log consists of the record(s) being modified, old values of the data items in the case of an update, and the values of the data item. The old values will be used by the recovery system to undo the modifications made by a transaction in case a system crash occurs before the completion of the transaction. When a system crash occurs after a transaction commits, the new values will be used by the recovery system to redo the changes made by the transaction and thus ensure that the modifications made by a committed transaction are correctly reflected in the database.

The transaction shown in before consists of reading in the value of some data item X and modifying it by a certain amount. The transaction then reads in the value of another data item Y and modifies it by an equal but opposite amount. The transaction may subtract, let us say, a quantity n from the inventory for part Px and add this amount to quantities of that item shipped to customer Cy. For consistency this transaction must be completed atomically. A system crash occurring at any time before time t_9 will require that the transaction be undone.

A system crash occurring after t_9, when the commit transaction marker is written to the log, requires that we redo the transaction to ensure that all of the changes made by this transaction are propagated to the physical database. According to the write-ahead log strategy, the redo portion of the log need not be written to the log until the commit transaction is issued by the program performing the transaction.

In the strategy of writing partial updates : If the crash occurs just during or after t_4, the recovery process, when it examines the log, find that the commit transaction marker is missing and, hence will undo the partially completed transaction. To do this it will use the old value for the modified field. And restore the database to the consistent state that existed before the crash and before transaction was started. If the crash occurs after step t_9 is completed, the recovery system will find an end-of-transaction in the log.

However, since the log was written before the database, all modifications to the database may not have been propagate d to the database. Thus to ensure that all modifications made by transaction are propagated to the database, the recovery system will redo the committed transaction. To do this it uses the new values of the appropriate fields of the records.

Q31. What are the steps one must take with its database management system, in order to ensure disaster recovery? Define the process of recovery in case of disaster.

Ans. Disasters refers to circumstances that result in the loss of the physical database stored on the nonvolatile storage medium.

The data stored in stable storage consists of the archival copy of the database and the archival log of the transactions on the database represented in the archival copy.

The disaster recovery process requires a global redo. In this case the changes made by every transaction in the archival log are redone using the archival database as the initial version of the current database.

- As the order of redoing the operation must be same as the original order, hence the archival log must be chronology ordered.
- Since the archival database should be consistent, it must be a copy of the current database in a quiescent stage.

The recovery operation consists of redoing the changes made by committed transactions from the archive log on the archive database. A new consistent archive database copy can be generated during this recovery process.

Q32. Compare and contrast the features of log based recovery mechanism versus check pointing based recovery. Suggest applications where you will prefer log based recovery scheme over check pointing. Give an example of check pointing based recovery scheme.

Ans. Log contains the redundant data, which will be used in case of recovery. Log information is kept on stable storage. In case of a system crash losing volatile information. The modified data in buffers and log information in buffers will be lost. The transactions having commit marker will be redone and the transactions which have missing commit marker will be undone. It means that longer the system operates without a crash the longer it will take to recover from the crash. The recovery operation after a system crash will have to reprocess all transactions from the time of start up of the system. These problems can be avoided by using a means to propagate volatile information to stable storage at regular intervals. This scheme is called check pointing check pointing is used to limit the volume of log information to be processed in case of a system failure.

We will prefer log based recovery over check pointing where we have low traffic to database and it has high cost to checkpoint frequently such as in an EMI calculation application. Check pointing scheme will used where so many transactions are running per second and changing data. Such as in an online reservation system.

Recovery is a process of restoring database to the consistent state that existed before the failure. It is an integral part of the database. The recovery scheme must minimize the time for which the database is not usable after a crash.

The most widely used structure for recording database modifications is the log. The log is a sequence of log records all the update activities in the database. The log contains redundant data required to recover database to a consistent state. The log is generally stored on stable storage.

For each transaction, the following data is recorded on the log:-

- A start-of-transaction Marker.
- The transaction identifier, which includes the who & where information.
- The record identifies, which includes identifiers for record occurrences.
- The operation (Insert, Delete, Modify) performed on the records.
- The previous value(s) of the Modified data. This information is required for undoing the change mode by a partially completed transaction.

- A Commit transaction marker of the transaction is committed otherwise an abort or rollback transaction marker.

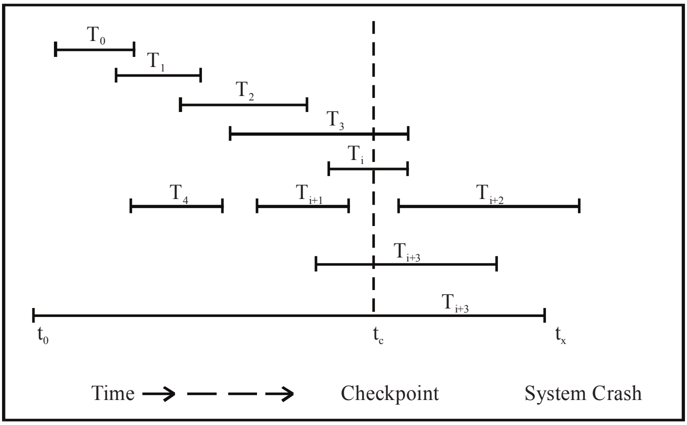
The log is written before any update is made to he database. This is called write ahead-log strategy.

Checkpoints ⇒ A scheme called checkpoint is used to limit the volume to volume of log information that has to be handled and processed in the event of a system failure involving the loss of volatile information. The checkpoint scheme is an additional component of the loging scheme described above.

In the case of a system crash, the log information being collected in buffers will be lost. A checkpoint operation, performed periodically, copies log information on to stable storage. The information and operations performed at each checkpoint consist of the following:

- A start-of-checkpoint record giving the identification that it is a checkpoint along with the time and data of the checkpoint is written to the log on a stable storage device.
- All log information from the buffers in the volatile storage is copied to the log on stable storage.
- All database updates from the buffers in the volatile storage are propagated to the physical database.
- An end-of-checkpoint record is written and the address of the checkpoint record is saved on a file accessible to the recovery routine on start-up after a system crash.

Example : Let us consider a set of transactions with the checkpoint and system crash time, as shown below:-



A checkpoint is taken t_c time Tc, which indicates that at that time all log and data buffers were propagated to storage.

The transactions T_0, T_1, T_2, T_4 were committed and their modifications are reflected in the database. With the checkpoint scheme, these transactions are not required to be redone after the system crash.

The transactions T_i and T_{i+3} are to be redone at the point of checkpoint, where the transaction T_{i+2} has to be undo, as no log has been written for this transaction.

Q33. Assume that the Railway reservation system is implemented using an RDBMS. What concurrency control measures would you take in order to avoid concurrency related problems in the system above? Also suggest two measures to avoid deadlock in this system.

Ans. In the Railway Reservation system the following concurrency control measures would be required in order to avoid concurrency related problems :

1. A seat can not be reserved to more than one person

2. The PNR number issued must be unique in the entire system.

Measures to avoid deadlock in the above system.

1. Request grant of the resources must not form a cycle.

2. Time stamp_Based ordering can be used.

Q34. Explain working of the multiversion technique in the case of concurrency control, with the help of an example.

Ans. Multi-version concurrency control (MVCC) is a standard technique for avoiding conflicts between reads and writes of the same object. MVCC guarantees that each transaction sees a consistent view of the database by reading non-current data for objects modified by concurrent transactions. MVCC is a fairly common technique in database implementation. For example PostgreSQL uses MVCC.

For data-item x the database could keep the multiversion is the form of a set of triples costing the values, the tune entered and the time modified as shown below:

Variable: { { value, time entered, time modified, {   }, ...}

$$x : \{\{x_0, t_0, t_1,\}, \{x_1, t_1, t_2\}....... \{x_n, t_n . \text{tp3}\}$$

When a transactions needs to read a data-item such as x for which multiple version exit, the DBMS selects one of the version of the data-item. The value read by a transaction must be consistent with some serial execution of the transaction with a single version of the database. Thus the concurrency control problem is transferred into the selection of the correct version from the multiple version of a data item. With the multiversion technique, write operations can occur concurrently.

Example : Assume X1, X2, …, Xn are the version of a data item X created by a write operation of transactions. With each Xi a read_TS (read timestamp) and a write_TS (write timestamp) are associated.
read_TS(Xi) : The read timestamp of Xi is the largest of all the timestamps of transactions that have successfully read version Xi.
write_TS(Xi) : The write timestamp of Xi that wrote the value of version Xi. A new version of Xi is created only by a write operation.

Q35. Define triggers.
Ans. DBMS allow you to define procedures that are implicitly executed when an INSERT, UPDATE or DELETE statement is issued against the associated table. These types of procedures are called database trigger.
Use :
(i) And it data modification
(ii) Log event transparently
(iii) Enforce complex business rules
(iv) Desire column values automatically
(v) Implement complex security authorizations
(vi) Maintain replicate tables.

Q36. What are Triggers, Cursors and Alerts? What is their requirement in DBMS? How are Triggers different from stored procedures?
Ans. Trigger : A trigger is a statement that the system executes automatically as a side effect of a modification to the database.
To design a trigger mechanism, we must meet two requirements
(a) Specify when a trigger is to be executed: That is broken up into an event that causes the trigger to be checked and a condition that must be satisfied for trigger execution to proceed.
(b) Specify the actions to be taken when the trigger executes : The above model of triggers is referred to as the event condition action model for triggers. The database stores triggers as they are regular data, so that they are persistent

and are accessible to all database operations. Once trigger is stored in the database, the database system takes on responsibility of executing it whenever the specified events occurs and corresponding condition is satisfied.

Use: Triggers have many uses, such as implementing business rules, audit logging and even carrying out actions outside the database.

Trigger as supportive for integrity constraints : E.g., in a banking system, instead of allowing negative balances, the bank create a loan in the amount of overdraft and set balance to 0.

To do this,

Condition for executing trigger is: update to account relation that results in a negative value.

Actions to be taken are:

(i) Insert a new record/tuple/row in the load relation with
s[loan-number] = t[account-number]
s[branch-name] = t[branch-name]
s[amount] = -(t[balance])

(ii) Insert a new record/row/tuple in borrower relation with
u[customer-name] = t[customer-name]
u[loan-number] = t[loan-number]

(iii) Set t[Balance] to 0

Define in SQL

```
CREATE TRIGGER overdraft-trigger AFTER UPDATE OF balance ON
account
REFERENCING NEW ROW AS nrow
FOR EACH ROW
WHEN nrow.balance < 0
BEGIN ATOMIC
INSERT INTO loan values
        (nrow.account-number, nrow.branch_name, -(nrow.balance));
INSERT INTO borrower
        (select customer-name, account-number
        from depositor
        where nrow.account-number = depositor.account-number);
UPDATE ACCOUNT set balance = 0
        where account.account-number = nrow.account-number;
END
```

Cursor: A cursor basically is a pointer to a query result and is used to read attribute values of selected tuples into variables.

Use: Another important feature of PL/SQL is that it offers a mechanism to process query results in a tuple-oriented way, that is, one tuple at a time. For this, cursors are used. A cursor typically is used in combination with a loop construct such that each tuple read by the cursor can be processed individually.

Example: A cursor declaration specifies a set of tuples (as a query result) such that the tuples can be processed in a tuple-oriented way (i.e., one tuple at a time) using the **fetch** statement. A cursor declaration has the form

cursor <cursor name> [(<list of parameters>)] **is** <**select** statement>;

The cursor name is an undeclared identifier, not the name of any PL/SQL variable. A parameter has the form <parameter name> <parameter type>. Possible parameter types are **char, varchar2, number, date** and **Boolean** as well as corresponding subtypes such as **integer**.

Parameters are used to assign values to the variables that are given in the **select** statement.

Example: We want to retrieve the following attribute values from the table EMP in a tuple oriented way: the job title and name of those employees who have been hired after a given date and who have a manager working in a given department.

cursor employee cur (start date **date**, dno **number**) **is**
select JOB, ENAME **from** EMP E **where** HIREDATE > start date
and exists (**select** * **from** EMP
where E.MGR = EMPNO **and** DEPTNO = dno);

If (some) tuples selected by the cursor will be modified in the PL/SQL block, the clause **for update** [(<column(s)>)] has to be added at the end of the cursor declaration. In this case selected tuples are locked and cannot be accessed by other users until a **commit** has been issued.

Alert: The Database Alert sends an alert message with a description of the problem as a record to a SQL database. You can then use database tools to provide more advanced recording, sorting and reporting on your monitoring data.

Difference Between Trigger and stored procedure:

1. Stored Procedures are to be invoked/run explicitly while Trigger gets executed by system implicitly as side effect of some transaction.

2. Stored procedures are used to execute frequently required transactions while Triggers are used to provide Integrity constraints of database system

3. When users create a trigger used have to identify event and action of users trigger but when user create Stored Procedure user don’t identify event and action.

4. Trigger is run automatically if the event is occurred but Stored Procedure don't run automatically but user have to run it manually.

5. Within a trigger user can call specific Stored Procedure but within a Stored Procedure user can't call a trigger.

6. Stored Procedure can pass the parameters which is not a case with Triggers.

7. A Trigger can call the specific Stored Procedure in it but the reverse is not true.

Q37. How are remote back up recovery process helpful in transaction processing methods?

Ans. They increase the availability of the system. Also provides a situation through which we may avoid natural disaster or catastrophic failures.

Q38. Define log based recovery process.

Ans. All recovery process requires redundancy. Log based recovery process records a consistent state of database and all the transaction logs after that on a stable storage. In case of any failure the stable log and the database states are used to create the consistent database state.

Q39. Elaborate the dual paging problem.

Ans. When a page buffer under an operating system control is written back to disk swap space by operating system and is required by DBMS, this require two block transfer, hence the name dual paging problem.

DATABASE SECURITY

Q40. Write the syntax for 'Revoke Statement' that revokes with grant option.

Ans. REVOKE GRANT OPTION FOR <permissions>
ON <table>
FROM <user/role>

Q41. What does DENY command do?

Ans. It creates a 'NOT PERMITTED' condition in the database Access control.

Q42. The most secure database is found in ....

Ans. Second graded Bunkers.

Q43. ……………….. is the process of limiting Access to the Database Server.
Ans. Server Security

Q44. What are the different ways of preventing open access to the Internet?
Ans. Trusted IP address.
Server Account Disabling
Special Tools, Example: Real source by ISS.

Q45. Write the syntax for granting permission to alter database.
Ans. GRANT ALTER DATABASE TO Dinesh

Q46. On what systems does the security of a Database Management system depend?
Ans. Operating system,
Application that use the database.
Service that interact, and
The Web Server.

Q47. List the different types of security threats. Explain three major defence mechanisms required to be built into the database management system.
Ans. Security threats can be categorised into two categories :
1. Accidental Security Threats – It includes :
(i) Improper Recovery Procedures
(ii) Concurrent usage anomalies.
(iii) System error.
(iv) Improper authorisation .
(v) Hardware failures.

2. Malicious or Intentional Security Threats – It includes :
(i) An operator or system programmer can intentionally
(ii) bypass normal security mechanisms.
(iii) An unauthorized user can get access to secure terminal.
(iv) Authorized users could pass on sensitive information on duress.

<u>Defence Mechanisms</u>

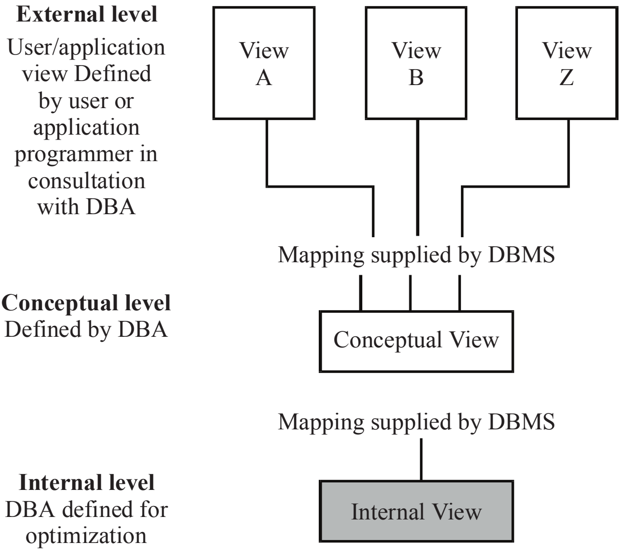
1. Human factors – It is the outer most level which encompass the ethical, legal, and social environments.

2. Physical security - physical security mechanisms include appropriate locks and keys and entry logs to computing facility and terminals.

3. Administrative controls – These determine what information will be accessible to what class of users, and the type of access that will be allowed to this class.

Q48. How can views be useful in enforcing security? What are the types of updates that are allowed through views?

Ans. View Mechanism – The architecture of most of the database models is divided into three levels: The external level, the conceptual level, and the internal level. How we see the data at these levels is called view at the particular level. How these all views are mapped with each other is called view mechanism.



External or User view – Only that portion of database is supplied to the user that is relevant to that particular user. External view is defined by the DBA by considering the requirements and authorities of the user. External view is generated by extracting the objects from conceptual view.

Conceptual View – Conceptual view represents the entire database. It internal methods of deriving the objects in conceptual view from the internal view. The data description at this level is in a format, independent of its physical representation.

It also contains methods to check data integrity and consistency.

Internal View ⇒ Indicates how the data will be stored. It describes the data structures and access methods.

Example which illustrate the use of view to enforce security :-
Let's create a view for use by the departmental secretary consisting of the attributes. Employee-name, Room and Phone-no.
The tuples accessible are limited to the employees in the secretary's department.
Create view EMP-ADDRESSES (Name, Room - No, Phone) as
(Select (e.Employee_Name, e.Room, e.Phone-No)
From EMPLOYEE e
where e.Department = (Select (department) from HEAD
where Secretary = 'secretary - name')).
Having created this view, the secretary is granted appropriate access rights to any tuple of this relation and is allowed to update Room - No, phone by means of the appropriate grant statement.
Update are allowed through a view defined using a simple query involving a single base relation and containing either the primary key or a candidate key of the base relation.

Q49. Assuming that you are the data security administrator of a public sector bank, what are the different security and privacy measures that you will propose for its customer's data?
Ans. Here the question has a situation of a data security administration of a bank. The Data security Administrator has to implement security & privacy measures on Bank database. This is important because a large number of customers are allowed to access the database to prevent unauthorized access of data of one user from others. The implementation of various security & privacy measures on database is necessary to save the customers data from malicious manipulation. Keeping in view the need for security privacy Measures, we can use two types of security mechanisms:
(a) Security Mechanisms involving Access protection, user accounts & database

audits.

(b) Security mechanisms involving granting & Revoking of privileges.

Q50. How can you recover from media failure on which your database was stored? Describe the mechanism of such recovery.

Ans. Media failure is the biggest threat to your data. A media failure is a physical problem that occurs when a computer unsuccessfully attempts to read from or write to a file necessary to operate the database. Common types of media problems include:

- A disk drive that holds one of the database files experiences a head crash.
- A datafile, online or archieved redo log, or control file is accidentally deleted, overwritten, or corrupted.

The technique you use to recover from media failure of a database file depends heavily on the type of media failure that occurred. For example, the strategy you use to recover from a corrupted datafile is different from the strategy for recovering from the loss of the control file.

The basic steps for media recovery are:

- Determine which files to recover.
- Determine the type of media recovery required: complete or incomplete, open database or closed database.
- Restore backups or copies of necessary files: datafiles, control files, and the archived redo logs necessary to recover the datafiles.

Note :

If you do not have a backup, then you can still perform recovery if you have the necessary redo logs and the control file contains the name of the damaged file. If you cannot restore a file to its original location, then you must relocate the restored file and rename the file in the control file.

- Apply redo records to recover the datafiles. (When using Recovery Manager, apply redo records or incremental backups, or both.)
- Reopen the database. If you perform incomplete recovery or restore a backup control file, then you must open the database with the `RESETLOGS` option.

The following techniques are used to recover from storage media failure.

(i) Data Base Backup – The whole database and the log are periodically copied on to a cheap storage media, like tape in case of failure, the latest backup copy is reloaded from the tape to the worky disk.

(ii) Volatile Storage – For avoiding failure of volatile storage media, we should provide continuous power supply by using batteries.

(iii) Mirroring – A shadow of all transaction is maintained in a separate disk, so in case of failure, immediately service can be provided by making mirror disk as the primary disk.

(iv) Use of RAID Technology – A redundancy array of disk is maintained to recover from such catastrophic failure.

(v) System Log Backup – To avoid losing all the effects of transactions that have been executed since the last backup, backup of the system log is taken at more regular intervals then full database backup.

Q51. What are the several forms of authorising database users ? Can a user pass his authorisation to another user ? How can you withdraw an authorisation ?

Ans. Keeping in view the need for security privacy measures, we can use two types of security mechanisms :-

(a) Security Mechanism involving access protection, user accounts and database audits.

(b) Security mechanisms involving granting and revoking of privileges.

(a) Security Mechanism involving access protection, user accounts and database audits – Following measures are adopted under this category :

(i) Account Number and Password – Whenever a person or a group of person needs to access a database system, the DBA creates a new account number and password for the user if there is a legitimate need to access the database. The user must log into the DBMS by entering the account number and password whenever database access is needed. The DBMS checks that the account number and password are valid. If they are, the user is permitted to use the DBMS and to access the database.

(ii) Log-in Session – The database system must also keep track of all operations on the database that are applied by a certain user throughout each database interactions that a user performs from the time of logging to the time of logging-off. It is particularly important to keep track of updated operations that are applied to the database so that if the database if tampered with, the DBA can find out which user did the tampering.

(b) Security mechanisms involving granting and revoking of privileges: The typical method of enforcing discretionary access control in a database system is based on granting and revoking of privileges.

Informally, there are two levels of assigning privileges for the user of database system :

(i) The Account Level – At this level the DBA Specifies privileges to a particular account. The privileges can be Insert, Delete etc.

(ii) The Relational (table) level – Privileges at the relational level specifies for each user the individual relation on which each type of command can be applied. The relation can be base relation or virtual relation (view). Some privileges also refer to individual columns of relation.

The granting and revoking privileges generally follows an authorization model for discretionary privileges known as access matrix model where the rows of a matrix M represent subjects (users, accounts, programs) and the columns represents objects (relations, records, columns, views and operations). Each position (i, j) in the matrix represents the type of privileges (read, write, update) that subject i holds on object j.

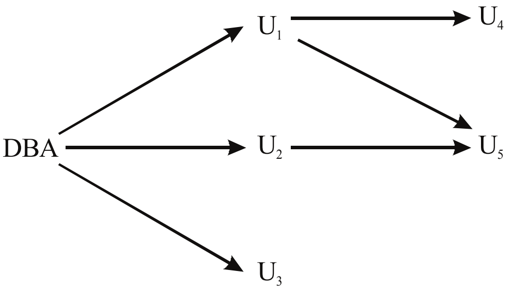
To control the granting and revoking of relation privileges each relation r in a database is assigned an owner account which is typically the account that was used when the relation was created at first place. The owner of the relation is given all privileges on that relation. The owner account holder can pass privileges on any of the owned relation to other users, by granting privileges to other accounts.

Passing Authorization from one user to another ⇒ PROPAGATE ACCESS CONTROL is an additional right that determines if this subject is allowed to propagate the right over the object of other subjects. Thus a subject S may be assigned an access right R over an object O and in addition the right grant this access right to another subject. The right R may include the right the PROPAGATE ACCESS CONTROL.

When the subject user has the PROPAGATE ACCESS CONTROL right over an object he can pass all or part of his right to another subject, for instance, to another user including PROPAGATE ACCESS CONTROL right. This leads to Authorization Grant Tree.

Authorization Grant Tree ⇒ The needs of authorization Grant Tree are the users. The Tree includes an edges $U_i \rightarrow\!| \, U_j$ if user U_i grant rights to user U_j the root of the graph is database Administrator.

In the following figure.



User U_5 is granted authorization by both U_1 & U_2, U_4 is grant authorization by only user U_1.

Withdrawal of Authorization ⇒ Authorization can be withdrawn by the DBA, suppose a Database Administrator decides to revoke the authorization of user U_1, since U_4 has authorization through U_1, that authorization should also be revoked as well.
Since user U_5 has been granted authorization by both U_1 & U_5 since DBA doesn't revoke the rights from U_2, user U_5 retains the rights.

Q52. Write two ways of protecting against inference attacks.
Ans. a) Control to the Queries.
b) Control individual items that are being displayed.

Q53. With .... sensitive data values are not provided; the query is rejected without response.
Ans. Suppression.

Q54. Write a sequence of queries that would disclose the name of the person who scored the highest marks.
Ans.

Subject Maximum Marks = 100	Neha	Richa	Neelam
Maths	50	70	90
Science	60	80	50
Total	110	150	140

Ans. The highest marks can be found as:
Find the total marks
Find the max. marks
Find the names of student whose total is the set of maximum marks.

Q55. .... & .... currently relies on the logon security present in the Web Server.
Ans. Sybase and Informix

Q56. For user Authentication, Microsoft provides its .... mechanism.
Ans. Tried-and-true challenge/response .

Q57. What is multilevel Sensitive Database?
Ans. The database with more than one i.e., various levels of security.

Q58. Name the different techniques for multilevel security.
Ans. i) Partitioning
ii) Encryption

Q59. .... while providing SSL & S-HTTP security, its using java as a basic component of its security model.
Ans. Oracle

Q60. What is Audit trail? Give four benefits provided by Audit trail to DBMS. [June-07, Q5(b)]
Ans. An audit trail tracks and reports activities around confidential data. Many companies have not realised the potential amount or risk associated with sensitive information within database unless they run an internal audit which details who has access to sensitive data and have assessed it. Consider the situation that a DBA who has complete control of database information may conduct a security breach, with respect to business details and financial information. This will cause tremendous loss to the company.

(1) In such a situation database audit helps in locating the source of the problem. The database audit process involves a review of log files to find and examine all reads and writes to database items during a specific time period, to ascertain mischief if any, banking database is one such database which contains very critical data and should have the security feature of auditing.

(2) An audit trail is a log that is used for the purpose of security auditing

(3) Database auditing is one of the essential requirements for security especially, for companies in possession of critical data. Such companies should define their auditing strategy based on their knowledge of the application or databases activity. Auditing need not be of the type “all or nothing.

(4) One must do intelligent auditing to save time and reduce performance concerns. This also limits the volume of logs and also causes more critical security events to be highlighted.

Q61. What do you mean by Multilevel Security? Discuss the techniques involved in support of multilevel security. [June-07, Q5(c)]

Ans. In certain applications data items can be classified under various levels of security. Some such security levels may be Secret, Classified, etc. The database system that supports such security features are known as Multilevel Sensitive Databases. Such systems may have the following three security features:

- Security of different objects may be different from the other attribute values of that tuple or security may be different from the other values of the attributes.
- Thus, every individual element is the micro items for security.
- In addition, security against statistical queries may also need to be provided.
- A number of security level needs to be defined for such security.

There are many techniques to support multi-level security. Two important methods are:

(i) Partitioning

In this approach the original database is divided into partitions. Each of the partitions has a different level of security.

However, the major drawback of this approach is that the database looses the advantage of a relation.

(ii) Encryption

Encryption of critical information may help in maintaining security as a user who accidentally receives them cannot interpret the data. Such a scheme may be used when a user password file is implemented in the database design. However, the simple encryption algorithms are not secure, also since data is available in the database it may also be obtained in response to a query.

There are may more schemes to handle such multi-level database security.

Q62. An airline reservation allows many customers to book tickets simultaneously. What are the basic concurrency related problems that you may encounter, in case no concurrency control mechanism is in place? What is the solution proposed by you to overcome the concurrency related problem as above?

Ans. In airline reservation system a number of transactions are running and accessing the database. Transactions are reading and modifying data item concurrently. In this environment we will have to tackle following problems:

(1) Lost update problem :- Consider a transaction T_1 changes availability of seats initially 30 to 29. Below 30 seats tariff changes T_2 reserves two more seats with the changed tariff. Assume T_2 runs earlier than T_1 than T_2 will change the tariff and T_1 will lose it's advantage.

(2) inconsistent read problem :- Arises when only one transaction modifies a set of data while that set of data is being used by other transactions. Consider transaction T_1 change available seats to a level where tariff changes while T_2 calculate the tariff. It means while T_1 has changed the available seats still T_2 is showing old tariff, creating confusion.

(3) The phantom phenomenon :- This problem arises when a transactions is using a record from a table. That particular record is locked but table is not locked it means another transaction T_2 may add new records to the table that will tell a lie about the factor stated by T_1 There new records will be called a phantom records.

Now the solutions to these problems can be concurrency control mechanisms. These mechanisms can be broadly categorized into two categories.

(i) Locking

(ii) Time-stamping

Locking : Is a concurrency control mechanism which locks the data item to prevent multiple transactions from accessing the item concurrently. A lock is a variable associated with each data item that describes the status (busy or free) with respect to possible operations that can be applied to it.

In time stamp based methods, a serial order is created among the concurrent transactions by assigning to each transaction a unique non-decreasing number. The usual value assigned to each transaction is the system clock value at the start of the transaction, hence the name time stamp ordering.

The value can then be used in deciding the order in which the conflict between two transactions in resolved. A transaction with a smaller time-stamp value is considered to be an 'order' transaction than another transaction with a larger time-stamp value.

Q63. Design a database security scheme for a library issue and return system.
(Hint: you may define security levels for database access and external schema)

Ans. A library issue & return system will issue the books to the valid members & also take return of the books.

Here this system will keep information about members. Their renewals, number of books, due date of return etc.

The security mechanism is needed for such database system so that any invalid member could not take the book.

Besides any member whose membership needs to be renewed could not take the books. Issued books should be returned on due dates and if they are kept after the due date of return by the members, then fine should be charged.

The database security mechanisms can be divided into two categories :-

(i) Discretionary Security Mechanisms used to grant privileges to users including the capability to access specific datafiles records in a specified mode such as read, insert, delete, update.

(ii) Mandatory Security Mechanisms are used to enforce multilevel security by classifying data & users into various security classes. Typical security classes are:

- TS (Top Secret)
- Secret (S)
- Confidential (C)
- Unclassified (U)

Besides the above security classes, four levels of defence are generally recognized for database security :

(a) Human Factors
(b) Physical Security
(c) Administrative Control
(d) DBMS & O/S Security Mechanisms

(a) Human Factors : It encompasses ethical, legal and social environments. An organization depends on these factors to provide a certain degree of protection. An organization usually performs some type of clearance procedure for a personal who is going to be dealing with the sensitive information, including that contained in the database. This clearence procedure can be very informal one, in the form of reliability & trust that an employee has entered in the eye of management or the clearence procedure could be formal one.

(b) Physical Security : Physical Security mechanisms include appropriate locks & keys and entry logs to computing facility & terminals.

(c) Administrative Control : Administrative control includes security and access control policies that determine what information will be accessible to what class of users & the type of access that will be allowed to this class.

(d) DBMS & O/S Security Mechanisms : The database depends upon some of the protection features of the O/S for security. Among the O/S features required are:-

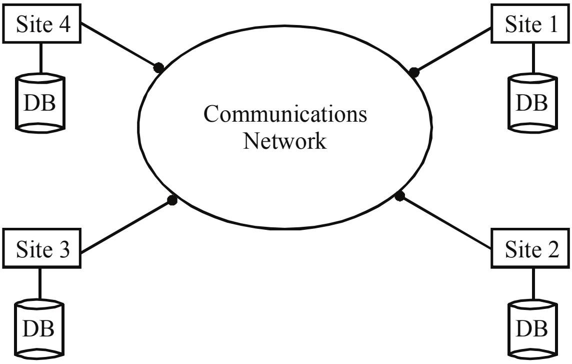
The proper mechanisms for the identification and verification of users
Each user is assigned an account number and a password. The O/S ensures that access to the system is denied unless the account number and password are valid. In addition, the DBMS could also require a number & password before allowing the users to perform any database operations.

The protection of data & program both in primary & secondary memories
This is usually done by the O/S to avoid direct access to the data in primary memory or to online files.

Q64. What is a distributed database management system? What are four advantages and two disadvantages of a DDBMS? Why must a distributed database system be relational?

Ans. A **distributed database** is a database that is under the control of a central database management system in which storage devices are not all attached to a common CPU. It may be stored in multiple computers located in the same physical location, or may be dispersed over a network of interconnected computers.

Collections of data (e.g.: in a database) can be distributed across multiple physical locations. A distributed database is distributed into separate partitions / fragments. Each partition / fragment of a distributed database may be replicated (i.e.: redundant fallovers, RAID like). Besides distributed database replication and fragmentation, there are many other distributed database design technologies. For example, local autonomy, synchronous and asynchronous distributed database technologies. These technologies' implementation can and does definitely depend on the needs of the business and the sensitivity/confidentiality of the data to be stored in the database. And hence the price the business is willing to spend on ensuring data security, consistency and integrity.



Distributed database Architecture

Advantages of distributed databases:

- **Reflects organizational structure –** Database fragments are located in the departments they relate to.
- **Local autonomy –** A department can control the data about them (as they are the ones familiar with it.)
- **Improved availability –** A fault in one database system will only affect one fragment, instead of the entire database.
- **Improved performance –** Data is located near the site of greatest demand, and the database systems themselves are parallellized, allowing load on the databases to be balanced among servers. (A high load on one module of the database won't affect other modules of the database in a distributed database.)
- **Economics –** It costs less to create a network of smaller computers with the power of a single large computer.
- **Modularity –** Systems can be modified, added and removed from the distributed database without affecting other modules (systems).

Disadvantages of distributed databases :

- **Complexity –** Extra work must be done by the DBA's to ensure that the distributed nature of the system is transparent. Extra work must also be done to maintain multiple disparate systems, instead of one big one. Extra database design work must also be done to account for the disconnected nature of the database – for example, joins become prohibitively expensive when performed across multiple systems.

- **Economics –** Increased complexity and a more extensive infrastructure means extra labour costs.
- **Security –** Remote database fragments must be secured, and they are not centralized so the remote sites must be secured as well. The infrastructure must also be secured (e.g.: by encrypting the network links between remote sites).
- **Difficult to maintain integrity –** In a distributed database enforcing integrity over a network may require too much networking resources to be feasible.
- **Inexperience –** Distributed databases are difficult to work with, and as a young field there is not much readily available experience on proper practice.

Q65. How does the architecture of DDBMS differ from that of a centralised DBMS?

Ans. Reference architecture shows data distribution, which is an extension of the ANSI/SPARC three level architecture for a centralised DBMS. The reference architecture consists of the following :

- A set of global external schemas to describe the external view for the entire DBMS,
- A global conceptual schema containing the definition of entities, relationships, constraints and security and integrity information for the entire system,
- A fragmentation and allocation schema, and
- A set of schemas for each local DBMS a per ANSI/SPARC architecture.

Q66. What additional functions does a Distributed Database System have over a centralised database system.

Ans. (i) Access to remote sites, transfer of queries and data through a network,
(ii) A System catalogue for a distributed system having data distribution details,
(iii) Distributed query processing including query optimisation and remote data access,
(iv) Concurrency control to maintain consistency of replicated sites,
(v) Proper security control for proper authorization/access privileges of the distributed data, and
(vi) Recovery services to check the failures of individual sites and communication links.

Q67. List the basic objectives of concurrency control mechanism in a distributed database.

Ans. The basic objectivs of concurrency control mechanism in a distributed database are :

(1) to ensure that various data items and their replications are in a consistent state, when concurrent transactions are going on at the same time, and

(2) to ensure that all the concurrent transactions get completed in finite time.

The objectives as above should be fulfilled by the concurrency control mechanism under the following:

- failure of a site or communication link,
- strict performance guidelines supporting parallelism of queries, and minimal storage and computational overheads, and
- dealing with the communication delays of the networked environment.

Despite the above considerations it should support atomicity.

Q68. Discuss the problem faced by distributed database system.

Ans. A distributed database system has to deal with the following two types of problems that occur due to distributed concurrent transactions:

- Concurrent transactions related problems that occur in a centralised database, (namely, lost updates, read uncommitted and inconsistent analysis).
- In addition, concurrency related problems arise due to data distribution and replication. For example, the need to update a data item that is located at multiple locations. Such data item require to be updated at all the locations or on none at all.

Q69. Why can we not detect deadlocks in a distributed database with a simple wait for graph?

Ans. Same set of transactions at different sites may have locked certain resources for which they may be waiting although locally it may seem that T_1 is waiting for T_2 on site 1 and T_2 is waiting for T_1 at site 2, but only on combining these two we know that both transactions cannot proceed. Hence, a deadlock occurs.

Q70 What are the issues of query processing in a distributed database?

Ans. A query may be answered by more than one site, in such a case, we would, require that the communication cost among sites should be minimum. You may do data processing at the participant this communication cost.

Q71. What are the serialisability requirements for a distributed database?

Ans. If transaction are distributed at different sites then they should be executed in exactly the same sequence at all the sites.

Q72. Define 2PC.

Ans. In a DDBMS since a transaction is being executed at more than one site, the commit protocol may be divided into two phases. The basic objective of this two-phase commit protocol is to make sure that a transaction and all its sub-transactions commit together or do not commit at all. A two-phase commit protocol ensures integrity and handles remote prcedure calls and triggers.

A 2PC consists of two phases :

- voting phase and
- decision phase.

For each transaction, there is a coordinator who controls the 2 PC. The sub-transasction sites act as participants. The logical steps of 2PC are :

- coordinator asks its participants whether they are prepared to commit the transaction, or not.
- participants may votes to commit, abort or fail to respond within a time-out period,
- the coordinator makes a decision to commit if all participants say commit or abort even if one participant says so or, even if, one of the participant does not respond.
- global decision is communicated to all the participants.
- if a participant votes to abort, then it is free to abort the transaction immediately as the result of this voting will definitely be abort. This is known as a **unilateral** abort,
- if a participant votes to commit, then it waits for the coordinator to send the *global commit* or *global abort* message,
- each participant maintains its local log so that it can commit or rollback the local sub-transaction reliably. In 2PC participants wait for messages from other sites, thus, unnecessarily blocking the participating processes, even though a system of timeouts is used.

Q73. Discuss the need for 3PC. Also define 2PC.

Ans. The 2PC is a good protocol to be implemented in a DDBMS. However, it has a small disadvantage-it can block participants in certain circumstances. For example, processes that encounter a timeout after voting COMMIT, but before receiving the global commit or abort message from the coordinator, are waiting for the message and doing nothing or in other words are *blocked*. Practically, the probability of blocking is very low for most existing 2PC implementations. Yet an alternative non-blocking protocol- the **three-phase commit (3PC)** protocol exits. This protocol does not block the participants on site failures, except for the case when all sites fail. Thus, the basic conditions that this protocol requires are :

- no network partitioning,
- at least one available site,
- at most *K* failed sites (called *K-resilient).*

Thus, the basic objective of the 3PC is to remove the blocking period for the participants who have voted COMMIT, and are thus, waiting for the global abort/commit message from the coordinator. This objective is met by the 3PC by adding another phase-the third phase. This phase called **pre-commit** is introduced between the voting phase and the global decision phase.

If a coordinator receives all the commit votes from the participants, it issues a global PRE-COMMIT message for the participants. A participant on receiving the global pre commit message knows that this transaction is going to commit definitely.

The coordinator on receiving the acknowledgement of PRE-COMMIT message from all the participants issues the global COMMIT. A Global ABORT is still handled the same way as that of in 2PC.

Q74. Why do we need replication servers?

Ans. We need replication servers because they help in:

- High availability
- Low communication overheads due to local data availability,
- Disaster recovery, and
- Distribution of functions of a system across organisation, etc.

Q75. What are the steps of two phase commit protocol?

Ans.

COORDINATOR	PARTICIPANTS
Prepare	Listen to Coordinator
Wait for vote	Vote
Collect vote if all COMMIT	Wait for Global Decision
Send Global COMMIT and ABORT to all participant	Receive decision acts accordingly and send acknowledgement
Waits for acknowledgement from all participant	TRANSACTION OVER
TRANSACTION OVER	

Chapter-6

ENHANCED DATABASE MODELS

Q1. What are Object Oriented Design (OOD) techniques? Explain the OOD process.

Ans. OOD techniques are useful for development of large, complex systems. It has been observed that large projects developed using OOD techniques resulted in a 30% reduction in development times and a 20% reduction in development staff effort, as compared to similarly sized projects using traditional software development techniques.

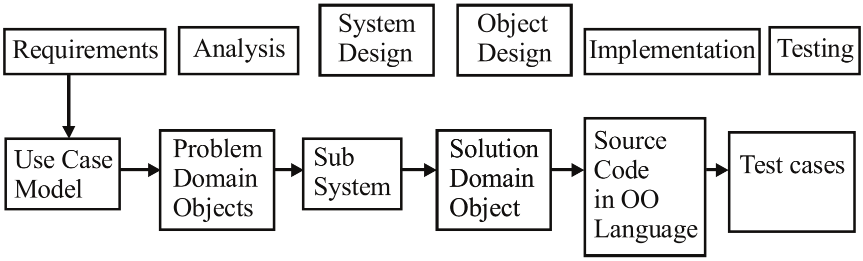
Although object oriented methods have roots that are strongly anchored back in the 60s, structured and functional methods were the first to be used. This is not very uncommon, since functional methods are inspired directly by computer architecture (a proven domain, well known to computer scientists). The separation of data and program as exists physically in the hardware was translated into functional methods. This is the reason why computer scientists got into the habit of thinking in terms of system functions. Later it was felt that hardware should act as the servant of the software that is executed on it rather than imposing architectural constraints on system development process. Moving from a functional approach to an object oriented one requires a translation of the functional model elements into object model elements, which is far from being straightforward or natural. Indeed, there is no direct relationship between the two sets, and it is, therefore, necessary to break the model elements from one approach in order to create model element fragments that can be used by the other. In the initial phases of OO design a mixed approach was followed computer scientist tend to use functional approach in analysis phase and OO approach in design phase. The combination of a functional approach for analysis and an object-oriented approach for design and implementation does not need to exist today, as modern object-oriented methods cover the full software lifecycle.

An object-oriented design process :

1. Define the context and modes of use of the system

2. Designs the system architecture
3. Identifies the principal system objects
4. Identifies concurrency in the problem
5. Handling boundary conditions
6. Develops design models
7. Specifies object interfaces

Object oriented design is concerned with developing OO model of software systems to implement the identified requirement during the analysis phase.



Software life cycle activity

Software developers, data base administrators (DBAs), need to be familiar with the basic concepts of object-orientation. The object-oriented (OO) paradigm is a development strategy based on the concept that systems should be built from a collection of reusable components called objects. Instead of separating data and functionality as is done in the structured paradigm, objects encompass both. While the object oriented paradigm sounds similar to the structured paradigm, as you will see in this course material it is actually quite different. A common mistake that many experienced developers make is to applying similar software-engineering principles to OO design. To succeed one must recognize that the OO approach is different than the structured approach.

Q2. Differentiate between Relational DB and OO Databases.
Ans.

Relational Databases	Object-Oriented Databases
(1) Based on mathematical principles called relational algebra.	(1) Supports all fundamental object modeling concepts: Classes, Attributes, Methods, Associations, Inheritance
(2) Data are represented by a two dimensional table with columns and rows	(2) Support for complex objects
(3) Implements standard query language called SQL	(3) Provides for mapping an object model to an OO-database
(4) Most RDBMS supports various constraints, like referential integrity.	(4) Determine which objects are persistent
	(5) Perform normal requirement analysis and object design
	(6) Create single attribute indices to reduce performance bottlenecks.

Q3. Give advantages and disadvantages of OO Database.

Ans.

Advantages of OO Database	Disadvantages OO Database
(1) Supports all fundamental object modeling concepts like Classes, Attributes, Methods, Associations, Inheritance. (2) Maps an object model to an OO-databases. (3) Determine which objects are Persistent (4) Performs normal requirement analysis and object design (5) Creates single attribute indices to reduce performance bottlenecks. (6) Supports complex objects. (7) Extensibility of data types. (8) Improves performance with efficient caching. (9) Versioning (10) Faster development and easy maintenance through inheritance and reusability.	(1) Strong opposition from the established players of relational database. (2) Lacks rigorous theoretical foundation. (3) Retrogressive to the old pointer systems. (4) Lacks standard ad hoc query language like SQL. (5) Lack of standards affects OO database design (6) Lack of business data design and management tools (7) Steep learning curve.

Q4. What is the need for object – oriented databases?

Ans. The object oriented databases are needed for:

- Representing complex types.
- Representing inheritance, polymorphism
- Representing highly interrelated information

- Providing object oriented solution to databases bringing them closer to OOP.

Q5. Represent and address using SQL that has a method for locating pin-code information.

Ans.

```
CREATE TYPE Addrtype As
          houseNo      CHAR (8),
          street       CHAR (10),
          colony       CHAR (10),
          state        CHAR (8),
          pincode      CHAR (6),
);
METHOD pin ( ) RETURNS CHAR (6);
CREATE METHOD pin ( ) RETURNS CHAR (6);
FOR Addrtype
BEGIN
. . . . .
END
```

Q6. How will you represent a complex data type?

Ans. Primarily by representing it as a single attribute. All its components should also be referenced separately.

Q7. How can you establish a relationship with multiple tables?

Ans. The relationship can be established with multiple tables by specifying the keyword "SCOPE. For example:

```
Create table my library
{
          mybook REF (Book) SCOPE library;
          myStudent REF (Student) SCOPE student;
          mySupplier REF (Supplier) SCOPE supplier;
};
```

Q8. Create a table using the type created in question 3 above.

Ans.

```
CRATE TABLE address OF Addrtype
(
          REF IS added SYSTEM GENERATED,
          PRIMARY KEY (houseNo, pincode)
};
```

Q9. List the major considerations while converting ODL designs into relational designs.

Ans. The major considerations while converting ODL designs into relational designs are as follows :

(a) It is not essential to declare keys for a class in ODL but in Relational design now attributes have to be created in order for it to work as a key.

(b) Attributes in ODL could be declared as non-atomic whereas, in Relational design, they have to be converted into atomic attributes.

(c) Methods could be part of design in ODL but, they can not be directly converted into relational schema although, the SQL supports it, as it is not the property of a relational schema.

(d) Relationships are defined in inverse pairs for ODL but, in case of relational design, only one pair is defined.

For example, for the book class schema the relation is :

Book (ISBNNO, TITLE, CATEGORY, fauthor, sauthor, tauthor)

Thus, the ODL has been created with the features required to create an object oriented database in OODBMS.

Q10. What is OQL?

Ans. Object Query Language (OQL) is a standard query language which takes high-level, declarative programming of SQL and object-oriented features of OOPs.

Q11. Create a class staff using ODL that also in the Book class which is given below

```
class Book
{
        attribute string ISBNNO;
        attribute string TITLE;
        attribute enum CATEGORY
           {text, reference, journal} BOOKTYPE;
        attribute struct AUTHORS
           {string fauthor, string sauthor, string tauthor}
        AUTHORLIST;
};
```

Ans.
```
class staff
{
           attribute string STAFF_ID;
```

```
        attribute string STAFF_NAME;
        attribute string DESIGNATION;
        relationship set <Book> issues
        inverse Book : : issuedto;
};
```

Q12. Find the list of books that have been issued to "Dinesh".

Ans. SELECT DISTINCT b.TITLE
FROM BOOK b
WHERE b.issuedto.NAME = "Dinesh"

Q13. What modifications would be needed in the Book class because of the table created by the above query?

Ans. The Book class needs to represent the relationship that is with the staff class.

This would be added to it by using the following commands:

```
RELATIONSHIP SET < Staff > issuedto
        INVERSE :: issue Staff
```

Q14. What is persistency? Explain with an example, how persistent data are identified.

Ans. All computer programs operate on data. If you require data that persists beyond the lifetime of a single program execution, then you will need to use a permanent data store. There are various reasons for waiting persistent data:

- Persistent data allows the same program to resume processing at a later data.
- Data stores are often useful for historical or archival purposes.
- There are several approaches of providing past steno data services: Files, special purpose hardware some new object oriented data bases are especially convenient in their data storage services. To a large extent, such as advanced object oriented data bases looks like an object oriented language, except that the orthogonal property of persistence may be specified for data structures.

Example:

```
define class department:
type      set (department);
operations add dept (d: department): boolean;
/* adds a department to the department set object */
```

```
remove _ dept (d: department): boolean;
/* removes a department from the department set object */
create _ dept _ set: department set;
destroy_dept_set: boolean;
end department set
……….
persistent name All departments: Departments;

/*All departments is a persistent named object of type department set */
d:= create_dept;
….. /* create a new department object in variable d */
b:= All departments.add_dept(d);
/* make d: persistent by adding it to the persistent set all department */
……….
```

Q15. What is XML?

Ans. XML stands for Extensible Markup language. It is used to describe documents and data in standardised, text-based format, easily transportable *via* standard Internet protocols, XML, is based on the *mother* of all markup languages-Standard Generalised Markup Language (SGML).

SGML is remarkable inspiration and basis for all modern markup languages. The first popular adaptation in SGML was HTML, primarily designed as a common language for sharing technical documents. The advent of the Internet facilitated document exchange, but not document display. Hypertext Markup Language (HTML) standardises the description of document layout and display, and is an integral part of every Web site today.

Although SGML was a good format for document sharing, and HTML was a good language for describing the document layout in a standardised way, there was no standardised way of describing and sharing data that was stored in the document. For example, an HTML page might have a body that contains a listing of today's share prices. HTML can structure the data using tables, colour etc., once they are rendered as HTML; they no longer are individual pieces of data to extract the top ten shares. You may have to do a lot of processing.

Thus, there was a need for a tag-based markup language standard that could describe data more effectively than HTML, while still using the very popular and standardised HTTP over the Internet. Therefore, in 1998 the World Wide Web Consortium (W3C) came up with the first Extensible Markup Language (XML) Recommendations. Now, the XML (eXtended Markup Language) has emerged as the standard for structuring and exchanging data over the Web.

Please note : XML is case sensitive whereas HTML is not. So, you may see that XML and databases have something in common.

Q16. Define DTD.

Ans. A DTD is used to define the syntax and grammar of a document, that is, it defines the meaning of the document elements. XML defines a set of key words, rules, data types, etc to define the permissible structure of XML documents. In other words, we can say that you use the DTD grammar to define the grammar of your XML documents. The form of DTD is :

<! DOCTYPE name >

Or

<! DOCTYPE name [a_dtd_definition_or_declaration]>

The name, while not necessarily the document name, must be the same name as that of the document root node.

The second point of interest with DOCTYPE is that after the name you can declare your Document Type Definition (DTD), the assembling instruction for the document.

Q17. What is semi-structured data?

Ans. Semi-structured data is that has some structure, but the structure may not be rigid, regular or complete and generally the data does not conform to a fixed schema. Sometimes the term schema-less or self-describing is used to describe such data.

Q18. What is XML? How does XML compare to SGML and HTML?

Ans. Most document on Web are currently stored and transmitted in HTML. One strength of HTML is its simplicity. However, it may be one of its weaknesses with the growing needs of users who want HTML document to be more attractive and dynamic. XML is a restricted version of SGML, designed especially for Web documents. SGML defines the structure of the document (DTD), and text separately. By giving documents a separately defined structure, and by giving web page designers ability to define custom structures, SGML has and provides extremely powerful document management system but has not been widely accepted as it is very complex. XML attempts to provide a similar function to SGML, but is less complex. XML retains the key SGML advantages of extensibility, structure, and validation. XML cannot replace HTML.

Q19. Why is XML case sensitive, whereas SGML and HTML are not?

Ans. XML is designed to work with applications that might not be case sensitive and in which the case folding (the conversion to just one case) cannot be predicted. Thus, to avoid making unsafe assumptions, XML takes the safest route and opts for case sensitivity.

Q20. Why is it possible to define your own tags in XML but not in HTML?

Ans. In HTML, the tags tell the browser what to do to the text between them. When a tag is encountered, the browser interprets it and displays the text in the proper form. If the browser does not understand a tag, it ignores it. In XML, the interpretation of tags is not the responsibility of the browser. It is the programmer who defines the tags through DTD or Schema.

Q21. Is it easier to process XML than HTML?

Ans. Yes, for two reasons. The first is that you normally model your data better with XML than with HTML. You can capture the hierarchical structure giving meaning to your information. The second reason is that XML files are well formed. You have a much better idea what comes in the data stream. HTML files, on the other hand, can take many forms and appearances.

Q22. Discuss the advantages of XML?

Ans.

1. Simplicity

2. Open standard and platform/vendor-independent

3. Extensibility

4. Reuse

5. Separation of content and presentation

6. Improved load balancing

7. Support for integration of data from multiple sources

8. Ability to describe data from a wide variety of applications.

Q23. Which three attributes can appear in an XML declaration?

Ans. The attributes you can use in an XML document are: version (required; the XML version), encoding (optional; the character encoding), and standalone (optional; “yes” if the document does not refer to any external document or entities, “no” otherwise).

Q24. What is an XML element? What is an XML attribute?

Ans. An XML element is the basic data-holding construct in an XML document. It starts with an opening tag, can contain text or other elements, and ends with a closing tag, like this: <greeting> hi </greeting>. An attribute gives you more information, and is always assigned a value in XML. Here’s how you might add an attribute named language to this element:
<greeting language= “en”>hi</greeting>.

Q25. What are two XML constructs that let you specify an XML document’s syntax so that it can be checked for validity?

Ans. Document Type Definitions (DTDs) and XML schemas.

Q26. What are some of the requirements for an XML document to be well-formed?

Ans. 1. An XML document must contain one or more elements.
2. One element, the root element, must contain all the other elements.
3. Each element must nest inside any enclosing elements correctly.

Q27. What is the difference between a well formed XML document and a valid XML Document?

Ans. XML document that conforms to structural and notational rules of XML is considered well-formed. XML Validating processor checks that an XML document is well-formed and conforms to a DTD, in which case the XML document is considered valid. A well formed xml document may not be a valid document.

Q28. Why is the following XML document not well-formed?

```
<?xml version = “.10” standalone = “yes”?>
<employee>
<name> dinesh </name>
<position> Professor</position>
</employee>
<employee>
<name> vikas </name>
<position> Clerk </position>
```

```
</employee>
```

Ans. In a well-formed XML document, there must be one root element that contains all the others.

Q29. Where do you see problem with this XML document?

```
<?xml version = “1.0” encoding= “UTF-8” standalone= “yes”?>
<!DOCTYPE document[
<!ELEMENT document (employee)*>
<!ELEMENT employee (name, hiredate)>
<!ELEMENT NAME (#PCDATA)>
<!ELEMENT hiredate (#PCDATA)>]>
<document>
<employee>
<hiredate>February 25, 2008</hiredate>
<name>
Sanya Mittal
</name>
</employee>
</document>
```

Ans. The <hiredate> and <name> elements appear in the wrong order.

Q30. What’s wrong with this XML document?

```
<?xml version= “1.0” encoding= “UTF-8” standlone= “yes”?>
<!DOCTYPE document[
<!ELEMENT document (employee)*>
<!ELEMENT employee (hiredate, name)>]>
<document>
<employee>
<hiredate>February 25, 2008</hiredate>
<name>
Sanya Mittal
</name>
</employee>
</document>
```

Ans. The <hiredate> and <name> elements are not declared in the DTD.

Q31. Describe the differences between DTD and the XML Schema?

Ans.

1. It is written in a different (non-XML) syntax;
2. It has no support for namespaces;
3. It only offers extremely limited data typing.

Q32. How can you use an XML schema to declare an element called <name> that holds text content?
Ans. You can declare the element like this:
<xsd:element name = “name” type= “xsd:string”/>

Q33. Which namespace uses XML schemas?
Ans. The namespace that is used by XML schemas is www.w3.org/2001/XMLSchema.

Q34. What is XSL Transformation (XSLT)?
Ans. XML Stylesheet Language (XML) is specifically used to define how an XML document’s data is rendered and to define how on XML document can be transformed into another document. XSLT, A subset of XSL, is a language in both the markup and programming sense, providing a mechanism to transform XML structure into either another XML structure, HTML, or any number of other text-based formats (such as SQL). XSLT’s main ability is to change the underlying structures rather than simply the media representations of those structures, as with Cascading Style Sheet (CSS).

Q35. What is the difference between DOM & SAX APIs?
Ans. Document Object Model (DOM) & Simple API for XML (SAX) are two popular models to interpret an XML document programmatically. DOM (Document Object Mode) is a tree-based API that provides object-oriented view of data. It describes a set of platform and language-neutral interfaces that can represent any well-formed XML/HTML document. While SAX is an event based, serial-access API for XML that uses callbacks to report parsing events to the application. Unlike tree-based APIs do not build an in memory tree representation of the XML document.

Q36. What is the difference between X path & X link?
Ans. X Path is declarative query language for XML that provides a simple syntax for addressing parts of an XML document. It is designed for use with XSLT (for pattern matching). With X Path, collections of elements can be retrieved by specifying a directory-like path, with zero or more conditions

placed on the path. While X Link allows elements to be inserted into XML Documents to create and describe links between resources. It uses XML syntax to create structures that can describe links similar to simple unidirectional hyperlinks of HTML as well as more sophisticated links.

Q37. How can you store an XML document in any relational database?

Ans. 1. XML data can be stored as strings in a relational database.

2. Relations can represent XML data as tree.

3. XML data can be mapped to relations in the same way that E-R schemes are mapped to relational schemas.

Q38. What is XQuery?

Ans. Data extraction, transformation, and integration are well-understood database issues that rely on a query language. SQL does not apply directly to XML because of the irregularity of XML data. W3C recently formed an XML Query Working Group to produce a data model for XML documents, set of query operators on this model, and query language based on query operators. Queries operate on single documents or fixed collections of documents, and can select entire documents or subtrees of documents that match conditions based on document content/structure. Queries can also construct new documents based on what has been selected.

<u>Introducing to Data Warehousing</u>

Q39. Describe the architecture of Datawarehouse with the help of a block diagram. Briefly discuss the role of each component of the datawarehouse architecture. [June-07, Q2(b)]

Ans. A data warehouse is defined as subject-oriented, integrated, nonvolatile, time–variant collection.

The Basic Components of a Data Warehouse

A data warehouse basically consists of three components:
The Data Sources
The ETL and
The Schema of data warehouse including meta data.

Figure defines the basic architecture of a data warehouse. The analytical reports are not a part of the data warehouse but are one of the major business application areas including OLAP and DSS.

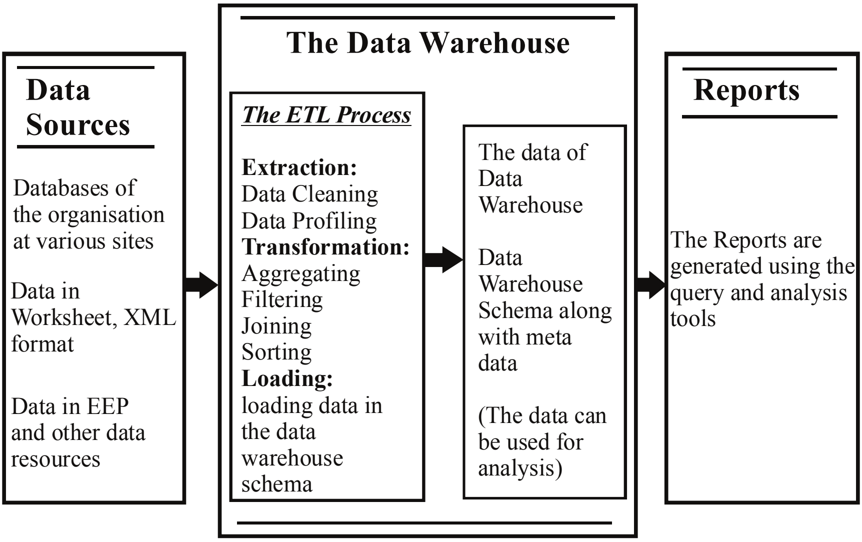

The Data Warehouse
Data Sources
Databases of the organisation at various sites
Data in Worksheet, XML format
Data in EEP and other data resources
The ETL Process
Extraction:
Data Cleaning
Data Profiling
Transformation:
Aggregating
Filtering
Joining
Sorting
Loading:
loading data in the data warehouse schema
The data of Data Warehouse
Data Warehouse Schema along with meta data
(The data can be used for analysis)
Reports
The Reports are generated using the query and analysis tools

Figure : The Data Warehouse Architecture

The data Sources

The data of the data warehouse can be obtained from many operational systems. A data warehouse interacts with the environment that provides most of the source data for the warehouse. By the term environment, we mean, traditionally developed applications. In a large installation, hundreds or even thousands of these database systems or files based system exist with plenty of redundant data.

The warehouse database obtains most of its data from such different forms of legacy systems – files and databases. Data may also be sourced from external sources as well as other organisational systems, for example, an office system. This data needs to be integrated into the warehouse.

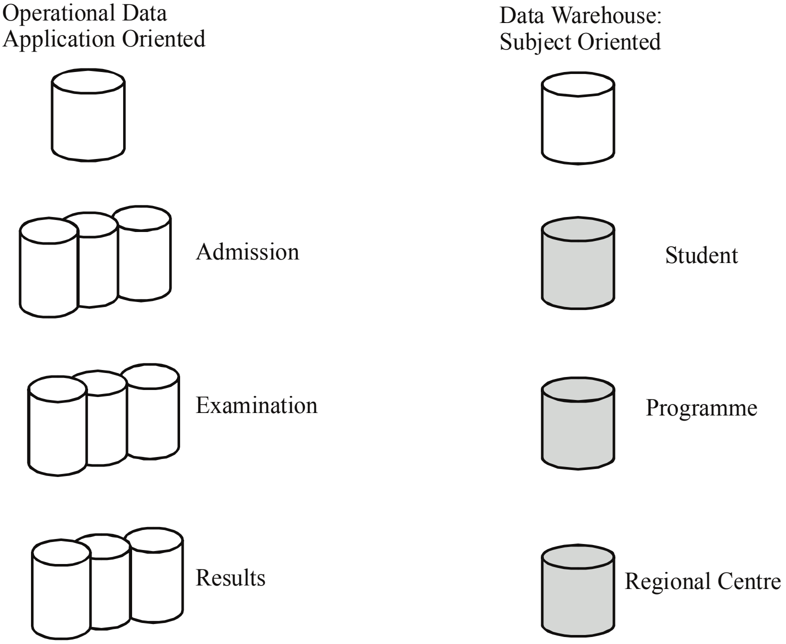
Data of Data Warehouse

A data warehouse has an integrated, “subject-oriented”, “time-variant” and “non-volatile” collection of data. The basic characteristics of the data of a data warehouse can be described in the following way:

(i) Integration : Integration means bringing together data of multiple, dissimilar operational sources on the basis of an enterprise data model. The enterprise data model can be a basic template that identifies and defines the organisation’s key data items uniquely. It also identifies the logical relationships between them ensuring organisation wide consistency in terms of:

Data naming and definition: Standardizing for example, on the naming of "student enrolment number" across systems.
Encoding structures : Standardizing on gender to be represented by "M" for male and "F" for female or that the first two digit of enrolment number would represent the year of admission.
Measurement of variables: A Standard is adopted for data relating to some measurements, for example, all the units will be expressed in metric system or all monetary details will be given in Indian Rupees.

(ii) Subject Orientation : The second characteristic of the data warehouse's data is that its design and structure can be oriented to important objects of the organisation. These object such as STUDENT, PROGRAMME, REGIONAL CENTERS etc., are in contrast to its operational systems, which may be designed around applications and functions such as ADMISSION, EXAMINATION and RESULT DECLARATIONS (in the case of a University).



Operations system data orientation vs. Data warehouse data orientation

(iii) Time-Variance : The third defining characteristic of the database of data warehouse is that it is time-variant, or historical in nature. The entire data in the data warehouse is/was accurate at some point of time. This is, in contrast with operational data that changes over a shorter time period. The data warehouse's constrains data that is data-stamped, and which is historical data.

(iv) Non–volatility (static nature) of data : Data warehouse data is loaded on to the data warehouse database and is subsequently scanned and used, but is not updated in the same classical sense as operational system's data which is updated through the transaction processing cycles.

Decision Support and Analysis Tools

A data warehouse may support many OLAP and DSS tools. Such decision support applications would typically access the data warehouse database through a standard query language protocol; an example of such a language may be SQL. These applications may be of three categories: simple query and reporting, decision support systems and executive information systems.

The meta data directory component defines the repository of the information stored in the data warehouse. The meta data can be used by the general users as well as data administrators. It contains the following information.

(i) the structure of the contents of the data warehouse database,
(ii) the source of the data,
(iii) the data transformation processing requirements, such that, data can be passed from the legacy systems into the data warehouse database,
(iv) the process summarization of data,
(v) the data extraction history, and
(vi) how the data needs to be extracted from the data warehouse.

The first step in data warehousing is, to perform data extraction, transformation, and loading of data into the data warehouse. This is called ETL that is Extraction, Transformation, and Loading. ETL refers to the methods involved in accessing and manipulation data available in various sources and loading it into a target data warehouse. Initially the ETL was performed using SQL programs, however, now there are tools available for ETL processes. The manual ETL was complex as it required the creation of a complex code for extracting data from many sources. ETL tools are very powerful and offer many advantages over the manual ETL. ETL is a step-by-step process. As a first step, it maps the data structure of a source system to the structure in the target data warehousing system. In the second step, it cleans up the data using the process of data transformation and finally, if loads the data into the target system.

Q40. List some uses of data warehousing in Banking, Airline, Hospital and Investment & Insurance.

Ans. (1) Banking - Creating new schemes for loans and other banking products, helps in operations, identities information for marketing.

Strategic Uses : Finding trends for customer service, service promotions, reduction of expenses.

(2) Airline : Operations, marketing

Strategic Uses : Crew assignment, aircraft maintenance plans, fare determination, analysis of route profitability, frequent-flyer program design.

(3) Hospital : Operation optimisation

Reduction of operational expenses, scheduling of resources.

(4) Investment and Insurance : Insurance product development, marketing Risk management, financial market analysis, customer tendencies analysis, portfolio management.

Q41. List the advantages and characteristics of data warehouses.

Ans. 1) It provides historical information that can be used in many different forms of comparative and competitive analysis.

2) It enhances the quality of the data and tries to make it complete.

3) It can help in supporting disaster recovery although not alone but with other backup resources.

Characteristics of Data Warehouses : Data warehouses have the following important features :

(1) Multidimensional conceptual view : A data warehouse contains data of many operational systems, thus, instead of a simple table it can be represented in multidimensional data form also.

(2) Unlimited dimensions and unrestricted cross-dimensional operations: Since the data is available in multidimensional form, it requires a schema that is different from the relational schema.

(3) Dynamic sparse matrix handling : This is a feature that is much needed as it contains huge amount of data.

(4) Client/server architecture : This feature helps a data warehouse to be accessed in a controlled environment by multiple users.

(5) Accessibility and transparency, intuitive data manipulation and consistent reporting performance : This is one of the major features of the data warehouse. A data warehouse contains, huge amounts of data, however, that should not be the reason for bad performance or bad user interfaces. Since the objectives of data warehouse are clear, therefore, it has to support the following easy to use interfaces, strong data manipulation, support for applying and reporting of various analyses and user-friendly output.

Q42. What is a dimension, how is it different from a fact table?

Ans. A dimension may be equated with an object. For example in a sales organisation the dimensions may be salesperson, product and period of a quarterly information. Each of these is a dimension. The fact table will represent the fact relating to the dimensions. For the dimensions above, a fact table may include sale (in rupees) made by a typical sales person for the specific product for a specific. This will be an actual date, thus is a fact. A fact, thus, represents an aggregation of relational data on the dimensions.

Q43. What are the key concerns when building a data warehouse?

Ans. Following are the key concerns when building a data warehouse:

- How will the data be acquired?
- How well it be stored?
- The type of environment in which the data warehouse will be implemented?

Q44. How is snowflake schema different from other schemes?

Ans. The primary difference lies in representing a normalized dimensional table.

Q45. What are the major issues related to data warehouse implementation?

Ans.

- Creation of proper transformation programs
- Proper training of development team
- Training of data warehouse administrator and end users
- Data warehouse maintenance.

Q46. What are OLAP, MOLAP and ROLAP?

Ans. OLAP refers to the statistical processing of multidimensional such that the results may be used for decision-making. MOLAP and ROLAP are the two implementations of the OLAP. In MOLAP the data is stored in the multidimensional form in the memory whereas in the ROLAP it is stored in the relational database form.

Q47. Define the terms: DSS and ESS.

Ans. The DSS is a decision support system and not a decision-making system. It is a specific class of information system that supports business and organisational decision-making activities. A DSS is an interactive software-based system that helps decision makers compile useful information from raw data or documents, personal knowledge, etc. This information helps these decision makers to identify and solve problems and take decisions.

An Executive Information System (EIS) facilitates the information and decision making needs of senior executives. They provide easy access to relevant information (both internal as well as external), towards meeting the strategic goals of the organisation. These are the same as for the DSS.

Q48. How does data warehouse differ from materialized views?

Ans. The difference may be:

- Data warehouse has a multi-dimensional schema whereas views are relational in nature.
- Data warehouse extracts and transforms and then stores the data into its schema that is not true for the materialized views.
- Materialized views needs to be upgraded on any update, whereas, a data warehouse does not need updation.
- Data warehouse data is time-stamped, thus, can be differentiated from older versions that is not true for the materialized views.

Q49. How is data mart different from data warehouse?

Ans. The basic constructs used to design a data warehouse and a data mart are the same. However, a Data Warehouse is designed for the enterprise, while Data Marts may be designed for a business division/department level. A data mart contains the required subject specific data for local analysis only.

Q50. What do you mean by Data mining? How does Database processing differ from Data mining processing? [June-07, Q1(c)]

Ans. Data is growing at a phenomenal rate today and the users expect more sophisticated information from this data. There is need for new techniques and tools that can automatically generate useful information and knowledge from large volumes of data. Data mining is one such technique of generating hidden information from the data. Data mining can be defined as : "an automatic process of extraction of non-trivial or implicit or previously unknown but potentially useful information or patterns from data in large databases, data warehouses or in flat files".

Data mining is related to data warehouse in this respect that, a data warehouse is well equipped for providing data as input for the data mining process. The advantages of using the data of data warehouse for data mining are many some of them are listed below:

- Data quality and consistency are essential for data mining, to ensure, the accuracy of the predictive models. In data warehouses, before loading the data, it is first extracted, cleaned and transformed. We will get good results only if we have good quality data.
- Data warehouse consists of data from multiple sources. The data in data warehouses is integrated and subject oriented data. The data mining process performed on this data.
- In data mining, it may be the case that, the required data may be aggregated or summarised data. This is already there in the data warehouse.
- Data warehouse provides the capability of analysing data by using OLAP operations, Thus, the result of a data mining study can be analyzed for hirtherto, uncovered patterns.

Database Processing Vs. Data Mining Processing

Let us, first, differentiate between database processing and data mining processing: The query language of database processing is well defined and it uses SQL for this while, the data mining, the query is poorly defined and there is no precise query language. The data used in data processing is operational data, while, in data mining, it is historical data i.e., it is not operational data. The output of the query of database processing is precise and is the subset of the data, while, in the case of data mining the output is fuzzy and it is not a subset of the data.

	Data Processing	**Data Mining**
Query	• Well defined • SQL	• Poorly defined • No precise query language
Data	• Uses Operational data	• Does not use Operational data
Output	• Precise • Subset of database	• Fuzzy • Not a subset of database

Some of the examples of database queries are as follows:

- Find all credit card applicants with the last name Veenu.
- Identify customers who have made purchases of more than Rs. 10,000/-in the last month.
- Find all customers who have purchased shirt(s).

Some data mining queries may be:

- Find all credit card applicants with poor or good credit risks.
- Identify the profile of customers with similar buying habits.
- Find all items that are frequently purchased with shirt (s).

Q51. What is the difference between the following :
(i) Database processing and data Mining processing.
(ii) Data Mining and KDD.

Ans. (i) The query language of database processing is well defined and it uses SQL for this, while, the data mining, the query is poorly defined and there is no precise query language. The data used in data processing is operational data, while, in data mining, it is historical data i.e., it is not operational data.

Example of database query are as follows : Identify customers who have made purchase of more than Rs.20,000/- in the last month.

Example of data mining query may be : Find all credit card applicants with poor or good credit risks.

(ii) Knowledge Discovery in Databases (KDD) is the process of finding useful information, knowledge and patterns in data while data mining is the process of using of algorithms to automatically extract desired information and patterns, which are derived by the Knowledge Discovery in Databases process.

Q52. What are different data mining tasks?

Ans. The different data-mining tasks are: Classification, Clustering and Association Rule Mining.

Q53. What is the difference between data mining and OLTP?

Ans. The Query language of OLTP is well defined and it uses SQL for it, while, for data mining the query is poorly defined and there is no precise query language. The data used in OLTP is operational data while in data mining it is historical data. The output of the query of OLTP is precise and is the subset of the data while in the case of data mining the output is fuzzy and it is not a subset of the data.

Q54. How is clustering different from classification?

Ans. In classification, the classes are predetermined, but, in search for interesting relationships the groups are not predetermined. The number of clusters has to be given by the user.

Q55. What is the classification of data? Give some examples of classification.

Ans. The classification task maps data into predefined groups or classes. The class of a tuple is indicated by the value of a user-specified goal attribute. Tuples consists of a set of predicating attributes and a goal attribute. The task is to discover some kind of relationship between the predicating attributes and the goal attribute, so that the discovered knowledge can be used to predict the class of new tuple(s).

Some of the examples of classification are: Classification of students grades depending upon their marks, classification of customers as good or bad customer in a bank.

Q56. What do you mean by clustering? Briefly describe the concept of Hierarchical clustering. How is clustering related to Data mining?

[June-07, Q3(a)]

Ans. Clustering is grouping thing with similar attribute values into the same group. Given a database D=$\{t_1, t_2,\ldots,t_n\}$of tuples and an integer value k, the clustering problem is to defined a mapping where each tuple t_i is assigned to one cluster k_j, 1<=j<=k.

A Cluster, K_j, contains precisely those tuple mapped to it. Unlike the classification problem, clusters are not known in advance. The user has to enter the value of the number of clusters k.

In other words a Cluster can be defined as the collection of data objects that are similar in nature, as per certain defining property, but these objects are dissimilar to the objects in other clusters.

Some of the clustering examples are as follows:

- To segment the customer database of a department store based on similar buying patterns.
- To identify similar Web usage patterns etc.

Clustering is a very useful exercise specially for identifying similar groups from the given data. Such data can be about buying patterns, geographical locations, web information and many more. Clustering is related to data mining. Data mining problems are solved by clustering.

Some of the clustering issues are as follows:

- **Outlier handling:** How will outlier be handled? (outliers are the objects that do not comply with the general behaviour or model of the data) Whether it is to be considered or it is to be left aside while calculating the clusters?
- **Dynamic data:** How will you handle dynamic data?
- **Interpreting results:** How will the result be interpreted?
- **Evaluating results:** How will the result be calculated?
- **Number of clusters:** How many clusters will you consider for the given data?
- **Data to be used:** Whether you are dealing with quality data or the noisy data? If, the data is noisy how is it to be handled?
- **Scalability :** Whether the algorithm that is used is to be scaled for small as well as large data set/database.

There are many different kinds of algorithm for clustering.

Hierarchical Clustering

In this method, the clusters are created in levels and depending upon the threshold value at each level the clusters are again created.

An agglomerative approach begins with each tuple in a distinct cluster and successively merges clusters together until a stopping criterion is satisfied. This is the bottom up approach.

A divisive method beings with all tuples in a single cluster and performs splitting until a stopping criterion is met. This is the top down approach.

A hierarchical algorithm yields a dendrogram representing the nested grouping of tuples and similarity levels at which grouping change. Dendrogram is a tree data structure which illustrates hierarchical clustering techniques. Each level shows cluster for that level. The leaf represents individual cluster while the root represent one cluster.

Q57. List some applications data mining.

Ans. Following are the some applications data mining :

(1) Marketing and sales data analysis : A company can use customer transactions in their database to segment the customers into various types. Such companies may launch products for specific customer bases.

(2) Investment analysis : Customers can look at the areas where they can get good returns by applying the data mining.

(3) Loan approval : Companies can generate rules depending upon the dataset they have. On that basis they may decide to whom, the loan has to be approved.

(4) Fraud detection : By finding the correlation between faults, new faults can be detected by applying data mining.

(5) Network management : By analysing pattern generated by data mining for the networks and its faults, the faults can be minimised as well as future needs can be predicted.

(6) Risk Analysis : Given a set of customers and an assessment of their risk-worthiness, descriptions for various classes can be developed. Use these descriptions to classify a new customer into one of the risk categories.

(7) Brand Loyalty : Given a customer and the product he/she uses, predict whether the customer will be changing their products.

(8) Housing loan prepayment prediction : Rule discovery techniques can be used to accurately predict the aggregate number of loan prepayments in a given quarter as a function of prevailing interest rates, borrower characteristics and account data.

Q58. List the example of problems which can be solved by data mining and also discuss briefly the approaches to data mining problems.

Sample Data Mining Problems : problems that can be solved through the data mining process.

(1) Mr. Suraj Gupta manages a supermarket and the cash counters, he adds transactions into the database. Some of the questions that can come to Mr. Gupta's mind are as follows:

(a) Can you help me visualize my sales?

(b) Can you profile my customers?

(c) Tell me something intersecting about sales such as, what time sales will be maximum etc.

He does not know statistics, and he does not want to hire statisticians.

The answer of some of the above questions may be answered by data mining.

(2) Mr. Avinash Arun is an astronomer and the sky survey has 3 tera-bytes (10^{12}) of data, 2 billion objects. Some of the questions that can come to the mind of Mr. Arun are as follows:

(a) Can you help me recognize the objects?

(b) Most of the data is beyond my reach. Can you find new/unusual items in my data?

(c) Can you help me with basic manipulation, so I can focus on the basic science of astronomy?

He knows the data and statistics, but that is not enough. The answer to some of the above questions may be answered once again, by data mining.

Please note : The use of data mining in both the questions given above lies in finding certain patterns and information. Definitely the type of the data in both the database as given above will be quite different.

Approaches to Data Mining Problems: The approaches to data mining problems are based on the type of information/knowledge to be mined. We will emphasis on three different approaches:

(a) Classification

(b) Clustering

(c) Association Rules

The classification task maps data into predefined groups or classes. The class of a tuple is indicated by the value of a user-specified goal attribute. Tuples consists of a set of predicating attributes and a goal attribute. The task, is to discover, some kind of relationship between the predicating attributes and the goal attribute, so that, the discovered information/knowledge can be used to predict the class of new tuple(s).

The task of clustering is to group the tuples with similar attribute values into the same class. Given a database of tuples and an integer value k, the Clustering is to define a mapping, such that, tuples are mapped to different cluster.

The principle is to maximize intra-class similarity and minimize the interclass similarity. In clustering, there is no goal attribute. So, classification is supervised by the goal attribute, while clustering is an unsupervised classification.

The task of association rule mining is to search for interesting relationships among items in a given data set. Its original application is on "market basket data". The rule has the form $X \rightarrow Y$, where X and Y are sets of items and they do not intersect. Each rule has two measurements, support and confidence. Given the user-specified minimum support and minimum confidence, the task is to find, rules with support and confidence above, minimum support and minimum confidence.

Q59. What is association rule mining?

Ans. The task of association rule mining is to search for interesting relationships among items in a given data set. Its original application is on "market basket data". The rule has the form $X \rightarrow y$, where X and Y are sets of items and they do not intersect.

Q60. Explain Apriori Algorithm.

Ans. Apriori is used for finding frequent itemsets : The apriori algorithm applies the concept that if an itemset has minimum support, then all its subsets also have minimum support. An itemset having minimum sopport is called frequent itemset or large itemset. So any subset of a frequent itemset must also be frequent.

Apriori algorithm generates the candidate itemsets to be counted in the pass, by using only the large item set found in the previous pass - without considering the transactions in the database.

It starts by finding all frequent 1-itemsets (itemsets with 1 item); then consider 2 -itemsets from these 1-itemsets, and so forth. During each iteration only candidates found to be frequent in the previous iteration are used to generate a new candidate set during the next iteration. The algorithm terminates when there are no frequent k-itemsets.

Notations that are used in Apriori algorithm are given below :

k-itemset	An itemset having k items
L_k	Set of frequent k-itemset (those with minimum support)
C_k	Set of candidate k-itemset (potentially frequent itemsets)

Apriori algorithm function takes as argument L_{k-1} and returns a superest of the set of all frequent k-itemsets. It consists of a join step and a prune step.

The Apriori algorithm is given below :

APRIORI

```
1. k = 1
2. Find frequent set $L_k$ from $C_k$ of all candidate itemsets
3. Form $C_{k+1}$ from $L_k$ ;k = k + 1
4. Repeat 2-3 untill $C_k$ is empty
```

Details about steps 2 and 3

Step 2: S

Can the data set D and count each itemset in C_k , if it is greater than minimum support, it is frequent.

Step 3: i) For k = 1, = all frequent 1-itemsets. (all individual items).

ii) For k >1, generate C_k from Lk–1 as follows:

The join step

C_k = k–2 way join of L_{k-1} with itself

If both $\{a_1, ..., a_{k-2}, a_{k-1}\}$ & $\{a_1, ..., a_{k-2}, a_k\}$ are in L_{k-1} ,then add$\{a_1, ..., a_{k-2}, a_{k-1}, a_k\}$ to C_k

(We keep items sorted).

The prune step

Remove $\{a_1, ..., a_{k-2}, a_{k-1}, a_k\}$ if it contains a non-frequent (k-1) subset. In the prune step, delete all itemsets $c \in C_k$ such that some (k-1)-subset of C is not in $L_{k-1}\}$

Q61. Apply the Apriori algorithm for generating large itemset on the following dataset:

Transaction ID	**Items purchased**
T100	$A_1a_3a_4$
T200	$A_2a_3a_5$
T300	$A_1a_2a_3a_5$
T400	A_2a_5

Ans. The dataset D given for the problem is:

Transaction ID	Items purchased
T100	$A_1a_3a_4$
T200	$A_2a_3a_5$
T300	$A_1a_2a_3a_5$
T400	A_2a_5

Assuming the minimum support as 50% for calculating the large item sets. As we have 4 transaction at least 2 transaction should have the data item.

1. scan D $\rightarrow C_1$; a_1:2, a_2:3, a_3:3, a_4:1, a_5:3
$\rightarrow L_1$: a_1:2, a_2:3, a_3:3, a_5:3
$\rightarrow C_2$: a_1a_2, a_1a_3, a_1a_5, a_2a_3, a_2a_5, a_3a_5

2. scan D $\rightarrow C_2$: a_1a_2:1, a_1a_3:2, a_1a_5:1, a_2a_3:2, a_2a_5:3, a_3a_5:2
$\rightarrow L_2$: a_1a_3:2, a_2a_3:2, a_2a_5:3, a_3a_5:2
$\rightarrow C_3$: $a_1a_2a_3$, $a_1a_2a_5$, $a_2a_3a_5$
$\rightarrow$ Pruned C_2: $a_2a_3a_5$

3. scan D $\rightarrow L_3$: $a_2a_3a_5$;2

Thus L= $\{L_1, L_2, L_3\}$

Chapter-7

EMERGING TRENDS AND EXAMPLE DBMS ARCHITECTURES

Q1. Explain the need and features for the following advanced database application systems:

Ans. (i) Multimedia Database

A Multimedia Database Management System (MMDBMS) provides support for multimedia datatypes required for digital images, audio, video, animation and graphics along with textual data.

Main application or need of Traditional Database Management System is to store frequently used data, represent data in various layout/forms which are stored once. In revolutionary world of Technology, the huge amount of data in different multimedia-related application need consistency, concurrency, integrity, security and high availability of data. For example, application like digital signature authentication, eye ratina authentication, speech recognition, text to speech conversion, speech to text conversion, animation in movie etc. require processing, retrieval, searching like database operation or multimedia type of data. Thus, Multimedia and its application have experience tremendous growth of Multimedia database experience tremendous growth of Multimedia database.

Features :

- Provides features of traditional database.
- Provide homogeneous framework for storing, processing, retrieving, transmitting and presenting wide variety of multimedia data types.
- Provide large volume of formats for variety of multimedia data types.
- Huge size of MMDBMS.
- Manage different type of input, output and storage devices.
- Size of multimedia data is larger than traditional text data. So, it should provide variety of data compression and storage formats for various types of data.
- It consists of information about sampling rate, resolution, frame rate, encoding scheme etc. of various media data.

- It comprise additional information like keywords to make retrieval rapidly.
- It needs to synchronise multiple media types relating to one single multimedia object.

(ii) Geographic databases :

Need for Geographic database : In modern era of information and communication Network technology, Electronic device and GIS (Geographic Information System) related application provide Geographical information on computer devices.

Application like country map, space map, weather forecasting, global positioning system, space aircraft designing or planning, site map etc. require geographic information.

Thus, Geographic Database is aimed to provide Geographic information.

Every time processing to convert Geographic information as digital representation of land, elevation or top view of earth surfaces require lots of time. Geographic Database can help to reduce this processing time.

Features of Geographic database :

- Provide Geographic information as elevation, top plan of part of earth surface.
- The data of GIS are represented in graphical form.
- Geographical data can be stored in any format like vector data, raster data or combination of both.
- It stores and or provides Geographic information in either 2-dimensional or 3-dimensional.
- It provides information as collection of layers.
- Most of accesses to the database would be read only.

(iii) Knowledge databases :

Need : Knowledge based systems get their power from the expert knowledge that has been coded into facts, rules heuristics. The knowledge is stored in a knowledge base separate from the control and inferencing components. This makes it possible to add new knowledge or refine existing knowledge without recompiling the control inferencing programs. This simplifies the construction and maintenance of knowledge based systems.

A knowledge base system may defined as an extension of the database concept containing not only data but also facts, rules heuristics and procedures. The knowledge base is created by or for the end-user using the shell. Here, the knowledge is stored in a knowledge base separate from the control and inferencing components to make it possible to add new knowledge and inferencing programs. This greatly simplifies the construction and maintenance of knowledge base systems.

Features :

1) Has information at a higher level of obstruction.

2) Significantly smaller than database and changes are gradual.

3) By logic or rules or frames or semantics.

4) Updation performed by domain experts.

5) Correctness in a sense is very exclusive.

6) Has the power of in facing.

7) Used for data-analysis.

8) How to have a correction with the system and provide needed data to obtain the solution.

Q2. What are the reasons for the growth of multimedia data? What are the contents of multimedia database?

Ans. Reasons for the growth of multimedia data

(a) Advanced technology in terms of devices that were digital in nature and support capture and display equipment.

(b) High speed data communication network and software support for multimedia data transfer.

(c) Newer application requiring multimedia support.

Content of multimedia database which can be of two basic types:

(a) Media Content

(b) Meta data, which includes media, format data, media keyword data and media feature data.

Q3. List four application areas of multimedia databases.

Ans. Four application areas of multimedia databases are listed below as:

Medical media databases

Bio-informatics

Home media

News etc.

Q4. List the challenges in designing multimedia databases.

Ans. Challenges in designing multimedia databases are the following:

(a) Support for different types of input/output

(b) Handling many compressions algorithms and formats
(c) Differences in OS and hardware
(d) Integrating to different databases models
(e) Support for queries for a variety of media types
(f) Handling different kinds of indices
(g) Data distribution over the world etc.

Q5. What is GIS? What are its applications?

Ans. GIS is a spatial database application where the spatial and non-spatial data is represented along with the map. Some of the applications of GIS are:

- Cartographic applications
- 3-D Digital modeling applications like land elevation records.
- Geographic object applications like traffic control system.

Q6. What are the database requirements for Genome?

Ans. The data may need to be organised for the following three levels:

- **Geonomics**: Where four different types of data are represented. The physical data may be represented using eight different fields.
- **Gene expression**: Where data is represented in fourteen different fields
- **Proteomics**: Where data is used for five research problems.

Q7. List the requirements of a GIS.

Ans. Following are the requirements of a GIS:

- Data representation through vector and raster
- Support for analysis of data
- Representation of information in an integrated fashion
- Capture of information
- Visualization of information
- Operations on information

Q8. What are the features of deductive databases?

Ans. Following are the features of deductive databases:

(i) They represent information using facts and rules
(ii) New facts and rules an be deduced
(iii) Used in expert system type of applications.

Q9. What is a good knowledge base?

Ans. A good knowledge database will have good information, good classification and structure and an excellent search engine.

Q10. Define the term Mobile Database? Also discuss about the future scope of Mobile Databases.

Ans. A mobile database is a database that can be connected to by a mobile computing device over a mobile network. The client and server have wireless connections. A cache is maintained to hold frequent data and transactions so that they are not lost due to connection failure.

The use of laptops, mobiles and PDAs is increasing and likely to increase in the future with more and more applications residing in the mobile systems. It is clear that a large percentage of mobile users will require the use of a database of some sort. Many applications such as databases would require the ability to download information from an information repository and operate on this information even when "out of range" or disconnected.

An example of this is a mobile workforce. In this scenario user would require to access and update information from files in the home directories on a server or customer records from a database. This type of access and work load generated by such users is different from the traditional workloads seen in client server systems of today. With the advent of mobile databases, now users can load up their smart phones or PDAs with mobile databases to exchange mission-critical data remotely without worrying about time or distance. Mobile databases let employees enter data on the fly. Information can be synchronized with a server database at a later time.

Q11. What are the characteristics of Mobile Database. Also discuss in brief about major drawbacks of mobile databases.

Ans. The mobile environment has the following characteristics :

(1) Communication Latency.

(2) Intermittent wireless connectivity.

(3) Limited battery life.

(4) Changing location of the client.

Major drawback of mobile databases.

Major drawback of mobile databases - is the limitation of power and available size of the display unit has found newer technologies like use of flash, power-saving disks, low-powered and power saving displays. However, the mobile devices since are normally smaller in size requires creation of presentation standards. One such protocol in the areas of wireless networking is the Wireless Access Protocol (WAP).

Q12. List the steps required to create a web database?

Ans. The following steps are required to create a web database:

- Create a database and put it on a database server.
- Create a connection string to connect to the server through a valid username and password
- Open the connection to bring in the required data based on the suitable query interactions
- Format and display the data at the client site.

Q13. What are the advantages of using wireless LAN?

Ans. Wireless LANs may allow low cost communication between two mobile units the are located in the same LAN area. Thus, it may result in reduced cost of operation.

Q14. List the different characteristics of mobile databases?

Ans. The following are the characteristics of mobile databases:

- Mobile database relies on the broadcast of data
- Mobile stations may be working in standalone mode most of the time
- The data on a mobile workstation may be inconsistent
- Mobile database may be made scalable
- Mobile database are closer to distributed database technology
- A mobile unit may be changing its physical location. It is to be reached at all locations.

Q15. Discuss the need for ODBC?

Ans. ODBC allows using standard SQL commands to be used on any database on any DBMS. It gives application designer freedom from learning the features of individual DBMS, or OS etc. Thus, it simplifies the task of database programmers.

Q16. Why do you need JDBC? What happens when a DBMS does not have a JDBC driver?

Ans. JDBC is an API that allows Java programmers to access any database through the set of its standard API. In case a JDBC driver is not available for a DBMS then the ODBC-JDBC bridge can be used to access data.

Q17. List the components required for implementing ODBC?

Ans. The following three components are required to implement ODBC:

- The application
- The core ODBC library
- The database driver for ODBC

Q18. List the basic functions of a digital library?

Ans. A digital library supports

- Content management
- Search
- License management
- Link management
- Meta data storage

Q19. List the different types of hardware and software required for digital libraries?

Ans. Digital libraries require:

- Very large secondary storage
- Distributed array of powerful servers
- Large bandwidth and data transfer rate
- A reliable library software.

Q20. What is Datagrid? What is the requirement of datagrid (give at least 3 requirements)? Describe the structure of datagrid with the help of a block diagram. [June-07, Q3(c)]

Ans. A data grid can be seen as the process of creating a virtual database across hardware of almost the entire data that exists in some form. Thus, the key concept is Virtual Database. The basic concept behind a data grid is that an application need not know either the place or the DBMS where the data is stored; rather the application is only interested in getting the correct results. Figure shows a typical structure for a data grid.

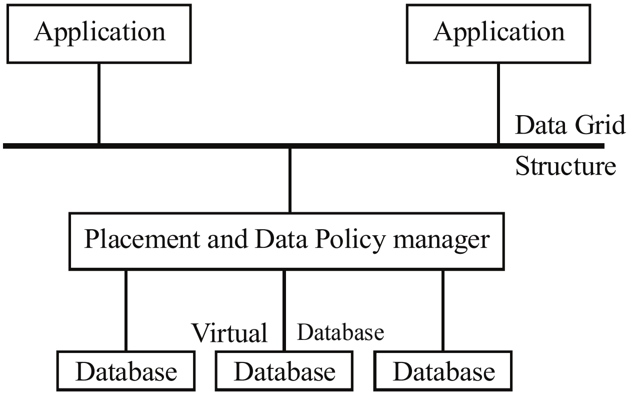


Figure : A Data Grid

A data grid should address the following issues:

- It should allow data security and domain specific description of data. Domain specific data helps in the identification of correct data in response to a query.

- It should have a standard way of representing information. [The possible candidates in this category may be XML].
- It should allow simple query language like SQL for accessing information.
- The data requirements should be fulfilled with a level of confidence, (that is there has to be a minimum quality of service in place).
- There needs to be a different role assigned to the data administrator. The data administrators should not be concerned with what the data represent. Data domain specification should be the role responsibility of the data owner. Unfortunately, this does not happen in the present database implementations.
- The separation of the data administration and the data manager will help the database administrator to concentrate on data replication and query performance issues based on the DBMS statistics.

Thus, a data grid virtualizes data. Many DBMSs today, have the ability to separate the roles of the database administrator and the data owner from the application access. This needs to be extended to the grid across a heterogeneous infrastructure.

Some of the major requirements of a data grid are:

- **Handling of failure of a data source:** A grid may have replicas and caches, but grid applications tend to access a data resource. A data grid is flexible and it should have a middleware that automatically moves the operations to either another data resource with a similar data set or to multiple data resources each with a subset of the data.
- **Parallel access for local participant:** some organisations may have very large data, thus, a query on that would take longer and such a query could be beyond the performance criteria set for the grid. Therefore, it may be a good idea to use a "virtual distributed" database to fetch data in parallels and keep processing it.
- **Global access through SQL :** A data grid would require dynamic selection of data sources. Therefore, it requires a complete SQL query transformation and optimisation capability.
- A data grid application may need to access data sources such as content management systems, Excel files, or databases not yet supported.

Q21. What are the advantages of Data Grid?

Ans. Data Grid is helpful in the sharing of large amount of information on a particular topic. Thus, allowing a worldwide repository of information, while, on the other hand, still giving full control of information to the creator.

PostgreSQL

Q22. What are the basic features of PostgresSQL?

Ans. PostgreSQL supports the following features:

- ANSI SQL 2 compliance,
- Support for transactions, triggers, referential integrity and constraints,
- High level language support,
- Inheritance, complex data types and polymorphism,
- Built in complex data types like IP address,
- Aggregate functions, collections and sequences,
- Portability, and
- ODBC and JDBC drivers.

Q23. What are the different types of interfaces in PostgresSQL?

Ans. PostgreSQL supports both the terminal monitor interfaces and program driven interfaces.

Q24. What are the basic processes in PostgresSQL?

Ans. PostgreSQL has the following three processes:

- Postmaster
- Server process
- Client processes

Q25. How are the transactions supported in PostgreSQL?

Ans. Each SQL statement is treated as a transaction. It also has provision for multi statement transactions.

Q26. List the add-on non-standard types in PostgreSQL?

Ans. Some of these types are: complex; domains, cstring, record, trigger, void etc.

Q27. How does PostgreSQL perform storage and indexing of tables? Briefly discuss the type of indexes involved in PostgreSQL?

[June-07, Q4(a)]

Ans. PostgreSQL defines a number of system columns in all tables. These system columns are normally invisible to the user, however, explicit queries can report these entries. These columns, in general, contains meta–data i.e., data about data contained in the records of a table.

Thus, any record would have attribute values for the system-defined columns as well as the user-defined columns of a table. The following table lists the system columns.

Column Name	Description
oid (object identifier)	The unique object identifier of a record. It is automatically added to all records. It is a 4-byte number. It is never re-used within the same table.
tableoid (table object identifier)	The oid of the table that contains a row. The pg_class system table relates the name and oid of a table.
xmin (transaction minimum)	The transaction identifier of the inserting transaction of a tuple.
Cmin (command minimum)	The command identifier, starting at 0, is associated with the insetting transaction of a tuple.
xmax (transaction maximum)	The transaction identifier of a tuple's deleting transaction. If a tuple has not been deleted then this is set to zero.
cmax (command maximum)	The command identifier is associated with the deleting transaction of a tuple. Like xmax, if a tuple has not been deleted then this is set to zero.
ctid (tuple identifier)	This identifier describes the physical location of the tuple within the database. A pair of numbers are represented by the ctid: the block number, and tuple index within that block.

Figure : System Columns

If the database creator does not create a primary key explicitly, it would become difficult to distinguish between two records with identical column values. To avoid such a situation PostgreSQL appends every records with its own object identifier number, or OID, which is unique to that table. Thus, no two records in the same table will ever have the same OID, which, also mean that no two records are identical in a table. The oid makes sure of this.

Internally, PostgreSQL stores data in operating system files. Each table has its own file, and data records are stored in a sequence in the file. You can create an index on the database. An index is stored as a separate file that is sorted on one or more columns as desired by the user.

Indexes

Indexes allow fast retrieval of specific rows from a table. For a large table using an index, finding a specific row takes fractions of a second while non-indexed entries will require more time to process the same information. PostgreSQL does not create indexes automatically. Indexes are user defined for attributes or columns that are frequently used for retrieving information.

For example, you can create and index such as:

```
mydb => CREATE INDEX stu_name ON Student (first_name);
```

PostgreSQL supports many types of index implementations. These are:

- **B-Tree Indexes:** These are the default type index. These are useful for comparison and range queries.
- **Hash Indexes:** This index uses linear hashing. Such indexes are not preferred in comparison with B-tree indexes.
- **R-Tree Indexes :** Such index are created on built-in-spatial data types such as box, circle for determining operations like overlap etc.
- **GiST Indexes:** These indexes are created using Generalised search trees. Such indexes are useful for full text indexing, and are thus useful for retrieving information.

Q28. What are system columns?

Ans. System columns are added by PostgrSQL to all the tables. These are oid (object identifier), tableoid, xmin, xmax, ctid.

Q29. What are the different steps for query evaluation?

Ans. The steps are:

- Query submission, the
- Parsing by the parser to query tree,
- Transformation of query by rewrite system
- Creation of query evaluation plan by the optimiser, and
- Execution by executor.

Chapter-8

ORACLE

Q1. What is Oracle ?

Ans. The Oracle Database (commonly referred to as Oracle RDBMS or simply as Oracle), a relational database management system (RDBMS) software product released by Oracle Corporation, has become a major factor in database computing.

Q2. Explain about the Physical and logical structuring of Oracle.

Ans. An Oracle database system comprises at least one instance of the application, along with data storage. An instance comprises a set of operating-system processes and memory-structures that interact with the storage.

The Oracle RDBMS stores data logically in the form of tablespaces and physically in the form of data files. Tablespaces can contain various types of memory segments; for example, Data Segments, Index Segments etc. Segments in turn comprise one or more extents. Extents comprise groups of contiguous data blocks. Data blocks form the basic units of data storage. At the physical level, data-files comprise one or more data blocks, where the block size can vary between data-files.

Oracle database management keeps track of its computer data storage with the help of information stored in the SYSTEM tablespace. The SYSTEM tablespace contains the data dictionary — and often (by default) indexes and clusters. (A data dictionary consists of a special collection of tables that contains information about all user-objects in the database). Since version 8i, the Oracle RDBMS also supports "locally managed" tablespaces which can store space management information in bitmaps in their own headers rather than in the SYSTEM tablespace (as happens with the default "dictionary-managed" tablespaces).

If the Oracle database administrator has instituted Oracle RAC (Real Application Clusters), then multiple instances, usually on different servers, attach to a central storage array. This scenario offers numerous advantages, most importantly performance, scalability and redundancy. However, support becomes more complex, and many sites do not use RAC. In version 10g, grid computing has introduced shared resources where an instance can use (for example) CPU resources from another node (computer) in the grid.

The Oracle DBMS can store and execute stored procedures and functions within itself. PL/SQL (Oracle Corporation's proprietary procedural extension to SQL), or the object-oriented language Java can invoke such code objects and/or provide the programming structures for writing them.

Q3. Discuss about Schemas in Oracle.

Ans. Oracle database conventions refer to defined groups of ownership (generally associated with a "username") as schemas.

Most Oracle database installations traditionally come with a default schema called SCOTT. After the installation process has set up the sample tables, the user can log into the database with the username scott and the password tiger. The name of the SCOTT schema originated with Bruce Scott, one of the first employees at Oracle (then Software Development Laboratories), who had a cat named Tiger.

The SCOTT schema has seen less use as it uses so few of the features of a modern release of Oracle. Most recent examples reference the default HR or OE schemas.

Other default schemas include:
SYS (essential core database structures and utilities)
SYSTEM (additional core database structures and utilities, and privileged account)
OUTLN (utilized to store metadata for stored outlines for stable query-optimizer execution plans 3).
BI, IX, HR, OE, PM, and SH (expanded sample schemas containing more data and structures than the older SCOTT schema)

Q4. Define the term Tablespaces in the context of Oracle RDBMS.

Ans. default tablespaces include:
SYSTEM (essential core database structures and utilities)

SYSAUX (extra/extended data to supplement the SYSTEM schema)
TEMP (temporary tablespace)
UNDOTBS1 (undo tablespace)
USERS (default users tablespace created by the Database Configuration Assistant - but replaceable by the DBA)

Q5. Discuss about the Memory architecture of Oracle.
Ans. Oracle Memory architecture consists of the following :
(i) System Global Area
Each Oracle instance uses a System Global Area or SGA — a shared-memory area — to store its data and control information.

Each Oracle instance allocates itself an SGA when it starts and de-allocates it at shutdown time. The information in the SGA consists of the following elements, each of which has a fixed size, established at instance startup:
* **the database buffer cache:** this stores the most recently-used data blocks. These blocks can contain modified data not yet written to disk (sometimes known as "dirty blocks"), unmodified blocks, or blocks written to disk since modification (sometimes known as clean blocks). Because the buffer cache keeps blocks based on a most-recently-used algorithm, the most active buffers stay in memory to reduce I/O and to improve performance.
* **the redo log buffer:** this stores redo entries - a log of changes made to the database. The instance writes redo log buffers to the redo log as quickly and efficiently as possible. The redo log aids in instance recovery in the event of a system failure.
* **the shared pool:** this area of the SGA stores shared-memory structures such as shared SQL areas in the library cache and internal information in the data dictionary. An insufficient amount of memory allocated to the shared pool can cause performance degradation.

(ii) Library cache
The library cache stores shared SQL, caching the parse tree and the execution plan for every unique SQL statement.

If multiple applications issue the same SQL statement, each application can access the shared SQL area: this reduces the amount of memory needed and reduces the processing-time used for parsing and execution planning.

(iii) Data dictionary cache

The data dictionary comprises a set of tables and views that map the structure of the database.

Oracle stores information here about the logical and physical structure of the database. The data dictionary contains information such as the following:

(a) User information, such as user privileges
(b) Integrity constraints defined for tables in the database
(c) Names and datatypes of all columns in database tables
(d) Information on space allocated and used for schema objects

The Oracle instance frequently accesses the data dictionary in order to parse SQL statements. The operation of Oracle depends on ready access to the data dictionary: performance bottlenecks in the data dictionary affect all Oracle users. Because of this, database administrators should make sure that the data dictionary cache has sufficient capacity to cache this data. Without enough memory for the data-dictionary cache, users see a severe performance degradation. Allocating sufficient memory to the shared pool where the data dictionary cache resides precludes these particular performance problems.

(iv) Program Global Area

The Program Global Area or PGA memory-area contains data and control-information for Oracle's server-processes.

The size and content of the PGA depends on the Oracle-server options installed. This area consists of the following components:

stack-space: the memory that holds the session's variables, arrays, and so on.
session-information: unless using the multithreaded server, the instance stores its session-information in the PGA. (In a multithreaded server, the session-information goes in the SGA.)

(v) Process architecture

The Oracle RDBMS typically relies on a group of processes running simultaneously in the background and interacting to further and monitor database operations. Such processes (and their standard abbreviations) can include:

archiver processes (ARCn)
checkpoint process (CKPT)
database writer processes (DBWn)
dispatcher processes (Dnnn): multiplex server-processes on behalf of users
memory-manager process (MMAN): used for internal database tasks such as Automatic Shared Memory Management
job-queue processes (CJQn)
log-writer process (LGWR)
logical standby coordinator process (LSP0): controls Data Guard log-application
media-recovery process (MRP): detached recovery-server process
memory-monitor process (MMON)
memory-monitor light process (MMNL): gathers and stores Automatic Workload Repository (AWR) data
process-monitor process (PMON)
process-spawner (PSP0): spawns Oracle processes
queue-monitor processes (QMNn)
recoverer process (RECO)
remote file-server process (RFS)
shared server processes (Snnn): serve client-requests
system monitor process (SMON)

Q6. What are Data Definition statements in Oracle?
Ans. These statements create, alter, maintain and drop schemes objects.

Q7. Name the main ORACLE tools for Application Development.
Ans. ORACLE forms Developer
ORACLE Reports Developer
ORACLE Designer
ORACLE J Developer
ORACLE Discoverer Administrative Edition
ORACLE Portal.

Q8. What are the in-built data types of ORACLE?
Ans.
a) Character
b) Numeric
c) DATE data type
d) LOB data type

e) RAW and LONGRAW

f) Rowed and UNROWID data type

Q9. What are the advantages of B+ tree structure?

Ans.

a) All leaf Block have same depth,

b) B-Tree indexes are balanced,

c) Blocks of the B+ tree are 3 quarters full on the average,

d) B+ tree have excellent retrieval performance,

e) Inserts, updates and deletes are all efficient, and

f) Good Performance,

Q10. What are the different files used in ORACLE?

Ans.

a) Data files

b) Redo log files

c) Control files

Q11. What does a data Dictionary store?

Ans.

a) Valid users of data types,

b) Information about integrity Constraints defined for the tables, and

c) The Amount of space Allocated for a scheme object.

Q12. Expand the following:

a) LGWR

b) ARC

c) RECO.

Ans.

a) Log writers,

b) Archiver,

c) Recover.

Q13. What is a process in Oracle?

Ans. A Process is a thread of control or a mechanism in an operating system that can run a series of steps.

Q14. Discuss about backup & recovery in Oracle.

Ans. As every database administrator knows, backing up a database is rather mundane but necessary task. An improper backup makes recovery difficult, it is not possible. Unfortunately, people often realise the extreme importance of this everyday task only when it is too late - usually after losing critical business data due to the failure of a related system.

Recovery Manager : Typical backups include complete database backups (the most common type), tablespace backups, datafile backups, control file backups, and archived redo log backups. Oracle8 introduced the Recovery Manager (RMAN) for serve-managed backup and recovery of the database. Previously, Oracle's Enterprise Backup Utility (EBU) provided a similar solution on some platforms. However, RMAN, with its Recovery Catalogue we stored in an Oracle database, provides a much more complete solution. RMAN can automatically locate, back up, restore, and recover datafiles, control files, and archived redo logs, RMAN, since Oracle9i, can restart backups, restore and implement recovery window policies when backups expire. The Oracle Enterprise Manager Backup Manager provides a GUI-based interface to RMAN. Oracle Enterprise Manager 10g introduces a new improved job scheduler that can be used with RMAN and other scripts, and that can manage automatic backups to disk.

Incremental Backup and Recovery : RMAN can perform incremental backups of Enterprises Edition databases. Incremental backups, backup only the blocks modified since the last backup of a datafile, tablespace or database; thus, they are smaller and faster than complete backups. RMAN can also perform point-in-time recovery, which allows the recovery of data until just prior to an undesirable event (such as the mistaken dropping of a table).

Oracle Storage Manager and Automated Disk Based Backup and Recovery : Various media-management software vendors support RMAN. Since Oracle8i, a Storage Manager has been developed with Oracle to provide media-management services, including the tracing of tape volumes, for up to four devices. RMAN interfaces automatically with the media-management software to request the mounting of tapes as needed for backup and recovery operations.

Oracle Database 10g introduces Automated Disk Based Backup and Recovery. The disk acts as a cache, and archives and backups can then be copied to tape. The disk 'cache' can also serve as a staging area for recovery.

Q15. What are the three basic steps involved in SQL tuning?

Ans.

(a) Identifying high load on top of SQL Statements.

(b) Verifying that the execution plans perform reasonably, and

(c) Implementing corrective actions to generate better execution plans.

Q16. What is Parsing?

Ans. Translating a SQL statement and verifying it to be a valid statement.

Q17. What are heterogeneous services?

Ans. It allows no–oracle data and services to be accessed from an Oracle database through generic connectivity.

Q18. Name the basic technology used to move data from are ORACLE database to another.

Ans.

a) Basic replication,

b) Advanced replication

c) Transportable replication,

d) Advanced queuing and streams, and

e) Extraction, transformation, loading.

Q19. What are the different types of backup?

Ans.

a) Complete database,

b) Tablespace,

c) Data file

d) Control file, and

e) Achieved Redo Log.

Q20. What is Oracle Enterprise manager?

Ans. A complete interface for enterprise wide application development.

Q21. What is Parallel Execution?

Ans. The process of making multiple processes run together.

Q22. What is ETL?

Ans. ETL means extraction, transformation and loading. It is basically the process of extracting data from source systems and transferring it to the data warehouse.

Q23. What does system security include?

Ans. It includes the mechanisms that control access and use of the database at the system level.

Q24. How does Oracle manage database security?]

Ans. Oracle manages database security using:

- Authentication,
- Authorisation,
- Access restrictions,
- Security policies, and
- Database Auditing.

Q25. What are the different integrity constraints supported by Oracle.

- NOT NULL
- UNIQUE KEY
- PRIMARY KEY
- FOREIGN KEY
- CHECK

Q26. What is a trigger?

Ans. It is a event driven procedure.

Q27. Name the different types of keys.

Ans.

- Primary
- Unique
- Foreign
- Referenced

Question Papers

MCS-043 : ADVANCED DATABASE DESIGN
December, 2006

Note : Question number 1 is **compulsory**. Attempt any **three** questions from the rest.

1. (i) Define multi-valued dependencies. Which multi-valued dependencies (MVDs) hold for the following relation? Explain. *5*

P_No	Colour	Size
P1	Red	Large
P1	Green	Large
P1	Red	Medium
P1	Green	Medium
P1	Red	Small
P1	Green	Small
P2	Black	Large
P2	Black	Small

In this relation each product comes in the range of colours and sizes.

Ans. multivalued dependency: A multivalued dependency is a full constraint between two sets of attributes in a relation.

In contrast to the functional dependency, the multivalued dependency requires that certain tuples be present in a relation. Therefore, a multivalued dependency is also referred to as a tuple-generating dependency. The multivalued dependency plays a role in the 4NF database normalization.

Formal definition

Let R be a relation schema and let $\alpha \subseteq R$ and $\beta \subseteq R$. The multivalued dependency $\alpha \rightarrow \beta$ holds on R if, in any legal relation r(R), for all pairs of tuples t1 and t2 in r such that t1[α] = t2[α], there exist tuples t3 and t4 in r such that

t1[α] = t2[α] = t3[α] = t4[α]

t3[β] = t1[β]

t3[R - β] = t2[R - β]

t4[β] = t2[β]

$t4[R - \beta] = t1[R - \beta]$[1]

The given relation holds – two
MVDS
P_No →→ colour
P_No →→ size

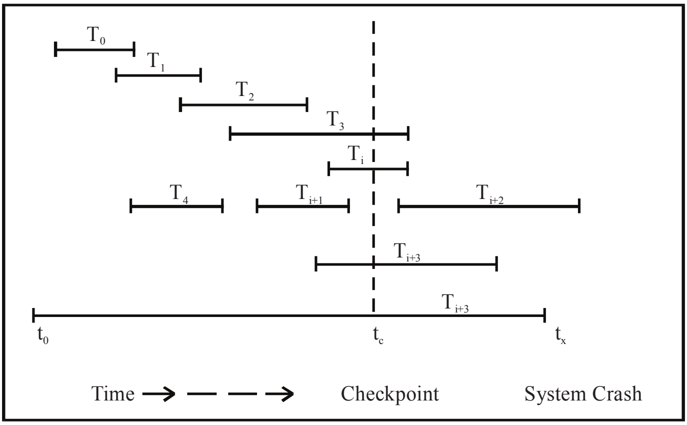
(ii) How is the checkpoint information used in the recovery operation following a system crash? ***5***

Ans. A scheme called checkpoint is used to limit the volume of information that have to be handled and processed in the event of a system failure involving the loss of volatile information.

Check Point Scheme ⇒ In the case of a system crash the log information being collected in buffers will be lost. A checkpoint operation performed periodically, copies log information into stable storage. The information and operations performed at each checkpoint consist of the following :

- A start of checkpoint record giving the identification that it is checkpoint along with the time and date of the checkpoint written to the log in a stable storage device.
- All log information from the buffers in the volatile storage is copied to the log on stable storage.
- All database update from the buffers in the volatile storage are propagated to the physical database.

Example : Let us consider a set of transactions with the checkpoint and system crash time, as shown below :-



A checkpoint is taken at time tc, which indicates that at that time all log and data buffers were propagated to storage.

The transactions T0, T1, T2, T4, were committed and their modifications are reflected in the database. With the checkpoint scheme, these transactions are not required to be redone after the system crash.

The transaction Ti and Ti+3 are to be redone at the point of checkpoint, where the transaction Ti+2 has to be undo, as no log has been written for this transaction.

(iii) Show the cost calculation for a simple hash-join operation. Make the necessary assumptions, if any. ***6***

Ans. Cost calculation for Simple Hash-Join

(i) Cost of partitioning r and s: all the blocks of r and s are read once and after partitioning written back, so cost 1 = 2 (blocks of r + blocks of s).

(ii) Cost performing the hash-join using build and probe will require at least one block transfer for reading the partitions

Cost 2 = (blocks of r + blocks of s)

(iii) There are a few more blocks in the main memory that may be used for evaluation, they may be read or written back. We ignore this cost as it will be too less in comparison to cost 1 and cost 2.

Thus, the total cost = cost 1 +cost 2.

= 3 (blocks of r + blocks of s)

Cost of Hash –Join requiring recursive partitioning :

(i) The cost of partitioning in this case will increase to number of recursion required, it may be calculated as:

Number of iteration required = ([log $_{M-1}$ (blocks of s] –1)

Thus, cost 1 will be modified as :

= 2 (blocks or r + blocks of s) × ([log $_{M-1}$ (block of s)] –1])

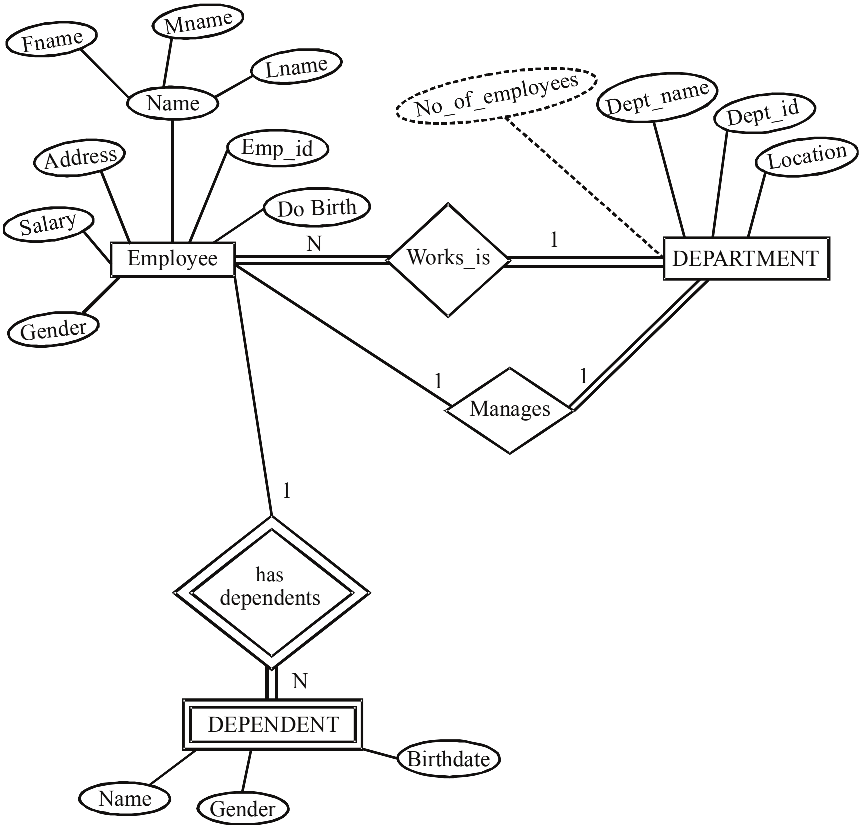
The cost for step (ii) and (iii) here will be the same as that given in steps (ii) and (iii) above.

Thus, total cost = 2 (blocks of r + blocks of s) ([log $_{M-1}$ (locks of s) –1]) + (blocks of r + blocks of s).

Because s is in the inner term in this expression, it is advisable to choose the smaller relation as the build relation. If the entire build input can be kept in the main memory, n can be set to 1 and the algorithm need not partition the relations but may still build an in-memory index, in such cases the cost estimate goes down to (Number of blocks r + Number of blocks of s).

(iv) A company database needs to store information about employees, departments and children of employees. Employees work in department each department is managed by an employee, we are not interested in information about a child once the parent leaves the company. Draw an E-R diagram that captures this informations. ***5***

Ans.



(v) How is UML related to E-R diagram? ***4***

Ans. Entity relationship model is a high level conceptual data model. It allows us to describe the data involved in a real world enterprise in terms of objects and their relationship. It is widely used to develop an initial design of a database.

Data models like E-R model still persists but systems built using object modeling techniques like UML are now preferred. Both UML and ER model focus on object as an implicit way to analyze, conceive and observe user requirements. Since UML is accepted by the objects management group (OMG) as the standard modeling object-oriented programme it becomes one of the automatic choices.

(vi) In timestamp-based concurrency control, transactions are assigned a timestamp at startup; how is it used to ensure serializability? How does the Thomas Write rule improve concurrency? *6*

Ans. Each transaction is issued a timestamp when it enters a system. If an old transaction T_i has a time-stamp TS(T_i), a new transaction T_j is assigned time-stamp TS(T_j) such that TS(T_i) < TS(T_j). This protocol manages concurrent execution in such a manner that the time-stamps determine the serialisability order. In order to ensure such behavior, the protocol needs to maintain for each data Q two time-stamp values.

- W- time-stamp (Q) is the largest time-stamp of any transaction that executed write (Q) successfully.
- R- time-stamp (Q) is the largest time-stamp of any transaction that executed read (Q) successfully.

The timestamp ordering protocol executes any conflicting read and write operations in timestamp order. Suppose a transaction Ti issues a read (Q).

1) If TS (T_i) <W –timestamp (Q) ,-> Ti needs to read a value of Q that was already overwritten. Action: reject read operation and rolled back $T_{i.}$

2) If TS (T_i) =W –timestamp (Q) ,-> It is OK. Action: Execute read operation and set R- timestamp (Q) to the maximum of R-timestamp (Q) and TS (T_i).

Suppose that transaction $T_{i.,}$ issues write (Q) :

1) If TS (Ti) < R – timestamp (Q), -> the value of Q that $T_{i.}$ is writing was used previously, this value should have never been produced. Action: reject the write operation. Roll back $T_{i.}$

2) If TS (Ti) < W – timestamp (Q), -> T_i is trying to write an obsolete value of Q. Action :Reject write operation and roll back $T_{i.}$

3) Otherwise, execute write operation and set W- timestamp (Q) to TS (Ti).

Thomas ‘Write Rule: Modified versions of the timestamp- ordering protocol in which obsolete write operations may be ignored under certain circumstances. When $T_{i.}$ Attempts to write data item Q, if TS(T_i) < W – timestamp(Q) , then T_i is attempting to write an obsolete value of {Q}. Hence , rather than rolling back $T_{i.}$ as the timestamp ordering protocol would have done, this { write} operation can be ignored. Otherwise this protocol is the same as the timestamp ordering protocol. Thomas 'Write Rule allows greater potential concurrency'.

(vii) What is XML and how does XML compare to HTML? *4*

Ans. The Extensible Markup Language (XML) is a general-purpose markup language. Its primary purpose is to facilitate the sharing of data across different information systems, particularly via the Internet. It is a simplified subset of the Standard Generalized Markup Language (SGML), and is designed to be relatively human-legible. By adding semantic constraints, application languages can be implemented in XML. These include XHTML, RSS, MathML, GraphML, Scalable Vector Graphics, MusicXML, and thousands of others. Moreover, XML is sometimes used as the specification language for such application languages. XML is recommended by the World Wide Web Consortium. It is a fee-free open standard. The W3C recommendation specifies both the lexical grammar, and the requirements for parsing.

HTML	XML
1. HTML only provides predefined tags, such as header, paragraph, heading etc. In other words, users cannot add to this predefined list of tags.	**1.** XML does not have predefined tags and tags can be defined using a Document Type Definition (DTD). It is possible to define tags for a particular domain or application.
2. HTML is mainly designed to represent a presentation structure of a document. It enables a browser to interpret and display the document to humans. This makes HTML, more suitable for machine-human interaction rather than machine-machine interaction.	**2.** XML is basically designed to represent the logical structure of documents. The tag names give abstract semantics to the associated data. Hence XML can be used more effectively for machine-machine interaction.
3. In traditional HTML, the content and presentation format/style are mixed (i.e. not separated).	**3.** In XML, the content and presentation format/style are separated. The former is written into an XML document, while the latter is specified in style sheers using XSL or CSS.
4. Search ing for and retrieval of specific information based on semantic content is difficult and is sensitive to minor alternative in the document, such as adding blank lines, etc.	**4.** Searching and retrieval of information is greatly simplified and not particularly sensitive to document layout and minor changes to it.
5. In general, HTML searches returns large amounts of data because of its poorly defined logical structure.	**5.** XML searches return data more precisely associated with the query because of its well-defined logical structure.

6. In HTML, links are unidirectional, i.e., it connects only two resources and a link has no semantics.	6. In XML, with Xlink, one can use bi-directional links, link up more than two resources and specify a "role" to the link by associating semantics with it.
7. Compared with XML, HTML has a relatively loose syntax. For example, some syntactical errors may be acceptable (i.e., the document may still be processed even if a syntactical error occurs).	7. The syntax for XML is strictly defied.

(viii) What is an integrity constraint? How are integrity constraints supported by Oracle? Briefly describe. *5*

Ans. An integrity constraint is a declarative way to define a business rule for a column of a table. An integrity constraint is a statement about table data that is always true and that follows these rules:

- If an integrity constraint is created for a table and some existing table data does not satisfy the constraint cannot be enforced.
- After a constraint is defined, if any of the results of a DML statement violate the integrity constraint, then the statement is rolled back, and an error is returned.

Integrity constraints are defined with a table and are stored as part of the table's definition in the data dictionary, so that all database applications adhere to the same set of rules. When a rule changes, it only need to be changed once at the database level and not for each application.

The following integrity constraints are supported by Oracle:

- NOT NULL : Disallows nulls (empty entries) in a table's column.
- UNIQUE : Disallows duplicate values in a column or set of columns.
- PRIMARY KEY : Disallows duplicate values and nulls in a column or set of columns.
- FOREIGN KEY : Requires each value in a column or set of columns to match a value in a related table's UNIQUE or PRIMARY KEY. FOREIGN Key integrity constraints also define referential integrity actions that dictate what Oracle should do with dependent data if the data references are altered.
- CHECK : Disallows values that do not satisfy the logical expression of the constraint.

2.(i) Briefly describe the architecture of distributed databases with the help of a diagram. ***10***

Ans. The architecture of distributed databases may be looked at as, an extension of ANSI SPARC three level architecture. The ANSI SPARC architecture has been defined for the centralized system and can be extended for the distributed database system with various level defined for data distribution as well.

Architecture defines the following additional schema for distributed database systems :

1) A set of global schemas that would define application interface. Thus, a global schema can be further divided into

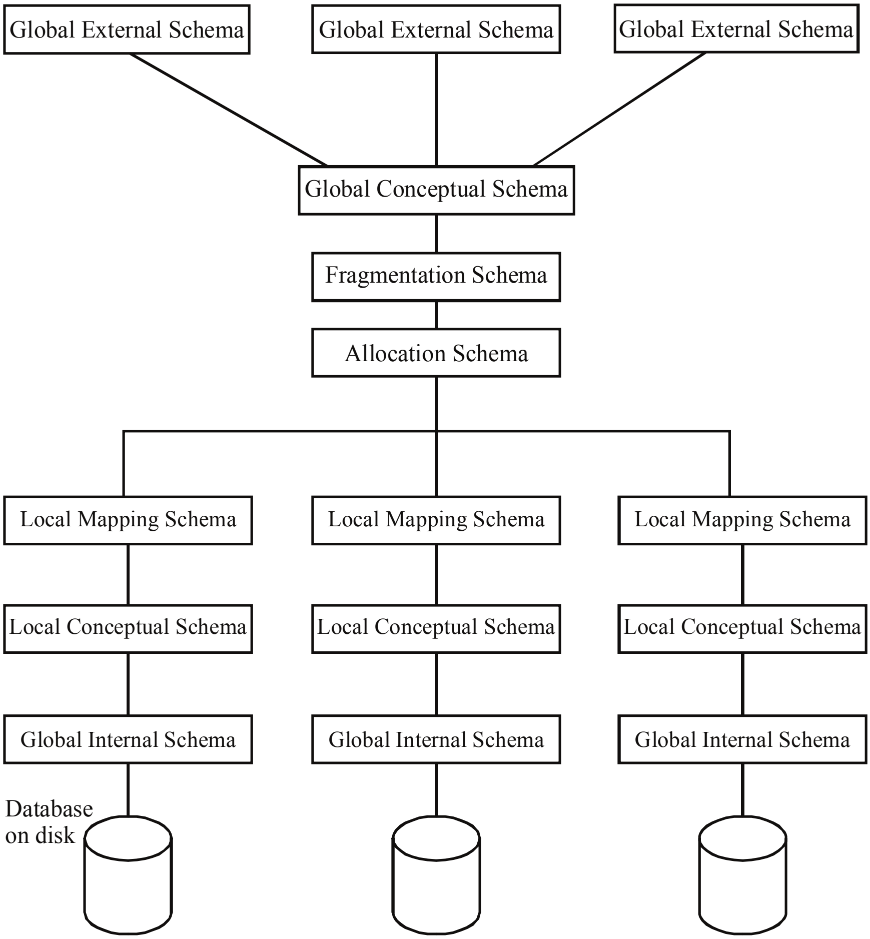
- Global external schema, and
- Global conceptual schema

2) Fragmentation and allocation schema will determine various fragments and their allocation to various distributed database sites.

3) Finally, it would require basic schemas as proposed by ANSI / SPARC architecture for local databases.

I. (a) Global External Schema : The basic objective of providing a global external schema is to define the external application interface for a distributed database application. This level is similar to the external schema as defined for the ANSI SPARC architecture. Thus, this interface must support the concept of data independence.

(b) Global Conceptual Schema : It defines logical database application as if there is, no distribution of data. Conceptually, this schema is almost similar to the conceptual level of ANSI / SPARC architecture. Thus, this schema defines the entities, relationship, and constraints including security and integrity constraints for global database application. This level is responsible for ensuring physical data independence between an application and the distribution environment where it is to be implemented.



Architecture for DDBMS

II. Fragmentation and allocation schema : It is required to describe the data fragments that have been created for the distributed database, along with, their replicas that have been allocated to various database sites. This, schema, thus, basically is for the identification of data distribution among the distributed sites.

III. Local schemas : The local schema at each site can consists of conceptual and internal schema. However, to map the fragmentation and allocation schema to local external objects may require local mapping schema. It is this schema (i.e. local mapping schema), that hides the details of various DBMSs, thus providing the DBMS independence.

(ii) How does the architecture of DDBMS differ from that of centralized DBMS? ***5***

Ans. If the data of an organization is stored at the single location under the strict control of the DBMS, then, it is termed as a centralized database system. A centralized system has the advantage of controlling data under a control authority, so, the chances of integrity failure less, inaccuracy are very low. In such a system it is easier to enforce business, rules and the system can also be made very secure.

A distributed database system consist of a collection of sites, each of which may participate in the execution of transactions, which access data at one site or several sites. The main difference between centralized and distributed database systems is that in the former, the data resides in one single centralized control, while in the latter, the data resides in several sites under the control of local distributed DBMS components which are under the control of one DBMS.

In a centralized system, the readability and availability is dependent on one system only and on the communication links. The load on the central system may be very high if multiple transactions are going on and results in delay.

In a DDBMS, all the above problems are solved to an extent, as most of the data access can new be local, however, it adds overhead on part of maintaining databases at several sites. The sites must coordinate if global query is to be handled. Thus, it will increase the overall complexity of the system.

(iii) Identify the functional dependencies which hold for the following relation: ***5***

P_No	Factory	Wareh ouse
P1	F1	WH1
P1	F2	WH1
P2	F2	WH1
P3	F2	WH2
P4	F2	WH2

The above relation indicates where each product is made and where it is stored. Each product may be produced at a number of different factories, but may only be stored at one warehouse.

Ans. P_No, factory → warehouse

3.(i) Make a comparison between OODBMS and object relational database. ***10***

Ans. An object oriented database management system is created on the basis of persistent programming paradigm whereas, a object relational is built by creating object oriented extensions of a relational system. In fact both the products have clearly defined objectives. The following table shows the difference among them:

Object Relational DBMS	Object Oriented DBMS
The features of these DBMS include: s Support for complex data types s Powerful query languages support through SQL. s Good protection of data against programming errors	The features of these DBMS include: s Supports complex data types, s Very high integration of database with the programming language. s Very good performance. s But not as powerful at querying as relational.
One of the major asset here is SQL. Although, SQL is not as powerful as a Programming Language, but it is none-the-less essentially a fourth generation language, thus, it provides excellent protection of data from the Programming errors.	It is based on object oriented programming languages, thus, are very strong in programming, however, any error of a data type made by a programmer may effect many users.
The relational model has a very rich foundation for query optimization, which helps in reducing the time taken to execute a query.	These databases are still evolving in this direction. They have reasonable systems in place.
These databases make the querying as simple as in relational even, for complex data types and multimedia data.	The querying is possible but somewhat difficult to get.
Although the strength of these DBMS is SQL, it is also one of major weaknesses from the performance point of view in memory applications.	Some applications that are primarily run in the RAM and require a large number of database accesses with high performance may find such DBMS more suitable. This is because of rich programming interface provided by such DBMS. However, such application may not support very strong query capabilities. A typical example of one such application is database required for CAD.

(ii) Describe an application scenario (in brief) for which you would choose an ORDBMS and explain why. ***6***

Ans. A relational database management system (RDBMS) that supports object classes as data types is called **object-relational database management**. The term "object relational" is also applied imprecisely by the database industry to any number of products (especially SQL-based ones) that have some object-oriented and pseudo-relational features. There is no formal definition for this commercial usage of the term.

An object-relational database (ORD) or object-relational database management system (ORDBMS) is a relational database management system that allows developers to integrate the database with their own custom data types and methods. The term *object-relational database* is sometimes used to describe external software products running over traditional DBMSs to provide similar features; these systems are more correctly referred to as object-relational mapping systems.

Whereas RDBMS or SQL-DBMS products focused on the efficient management of data drawn from a limited set of data types (defined by the relevant language standards), an object-relational DBMS allows software developers to integrate their own types and the methods that apply to them into the DBMS. The goal of ORDBMS technology is to allow developers to raise the level of abstraction at which they view the problem domain.

In an RDBMS, it would be fairly common to see SQL statements like this:

```
CREATE TABLE Customers  (
      Id              CHAR(12)          NOT NULL PRIMARY KEY,
      Surname         VARCHAR(32)       NOT NULL,
      FirstName       VARCHAR(32)       NOT NULL,
     DOB              DATE              NOT NULL
);
SELECT InitCap(Surname) || ‘, ‘ || InitCap(FirstName)
      FROM Customers
WHERE Month(DOB) = Month(getdate())
     AND Day(DOB) = Day(getdate())
```

Most current SQL databases allow the creation of custom functions, which would allow the query to be expressed as:

```
SELECT Formal(Id)
     FROM Customers
```

```
WHERE Birthday(Id) = Today()
```

In an object-relational database, one might see something like this, where the data types and expressions such as BirthDay() are user-defined.

```
  CREATE TABLE Customers (
    Id              Cust_Id             NOT NULL  PRIMARY KEY,
    Name            PersonName          NOT NULL,
    DOB             DATE                NOT NULL
  );
SELECT Formal( C.Name )
    FROM Customers C
   WHERE BirthDay ( C.DOB ) = TODAY;
```

Another advantage to the object-relational model is that the database can make use of the relationships between data to easily collect related records. In an address book application, an additional table would be added to the ones above to hold zero or more addresses for each user. Using a traditional RDBMS, collecting information for both the user and their address requires a “join”:

```
SELECT InitCap(C.Surname) ‖ ‘, ‘ ‖ InitCap(C.FirstName), A.city
      FROM Customers C, Addresses A
WHERE A.Cust_Id=C.Id — the join
      AND A.city=”New York”
```

The same query in an object-relational database is much simpler:

```
SELECT Formal( C.Name )
     FROM Customers C
WHERE C.address.city=”New York” — the linkage is ‘understood’
```

Many of the ideas of early object-relational database efforts have largely been added to SQL:1999. In fact, any product that adheres to the object-oriented aspects of SQL:1999 could be described as an object-relational database management product. For example, IBM’s DB2, Oracle database, and Microsoft SQL Server, make claims to support this technology and do so with varying degrees of success.

Applications scenario for an ORDBMS is better choice are :

Some application requires large volume of data to be handled. These application even requires that proper user should only be provided with proper data. If we have constructed a data structure then it should be shared whenever required. The following application normally requires that object oriented concepts to improve performance.

Engineering Design Databases : Computer Aided Design and Manufacturing applications use engineering design databases. The complex objects can be recursively partitioned into smaller objects. These objects posses different levels of data abstraction. This could be easily handled by object oriented approach.

Multimedia Databases : Now a days the office application uses various kinds of data such as text, video and sound etc. These data are called multimedia data. These multimedia data are stored as a sequence of bytes with variable length, and segments of data are linked together for easy reference. Thus object oriented approach supports up to a great extent to handle these data.

Expert System : The Artificial Intelligence and Expert system uses knowledge base. The data are in the form of knowledge. The representation of knowledge can be done using various techniques. The data to be stored are more complex. The rules are inserted into the knowledge bases. These knowledge bases can easily be handled by using object oriented approach.

(iii) Consider the following relations:
Society (S_id, S_name, S_cause, S_date,S_address)
Member (M_id, M_name, M_phone ,M_address)
Event (E_id, E_desc, S_id, M_id, E_date)
Write the relational algebraic queries for the following ; *4*

(a) List the event details by the societies for the cause of "AIDS"

Ans. π_{E_dosc} $(^{\sigma}\text{s\_cause} = \text{'AIDS'}$ (Society $\bowtie$ Event)).

(b) Show the details of all the members who have been involved in any event organized by the society whose name is "XYZ".

Ans. $\pi_{M\_id,\ M\_name\ M\_phone,\ M\_addres}$ $(^{\sigma}\text{s\_name} = \text{'xyz'}$ (society $\bowtie$ Event $\bowtie$ Member))

4.(a) What is clustering in data mining and how is it different from classification? Briefly describe the Nearest Neighbour Clustering algorithm. *15*

Ans. Clustering is grouping things with similar attribute values into the same group. Given a database D={ $t_1,t_2,\dots\dots,t_n$} of tuples and an integer value K, the clustering problem is to defined as mapping where each tuple t_i is assigned to one cluster K_j, 1<=j<=k.

A cluster K_j , contains precisely those tuples mapped to it. Unlike the classification problem, clusters are not known in advance. The user has to enter the value of the number of clusters k.

In other words a cluster can be defined as the collection of data objects that are similar in nature, as per certain defining property, but these objects are dissimilar to the objects in other clusters.

However , **classification** task maps data into predefined groups or classes. Given a database D={ $t_1,t_2,\dots\dots,t_n$} and a set of classes C={ $C_1,\dots\dots,C_m$}, the classification problem is to define a mapping f:D->C where each t_i, is assigned to one class, that is , it divides database D into classes specified in the Set C.

Examples of classification :

- Teachers classify students marks data into a set of grades as A, B, C, D, or F.
- Classification of the height of a set of persons into the classes tall, medium, or short.

Clustering is a very useful exercise especially for identifying similar groups from the given data. Such data can be about buying patterns, geographical locations, web information and many more.

Examples of Clustering :

- To segment the customer database of a departmental store based on similar buying patterns.
- To identify similar web usage patterns etc.

Nearest Neighbour Clustering

In this approach, items are iteratively merged into the existing clusters that are closest.

It is an incremental method the threshold , t , used to determine if items are added to existing clusters or a new cluster is created. This process continues until all patterns are labeled or no additional labeling occurs.

Algorithm of Nearest Neighbour Clustering is as follows

Input :

```
  $D = \{ t_1,t_2,\ldots\ldots,t_n\}$   //set of elements
  A                    //adjacency matrix showing distance between elements
  k                    //number of desired clusters
```

Output :

```
    K                  //set of clusters
Nearest Neighbour Clustering   :
$K_1=\{ t_i \};$
$K=\{ K_1 \};$
$k=1;$
for i=2 to n do
Find the $t_m$ in some cluster $K_m$ in K such that distance $(t_i , t_m)$ is the smallest;
If dis$(t_i , t_m) <= t$ then
$K_m= K_m \cup t_i$ ;
Else
$k=k+1;$
$k_k=\{t_i\};$
```

(b) What is a mechanism for deadlock detection ? Explain with the help of a diagram. ***5***

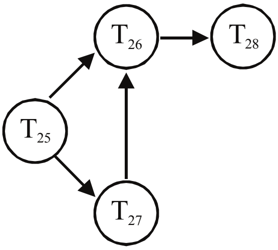
Ans. Deadlock Detection

Deadlocks can be described precisely in terms of a directed graph called a **wait for graph.** This graph consists of a pair $G = (V, E)$, where V is a set of vertices and E is a set of edges. The set of vertices consists of all the transaction in the system. Each element in the set E of edges is an ordered pair $T_i \rightarrow T_j$. If $T_i \rightarrow T_j$ is in E, then there is a directed edge from transaction T_i to T_j, implying that transaction T_i is waiting for transaction T_j to release a data item that it needs.

When transaction T_i requests a data item currently being held by transaction T_j then the edge $T_i \rightarrow T_j$ is inserted in the wait-for graph. This edge is remove only when transaction T_j is no longer holding a data item needed by transaction T_i.

A deadlock exists in the system if and only if the wait-for graph contains a cycle. Each transaction involved in the cycle is said to be deadlocked. To detect deadlocks, the systems needs to maintain the wait-for graph, and periodically to invoke an algorithm that searches for a cycle in the graph.

To illustrate these concepts, consider the wait-for graph in figure (1), which depicts the following situation:



Wait-for graph with no cycle

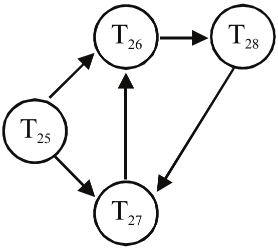
- Transaction T_{25} is waiting for transactions T_{26} and T_{27}.
- Transaction T_{27} is waiting for transactions T_{26}.
- Transaction T_{26} is waiting for transactions T_{28}.

Since the graph has no cycle, the system is not in a deadlock state.

Suppose now that transaction T_{28} is requesting an item held by T_{27}. The edge $T_{28} \rightarrow T_{27}$ is added to the wait-for graph, resulting in the new system state in figure (2). This time, the graph contains the cycle.

$$T_{26} \rightarrow T_{28} \rightarrow T_{27} \rightarrow T_{26}$$

implying that transactions T_{26}, T_{27} and T_{28} are all deadlock.



Wait-for graph with a cycle

If deadlocks occur frequently, then the detection algorithm should be invoked more frequently than usual. Data items allocated to deadlocked transactions will be unavailable to other transactions until the deadlock can be broken. In addition, the number of cycles in the graph may also grow. In the worst case, we would invoke the detection algorithm every time a request for allocation could not be granted immediately.

5. Explain the following terms :

(i) Object identity

Ans. An identity in object-oriented programming, object-oriented design and object-oriented analysis describes the property of objects that distinguishes them from other objects. This is closely related to the philosophical concept of identity.

Identity is closely associated with the location that objects are stored in. Some programming languages implement object identity by using the location of the object in computer memory as the mechanism for realizing object identity. However, objects can move from one place to another without change to their identity, and can be stored in places other than computer memory, so this does not fully characterize object identity

Identity of objects allows objects to be treated as black boxes. The object need not expose its internal structure. It can still be referred to, and its other properties can be accessed via its external behaviour associated with the identity. The identity provides a mechanism for referring to such parts of the object that are not exposed in the interface. Thus, identity is the basis for polymorphism in object-oriented programming

(ii) Serializability

Ans. In databases and transaction processing, serializability is the property of a schedule being serializable. It means equivalence (in its outcome, the resulting database state, the values of the database's data) to a serial schedule (serial schedule: No overlap in two transactions' execution time intervals; consecutive transaction execution). It relates to the isolation property of a transaction, and plays an essential role in concurrency control. Transactions are usually executed concurrently since their serial executions are typically extremely inefficient and thus impractical.

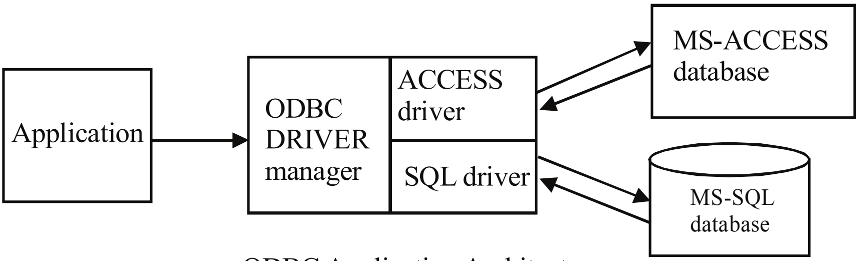
Serializability is the major criterion for the correctness of concurrent transactions' executions (i.e., transactions that have overlapping execution time intervals, and possibly access same shared resources), and a major goal for concurrency control. As such it is supported in all general purpose database systems. The rationale behind it is the following: If each transaction is correct by itself, then any serial execution of these transactions is correct. As a result, any execution that is equivalent (in its outcome) to a serial execution, is correct. Schedules that are not serializable are likely to generate erroneous outcomes. Well known examples are with transactions that debit and credit accounts with money. If the related schedules are not serializable, then the total sum of money may not be preserved. Money could disappear, or be generated from nowhere. These violations of possibly needed other invariant preservations are caused by one transaction writing, and "stepping on" and erasing what has been written by another transaction before it has become permanent in the database. It does not happen if serializability is maintained

(iii) ODBC

Ans. (iii) Open Database Connectivity (ODBC) is a standard software API specification for using database management systems (DBMS). ODBC is independent of programming language, database system and operating system. ODBC was created by the SQL Access Group and first released in September, 1992. ODBC is based on the Call Level Interface (CLI) specifications from SQL, X/Open (now part of The Open Group), and the ISO/IEC.

The ODBC API is a library of ODBC functions that let ODBC-enabled applications connect to any database for which an ODBC driver is available, execute SQL statements, and retrieve results.

The goal of ODBC is to make it possible to access any data from any application, regardless of which database management system (DBMS) is handling the data. ODBC achieves this by inserting a middle layer called a database driver between an application and the DBMS. This layer translates the application's data queries into commands that the DBMS understands.



ODBC Application Architecture

(iv) OLAP

Ans. On Line Analytical Processing, or OLAP, is an approach to quickly providing answers to analytical queries that are multidimensional in nature. OLAP is part of the broader category business intelligence, which also includes Extract transform load (ETL), relational reporting and data mining. The typical applications of OLAP are in business reporting for sales, marketing, management reporting, business process management (BPM), budgeting and forecasting, financial reporting and similar areas. The term OLAP was created as a slight modification of the traditional database term OLTP (On Line Transaction Processing).

Databases configured for OLAP employ a multidimensional data model, allowing for complex analytical and ad-hoc queries with a rapid execution time. Nigel Pendse has suggested that an alternative and perhaps more descriptive term to describe the concept of OLAP is Fast Analysis of Shared Multidimensional Information (FASMI). They borrow aspects of navigational databases and hierarchical databases that are speedier than their relational kin.

The output of an OLAP query is typically displayed in a matrix (or pivot) format. The dimensions form the row and column of the matrix; the measures, the values.

(v) Deductive database

Ans. A deductive database system is a database system which can make deductions (i.e.: infer additional rules or facts) based on rules and facts stored in the (deductive) database. Deductive database systems:

Mainly deal with rules and facts.

Use a declarative language (such as Prolog) to specify those rules and facts.

Use an inference engine which can deduce new facts and rules from those given.

A good example of a declarative language would be Prolog, but for databases Datalog is used more often. Datalog is both a syntactic subset of prolog and a database query language – it is designed specifically for working with logic and databases. Deductive databases are also known as logic databases, knowledge systems and inferential databases. The problem domain of an expert system / deductive database is usually quite narrow. Deductive databases are similar to expert systems - “traditional” expert systems have assumed that all the facts and rules they need (their knowledge base) will be loaded into main memory, whereas a deductive database uses a database (usually on disk storage) as its knowledge base. Traditional expert systems have usually also taken their facts and rules from a real expert in their problem domain, whereas deductive databases find their knowledge inherent in the data. Deductive databases and expert systems are mainly used for:

- Replicating the functionality of a real expert.
- Hypothesis testing.
- Knowledge discovery (finding new relationships between data).

MCS-043 : ADVANCED DATABASE DESIGN
June, 2007

Note : Question number 1 is **compulsory**. Attempt any **three** questions from the rest.

1. (a) What is multivalued dependency? How is 4NF related to multivalued dependency? Is 4NF dependency preserving in nature? Justify your answer. *4*

Ans. Multivalued dependencies are a consequence of first normal form (1NF), which disallowed an attribute in a tuple to have a set of *values.* If we have two or more multivalued *independent* attributes in the same relation schema, we get into a problem of having to repeat every value of one of the attributes with every value of the other attribute to keep the relation state consistent and to maintain the independence among the attributes involved. This constraint is specified by a multivalued dependency.

Formal Definition of Multivalued Dependency

A **multivalued dependency** (**MVD**) $\chi \twoheadrightarrow$ Y specified on relation schema R, where χ and Y are both subsets of R, specifies the following constraint on any relation state r of R: If two tuples t_1 and t_2 exist in r such that $t_1[\chi] = t_2[\chi]$, then two tuples t_3 and t_4 should also exist in r with the following properties, where we use Z to denote $(R - (\chi \text{ U} Y))$:

$t_3[\chi]=t_4[\chi] = t_1[\chi]=t_2[\chi]$.

$t_3[\gamma]=t_1[\gamma]$ and $t_4[\gamma]=t_2[\gamma]$. where the t_1, t_2, t_3, and t_4 are not necessarily

$t_3[Z]=t_2[Z]$ and $t_4[Z]=t_1[Z]$,

distinct.

(a)

EMP

ENAME	PNAME	DNAME
Sanjay	X	Johny
Sanjay	Y	Anita
Sanjay	X	Anita
Sanjay	Y	Johny

(b)

EMP_PROJECTS

ENAME	PNAME
Sanjay	X
Sanjay	Y

EMP_DEPENDENTS

ENAME	DNAME
Sanjay	Johny
Sanjay	Anita

(c)

SUPPLY

SNAME	PARTNAME	PROJNAME
Sanjay	Bolt	ProjX
Sanjay	Nut	ProjY
Adamearth	Bolt	ProjY
Winston	Nut	ProjZ
Adamearth	Nail	ProjX
Adamearth	Bolt	ProjX
Sanjay	Bolt	Projy

R1

SNAME	PARTNAME
Sanjay	Bolt
Sanjay	Nut
Adamearth	Bolt
Winston	Nut
Adamearth	Nail

R2

SNAME	PROJNAME
Sanjay	ProjX
Sanjay	ProjY
Adamearth	ProjY
Winston	ProjZ
Adamearth	ProjX

R3

PARTNAME	PROJNAME
Bolt	ProjX
Nut	ProjY
Bolt	ProjY
Nut	ProjZ
Nail	ProjX

Figure : Fourth and fifth normal form. (a) The EMP relation with two MVDs:

ENAME $\twoheadrightarrow$ PNAME and ENAME $\twoheadrightarrow$ DNAME. (b) Decomposing EMP into two relations in 4NF. (c) The relation SUPPLY with no MVDs satisfies 4NF but does not satisfy 5NF if the JD (R1, R2, R3) holds. (d) Decomposing SUPPLY into three 5NF relations.

Whenever $\chi \twoheadrightarrow Y$ holds, we say that χ multidetermines Y. Because of the symmetry in the definition, whenever $\chi \twoheadrightarrow Y$ holds in R, so does $\chi \twoheadrightarrow Z$. Hence, $\chi \twoheadrightarrow Y$ implies $\chi \twoheadrightarrow Z$, and therefore it is sometimes written as $\chi \twoheadrightarrow Y \mid Z$

Relation of 4NF with Multivalued Dependency:

A relation schema R is in 4NF with respect to a set D of functional and multivalued dependencies if for all multivalued dependencies in D^+ of the form $a \rightarrow\rightarrow b$, where $a \subseteq R$ and $b \subseteq R$, at least one of the following hold:

- $a \rightarrow\rightarrow b$ is trivial (i.e., $b \subseteq a$ or a $a \cup b = R$)
- a is a superkey for schema R

Is 4NF dependency preserving in Nature?

Yes, 4NF is dependency preserving.

(b) What are Assertions? What is the utility of assertions? How are Assertions different from Views? Describe both Assertions and Views with the help of example. ***6***

Ans. Refer to Chapter-4, Q.No.-1, Page No.-48

(c) What is UML? How does UML have an edge over other database designing tools? With the help of an example, describe the designing of database by using a UML class diagram. ***6***

Ans. UML is a standard modeling language used for modeling software systems of varying complexities. System can range from enterprise information systems to distributed web based systems.

Since UML is accepted by the Object Management Group (OMG) as the standard for modeling object-oriented programs it becomes one of the automatic choices. UML has quickly become one of the popularly used tools for modeling business and software application needs. The UML became popular due to the following reasons:

(1) It is very flexible. It allows for many different types of modeling. Some of them include:

- Business process modeling event workflow.
- Sequencing of events,
- Defining database architecture etc.

(2) It is language and platform-independent. It allows software architects to model any application, on any operating system in any programme language or network.

(3) UML, supports object-oriented methodologies. Even if the concepts are based on object-oriented methods, the resulting structure and behaviour of UML can be used to design both relational and object-oriented models.

(4) The integration of various SDLC stages through UML tools has brought the analysts, modeler, designers, and the software application developers closer to each other.

UML Class Diagrams and Database Design

The class diagram sometimes may be considered as an alternative notation to E-R diagram. In UML class diagram contains:

- 'Class' – is displayed as a box. It has three sections"

  "Top Section" – Class Name
  " Middle Section" – Attributes
  "Last Section" – Operations
- Associations : Relationship Types are called Associations.
- Links : Relationship instances are called Links.
- Binary Association : Represented as a line connecting the participating classes.
- Linked Attributed: Connected to the Association's line by dashed line.
- Multiplicities : The (Minimum and Maximum) constraints.
- Reflexive Association : A Recursive Relationship
- Aggregation: To represent a relationship between a whole object and its component parts.
- Different unidirectional and bi-directional Associations.

The specialisation/generalization can be represented in UML class diagrams as :

- Basic notation is to connect the subclass by vertical lines to horizontal lines.
- A blank triangle indicates a specialisation/generalization with the disjoint constraint and a filled triangle indicates an overlapping constraint.
- The root super class is known as the base class and the leaf nodes are called terminal nodes.

Let us explain the role of class diagram with generalization/specialisation hierarchy in a database structure design with the help of an example. Consider the following UML class diagram for a University.

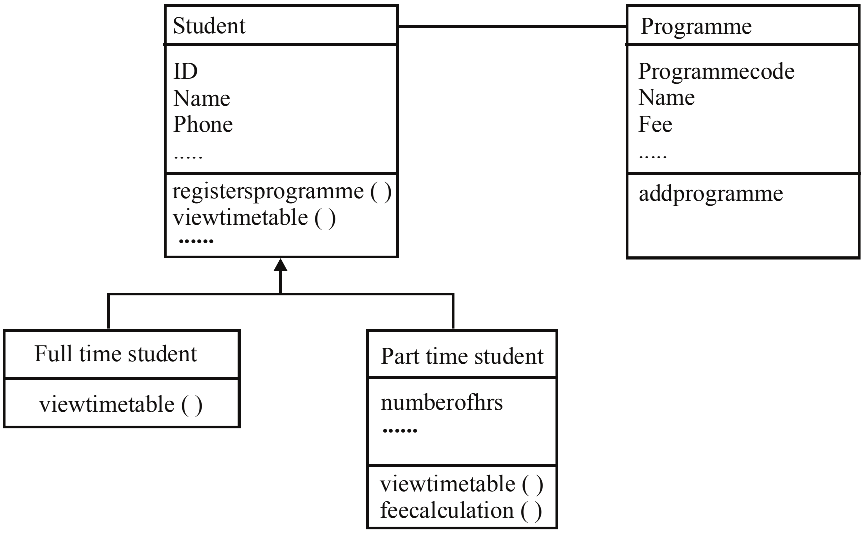


Figure : UML class diagram

Please note that in the diagram above, you can make clear-cut table design. One such possible design may be:
STUDENT (ID, Name, Phone, type (PT, FT))
PT (ID, numberofhrs)
PROGRAMME (Programmecode, Name, Fee)
Stuprog (ID, Programmecode)
You can also identify several functions/data processing functions/triggers relating to functions of classes. For example, feecalculation( ) may need to be implemented as a stored procedure while calculating the fee of part-time students, whereas addprogramme( ) is a simple data entry function. Thus, UML Class diagram with generalization/ specialisation is a very useful database design tool.

(d) With the help of a block diagram describe the phases of Query processing. How do we optimize a Query under consideration? Does Query optimisation contribute to the measurement of Query cost? Support your answer with suitable explanation. ***8***

Ans. Refer to Chapter-5, Q.No.-1, Page No.-68

(e) What do you mean by transaction? What are the properties of a transaction? Violation of which property causes Lost Update problem? Justify your answer with suitable example. ***5***
Ans. Refer to Chapter-5, Q.No.-15, Page No.-77

(f) Access control is the feature of which category of language (DSDL, DDL, DML, DCL)? At which level of ANSI-SPARC 3 level architecture does Access control work? Describe the commands used for Access control. Give an example in support of operation of each command. ***6***
Ans. Access control is the feature of data control language. It works on external level of ANSI–SPARC 2 level architecture the 2 commands for it are:
(1) Grant
(2) Revoke

Grant Command : SQL is also used in multiuser environments. The GRANT command is used to permit users access to the database. The syntax for GRANT command is:

```
GRANT <privilege_name> | ALL
ON <object> TO <user | PUBLIC>
[WITH GRANT OPTION];
```

<privilege_name> specifies the actions such as INSERT, DELETE, UPDATE, SELECT, etc. the user can perform on the <object>. The clause ALL indicates all the privileges.

<user> specifies the name of the user to whom the privileges have been granted. To grant the privileges to all the uses PUBLIC option can be used.
[WITH GRANT OPTION] is optional.
A user having privileges can pass on his privileges to another user by using the GRANT command. The recipient user can then grant the same privileges to some other user. For example, when an owner of a table grants privilege on the table to another user B, the privilege can be given with or without Grant Option. If the [WITH GRANT OPTION] is given, this means that B can also grant privilege to other user, else B cannot grant privilege to other users.

For example a query to provide SELECT permission on the DEPARTMENT table to user 'anita' and allow her to give further grants will be as follows:

```
GRANT SELECT ON DEPARTMENT
TO anita WITH GRANT OPTION;
```

We can also grant privileges on individual columns in a table. For example, to give user 'anita' , the privilege to update the column EMP_NAME, we will raise the following query:

```
GRANT UPDATE (EMP_NAME)
ON EMPLOYEE TO anita;
```

Revoke Command

The Revoke command is used to cancel database privileges from user(s). The syntax for revoke command is

```
REVOKE <privilege_name> | ALL
ON <object> FROM <user | PUBLIC>;
```

For example a query to revoke the privilege we gave to user 'anita' will be as follows:

```
REVOKE SELECT ON DEPARTMENT
FROM anita;
```

(g) Describe the reference architecture of a distributed DBMS, with the help of a block diagram. ***5***

Ans. Refer to Dec-2006, Q.No-2, Page No.-168

2. (a) Compare and contrast the following: ***8***

(i) Inclusion dependencies and Template dependencies

Ans. An inclusion dependency R.X<S.Y between two sets of attributes – X of a relation scheme R, and Y of a relation schema LS – is defined as the following constraint:

If r and s are the relation state of R and S respectively at any specific time then:

$_x(r(R)) \subseteq _y (s(S))$

The subnet relationship does not necessarily have to be a proper subnet. The sets of attributes on which the inclusion dependency is specified viz. X of R and Y of S above, must have the same number of attributes. In addition, the domains for each corresponding pair of attributes in X and Y should be compatible. The objectives of inclusion Dependencies are to formalise two important types of interrelation constraints that exist between the relations, thus cannot be expressed using FDs and MVDs that are:

(1) Referential integrity constraints,
(2) Class / subclass relationships
The common rules for making inferences from defined inclusion dependencies (Inclusion Dependency Inference Rule – IDIR) are:
IDIR1 (reflexivity): R.X< R.X
IDIR2 (attribute correspondence): if R.X < S.Y

Here
X = {A1, A2, ……., An} and
Y = {B1, B2, ….., Bn} and
Ai correspondence to Bi for 1 in

IDIR3 (transitivity): if R.X < S.Y and S.Y < T.Z then R.X < T.Z

Template Dependencies

The Template dependencies are the more general and natural class of data dependencies that generalizes the concepts of JDs. A template dependency is representation of the statement that a relation is invariant under a certain tableau mapping. Therefore, it resembles a tableau. It consists of a number of hypothesis rows that define certain variables with a special row at the bottom, called the conclusion row. A relation r satisfies a template dependency, if and only if, a valuation (say ρ)that successfully maps the hypothesis rows to tuples in a relation r, finds a map for conclusion row to a tuple in r.

Definition : A template dependency (TD) on a relation scheme R is a pair T = (T, w) where T={w1, w2, …..wk} is a set of hypothesis rows on R, and w is a single conclusion row on R. A relation r(R) satisfies TD T if for every valuation ρ of T such that $\rho(T) \subseteq$ r, ρ can be extended to show that $\rho(w) \in$ r. A template dependency is trivial if every relation over R satisfies it.

(ii) XML and HTML

Refer to Dec-2006, Q1(vii), Page No.-166

(iii) Centralised 2PL and Distributed 2PL

Ans. Centralised 2PL

As the name of this protocol suggests that in this protocol a single site known as the central site maintains the entire locking information. Therefore, it has only one lock manager that grants and releases locks. There is only one lock manager, for the entire distributed DBMS that can grant and release locks. This scheme can be modified by making the transaction manager responsible for making all the lock requests rather than the sub transaction's local transaction manager. Thus, the centralised lock manager needs to talk to only the transaction coordinator.

One of the major advantage of the centralised 2PL is that this protocol can detect deadlocks very easily as there is only one lock manager. However, this lock manager often, becomes the bottleneck in the distributed database and may make a system less reliable. This is, due to the fact, that in such a case, for the system to function correctly it needs to depend on a central site meant for lock management.

Distributed 2PL

The basic objective of this protocol is to overcome the problem of the central site controlling the locking. It distributes the lock managers to every site such that each the lock manager is responsible for managing the locks on the data items of that site. In case there is no replication of data, this protocol, thus, will be identical to the primary copy locking.

Distributed 2PL is a Read–One–Write–All type of protocol. It means that for reading a data item any copy of the replicated data may be read (as all values must be consistent), however, for any update all the replicated copies must be exclusively locked. This scheme manages the locks in a decentralised manner. The disadvantage of this protocol is that deadlock handling is very complex, as it needs to be determined by information from many distributed lock managers. Another disadvantage of this protocol is the high communication cost while updating the data, as this protocol would require locks on all the replicated copies and updating all of them consistently.

(iv) Partitioning Clustering and Nearest Neighbour Clustering
Ans.

	Partition Clustering	Nearest Neighbour Clustering
Technique or working	These techniques start with a randomly chosen or user-defined clustering, then optimize the clustering according to some validity measurement.	Nearest neighbor techniques are fairly intuitive in that they grow clusters by continually adding the nearest neighbor of a cluster to that cluster, provided that the distance falls below a specified threshold.
Distinction from other technique	It produce single partition.	An important distinction from many other techniques is that the distance measurement may not be metric.
Number of partition	Partitional clustering techniques produce a single partition of the data, not a structure like a dendrogram.	Nearest Neighbour produces many partition of data.
Implementation	Easy to implement	Less easy than Partition Clustering
Efficiency	Very efficient	Less efficient
Disadvantage	**1)** User must choose beforehand exactly how many clusters are desired **2)** The final results can vary depending upon the initial clustering. In fact, it may be desirable to run the algorithm several times with a different initial setup and choose the best resulting partition.	**1)** Expensive for large data set

(b) Describe the architecture of Datawarehouse with the help of a block diagram. Briefly discuss the role of each component of the datawarehouse architecture. *7*

Ans. Refer to Chapter-6, Q.No.-39, Page No.-125

(c) What do you mean by Data mining? How does Database processing differ from Data mining processing? *5*

Ans. Refer to Chapter-6, Q.No.-131

3. (a) What do you mean by clustering? Briefly describe the concept of Hierarchical clustering. How is clustering related to Data mining? *8*

Ans. Refer to Chapter-6, Q.No.-134

(b) Write short notes on the following: *6*

(i) JDBC

Ans. Java Database Connectivity (JDBC) provides a standard API that is used to access databases, regardless of the DBMS, through JAVA. There are many drivers for JDBC that support popular DBMSs. However, if no such driver exits for the DBMS that you have selected then, you can use a driver provided by Sun Microsystems to connect to any ODBC compliant database. This is called JDBC to ODBC Bridge. For such an application, you may need to create, an ODBC data source for the database before, you can access it from the Java application.

Connecting to a Database : In order to connect to a database, let us say an Oracle database, the related JDBC driver has to be loaded by the Java Virtual Machine class loader successfully.

```
// Try loading the Oracle database driver
try
{
Class.forName("oracle.jdbc.driver.OracleDriver").newInstance( );
}
catch (ClassNotFoundException ce)//driver not found
{
System.err.println ("Driver not found");
// Code to handle error
}
```

Now, you can connect to the database using the driver manage class that select the appropriate driver for the database. In more complex applications, we may use different drivers to connect to multiple databases. We may identify our database using a URL, which helps in identifying the database. A JDBC URL starts with "jdbc:" that indicates the use of JDBC protocol.

A sample database URL may be

```
jdbc:oracle@db.gullybaba.com:2000:student
```

To connect to the database, we need to connect with a username and password. Assuming it to be "username" and "password", the connection string would be:

```
// Try creating a database connection
```

```
connection db_conn=
DriverManager.getConection
          (jdbc:oracle@db.gullybaba.com:2000:student,
"username","password")
```

Thus, you will now be connected. Now the next step is to execute a query.

You can create a query using the following lines:

```
          Statement stmt =db_conn.CreateStatement( )
try {
          stmt.executeQuery (
                    // Write your SQL query using SQL and host language
          )
}catch (……….) {….}
stmt.close ( )
db_conn.close ( )
```

Thus, the JDBC standard allows you to handle databases through JAVA as the host language.

(ii) Spatial Databases

Ans. A spatial database keeps track of an object in a multi–dimensional space. A spatial database may be used to represent the map of a country along with the information about railways, roads, irrigation facilities, and so on. Such application are known as Geographic Information Systems (GILS).

The idea of a geographic database is to provide geographic information: therefore they are referred to also the Geographic Information System (GIS). A GIS is basically a collection of the Geographic information of the world. This information is stored and analyzed on the basis of data stored in the GIS. The data in GIS normally defines the physical properties of the geographic world, which includes:

- spatial data such as political boundaries, maps, roads, railways, airways, rivers, land elevation, climate etc.
- non-spatial data such as population, economic data etc.

(iii) PJNF

Ans. PJNF is defined using the concept of the join dependencies. A relation schema R having a set F of functional, multivalued, and join dependencies, is in PJNF (5 NF), if for all the join dependencies in the closure of F (referred to as F+) that are of the form

- (R1, R2,…,Rn), where each $Ri \subseteq R$ and $R = R1 \cup R2 \cup .... \cup Rn,$ at least on the following holds:
- * (R1, R2, . . .,Rn) is a trivial join dependency.
- Every Ri is a superkey for R.

PJNF is also referred to as the Fifth Normal From (5NF).

Let us first define the concept of PJNF from the viewpoint of the decomposition and then refine it later to a standard form.

Definition 1 : A JD* [R1, R2,. . ., Rn] over a relation R is trivial if it is satisfied by every relation r(R).
The trivial JDs over R are JDs of the form *[R1, R2, . . . , Rn] where for some i the Ri = R.

Definition 2 : A JD *[R1, R2,…,Rn] applies to a relation scheme R if R = R1 R2….Rn.

Definition 3: Let R be a relation scheme having F as the set of FDs and JDs over R. R will be in project-joint normal form (PJNF) if for every JD *[R1, R2, . . . Rn] which can be derived by F that applies to R, the following holds:

- The JD is trivial, or
- Every Ri is a super key for R.

For a database scheme to be in project-join normal form, every relation R in this database scheme should be in project-join normal form with respect to F.

(c) What is Datagrid? What is the requirement of datagrid (give at least 3 requirements)? Describe the structure of datagrid with the help of a block diagram. ***6***

Ans. Refer to Chapter-7, Q.No.-20, Page No.-146

4. (a) How does PostgreSQL perform storage and indexing of tables? Briefly discuss the type of indexes involved in PostgreSQL? ***7***

Ans. Refer to Chapter-7, Q.No.-27, Page No.-148

(b) What do you mean by tuning of SQL? What are the steps involved in tuning of SQL? Discuss each step briefly. *6*

Ans. An important facet of database system performance tuning is the tuning of SQL statements. SQL tuning involves three basic steps:

- Identifying high load or top SQL statements that are responsible for a large share of the application workload and system resources, by reviewing past SQL execution history available in the system.
- Verifying that the execution planes produced by the query optimizer for these statements perform reasonably.
- Implementing corrective actions to generate better execution plans for poorly performing SQL statements.

These three steps are repeated until the system performance reaches a satisfactory level or no more statements can be tuned.

Let us consider the Oracle to explain the tuning of SQL.

SQL tuning : Oracle SQL tuning is a phenomenally complex subject.

The goals of SQL tuning focus on improving the execution plan to fetch the rows with the smallest number of database “touches” (LIO buffer gets and PIO physical reads).

- **Remove unnecessary large-table full-table scans:** Unnecessary full-table scans cause a huge amount of unnecessary I/O and can drag-down an entire database. The tuning expert first evaluates the SQL based on the number of rows returned by the query. If the query returns less than 40 percent of the table rows, it needs tuning. The most common tuning remedy for unnecessary full-table scans is adding indexes. Standard b-tree indexes can be added to tables, and bitmapped and function-based indexes can also eliminate full-table scans. In some cases, an unnecessary full-table scan can be forced to use an index by adding an index hint to the SQL statement.
- **Cache small-table full-table scans :** In cases where a full-table scan is the fastest access method, the administrator should ensure that a dedicated data buffer is available for the rows. In Oracle7, you can issue “alter table xxx cache”. In Oracle8 and beyond, the small table can be cached by forcing it into the KEEP pool.
- **Verify optimal index usage :** This is especially important when using the rule-based optimizer. Oracle sometimes has a choice of indexes, and the tuning professional must examine each index and ensure that Oracle is using the proper index.

A strategic plan for Oracle SQL tuning : Many people ask where they should start when tuning Oracle SQL. Tuning Oracle SQL is like fishing. You must first fish in the Oracle library cache to extract SQL statements and rank the statements by their amount of activity.

Step 1: Identifying high-impact SQL: The SQL statements will be ranked according the number of executions and will be tuned in this order. The executions column of the v$sqlarea view and the *stats$sql_summary* or the *dba_hist_sql_summary* table can be used to locate the most frequently used SQL. Note that we can display SQL statements by:

- **Rows processed:** Queries that process a large number of rows will have high I/O and may also have impact on the TEMP tablespace.
- **Buffer gets:** High buffer gets may indicate a resource-intensive query.
- **Disk reads:** High disk reads indicate a query that is causing excessive I/O.
- **Memory KB:** The memory allocation of a SQL statement is useful for identifying statements that are doing in-memory table joins.
- **CPU secs:** This identifies the SQL statements that use the most processor resources.
- **Sorts:** Sorts can be a huge slowdown, especially if they're being done on a disk in the TEMP tablespace.
- **Executions:** The more frequently executed SQL statements should be tuned first, since they will have the greatest impact on overall performance.

Step 2: Determine the execution plan for SQL: As each SQL statement is identified, it will be "explained" to determine its existing execution plan. There are a host of third-party tools on the market that show the execution plan for SQL statements. The most common way of determining the execution plan for a SQL statement is to use Oracle's explain plan utility. By using explain plan, the Oracle DBA can ask Oracle to parse the statement and display the execution class path without actually executing the SQL statement.

This syntax is piped into the SQL optimizer, which will analyze the query and store the plan information in a row in the plan table identified by RUN1. Please note that the query will not execute; it will only create the internal access information in the plan table. The plan tables contains the following fields:

- **operation:** The type of access being performed. Usually table access, table merge, sort or index operation
- **options:** Modifiers to the operation, specifying a full table, a range table or a join
- **object_name:** The name of the table being used by the query component

- **Process ID:** The identifier for the query component
- **Parent_ID:** The parent of the query component. Note that several query components may have the same parent.

Step 3: Tune the SQL statement: For those SQL statements that possess a non-optimal execution plan, the SQL will be tuned by one of the following methods:

- Adding SQL "hints" to modify the execution plan
- Re-write SQL with Global Temporary Tables
- Rewriting the SQL in PL/SQL. For certain queries this can result in more than a 20x performance improvement. The SQL would be replaced with a call to a PL/SQL package that contained a stored procedure to perform the query.

(c) What is SQLJ? What are the requirements of SQLJ? Briefly describe the working of SQLJ. Can SQLJ use dynamic SQL? If yes, then how? Otherwise specify what type of SQL it can use. *7*

Ans. Refer to Chapter-4, Q.No.-11, Page No.-58

5. (a) What do you mean by Multiversioning? What are the various schemes available for multiversioning? Describe any one scheme in detail. *6*

Ans. Multi-version schemes maintain old versions of data item to increase concurrency.

These schemes are available for:

- Multi version Timestamp Ordering
- Multi-version Two-Phase Locking

In multi-version schemes each successfully write result in the creation of a new version of the data item written.

Timestamps can be used to label the versions. When a read (Q) operation is issued, you can select an appropriate version of Q based on the timestamp of the transaction, and return the value of the selected version. Read operation never has to wait as an appropriate version is returned immediately.

Multi-Version Timestamp Ordering

Each data item Q has a sequence of versions <Q1, Q2, ….,Qm>. Each version Qk contains three data fields:

Content – the value of version Qk.

W-timestamp (Qk) – timestamp of the transaction that created (wrote) the

version Qk
R–timestamp (Qk) – largest timestamp of a transaction that successfully read version Qk.

When a transaction T_i creates a new version Qk of Q, Qk's W-timestamp and R-timestamp are initialised to TS (T_i). R–timestamp of Qk is updated whenever a transaction T_j reads Qk, and TS (T_j) > R–timestamp (Qk).

(1) If transaction Ti issues a read(Q), then the value returned is the content of version Qk.
(2) If transaction Ti issues a write(Q), and if TS(Ti)< R-timestamp(Qk), then Transaction Ti is rolled back, otherwise, if TS(Ti)=W–timestamp(Qk), the content of Qk are overwritten a new version of Q is created.
The following multi–version technique ensure serialisability.

Suppose that transaction T_i issues a read (Q) or write (Q) operation. Let Qk denote the versions of Q whose write timestamp is the largest write timestamp less than or equal to TS (T_i).
(1) If transaction T_i issues a read (Q), then the value returned is the content of version Qk.
(2) If transaction T_i issues a write (Q), and if TS (T_i) < LR–timestamp(Qk), then transaction T_i is rolled back. Otherwise, if TS(T_i)= W–timestamp(Qk), the contents of Qk are overwritten, then a version of Q is created.
Read always succeeds. A write by T_i is rejected if some other transaction T_i (in the serialization order defined by the timestamp values) should read $T_{i's}$ write, has already read a version created by a transaction older than $T_{i.}$

Multi–Version Two-Phase Locking
It differentiates between read-only transactions and update transactions. Update transactions acquire read and write locks, and hold all locks up to the end of the transaction. That is, update transactions follow rigorous two-phase locking.
(1) Each successful write results in the creation of a new version of the data item written.
(2) Each version of a data item has a single timestamp whose value is obtained from a counter-transaction that is incremented during commit processing.

(b) What is Audit trail? Give four benefits provided by Audit trail to DBMS. *4*

Ans. Refer to Chapter-5, Q.No.-60, Page No.-103

(c) What do you mean by Multilevel Security? Discuss the techniques involved in support of multilevel security. ***5***
Ans. Refer to Chapter-5, Q.No.-61, Page No.-104

(d) What are Distributed databases? How is data distribution performed in DDBMS? Give two advantages and two disadvantages of DDBMS. ***5***
Ans. A distribute database system consists of a collection of sites, each of which may participate in the execution of transactions, which access data at one site, or several sites. The main difference between centralised and distributed database systems is that, in the former, the data resides in one single Centralised control, while in the latter, the data resides in several sites under to control of local distributed DBMS components which are under the control of one DBMS.

The primary advantage of distributed database systems is the ability to share and access data in a reliable and efficient manner.
(1) Data sharing and Distributed Control : The primary advantage of accomplishing data sharing by means of data distribution is that each site is able to retain a degree of control over data stored locally. Depending upon the design of the distributed database system, each local administrator may have a different degree of autonomy.
(2) Reflects organisational structure : Many organizations are distributed over several locations. If an organisation has many offices in different cities, databases used in such an application are distributed over these locations. Such an organisation may keep a database at each branch office containing details of the staff that work at that location, the local properties that are for rent, etc. The staff at a branch office will make local inquires to such data of the database. The company headquarters may wish to make global inquiries involving the access of data at all or a number of branches.

Disadvantages of Data Distribution
The primary disadvantage of distributed database systems is the added complexity required to ensure proper coordination among the sites. This increased complexity takes the form of:
- Higher Software development cost: Distributed database systems are complex to implement and, thus, more costly. Increased complexity implies that we

can expect the procurement and maintenance costs for DDBMS to be higher than those for a centralised DBMS. In additions to software, a distributed DBMS requires additional hardware to establish a network between sites.

- Greater potential for bugs: Since the sites of a distributed system operate concurrently, it is more difficult to ensure the correctness of algorithms. The art of constructing distributed algorithms is an active and important area of research.

A step-wise distributed database design methodology

Following is a step-wise methodology for distributed database design.

(1) Examine the nature of distribution. Find out whether an organisation needs to have a database at each branch office, or in each city, or possibly at a regional level. It has direct implication from the viewpoint of fragmentation. For example, in case database is needed at each branch office, the relations may fragment on the basis of branch number.

(2) Create a detailed global E-R diagram, if so needed and identify relations from entities.

(3) Analyse the most important transactions in the system and identify where horizontal or vertical fragmentation may be desirable and useful.

(4) Identify the relations that are not to be fragmented. Such relations will be replicated everywhere. From the global ER diagram, remove the relations that are not going to be fragmented.

(5) Examine the relations that are on one-site of a relationship and decide a suitable fragmentation schema for these relations. Relations on the many-side of a relationship may be the candidates for derived fragmentation.

(6) During the previous step, check for situations where either vertical or mixed fragmentation would be needed, that is, where the transactions require access to a subset of the attributes of a relation.

MCS-043 : ADVANCED DATABASE DESIGN
December, 2007

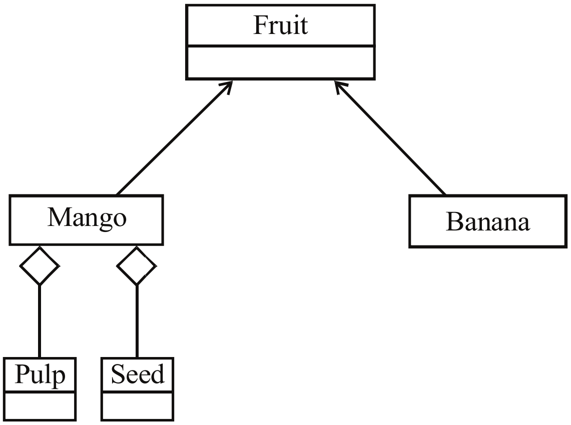
Note: Question number 1 is **compulsory.** Answer any **three** questions from the rest.

1. (a) Why is the functional dependency called so? Consider the following functional dependency:

if you study → you will pass

Create instances where this functional dependency will hold/not hold. ***12***

(b) Create and explain an object oriented database for the following UML diagram. Assume your attributes and functions. ***8***



(c) Consider the following table: ***10***

Employee_name	Project_name	Dependent_name
Mohan	X	Shyam
Mohan	Y	Ram
Mohan	X	Ram
Mohan	Y	Shyam

Identify the multivalued dependencies in the above table and write an SQL code to check whether the table satisfies the multivalued dependency identified by you.

(d) Explain the Apriori algorithm for finding frequent itemsets using an example. ***10***

2. (a) What are assertions? What is the syntax for declaration of an assertion? Also, give an example of assertion. ***10***

(b) Explain any two examples of vendor-specific security. ***10***

3. (a) Given the relational schemes:
ENROLL (S#, C#, Section) S# represents student number.
TEACH (Prof, C#, Section) C# represents course number
ADVISE (Prof, S#) Prof is a thesis advisor of S#
GRADES (S#, C#, Grade, Year)
STUDENT (S#, Sname) Sname is a student name
Write queries expressed in relational algebra.
(i) List all students taking courses with Mohan or Shyam.
(ii) List all students taking at least one course that their advisor teaches.
(iii) List those professors who teach more than one section of the same course. ***12***

(b) Explain the architecture of a Data warehouse with the help of a figure. ***8***

4. (a) Explain the tasks in the KDD process with the help of a figure. ***10***

(b) Explain the architecture of Oracle 10g with the help of a figure. ***10***

5. Consider the universal relation R = {A, B, C, D, E, F, G, H, I, J} and a set of functional dependencies.

$$F = \begin{Bmatrix} AB \rightarrow C \\ A \rightarrow DE \\ F \rightarrow GH \\ D \rightarrow IJ \end{Bmatrix}$$

Decompose R into BCNF. ***10***

(b) Explain the component architecture of DDBMS with the help of a figure. ***10***

MCS – 043: ADVANCED DATABASE DESIGNS
December, 2007

***Note:** Question number 1 is **compulsory**. Attempt any **three** questions from the rest.*

1. (a) Why is the functional dependency called so? Consider the following functional dependency:

if you study you will pass

Create instances where this functional dependency will hold/not hold.

Ans. When a single constraint is established between two sets of attributes from the database it is called *functional dependency*. A functional dependency denoted by $x \to y$ between two sets of attributes X and Y that are subset of universal relation "A" specifies a constraint on the possible tuples that can form a relation state of "A". The constraint is that, for any two tuples t1 and t2 in "A" that have t1(X) = t2(X), we must also have t1(Y) = t2(Y). It means that, if tuple t1 and t2 have same values for attributes X than $X \to Y$ to hold t1 and t2 must have same values for attributes Y.

The relation schema "A" determines the function dependency of Y on X $(X \to Y)$ when and only when:

(1) if two tuples in "A", agree on their X value then

(2) they **must** agree on their Y value.

This semantic property of functional dependency explains how the attributes in "A" are related to one another. A FD in "A" must be used to specify constraints on its attributes that must hold at all times.

Instance where FD hold

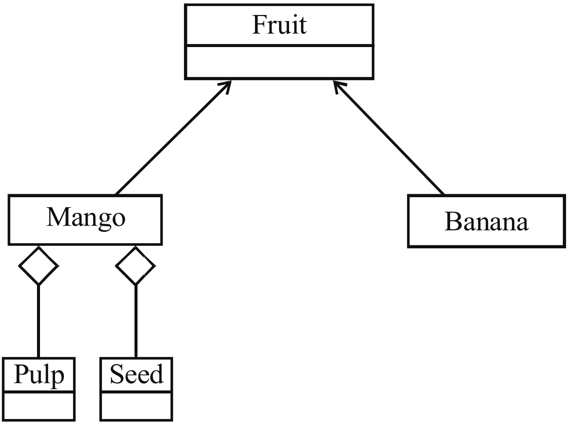
Name	Study	Pass
A	Yes	Yes
B	Yes	Yes
C	No	No
D	No	Yes

Instance where FD does not hold

Name	Study	Pass
A	Yes	Yes
B	Yes	No
C	No	No
D	No	Yes

← tuple violates FD is B

(b) Create and explain an object oriented database for the following UML diagram. Assume your attributes and functions.



Ans.

```
class fruit
(key name)
{ attribute string name;
attribute string color;
attribute string category;
};
class Mango extends fruit
{ attribute string variety;
attribute string shape;
relationship set <pulp> contain pulp :: pulp;
relationship set <seed> contain seed :: seed;
void getpulpqty ( );
void getseedsize ( );
};
class Banana extends fruit
{ attribute int size;
attribute string variety;
attribute string taste;
};
class pulp
{ attribute string color;
attribute string taste;
relationship Mango has :: has_pulp;
};
class seed
{ attribute string size;
```

relationship Mango has :: has_seed;
};

(c) Consider the following table:

Employee_name	Project_name	Dependent_name
Mohan	X	Shyam
Mohan	Y	Ram
Mohan	X	Ram
Mohan	Y	Shyam

Identify the multivalued dependencies in the above table and write an SQL code to check whether the table satisfies the multivalued dependency identified by you.

Ans. Multivalue Dependencies hold by the relation

1. Employee_Name $\twoheadrightarrow$ Project_Name
2. Employee_Name $\twoheadrightarrow$ Dependent_Name
3. Project_Name $\twoheadrightarrow$ Dependent_Name

SQL Statement for:

First MVD: Select Employee_Name, Count (Distinct Project_Name) From Emp Group By Employee_Name, Project_Name

Second MVD: Select Employee_Name, Count (Distinct Dependent_Name) From Emp Group By Employee_Name, Dependent_Name

Third MVD: Select Project_Name, Count (Distinct Dependent_Name) From Emp Group By Project_Name, Dependent_Name

(d) Explain the Apriori algorithm for finding frequent itemsets using an example.

Refer to Q.No.-60,61 Page No.-138-139

2. (a) What are assertions? What is the syntax for declaration of an assertion? Also, give an example of assertion.

Refer to Q.No.-1 Page No.-48-49

(b) Explain any two examples of vendor-specific security.

Ans. Individual vendors largely determine the security schemes that may be implemented to provide the link between the database and its interfaces.

Oracle: Oracle, provides SSL and S-HTTP security. Oracle uses Java as a basic component of its security model. The company created its Oracle Web Server to work most effectively with Oracle clients such as the solutions created with the Developer/2000 or other development tools.

Oracle modified the HTTP protocol to allow a straight/direct connection between the client and the server. This connection defines a session in which the user is identified by a generated ID.

These enhancements are also present in the Secure Network Server (SNS) that is included in the Oracle Universal Database a single login permits access to any Oracle database in an enterprise system.

The Java security classes are used by the oracle development tools to give complete security integration to the client.

Microsoft: Microsoft has included most of the key security technologies with Internet Information Server (IIS). For user authentication, Microsoft provides its tried-and-true challenge/response mechanism. Traditional login on the Web presents the same security as in the basic Windows NT login. Unfortunately, only Microsoft Internet Explorer browser supports this login approach.

For database access, IIS security has been integrated with Microsoft SQL Server through the Internet Database Connector. Although, users must login through an HTML login form, the information may be verified by a SQL Server stored procedure. Microsoft has also integrated the Kerberos security architecture into Windows NT Server. By releasing the server, Microsoft hopes to integrate the Kerberos native to the NT Server with public key security. Microsoft has already released a Certificate Server API in an attempt to create a Certificate Server standard.

3. (a) Given the relational schemes:
ENROLL (S#, C#, Section) S# represents student number.
TEACH (Prof, C#, Section) C# represents course number
ADVISE (Prof, S#) Prof is a thesis advisor of S#
GRADES (S#, C#, Grade, Year)
STUDENT (S#, Sname) Sname is a student name
Write queries expressed in relational algebra.
(i) List all students taking courses with Mohan or Shyam.

Ans.

(i)

$\pi_{\text{S\#}}(\sigma_{(\text{A.Sname} = \text{'Mohan'} \vee \text{A.SName} = \text{'Shyam'}) \wedge \text{Enroll.C\#=A.C\#}}$ (Enroll $\bowtie$ Student $\bowtie$ 9_{A}(Enroll)))

(ii) List all students taking at least one course that their advisor teaches.

$\pi_{\text{S\#}}(\sigma_{\text{ENROLL.S\#=ADVISE.S\#}}$ (ADVISE $\bowtie$ TEACH $\bowtie$ ENROLL))

(iii) List those professors who teach more than one section of the same course.

$$\pi_{\text{prof}}\left(\sigma_{\text{count}(\text{section})>1}\left(\text{g}_{\text{prof-c\#}}\left(\text{teachers}\right)\right)\right)$$

(b) Explain the architecture of a Data warehouse with the help of a figure.

Refer to Q.No.-39 Page No.-125-129

4. (a) Explain the tasks in the KDD process with the help of a figure.

Ans. The different tasks in KDD are as follows:

• **Obtains information on application domain:** It gathers knowledge from the domain relevant to the user.

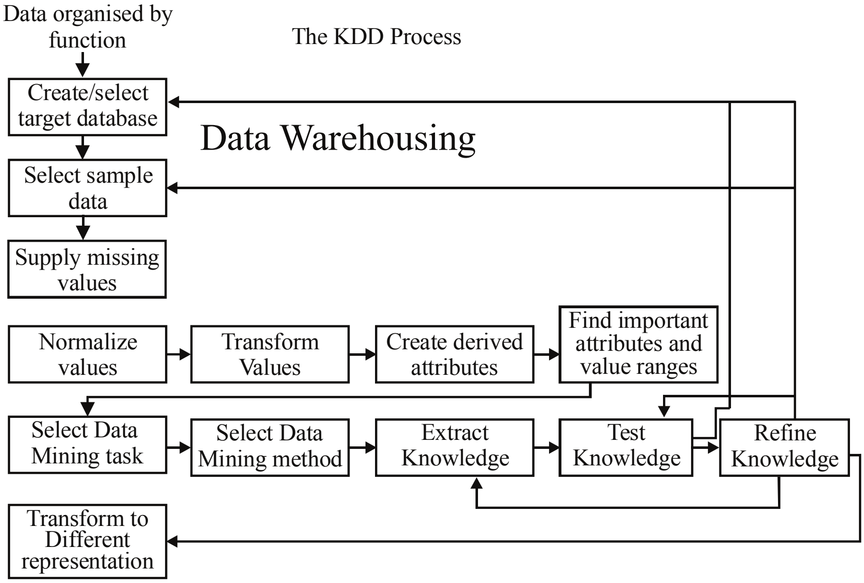
• **Extracting data set:** It includes extracting required data which will later, be used for analysis.

• **Data cleansing process:** It involves basic operations such as, the removal of noise, collecting necessary information to from noisy data, such as, deciding on strategies for handling missing data fields.

• **Data reduction and projection:** Using dimensionality reduction or transformation methods it reduces the effective number of dimensions under consideration.

• **Selecting data mining task:** In this stage we decide what the objective of the KDD process is. Whether it is classification, clustering, association rules etc.

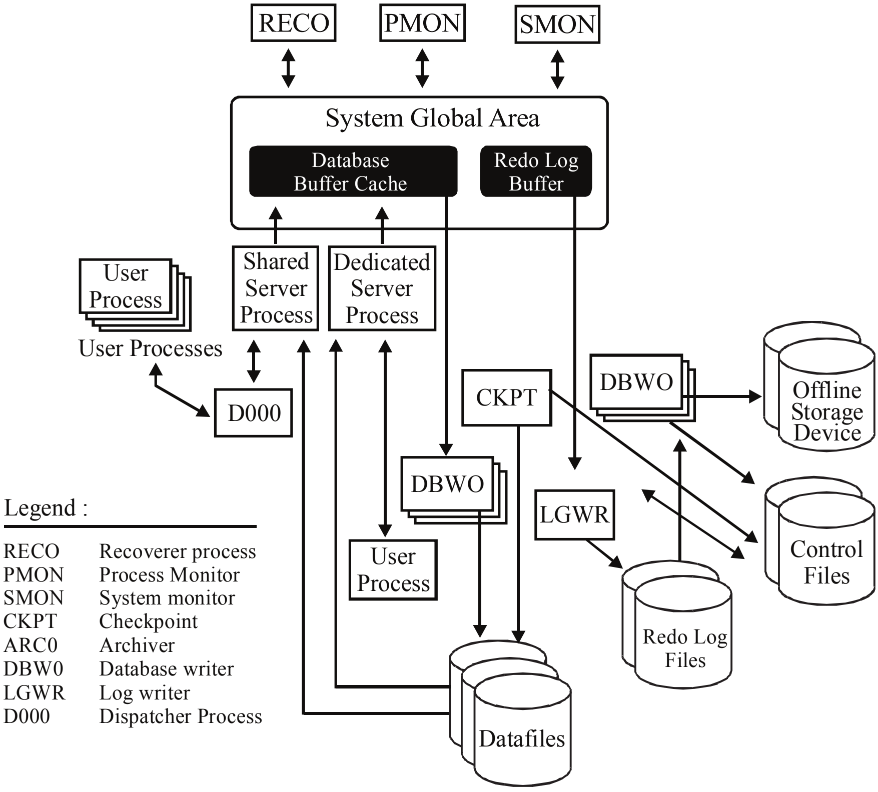
• **Selecting data mining method:** In this stage, we decide the methods and the parameter to be used for searching for desired patterns in the data.


Data organised by function
The KDD Process
Create/select target database
Data Warehousing
Select sample data
Supply missing values
Normalize values
Transform Values
Create derived attributes
Find important attributes and value ranges
Select Data Mining task
Select Data Mining method
Extract Knowledge
Test Knowledge
Refine Knowledge
Transform to Different representation

Tasks in the KDD process

· **Extraction of patterns:** It includes searching for desired patterns only, because, the data-mining model may generate a lot of patterns.
· Interpretation and presentation of pattern/model.

(b) Explain the architecture of Oracle 10g with the help of a figure.
Ans. A schematic diagram of Oracle database is given below:



Oracle 10g Architecture

1. Physical Database Structures: The physical database structures of an Oracle database, include datafiles, redo log files and control files.

Datafiles: Every Oracle database has one or more physical datafiles. The datafiles contain all the database data. The data of logical database structures, such as tables and indexes, is physically stored in the datafiles allocated for a database.

Control Files: Every Oracle database has a control file. A control file contains entries that specify the physical structure of the database.

Redo Log Files: Every Oracle database has a set of two or more redo log files. The set of redo log files is collectively known as the redo log for the database. A redo log is made up of redo entries (also called redo records).

Parameter Files: Parameters files contain a list of configuration parameters for that instance and database.

Backup Files: To restore a file is to replace it with a backup file. Typically, you restore a file when a media failure or user error has damaged or deleted the original file.

2. Logical Database Structures: The logical storage structures, including data blocks, extents and segments, enable Oracles fine-grained control of disk space use.

Tablespaces: A database is divided into logical storage units called tablespaces, which group related logical structures together. For example, tablespaces commonly group together all application objects to simplify administrative operations.

Each database is logically divided into one or more tablespaces. One or more datafiles are explicitly created for each tablespace to physically store the data of all logical structures in a tablespace.

Oracle Data Blocks: At the finest level of granularity, Oracle database data is stored in data blocks. One data block corresponds to a specific number of bytes of physical database space on disk. The standard block size is specified by the DB_BLOCK_SIZE initialization parameter.

Extents: The next level of logical database space is an extent. An extent is a specific number of contiguous data blocks, obtained in a single allocation, used to store a specific type of information.

Segments: Next, is the segment or the level of logical database storage. A segment is a set of extents allocated for a certain logical structure.

3. Oracle Data Dictionary: Each Oracle database has a data dictionary. An Oracle data dictionary is a set of tables and views that are used as a read-only reference on the database. For example, a data dictionary stores information about the logical and physical structure of the database.

4. Oracle Instance: An Oracle database server consists of an Oracle database and an Oracle instance. Every time a database is started, a system global area (SGA) is allocated and Oracle background processes are started. The combination of the background processes and memory buffers is known as an Oracle instance.

Instance Memory Structures: Oracle creates and uses memory structures to complete several jobs. For example, memory stores program code being run and data shared among users. Two basic memory structures associated with Oracle are: the system global area and the program global area. The following subsections explain each in detail.

System Global Area: The System Global Area (SGA) is a shared memory region that contains data and control information for one Oracle instance.

Oracle allocates the SGA when an instance starts and deallocates it when the instance shuts down. Each instance has its own SGA.

Database Buffer Cache of the SGA: Database buffers store the most recently used blocks of data. The set of database buffers in an instance is the database buffer cache.

Redo Log Buffer of the SGA: The redo log buffer stores redo entries – a log of changes made to the database. The redo entries stored in the redo log buffers are written to an online redo log, which is used if database recovery is necessary.

Shared Pool of the SGA: The shared pool contains shared memory constructs, such as shared SQL areas. A shared SQL area is required to process every unique SQL statement submitted to a database. A shared SQL area contains information such as the parse tree and execution plan for the corresponding statement.

Program Global Area: The Program Global Area (PGA) is a memory buffer that contains data and control information for a server process. A PGA is created by Oracle when a server process is started. The information in a PGA depends on the configuration of Oracle.

5. Oracle Background Processes: An Oracle database uses memory structures and processes to manage and access the database.

There are numerous background processes and each Oracle instance can use several background processes.

Process Architecture: A process is a "thread of control" or a mechanism in an operating system that can run a series of steps. Some operating systems use the terms job or task. A process generally has its own private memory area in which it runs. An Oracle database server has two general types of processes: user processes and Oracle processes.

Oracle Processes: Oracle processes are invoked by other processes to perform functions on behalf of the invoking process. Oracle creates server processes to handle requests from connected user processes. A server process communicates with the user process and interacts with Oracle to carry out requests from the associated user process.

5. (a) Consider the universal relation
R = {A, B, C, D, E, F, G, H, I, J} and a set of functional dependencies.

$$\mathbf{F} = \begin{Bmatrix} \mathbf{AB \rightarrow C} \\ \mathbf{A \rightarrow DE} \\ \mathbf{F \rightarrow GH} \\ \mathbf{D \rightarrow IJ} \end{Bmatrix}$$

Decompose R into BCNF.

Ans. A relation is in BCNF if each determinants is a candidate key. Therefore, the relations R, can be decomposed in following relations (based on each non-trivial FD).

R_1 (A B C)	$F_1 = \{AB \rightarrow C\}$
R_2 (A D E)	$F_2 = \{A \rightarrow DE\}$
R_3 (F G H)	$F_3 = \{F \rightarrow GH\}$
R_4 (D I J)	$F_4 = \{D \rightarrow IJ\}$
R_5 (A F)	$F_5 = \{AF \rightarrow \phi\}$ to preserve the key of the relation R

(b) Explain the component architecture of DDBMS with the help of a figure.

Refer to Q.No.-2(i) Page No.-168-170

MCS – 043: ADVANCED DATABASE DESIGN
June, 2008

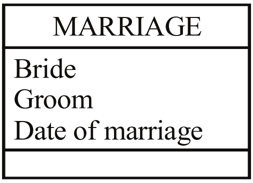
Note: *Question number 1 is* ***compulsory****. Answer any* ***three*** *questions from the rest.*

Q1. (a) What are cursors, stored procedures and triggers? Explain with an example of your choice.

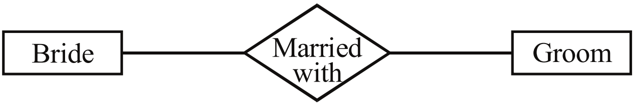
Refer to Page-92-94, Q.No.-36

(b) The concept of "marriage" is identified as a class in first view, as a relationship in second view and as an attribute in third view. How will you solve this using View-Integration problem? Suggest one good measure.

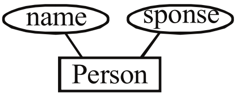
Ans. View 1:



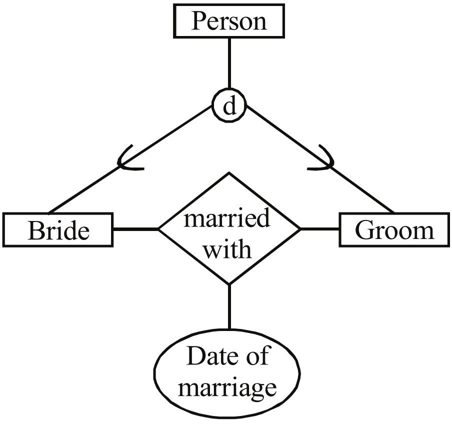
View 2:



View 3:



All the above three views can be integrated into one conceptual schema as given below:



(c) Differentiate between Logical and Physical database design. How can UML Diagrams help in database design?

Ans.

Basis	**Logical Database Design**	**Physical Database Design**
Task	Maps or transforms the conceptual schema (or an ER schema) from the high-level data model into a relational database schema.	The specifications for the stored database in terms of physical storage structures, record placement, and indexes are designed.
Choice of criteria	The mapping can proceed in two stages: ▪ System-independent mapping but data model-dependent ▪ Tailoring the schemas to a specific DBMS	The following criteria are often used to guide the choice of physical database design options: ▪ Response Time ▪ Space Utilization ▪ Transaction Throughput
Result	DDL statements in the language of the chosen DBMS that specify the conceptual and external level schemas of the database system. But if the DDL statements include some physical design parameters, a complete DDL specification must wait until after the physical database design phase is completed.	An initial determination of storage structures and the access paths for the database files. This corresponds to defining the internal schema in terms of Data Storage Definition Language.

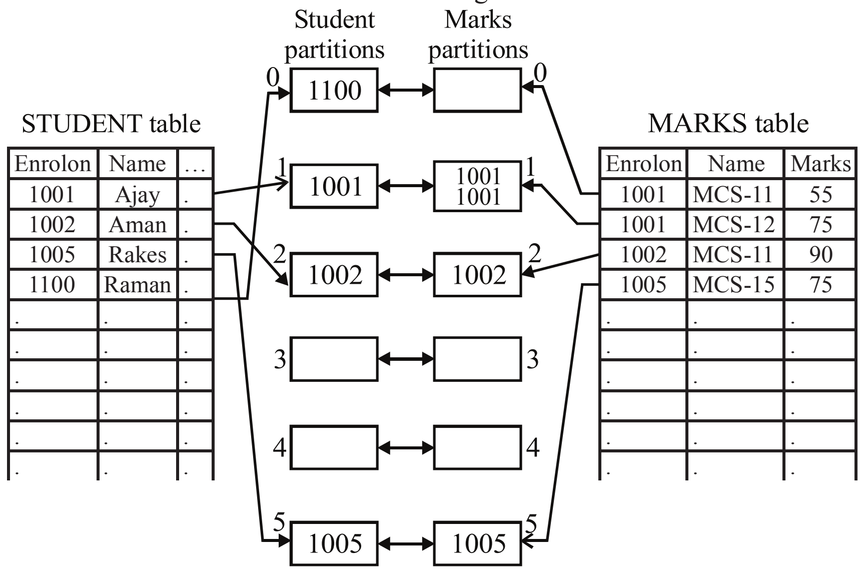
The database design is divided into several phases. The logical database design and physical database design are two of them. This separation is generally based on the concept of three-level architecture of DBMS, which provides the data independence. Therefore, we can say that this separation leads to data independence because the output of the logical database design is the conceptual and external level schemas of the database system which is independent from the output of the physical database design that is internal schema.

Now Refer : Page-183-184, Q.No.-1(c)

(d) Explain how Hash Join is applicable to Equi Join and Natural Joins. Explain the Algorithm and Cost calculation for Simple Hash Join.

Ans. This is applicable to both the equi-joins and natural joins. A hash function *h* is used to partition tuples of both relations, where *h* maps joining attribute (enroll no in our example) values to {0, 1, …, *n-1*}.

The join attribute is hashed to the join-hash partitions. In the example of *Figure* we have used mod 100 function to hashing and n = 100.



STUDENT table

Enrolon	Name	...
1001	Ajay	.
1002	Aman	.
1005	Rakes	.
1100	Raman	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.

MARKS table

Enrolon	Name	Marks
1001	MCS-11	55
1001	MCS-12	75
1002	MCS-11	90
1005	MCS-15	75
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.

Figure : A hash-join example

Once the partition tables of STUDENT and MARKS are made on the enrolment number, then only the corresponding partitions will participate in the join as: A STUDENT tuple and a MARKS tuple that satisfy the join condition will have the same value for the join attributes. Therefore, they will be hashed to equivalent partition and thus can be joined easily.

Algorithm for Hash-Join: The hash-join of two relations *r* and *s* is computed as follows:

· Partition the relation *r* and *s* using hashing function *h*.

· For each partition *si* of *s*, load the partition into memory and build an in-memory hash index on the join attribute.

· Read the tuples in *ri* from the disk, one block at a time. For each tuple in *ri* locate each matching tuple in *si* using the in-memory hash index and output the concatenation of their attributes.

In this method, the relation *s* is called the *build* relation and *r* is called the *probe* relation. The value *n* and the hash function *h* is chosen in such a manner that each *si* should fit in to the memory. Typically *n* is chosen as Number of blocks of s/Number of memory buffers] *f(M) where f is a "fudge factor", typically around 1.2. The probe relation partitions *ri* need not fit in memory.

Average size of a partition *si* will be less than *M* blocks using the formula for *n* as above thereby allowing room for the index. If the build relation *s* is very huge, then the value of *n* as given by the above formula may be greater than *M*

– *1* i.e., the number of buckets is > the number of buffer pages. In such a case, the relation *s* can be recursively partitioned, instead of partitioning *n* ways, use *M – 1* partitions for *s* and further partition the *M – 1* partitions using a different hash function.

Cost calculation for Simple Hash-Join:

(i) Cost of partitioning *r* and *s*: all the blocks of *r* and *s* are read once and after partitioning written back, so cost 1 = 2 (blocks of *r* + blocks of *s*).

(ii) Cost of performing the hash-join using build and probe will require at least one block transfer for reading the partitions

Cost 2 = (blocks of *r* + blocks of *s*)

(iii) There are a few more blocks in the main memory that may be used for evaluation, they may be read or written back. We ignore this cost as it will be too less in comparison to cost 1 and cost 2.

Thus, the total cost = cost 1 + cost 2

= 3 (blocks of *r* + blocks of *s*)

Cost of Hash-Join requiring recursive partitioning:

(i) The cost of partitioning in this case will increase to number of recursion required, it may be calculated as:

Number of iterations required = ([$\log_{M-1}$ (blocks of s)] – 1)

Thus, cost 1 will be modified as:

= 2 (blocks of *r* + blocks of *s*) × ([$\log_{M-1}$ (blocks of s)] – 1)

The cost for step (ii) and (iii) here will be the same as that given in steps (ii) and (iii) above.

Thus, total cost = 2(blocks of *r* + blocks of *s*) ([$\log_{M-1}$(blocks of *s*) – 1]) + (blocks of *r* + blocks of *s*).

Because *s* is in the inner term in this expression, it is advisable to choose the smaller relation as the build relation. If the entire build input can be kept in the main memory, *n* can be set to 1 and the algorithm need not partition the relations but may still build an in-memory index, in such cases the cost estimate goes down to (Number of blocks *r* + Number of blocks of *s*).

(e) How is a database management system different from a data warehouse? When we have sufficient tools and concepts to develop a DBMS, then why do we still design warehouses?

Ans. The first and most important difference between a classical, general purpose DBMS and a data warehouse–specific DBMS is how updates are performed. A classical, general purpose DBMS must be able to accommodate record-level, transaction-based updates as a normal part of operations. Because these updates are a regular feature of the general purpose DBMS, this DBMS must offer facilities for such items as:

Locking
COMMITs
Checkpoints
Log processing
Deadlock
Backout

Not only do these features become a normal part of the DBMS, they consume a tremendous amount of overhead. Interestingly, the overhead is consumed even when it isn't being used. In other words, at least some update and locking overhead that's dependent on the DBMS is required by a general purpose DBMS even when read-only processing is being executed. Depending on the general purpose DBMS, the overhead required by an update can be minimized, but it cannot be completely eliminated. For a data warehouse–specific DBMS, there's no need for any of the overhead of an update.

The second major difference between a general purpose DBMS and a data warehouse–specific DBMS regards basic data management. For a general purpose DBMS, data management at the block level includes space that's reserved for future block expansion at the moment of update or insertion. Typically, this space is referred to as freespace. For a general-purpose DBMS, freespace may be as high as 50%. For a data warehouse–specific DBMS, freespace always equals 0% because there's no need for expansion in the physical block, once loaded; after all, an update is not done in the data warehouse environment. Indeed, given the amount of data to be managed in a data warehouse, it makes no sense to reserve vast amounts of space that may never be used. Another relevant difference between the data warehouse and the general purpose environment that's reflected in the different types of DBMS is indexing.

A general purpose DBMS environment is restricted to a finite number of indexes. This restriction exists because as updates and insertions occur, the indexes require their own space and their own data management. In a data warehouse environment where there is no update and there is a need to optimize data access, there's a need (and an opportunity) for many indexes. Indeed, a much more robust and sophisticated indexing structure can be employed for data warehousing than for operational, update-oriented databases.

Beyond indexing, update and basic data management at the physical block level are some other basic differences between the data management capabilities and philosophies of a general purpose, transaction-processing DBMS and a data warehouse–specific DBMS. Perhaps the most basic difference is the ability to physically organize data in an optimal fashion for different kinds of access. Typically a general purpose DBMS physically organizes data for optimal transaction access and manipulation. Organizing in this fashion allows many different types of data to be gathered according to a common key and efficiently accessed in one or two I/Os. Data that's optimal for informational access usually has a very different physical profile; it's organized so that many different occurrences of the same type of data can be accessed efficiently in one or two physical I/Os. Another important difference between the classical transaction-processing database environment and the data warehouse environment is that the latter tends to hold much more data, measured in terabytes (10E15) and petabytes (10E18), than the former. I agree if you say you don't have this amount of data now but, as we'll see later, if you have a warehouse and it's running fine and your users like it, you might go one step further and integrate much more data within your company.

(f) How does granularity of data item affect the performance of concurrency control? How are granularity and database security related?
Ans. In the concurrency control schemes, we used each individual data item as a unit on which synchronization is performed. Can we allow data items to be of various sizes and define a hierarchy of data granularities, whereby the small granularities are nested within larger ones? It can be represented graphically as a tree (not tree-locking protocol). In such a situation, when a transaction locks a node in the tree explicitly, it implicitly locks all the node's descendants in the same mode. Granularity of locking (the levels in trees where locking is done) are:

· **Fine granularity (lower in the tree):** It allows high concurrency but has high locking overhead,

· **Coarse granularity (higher in the tree):** It has low locking overhead but also has low concurrency.

Figure shows an example of Granularity Hierarchy.

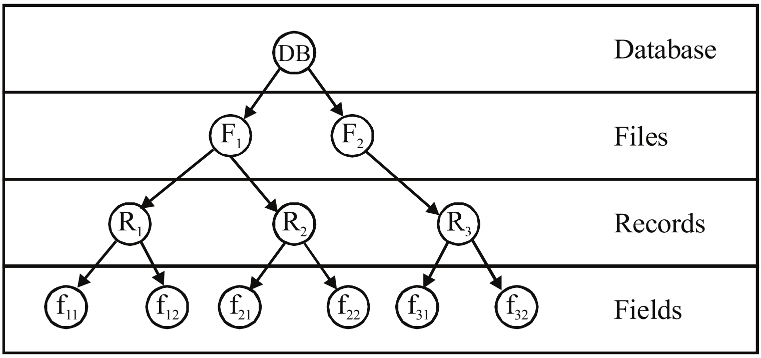


Figure : Hierarchical locking

The highest level in the example hierarchy is the entire database. The levels below are of file, record and field.

(g) What are the various problems that arise in distributed DBMS environment that are not encountered in a centralized DBMS environment?

Refer to Page-110, Q.No.-68

Q2. (a) Distinguish between the following:

(i) Embedded SQL and Dynamic SQL

Refer to Page-55-56, Q.No.-8

(ii) XML and HTML

Refer to Page-166-167, Q.No.-1(vii)

(iii) 2 PC and 3 PC

Ans. Major difference of 3PC occurs with 2PC, where a crashed participant could recover to a Commit state while all the others were still in state Ready. In that case, the remaining operational processes could not reach a final decision and would have to wait until the crashed process recovered. With 3PC, if any operational process is in its Ready state, no crashed process will recover to a state other than INIT, Abort, or PreCommit. For this reason, surviving processes can always come to a final decision. Finally, if the processes that P can reach are in state Precommit (and they form a majority), then it is safe to commit the transaction. Again, it can be shown that in this case, all other processes will either be in state Ready or at least, will recover to state Ready, Precommit, or Commit when they had crashed.

(iv) Data warehousing and Data mining

Ans. There is a lot of confusion concerning the terms data mining and data warehousing (also referred to as business intelligence in the marketplace today). To my chagrin, many IT professionals use the two terms interchangeably, with little hesitation or regard for the differences between the two types of applications. While the goals of both are related, and often overlap; data mining and data warehousing are dedicated to furnishing different types of analytics, for different types of users and therefore merit their own space.

By definition, data mining is intended for users who are statistically inclined. These analysts look for patterns hidden in data, which they are able to extract using statistical models. Data miners engage in question formulation based primarily on the "law of large numbers" to identify potentially useful relationships between data elements, which can be profitable to companies.

For instance, car insurance companies will sift through terabytes of data to link accident rates to demographic groups. They may start with a hypothesis that single men in the 18-25 age group who drive red sports cars are prone to drive recklessly (based on number of tickets they get and number of accidents they have) than older men who drive minivans. After sifting through their data, they may find that this hypothesis is not necessarily true. On the contrary, they may find that families who have multiple cars, among which one is a sports car (color is irrelevant), have more accidents and tickets when they have a teenager living in the house. Such cause-and-effect patterns help data miners quote premiums fairly on insurance rates. And hopefully allow them to reduce premiums where appropriate.

(b) List all the functional dependencies satisfied by the following relation:

A	B	C
a_1	b_1	c_1
a_1	b_1	c_2
a_2	b_1	c_1
a_2	b_1	c_3

Write an SQL-Code to identify whether a given functional dependency holds.

Ans. FDS:

$A \rightarrow B$

$C \rightarrow B$

SQL Code for

First FD: $A \rightarrow B$

Select A, Count (A) From R Group By A

Second FD: $C \rightarrow B$

Select C, Count (B) From R Group By C

(c) What are Semantic Databases? List features of Semantic Databases. Explain the process of searching the knowledge in these databases.

Ans. Semantic modeling provides a far too rich set of data structuring capabilities for database applications. A semantic model contains far too many constructs that may be able to represent structurally complex inter-relations among data in a somewhat more natural way.

Semantic modeling is one of the tools for representing knowledge especially in Artificial Intelligence and object-oriented applications. Thus, it may be a good idea to model some of the knowledge databases using semantic database system.

Some of the features of semantic modeling and semantic databases are:

· these models represent information using high-level modeling abstractions,

· these models reduce the semantic overloading of data type constructors,

· semantic models represent objects explicitly along with their attributes,

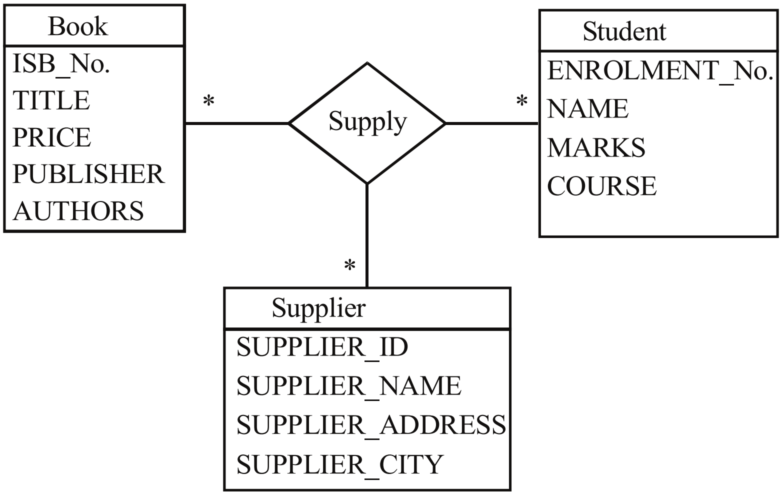
· semantic models are very strong in representing relationships among objects, and

· they can also be modeled to represent IS A relationships derived schema and also complex objects. Some of the applications that may be supported by such database systems in addition to knowledge databases may be applications such as bio-informatics, that require support for complex relationships, rich constraints and large-scale data handling.

Q3. (a) What is Data Mining? How is Data Mining a part of Knowledge Discovery process? What are the goals of Data Mining and Knowledge Discovery?

Ans. Data mining is the process of automatic extraction of interesting (non trivial, implicit, previously unknown and potentially useful) information or patterns from the data in large databases. Data mining is only one of the many steps involved in knowledge discovery in databases. The various steps in KDD are data extraction, data cleaning and preprocessing, data transformation and reduction, data mining and knowledge interpretation and representation. The different data-mining goals are: Classification, Clustering and Association Rule Mining.

(b) Create an object-oriented database using ODL for the following figure:



Ans. class Book
(key ISB_No.)
{ attribute string ISB_No;
attribute string title;
attribute float price;
attribute string publisher;
attribute string authors;
relationship set <student> student :: student;
relationship set <supplier> supplier :: supplier;

```
};
class student
(key Enrolment_No)
{ attribute number Enrolment_No;
attribute string name;
attribute number marks;
attribute string course;
relationship set <book> book :: book;
relationship set <supplier> supplier :: supplier;
};
class supplier
(key supplier_ID)
{ attribute number supplier_ID;
attribute string supplier_name;
attribute string supplier_address;
attribute string supplier_city;
relationship set <book> book :: book;
relationship set <student> student :: student;
};
```

(c) What is ODBC? How is ODBC implemented? How is access provided to Database on World Wide Web (WWW)?
Refer to Page-179, Q.No.-5(iii)

Q4. (a) What do you understand by Open Source Technologies? List some open source DBMS. Explain the working and features of PostgreSQL.

Ans. Open Source Definition

The Open Source Definition is used by the Open Source Initiative to determine whether or not a software license can be considered open source. The definition was based on the Debian Free Software Guidelines, written and adapted primarily by Bruce Perens.

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1) Free Redistribution : The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2) Source Code : The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3) Derived Works : The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4) Integrity of The Author's Source Code : The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5) No Discrimination Against Persons or Groups : The license must not discriminate against any person or group of persons.

6) No Discrimination Against Fields of Endeavor : The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7) Distribution of License : The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8) License Must Not Be Specific to a Product : The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9) License Must Not Restrict Other Software : The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

10) License Must Be Technology-Neutral : No provision of the license may be predicated on any individual technology or style of interface.

Some open source DBMSs are:

1. PostgreSQL

2. MySQL

PostgreSQL supports the following features:

- ANSI SQL 2 compliance,
- Support for transactions, triggers, referential integrity and constraints,
- High level language support,
- Inheritance, complex data types and polymorphism,
- Built in complex data types like IP address,
- Aggregate functions, collections and sequences,
- Portability, and
- ODBC and JDBC drivers.

(b) Explain the overview of ORACLE architecture. What are the main ORACLE tools available for application development? Can mobile use of data centric applications be performed by ORACLE?

Ans. Refer to Dec-2007, Q.No.-4(b)

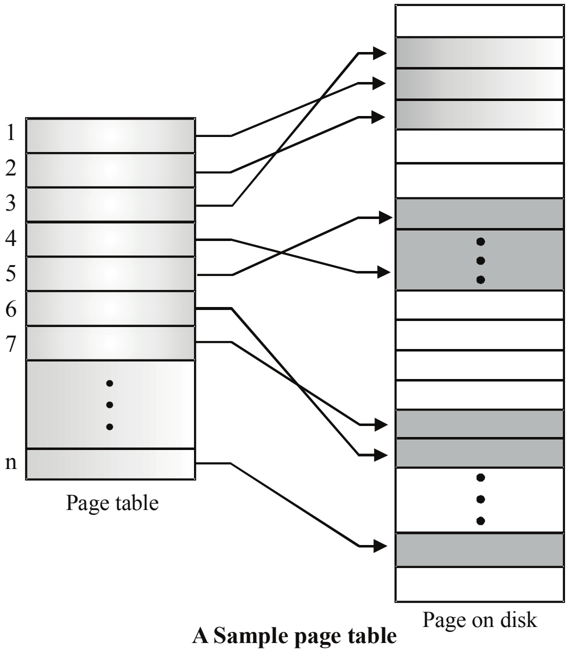
ORACLE forms Developer
ORACLE Reports Developer
ORACLE Designer
ORACLE J Developer
ORACLE Discoverer Administrative Edition
ORACLE Portal.

(c) What is data dictionary? List some features of data dictionary. What are the various approaches to implement a Distributed Database Catalogue?

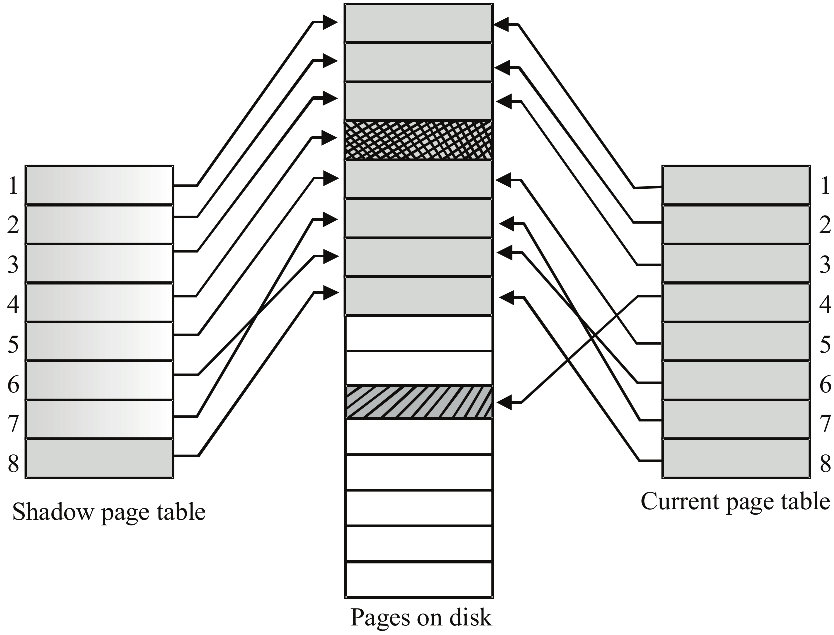
Refer to Page-64-67, Q.No.-18, 20 and 24

Q5. (a) What is shadow paging? Illustrate with an example. What are the advantages and disadvantages of shadow paging?

Ans. Shadow paging is an alternative to log-based recovery; this scheme is useful if transactions are executed serially. In this *two* page tables are maintained during the lifetime of a transaction—the **current page table** and the **shadow page table**. It stores the shadow page table in non-volatile storage, in such a way that the state of the database prior to transaction execution may be recovered (shadzow pages table is never modified during execution). To start with, both the page tables are identical. Only the current page table is used for data item accesses during execution of the transaction. Whenever any page is about to be written for the first time a copy of this page is made on an unused page, the current page table is then made to point to the copy and the update is performed on the copy.



A Sample page table



Shadow page table

To commit a transaction:

1. Flush all modified pages in main memory to disk

2. Output current page table to disk

3. Make the current page table the new shadow page table, as follows:

- keep a pointer to the shadow page table at a fixed (known) location on disk,
- to make the current page table the new shadow page table, simply update the pointer to point at the current page table on disk.

Once pointer to shadow page table has been written, transaction is committed. No recovery is needed after a crash — new transactions can start right away, using the shadow page table. Pages not printed to from current/shadow page table should be freed (garbage collected).

Advantages of shadow paging over log-based schemes:

- It has no overhead of writing log records,
- The recovery is trivial.

Disadvantages:

- Copying the entire page table is very expensive, it can be reduced by using a page table structured like a B$^+$-tree (no need to copy entire tree, only need to copy paths in the tree that lead to updated leaf nodes).
- Commit overhead is high even with the above extension (Need to flush every updated page and page table).
- Data gets fragmented (related pages get separated on disk).
- After every transaction is completed, the database pages containing old versions is completed, of modified data need to be garbage collected/freed.
- Hard to extend algorithm to allow transactions to run concurrently (easier to extend log based schemes).

(b) What are the various reasons for a transaction to fail in the middle of execution?

Ans. A DBMS may encounter a failure. These failures may be of the following types:

Transaction failure: An ongoing transaction may fail due to:

- **Logical errors:** Transaction cannot be completed due to some internal error condition.
- **System errors:** The database system must terminate an active transaction due to an error condition (e.g., deadlock).

System crash: A power failure or other hardware or software failure causes the system to crash.

- **Fail-stop assumption:** Non-volatile storage contents are assumed to be uncorrupted by system crash.
- **Disk failure:** A head crash or similar disk failure destroys all or part of the disk storage capacity.

- **Destruction is assumed to be detectable:** Disk drives use checksums to detect failure.

All these failures result in the inconsistent state of a transaction.

(c) What is statistical database? Discuss the problem of statistical database security.

Ans. Statistical databases are used to provide statistical information or summaries of values based on various criteria. For example, a database for population statistics may provide statistics based on age groups, income levels etc. Statistical database users are allowed to access database to retrieve statistical information but not to access the detailed confidential information about specific individuals. Security for statistical databases must ensure that information about individuals cannot be accessed. It sometimes possible to deduce or infer certain facts concerning individuals from queries that involve only summary. Statistics on groups, consequently this must not be permitted either. This problem is called as statistical database security problem.

(d) What is Distributed DBMS? Explain its architecture. What are the advantages of DDBMS over centralized databases?

Refer to Page-168-170, Q.No.-2(i)

Now Refer to Page-107-109, Q.No.-64

MCS – 043: ADVANCED DATABASE DESIGN
Dec, 2008

Note: Question number 1 is ***compulsory****. Answer any* ***three*** *questions from the rest.*

Q1. (a) Consider the following relation R (A, B, C):
R

A	B	C
1	2	3
4	2	3
5	3	3
5	3	4

Which of the following dependencies does not hold? Give reasons.

(i) $A \rightarrow B$

Ans. Dependency $A \rightarrow B$ is hold by the relation R.

(ii) $A \rightarrow\rightarrow B$

Ans. Dependency $A \rightarrow\rightarrow B$ is hold by the relation R.

(b) What is a data warehouse? Describe the process of ETL for a data warehouse.
Ans. A Data Warehouse can be considered to be a “corporate memory”. It is a repository of processed but integrated information that can be used for queries and analysis. Data and information are extracted from heterogeneous sources as they are generated. Academically, it is subject-oriented, time-variant and a collection of operational data.
ETL is Extraction, transformation and loading. ETL refers to the methods involved in accessing and manipulating data available in various sources and loading it into target data warehouse. The following are some of the transformations that may be used during ETL:
- Filter Transformation
- Joiner Transformation
- Aggregrator Transformation
- Sorting Transformation

(c) What is timestamp ordering? Explain timestamp based protocol for serializable schedule.
Ans. Each transaction is issued a timestamp when it enters the system. If an old transaction T_i has time-stamp $TS(T_i)$, a new transaction T_j is assigned time-stamp $TS(T_j)$ such that $TS(T_i) < TS(T_j)$. This protocol manages

concurrent execution in such a manner that the time-stamps determine the serialisability order. In order to assure such behaviour, the protocol needs to maintain for each data Q two timestamp values:

- **W-timestamp**(Q) is the largest time-stamp of any transaction that executed **write**(Q) successfully.

- **R-timestamp**(Q) is the largest time-stamp of any transaction that executed **read**(Q) successfully.

The timestamp ordering protocol executes any conflicting **read** and **write** operations in timestamp order. Suppose a transaction T_i issues a **read**(Q).

1. If TS(T_i) <**W**-timestamp(Q), $\Rightarrow$ T_i needs to read a value of Q that was already overwritten. Action: reject **read** operation and rolled back T_i.

2. If TS(T_i) $\geq$ **W**-timestamp(Q), $\Rightarrow$ It is OK. Action: Execute **read** operation and set R-timestamp(Q) to the maximum of R-timestamp(Q) and TS(T_j).

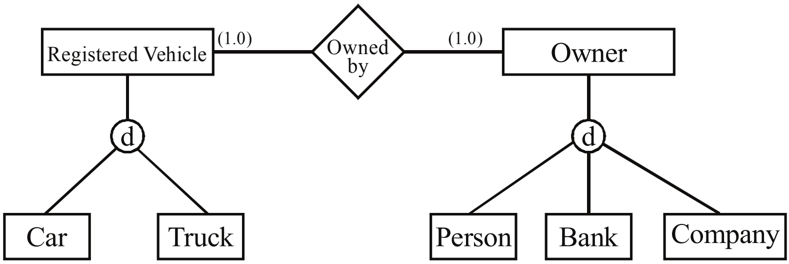
Suppose that transaction T_i issues **write** (Q):

1. If TS(T_i) < R-timestamp(Q), $\Rightarrow$ the value of Q that T_i is writing was used previously, this value should have never been produced. Action: Reject the **write** operation. Roll back T_i.

2. If TS(T_i) < W-timestamp(Q), $\Rightarrow$ T_i is trying to write an obsolete value of Q. Action: Reject **write** operation and roll back T_i.

3. Otherwise, execute **write** operation and set W-timestamp(Q) to TS(T_i).

(d) Consider the following requirements:
A car or truck can be a registered-vehicle. A person, bank or company can own a registered vehicle. Model this requirement using EER diagram.
Ans.



(e) Consider the following relations:
EMPLOYEE (Eno, Ename, Address, Phone, Salary, Dnum)
DEPT (Dnum, Dlocation, Dname)
PROJECT (Pno, Dname, Dnum)

Find the names of employees using relational algebra who work on all projects controlled by Department Number 5.

Ans.

$$\left(\pi ename,\ pno\left(Employee = Dept = Project\right)\right) \div \left(\pi pno\left(\sigma dnum = 5\left(Pr\,oject\right)\right)\right)$$

(f) How will you enforce Referential Integrity Constraint in Oracle? Explain with the help of one example.

Ans. The referential integrity constraint can be enforced in Oracle (i) at the time of table creation, and (ii) after the table creation.

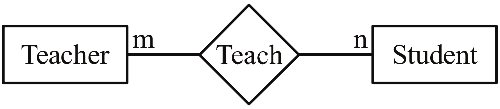
Example: (i) At the time of table creation

```
create table Dept
( DNo number (4) Primary key
DName var char2 (10));
create table emp
( ENo number (6) Primary key
Ename var char2 (20)
DNo number (4) references dept (dno));
```

(ii) After table creation

```
Alter table emp
Add constraint fk1 foreign key (dno) references
dept(dno);
```

(g) Create an object-oriented database using ODL for the following schema. Make suitable assumptions about attributes.



Ans. class Teacher

```
(Key Teacher id)
{
attribute string Teacher id;
attribute string name;
attribute string subject;
relationship set <student> student :: student;
};
class student
(Key Roll no)
}
```

```
attribute int Rollno;
attribute string sname;
attribute string class;
};
```

(h) What are the common database security failures? What are the SQL commands for granting permission? Why are statistical databases more prone to disclosure? Explain with the help of an example.

Ans. Common Database Security Failures: Database security is of paramount importance for an organisation, but many organisations do not take this fact into consideration, till an eventual problem occurs. The common pitfalls that threaten database security are:

Weak User Account Settings: Many of the database user accounts do not contain the user settings that may be found in operating system environments. For example, the user accounts name and passwords, which are commonly known, are not disabled or modified to prevent access.

Insufficient Segregation of Duties: No established security administrator role is defined in the database management of the organisation. This results in database administrators performing both the functions of the administrator, as well as the performance and operations expert. This may result in management inefficiencies.

Inadequate Audit Trails: The auditing capabilities of databases since it require keeping track of additional requirements, are often ignored for enhanced performance or disk space. Inadequate auditing results in reduced accountability. It also reduces the effectiveness of data history analysis.

Unused DBMS Security Features: The security of an individual application is usually independent of the security of the DBMS.

Permissions can be granted to a user or a role with the use of the SQL GRANT statement.

The syntax of this statement is:

```
GRANT <permissions>
[ON <table>]
TO <user/role>
[WITH GRANT OPTION]
```

Disclosure of data as indicated in the previous section is a major problem as disclosure may result in breach of security in some form or the other and thus, is not acceptable. Thus, the first step in this direction would be to reject any query that directly asks for sensitive information that is hidden. But, how about a sequence of queries that are raised for the purpose of statistics? For example, we may be able to determine the average marks obtained in a class of 50 student, but if only 2 students have opted for a subject then the first student who knows his/her marks can find the marks of the other student by issuing the average marks query. Thus, statistical queries should be permitted only when some minimum number of records satisfies a condition. Thus, the overall objectives are to make sure that security is not compromised.

Q2. (a) What is the significance of cursors in embedded SQL? Explain with the help of an example.

Ans. The Cursor can be defined as a pointer to the current tuple. It can also be defined as a portion of RAM allocated for the internal processing that has been set aside by the database server for database interactions.

Since most of the commercial RDBMS architectures are client-server architectures, on execution of an embedded SQL query, the resulting tuples are cached in the cursor. This operation is performed on the server. Sometimes the cursor is opened by RDBMS itself - these are called **implicit** cursors. However, in embedded SQL we need to declare these cursors explicitly - these are called **explicit cursors**.

(b) What are mobile databases? Explain the characteristics of mobile databases.

Refer to Chapter-7, Q.No.-10 and Q.No.-14

(c) What do you mean by deadlock? How can we prevent them? Write an algorithm that checks whether the concurrently executing transactions are in deadlock.

Refer to Chapter-5, Q.No.-2

Q3. (a) What is data mining? How is it different from OLTP? What is classification in context of data mining?

Refer to Chapter-6, Q.No.-50, Q.No.-53 and Q.No.-55

(b) What are data-marts? What is the significance of creating them?

Ans. Data marts can be considered as the database or collection of databases that are designed to help managers in making strategic decisions about business and the organisation. Data marts are usually smaller than data warehouse as they focus on some subject or a department of an organisation. Some data marts are also called dependent data marts and may be the subsets of larger data warehouses.

A data mart is like a data warehouse and contains operational data that helps in making strategic decisions in an organisation. The only difference between the two is that data marts are created for a certain limited predefined application. Even in a data mart, the data is huge and from several operational systems, therefore, they also need a multinational data model. In fact, the star schema is also one of the popular schema choices for a data mart.

(c) What is XML? How is it different from HTML? What are the advantages of XML? Create an XML schema for list of students and their marks.

Refer to Dec-2006, Q.No.-1(viii) and Chapter-6, Q.No.-22

```
<?xml version = “1.0” encoding = “VTF-8” standalone = “Type”?>
<!DOCTYPE document [
<!ELEMENT document (student) *>
<!ELEMENT Student (name, marks)>
<!ELEMENT NAME (# PCDATA)>
<!ELEMENT Marks (# PCDATA))]>
<document>
<student>
<name> XYZ </name>
<marks> 45 </marks>
</student>
</document>
```

Q4. (a) Compare and contrast Relational database management systems, Object-relational database management systems, Object-oriented database management systems. Suggest one application for each of these database management systems.

Refer to Chapter-6, Q.No.-2 and Dec-2006, Q.No.-3(i)

(b) Explain Join Dependency with the help of an example. To which normal form does it correspond? Functional dependencies and multivalued dependencies are special type of Join dependencies. Justify.

Ans. The fifth normal form deals with join-dependencies, which is a generalisations of the MVD. The aim of fifth normal form is to have relations that cannot be decomposed further. A relation in 5NF cannot be constructed from several smaller relations.

A relation R satisfies join dependency $*(R_1, R_2, ..., R_n)$ if and only if *R* is equal to the join of $R_1, R_2, ..., R_n$ where R_i are subsets of the set of attributes of *R*.

A relation *R* is in 5NF if for all join dependencies at least one of the following holds:

(a) *(R_1, R_2, ..., R_n) is a trivial join-dependency (that is, one of R_i is R)*

(b) *Every R_i is a candidate key for R.*

Q5. (a) Explain the centralized Two-phase Commit Protocol in Distributed Environment. Give the algorithm for both coordinator and participants.

Refer to Chapter-5, Q.No.-72 and Q.No.-75

(b) Develop a query plan for the following query and compute its cost: 'SALARY > 40000 (EMPLOYEE = DEPARTMENT)'

Make suitable assumptions of your own about the relation schema as well as the database statistics.

Ans. Assumption

Size of Employee relation = 1000 tuples

Size of Department relation = 400 tuples

No. of employees having salary > 4000 = 100

Step No.	**Operation**	**Cost of Query Processing time**	**Result Size**
1.	Employee = Department	1000 × 400	1000 tuples
2.	Selection operation	1000	100 tuples

MCS – 043: ADVANCED DATABASE DESIGN

Jun, 2009

***Note:** Question number 1 is **compulsory**. Answer any **three** questions from the rest.*

Q1. (a) Consider the following three transactions

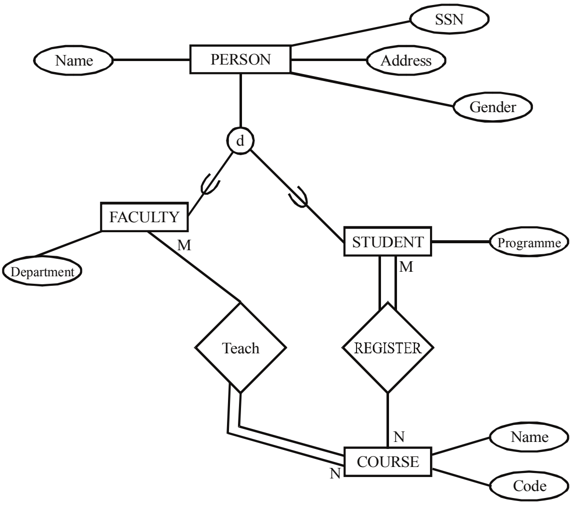
T_1	T_2	T_3
read (X)	read (X)	read (X)
X = X – 1000	display (X)	Y : = X
Write (X)		display (X)

Insert shared and exclusive locks in T_1, T_2 and T_3 such that the transactions when executed concurrently do not encounter any concurrency related problem.

Ans.

T_1	T_2	T_3
Lock-X(X)	Lock-S(X)	Lock-S(X)
Read (X)	Read (X)	Read (X) Y : = X
X = X – 1000	display (X)	display (X)
Write (X)	Unlock (X)	Unlock (X)
Unlock (X)		

(b) Consider the following EER Diagram



Derive relations from the above EER diagram.

Ans. Person (SSN, Name, Address, Gender)
Faculty (SSN, Department)
Student (SSN, Programme)
Course (Code, Name)
Teach (SSN, Code)
Register (SSN, Code)

(c) Define Simple Hash-Join and explain the process and cost calculation of Hash-Join with the help of example.
Make suitable assumptions of your own about the relation schema as well as the database statistics.

Refer to June-2008, Q.No.-1(d)

(d) Explain the characteristics of mobile databases. Give an application of mobile databases.

Refer to Dec-2008, Q.No.-2(b)

(e) The decision regarding indexing is a trade off between read-only queries and update queries. Elaborate the statement with the help of example.

Ans. If we are using read only queries then indexing gives very good performance as it provide binary search over linear search. The same performance is also provided with update queries while selecting the records but it introduces the overheads to update all the indexes created based on the table on which we are going to execute update query.

For example : Consider a relation emp(emp_no, name, city, deptno) with the indexes on emp_no , city and deptno.

If we want to use select query on emp table with selection criteria's on empno,city or deptno then it will give faster response time. But if we want to execute update query on this relation then all the three indices will be updated every time.

(f) Consider the following relational schema
emp (e-no, e-name, title)
pay (title, Sal)
Let P1: Sal < 3000 and
P2: Sal ≥ 3000 be 2 predicates
Perform a horizontal fragmentation of relation pay, with respect to these predicates, to obtain pay 1 and pay 2. Using this fragmentation of pay, perform further derived horizontal fragmentation of Emp based on title "Dr".

Ans. Pay1 (title, Sal) where Sal < 3000
Pay2 (title, Sal) where Sal ≥ 3000
Emp1(e_no, e_name, title) where title = "Dr"
Emp2(e_no, e_name, title) where title < > "Dr"

(g) What is ETL? What are different transformations that are needed during the ETL Process?

Refer to Dec-2008, Q.No.-1(b)

(h) Given the following semi structured data in XML create the DTD (Document Type Declaration) for it:

```
<document>
        <employee>
                <Name> Ramesh Jain </Name>
                <Address> H-1, 25, Delhi </Address>
                <Address> B-1, New office, Delhi </Address>
        </employee>
        <employee>
                <Name> Anuj </Name>
                <Address> 25, Gurgaon, Haryana </Address>
        </employee>
        </document>
```

Ans.

```
<?xml version = "1.0" encoding = "UTF – 8" standalone = "Yes"?>
<!DOCUMENT document [
<!ELEMENT document (employee) *>
<!ELEMENT employee (name, address *)>
<!ELEMENT name (# PCDATA)>
<!ELEMENT address (# PCDATA)>]>
```

Q2. (a) What is data mail and how it is different from dataware house?

Ans. The basic constructs used to design a data warehouse and a data mart are the same. However, a Data Warehouse is designed for the enterprise level, while Data Marts may be designed for a business division/department level. A data mart contains the required subject specific data for local analysis only.

(b) Consider the following query:
Select student-id, student-name, subject, marks from STUDENT, RESULT WHERE STUDENT. Student-id = RESULT.,Student-id
AND RESULT, MARKS > 60
Assume suitable relation and statistics.
Create a query evaluation plan for the above.

Ans. Assumptions

1. Size of the relation Student = 1000 tuples

2. Size of the relation Result = 2500 tuples

3. No. of tuples belongs to > 60 marks = 1200 tuples

Step No.	Operation	Cost	
		Intermediate Result	**Output**
1.	Student $\bowtie$ Result	1000 × 2500	2500 tuples
2.	Selection operation	2500	1200 tuples
3.	Projection operation	1200	1200 tuples

(c) State the main disadvantage of basic time stamp method for concurrency control. How can you overcome this advantage?

Refer to Dec-2006, Q.No.-1(vi)

(d) How audit trails is done in data base? How are they related to database security?

Refer to Chapter-5, Q.No.-60

Q3. (a) What were the limitations of relational databases and why there was a need for extending them to object-object databases? Consider we want to represent the information of a book comprising of ISBNNO, TITLE, CATEGORY (such as text, Reference, Journal) and list of AUTHORS create the database structure for:
(i) OODB using ODL (Object Definition Language)
(ii) an object relation database of Book.

Ans. Limitations of Relational Databases: Relational database technology was not able to handle complex application systems such as Computer Aided Design (CAD), Computer Aided Manufacturing (CAM) and Computer Integrated Manufacturing (CIM), Computer Aided Software Engineering (CASE) etc. The limitation for relational databases is that, they have been designed to represent entities and relationship in the form of two-dimensional tables. Any complex interrelationship like, multi-valued attributes or composite attribute may result in the decomposition of a table into several tables, similarly, complex interrelationships result in a number of tables being created. Thus, the main asset of relational databases viz., its simplicity for such applications, is also one of its weaknesses, in the case of complex applications.

(i) class Book

```
{ key ISBN No}
( attribute string ISBN No;
attribute string title;
attribute string category;
attribute string authors;
);
```

(ii) Book(ISBNNO, TITLE, CATEGORY, AUTHOR(Name, Area))

(b) Differentiate between JDBC and ODBC.

Ans. Database access from different interfaces may require some standards. The two standards that are quite useful are ODBC for accessing a database from any DBMS environment that support ODBC and JDBC that allows JAVA program access databases.

Open Database Connectivity (ODBC): Open Database Connectivity (ODBC) is a standard API (Application Programming Interface) that allows access to as database that is a part of any DBMS that supports this standard. By using ODBC statements in a program one may access any database in MS-Access, SQL Server, Oracle, DB2, even an Excel file, etc. to work with ODBC driver for each of the database that is being accessed from the ODBC. The ODBC allows program to use standard SQL commands for accessing database. Thus, we need not master the typical interface of any specific DBMS.

For implementing ODBC in a system the following components will be required:

- the applications themselves,
- a core ODBC library, and
- the database drives for ODBC.

Java Database Connectivity (JDBC): Accessing a database in Java requires, Java Database Connectivity (JDBC).

Java Database Connectivity (JDBC) provides a standard API that is used to access databases, regardless of the DBMS, through JAVA. There are many drivers for JDBC that support popular DBMSs. However, if no such driver exits for the DBMS that we have selected then, we can use a driver provided by Sun Microsystems to connect to any ODBC compliant database. This is called JDBC to ODBC Bridge. For such an application, we may need to create, an ODBC data source for the database before, we can access it from the Java application.

Connecting to a Database: In order to connect to a database, let us say an oracle database, the related JDBC driver has to be loaded by the Java Virtual Machine class loader successfully.

(c) What are various mechanism to deal with Dead lock? Explain any one with the help of example.

Refer to Chapter-5, Q.No.-21

Q4. (a) Explain MVD (Multi valued dependency) and Join Dependency with the help of an example each. Given the relation R{ABCDE} with FDS

$\{A \to BCDE,\ B \to ACDE,\ C \to ABDE\}$.

Where the join dependencies for R? Give thelosslessjoin decomposition for R.

Ans. Multivalued dependency: *The multivalued dependency $X \to\to Y$ is said to hold for a relation R(X, Y, Z) if, for a given set of value (set of values if X is more than one attribute) for attribute X, there is a set of (zero or more) associated values for the set of attributes Y and the Y values depend only on X values and have no dependence on the set of attributes Z.*

Please note that whenever $X \to\to Y$ holds, so does $X \to\to Z$ since the role of the attributes *Y* and *Z* is symmetrical.

The fifth normal form deals with join-dependencies, which is a generalisations of the MVD. The aim of fifth normal form is to have relations that cannot be decomposed further. A relation in 5NF cannot be constructed from several smaller relations.

A relation R satisfies join dependency **(R_1, R_2, ..., R_n)* if and only if *R* is equal to the join of *R_1, R_2, ..., R_n* where *R_i* are subsets of the set of attributes of *R*. A relation *R* is in 5NF if for all join dependencies at least one of the following holds:

(a) *(R_1, R_2, ..., R_n) is a trivial join-dependency (that is, one of R_i is R)*

(b) *Every R_i is a candidate key for R.*

Join Dependency is $B \rightarrow C$

R_1 (ABDE)

R_2 (BC)

R_3 (CA)

(b) How OLAP support query processing in dataware house?

Ans. Data warehouses are not suitably designed for transaction processing, however, they support increased efficiency in query processing. Therefore, a data warehouse is a very useful support for the analysis of data.

On Line Analytical Processing (OLAP) is an approach for performing analytical queries and statistical analysis of multidimensional data. OLAP tools can be put in the category of business intelligence tools along with data mining. OLAP tools require multidimensional data and distributed query-processing capabilities. Thus, OLAP has data warehouse as its major source of information and query processing.

(c) Differentiate between embedded SQL and dynamic SQL. Give an example of embedded SQL.

Refer to Chapter-4, Q.No.-8

Q5. Explain the following with the help of am example diagram, if any:

(a) Semantic databases

Refer to June-2008, Q.No.-2(c)

(b) Applications of data grid

Refer to Chapter-7, Q.No.-20

(c) Security features of oracle

Ans. Oracle includes security features that control the accessing and using of a database. For example, security mechanisms in Oracle:

- prevent unauthorized database access,

- prevent unauthorized access to schema objects, and
- audit user actions.

Associated with each database user is a schema by the same name. By default, each database user creates and has access to all objects in the corresponding schema.

Database security can be classified into two categories: system security and data security.

(d) Challenges in design of multimedia-database

Refer to Chapter-7, Q.No.-4

(e) Benefits of data dictionary

Refer to Chapter-4, Q.No.-20

(f) Steps of database design

Ans. 1. Logical Database Design using ER Modeling
2. ER to Relational Mapping
3. Normalization
4. Physical Database Design
5. Database Tuning
6. Denormalization

(g) Clustering in data mining

Refer to Chapter-6, Q.No.-56

(h) Application of data mining

Ans. Some of the applications of data mining are as follows:

- **Marketing and sales that analysis:** A company can use customer transactions in their database to segment the customers into various types. Such companies may launch products for specific customer bases.
- **Investment analysis:** Customers can look at the areas where they can get good returns by applying the data mining**.**
- **Loan approval:** Companies can generate rules depending upon the dataset they have. On that basis they may decide to whom, the loan has to be approved.
- **Fraud detection:** By finding the correlation between faults, new faults can be detected by applying data mining.
- **Network management:** By analyzing pattern generated by data mining for the networks and its faults, the faults can be minimized as well as future needs can be predicted.
- **Risk Analysis:** Given a set of customers and an assessment of their risk-

worthiness, descriptions for various classes can be developed. Use these descriptions to classify a new customer into one of the risk categories.

- **Brand Loyalty:** Given a customer and the product he/she uses, predict whether the customer will change their products.
- **Housing loan prepayment prediction:** Rule discovery techniques can be used to accurately predict the aggregate number of loan prepayments in a given quarter as a function of prevailing interest rates, borrower characteristics and account data.

MCS – 043: ADVANCED DATABASE DESIGN
December, 2009

Note: *Question number **1** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) Consider the following query:
SELECT ENAME, PNAME
FROM EMP, ASG, PROJ
WHERE
PROJ. TITLE = "Comp Engg"
AND ASG. ENO = EMP. ENO
AND ASG. PNO = PROJ.PNO
Assuming that the size of EMP relation is 400 tuples, PROJ has 100 tuples and ASG has 20,000 tuples. There are no indexes. Derive a query execution plan for above. Make suitable assumptions.

Ans. Assumption.
(1) Size of the relation Emp = 400 tuples
(2) Size of the relation ASG = 20,000 tuples
(3) Size of the relation PROJ = 100 tuples
Step No.

Operation	**Cost**	
	Intermediate Result	**Output**
(1) Proj. Title = "comp Eng" AND ASG. ENO = EMP.END AND ASG. PNO = PROJ.PNO	100 x 400 x 2000	20000
(2) Selection Operation for		
Emp PROJ ASG	400 100 20000	400 100 20000

(b) Two-phase locking protocol uses waiting, time stamping and optimistic methods use restacting, to avoid nonserializable execution. Justify the statement.

Ans. This protocol manages concurrent execution in such a manner that the timestamps determine the serialisability order. Each timestamp based transaction is issued a timestamp when it enters the system. If an old transaction Ti has timestamp Ts(Ti), a new transaction Tj is assigned timestamp Ts(Tj) such that Ts(Ti)<Ts<(Tj). There are two timestamp date(q) values.

W-timestamp (q) is the largest timestamp of any transaction that executed write q successfully.

R-times (q) is the largest time-stamp of any transaction that executed read (q) successfully. Serialisability order is determined by a timestamp given at the time of validation to increase concurrency. Thus TS(Ti) is given the value of validation (TJ).

(c) What is the difference between document type definition and XML schema? Explain using an example.

Ans. Both DTD and schemas are document definition languages, schema are written in XML, while DTDs use EBNF (Extended Bakus Naur Format) notation Thus, schema are extensible as they are written in XML. They are also easy to read, write and define.

DTD provide the capability for validation of the following:

- Element nesting
- Element occurrence constraints
- Permitted attributes
- Attribute types and default values.

However DTD do not provide control over the format and data types of element and attributes value; e.g. once an element or attribute has been declared to contain character data, no limits may be placed on the length, type, or format of that content.

XML standard includes

- Simple and complex data types
- Type derivation and inheritance
- Element occurrence constraints
- Name-space-aware elements and attribute declaration.

XML can use simple data types for passed character data and attribute values, and can also enforce specific rules on the contents of elements and attributes than DTDs.

(d) What are the differences between datamining and knowledge discovery in database (KDD)? Can these two techniques be used alternatively. Justify.

Ans. Refer to Ch-6, Q.No.-51(ii), Page No.-133 and;

No, we cannot use these techniques alternatively because one is the process of discovering the information and knowledge in data while other is the process of extracting these information and knowledge.

(e) What is a join dependency? Explain with an example. When a join dependency is referred as trivial?

Ans. A join dependency is a constraint on the set of legal relations over a database scheme. The join dependency plays an important role in the Fifth normal form, also known as project-join normal form, because it can be proven that if you decompose a scheme R in tables R1 to Rn, the decomposition will be a lossless-join decomposition if you restrict the legal relations on R to a join dependency on R called * (R1,R2,...Rn). It is to say that the set of relationships in the join dependency is independent of each other.

Example : Given a pizza-chain that models purchases in table Customer = { order-number, customer-name, pizza-name, delivery-boy }. It is obvious that you can derive the following relations:

customer-name depends on order-number
pizza-name depends on order-number
delivery-boy depends on order-number

Since the relationships are independent you can say there is a join dependency as follows: *((order-number, customer-name), (order-number, pizza-name), (order-number,delivery-boy)).

If each customer has his own delivery-boy however, you could have a join-dependency like this: *((order-number, customer-name), (order-number, delivery-boy), (customer-name, delivery-boy), (order-number,pizza-name)), but *((order-number, customer-name, delivery-boy), (order-number,pizza-name)) would be valid as well. This makes it obvious that just having a join dependency is not enough to normalize a database scheme.

Let R be a relation schema and let R1,R2,...,Rn be a decomposition of R.

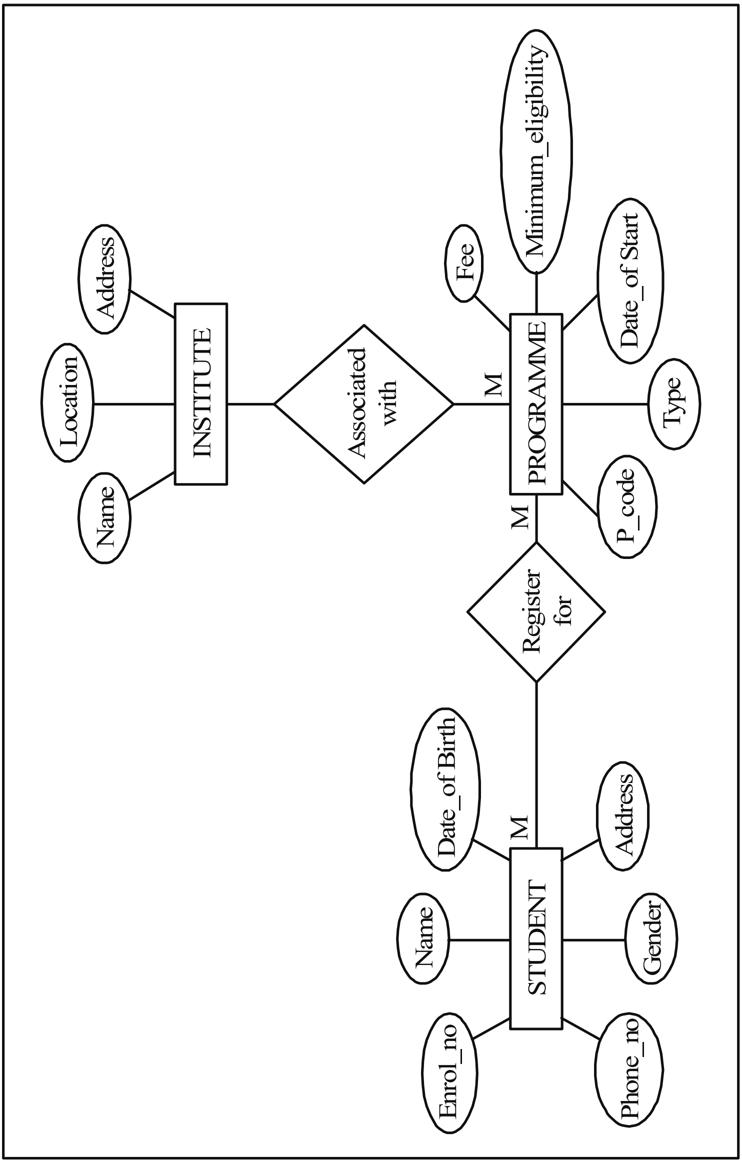
The relation r(R) satisfies the join dependency * (R1,R2,...Rn) if .

A join dependency is trivial if one of the Ri is R itself.

(f) Consider a small institute in which students register for programmes run by the institute. A program can be a full or a part time program or both. Every student necessarily registers in at least one programme and at most three programme. Assuming suitable attributes, design an EER diagram for the same.

Ans. Given two entities Students and programme run by the institute. Here in student entity possible assumption could be Enrolment_no, Name, DOB, Gender, Address, Phone no., and Program Entity will constitute of P-code, P-name, P_type, Date of start, Minimum eligibility, Fee.

Addition to these there will be another Entity Institute which will have attribute Institute Name, Location, and Address.



ER Diagram for Institute

(g) What are the different types of index implementations available in POST gre SQL. Explain each one of them.

Refer to Ch—7, (Indexes), Page No.-150

(h) What are triggers and what is their use? Explain with the help of an example.
Refer to Q.No.-36, Page No.-92

Q2. (a) Why is query expressed in relational algebra preferred over query expressed in SQL in query optimization. Explain this by taking a suitable example.

Ans. Query expressed in relational algebra preferred over query expressed in SQL in query optimization because :

1) It is more procedural, because algebraic expressions also give the order of application of operations in the computation of the query; thus, it is a more appropriate model for query optimization and system implementation.

2) Many existing approaches to query optimization and equivalence are (or can easily be) expressed in relational algebra.

3) Relational algebra uses relations (or sets) as operands, while calculus is based on tuple variables. For the optimization of queries, particularly with distributed environments or special-purpose database machines, set-oriented models are more appropriate than tuple-oriented models.

Example : Let us consider the relational schema:

S(S#,NAME), storing supplier's name and number SP(P#, S#), storing supplier and part numbers for a supply and a typical SQL query with nesting, taken from Date's book [3, sect. 7.2.14]. The query retrieves the name of suppliers who do not supply product 'P2':

SELECT NAME FROM S WHERE 'P2' * ALL SELECT P# FROM SP WHERE S#=S. S#

Notice that the clause ALL is used to indicate that only those suppliers should be retrieved for which there is no supply tuple with P# equal to P2; if the ALL clause were omitted, then the query would retrieve suppliers having at least one supply tuple with P# different than P2. The naming transformation produces the following query, in which attribute names are extended with relation names:

SELECT S.NAME FROM S WHERE 'P2' * ALL SELECT SP.P# FROM SP WHERE SP.S#=S.S#

This query can be parsed using EG; the preprocessing produces as result the equivalent query (without the ALL keyword):

(SELECT S.NAME FROM S MINUS SELECT S.NAME FROM S WHERE 'P2' = SELECT SP.P# FROM SP WHERE SP.S#=S.S#)

This query can be parsed using RG; thus, its meaning can be evaluated. We anticipate the notation for some algebraic operations:

PJ [A] (projection over a set A of attributes)

SL[p] (selection of tuples satisfying predicate p)

CP (Cartesian product) DF (difference)

JN[jp] (oin with join predicate jp). We use a prefix notation for unary operations

(e.g., PJ and SL), and an infix notation for binary operations (e.g., CP, DF, and JN). The meaning evaluation produces the following expression:
(PJ[S.SNAME] S)DF
(PJ [S.SNAME] SL[SP.P#= 'P2']
(PJ[SP.P#,S.NAME,S. S#] SL[SP.S#=S. S#; (S CP SP)))
Finally, the postprocessing step is applied to the above expression. The postprocessing eliminates useless projections and transforms selections over Cartesian products into joins, giving:
(PJ [S. SNAME] S) DF
(PJ [S.SNAME] ((SL[SP.P#= 'P2'] S) JN [SP. S#=S. S#] SP))

(b) Determine all 4NF violations for the relation schema R(X, Y, Z, W) with multivalued dependencies X→ →Y and X→ →Z. decompose the relation into 4NF.

Ans. A relation schema R is in 4NF with respect to a set D of functional and multivalved dependencies if for all multivalued dependencies in D of the form a → →b where ac̲R and bc̲R. At last one of the following hold:
a → →b is trival (ie) bc̲a or a au̲b = R)
a is a superkey for schema R. Since here in example there is no functional dependencies as the only key or all four attributes x, y,z,w. Thus each of the non trivial multivalued dependencies x → →y and x→ →z voilates 4NF. We must separate out the attributes of these dependencies, first decomposing into x,y and xzw. And then further decomposing the latter into xz and xw because x → →z is still a 4NF violation for xzw. xy, xz and xw. Also we know that a relation is in 4NF only if it is in BCNF and there are no true MVDs.

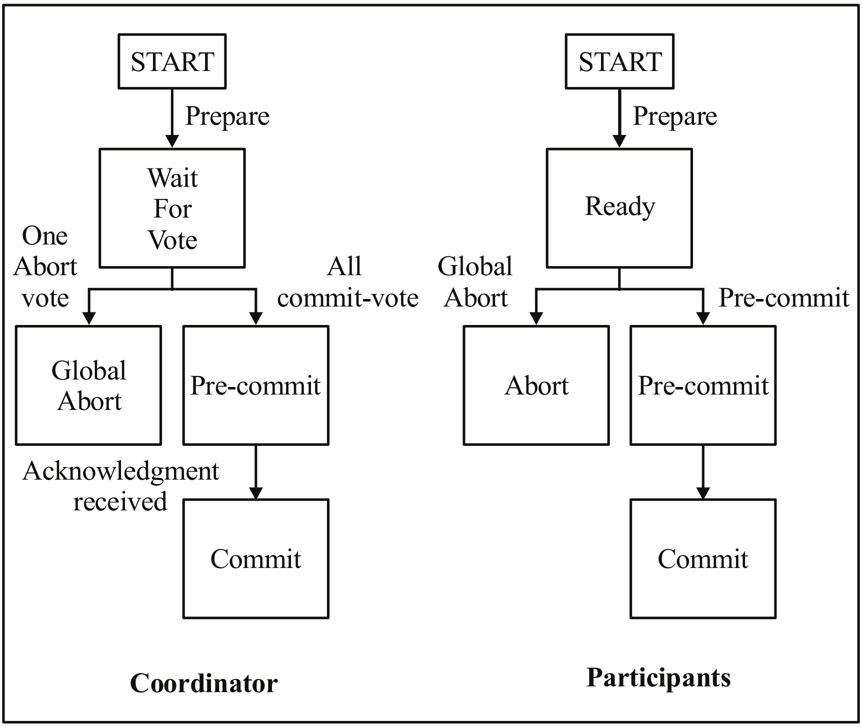
(c) The 3-phase commit protocol increases the system's availability and doesn't allow transaction to remain blocked until a failure is repaired. Justify the statement.

Ans. Refer to Ch–5, Q.No.-73, Page No.-111
The disadvantage with 2PC was that it can block participants in certain circumstances. For example process that encounter a timeout after voting COMMIT, but before receiving the global commit or abort message from the coordinator, are awaiting for the message and doing nothing or in other words are blocked.
But in case of 3PC it does not block the participants on site failures, except for the case when all sites fail. Thus the basic conditions that this protocol requires are:

- No network partitioning
- At least one available site
- At most k failed sites (called K-resilent)

The basic objective for 3PC is to remove the blocking period for the participant who have voted commit, and are thus, waiting for the global abort/commit message from the coordinator. This objective is met by adding another phase-called pre-commit is introduced between the voting phase and global decision phase. If a coordinator receives all the commit votes from the participants, it issues a global pre-commit message from the participant. A participants on receiving the global pre-commit messages knows that this transaction is going to commit definitely. The coordinator on receiving the acknowledgement of pre-commit message from all the participants issues the global COMMIT. A Global ABORT is still handled the same way as that of in 2 PC.

Thus, commit protocols ensure that DDBMS are in consistent state after the execution of a transaction.



(d) What are cursors? Explain with an example.

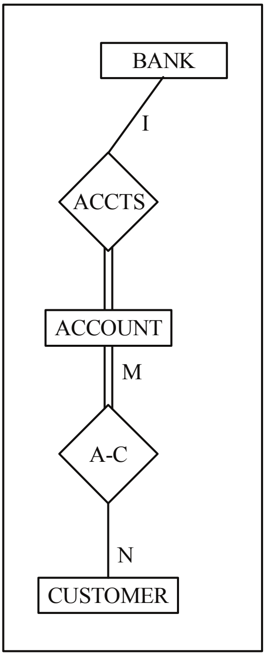
Ans. the cursor can be defined as a pointer to the current tuple. It can also be defined as a portion of RAM allocated for the internal processing that has been set aside by the database server for database interaction. This portion may be used for Query processing using SQL.

There are two types of cursors:

(1) Implicit cursors: This cursor is open implicitly by the RDBMS and after performing operation closed implicitly.

(2) Explicit cursor : These cursors are declared by the user in embedded SqL explicitly. Every cursor will have to be defined like:
Declare
Open
Close
For example :
Create cursor C_1 is
Select sname, enrolment, course From STUDENT; To Open cursor
Open C_1.

Q3. (a) Create an object oriented database using ODL for the following scheme. Make suitable assumptions about the attributes.



Answer the following query using OQL. List all the account of a customer whose name is 'Q' "XYZ".

Ans. The ODL for the following scheme is as follows after taking consideration into the following assumption.

BANK	ACCOUNT	CUSTOMER
Bank_ID	Type	Acc_no
Name	Acc_No	Cust_name
Branch_ID	Balance	Address
Acc_No	Open_Deete	Balance

Now the ODL for the following scheme is:

```
Class Bank
          { attribute Integer Bank_ID;
            attribute String Names;
           attribute Integer Branch_ID
              attribute Integer AccNo;
           };
Class Account
          { attribute String Type;
            attribute Integer Acc_No;
           attribute Integer Balance;
           attribute Date open_date;
          };
Class Customer
          { attribute Integer Acc_No;
           attribute String cust_name;
          attribute Integer Balance;
          attribute Struct Address
                   {String Add1, String city, String State, Integer Pincode}
                   Address last;
          };
```

In addition to these there will be some relationship set like.

```
Class Bank
{attribute Inter Bank_ID;
 attribute String Name;
                attribute Integer Branch_ID;
                attribute Integer Acc_No;
                relationship set <Account> openst
      inverse Account :: Opens by;
                 };
```

OQL:

Select C.Acc_No from customer C Where C. Name = ‘Q’ Or e.Name = ‘xyz’;
This Query is also written in the form of relationship as
Select C.Acc_No From customer C where c.open by. Name =’Q’ Or c.openby. Name = ‘xyz’;

(b) “A linear join help to exhibit parallelism in query execution”. Is the statement correct? Justify your answer.

Ans. A Linear Join can act as a tree with no branches. Each table in hierarchy is related to the table above it until we reach the top table.

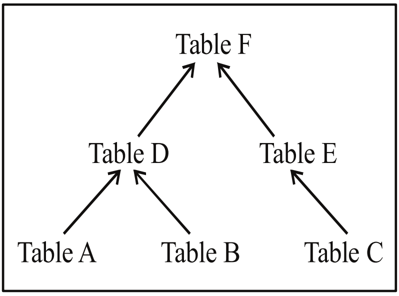


Table A,D,F constitute a linear hierarchy. Table CEF form another Linear Join hierarchy.

In a Linear Join, each pair of tables represent a one-to-many relationship, which the lower table of the pair in the “one” side and the higher table of the pair is the “many” side. Linear Join hierarchies can rely on any of the underlying Join conditions. Key Join, natural join, or On clause Join.

(c) Explain the “Deferred database modification” approach of log-based recovery.

Ans. The deferred database modification scheme records all the modification to the Log, but defers all the writes to after partial commit. Let us assume that transaction executes serially :

A transaction starts by writing <Ti start> record to Log. A write (x) operation results in a log record <Ti,X,V> being written, where V is the new value of X. The write is not performed on x at this time, but is defined. When Ti partially commits, <Ti commit> is written to the Log. Finally, the log records are read and used to actually execute the previously deferred writes. During recovery after a crash, a transaction needs to be redone if both <Ti Start> and <Ti commit> are there in the log. Redoing a transaction Ti (redoTi) sets the value of all data item updated by the transaction to the new values. Crashes can occur while

- the transaction is executing the original update or
- while recovery action is being taken.

(d) Define Multi-Version two-phase locking.

Refer to Page No.-197, (Multi–Version Two-Phase Locking)

Q4. (a) What is semi-structured data, explain with example. What is the difference between a well formed XML document and a valid XML document?

Refer Ch–6, Q.No.-17, Page No.-120 and;
Refer to Ch–6, Q.No.-27, Page No.-122

(b) What are views and what is their significance. How views are created in SQL explain using one example.

Refer to Ch–4, (View Definition), Page No.-49, and 50

(c) What is data warehousing? Discuss various characteristics of Data warehousing?

Refer to Page No.-125
Refer to Ch–4, (Characteristics of Data ware housing), Page No.-129

(d) Define multimedia databases and challenges in designing them.

Refer Ch–7, Q.No.-1, Page No.-140
Refer Ch–7, Q.No.-4, Page No.-142

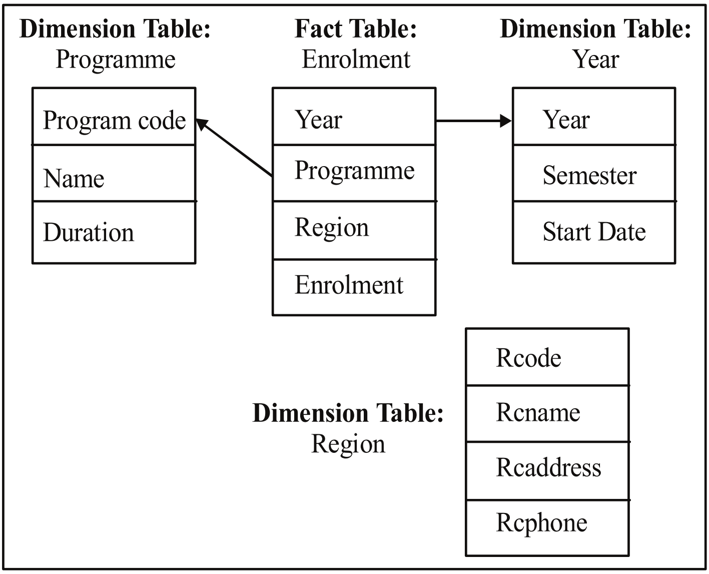
Q5. (a) Differentiate between star scheme and snowflake scheme using the same example.

Ans. A multidimensional storage model contains two types of tables : The dimension tables and the fact table. The dimension tables have tuples of dimension attributes, whereas the fact tables have one tuple each for a rewarded fact. In order to relate a fact to a dimension, we may have to use pointers. Let us demonstrate this with the help of an example.

Consider the University data warehouse where one of the data tables is the student enrolment table. The three dimension in such a case would be

Year
Programme
Region.

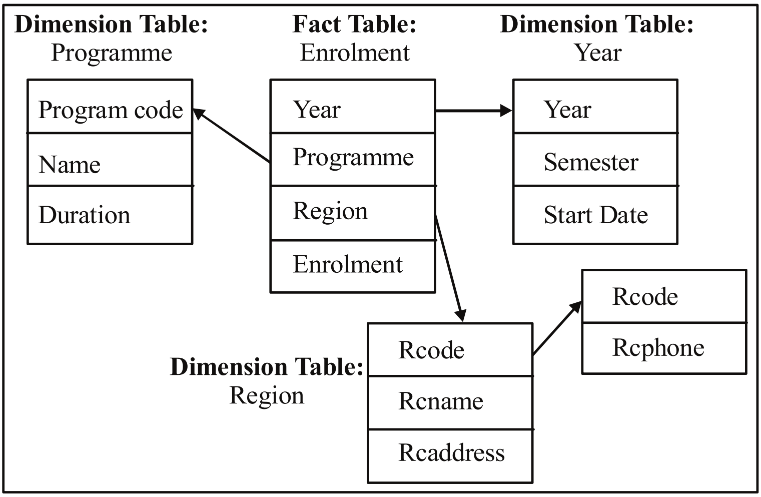
The star schema for such a data is



A star schema

Each dimension tables is a table for a single dimensions only and that is why this schema is known as star schema.

SNOFLAKE Schema : A snowflake schema has normalized but hierarchical dimensional tables. E.g. in star schema by Region dimension table, the value of the field Rcphone is multivalued, the Region dimension table is not normalized. Thus, we can create a snowflake schema for such situation as follows:



Data ware housing storage can also utilise indexing to support high performance access. Dimensional data can be indexed in star schema to tuples in the fact table by using a join index.

(b) Differentiate between:
(i) Data-mining queries and data base query
Refer to Page No.- 132 (Database Processing Vs. Data Mining Processing)

(ii) Clustering and classification approaches in data-mining.
Refer to Page No.-137(Approaches to Data Mining Problems)

(c) How does Oracle manage database security?
Ans. Refer to Ch–8, Q.No.-24, Page No.-151
While discussing Oracle database security issues we must recognize are secrecy or confidentiality, integrity, and availability. We know that the client must connect to the database via the listener. The client uses SQL*Net to "transfer requests from the client to the server via the underlying protocol and Operating System."(Oracle Corporation) Some examples of underlying Operating System protocols that SQL*Net supports are TCP/IP, SPX, and Digital DEC Pathworks. There are security issues inherent in any of the underlying Operating System protocols. As Oracle permissions are based on privileges. Users must access the database using their username and password. Once connected, users are able to create objects in their own schema only if they have a quota on their default tablespace. It is important to realize that users can only create objects in tablespaces that they have quotas on. An example of a user misusing quotas would be if he created a table in the tablespace userdata and then filled the tablespace by adding too many rows to the table. This may cause other users who had objects in the tablespace userdata to not be able to insert or update their data. It is important to examine user quotas periodically. There are many types of privileges associated with an Oracle database. This document will only examine very basic privileges like select, insert, update, and delete.

(d) What are deadlocks? How are they detected? Explain with the help of an example.
Refer to Ch–5, Q.No.-21, Page No.-79

.

MCS – 43: ADVANCE DATABASE DESIGN
June, 2010

***Note :** Question number **1** is **compulsory.** Answer **any three** questions from the rest.*

Q1. (a) The ABC Bank offers five types of Accounts : loan, checking, savings, daily interest saving and money market. It operates a number of branches within the country. A client of the bank can have any number of accounts. Accounts can be self or a joint account.

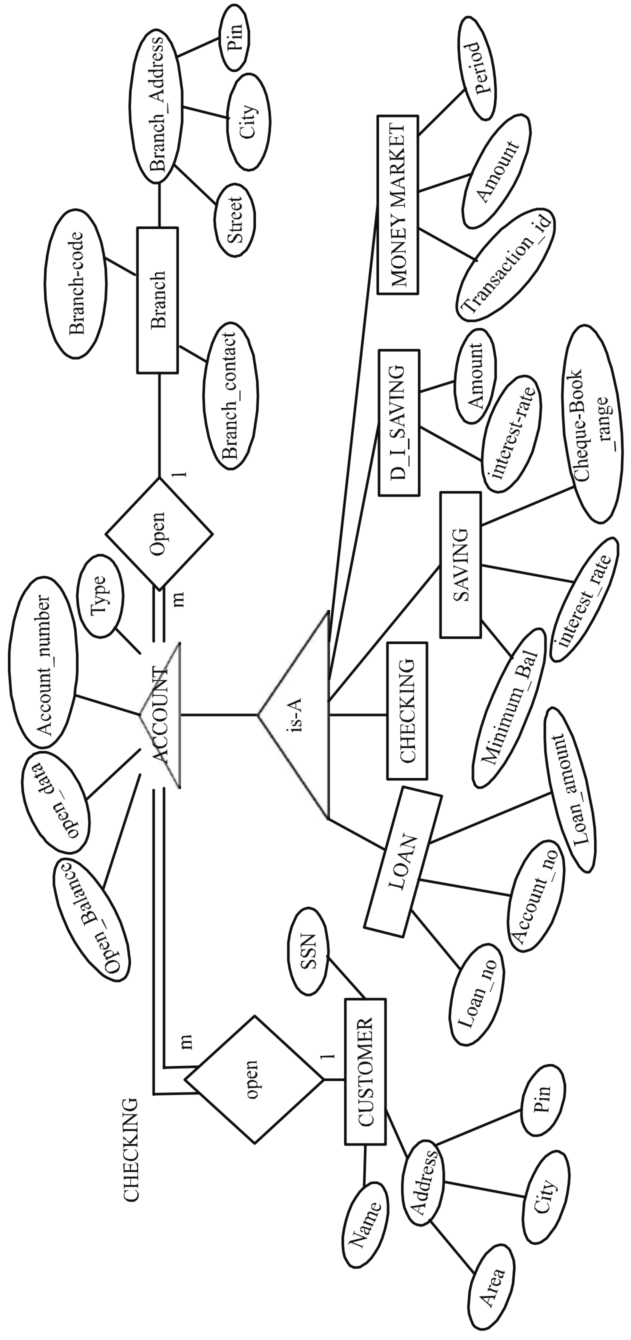
(i) Draw an EER diagram for the ABC bank identifying various entities, attributes and cardinality. Show meaningful relationship that exists among the entities.

Ans. Entities in the given problem will be Account, Branch, and customer.

Attributes of Account(entity) are Account-number, Type, open-date, opening-balance. For Branch entity Branch-code, Branch-contact, Branch-Address. For customer entity SSN, Name, Address etc. In Addition to these there can be worker entity.

Bank operates number of branches in a city. It also keeps track of branch details. It maintains branch information like branch-code, branch-address, branch-contact, branch-city. Branch code is unique for all branches. So branch is a strong entity.

A Branch can have many accounts while an Account is opened in one Branch only. Double lines indicates each account entity must have been opened in branch.



EER Diagram for ABC Bank

(ii) Translate the EER diagram to schema Relational Model.

Ans. For every ER-diagram we can construct a relational database which is a collection of tables.

Conversion of entity sets:- for each strong entity type E in ER diagram , we create a relation R containing all the simple attributes of E. The primary key of the relation R will be one of the key attributes of R.

ACCOUNT

Account-no:PK	Type	Open-Date	Open-Balance

BRANCH

Branch-code: Primary key	Brach-contact	Branch-Address

CUSTOMER

SSN: Primary key	Name	Address

Here between the table Branch and ACCOUNT there will be I:N relationship because a Branch can open many accounts. Branch-code is primary key in the branch table where as Account-number is the primary key in the Account table. In customer table SSN will be primary key because a particular SSN can be applied to only one people.

An addition to these there will be some more tables like LOAN, SAVING, D-I-SAVING, MONEY MARKET, CHECKING ETC.

LOAN

Loan-no: PK	Account-no: PK	Loan-amount

Here in this table both the Loan-no and Account-no will form composite Primary key and act as Primary key.

SAVING

MINIMUM-Balance	Interest-rate	Cheque-book range

MONEY MARKET

Transaction-id	Amount	Period

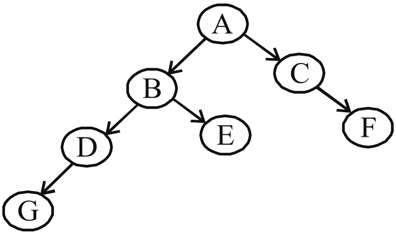
DAILY-INTEREST-SAVING

Account-no: Primary key	Interest-rate	Amount

(b) Explain the following protocols for concurrency control in transactions with the help of an illustration for each :

(i) Tree-protocol

Ans. In a tree based protocol only exclusive locks are permitted. The first lock by Ti may be on any data item. Subsequently, a data Q can be Locked by Ti only if the parent of Q is currently locked by Ti. Data items may be unlocked at any time-following figure shows a tree based protocol for the following lock (exclusive) – unlock requests from various transactions.



Tree based protocol:

Transaction T_1	Transaction T_2	Transaction T_3
Lock A	Lock B	Lock A
Lock B	Lock D	Lock C
Lock E	Lock G	Lock F
Unlock A	Unlock B	Unlock A
Unlock B	Unlock D	Unlock C
Unlock E	Unlock G	Unlock F

The tree protocol ensures conflict serialisability as well as freedom from deadlock for these transactions. Please note that transaction T_3 will get A only after it is set free by Transaction T_1. Similarly, T_2 will get B when it is set free by T_1. Unlocking may occur earlier in the tree-locking protocol than in the two-phase locking protocol. Thus tree protocol is a protocol with:-

- Shorter waiting time and increased concurrency
- That is deadlock free and does not require roll back.
- Where aborting a transaction can still lead to cascading roll backs.

However, in the tree-locking protocol a transaction may have to lock data time that is does not access. Thus it has:

- Increased locking overhead, and additional waiting time.
- Potential decrease in concurrency.

(ii) Timestamp-Based Protocol.

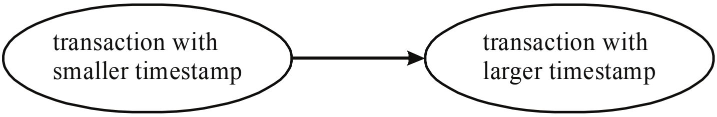
Ans. Time stamp-based Protocols:- Each transaction is issued a timestamp when it enters the system. If an old transaction Ti has time-stamp Ts(Ti), a new transaction Tj is assigned time-stamp Ts(Tj) such that Ts(Ti) < Ts(Tj). This protocol manages concurrent execution in such a manner that the time stamp determine the serialisability order. In order to assure such behaviour the protocols need to maintain for each data Q two timestamp values.

W-timestamp(Q) is the largest time-stamp of any transaction that executed write (Q) successfully.

R-time-stamp(Q) is the largest time-stamp of any transaction that executed read (Q) successfully.

The timestamp ordering protocol executes any conflicting read and write operations an timestamp order.

The timestamp ordering protocol guarantees serialisability if all the arcs in the precedence graph of the form as shown in following figure


transaction with smaller timestamp
transaction with larger timestamp

Timestamp protocol ensures freedom from a deadlock as no transaction can wait but the schedule may not be cascade-free, and may not even recoverable.

(c) With he help of a process diagram, explain the various tasks involved in the Knowledge Discovery in Databases (KDD) process.

Refer to Q.No.-4(a), Page No.-207

(d) Explain the role of ODBC and JDBC with the help of an example.

Ans. Refer to Page No.-179

Refer to Q.No.-3(b)(i), Page No.-190

(e) Is the following XML document well formed? Justify your answer :

<?xml version = “1.0” standalone= “yes” ?>
< employee>
< name > Amit </ name >
< position > Professor </ position>
</ employee >
< employee >
< name > Sumit >/name >
< position > Reader </ position >
</ employee >

Ans. No, the given XML document is not well formed. Here the tag opens at the beginning and closes at the end of the beginning statement as a well-formed XML document, these must be one root element that contains all the others.

Q2. (a) What are multimedia databases (MMDBs)? List some of the applications of MMDBs. Describe various contents of MMDBs. Also, mention the challenges in designing of MMDBs.

Refer to Ch–7, Q.No.-1, 2, 3 and 4, Page No.-140

(b) Define Multi-valued dependencies and Join dependencies. Give an example of each. State fourth and fifth normal form.

Refer to Q.No.-1(i), Page No.-161

Now Refer to Dec-2009, Q.No.-1(e)

Now Refer to Page No.-183, and 7.

Q2. (c) Consider a relation R (A, B, C) with functional dependencies $AB \longrightarrow C$ and $C \longrightarrow A$. Decompose the relation R into BCNF relations.

Ans. Boyce-codd Normal form is an extension of 3NF:-

Definition:- A relation is in BCNF if it is in 3NF and if every determinant is a candidate key.

Here the given relation is R(A, B, C) with functional dependencies $AB \rightarrow C$ and $C \rightarrow A$.

There is one structure of FD's that causes trouble when we decompose $AB \rightarrow C$ and $C \rightarrow A$.

There are two keys {A, B} and $\{A, C\}$.

The problem is that we use AC and BC as our database schema, we cannot enforce the functional Dependences. We cannot enforce the FD^{s} $AB \rightarrow C$ by checking the FD $AB \rightarrow C$ by checking $FD^{'s}$ in these decomposed relation.

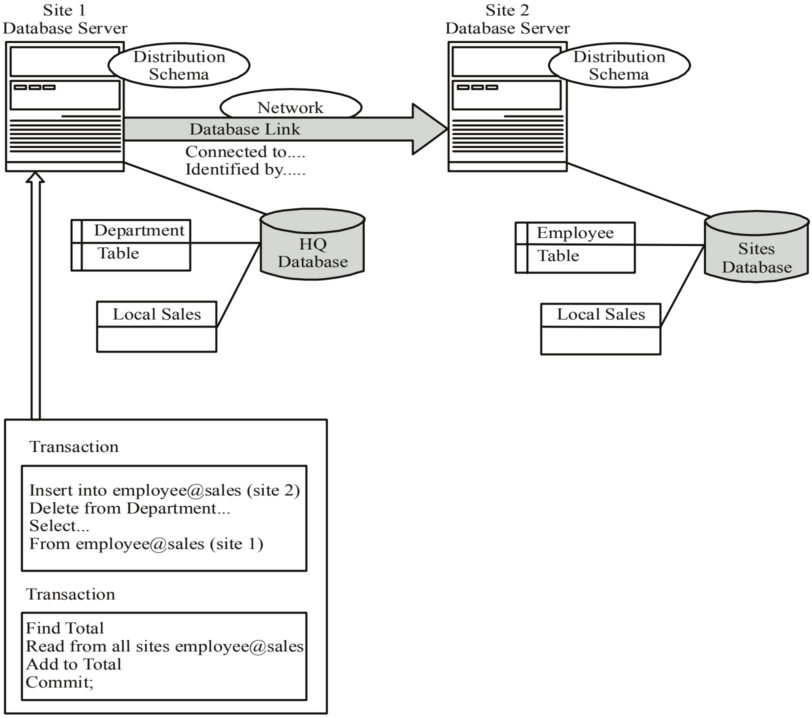
Q3. (a) With the help of a diagram, explain the reference architecture of Distributed DBMS. How is this different from component Architecture of DDBMS?

Ans. Refer to Dec-2006, Q.No.-2(i), Page No.-168 and;

Component architecture of DDBMS: Component Architecture of DDBMS From the viewpoint of commercial implementation a DDBMS would require client server implementation with the backbone of a network. Thus, such an architecture has the following components which are not in reference architecture:

Local DBMS Component (Clients and Servers): At a local database site a DDBMS communicates with a local standard DBMS. This site therefore can have its own client server architecture. A computer on such a site (also referred to as a node) may be a client or server depending on its role at that movement of time.

Data Communications Component: This communication should be established with remote clients may not be connected to a server directly. It is the data communication component that enables direct or indirect communication among various sites. For example, in the Figure 2 the transactions that are shown occurring on Site 1 will have direct connection to Site 1, but they will have an indirect communication to Site 2, to access the local sales data of that site. Thus, a network is necessary for a distributed database system.



Thus, we have clients, servers and the network in the distributed databases.

Global System Catalogue: One of the ways of keeping information about data distribution is through the system catalogue of the DBMSs. In a DDBMS we need to keep a global system catalogue. This catalogue itself might be distributed in implementation. In addition to basic information about the databases it should keep information on data fragmentation and replication schema.

Database Links: A database in a distributed database is distinct from the other databases in the system. Each of these distributed databases may have their own global database name. All these database components would also have certain local names.

Q3. (b) Explain the following two ways to implement the object-oriented concepts in DBMS :

(i) To extend the existing RDBMS to include object orientation.

Ans. The RDBMS technology has been enhanced over the period of last two decades. The RDBMS are based on the theory of relations and thus are developed on the basis of proven mathematical background. Hence, they can be proved to be working correctly. Thus, it may be a good idea to include the concepts of object orientation so that, they are able to support object-oriented technologies too. The first two concepts that were added include the concept of complex types, inheritance, and some newer types such as multisets and arrays. One of the key concerns in object-relational database are the storage of tables that would be needed to represent inherited tables, and representation for the newer types.

One of the ways of representing inherited tables may be to store the inherited primary key attributes along with the locally defined attributes. In such a case, to construct the complete details for the table, you need to take a join between the inherited table and the base class table.

The second possibility here would be, to allow the data to be stored in all the inherited as well as base tables. However, such a case will result in data replication. Also, you may find it difficult at the time of data insertion.

As far as arrays are concerned, since they have a fixed size their implementation is straight forward However, the cases for the multiset would desire to follow the principle of normalisation in order to create a separate table which can be joined with the base table as and when required.

(ii) To create a new DBMS that is exclusively devoted to OODBMS.

Ans. The database system consists of persistent data. To manipulate that data one must either use data manipulation commands or a host language like C using embedded command. However, a persistent language would require a seamless integration of language and persistent data.

Please note: The embedded language requires a lot many steps for the transfer of data from the database to local variables and vice-versa. The question is, can we implement an object oriented language such as C++ and Java to handle persistent data? Well a persistent object-orientation would need to address some of the following issues:

Object persistence: A practical approach for declaring a persistent object would be to design a construct that declares an object as persistent. The difficulty with this approach is that it needs to declare object persistence at the time of creation, An alternative of this approach may be to mark a persistent object during run time. An interesting approach here would be that once an object has been marked persistent then all the objects that are reachable from that object should also be persistent automatically.

Object Identity: All the objects created during the execution of an object oriented program would be given a system generated object identifier, however, these identifiers become useless once the program terminates. With the persistent objects it is necessary that such objects have meaningful object identifiers. Persistent object identifiers may be implemented using the concept of persistent pointers that remain valid even after the end of a program.

Storage and access: The data of each persistent object needs to be stored. One simple approach for this may be to store class member definitions and the implementation of methods as the database schema. The data of each object, however, needs to be stored individually along with the schema. A database of such objects may require the collection of the persistent pointers for all the objects of one database together. Another, more logical way may be to store the objects as collection types such as sets. Some object oriented database technologies also define a special collection as **class extent** that keeps track of the objects of a defined schema.

Q4. (a) What is a (DW) Data Warehouse? Explain the basic components of a DW.

Refer to Ch–6, Q.No.-39, Page No.-125

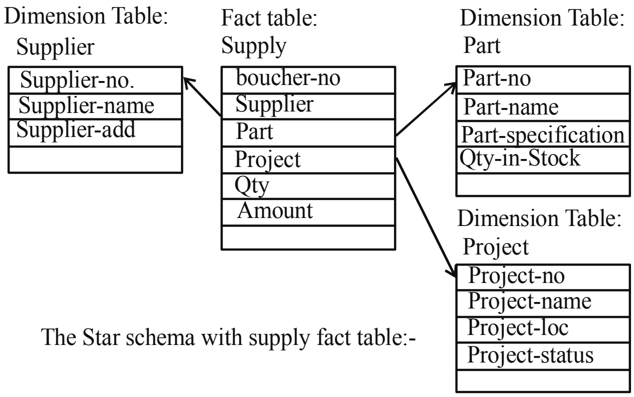
Q4. (b) Consider a Supply Data of an organization having three dimensions as Supplier, Part and Project. Draw a star schema with supply as the fact table. Make suitable assumption.

Ans. A multidimensional storage model contains two types of tables i.e. the dimensional table and the fact table. The dimensional tables have one tuples each for a recorded fact.

Here in our example of three dimension would be

- supplier
- part
- project

and the fact table will be suppliers:-



Dimension Table: Supplier

Supplier-no.
Supplier-name
Supplier-add

Fact table: Supply

boucher-no
Supplier
Part
Project
Qty
Amount

Dimension Table: Part

Part-no
Part-name
Part-specification
Qty-in-Stock

Dimension Table: Project

Project-no
Project-name
Project-loc
Project-status

The Star schema with supply fact table:-

Q4. (c) Explain the following in the context of ORACLE/POSTGRESQL:

(i) Triggers – Refer Q.No.-35, Page No.-92

(ii) Security – Refer Dec–2009, Q.No.-5(c)

(iii) Data Dictionary

Ans. Each oracle database has a data dictionary. An oracle data dictionary is a set of tables and views that are used as a read-only reference on the database. For example the data dictionary stores information about the logical and physical structure of the database. A data dictionary stores information like

- The valid users of an oracle database
- Information about integrity constraints defined for tables in the database
- The amount of space allocated for a schema object and how much of it is in use.

A data dictionary is created when a database is created.

(iv) Indexing

Refer to Page No.-150

Q5. (a) With reference to special Databases and GIS explain the following:

(i) Application of Geographic Databases

Ans. The application of the geographic data base can be categorized into three broad categories. These are:

Cartographic Applications:- Used for analysis of cartographic information in a number of layers. Some of the basic application in this category would be to analyze crop yields, imigration, facility planning, evaluation of land use facility and landscape management, traffic monitoring system etc. These applications need to store data as per the required applications.

3-D Digital Modeling Applications:- Such applications store information about the digital representation of land, and elevations of ports of earth surface at sample points. Then, a surface model is fitted in using the interpolation and visualization techniques-such models are very useful in each science oriented studies.

The third kind of application of such information system is using the geographic objects applications. Such applications are required to store additional information about various regions or objects.

(ii) Requirements of GIS

Ans. Requirement of GIS:- The data in GIS needs to be represented in graphical form, such data would require any of the following formats:

- **Vector Data:-** In such representations, the data is represented using some geometric objects such as line, square, circle etc. E.g. we can represent a road using a sequence of line segments.
- **Raster Data:-** Here data is represented using an attribute value for each pixel or voxel (a three dimensional point) Raster data can be used to represent three dimensional elevation using a format termed as digital elevation format.

For object related applications a GIS may include a temporal structure that records information about some movement related detail such as traffic movement.

A GIS must also support the analysis of data. Some of data analysis operation are (i) analyzing soil erosion (ii) measurement of gradients (iii) computing shortest paths.

(iii) Operations on the data captured in GIS

Ans. Some operations are:- Interpolation for locating elevations at some intermediate points with reference to same points.

- Some operations may be required for data enhancement, smoothing the data interpreting the terrain etc.
- Creating a proximity analysis to determine the distances among the areas of interests.
- Performing image enhancement using image processing algorithms for the raster data.
- Performing analysis related to networks of specific type like road network.

Q5. (b) Consider the following query :
SELECT Empld, EmpName, DeptName, DeptLve
FROM Employee, Department
WHERE Employee. Dept No = Department.
Dept No AND Employee, Salary > 10000
Create a query evaluation plan for the query given above. Make suitable assumption about the relation and statistics.

Ans. Assumption:-Size of the Employee relation = 1000 tuples.
Size of Department relation = 400 tuples
No. of employees having salary >10000 = 100

Step No.	Operation	Cost of query Processing	Result Size
1.	Employee = Department	1000 × 400	1000
2.	Selection operation	1000	100

Q5. (c) Explain Cursors. Explain the role of cursors in Embedded SQL with the help of an example.

Ans. Refer to Dec–2009, Q.No.-2(d)

Example of embedded cursor:

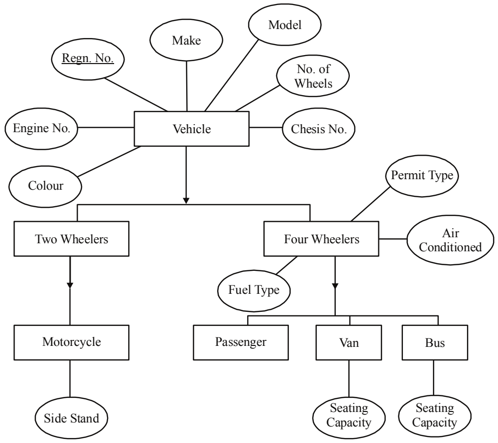
```
Excel SQL Begin Declare Sections;
Char enrolno[10], name[26], P-code[4], qrade
int phone;              int Sql code;
Char SqLSTATE[6]
EXEC SQL END DECLARE SECTION;
/* The connection needs to be established with SqL*/
Printf (“enter the programme code);
Scanf(“%S, & D-code);
EXEC SQL DECLARE CURSOR GUPDATE
Select enrolno, name, Phone, grade from Student where
Progcode = : P-code
For update of grade;
Exec SQL open gupdate;
Exec SQL fetch form Gupdate
INTO: enrlono, : name, : Phone, : grade;
While (SQL CODE = = 0)  {
              Printf (“enter grade for enrolment number, “%S “, enrolno);
              Scanf (“% c “, grade);
              Exec SQL
                     UPDATE STUDENT
                     Set grade = : grade
                     Where current of GUPDATE
                     Exec SQL Fetch from GUPDATE;
                     }
                EXEL SqL Close GUPDATE;
```

MCS – 43: ADVANCED DATABASE DESIGN
December, 2010

***Note:** Question number **1** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) Design a generalization specialization hierarchy for a motor vehicle sales company. The company sells motorcycles, passenger – cars, vans and buses. Justify your placement of attributes at each level of the hierarchy.

Ans.



(1) The attribute Regn_No, Engine_No, ChesisNo, Make, Model, colour and No of wheels are common for all types of vehicles.

(2) The attribute side stand is applicable to two wheelers only whether they have it or not.

(3) The attributes Permit Type, Fuel Type, Air Conditioned are common to all four wheelers.

(4) Seating capacity varies in vans and buses only,

(b) Explain the role of system catalogue in Database administration.

Ans. The database administration is a specialized database activity that is performed by a database administrator. The system catalogue has an important role to play in the database administration. Some of the key areas where the system catalogue helps the database administrator are defined below:

Enforcement of Database Integrity: System catalogue is used to store information on keys, constraints, referential integrity, business rules, triggering events, etc. on various tables and related objects. Thus, integrity enforcement would necessarily require the use of a data dictionary.

Enforcement of Security: The data dictionary also stores information on various users of the database systems and their access rights. Thus, enforcement would necessarily require the use of a data dictionary.

Support for Database System Performance: The data dictionary contains information on the indexes, statistics, etc. Such information is very useful for query optimization. Also such information can be used by the database administrator to suggest changes in the internal schema.

Data dictionary can also support the process of database application development and testing (although this is not a direct relationship to database administration) as they contain the basic documentation while the system are in the process of being developed.

(c) Explain Hash – Join in the context of query processing with the help of an example.

Ans. This physical operator supports the hash join algorithm. hash join may consume more runtime resources, but is valuable when the joining columns do not have useful indexes or when a relatively large number of rows satisfy the join condition, compared to the product of the number of rows in the joined tables.

For example: Hash join assign each tuple of Employee and of Department to a "bucket" by applying a hash function to its WorkDept (DeptNo) value. Within each bucket, look for Employee/Department tuple pairs for which WorkDept = DeptNo.

(d) Explain the role of checkpoints in log-based recovery with the help of an example.

Refer to Ch–5, Q.No.-32, Page No.-89

(e) Consider Employee records with attributes EmpNo, EmpName, Department (comprising of DeptNo, DeptName, Deptloc) and salary (comprising of Basic and HRA). Define this Employee record using object Definition language.

Ans. Class Department

```
               {
                     attribute integer DeptNo;
                      attribute string DeptName;
                       attribute string Deptloc;
                }
Class Salary
               {
```

```
                        attribute real Basic;
                        attribute real HRA;
                          attribute string Deptloc
                   };
Class Employee
                   {
                        attribute integer EmpNo;
                         attribute string EmpName;
                          attribute Department  Dept;
                         attribute Salary sal;
                   }
```

(f) List five differences between Object Oriented and Object Relational Databases.

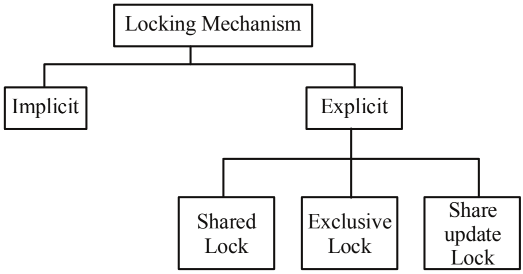
Refer to Dec–2006, Q.No.- 3(i), Page No.-171

(g) Explain the following in the context of ORACLE/POSTGRESQL.

(i) Locking Mechanisms.

Ans. Locking mechanisms

In Oracle the following types of locks are available:



Implicit locks maintained by Oracle itself when a modification is performed on database.

Explicit locks lock mechanism is to used by the user explicitly. The syntax for explicit locking is:

Lock <table name> IN {SHARE | EXCLUSIVE | SHARE UPDATE} MODE [NO WAIT]

This statement is followed by modifications and then either COMMIT or ROLLBACK to unlock the table.

(ii) Oracle Instance.

Refer to Dec–2007, Page No.-209, Point No.-4

(iii) Table spaces.

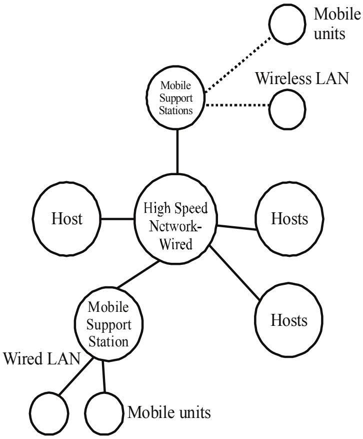
Refer to Ch–8, Q.No.- 4, Page No.-152

(iv)Indexes.

Refer to Ch–7, Page No.-150[Indexes]

Q2. (a) Explain the basic framework for mobile computing and discuss the characteristics of mobile databases. Also mention the challenges of mobile computing.

Ans. Mobile Computing is primarily a distributed architecture. This basic architecture is shown below:



A mobile computing environment consists of:

- host Computers which are fixed,
- mobile support stations,
- mobile client units, and
- networks.

Characteristics of Mobile Databases

The mobile environment has the following characteristics:

(1) ***Communication Latency:*** Communication latency results due to wireless transmission between the sources and the receiver. But why does this latency occur? It is primarily due to the following reasons:

(a) due to data conversion/coding into the wireless formats,

(b) tracking and filtering of data on the receiver, and

(c) the transmission time.

(2) ***Intermittent wireless connectivity:*** Mobile stations are not always connected to the base stations. Sometimes they may be disconnected from the network.

(3) ***Limited battery life:*** The size of the battery and its life is limited. Information communication is a major consumer of the life of the battery.

(4) ***Changing location of the client:*** The wireless client is expected to move from a present mobile support station to an other mobile station where the device has been moved. Thus, in general, the topology of such networks will keep on changing and the place where the data is requested also changes. This would require implementation of dynamic routing protocols.

However, some of the challenges for mobile computing are:

(i) ***Scalability:*** As the number of stations increase, latency increases. Thus, the time for servicing the client also increases. The results in increase in latency, thus more problems are created for data consistency.

The solution: Do data broadcasting from many mobile stations thus, making the most recent information available to all, thus eliminating the enough latency time.

(ii) ***Data Mobile problem:*** Client locations keeps on changing in such networks thus, keeping track of the location of the client is important for, the data server and data should be made available to the client from the server which is minimum latency way from the client.

(b) Explain the role of UML diagrams in database design with the help of examples.

Refer to June–2008, Q.No.- 1(c), Page No.-213

(c) Explain ACID properties in transactions.

Refer to Ch–5, Q.No.- 15, Page No.-77-78

Q3. (a) State 3NF and BCNF. Explain their differences with the help of examples.

Ans. Formal Definition: A relation is in Boyce/Codd normal form (BCNF) if and only if every determinant is a candidate key. [A determinant is any attribute on which some other attribute is (fully) functionally dependent.]

Boyce-Codd normal form is stricter than 3NF, meaning that every relation in BCNF is also in 3NF; however, a relation in 3NF is *not necessarily* in BCNF. A relation schema is an BCNF if whenever a functional dependency X->A holds in the relation, then X is a superkey of the relation. The only difference between BCNF and 3NF is that condition of 3NF (i.e 3NF also require prime attribute), which allows A to be prime if X is not a superkey, is absent from BCNF.

Consider, as an example, the relation Professor:

Professor (Professor code**,** Dept., Head of Dept., Parent time)

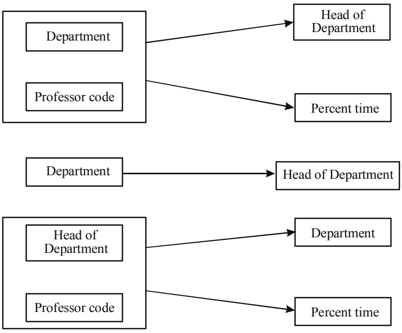
Assuming that

1. The percentage of the time spent in each department is given.

2. A professor can work in more than one department

3. Each department has only one Head of Department

The relationship diagram for the above relation is given below. The two possible composite keys are professor code and Dept. or Professor code and Head of Department. Observe that department as well as Head of Department are not non-key attributes. They are a part of composite key.



Dependency Diagram of Professor Relation

Professor Code	Department	Head of Dept.	Parent
P1	Physics	Dinesh	50
P1	Mathematics	Krishnan	50
P2	Chemistry	Rao	25
P2	Physics	Dinesh	75
P3	Mathematics	Krishnan	100

The relation given in above table is in 3NF. Observe, however, the names of Dept. and Head of Dept. are duplicated. Further, if professor P2 resigns, rows 3 and 4 are deleted. We lose the information that Rao is the Head of Department in Chemistry.

The normalization of the relation is done by creating a new relation for Dept. and Head of Dept. and deleting Head of Dept. from Professor relation. The normalized relations are shown in the following table.

(a)

Professor Code	Department	Percent time
P1	Physics	50
P1	Mathematics	50
P2	Chemistry	25
P2	Physics	75
P3	Mathematics	100

(b)

Department	Head of Dept.
Physics	Dinesh
Mathematics	Krishnan
Chemistry	Rao

And the dependency diagrams for these new relations in figure. The dependency diagram gives the important clue to this normalization step as is clear from the figure below.

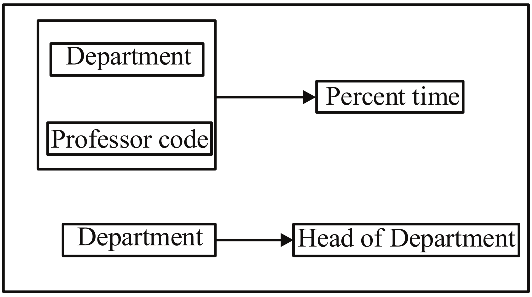


Figure : Dependency diagram of Professor relation

(b) Write SQL statement for the following.

(i) Grant select on Employee table to user Amit.

Ans. Grant select on Employee to Amit;

(ii) Revoke Insert, Update and Delete permission on Department table from user Sumit.

Ans. Revoke Insert, Update, Delete on Department from Sumit.

(c) Define a trigger CHECK_KEY that does not allow duplicate values in attribute EmpNo. in Employee table.

Ans. Create trigger check_key before Insert on Employee for each row
Declare
c integer:
Begin
Select count (Emp. No) into c From Employee where EmpNo = : new. Emp_No;
If c>0 Then raise_application_error (-22000, 'Duplicate Key value');
Endif
end;

(d) Explain the following in the context of data mixing with the help of an example.

(i) Classification.

Refer to Ch–6 Q.No.-55, Page No.-134

(ii) Clustering.

Refer to Ch–6, Q.No.-56, Page No.-134

Q4. (a) Explain the following locking protocols in the context of distributed systems :

(i) Centralized 2PL

(ii) Distributed 2PL

Refer to June–2007, Q.No.2(a)(iii),Page No.-188

(b) Explain the following in the context of XML:

(i) XML Tags.

Ans. A key aspect of dealing with XML documents is to be able to easily access their content. XPath, a W3C recommendation since 1999, provides an easy notation for specifying and selecting parts of an XML document. The JSTL XML tag set, listed in Table , is based on XPath.

Area	**Function**	**Tags**
XML	Core	out parse set
	Flow Control	choose when otherwise forEach if
	Transformation	transform param

The XML tags use XPath as a *local* expression language; XPath expressions are always specified using attribute select. This means that only values specified for select attributes are evaluated using the XPath expression language. All other attributes are evaluated using the rules associated with the global expression language.

(ii) Document Type Declarations (DTD).

Refer to Ch–6, Q.No.-16, Page No.-120

(iii) XML Schema.

Ans. An **XML schema** is a description of a type of XML document, typically expressed in terms of constraints on the structure and content of documents of that type, above and beyond the basic syntactical constraints imposed by XML itself. These constraints are generally expressed using some combination of grammatical rules governing the order of elements, Boolean predicates that the content must satisfy, data types governing the content of elements and attributes, and more specialized rules such as uniqueness and referential integrity constraints. There are languages developed specifically to express XML schemas. The Document Type Definition (DTD) language, which is native to the XML specification, is a schema language that is of relatively limited capability, but that also has other uses in XML aside from the expression of schemas. Two more expressive XML schema languages in widespread use are XML Schema (with a capital *S*) and RELAX NG.

(iv) X Query.

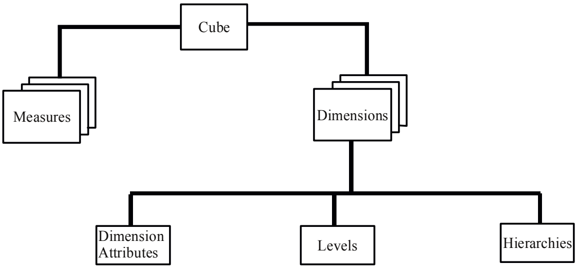
Refer to Ch–6, Q.No.-38, Page No.-125

Q5. Explain the following with the help of an example, if needed:

(a) Multidimensional data modeling for Data warehouse.

Ans. Multidimensional structure is defined as "a variation of the relational model that uses multidimensional structures to organize data and express the relationships between data". The structure is broken into cubes and the cubes are able to store and access data within the confines of each cube. "Each cell within a multidimensional structure contains aggregated data related to elements along each of its dimensions". Even when data is manipulated it remains easy to access and continues to constitute a compact database format. The data still remains interrelated. Multidimensional structure is quite popular for analytical databases that use online analytical processing (OLAP) applications. Analytical databases use these databases because of their ability to deliver answers to complex business queries swiftly. Data can be viewed from different angles, which gives a broader perspective of a problem unlike other models.

Figure : Diagram of the Multidimensional Model



(b) GNOME databases and their applications.

Ans. GNOME-DB is the database application of GNOME Office, the office suite of the GNOME desktop. The project aims to provide a free unified data access architecture to the GNOME project for all Unix platforms. GNOME-DB is useful for any application that accesses persistent data (not only databases, but data), since it now contains a data management API. GNOME-DB is useful for any application that accesses persistent data (not only databases, but data), since it now contains a pretty good data management API. GNOME-DB's production corresponds to the Libgda library which is mainly a database and data abstraction layer, and includes a GTK+ based UI extension, and some graphical tools.

(c) Domain Key Normal Form (DKNF).

Ans. We can also always define stricter forms that take into account additional types of dependencies and constraints. The idea behind domain-key normal form is to specify, (theoretically, at least) the "ultimate normal form" that takes into account all possible dependencies and constraints. A relation is said to be in DKNF if all constraints and dependencies that should hold on the relation can be enforced simply by enforcing the domain constraints and the key constraints specified on the relation.

For a relation in DKNF, it becomes very straightforward to enforce the constraints by simply checking that each attribute value in a tuple is of the appropriate domain and that every key constraint on the relation is enforced. However, it seems unlikely that complex constraints can be included in a DKNF relation; hence, its practical utility is limited.

(d) Log - Based Recovery algorithm.

Refer to Ch–5, Q.No.- 29, Page No.-85

MCS – 43: ADVANCED DATABASE DESIGN
June, 2011

Note: *Question number* ***one*** *is* ***compulsory****. Answer* ***any three*** *questions from the rest.*

Q1. (a) Which MVDs (multivalued dependency) hold for the following table:

P-No.	Colour	Size
P_1	C_1	S_1
P_1	C_2	S_1
P_1	C_1	S_2
P_1	C_2	S_2
P_1	C_1	S_3
P_1	C_2	S_3
P_2	C_3	S_1
P_2	C_3	S_3

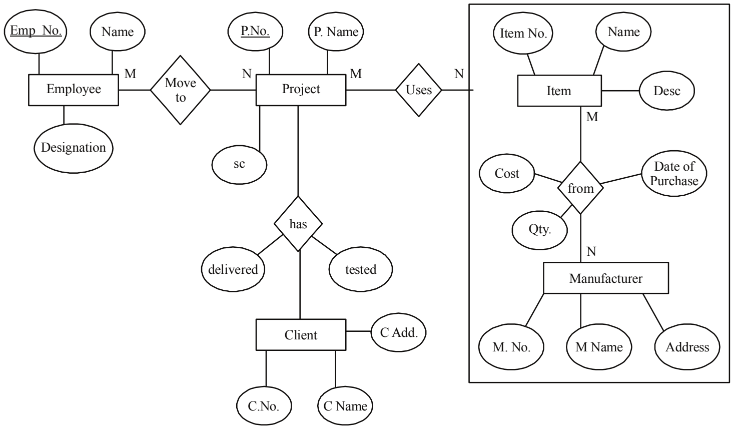
Each product (P) comes in a range of colours (C) and sizes(S)

Ans. P-No-> Colour
P-No-> Size
Colour-> Size
Size->P-No
Size->Colour

(b) The organization called ABC undertakes several kinds of projects. Each employee can move on one or more projects. Each project is undertaken on the request for several projects. Each project has only one client. A project can use a number of items from different manufacturers and an item may be used by several projects. Before delivery of items to a client, it is tested by testing group in the organization.

Draw an E-R diagram and convert it into a relational schema. Also identify primary key in relational schema. Also identify primary key in each relation.

Ans.



Relational Schema
Employee (EmpNo, Ename, Designation)
Project (P.No, PName, PDesc)
Client (C.No, CName, CAddr)
Item (ItemNo, Name, Desc)
Manufacturer (M-No., MName, Address)
Moveto (EmpNo, P.No)
has (P.No, CNo, delivered, tested)
uses (P.No, ItemNo, M_No)

(c) Discuss the shadow paging recovery scheme.
Refer to June–2008, Q.No.- 5(a), Page No.-223

(d) Describe object definition language with the help of an example.
Ans. Object Definition Language (ODL) is the specification language defining the interface to object types conforming to the ODMG Object Model. Often abbreviated by the acronym ODL.
This language's purpose is to define the structure of an Entity-relationship diagram.
Class declarations
Interface < name > { elements = attributes, relationships, methods }
Element Declarations
attributes (< type > : < name >);
relationships (< rangetype > : < name >);
  Type Date Tuple {year, day, month}
  Type year, day, month integer
  Class manager

```
   attributes(id : string unique
               name : string
               phone : string
               set employees : Tuple { [Employee], Start_Date : Date })
 Class Employee
   attributes(id : string unique
               name : string
               Start_Date : Date
               manager : [Manager])
```

(e) How does embedded SQL differ from Dynamic SQL? With the help of an example, describe the implementation of cursors and triggers.
Refer to Ch–4, Q.No.- 10, Page No.-57 & Ch–5, Q.No.-36, Page No.-92

(f) How does Oracle manage database security?
Refer to Ch–8, Q.No.- 24, Page No.-159

(g) When is it useful to have replication or fragmentation of data in distributed system? Explain.
Ans. Examine the nature of distribution. Find out whether an organization needs to have a database at each branch office or in each city or possibly at a regional level. It has direct implication from the viewpoint of fragmentation. For example, in case database is needed at each branch office, the relations may fragment on the basis of branch number.
Analyze the most important transactions in the system and identify where horizontal or vertical fragmentation may be desirable and useful.
Identify the relations that are not to be fragmented. Such relations will be replicated everywhere. From the global ER diagram, remove the relations that are not going to be fragmented.

Q2. (a) Distinguish between the followings with examples.
(i) Time stamping and Two-Phase locking.
Ans. The Two-Phase Locking Protocol
One protocol that ensures serializability is the **two-phase locking protocol.** This protocol requires that each transaction issue lock and unlock requests in two phases:
1. Growing phase. A transaction may obtain locks, but may not release any lock.
2. Shrinking phase. A transaction may release locks, but may not obtain any new locks.
Initially, a transaction is in the growing phase. The transaction acquires locks as needed. Once the transaction releases a lock, it enters the shrinking phase, and it can issue no more lock requests.
Now refer to Dec–2008, Q.No.-1(c), Page No.-227

(ii) Data mining queries and database queries.

Refer to Ch–6, Q.No.-5 & Page No.-132 [Database Processing Vs. Data Mining Processing]

(b) Consider the following relations:
Teacher(T#, T Name)
Practical-Paper (P#, P-Name,Tname)
Conducts (T#, P#)
Write the relational algebra expression for the following queries.
(i) Get those teacher numbers (T#) who are not conducting practical number $P_{2.}$

Ans. $\pi_{T\#}(\sigma_{P\#<>2}(\text{Teacher} \bowtie \text{ Conducts} \bowtie \text{Pr actical Paper}))$

(ii) Get details of those teachers who are conducting practical numbers P_1 to P_4.

Ans. $\pi_{\text{Teacher}\#}(\text{Teacher} \bowtie (\text{Conducts} \div (\pi_{P\#}(\sigma_{P\#>='P1' \wedge P\#<='P4'}(\text{CONDUCTS})))))$

(c) What problems occur in the databases system when transactions do not satisfy ACID properties? Explain explicitly using suitable examples.

Ans. Atomicity failure

The transaction subtracts 10 from A and adds 10 to B. If it succeeds, it would be valid, because the data continues to satisfy the constraint. However, assume that after removing 10 from A, the transaction is unable to modify B. If the database retains A's new value, atomicity and the constraint would both be violated. Atomicity requires that both parts of this transaction complete or neither.

Consistency failure

Consistency is a very general term that demands the data meets all validation rules. In the previous example, the validation is a requirement that A + B = 100. Also, it may be implied that both A and B must be integers. A valid range for A and B may also be implied. All validation rules must be checked to ensure consistency.

Assume that a transaction attempts to subtract 10 from A without altering B. Because consistency is checked after each transaction, it is known that A + B = 100 before the transaction begins. If the transaction removes 10 from A successfully, atomicity will be achieved. However, a validation check will show that A + B = 90. That is not consistent according to the rules of the database. The entire transaction must be undone.

Isolation failure

To demonstrate isolation, we assume two transactions execute at the same time, each attempting to modify the same data. One of the two must wait until the other completes in order to maintain isolation.

Consider two transactions. T_1 transfers 10 from A to B. T_2 transfers 10 from B to A. Combined, there are four actions:

- subtract 10 from A
- add 10 to B.
- subtract 10 from B
- add 10 to A.

If these operations are performed in order, isolation is maintained, although T_2 must wait. Consider what happens, if T_1 fails half-way through. The database eliminates T_1's effects, and T_2 sees only valid data.

By interleaving the transactions, the actual order of actions might be: *A* “ 10, *B* “ 10, *B* + 10, *A* + 10. Again consider what happens, if T_1 fails. T_1 still subtracts 10 from A. Now, T_2 adds 10 to A restoring it to its initial value. Now T_1 fails. What should A's value be? T_2 has already changed it. Also, T_1 never changed B. T_2 subtracts 10 from it. If T_2 is allowed to complete, B's value will be 10 too low, and A's value will be unchanged, leaving an invalid database. This is known as a write-write failure, because two transactions attempted to write to the same data field.

Durability failure

Assume that a transaction transfers 10 from A to B. It removes 10 from A. It then adds 10 to B. At this point, a “success” message is sent to the user. However, the changes are still queued in the disk buffer waiting to be committed to the disk. Power fails and the changes are lost. The user assumes that the changes have been made.

Q3. (a) What is ODBC? What are requirements of ODBC? Describe the components required for implementation of ODBC.

Refer to Q.No.-5(iii), Page No.-179 & Ch–7, Q.No.- 15 &17, Page No.-145

(b) What are the different types of index in PostgreSQL? Explain each one of them.

Refer to Page No.-150[Indexes]

(c) What is the difference between document type definition and XML schema? Explain with an example.

Refer to Dec–2009, Q.No.- 1(c), Page No.-244

Q4. (a) What do you understand by query optimization? What are query trees? Explain with an example.

Ans. Query tree is an internal representation of an SQL statement where the single parts that built it are stored separately.

Consider the following query on COMPANY database: "Find the name of employee born after 1967 who work on a project named 'Greenlife' ".

The SQL query is:

SELECT Name
FROM EMPLOYEE E, JOIN J, PROJECT P
WHERE E.EID = J.EID and PCode = Code and Bdate > '31-12-1967' and P.Name = 'Greenlife';

The query tree for this SQL query is

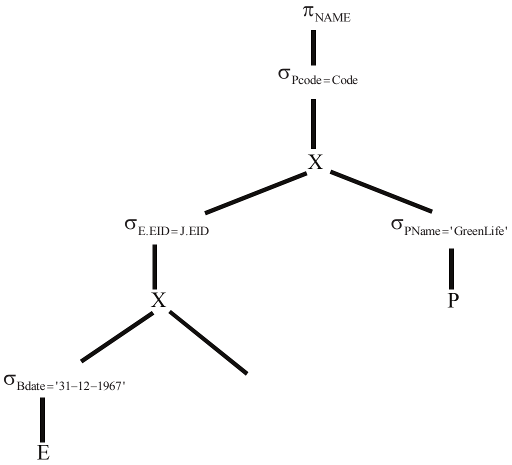


Figure : query tree for query in example

(b) What is multiversion concurrency control? Explain how multiversion concurrency control can be achieved based on time stamp ordering?

Ans. Refer to June 2007, Q.No.-5(a), Page No.-196

(c) List steps involved in building of Dataware house.

Ans. The data warehouse technologies use very diverse vocabulary. Although the vocabulary of data warehouse may vary for different organizations, the data warehousing industry is in agreement with the fact that the data warehouse lifecycle model fundamentally can be defined as the model consisting of five major phase – design, prototype, deploy, operation and enhancement.

(1) Design: The design of database is to be done for available data inventories, DSS analyst requirements and analytical needs. It needs to produce a robust star schema or snowflake schema. Key activities in the design phase may include communication with the end users, finding the available catalogues, defining key performance and quality indicators, mapping of decision-making processes as per the information needs at various end user levels, logical and physical schema design, etc.

(2) Prototype: A data warehouse is a high cost project, thus, it may be a good idea to deploy it partially for a select group of decision-makers and database practitioners in the end user communities. This will help in developing a system that will be easy to accept and will be mostly as per the user's requirements.

(3) Deploy: Once the prototype is approved, then the data warehouse can be put to actual use. A deployed data warehouse comes with the following processes;

(4) Operation: Once deployed the data warehouse is to be used for day-to-day operations. The operation in data warehouse includes extracting data, putting it in database and output of information by DSS.

(5) Enhancement: These are needed with updating of technology, operating processes, schema improvements, etc. to accommodate the change.

Q5. (a) What are views? How are they implemented? Can views be used for data manipulation? Explain with the help of an example.

Refer to Ch–4, Q.No.-1-3, Page No.-48

(b) Describe normalization using join dependency with the help of an example.

Refer to Dec–2008, Q.No.- 4(b), Page No.-232

(c) Explain the following terms in the context of DBMS:

(i)Multilevel Security

Refer to Ch–5, Q.No.- 61, Page No.-104

(ii) Auditing and Control

Ans. The auditing process is relevant in the database environment to verify that the automated operations are properly implemented and executed. Since data entry and maintenance is done online, the history of the evolution of a piece of data is no longer available in the database, which will contain the latest value only.

In addition to the above, functions that were previously separated and controlled by distinct parts of an organization (for, example initiation of a transaction and recording the financial and operational part of the transaction) are integrated into the database environment. This integration of operations, while reducing redundancy, causes the loss of independent scrutiny and corrections. Furthermore, in an online system where a number of transactions are executed concurrently, it is very difficult to reproduce the same sequence of processing. All these factors point to need for an audit trail in the database environment.

The audit trail to be maintained for audit purposes has some similarity with the data collected for recovery operations. Thus for each update operation, the before and after image of the data objects undergoing modification are recorded. All logons, read operations, and suspect or illegal operations are recorded. This information can be used to analyze the practice of the users of the database, detect any attempted violations, help correct errors in design or execution, and improve the control procedure.

The control of the database starts with the design of the database and the application programs that will be using the data. One of the first control principles is that of separating the responsibilities. This can be practiced by assigning separate teams for the design and implementation of the program and for its validation and installation.

The integrity control mechanisms should be integrated in the database and the data entry function should be validated by the application programs.

(iii) Redo log files

Ans. Every Oracle database has a set of two or more redo log files. The set of redo log files is collectively known as the redo log for the database. A redo log is made up of redo entries (also called redo records)

The primary function of the redo log is to record all changed made to the data. If a failure prevents modified data from being permanently written to the data files, then the changes can be obtained from the redo log, so work is never lost. To protect against failure involving the redo log itself, Oracle allows a multiplexed redo log so that two or more copies of the redo log can be maintained on different disks.

The information in a redo log file is used only to recover the database from a system or media failure that prevents database data from being written to the data files. For example, if an unexpected power shortage terminates database operation, then the data in the memory cannot be written to the data files, and the data is lost. However, lost data can be recovered when the database is opened, after power is restored. By applying the information in the most recent redo log files to the database data files, Oracle restores the database to the time when the power failure had occurred. The process of applying the redo log during a recovery operation is called rolling forward.

(iv) Characteristics of DBMS

Ans. (1) Reduction or Redundancies : In database approach data can be stored at a single place or with controlled redundancy under DBMS, which saves space and does not permit inconsistency.

(2) Shared Data : A DBMS allows the sharing of database under its control by any number of application programs or users. A database belongs to the entire organisation and is shared by all authorized user.

(3) Data Independence : Database Management systems separates data descriptions from data. Hence it is not affected by changes. This is called Data Independence, where details of data are not exposed. DBMS provides an abstract view and hides details. For example, logically we can say that the interface or window to data provided by DBMS to a user may still be the same although the internal structure of the data may be changed.

(4) Improved Integrity : Data Integrity refers to validity and consistency of data. Data Integrity means that the data should be accurate and consistent. This is done by providing some checks or constraints. These are consistency rules that the database is not permitted to violate. Constraints may apply to data items within a record or relationships between records.

(5) Efficient Data Access : DBMS utilises techniques to store and retrieve the data efficiently at least for unforeseen queries. A complex DBMS should be able to provide services to end users, where they can efficiently retrieve the data almost immediately.

(6) Multiple User Interfaces : Since many users having varying levels of technical knowledge use a database, a DBMS should be able to provide a variety of interfaces. This includes–

(a) query language for casual users,

(b) programming language interfaces for application programmers,

(c) forms and codes for parametric users,

(d) menu driven interfaces, and

(e) natural language interfaces for standalone users, these interfaces are still not available in standard form with commercial database.

(7) Representing complex relationship among data : A database may include varieties or data interrelated to each other in many ways. A DBMS must have the capability to represent a variety of relationships among the data as well as to retrieve and update related data easily and efficiently.

MCS – 43: ADVANCED DATABASE DESIGN
December, 2011

***Note:** Question number **one** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) Identify the functional dependencies which hold for the following table:

Emp No.	Name	Address	Dept	Dept. Manager
E_1	N_1	A_1	D_1	E_1
E_2	N_2	A_2	D_1	E_1
E_3	N_1	A_2	D_1	E_1
E_4	N_3	A_3	D_2	E_9
E_9	N_4	A_4	D_2	E_9

Ans. Partial Dependency
Emp no-> Name
Emp no->Dept
Transitive Dependency
Empno->name and name->address therefore, empno->address
Multivalued dependency
Emp no->> Dept

(b) Construct an E-R diagram for a training institute which imparts soft skills. The institute maintains records about instructors, students, classes, assignments, results (Theory as well as practicals) class timings for each student. The number of subjects, in which the candidate is enrolled and past performances in different subjects, is recorded. Document all assumptions that you make about the mapping constraints.

Ans. Assumptions:

(1) Instructor instructs a class. One instructor must instruct at least one class.

(2) Instructor's name, age, id, gender and address are recorded, id being the primary key.

(3) The instructor must have at least one student to teach.

(4) Student's ID, name, age and gender are also recorded.

(5) Instructor and students work in a class. Both instructor and students must be enrolled in at least one class.

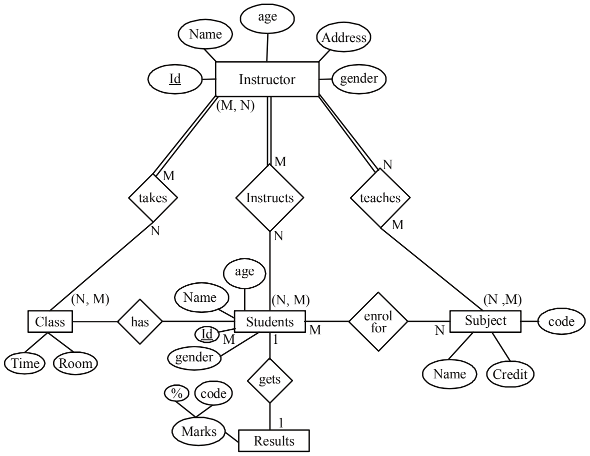
(6) Instructor teaches subjects to students.

(7) An instructor can teach many subjects.

(8) A student can enroll for many subjects.

(9) Subjects include name, credit and subject code with them.

(10) A student gets result on the basis of their marks. Marks can be represented as percentage or grades.



(c) How is the check pointing information used in the recovery operation following a system crash in DBMS? Explain.

Ans. Checkpoint-Recovery is a common technique for incorporating a program or system with fault tolerant qualities, and grew from the ideas used in systems which employ transaction processing. It allows systems to recover after some fault interrupts the system, and causes the task to fail, or be aborted in some way. While many systems employ the technique to minimize lost processing time, it can be used more broadly to tolerate and recover from faults in a critical application or task. The basic idea behind checkpoint-recover is the saving and restoration of system state. By saving the current state of the system periodically or before critical code sections, it provides the baseline information needed for the restoration of lost state in the event of a system failure. While the cost of checkpoint-recovery can be high, by using techniques like memory exclusion, and by designing a system to have as small a critical state as possible may minimize the cost of checkpointing enough to be useful in even cost sensitive embedded applications.

When a system is checkpointed, the state of the entire system is saved to non-volatile storage. The checkpointing mechanism takes a snapshot of the system state and stores the data on some non-volatile storage medium. Clearly, the cost of a checkpoint will vary with the amount of state required to be saved and the bandwidth available to the storage mechanism being used to save the state.

(d) Explain the concept of inheritance in object oriented database system, with the help of an example.

Ans. Inheritance is a way to reuse code of existing objects, or to establish a subtype from an existing object, or both, depending upon programming language support. In *classical inheritance* where objects are defined by classes, classes can inherit attributes and behavior from pre-existing classes called base classes, *superclasses*, *parent classes* or *ancestor classes*. The resulting classes are known as *derived classes*, *subclasses* or *child classes*.

Example:

Vehicle

Char* number_plate;
Char* model;
date data_last_overhual;
date next_overhual;

Truck

int license;
float price;

Bus
int seats;

(e) What are assertions? Explain with an example.

Refer to Ch-4, Q.No.-1, Page No.-48

(f) How can you protect your database from statistical query attacks? Explain.

Ans. One way to protect your database is to not to construct SQL queries as Strings within application code. If SQL queries are constructed this way, then the application is taking text directly from a web control on the web page and generating a SQL statement from it. Good hackers can enter SQL queries within the web control that will instruct the SQL to perform some task other than what was originally expected.

(g) Explain Clustering in data mining.

Refer to Ch-6, Q.No.-56, Page No.-134

Q2. (a) Distinguish between the followings with appropriate examples.

(i) Centralized two phase locking and Distributed two phase locking.

Refer to June-2007, Q.No.-2(a)(iii), Page No.-188

(ii) XML and HTML

Refer to June-2007, Q.No.-2(a)(ii), Page No.-188

(b) Consider the following database employee (emp_name, street, city), working (emp_name, factory, name_salary) factory (factory_name, city) Manager (emp_name, manager_name) Write the relational algebra expressions for the following queries.

(i) Find the names, streets and cities of all factory employees who work for factories F_1 and F_5 and earn more than 25000.

Refer to Ch-5, Q.No.-11(iv), Page No.-73

(ii) Find all the factory employees who live in the same city as the factory where they are working.

Refer to Ch-5, Q.No.-11(ii), Page No.-73

(c) With the help of an example explain insertion and deletion of aromatics.

Ans. Insertion Anomaly

The insertion anomaly occurs when a new record is inserted in the relation. In this anomaly, the user cannot insert a fact about an entity until he has an additional fact about another entity.

Example: A table which stores records for a company's sales people & the clients for whom they are responsible. Because client is a required field - if a newly hired sales representative must complete several weeks of training before being allowed to call on clients, it is not possible to record him in the table during training. Or, if we do add new hires, we must create "dummy" clients as placeholders.

Deletion Anomaly: The deletion anomaly occurs when a record is deleted from the relation. In this anomaly, the deletion of facts about an entity automatically deleted the fact of another entity.

Modification Anomaly: The modification anomaly occurs when the record is updated in the relation. In this anomaly, the modification in the value of specific attribute requires modification in all records in which that value occurs.

Example: Let's say 'Gully' takes on a temporary research assignment that requires her to give up her existing clients for next 6 months. Because we can not delete just the client value, we are faced with the choice of deleting her record completely from the sales Rep. table.

Q3. (a) Describe the reference architecture of a distributed DBMS with the help of a block diagram.

Refer to Dec-2006, Q.No.-2(i), Page No.-168

(b) How does postgre SQL perform storage and indexing of tables? Briefly discuss the type of indexes involved in postgre SQL.

Refer to Ch-7, Q.No.-27, Page No.-148

(c) What is semi structured data? Explain with an example.

Refer to Ch-6, Q.No.-17, Page No.-120

Example:

- name: Peter Wood
- email: ptw@dcs.bbk.ac.uk, p.wood@bbk.ac.uk
- name:
  - o first name: Mark
  - o last name: Levene
- email: mark@dcs.bbk.ac.uk

Q4. (a) Define Hash join and explain the process and cost calculation of Hash join with the help of an example.

Refer to June-2008, Q.No.-1(d), Page No.-213

(b) Describe two phase commit protocol in distributed databases.

Refer to Ch-5, Q.No.-72, Page No.-111

(c) List the features of semantic database.

Refer to June-2008, Q.No.-2(c), Page No.-219

Q5. (a) Discuss the 5th normal form and domain key normal form with a suitable example in each.

Refer to Ch-1, Q.No.-10, Page No.-7 & Dec-2010, Q.No.-5(c), Page No.-276

(b) What do you mean by deadlock in DBMS? How can you detect a deadlock? Suggest a technique that can be used to prevent it.

Refer to Ch-5, Q.No.-24, Page No.-83 & Q.No.-21 , Page No.-79

(c) What are challenges in designing multimedia databases? Discuss.

Refer to Ch-7, Q.No.-4, Page No.-143

MCS – 43: ADVANCED DATABASE DESIGN
June, 2012

Note: *Question number **one** is compulsory. Answer **any three** questions from the rest.*

Q1. (a) Is the following XML document well formed? Justify your answer:

```
<?xml version = “1.0” standalone = “yes” ?>
<employee>
        <name>Amita</name>
        <department>English</department>
</employee>
<employee>
        <name>Sumita</name>
        <department>Hindi</department>
</employee>
```

Refer to June-2010, Q.No.-1(e), Page No.-260

(b) Determine all 4NF violations for the relation schema R (X, Y, Z, W) with multivalued dependencies X $\twoheadrightarrow$ Y and X $\twoheadrightarrow$ Z. Decompose the relation in to 4NF.

Ans. Fourth normal form (4NF) is a normal form used in database normalization. Introduced by Ronald Fagin in 1977, 4NF is the next level of normalization after Boyce–Codd normal form (BCNF). Whereas the second, third, and Boyce–Codd normal forms are concerned with functional dependencies, 4NF is concerned with a more general type of dependency known as a multivalued dependency. A Table is in 4NF if and only if, for every one of its non-trivial multivalued dependencies $X \twoheadrightarrow Y$, X is a superkey—that is, X is either a candidate key or a superset.

A multivalued dependency $X \twoheadrightarrow Y$ signifies that if we choose any x actually occurring in the table (call this choice x_c), and compile a list of all the $x_c yz$ combinations that occur in the table, we will find that x_c is associated with the same y entries regardless of z.

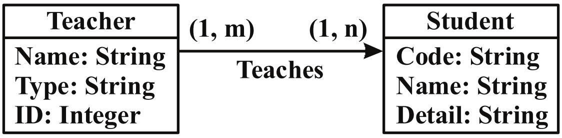
(c) What are triggers? Explain the significance of triggers with the help of an example in SQL.

Refer to Ch-5, Q.No.-35, Page No.-92

(d) What is Data Warehousing? Discuss various characteristics of Data Warehousing.

Refer to Dec-2008, Q.No.-1(b), Page No.-227 & Ch-6, Q.No.-41, Page No.-129

(e) Create an object oriented database (using ODL) for the following class diagram,



Ans. Teacher

Name	Type	ID
Sita	English	T01
Mukesh	Maths	T02

Code	Name	Detail
C01	Hari	I year
C05	Megha	IIIyear

(f) Explain with the help of an example, the log based recovery scheme, using deferred database modification approach.

Refer to Ch-5, Q.No.-29, Page No.-85

(g) Explain the role of query optimizer in oracle.

Refer to Ch-5, Q.No.-1, Page No.-68

(h) Why do you need 3 Phase Commit (3PC) Protocol in case of distributed Databases? Explain the 3PC protocol with the help of a diagram.

Refer to Ch-5, Q.No.-73, Page No.-111

Q2. (a) What is multilevel security? What are typical security levels?

Refer to Ch-5, Q.No.-61, Page No.-104 & Q.No.-63 , Page No.-106

(b) How database Queries differ from data mining Queries? Explain the K – means clustering in data mining algorithm with the help of an example.

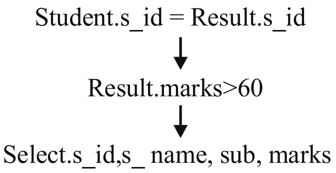
Refer to Page No.-132 [Database Processing Vs. Data Mining Processing] & Ch-6, Q.No.-56, Page No.-134

(c) Consider the following Query

Select student_id, student_name, subject, marks
From STUDENT, RESULT
Where STUDENT. student_id = RESULT.
Student_id
And RESULT. marks > 60

Create a Query evaluation plan for the above Query, assume suitable relations and statistics.

Ans. Query Evaluation plan:



Q3. (a) What are multimedia databases? Discuss the challenges in designing the multimedia databases.

Refer to Ch-7, Q.No.-1, Page No.-140 & Q.No.-4, Page No.-142

(b) "2 Phase locking protocol uses waiting, where as time stamping method uses Roll back of Transaction, to avoid non serializable execution". By considering the same transaction schedule, compare the above two execution strategies for concurrent transactions.

Refer to Dec-2009, Q.No.-1(b), Page No.-243 & June-2011, Q.No.-2(a)(i), Page No.-279

(c) Consider the following relations:
Student (id, name, age, programme)
(fee-paid (id, date, amount)

(i) Create a view using SQL for a student whose id is "001". The student is allowed to see information about himself/herself.

Ans.
```
CREATE VIEW Stud AS
     SELECT * FROM Student
     WHERE id = 001
```

(ii) The student has only read-access to his/her data. Write the appropriate SQL code for the above.

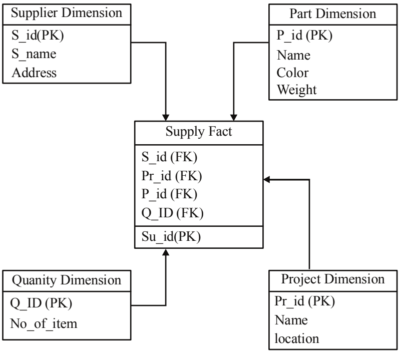
Ans.
```
Select * from Stud
```

(iii) Create another view for Administrator who can access and modify all the data.

Ans.
```
CREATE VIEW Admin AS
     SELECT * FROM Student, fee-paid
     WHERE student.id = feepaid.id
```

Q4. (a) Consider a supply data of an organization having three dimensions as SUPPLIER, PART and PROJECT where a supplier "s" supplies part "p" to project "r" in quantity "q". Draw a star schema with SUPPLY as fact table. Make suitable assumptions. How does star schema differs from snow flake schema?

Ans.



	Snowflake Schema	Star Schema
Ease of maintenance/change:	No redundancy and hence more easy to maintain and change	Has redundant data and hence less easy to maintain/change
Ease of Use:	More complex queries and hence less easy to understand	Less complex queries and easy to understand
Query Performance:	More foreign keys-and hence more query execution time	More foreign keys-and hence more query execution time
Normalization:	Has normalized tables	Has De-normalized tables
Type of Datawarehouse:	Good to use for small data warehouses / data marts	Good for large data warehouses
Joins:	Higher number of Joins	Fewer Joins

(b)With the help of a diagram, explain the reference architecture of Distributed DBMS.

Refer to Dec-2006, Q.No.-2(i), Page No.-168

(c) List the characteristics and challenges in implementation of mobile databases.

Refer to Dec-2010, Q.No.-2(a), Page No.-270

Q5. (a) What are the different types of index implementations available in POST gre SQL? Explain each one of them.

Refer to Page No.-150[Indexes]

(b) Explain the following:

(i) Dynamic SQL

Refer to Ch-4, Q.No.-10, Page No.-57

(ii) OLAP and its types

Ans. Refer to Page No.-179[OLAP]

The different types of OLAP are defined based on the underlying design on which the OLAP model is developed. When the OLAP database resides on a relational database then that is called ROLAP, Relational OLAP. When the OLAP data is stored as independent proprietary format by creating multi-dimensional cubes, they are called MOLAP that is Multidimensional OLAP. When an OLAP design uses the hybrid of OLAP and MOLAP, it is called HOLAP, Hybrid OLAP. HOLAP uses both the technologies to analyze the data where users can go from MOLAP to ROLAP by drilling through the data.

(iii) Spatial databases

Refer to Page No.-192[Spatial Databases]

(iv) Semantic databases

Refer to June-2008, Q.No.-2(c), Page No.-219

(c) What is the significance of creating Data Dictionary in DBMS? Explain the statistics stored in the Data Dictionary, with the help of an example.

Refer to Ch-4, Q.No.-20, Page No.-65

To give people the experience of happiness and comfort with your vibrations of peace and happiness is true service.

MCS – 43: ADVANCED DATABASE DESIGN
December, 2012

***Note:** Question number **one** is compulsory. Answer **any three** questions from the rest.*

Q1. (a) Given the following semi structured data in XML, create the DTD (Document Type Declaration) for it:

```
<DOCUMENT>
<STUDENT>
        <NAME> RAJESH </NAME>
        <CLASS> X-A        </CLASS>
        <SCHOOL> K.V; R.K.PURAM,
DELHI</SCHOOL>
</STUDENT>
<STUDENT>
        <NAME> SANJAY</NAME>
        <CLASS> XI-C </CLASS>
        <SCHOOL) GURUKUL, GHAZIBAD
        </SCHOOL>
<STUDENT>
</DOCUMENT>
```

Ans. < ? xml version = “1.0” encoding = “UTF-8” stand alone = “yes” ?>

```
<! DOCUMENT document [
<! ELEMENT document (student) * >
<! ELEMENT student (name, class, school)>
<! ELEMENT name (# PCDATA) >
<! ELEMENT class (# PCDATA) >
<! ELEMENT school (# PCDATA)>]>
```

(b) How BCNF is different from 3NF. Explain with the help of an example. Consider a relation R(A, B, C) with functional dependencies AB → C and C → A. Decompose the relation ‘R’ into BCNF relations.

Refer to Dec-2010, Q.No.-3(a), Page No.-271 & June-2010, Q.No-2(c), Page No.-261

(c) What are views and what is their significance? How views are managed in SQL, explain using an example?

Refer to Ch-4, Page No.-49[View] & Q.No.-3, Page No.-50

(d) What is data mart? How is it different from data warehouse? How the management of data in a database, is different from the management of data in a data warehouse. Explain using example.

Ans. Refer to Dec-08, Q.No.- 3(b) Page No.-231, & Ch-6, Q.No.-49 , Page No.-131

As one of the oldest components associated with computers, the database management system (DBMS), is a computer software program that is designed as the means of managing all databases that are currently installed on a system hard drive or network. Different types of database management systems exist, with some of them designed for the oversight and proper control of databases that are configured for specific purposes. Here are some examples of the various incarnations of DBMS technology that are currently in use, and some of the basic elements that are parts of DBMS software applications.

Data is the most valuable enterprise asset. A good integrated data management strategy will enhance an organisation's ability to develop valuable insights that provides greater business value. This strategy is driven by the following needs:

- Need Integrated Data: Disparate data sources lead to information silos resulting in decision deficiencies
- Need Accurate Data: Lack of data standards lead to data quality issues and therefore distorted insights
- Need Data Governance: Inadequate definitions, unclear ownership and lack of standards can lead to inconsistencies in organisational data management.

(e) Consider the following three transactions

T_1	**T_2**	**T_3**
Read (x)	**Read (x)**	**Read (x)**
x=x–1000	**display (x)**	**Y : = (x)**
Write (x)		**display (x)**

Insert shared and exclusive locks in T_1, T_2 and T_3 such that the transactions when executed concurrently, do not encounter and concurrency related problem.

Refer to June-2009, Q.No.-1(a), Page No.-234

(f) What is Datagrid? Show typical structure of datagrid. What are the application areas of datagrid?

Ans. Refer to Ch-7, Q.No.-20, Page No.-146

Applications:

A data grid includes most of relational database capabilities including schema integration formal conversion of data, distributed query support, etc.

(g) What are Web-databases? How do you create them?

Ans. The term web database can be used in at least two different way:

Definition 1: A web database may be defined as the organised listing of web pages for a particular topic. Since the number of web pages may be large for a topic, a web database would require strong indexing.

Definition 2: A web database is a database that can be accessed through the web.

Now refer to Ch-7, Q.No.-12, Page No.-145

(h) Explain shadow paging recovery scheme with the help of diagram.

Refer to June-08, Q.No.-5(a), Page No.-223

Q2. (a) Illustrate using an example the Apriori algorithm for association rule mining.

Refer to Ch-6, Q.No.-60 & 61, Page No.-138-139

(b) Discuss, how Oracle manages database security?

Refer to Ch-8, Q.No.-24, Page No.-159

(c) Consider the following SQL Query on the employee relation.

SELECT LNAME, FNAME
FROM EMPLOYEE
WHERE SALARY > (SELECT MAX (SALARY) FROM EMPLOYEE WHERE DNO = 5)

Derive an execution plan for the Query. Give a measure of Query Cost. Make suitable assumptions.

Ans.

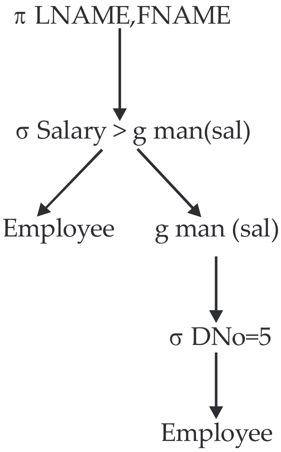


Figure: Query Execution Plan

Measurement of Query Cost:

Assumptions

(1) Employee table is not sorted.
(2) Employee table has N blocks
(3) Average seek time = t_s
(4) Average block read cost = b_r

Query Cost $= \underset{\text{for main query}}{(N \times t_s \times b_r)} + \underset{\text{for subquery}}{(N \times t_s \times b_r)}$

Q3. (a) What are mobile databases? Discuss the characteristics of mobile databases. Give an application of mobile database.

Ans. Refer to Ch-7, QNo.-10 & 14, Page No.-144-145

Applications of mobile databases:

(1) Electronic vallet (e-vallet)

(2) Mobile cash (m-cash)

(b) Explain how the 3 phase commit protocol increases the systems availability and doesn't allow transaction to remain blocked until a failure is repaired, with a suitable example.

Refer to Dec-2009, Q.No.-2(c), Page No.-248

(c) What is meant by the term "ETL"? Discuss the transformations required during the ETL process.

Ans. ETL refers to the methods involved in accessing and manipulating data available in various sources and loading it into a target data warehouse. Initially the ETL was performed using SQL programs, however, now there are tools available for ETL processes.

Transformations are useful for transforming the source data according to the requirements of the data warehouse. The process of transformation should ensure the quality of the data that needs to be loaded into the target data warehouse. Some of the common transformations are:

Filter Transformation: Filter transformations are used to filter the rows in a mapping that do not meet specific conditions. For example, the list of employees of the Sales department who made sales above ₹50,000/- may be filtered out.

Joiner Transformation: This transformation is used to join the data of one or more different tables that may be stored on two different locations and could belong to two different sources of data that may be relational or from any other sources like XML data.

Aggregator Transformation: Such transformations perform aggregate calculations on the extracted data. Some such calculations may be to find the sum or average.

Sorting Transformation: requires creating an order in the required data, based on the application requirements of the data warehouse.

Q4. (a) What is "tuning a database"? What are the goals of tuning in relational database system? Why is Query tuning required?

Ans. Database tuning describes a group of activities used to optimise and homogenise the performance of a database. It usually overlaps with query tuning, but refers to design of the database files, selection of the database management system (DBMS), operating system and CPU the DBMS runs on. The goal is to maximise use of system resources to perform work as efficiently and rapidly as possible. Most systems are designed to manage work efficiently, but it is possible to greatly improve performance by customising settings and the configuration for the database and the DBMS being tuned.

Sql Statements are used to retrieve data from the database. We can get same results by writing different sql queries. But use of the best query is important when performance is considered. So you need to sql query tuning based on the requirement. Here is the list of queries which we use regularly and how these sql queries can be optimised for better performance.

SQL Tuning/SQL Optimisation Techniques:

(1) The sql query becomes faster if you use the actual columns names in SELECT statement instead of than '*'.

(2) HAVING clause is used to filter the rows after all the rows are selected. It is just like a filter. Do not use HAVING clause for any other purposes.

(3) Sometimes you may have more than one subqueries in your main query. Try to minimise the number of subquery block in your query.

(4) Use operator EXISTS, IN and table joins appropriately in your query.

(a) Usually IN has the slowest performance.

(b) IN is efficient when most of the filter criteria is in the sub-query.

(c) EXISTS is efficient when most of the filter criteria is in the main query.

(5) Use EXISTS instead of DISTINCT when using joins which involves tables having one-to-many relationship.

(6) Try to use UNION ALL in place of UNION.

(7) Be careful while using conditions in WHERE clause.

(8) Use DECODE to avoid the scanning of same rows or joining the same table repetitively. DECODE can also be made used in place of GROUP BY or ORDER BY clause.

(9) To store large binary objects, first place them in the file system and add the file path in the database.

(10) To write queries which provide efficient performance follow the general SQL standard rules.

(a) Use single case for all SQL verbs

(b) Begin all SQL verbs on a new line

(c) Separate all words with a single space

(d) Right or left aligning verbs within the initial SQL verb

(b) Explain the following two ways to implement the object oriented concepts in DBMS:

(i) To extend the existing RDBMS to include object orientation

Ans. The RDBMS technology has been enhanced over the period of last two decades. The RDBMS are based on the theory of relations and thus are developed on the basis of proven mathematical background. Hence, they can be proved to be working correctly. Thus, it may be a good idea to include the concepts of object orientation so that, they are able to support object-oriented technologies too. The first two concepts that were added include the concept of complex types, inheritance, and some newer types such as multisets and arrays. One of the key concerns in object-relational database are the storage of tables that would be needed to represent inherited tables, and representation for the newer types.

One of the ways of representing inherited tables may be to store the inherited primary key attributes along with the locally defined attributes.

The second possibility here would be, to allow the data to be stored in all the inherited as well as base tables. However, such a case will result in data replication.

(ii) To create a new DBMS that is exclusively devoted to Object Oriented DBMS

Ans. The database system consists of persistent data. To manipulate that data one must either use data manipulation commands or a host language like C using embedded command. However, a persistent language would require a seamless integration of language and persistent data.

A practical approach for declaring a persistent object would be to design a construct that declares an objects as persistent. The difficulty with this approach is that it needs to declare object persistence at the time of creation, An alternative of this approach may be to mark a persistent object during run time.

All the objects created during the execution of an object oriented program would be given a system generated object identifier, however, these identifiers become useless once the program terminates. With the persistent objects it is necessary that such objects have meaningful object identifiers. Persistent object identifiers may be implemented using the concept of persistent pointers that remain valid even after the end of a program.

(c) Explain data fragmentation in DDBMS with the help of an examples.

Ans. DATA FRAGMENTATION

Data fragmentation allows you to break a single object into two or more segments or fragments. The object might be a user's database, a system database, or a table. Each fragment can be stored at any site over a computer network.

Information about data fragmentation is stored in the distributed data catalog (DDC), from which it is accessed by the TP to process user requests

There are three types of data fragmentation strategies:

- **Horizontal fragmentation** refers to the division of a relation into subsets (fragments) of tuples (rows). Each fragment is stored at a different node, and each fragment has unique rows. However, the unique rows all have the same attributes (columns). In short, each fragment represents the equivalent of a SELECT statement, with the WHERE clause on a single attribute.
- **Vertical fragmentation** refers to the division of a relation into attribute (column) subsets. Each subset (fragment) is stored at a different node, and each fragment has unique columns—with the exception of the key column, which is common to all fragments.

- **Mixed fragmentation** refers to a combination of horizontal and vertical strategies. In other words, a table may be divided into several horizontal subsets (rows), each one having a subset of the attributes (columns).

To illustrate the fragmentation strategies, let's use the CUSTOMER table for the XYZ Company, depicted in Figure bellow.

Table name: CUSTOMER

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
10	Sinex, Inc.	12 Main St.	TN	3500.00	2700.00	3	1245.00
11	Martin Crop.	321 Sunset Blvd.	FL	6000.00	1200.00	1	0.00
12	Mynux Corp.	910 Eagle St.	TN	4000.00	3500.00	3	3400.00
13	BTBC, Inc.	Rue du Monde.	FL	6000.00	5890.00	3	1090.00
14	Victory, Inc.	123 Maple St.	FL	1200.00	550.00	1	0.00
15	NBCC Crop.	909 High Ave.	GA	2000.00	350.00	2	50.00

Q5. (a) Explain the following in context of ORACLE/PostGRESQL.

(i) Triggers (ii) Security (iii) Data dictionary

Refer to June-2010, Q.No.-4(c), Page No.-264

(b) Explain the following:

(i) Embedded SQL

Ans. The embedded SQL statements can be put in the application program written in C, Java or any other host language. These statements sometime may be called static.

The term 'static' is used to indicate that the embedded SQL commands, which are written in the host program, do not change automatically during the lifetime of the program. Thus, such queries are determined at the time of database application design. For example, a query statement embedded in C to determine the status of train booking for a train will not change. However, this query may be executed for many different trains.

(ii) Deductive Databases

Refer to Page No.-180

(iii) OLAP

Refer to Page No.-179

(iv) Join dependency

Refer to Dec-2009, Q.No.-1(e), Page No.-244

(c) What is multiversion concurrency control? Explain, how multiversion concurrency control can be achieved by using Time Stamp Ordering.

Refer to June-2011, Q.No.-3(b), Page No.-282

MCS – 43: ADVANCED DATABASE DESIGN
June, 2013

***Note:** Question number **one** is compulsory. Answer **any three** questions from the rest.*

Q1. (a) Consider the following instance of a relation:

Employee Name	Project Name	Equipment
Sanjay	Inventory control	Computer
Sanjay	Secure website	Mobile
Sanjay	Secure website	Computer
Sanjay	Inventory control	Mobile
Harish	Intranet	Computer
Harish	Secure website	Computer

You may assume that projects are independent of equipments that may be allocated to a person. Also assume that employee name, project name and equipment are unique. Perform the following tasks for the description given above:

(i) Identify the primary key of the relation. Justify your selection.

Ans. The primary key of the relation is {Employee Name, Project Name, Equipment} because there is no FD exist in the relation therefore all the attributes have to take in the primary key.

(ii) Identify the FDs and MVDs in the relation. Give reasons for your selection.

Ans. There is no FD exist in the relation. The MVDs are (Employee Name →→ Project Name,
Employee Name →→ Equipment, Project Name →→ Equipment,
Project Name →→ Employee Name, Equipment →→ Project Name,
Equipment →→ Employee Name)

(iii) Normalise the relation upto 4th Normal Form (4NF). Show that your decomposition is lossless.

Ans. Loseless 4NF decomposition are

EP

Employee Name	Project Name
Sanjay	Inventory Control
Sanjay	Secure Website
Harish	Intranet
Harish	Secure Website

PE

Project Name	Equipment
Inventory Control	Computer
Secure Website	Mobile
Secure Website	Computer
Inventory Control	Mobile
Intranet	Computer

EE

Employee Name	Equipment
Sanjay	Computer
Sanjay	Mobile
Harish	Computer

The above decomposition is loseless because $\left(EP \bowtie PE \bowtie EE\right) \equiv EPE$

(b) Consider the following two tables:

Employee

EID	Name	Project
1	**Mohan**	**P_1**
2	**Shyam**	**P_2**
3	**Deep**	**P_3**
4	**Deepak**	**P_4**
.		
.		
.		

Project

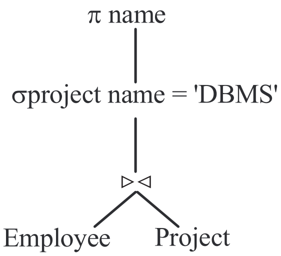
Project	Project Name
P_1	**DBMS**
P_2	**OS**
P_3	**Networks**
P_4	**OOPS**
.	
.	
.	

Consider the query

"List the names of all the employees working on DBMS project."

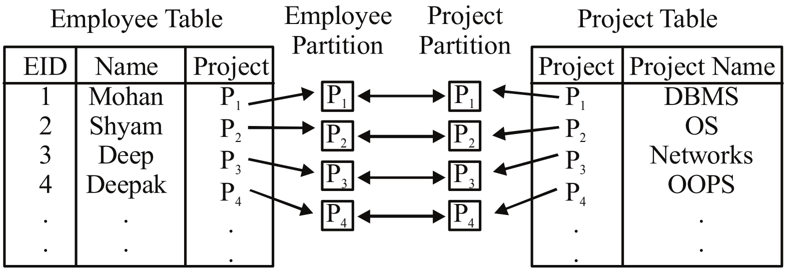
(i) Draw the query tree for the query.

Ans. π name (σ project name = 'DBMS' (EMPLOYEE $\bowtie$ PROJECT))



(ii) Assume that hash join is to be used to join the two tables, show the process of joining the two, tables using the hash join.

Ans.



EID	Name	Project
1	Mohan	P_1
2	Shyam	P_2
3	Deep	P_3
4	Deepak	P_4
.	.	.
.	.	.

Project	Project Name
P_1	DBMS
P_2	OS
P_3	Networks
P_4	OOPS
.	.
.	.

(c) Assume that a departmental store has many branches, it has many products and the sales information is recorded for every year. For example,

Year	Product Code	Branch Code	Sales amount
2009	A001	B001	5000
2009	A001	B002	6000
2010	A001	B002	3000
2010	A002	B002	2000
.			
.			
.			
.			

Identify the dimension tables and fact tables and create the star schema for above.

Ans.

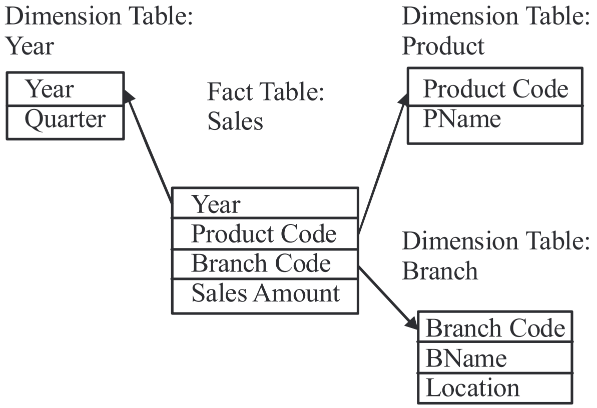


Figure: Star Schema

(d) Differentiate between the following:

(i) JDBC versus ODBC

Refer to Page No.-190 [JDBC] & Page No.-179 [ODBC]

(ii) B - Tree indexes versus R - Tree indexes used in postgre SQL

Ans. B-Tree Indexes: These are the default type index. These are useful for comparison and range queries.

R-Tree indexes: Such index are created on built-in spatial data types such as box, circle for determining operations like overlap etc.

(e) What is a system catalogue? Write SQL command to show all the table names in a database.

Ans. The system catalogue is a collection of tables and views that contain important information about a database. It is the place where a relational database management system stores schema metadata, such as information about tables and columns, and internal bookkeeping information. A system catalogue is available for each database. Information in the system catalogue defines the structure of the database.

The terms system catalogue and data dictionary have been used interchangeably in most situations.

(f) Consider that the adjustment of salary of the faculty members is done as follows, where Fac_Salary i represents the salary of i^{th} faculty member:

Transaction 1: Fac_Salary i: = Fac_Salary i + 1025

Transaction 2: Fac_Salary i: = Fac_Salary i * 1.1

If the above two transactions are run concurrently, what type of problems can occur. Justify.

Ans. If the given two transactions run concurrently, the lost update problem may occur. For example, consider the following schedule:

T_1	T_2	fac_salary i
Read (fac_salary i)		5000
	Read (fac_salary i)	5000
	fac_salary * = 1.1	
fac_salary + = 1.25		
	write (fac_salary i)	5500
write(fac_salary i)		6025

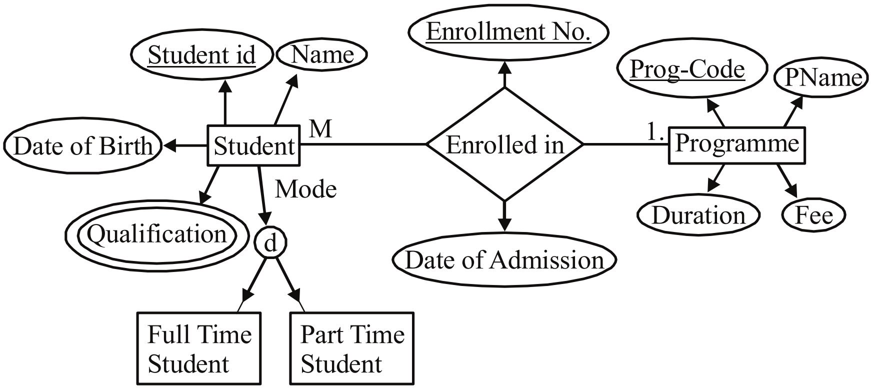
In the above schedule the update made by transaction T_2 is overwritten by transaction T_1. (Loss of update made by T_2)

(g) What is a multimedia database? What are the challenges in designing a multimedia database?

Refer to Ch-1, Q.No.-1 & 4, Page No.-140-142

Q2. (a) A University has many programmes. A student can enrol only one of these programme as full time or part time student. Create an EER diagram for the above. Make suitable assumptions. Convert the EER diagram into suitable relations.

Ans.



EER to Relational mapping:

Student (Student_id, Name, Date of Birth, mode)

Qualification (Student_id, Qualification)

Programme (Prog-code, PName, Duration, fee)

enrolled in (Enrol-no, Student_id, Prog-code, Date of admi)

(b) What is the need of exclusive and shared mode locks? Explain how these locks may be used in transaction management with the help of an example?

Ans. A lock based mechanism is used to maintain consistency when more than one transaction processes is executed. This mechanism controls concurrent access to data items. As per this mechanism the data items can be locked in two modes.

(1) Exclusive (X) mode: Data item can be both read as well as written. X-lock is requested using lock-X instruction.

(2) Shared (S) mode: Data item can only be read. S-lock is requested using lock-S instruction.

Lock requests are made to the concurrency-control manager. The transaction can proceed only after request is granted. The Lock-compatibility matrix shows when lock request will be granted (True).

	Unlocked	**S**	**X**
S	True	True	True
X	True	False	False

In general, a transaction may be granted a lock on an item if the requested lock is compatible with locks already held on the item by other transactions. Any number of transactions can hold shared locks on an item, but if any transaction holds an exclusive lock on the item, then no other transaction may hold any lock on the item. If a lock cannot be granted, the requesting transaction is made to wait till all incompatible locks held by other transactions have been released. The lock is then granted. The following is an example of a transaction that requests and is granted/permitted locks:

```
T2 : lock-S(X);
read(X);
unlock (X);
lock-S(Y);
read (Y);
unlock (Y);
display (X + Y)
```

Locking as above is not sufficient to guarantee serialisability–if X and Y are updated in-between the read of X and Y by some other transaction, the displayed sum would be wrong. A locking protocol is a set of rules to be followed by all transactions while requesting and releasing locks. Locking protocols restrict the set of possible schedules.

(c) What is data mining? Explain with the help of an example situation. What is classification approach of data mining? How is it different then clustering?

Ans Data mining is the process of automatic extraction of interesting (non trivial, implicit, previously unknown and potentially useful) information or patterns from the data in large databases.

Example:

Now that we have defined data, information and knowledge let us define some of the problems that can be solved through the data mining process.

Mr Ramniwas Gupta manages a supermarket and the cash counters, he adds transactions into the database. Some of the questions that can come to Mr. Gupta’s mind are as follows:

(a) Can you help me visualise my sales?

(b) Can you profile my customers?

(c) Tell me something interesting about sales such as, what time sales will be maximum etc.

He does not know statistics, and he does not want to hire statisticians.

The answer of some of the above questions may be answered by data mining.

The classification task maps data into predefined groups or classes. The class of a tuple is indicated by the value of a user-specified goal attribute. Tuples consists of a set of predicating attributes and a goal attribute. The task is to discover some kind of relationship between the predicating attributes and the goal attribute, so that the discovered knowledge can be used to predict the class of new tuple(s).

Q3. (a) Explain the process of log based recovery using immediate database modification scheme with the help of an example.

Ans. Log-Based Recovery

A log is maintained on a stable storage media. The log is a sequence of log records, and maintains a record of update activities on the database. When transaction T_i starts, it registers itself by writing a < T_i start> log record.

Before T_i executes write(X), a log record < T_i X,V_1,V_2> is written, where V_1 is the value of X before the write (undo value), and V_2 is the value to be written to X (redo value).

- Log record notes that T_i has performed a write on data item X. X had value V_1 before the write, and will have value V_2 after the write.
- When T_i finishes it last statement, the log record < T_i commit> is written. We assume for now that log records are written directly to a stable storage media (that is, they are not buffered).

Two approaches for recovery using logs are:

- Deferred database modification.
- Immediate database modification.

Immediate Database Modification: The immediate database modification scheme allows database updates on the stored database even of an uncommitted transaction. These updates are made as the writes are issued (since undoing may be needed, update logs must have both the old value as well as the new value). Updated log records must be written before database item is written (assume that the log record is output directly to a stable storage and can be extended to postpone log record output, as long as prior to execution of an output (Y) operation for a data block Y all log records corresponding to items Y must be flushed to stable storage).

The recovery procedure in such has two operations instead of one:

- undo (T_i) restores the value of all data items updated by T_i to their old values, moving backwards from the last log record for T_i ,
- redo (T_i) sets the value of all data items updated by T_i to the new values, moving forward from the first log record for T_i .

When recovering after failure:

- Transaction T_i needs to be undone if the log contains the record < T_i start>, but does not contain the record < T_i commit>.
- Transaction T_i needs to be redone if the log contains both the record < T_i start> and the record < T_i commit>.

Undo operations are performed first, then redo operations.

Example:

Consider the log as it appears at three instances of time.

<T_1 start>	<T_1 start>	<T_1 start>
<T_1, X10000, 9000>	<T_1, X10000, 9000>	<T_1, X10000, 9000>
<T_1, Y8000, 9000>	<T_1, Y8000, 9000>	<T_1, Y8000, 9000>
	<T_1, Commit>	<T_1, Commit>
	<T_2 start>	<T_2 start>
	<T_2, Z20000, 19000>	<T_2, Z20000, 19000>
		<T_2, Commit>
(a)	(b)	(c)

Recovery actions in each case above are:

(a) undo (T_1): Y is restored to 8,000 and X to 10,000.

(b) undo (T_2): and redo (T_1): Z is restored to 20,000, and then X and Y are set to 9,000 and 9,000 respectively.

(c) redo (T_1) and redo (T_2): X and Y are set to 9,000 and 9,000 respectively. Then Z is set to 19,000

(b) (i) What is difference between serial schedule and serialisable schedule? What are the conditions for a schedule of transactions to be serialisable, give an example of a serialisable schedule?

Ans.A serial schedule is when all the operations of one transactions appear together (not mixed with the operations of any other transactions on the schedule).

A serialisable schedule is a weaker term–it is a schedule where the operations of different transactions may be mixed together on the schedule, so long as they are conflict-equivalent to some serial schedule.

A schedule can be called as serialisable schedule if it satisfies either view-serializability and/or conflict-serializability.

View-serialisability of a schedule is defined by equivalence to a serial schedule (no overlapping transactions) with the same transactions, such that respective transactions in the two schedules read and write the same data values ("view" the same data values).

Conflict-serializability is defined by equivalence to a serial schedule (no overlapping transactions) with the same transactions, such that both schedules have the same sets of respective chronologically ordered pairs of conflicting operations (same precedence relations of respective conflicting operations).

Example:

the following schedule is a conflict serialisable schedule:

$$w_1(A), r_2(A), w_1(B), w_3(C), r_2(C), r_4(B), w_2(D), w_4(E), r_5(D), w_5(E)$$

(ii) What are the different types of θ - join operations available in relational algebra?

Ans. Natural join $\left(\bowtie\right)$

Natural join ($\bowtie$) is a binary operator that is written as (R $\bowtie$ S) where R and S are relations. The result of the natural join is the set of all combinations of tuples in R and S that are equal on their common attribute names. For an example consider the tables Employee and Dept and their natural join:

Employee

Name	EmpId	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Dept

DeptName	Manager
Finance	George
Sales	Harriet
Production	Charles

Employee ∞ Dept

Name	EmpId	DeptName	Manager
Harry	3415	Finance	George
Sally	2241	Sales	Harriet
George	3401	Finance	George
Harriet	2202	Sales	Harriet

θ -join and equijoin

Consider tables Car and Boat which list models of cars and boats and their respective prices. Suppose a customer wants to buy a car and a boat, but she does not want to spend more money for the boat than for the car. The θ -join $\left(\bowtie_\theta\right)$ on the relation Car Price $\geq$ Boat Price produces a table with all the possible options. When using a condition where the attributes are equal, for example price, then the condition may be specified as Price=Price or alternatively (Price) itself.

Car

CarModel	CarPrice
CarA	20,000
CarB	30,000
CarC	50,000

Boat

BoatModel	BoatPrice
Boat1	10,000
Boat2	40,000
Boat3	60,000

Car ∞ Boat

CarPrice ≥ BoatPrice

CarModel	CarPrice	BoatModel	BoatPrice
CarA	20,000	Boat1	10,000
CarB	30,000	Boat1	10,000
CarC	50,000	Boat1	10,000
CarC	50,000	Boat2	40,000

If we want to combine tuples from two relations where the combination condition is not simply the equality of shared attributes then it is convenient to have a more general from of join operator, which is the θ -join (or theta-join). The θ -join is a binary

R $\bowtie$ S R $\bowtie$ S

operator that is written as a θ b or u θ v where a and b are attribute names, θ is a binary relation in the set $\{<,\leq,=,>,\geq\}$, v is a value constant, and R and S are relations. The result of this operation consists of all combinations of tuples in R and S that satisfy the relation θ. The result of the θ -join is defined only if the headers of S and R are disjoint, that is, do not contain a common attribute. The simulation of this operation in the fundamental operations is therefore as follows:

$$R \bowtie_\theta S = \sigma_\theta (R \times S)$$

In case the operator is the equality operator (=) then this join is also called an equijoin.

Outer join

An outer join does not require each record in the two joined tables to have a matching record. The joined table retains each record–even if no other matching record exists. Outer joins subdivide further into left outer joins, right outer joins, and full outer joins, depending on which table's rows are retained (left, right, or both).

Left outer join

The result of a left outer join (or simply left join) for table A and B always contains all records of the "left" table (A), even if the join-condition does not find any matching record in the "right" table (B). This means that if the ON clause matches 0 (zero) records in B (for a given record in A), the join will still return a row in the result (for that record) —but with NULL in each column from B. A left outer join returns all the values from an inner join plus all values in the left table that do not match to the right table.

For example, this allows us to find an employee's department, but still shows that employee(s) even when they have not been assigned to a department (contrary to the inner-join example above, where unassigned employees were excluded from the result).

Right outer join

A right outer join (or right join) closely resembles a left outer join, except with the treatment of the tables reversed. Every now from the "right" table (B) will appear in the joined table at least once. If no matching row from the "left" table (A) exists, NULL will appear in columns from A for those records that have no match in B.

A right outer join returns all the values from the right table and matched values from the left table (NULL in the case of no matching join predicate). For example, this allows us to find each employee and his or her department, but sill show departments that have no employees.

Full outer join

Conceptually, a full outer join combines the effect of applying both left and right outer joins. Where records in the FULL OUTER JOINed tables do not match, the result set will have NULL values for every column of the table that lacks a matching row. For those records that do match, a single row will be produced in the result set (containing fields populated from both tables).

For example, this allows us to see each employee who is in a department and each department that has an employee, but also see each employee who is not part of a department and each department which doesn't have an employee.

(iii) Explain the concept of “cursor” and “Trigger” in SQL.

Refer to Ch-5, Q.No.-36, Page No.-93 & 94

Q4. (a) Consider the following DTD in XML:

```
<[
<?xml version =”1.0" encoding =”UTF – 8"?>
<!DOCTYPE Customer>
<!ELEMENT Customer (Person, Address+)>
<!ELEMENT Person (F name, L name)>
<!ELEMENT F name (# PCDATA)>
<!ELEMENT L name (# PCDATA)>
<! ELEMENT Address (# PCDATA)>
]>
```

Create at least two customer records in XML format using the DTD.

Ans.

```
<DOCUMENT>
<CUSTOMER>
        <PERSON>
<FNAME> Ramesh </FNAME>
<LNAME> chand </ LNAME>
</PERSON>
        <Address> Ashok Vihar <Address>
</Customer>
<Customer>
<PERSON>
        <Fname> Anil </Fname>
        <Lname> Kapoor </Lname>
        </Person>
<Address> Juhu, Mumbai </Address>
<Address> Delhi </Address>
</Customer>
</Document>
```

(b) Define the term join dependency with the help of an example.

Refer to Dec-2008, Q.No.-4(b), Page No.-232

(c) Explain Horizontal and Vertical Fragmentation in distributed DBMS.

Ans. Horizontal Fragmentation: refers to the division of a relation into subsets (fragments) of tuples (rows). Each fragment is stored at a different node, and each fragment has unique rows. However, the unique rows all have the same attributes (columns). In short, each fragment represents the equivalent of a SELECT statement, with the WHERE clause on a single attribute.

Vertical fragmentation: refers to the division of a relation into attribute (column) subsets. Each subset (fragment) is stored at a different node, and each fragment has unique columns—with the exception of the key column, which is common to all fragments. This is the equivalent of the PROJECT statement SQL.

(d) Explain the role of the following files in oracle.

(i) Control files **(ii) Data files**

Refer to Page No.-208

Q5. (a) Define the following giving an example/diagram, if needed.

(i) Data Grid

Refer to Ch-7, Q.No.-20, Page No.-146

(ii) Data Mart

Refer to Dec-2008, Q.No.-3(b), Page No.-231

(iii) Deadlock

Refer to Chapter-5, Q.No.-24, Page No.-83

(iv) Check Point

Refer to Page No.-162[Check Point]

(v) Referential Integrity constraint

Ans. This constraint identifies any column referencing the PRIMARY KEY in another table. It establishes a relationship between two columns in the same table or between different tables. For a column to be defined as a Foreign Key, it should be a defined as a Primary Key in the table which it is referring. One or more columns can be defined as Foreign key.

Syntax to define a Foreign key at column level:

[CONSTRAINT constraint_name] REFERENCES Referenced_Table_ name (column_name)

Syntax to define a Foreign key at table level:

[CONSTRAINT constraint_name] FOREIGN KEY (column_name) REFERENCES referenced_table_name(column_name);

(b) What are the different types of security features needed for a multilevel security system? Explain the process of partitioning and encryption in a multilevel security system.

Ans. MULTILEVEL SECURITY

System may have the following three security features:

- Security of different objects may be different from the other attribute values of that tuple or security may be different from the other values of the attributes.
- Thus, every individual element is the micro item for security.
- In addition, security against statistical queries may also need to be provided.
- A number of security level needs to be defined for such security.

There are many techniques to support multi-level security. Two important methods of these are:

Partitioning: In the approach the original database is divided into partitions. Each of the partitions has a different level of security.

However, the major drawback of this approach is that the database looses the advantage of a relation.

Encryption: Encryption of critical information may help in maintaining security as a user who accidentally receives them cannot interpret the data. Such a scheme may be used when a user password file is implemented in the database design. However, the simple encryption algorithms are not secure, also since data is available in the database it may also be obtained in response to a query.

MCS – 43: ADVANCED DATABASE DESIGN

December, 2013

***Note:** Question number **one** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) Consider the following situation:

"A company has two employee named "A" and "B". The company has two projects P1 and P2 project P1 uses languages C and C++, whereas project P2 uses languages C and JAVA. Employee whose name is "A" works on both the projects P1 and P2, whereas employee named "B" works on project P2 only."

(i) Represent the information given above as a relation/table S (employee name, project, language).

(ii) List the FDs and MVDs in the table above.

(iii) Normalise the table upto 4th Normal Form.

(b) Define the term-"view" with the help of an example.

(c) Consider the following relations.

Supplier (sid, Sname)

parts (pid, pname)

Supp-part (sid, pid, quantity)

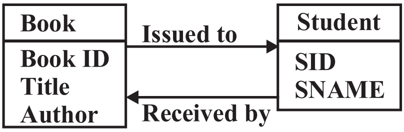
(i) Write a relational algebraic expression for the following query:

"List the names of the supplier's who have supplied the part whose pname is "Bolt".

(ii) Draw the query tree for the query given above.

(d) How is a checkpoint useful in log based recovery? Explain it with the help of an example/diagram

(e) Represent the following using Object Definition language (ODL)



Please note that diagram indicates that a book is issued to a student who receives it. The attributes of Book objects are Book ID, Title and Author; and the attributes of student objects are SID and SNAME.

(f) What is a data warehouse? Why is it needed by an organisation? Explain the process of Data Extraction, Transformation and Loading (ETL) in the context of a data warehouse

(g) What is ODBC? Why is it needed? What are the various components needed to implement ODBC?

(h) What are control files in Oracle? What does a control file contain? What is the role of control file in starting an instance of oracle. What is the purpose of redo log file during this time.

Q2. (a) Draw an EER diagram for the following situation: A vehicle can be owned by an Individual or an organisation. In case the vehicle is owned by an organisation then the database should store the name and designation of contact person in the organisation, the office address, and the name of the organisation along with name of the head. However, for an individual owner just name, address and phonenumber of owner needs to be recorder. "You may assume that the basic information that is to be stored about the vehicle should be-vehicle registration number type, date of model, date of purchase."
(b) Convert the EER diagram created in part (a) to equivalent RDBMS tables having proper keys and constraints.
(c) Define the term data mining. How data mining is different from Knowledge Discovery in data bases (KDD)? Explain the steps of KDD with the help of an example.
(d) Explain any three common database security failures. Explain with the help of an example, how SQL can be used to enforce access control in a database.

Q3. (a) Assume the following three transactions:
T1: "Increment the X account by an amount ₹1000/- and Decrement Y is account by an amount. ₹1000/-"
T2: Decrement X account by an amount of ₹2000/- and increment Z's account by ₹2000/-
T3: Add the amount present in the accounts X, Y and Z to Total amount use the shared (s) mode and exclusive (k) mode locks to write the pseudo code of transactions T1, T2 and T3, such that they can be executed concurrently. Explain your answer.
(b) What is embedded SQL? Give an example of embedded SQL statement in 'C' language. What is the role of cursor in embedded SQL? Explain with the help of an example.
(c) Differentiate between the following:
(i) JDBC versus ODBC
(ii) Distributed Database versus centralized Database
(iii) Redo versus undo

Q4. (a) Consider the following XML document:
<?xml version = "1.0"encity ="UTF — 8"?>
<Customer>
<name>

```
        <fname>Lokesh </fname>
        <lname> Jain </lname>
</name>
        <type>Home</ type>
        <phone>297511</ phone>
</customer>
<customer>
        <name>
                <fname> Ramesh</fname>
                <lname>Nagesh</ lname>
        </name>
        <type> Company</type>
        <phone>293575</phone>
        <phone>33333</ phone>
</ customer>
```

Write the DTD for the XML document given above.

(b) What is the role of a trigger? Explain with the help of an example. How is a trigger different to a stored procedure? Explain.

(c) Consider the following Relation schema R (Faculty, Dean, Department, Chairperson, professor, Rank, student) and the following set of functional dependencies:

Faculty → Dean

Dean → Faculty

Department → Chairperson

Professor → Rank, chairperson

Department → Faculty

Student → Department, Faculty, Dean

Professor, Rank → Department, Faculty

Obtain 3NF decomposition of the above relation schema and show each step explicitly

Q5. (a) Explain the following with the help of an example/diagram, if needed.

(i) Role of system catalogue

(ii) Nested loop join

(iii) Timestamp and its use

(iv) Deadlock

(v) Fragmentation and allocation schema

(vi) Snoflake schema in data warehouse

(b) Explain the multidimensional Data Modeling for a Datawarehouse with an example.

MCS – 43: ADVANCED DATABASE DESIGN
June, 2014

Note: *Question number* ***one*** *is* ***compulsory****. Answer* ***any three*** *questions from the rest.*

Q1. (a) Explain lossless decomposition and dependency preserving. Consider the following relation scheme:
R (A, B, C, D, E, F) and FDs
$\mathbf{A \rightarrow BC, C \rightarrow A, D \rightarrow E, F \rightarrow A, E \rightarrow D}$

Is the decomposition of R into R_1 (A, C, D), R_2 (B, C, D) and R_3 (E, F, D) lossless and dependency preserving.

Ans. Lossless means functioning without a loss. In other words, retain everything. Important for databases to have this feature.

- Let R be a relation schema.
- Let F be a set of functional dependencies on R.
- Let and form a decomposition of R.
- The decomposition is a lossless-join decomposition of R if at least one of the following functional dependencies are in F^+

$(1) R_1 \cap R_2 \rightarrow R_1$

$(2) R_1 \cup R_2 \rightarrow R_2$

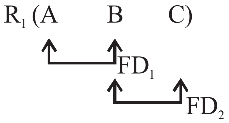
A decomposition $D = \{R_1, R_2, ..., R_n\}$ of R is dependency-preserving with respect to F if the union of the projections of F on each R_i in D is equivalent to F; that is if $(F_1 \cup F_2 \cup ... \cup F_n)^+ = F^+$

Example:

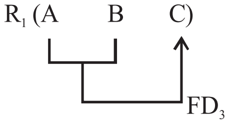
- R(A B C D)
- $FD_1 : A \rightarrow B$
- $FD_2 : B \rightarrow C$
- $FD_3 : C \rightarrow D$
- **Decomposition:**

R_1 **(A B C)** R_2 **(C D)**

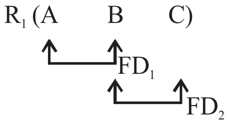
- $FD_1 : A \rightarrow B$
- $FD_2 : B \rightarrow C$
- $FD_3 : C \rightarrow D$



- $FD_1 : A \rightarrow B$
- $FD_2 : B \rightarrow C$
- $FD_3 : C \rightarrow D$



- $FD_1 : A \rightarrow B$
- $FD_2 : B \rightarrow C$
- $FD_3 : C \rightarrow D$



Have all 3 functional dependencies. Therefore, it's preserving the dependencies. Yes, the decomposition of R into (A, C, D), (B, C, D) and (E, F, D) lossless and dependency preserving.

(b) What is meant by a schedule in the context of concurrent transactions in Database? Also explain serial and serialisable schedules with the help of a suitable example.

Ans. A schedule is a collection of many transactions which is implemented as a unit. Depending upon how these transactions are arranged in within a schedule, a schedule can be of two types:

- Serial: The transactions are executed one after another, in a non-preemptive manner.
- Concurrent: The transactions are executed in a preemptive, time shared method.

In Serial schedule, there is no question of sharing a single data item among many transactions, because not more than a single transaction is executing at any point of time. However, a serial schedule is inefficient **in the sense that** the transactions suffer for having a longer waiting time and response time, as well as low amount of resource utilisation.

Let us consider there are two transactions T_1 and T_2,whose instruction sets are given as following. T_1 is the same as we have seen earlier, while T_2 is a new transaction.

T_1	T_2
Read (A)	
A = a – 100	Write (A)
Read (B)	
B = B + 100	Write (B)

T_1	T_2
Read (A)	
Temp = a*0.1	Read (C)
	C = C+ Temp
Write (C)	

T_2 is a new transaction which deposits to account C 10% of the amount in account A.

If we prepare a serial schedule, then either T_1 will completely finish before T_2 can begin, or T_2 will completely finish before T_1 can begin. However, if we want to create a concurrent schedule, then some Context Switching need to be made, so that some portion of T_1 will be executed, then some portion of T_2 will be executed and so on. For example say we have prepared the following concurrent schedule.

T_1	T_2
Read (A)	
A = A – 100	
	Write (A)

T_1	T_2
Read (B)	
B=B+100	
	Write (B)

In concurrent schedule, CPU time is shared among two or more transactions in order to run them concurrently. However, this creates the possibility that more than one transaction may need to access a single data item for read/ write purpose and the database could contain inconsistent value if such accesses are not handled properly.

Serializability: When several concurrent transactions are trying to access the same data item, the instructions within these concurrent transactions must be ordered in some way so as there are no problem in accessing and releasing the shared data item.

There are two aspects of Serializability which are described here:

(1) Conflict Serializability

(2) View Serializability

T_1	T_2
Write (Q)	
	Read (Q)
Read (R)	
	Write (Q)

T_1	T_2
Write (Q)	
Read (R)	
	Read (Q)
	Write (Q)

Figure: Conflict Serializability

T_1	T_2
Write (Q)	
	Read (Q)
Read (R)	
	Write (Q)

T_1	T_2
Write (Q)	
Read (R)	
	Read (Q)
	Write (Q)

Figure: Views Serializability

(c) Define locking in concurrency control. Discuss the various types of locking techniques.

Ans. Locking is Technique used to control concurrent access to Data. Locking is the one of the most widely used technique to ensure Serializability. Principle used in locking Technique is as follows. "A transaction must contain a read or write lock on data item before it can perform a read or write operation".

The basic rules implemented in locking technique are as follows

- If a transaction has read lock on a data item. It can performer only read operation not write.
- If a transaction has read lock on a data item and other transaction want to use it then it can implement only read lock not write lock on this data.
- If a transaction has write lock on a data item. It can performer Both read operation and write operation.
- If a transaction has write lock on a data item and other transaction want to use it then it can't implement neither read lock nor write lock on this data.

Types of locking Technique

- The two Phase locking protocol
- Time Stamping Protocol
- Validation Based protocol

The two Phase locking protocol:

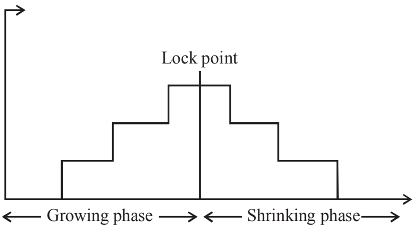
The two-phase locking protocol is used to ensure the Serializability in Database. This protocol is implemented in two phase:

- Growing Phase-

In this phase we put read or write lock based on need on the data. In this phase we does not release any lock.

- Shrinking Phase-

This phase is just reverse of growing phase. In this phase we release read and write lock but doesn't put any lock on data.



Time Stamping Protocol: In Time stamping Protocol we select sequence of transaction in advance by using Time stamping concept. We add a special variable time stamp (a unique, fixed non decreasing) to each transaction in system. This number can be system clock value. When a new transaction entered in the system, current value of cock is assigned to transaction as time stamp value. Value of time stamp is incremented every time after a new transaction interred in the system.

Example: If there are two transaction t1 and t2 in system. Transaction t1 has s1 time stamp value and transaction t2 has s2 time stamp value and s1<s2. In this case Transaction t1 processed first then transaction t2.

Validation Based protocol: Two-phase locking protocol and Time stamping Protocol are slow in working because they worked in two steps, so we use validation based protocol which is faster than Two-phase locking protocol and Time stamping Protocol.

In Validation based protocol, it doesn't update entire data base in one step. It keeps local copies of all updates during transaction execution. It works in three steps

(1) Read Phase: In this step transaction is activated and it reads last committed value from DB and put these values in local variables. All updates implemented in these local variables of database.

(2) Validation Phase: In this step it checks read value during the first phase against current values. It also check consistency of data base after modification performed in database.

(3) Write Phase: If validation phase say that data base is in consistent state then all update made by transaction which are in local variable are applied in database for permanent storage. If validation phase say that database is not consistent or it violate serializability then it discard/rollback all updates and it restart transactions.

(d) How does Boyce-codd Normal form differ from 3NF? Why is it considered stronger than 3NF, explain using a suitable example.

Ans. Normalisation is a process that is carried out to minimise the redundancies that are present in data in relational database. This process will mainly divide large tables in to smaller tables with fewer redundancies. These smaller tables will be related to each other through well-defined relationships. In a well normalised database, any alteration or modification only a single table. Third normal form (3NF) was introduced in 1971 by Edgar F. Codd, who is also the inventor of normalisation. Boyce-codd Normal form (BCNF) was introduced in 1974 by codd and Raymond F. Boyce.

Both 3NF and BCNF are normal forms that are used in relational databases to minimise redundancies in tables. In a table that is in the BCNF normal form, for every non-trivial functional dependency of the form A is a super-key whereas, a table that compiles with 3NF should be in the 2NF, and every non-prime attribute should directly depend on every candidate key of that table. BCNF is considered as a stronger normal form than the 3NF and it was developed to capture some of the anomalies that could not be captured by 3NF. Obtaining a table that compiles with the BCNF form will require decomposing a table that is in the 3NF. This decomposition will result in additional join operations (or Cartesian products) when executing queries. This will increase the computational time. On the other hand, the tables that comply with BCNF would have fewer redundancies than tables that only comply with 3NF. Furthermore, most of the time, it is possible to obtain a table that comply with 3NF without hindering dependency preservation and lossless joining. But this is not always possible with BCNF.

Example:-

For every functional
Dependency X→Y in a set F
Of functional dependencies
Over relation R, either:
–Y is a subset of X or,
–X is a super key of R
–Y is a subset of K for some key K of R

3NF has some redundancy
BCNF does not
Unfortunately, BCNF is not dependency preserving, but 3NF is

Account	Client	Office
A	Joe	1
B	Mary	1
A	John	1
C	Joe	2

Client, Office → Client, Office, Account
Account → Office

Account	Office
A	1
B	1
C	2

Account → Office

Lossless-dependencies

Account	Client
A	Joe
B	Mary
A	John
C	Joe

Non-trivial FDs

Q2. (a) Discuss the five basic operations of relational algebra with suitable example for each.

Ans. There are five basic operations:

(1) Union,

(2) Difference,

(3) Cartesian Product,

(4) Projection,

(5) Selection.

Union- The Union of relations and S is defined as:

$R \bigcup S = \{\langle x_1,...,x_n\rangle : \langle x_1,...,x_n\rangle \in R \, V \langle x_1,...,x_n\rangle \in S\}.$

It is the set of tuples that are in R or S. This operation may be applied to the relations of the same arity only, so all the tuples in the result have the same number of components (values).

Difference- The Difference of relations R and S, defined as:

$R - S = \{\langle x_1,...,x_n\rangle \in R : \langle x_1,...,x_n\rangle \notin S\}.$

It is the set of tuples which are in R but not in S. It is required that R and S have the same arity.

Cartesian Product-The Cartesian Product of relations R and S is defined as:

$R \times S = \{\langle x_1,...,x_n,y_1,...,y_m\rangle : \langle x_1,...,x_n\rangle \in R \wedge \langle y_1,...,y_m\rangle \in S\}.$

Assume that R has arity n and S has arity m. R X S is defined as the set of all possible (n + m) tuples which first n components form a tuple in R and last m form a tuple in S. The result has the arity equal to n + m.

Projection- The Projection is denoted as p. Assuming that there is a relation R of arity K, then:

$$\pi_{i_1,i_2,...,i_m}(R).$$

It denotes the projection of R onto components $i_1, i_2, ...i_n$ where I_j are distinct integers in range 1....... k. That is the set of m-tuples $(a_1, a_2, ...a_m)$ such that there is some k-tuple $(b_1, b_2, ...b_m)$ in R for which $a_i = b_{ij}$ for j=1, 2, ..., m.

Selection- The S election is denoted as:

$$^{\sigma}F(R).$$

Where R is a relation and F is a formula involving:

- Operands that are constants or component numbers; component i is represented by $ i,
- The arithmetic comparison operators <, = , >.
- The logical operators Ù (and), Ú (or), Ø(not).

Then $^{\sigma}F(R)$ is the set of tuples t in R such that, when for all i the i-th component of t is substituted for any occurrences of $i in formula F, the formula F becomes true. As for projection if a relation has named columns, then the formula in the selection can refer to columns by name instead of by number. The arity of $^{\sigma}F(R)$ is the same as the arity of R..

(b) Describe the architecture of distributed databases with the help of a diagram.

Ans. Distributed database systems are very diverse in nature; hence it may be a good idea to define reference architecture for such database systems. Reference architecture may also define the following additional schema for distributed database systems:

- A set of global schemas that would define application interface. Thus, a global schema can be further divided into:
- Global external schema
- Global conceptual schema
- Fragmentation and allocation schema will determine various fragments and their allocation to various distributed database sites.
- Finally, it would require basic schemas as proposed by ANSI/SPARC architecture for local databases.

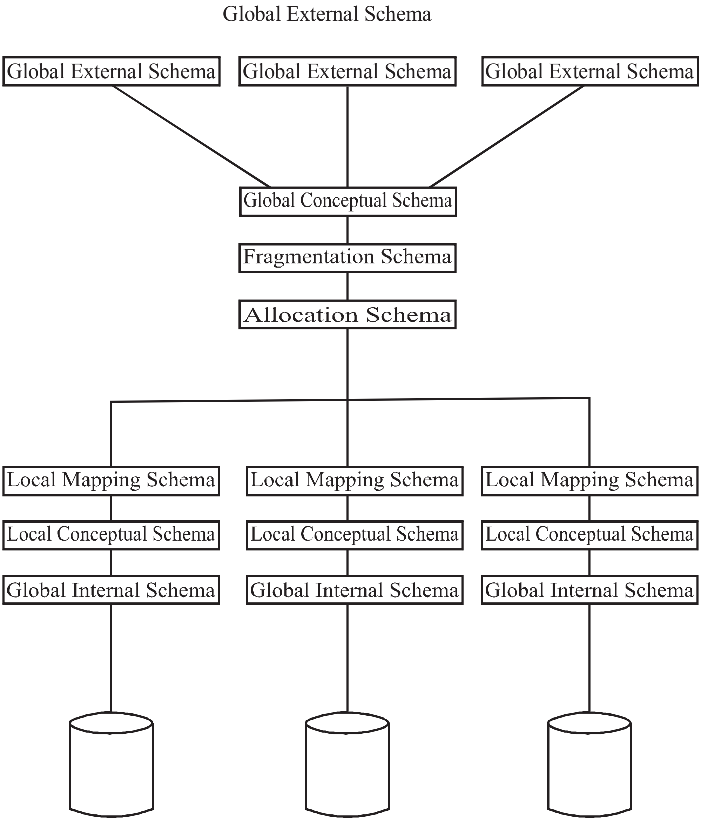
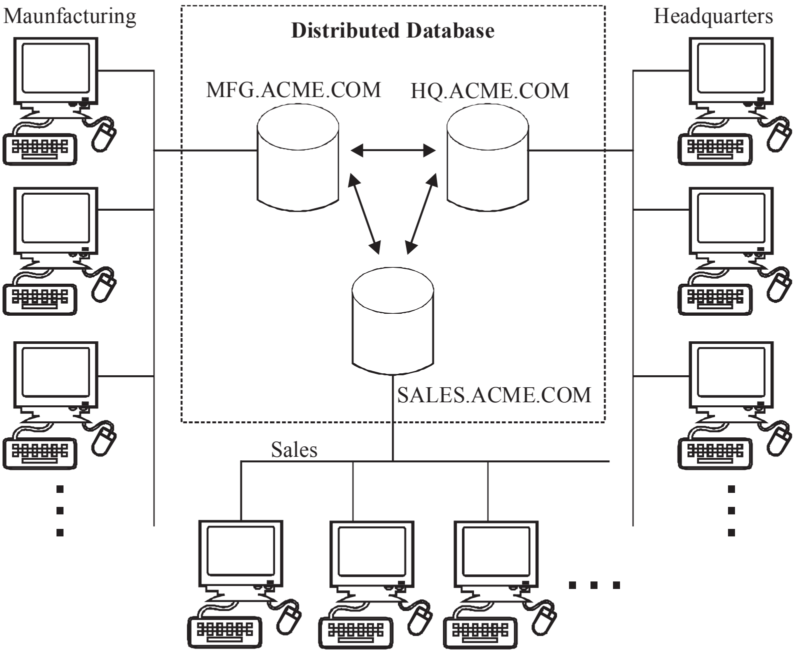


Figure: Architecture for Distributed database system

Example:



Q3. (a) Consider the following scheme for project database:
Project (PR_NO, PR_Name, PR_Manager)
Employee (Emp_NO, Emp_Name)
Assigned_To (PR_NO, Emp_NO)
(i) Write the DDL statements for the Project Database. Clearly specify the primary and foreign keys.
(ii) Write the following queries in SQL:
(a) List the details of employees working on PR_NO "A34" and "B64".
(b) Delete the record of employee whose Emp_NO is "E64221".
(c) List the name of employees who are working on a project for which "Ramesh" is a Project Manager.

Ans. (i)

```
CREATE TABLE Project (
                PR_NO           INTEGER PRIMARY KEY,
                PR_Name         VARCHAR (50) NOT NULL,
                PR_Manager      VARCHAR (50) NOT NULL,
                );
   CREATE TABLE Employee (
                Emp_NO INTEGER PRIMARY KEY,
                Emp_Name        VARCHAR (50) NOT NULL,
                );
```

```
        CREATE TABLE Assigned (
                        PR_NO           INTEGER FOREIGN KEY,
                        Emp_NO INTEGER FOREIGN KEY,
                        );
```

(ii)

```
(a) SELECT Emp. Emp_NO, Emp. Emp_Name FROM Employee Emp,
Assigned A
WHERE Emp.Emp_NO=A.Emp_NO (SELECT * FROM Assigned A
                                WHERE A.PR_NO= "A34" AND
                                A.PR_NO= "B64"
                                );
(b) DELETE from Employee
WHERE Emp_NO= "E64221";
(c) SELECT Emp_Name from Employee, Project, Assigned
WHERE Assigned. PR_NO=Project.PR_NO and PR_Manager=" Ramesh";
```

(b) Define weak entity set in ER diagram. How are keys of the weak entities identified? Discuss the mapping of strong entity set and weak entity set into relations.

Ans. Weak entity is an entity that cannot be uniquely identified by its attributes alone. In entity relationship diagrams a weak entity set is indicated by a bold (or double-lined) rectangle (the entity) connected by a bold (or double-lined) type arrow to a bold (or double-lined) diamond (the relationship). This type of relationship is called an identifying relationship and in IDEF1X notation it is represented by an oval entity rather than a square entity for base tables.

An identifying relationship is one where the primary key is populated to the child weak entity as a primary key in that entity.

Weak entities do not have sufficient attributes to form a primary key on its own. It is represented by a double rectangle. It contains a partial key or discriminator represented by a dashed underline. The member of weak entity set is called as subordinate entity set. The primary key of weak entity set is a combination of partial key and primary key of the strong entity set.

The line connecting weak entity set with the identifying relationship is double.

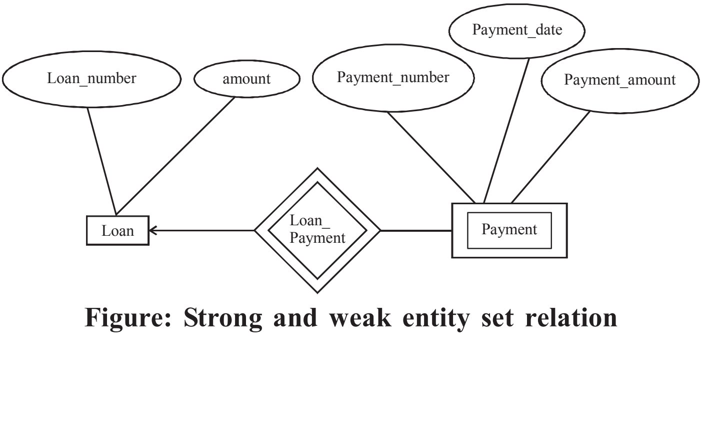


Figure: Strong and weak entity set relation

Q4. (a) Describe the term MVD (Multi-Valued Dependency) and JD (Join Dependency) in the context of relational DBMS by giving are example. Also, differentiate between 4 NF and 5 NF with an example.

Ans. A multivalued dependency is a full constraint between two sets of attributes in a relation. In contrast to the functional independency, the multivalued dependency requires that certain tuples be present in a relation. Therefore, a multivalued dependency is also referred as a tuple-generating dependency. The multivalued dependency also plays a role in 4NF normalisation.

MVD Formal definition

Let R be a relation scheme and X and Y are subsets of R,

X ->> Y

holds if for all pairs of tuples s and t in r, such that

s[X] = t[X] there also exist tuples u and v where

s[X] = t[X] = u[X] = v[X]

s[Y] = u[Y]

t[R-XY] = u[R-XY]

s[R-XY] = v[R-XY]

t[Y] = v[Y]

MVD Example

Course ->> Instructor

Course ->> Text

Course (Y)	**Instructor (X)**	**Text (R-XY)**
Intro	Kruse	Intro to CS
Intro	Wright	Intro to CS
CS1	Thomas	Intro to Java
CS1	Thomas	CS Theory Survey
CS2	Rhodes	Java Data Structures
CS2	Rhodes	Unix
CS2	Kruse	Java Data Structures
CS2	Kruse	Unix

Join dependencies are particularly important in connection with the decomposition technique for schema design and normalisation. The main goal of the decomposition technique is to avoid redundancies due to data dependencies by decomposing a relation into smaller parts. A good decomposition should have the lossless join property, meaning that no information should be lost after the decomposition.

A join dependency (JD) over a relation schema R [U] is an expression of the form È” $[X_1,...,X_n]$, where $X_1 U....UX_{n=U.}$ An instance I of R [U] satisfies È” $[X_1,...,X_n]$ if

In other word, an instance satisfies the join dependency if it is equal to the join o $I=\pi x_1(I)\bowtie \ldots \bowtie \pi x_n(I)$ f its projections on the sets of attributes $X_1,\ldots\ldots,X_n$. X1,……,Xn.

Difference between 4NF and 5nf:-

Fourth normal form requires that a table be BCNF and contain no multi-valued dependencies. 4NF solves the problem of multivalued dependencies, which we would encounter with our above Sales table when a customer purchased multiple products with a single order – we would have numerous rows that were dependent on one another without any clear definition of that dependency in the table.

Fifth normal form, also known as join-projection normal form (JPNF), states that no non-trivial join dependencies exist. 5NF states that any fact should be able to be reconstructed without any anomalous results in any case, regardless of the number of tables being joined. A 5NF table should have only candidate keys and its primary key should consist of only a single column.

Example:

Branch_staff_owner		
Branch_No	S_Name	O_Name
B003	AnnBeech	Carol Farrel
B003	David Ford	Carol Farrel
B003	AnnBeech	TinaMurphy
B003	David Ford	TinaMurphy

Branch_staff_owner	
Branch_No	S_Name
B003	AnnBeech
B003	David Ford

Branch_staff_owner	
Branch_No	O_Name
B003	Carol Farrel
B003	TinaMurphy

(b) How are data marts different from data warehouse? Explain the different types of data marts.

Ans. Data Warehouse:

- Holds multiple subject areas
- Holds very detailed information
- Works to integrate all data sources
- Does not necessarily use a dimensional model but feeds dimensional models.

Data Mart:

- Often holds only one subject area- for example, Finance, or Sales
- May hold more summarised data (although many hold full detail)
- Concentrates on integrating information from a given subject area or set of source systems
- Is built focused on a dimensional model using a star schema.

There are two types of data marts:

1. Independent or stand-alone data mart and
2. Dependent data mart.

Stand-alone data mart

A Stand-alone data mart focuses exclusively on one subject area and it is not designed in an enterprise context.

Dependent data mart

According to Bill Inmon, a dependent data mart is a place where its data comes from a data warehouse. Data in a data warehouse is aggregated, restructured, and summarised when it passes into the dependent data mart.

(c) Explain, Business Intelligence in context of data warehousing.

Ans. A data warehouse is an integrated collection of data and can help the process of making better business decisions. Several tools and methods are available to that enhances advantage of the data of data warehouse to create information and knowledge that supports business decisions. Two such techniques are-

- Decision-support systems
- Online analytical processing

Decision-support systems- The DSS is a decision support system and NOT a decision-making system. DSS is a specific class of computerised information systems that support the decision-making activities of an organisation. A properly designed DSS is an interactive software based system that helps decision makers to compile useful information from raw data, documents, personal knowledge, and/or business models to identify and solve problems and make decisions.

Online analytical processing- It is an approach for performing analytical queries and statistical analysis of multidimensional data. OLAP tools can be put in the category of business intelligence tools along with data mining. Some of the typical applications of OLAP may include reporting of sales projections, judging the performance of a business, budgeting and forecasting etc.

OLAP tools require multidimensional data and distributed query-processing capabilities. Thus, OLAP has data warehouse as its major source of information and query processing.

Q5. Explain the following with the help of examples or illustration.

(a) Postages SQL

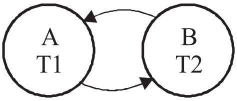
(b) Deadlock Recovery

(c) Semantic Query Optimisation

(d) Spatial and Multimedia Databases

Ans. (a) **Postgre SQL** is a decision many must make when approaching open-source relational database management systems. It is time-proven solutions that compete strongly with proprietary database software. PostgreSQL was assumed to be a more densely featured database system often described as an open-source version of Oracle. PostgreSQL can compress and decompress its data on the fly with a fast compression scheme to fit more data in an allotted disk space. The advantage of compressed data, besides saving disk space, is that reading data takes less IO, resulting in faster data reads.

(b) **Deadlock Recovery:** A set of two or more processes are deadlocked if they are blocked (i.e., in the waiting state) each holding a resource and waiting to acquire a resource held by another process in the set. A process is deadlocked if it is waiting for an event which is never going to happen.



Deadlock depends on the dynamics of the execution. All of the following four necessary conditions must hold simultaneously for deadlock to occur:

Mutual exclusion: only one process can use a resource at a time.

Hold and wait: a process holding at least one resource is waiting to acquire additional resources which are currently held by other processes.

No preemption: a resource can only be released voluntarily by the process holding it.

circular wait: a cycle of process requests exists (i.e., P0 is waiting for a resource hold by P1 who is waiting for a resource held by Pj ... who is waiting for a resource held by P(n-1) which is waiting for a resource held by Pn which is waiting for a resource held by P0). Circular wait implies the hold and wait condition. Therefore, these conditions are not completely independent.

(c) Semantic Query Optimisation: Semantic query optimisation is the process of transforming a query issued by a user into a different query which, because of the semantics of the application, is guaranteed to yield the correct answer for all states of the database. While this process has been successfully applied in centralised databases, its potential for distributed and heterogeneous systems is enormous, as there is the potential to eliminate inter-site joins which are the single biggest cost factor in query processing. Further justification for its use is provided by the fact that users of heterogeneous databases typically issue queries through high-level languages which may result in very inefficient queries if mapped directly, without consideration of the semantics of the system.

(d) Spatial and Multimedia Databases: A spatial database is a database that is optimised to store and query data that represents objects defined in a geometric space. Most spatial databases allow representing simple geometric objects such as points, lines and polygons. Some spatial databases handle more complex structures such as 3D objects, topological coverages, linear networks, and TINs. While typical databases are designed to manage various numeric and character types of data, additional functionality needs to be added for databases to process spatial data types efficiently. These are typically called geometry or feature. The Open Geospatial Consortium created the Simple Features specification and sets standards for adding spatial functionality to database systems.

A multimedia database is a system able to store and retrieve objects made up of text, images, sounds, animations, voice, video, etc. The wide range of applications for MMDBs leads to a number of different problems with respect to traditional database systems, which only consider textual and numerical attributes.

These problems include:

- data modeling;
- support for different data types;
- efficient data storing;
- data compressing techniques;
- index structures for non-traditional data types;
- query optimisation;
- presentation of objects of different types.

MCS – 43: ADVANCED DATABASE DESIGN
December, 2014

Note: *Question number* ***one*** *is* ***compulsory****. Answer* ***any three*** *questions from the rest.*

Q1. (a) How do UML diagrams help in designing the database? Discuss with the help of an example.
(b) How does data granularity affect the performance of concurrency control? Do you think that data granularity and database security are interrelated? Justify your answer.
(c) Compare and contrast the distributed DBMS environment with the centralised DBMS environment.
(d) What are semantic databases? List the features of semantic databases. Explain the process of searching the knowledge in these databases.
(e) What is shadow paging? Give two advantages and two disadvantages of shadow paging.
(f) What is a data warehouse? Describe the process of ETL for a data warehouse.
(g) What are data marts? Briefly discuss the method of creating the data marts.
(h) Explain the role of Query Optimiser in Oracle.

Q2. (a) Explain the algorithm and cost calculation for Simple Hash Join.
(b) Differentiate between the following:
(i) Embedded SQL and Dynamic SQL
(ii) XML and HTML
(iii) 2 PC and 3 PC Protocol
(iv) Data Warehousing and Data Mining

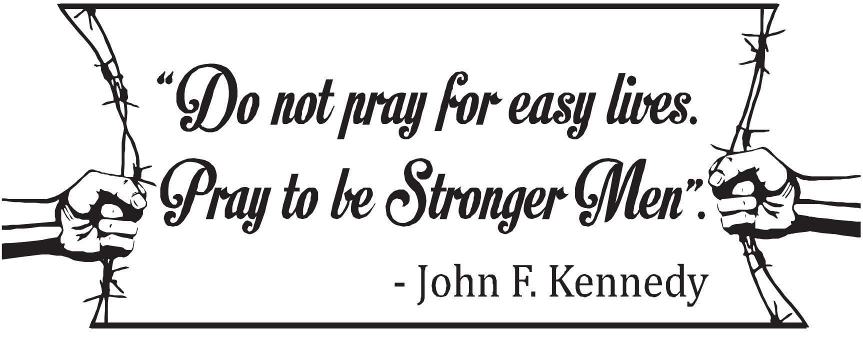
Q3. (a) Give suitable example to discuss the Apriori algorithm for finding frequent itemsets.
(b) Write a short note, with suitable example, for each of the following:
(i) Vendor-Specific Security
(ii) Multilevel Security

Q4. (a) What are cursors, stored procedures and triggers? Give SQL syntax for each and discuss the utility aspect of each.

(b) Explain Join-Dependency with the help of an example. With which normal form is it associated? Functional dependency and Multivalued dependency are special types of join dependencies. Justify.

Q5. (a) What do you mean by Deadlock? How can we prevent Deadlock? Write an algorithm that checks whether the concurrently executing transactions are in deadlock or not.

(b) Compare and contrast Relational DBMS with Object-Relational DBMS and Object-Oriented DBMS. Suggest one application for each of these DBMS.



MCS – 43: ADVANCED DATABASE DESIGN
June, 2015

Note: *Question number* ***one*** *is* ***compulsory****. Answer* ***any three*** *questions from the rest.*

Q1. (a) What is multivalued dependency? How is 4 NF related to multivalued dependency? Is 4 NF decomposition dependency preserving? Justify your answer.

(b) Differentiate between Assertions and Views. Discuss each with a suitable example.

(c) What is UML? How does UML have an edge over the other database designing tools? Describe database designing by using a UML class diagram.

(d) What do you mean by the term database transaction? Briefly discuss the properties of database transactions performed in a concurrent environment.

(e) Describe the reference architecture of a distributed DBMS, with the help of a block diagram.

(f) How does database differ from a data warehouse? How does processing in database differ from that in data warehouse?

(g) What is Datagrid? Describe the structure of datagrid, with the help of a block diagram. Why is the datagrid required?

(h) What do you mean by tuning of SQL? Briefly discuss each step involved in the tuning of SQL.

Q2. (a) Describe the phases of Query processing by using a block diagram. Discuss the process of Query optimisation, by using suitable example.

(b) Compare and contrast the following:

(i) Inclusion dependencies and Template dependencies

(ii) XML and HTML

(iii) Centralised 2PL and Distributed 2PL

(iv) K-Means clustering and Nearest Neighbour clustering

Q3. (a) Discuss classification as a tool in Data Mining. Describe the ID 3 algorithm for classifying data sets with a suitable example.

(b) What do you mean by Multiversioning? What are the various schemes available for multiversioning? Describe any two schemes in detail.

Q4. (a) How does PostgreSQL perform storage and indexing of tables? Briefly discuss the types of indexes involved in PostgreSQL.

(b) What is SQLJ? What are the requirements of SQLJ? Briefly describe the working of SQLJ. "Can SQLJ use dynamic SQL?" If yes, then how? Otherwise, specify the type of SQL if can use.

Q5. (a) Write short notes on any two of the following:

(i) JDBC

(ii) PJNF

(iii) Spatial Databases

(b) What do you mean by Multilevel Security? Discuss the methods involved in the support of multilevel security.

(c) What is Audit trail? Give four benefits provided by Audit trail in the context of DBMS.

(d) Discuss the constraints associated with Generalisation in an E-R Model.

Whenever you do
a thing, act as if all
the world watching.
- Thomas Jefferson

MCS – 43: ADVANCED DATABASE DESIGN
December, 2015

***Note:** Question number **one** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) Consider the following situation:
A subject can be taught by several teachers. The subject has defined textbooks (may be one or more). The following represents this information:

Course	Can be taught by	Textbooks to be used
MCS-01	• Mr. X • Mr. Y	• C.J. Date book • IGNOU material • Korth book
MCS-02	• Mr. A • Mr. B	• Fundamentals • IGNOU material

Perform the following tasks for the above:
(i) Represent the data shown above in a relation Course_Teacher_Text (Course, Teacher, Text).
(ii) Identify the FDs and MVDs in the relation created as above. Also identify the primary key.
(iii) Normalise the relation into 4NF.
(iv) Show that the normalisation of relations as above will result in lossless decomposition.
(b) Consider the following relations:
Student (enrolmentno, name, programme)
Result (enrolmentno, subjectcode, marks)
Consider the query on these relations:
"List the enrolment number, name, subject code and marks of a student whose enrolment number is "120000111"."
(i) Represent the query using SQL.
(ii) Convert the SQL query into equivalent relational algebraic query.
(iii) Draw the query tree for the above relational algebraic query.
(iv) Using the query tree, transform the relation algebraic query to an equivalent relation algebraic query, which may reduce the query evaluation cost.
(c) Differentiate between object relational database management system and object oriented database management system.
(d) What is a star schema in the context of a data warehouse? Explain with the help of an example.

(e) Explain the use of JDBC with the help of an example. Compare JDBC with ODBC.

(f) Explain the data integrity and triggers in Oracle database management system.

Q2. (a) A University has many Professors. Some of these Professors are full time employees, whereas some of them are visiting faculties. Every Professor teaches one or more courses. University stores the name, qualification and experience of every Professor. In case the Professor is a full time Professor his/her date of employment is stored. In case the Professor is a visiting Professor his/her host institution's name is recorded. Every course of the University has a course name, credits and course code.

(i) Draw an EER diagram for the University as defined above.

(ii) Convert the EER diagram to equivalent relations having proper keys and constraints.

(b) Explain any two security failures that are possible in a database system. What is meant by table access control in the context of database security? Explain two SQL commands that can be used for access control with the help of examples.

(c) List any four transaction management features of PostgreSQL.

Q3. (a) Consider the following transactions:

T1: Transfer ₹5,000 from Account A to Account B. You may assume that the required amount is available in Account A.

T2 : Add an amount ₹1,000 as interest in both the accounts-Account A and Account B.

(i) Write the pseudocode of the two transactions. You must assume read-before-write protocol.

(ii) What are the different problems that can occur if both the transactions are executed concurrently?

(iii) Rewrite the pseudocode of transactions including Lock and Unlock statements to ensure that the transactions execute without any concurrency related problem.

(b) What is Data Mining? Why is it useful? What is meant by classification in data mining? Explain with the help of an example.

(c) List the features of the following database models:

(i) Mobile databases

(ii) Multimedia databases

Q4. (a) Explain the utility of XML with the help of an XML document that shows the list of customers of an organisation. You need to store customer name, current address and one or more phone numbers for each customer. Create the customer list with at least two customers. Also create the DTD that verifies the created XML document.

(b) What is meant by the term "View" in the context of Database Management System?

Why is it used?

Consider a student database:

Student (enrolmentno, name, programme)

Course (course_code, course_name)

Registration (enrolmentno, course_code)

Create a view for a student whose registration number is "101" and shows the list of all the courses registered by him/her. The list should show the course_code and course_name registered by the student whose enrolment number is "101".

(c) Define the following giving their purpose:

(i) Object definition language

(ii) Semi-structured data

(iii) Data Mart

Q5. Explain each of the following with the help of an example/diagram, if needed:

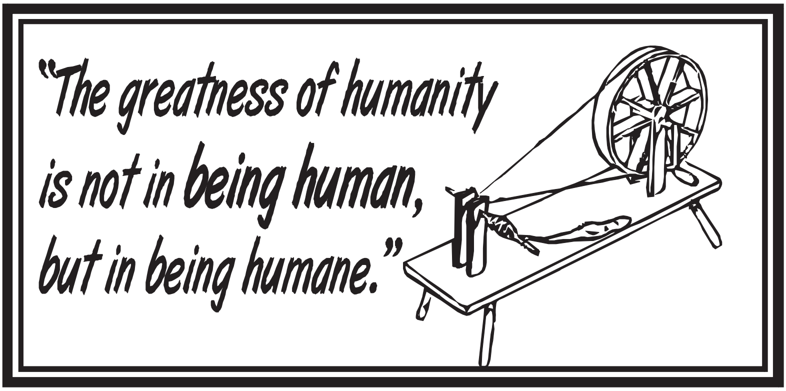
(a) Knowledge database

(b) Data Dictionary

(c) Data Replication in distributed database

(d) Properties of transactions

(e) Checkpoints in the context of database recovery



MCS – 43: ADVANCED DATABASE DESIGN
June, 2016

***Note:** Question number **one** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) What are Triggers? Explain the significance of triggers with the help of an example.
(b) Explain the Log-Based recovery scheme, using deferred database modification approach. Give a suitable example in your explanation.
(c) What are mobile databases? List the characteristics of mobile databases. Discuss the challenges of implementing mobile databases.
(d) List the index implementation available in PostgreSQL. Explain each index available in PostgreSQL.
(e) What is Data Dictionary? What is the significance of creating Data Dictionary in DBMS? Explain the statistics stored in the Data Dictionary.
(f) What are Database Security features? Discuss with example how Oracle manages database security.
(g) What are web databases? How do you create them? List the steps.
(h) What are views? What is the significance of views in DBMS? How are views managed in SQL? Explain with an example.

Q2. (a) How do Database Queries differ from Data Mining Queries? Explain K-Means Clustering algorithm for Data Mining, with suitable example.
(b) Explain any two of the following:
(i) OLAP and its types
(ii) Spatial Databases
(iii) Dynamic SQL
(c) Explain Data fragmentation in distributed DBMS, with the help of an example.

Q3. (a) Discuss the term 'Association rule mining'. Write the Apriori algorithm for association rule mining. Illustrate it using an example.
(b) Explain any two of the following with suitable examples:
(i) Semantic Databases
(ii) Temporal Databases
(iii) Embedded SQL
(c) Discuss 4NF with suitable example. Is 4NF decomposition dependency preserving? Justify.

Q4. (a) In PostgreSQL, explain how B-Tree Indexes are different from R-Tree Indexes.

(b) Discuss any three of the following with suitable examples:

(i) UML Class Diagram

(ii) Query Optimization

(iii) Assertion

(iv) Multiversioning

Q5. (a) What are checkpoints? How are checkpoints used in log-based recovery? Explain with the help of an example and a diagram.

(b) What are control files in Oracle? What does a control file contain? What is the role of control file in starting an instance of Oracle? Discuss the role of redo log file in starting an instance of Oracle.

(c) Describe ID3 algorithm for classifying datasets, with suitable example.

> I claim that human mind or
> human society is not divided into
> watertight compartments called
> social, political and religious.
> All act and react upon
> one another.
>
> *Mahatma Gandhi*

MCS – 43: ADVANCED DATABASE MANAGEMENT SYSTEMS
December, 2016

***Note:** Question number **1** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) What is a Data warehouse? How is it different from a Database? What are the characteristics of a Data warehouse?

(b) What are multimedia databases? Discuss the challenges in the designing of multimedia databases.

(c) How does Distributed DBMS differ from Centralized DBMS? Explain the reference architecture of Distributed DBMS.

(d) Discuss the term multilevel security in DBMS. What are the typical security levels?

(e) Explain 3-phase commit protocol. Why do we need 3-phase commit protocol in Distributed DBMS?

(f) What is meant by the term 'ETL'? Discuss the transformations required during the ETL process.

(g) Explain shadow paging recovery scheme, with the help of a diagram.

(h) What is a Datagrid? Elaborate the structure of a datagrid? What are the application areas of a datagrid?

Q2. (a) What do you understand by the term 'Database Tuning'? What are the goals of tuning a database? What is the requirement of tuning the database queries?

(b) What is multiversion concurrency control? How can multiversion concurrency control be achieved by using time stamp ordering?

(c) How do you perform cost calculation for simple Hash-Join?

Q3. (a) What are mobile databases? Discuss the characteristics of mobile databases. Give an application of mobile databases.

(b) Explain the following:

(i) PJNF

(ii) SQLJ

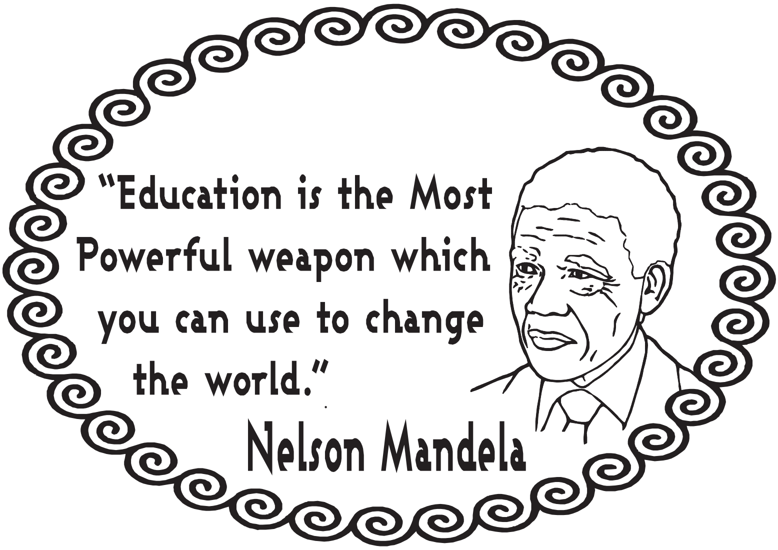
(iii) Audit Trail

(iv) Deductive Database

(c) What do you understand by 'Transaction' in DBMS? Discuss the properties of concurrent transactions performed in DBMS.

Q4. (a) Differentiate between following:
(i) XML and HTML
(ii) JDBC and ODBC
(iii) Clustering and Classification
(iv) Serial schedule and Serializable schedule
(b) Explain “log based” recovery using immediate database modification scheme. Give suitable example for explanation.
(c) Discuss the concept of exclusive locks and shared locks. How are these two locks used in transaction management? Discuss with an example.

Q5. (a) Define the term Data mining. How is data mining different from Knowledge Discovery in Databases (KDD)? Explain the steps of KDD with the help of an example.
(b) How are object oriented databases different from relational databases? Write an ODL syntax to create a class Employee that also references the Project class.
(c) Explain the role of Query optimizer in Oracle.
(d) What are web databases? Discuss the steps required in creating a web database.



MCS – 43: ADVANCED DATABASE MANAGEMENT SYSTEMS
June, 2017

***Note:** Question number **1** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) Consider the following relations:

Customer (c_id, cust_name, cust_phone)

Purchase (c_id, item_code, quantity)

Consider the query "List the c_id, cust_name, cust_phone, item_code and quantity of all the items purchased by the customer whose c_id = "C001"."

Perform the following tasks for the above:

(i) Write the above query using relational algebra and draw the query tree for the same.

(ii) Transform the query tree to an equivalent query tree such that the query evaluation cost may be reduced.

(b) What is join dependency? When is join dependency considered as trivial? Explain with suitable example.

(c) How does embedded SQL differ from dynamic SQL? Give an example for each.

(d) Discuss the term "ETL". List the transformations required to perform the ETL process.

(e) What is OLAP? How does OLAP support query processing in a data warehouse?

(f) Differentiate between XML schema and Document type definition. Give suitable examples for each.

(g) Construct an EER diagram for the following description:

"A University maintains records of its students and the programmes in which they have enrolled. It stores student id, name, address, and phone number of a student and programme code, programme name and duration of a programme. A student is either a full time student or a part time student (only one of the types). A student can register for many programmes and a programme can have many students."

(h) Write the SQL commands for giving permission to a user named "DBM" for creating new tables. Also write the command if the above permission is to be removed.

Q2. (a) What are Multivalued dependencies? When can we say that multivalued dependency is trivial? Discuss with a suitable example.

(b) List the various UML diagrams. How do UML class diagrams contribute to Database Design?

(c) What is a system catalogue? Discuss the role of system catalogues in Database Administration.

Q3. (a) What is semi-structured data? Explain with an example. How does a valid XML document differ from a well-formed XML document?

(b) What is Data Warehousing? Discuss the various characteristics of Data Warehousing.

(c) What is query optimization? Why is a query expressed in relational algebra preferred over a query expressed in SQL? What are the factors that contribute to the cost of a query?

Q4. (a) Differentiate between the following:

(i) Spatial Databases and Temporal Databases

(ii) 2-phase commit and 3-phase commit

(b) What is a datagrid? What is the utility of a datagrid? Draw a block diagram to describe the structure of a datagrid.

(c) How do clustering and classification differ? Describe Bayesian classification, with suitable example.

Q5. Explain any *five* of the following:

(a) K-mean Clustering

(b) SQLJ and its requirements

(c) Statistical Database Security

(d) Buffer Management

(e) Data Marks

(f) GNOME Databases

(g) JDBC

MCS – 43: ADVANCED DATABASE MANAGEMENT SYSTEMS
December, 2017

***Note:** Question number **1** is **compulsory**. Answer any **three** questions from the rest.*

Q1. (a) How does Data-mining differ from the technique of Knowledge Discovery in Database (KDD)? Can we use these two techniques interchangeably or alternatively? Justify.

Refer to Ch-6, Q.No.-51(ii), Page No.-133 & Dec.-2009, Q.No.-1(d), Page No.-244

(b) Explain the index implementations available in PostgreSQL, with suitable examples.

Refer to Ch-7, Q.No.-27, Page No.-148

(c) Consider the following data:

Employee Code (EC)	Project Code on which employee works (PC)	Tools that employee can work on (TC)
E1	P1 P2	T1 T2
E2	P3 P4	T2 T3

Employee with code E1 works on two projects, P1 and P2, and can use two tools T1 and T2. Likewise, E2 works on projects P3 and P4 and can use tools T2 and T3. You may assume that projects and tools are independent of each other, but depend only on employee.

Perform the following tasks for the data:

(i) Represent the data as above in a relation, namely, Employee_Project_Tool (EC, PC, TC).

(ii) List all the MVDs in the relation created in part (i) above. Also identify the primary key of the relation.

(iii) Normalise the relation created in part (i) into 4NF.

(iv) Join the relations created in part (iii) above and show that they will produce the same data on joining as per the relation created in part (i).

Same as June-2013, Q.No.-1(a), Page No.-302

(d) What is Granularity in databases? How does granularity relate to the security of databases? In a concurrent environment, how does granularity affect the performance?

Ans. Granularity could be defined as any entity whose data fields could be sub divided.

For e.g. when you take an entity called as a person. A person's identity could be further divided into following:

- Name, Address, Gender, City, State, Country etc.
- This means that a person as an entity has high granularity. Since there are many sub divisions of the same entity.
- But if we were to choose gender-this could simply be maximum 3 values-
- Male, Female and Transgender
- This means that Gender as a field/attribute has low granularity.

Now, Refer to June-2008, Q.No.-1(f), Page No.-217

(e) What are Semantic databases? Give the features of semantic databases. Discuss the process of searching the knowledge in these databases.

Refer to June-2008, Q.No.-2(c), Page No.-219

(f) Illustrate the concept of Shadow Paging with suitable example. Give the advantages and disadvantages of shadow paging.

Refer to June-2008, Q.No.-5(a), Page No.-223

(g) How does Clustering differ from Classification? Briefly discuss one approach for both, i.e., clustering and classification.

Refer to Ch-6, Q.No.-54, 56, Page No.-134 & Dec-2006, Q.No.-4(a), Page No.-174

Q2. (a) What are Alerts, Cursors, Stored Procedures and Triggers? Give the utility of each. Explain each with suitable code of your choice.

Refer to Ch-5, Q.No.-36, Page No.92

(b) What is Simple Hash Join? Discuss the algorithm and cost calculation process for simple hash join. Explain how Hash Join is applicable to Equi Join and Natural Join.

Refer to June-2008, Q.No.-1(d), Page No.-213

Q3. (a) Differentiate between the following:

(i) Database Queries and Data-mining Queries

Refer to Ch-6, Q.No.-50, Page No.-132

(ii) Star Schema and Snowflake Schema

Give example for each while differentiating.

Refer to Dec-2009, Q.No.-5(a), Page No.253

(b) What are Deadlocks? How are they detected? Explain with the help of an example.

Refer to Ch-5, Q.No.-21, Page No.79

(c) What are Mobile Database? Explain the characteristics of mobile databases. Give an application of mobile databases.

Refer to Ch-7, Q.No.-10, 14, Page No.-144, 145 & Dec.-2012, Q.No.-3(a), Page No.298

Q4. (a) What are Views in SQL? What is the significance of views? Give an example code in SQL to create a view.

Refer to Ch-4, Q.No.-1, Page No.-48

(b) (i) Create an XML document that stores information about students of a class. The document should use appropriate tags/attributes to store information like student_id (unique), name (consisting of first name and last name), class (consisting of class no. and section) and books issued to the student (minimum 0, maximum 2). Create at least two such student records.

Refer to Gullybaba.com (Download section)

(ii) Create the DTD for verification of the above student data.

Refer to Gullybaba.com (Download section)

(c) Explain the features and challenges of multimedia databases.

Refer to Ch-7, Q.No.-1, 4, Page No.-140, 142

Q5. (a) What is Data Dictionary? Give the features and benefits of a data dictionary. What are the disadvantages of a data dictionary?

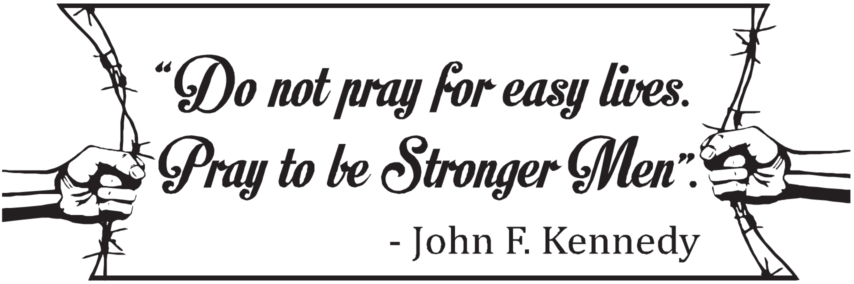
Refer to Ch-4, Q.No.-18, 19, 20, Page No.-64, 65

(b) What is the utility of Multi-Version schemes in databases? Discuss any one multi-version scheme, with suitable example.

Refer to June.-2007, Q.No.-5(a), Page No.196

(c) What is Association Rule Mining? Write Apriori Algorithm for finding frequent itemset. Discuss it with suitable example.

Refer to Ch-6, Q.No.-59, 60, Page No.-137, 138



MCS – 43: ADVANCED DATABASE DESIGN
June, 2018

***Note:** Question number **one** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) What are Cursors, Stored Procedures and Triggers? Explain each with the help of an example code.
(b) Differentiate between Logical and Physical Database design.
(c) What is a View? Explain any two strategies for implementing the Views.
(d) What is Hash join? How is Hash join between two relations computed? Explain the algorithm and cost calculation for simple hash join.
(e) Differentiate between a database management system and a data warehouse. What is the need for a data warehouse?
(f) What is granularity of data? How does granularity of data items affect the performance of concurrency control?
(g) Differentiate between Centralised DBMS and Distributed DBMS.
(h) What is Timestamp Ordering? Explain timestamp based protocol for serialisable schedule.

Q2. (a) What is Data Mining? How does Data Mining differ from OLTP? Discuss Classification as a technique for data mining.
(b) What are Data Marts? Briefly discuss the significance of data marts.
(c) What is XML? How does XML differ from HTML? What are the advantages of XML ? Create an XML schema for a list of students and their marks.

Q3. (a) Differentiate between Two-phase commit protocol and Three-phase commit protocol in distributed databases. “The three-phase commit protocol increases the system’s availability and does not allow transactions to remain blocked until a failure is repaired.” Justify the statement.
(b) What are Semantic Databases? List the features of semantic databases. Explain the process of searching the knowledge in semantic databases.
(c) What is a Data Dictionary? List some, features of a data dictionary. What are the various approaches to implement a Distributed Database Catalogue?

Q4. (a) What is Shadow Paging? Illustrate with an example. Give the advantages and disadvantages of shadow paging.

(b) Define Multi-valued Dependency. What is Trivial Multi-valued Dependency? State the fourth normal form.

(c) Explain any one clustering technique for data mining.

(d) What is Query Optimisation? Briefly discuss the techniques of Query Optimisation with suitable examples.

Q5. (a) Explain the following:

(i) Dynamic SQL

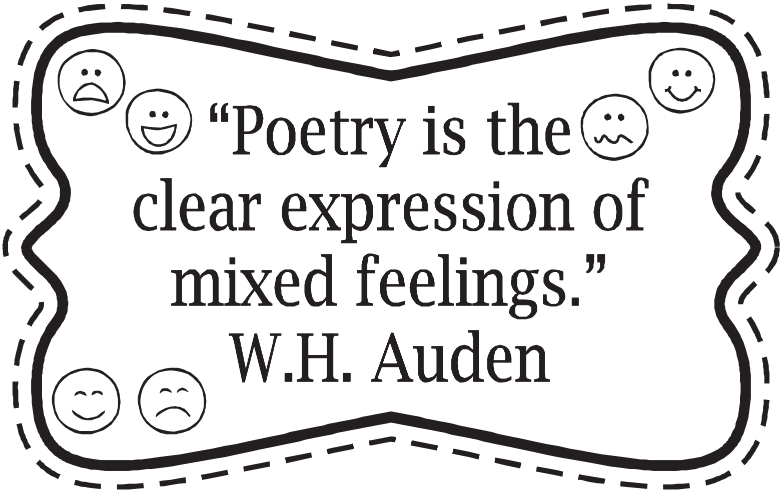
(ii) OLAP

(iii) Spatial Databases

(iv) Temporal Databases

(b) What are Mobile Databases? List the characteristics and challenges of mobile databases.

(c) Explain Join Dependency with the help of an example. To which normal form does it correspond? "Functional Dependencies and Multivalued Dependencies are special types of Join Dependencies." Justify the statement.



MCS – 43: ADVANCED DATABASE DESIGN
December, 2018

***Note:** Question number **one** is **compulsory**. Answer **any three** questions from the rest.*

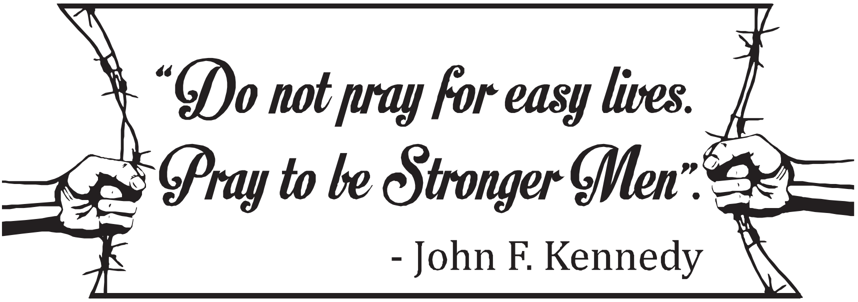
Q1. (a) Explain the process of Query optimisation with suitable example.
(b) What is the difference between Document Type Definition (DTD) and XML Schema? Explain using an example.
(c) Explain Data mining in the context of knowledge discovery in databases.
(d) What is Join dependency? Explain it with the help of an example. What is trivial join dependency'?
(e) Consider a small institute in which students register for programmes run by the institute. A programme can be a full or part time programme or both. Every student necessarily registers in at least one programme and at most three programmes, Assuming suitable attributes, design an EER Diagram for the same.
(f) Explain the reference architecture for distributed database management system.
(g) What are triggers? Explain the utility of triggers in DBMS. Give suitable SQL code for triggers.
(h) What is a System catalogue? What is the information stored in catalogue of RDBMS?

Q2. (a) Compare and contrast the following:
(i) JDBC and ODBC B-Tree Indexes and R-Tree Indexes used in Postgre SQL.
(b) What is multiversion two-phase locking? Explain with an example.
(c) What are the different type of security features, needed for a multilevel security system? Explain the encryption technique for a multilevel security system.

Q3. (a) Describe the following with suitable example or a diagram:
(i) Data Grid
(ii) Data Mart
(iii) Deadlock
(iv) Checkpoint
(v) Referential Integrity Constraint
(b) What are views? What is their significance in DBMS? How are views created in SQL? Explain it with the help of an SQL statement.

Q4. (a) Differentiate between the following:
(i) Centralised and Distributed Databases
(ii) Clustering and Classification approaches in Data Mining
(iii) Star Schema and Snowflake Schema
(b) What are deadlocks? How are deadlocks detected? Explain with the help of an example.

Q5. (a) What is semi-structured data? Explain with the help of an example. What is the difference between a well formed XML document and a valid XML document?
(b) What is data warehousing? Discuss various characteristics of data warehousing?
(c) What are multimedia databases? Discuss the challenges of designing multimedia databases.
(d) What is multi-valued dependency? State the fourth normal form.



MCS – 43: ADVANCED DATABASE DESIGN
June, 2019

***Note:** Question number **1** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) What is a data warehouse? Describe the process of ETL for data warehouse.

Refer to Dec-2008, Q.No.-1(b), Page No.-227

(b) What is the significance of cursors in embedded SQL? Explain with the help of an example.

Refer to Dec-2008, Q.No.-2(a), Page No.-231

(c) What is OLAP ? How does OLAP differ from OLTP?

Refer to Dec-2006, Q.No.-5(iv), Page No.-179

Parameters	OLTP	OLAP
Process	It is an online transactional system. It manages database modification.	OLAP is an online analysis and data retrieving process.
Characteristic	It is characterised by large numbers of short online transactions.	It is characterised by a large volume of data.
Functionality	OLTP is an online database modifying system.	OLAP is an online database query management system.
Method	OLTP uses traditional DBMS.	OLAP uses the data warehouse.
Query	Insert, Update, and Delete information from the database.	Mostly select operations
Table	Tables in OLTP database are normalised.	Tables in OLAP database are not normalised.
Source	OLTP and its transactions are the sources of data.	Different OLTP databases become the source of data for OLAP.
Data Integrity	OLTP database must maintain data integrity constraint.	OLAP database does not get frequently modified. Hence, data integrity is not an issue.
Response time	It's response time is in millisecond.	Response time in seconds to minutes.
Data quality	The data in the OLTP database is always detailed and organise.	The data in OLAP process might not be organised.
Usefulness	It helps to control and run fundamental business tasks.	It helps with planning, problem-solving, and decision support.
Operation	Allow read/write operations.	Only read and rarely write.
Audience	It is a market orientated process.	It is a customer orientated process.
Query Type	Queries in this process are standardised and simple.	Complex queries involving aggregations.
Back-up	Complete backup of the data combined with incremental backups.	OLAP only need a backup from time to time. Backup is not important compared to OLTP

Design	DB design in application oriented. Example: Database design changes with industry like Retail, Airline, Banking, etc.	DB design is subject oriented, Example: Database design changes with subjects like sales, marketing, purchasing, etc.
User type	It is used by Data critical users like clerk, DBA & Data Base professionals.	Used by Data knowledge users like workers, managers, and CEO.
Purpose	Designed for real time business operations.	Designed for analysis of business measures by category and attributes.
Performance metric	Transaction throughput is the performance metric	Query throughput is the performance metric.
Number of users	This kind of Database users allows thousands of users.	This kind of Database allows only hundreds of users.
Productivity	It helps to increase user's self-service and productivity	Help to increase productivity of the business analysts.
Challenge	Data Warehouses historically have been a development project which may prove costly to build.	An OLAP cube is not an open SQL server data warehouse. Therefore, technical knowledge and experience is essential to manage the OLAP server.
Process	It provides fast result for daily used data.	It ensures that response to the query is quicker consistently.
Characteristic	It is easy to create and maintain.	It lets the user create a view with the help of a spreadsheet.
Style	OLTP is designed to have fast response time, low data redundancy and is normalised.	A data warehouse is created uniquely so that is can integrate different data sources for building a consolidated database.

(d) What are data marts? What is the significance of creating data marts?

Refer to Dec-2008, Q.No.-3(b), Page No.-231

(e) What in join dependency? Explain with an example trivial join dependency.

Refer to Dec-2008, Q.No.-4(b), Page No.-232

(f) What is the effect of data granularity over database security ? Give suitable example.

Refer to June-2008, Q.No.-1(f), Page No.-217

(g) What is the significance of creating views ? How is a view created using SQL ? Explain using an example.

Refer to Ch-4,(view definition), Page No.-49, 50

(h) How is hash-join of two relations r and s computed ?

Refer to Dec-2006, Q.No.-1(iii), Page No.-163

Q2.(a) Explain the terms lossless decomposition and dependency preserving decomposition. Consider a relation R (A , B, C, D, E, F) with functional dependency set

$$\mathbf{FD} = \{\mathbf{A \rightarrow BC, C \rightarrow A, D \rightarrow E, F \rightarrow A, E \rightarrow D}\}$$

If R is decomposed into R_1 (A, C, D); R_2 (B, C, D,); R_3 (E, F, D), then check whether the decomposition is both lossless and dependency preserving.

Refer to June-2014, Q.No.-1(a), Page No.-318

(b) How do distributed databases differ from the centralized databases? Describe the architecture of distributed databases with the help of a diagram.

Ans. The main differences between centralised and distributed databases arise due to their respective basic characteristics. Differences include but are not limited to:

- Centralised databases store data on a single CPU bound to a single certain physical/geographical location. Distributed databases, however, rely on a central DBMS which manages all its different storage devices remotely, as it is not necessary for them to be kept in the same physical and/or geographical location.
- As outlined above, centralised databases are easier to maintain up to date than distributed databases. This is so because distributed databases require additional (often manual) work to keep the data stored relevant, and to avoid data redundancy, as well as to improve the overall performance.
- If data is lost in a centralised system, retrieving it would be much harder. If, however, data is lost in a distributed system, retrieving it would be very easy, because there is always a copy of the data in a different location of the database.
- Designing a centralised database is generally much less complex than designing a distributed database, as distributed database systems are based on a hierarchical structure.

Now, Refer to June-2014, Q.No.-2(b), Page No.-325

Q3. (a) What is multivalued dependency ? How is multivalued dependency related to 4NF? Explain with suitable example.

Refer to June-2007, Q.No.-1(a), Page No.-181

(b) Justify the statement "BCNF is stronger than 3NF".
Refer to June-2014, Q.No.-1(d), Page No.-322

(c) Differentiate between star schema and snowflake schema.
Refer to Dec-2009, Q.No.-5(a), Page No.-253

(d) Differentiate between ODBC and JDBC. What are the components required for implementing ODBC in a system?
Refer to June-2009, Q.No.-3(b), Page No.-238 and Refer to Ch-7, Q.No.-17, Page No.-145

Q4. (a) What is k-means clustering ? How does it differ from nearest neighbour clustering ?
Ans. In the K-Means clustering, initially a set of clusters is randomly chosen. Then iteratively, items are moved among sets of clusters until the desired set is reached. A high degree of similarity among elements in a cluster is obtained by using this algorithm. For this algorithm a set of clusters $K_i=\{t_{i1},t_{i2},\ldots,t_{im}\}$ is given , the cluster mean is:
$m_i = (1/m)(t_{i1} + \ldots + t_{im})$
where t_i represents the tuples and m represents the mean
The K-Means algorithm is as follows:

Input:

```
D= {t_1,t_2,...t_n} //Set of elements
A                   //Adjacency matrix showing distance between elements.
k                   //Number of desired clusters.
```

Output:

```
K                   //Set of Clusters
```

K-Means Algorithm:

```
Assign initial values for means $m_1,m_2..m_k$;
Repeat
    Assign each item $t_i$ to the cluster which has the closest mean;
    Calculate new mean for each cluster;
Until convergence criteria is met.
```

Now, Refer to Dec-2006, Q.No.-4(a), Page No.-174

(b) What is Audit Trail ? Give benefits of audit trail in context of DBMS.
Refer to Ch-5, Q.No.-60, Page No.-103

(c) Discuss classification as a tool of data mining. Describe ID 3 algorithm for classifying datasets with a suitable example.

Ans. ID3 Algorithm for Classification

This algorithm creates a tree using the algorithm given below and tries to reduce the expected number of comparisons.

Algorithm: ID3 algorithm for creating decision tree from the given training data.

Input: The training data and the attribute-list.

Output: A decision tree.

Process:

Step 1: Create a node N

Step 2: If sample data are all of the same class, C (that is probability is 1) then return N as a leaf node labeled class C

Step 3: If attribute-list is empty then return N as a leaf node label it with the most common class in the training data; // majority voting

Step 4: Select split-attribute, which is the attribute in the attribute-list with the highest information gain;

Step 5: label node N with split-attribute;

Step 6: for each known value Ai, of split-attribute // partition the samples Create a branch from node N for the condition: split-attribute = Ai; // Now consider a partition and recursively create the decision tree:

Let xi be the set of data from training data that satisfies the condition:

split-attribute = A_i if the set xi is empty then attach a leaf labeled with the most common class in the prior set of training data; else attach the node returned after recursive call to the program with training data as xi and new attribute list = present attribute-list – split-attribute;

End of Algorithm.

Now, Refer to Ch-6, Q.No.-55, Page No.-134

Q5. (a) How does PostgreSQL perform storage and indexing? Discuss the types of indexes involved in PostgreSQL with suitable examples.

Refer to Ch-7, Q.No.-27, Page No.-148

(b) What is SQLJ? Give requirements of SQLJ. Discuss the working of SQLJ. "Can SQLJ use dynamin SQL?" If yes, then how? Otherwise, specify the type of SQL it can use.

Refer to Ch-4, Q.No.-11, Page No.-58

MCS – 43: ADVANCED DATABASE DESIGN
December, 2019

Note: *Question number **one** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) What are triggers? What is the utility of triggers in DBMS? Explain with the help of an example.
(b) Discuss data mining in the context of knowledge discovery in databases.
(c) What are mobile databases? Give characteristics of mobile databases.
(d) What is shadow paging? Give advantages and disadvantages of shadow paging.
(e) Differentiate between logical database design and physical database design.
(f) How hash join is applicable to Equi Join and Natural Join?
(g) What is an ETL process? Describe the different transformations performed during ETL process.
(h) Compare Assertions and Views. Discuss each with a suitable example.

Q2. (a) Differentiate between the following:
(i) Centralized 2 PL and Distributed 2 PL
(ii) XML and HTML
(iii) Inclusion dependencies and Template dependencies
(iv) Spatial databases and temporal databases
(b) What do you understand by the term "Multiversioning"? What are the various schemes available for multiversioning? Explain any two scheme.

Q3. (a) What is Join Dependency? How join dependency relates to SNF?
(b) What is a Datagrid? Give a block diagram to describe the structure of datagrid. What is the utility of datagrid?
(c) What is a serializable schedule? How does serializable schedule differ from a serial schedule? What are the problems, associated with both schedules? Explain the time stamp based protocols for serializable schedule.

Q4. (a) Why Database query expressed in relational algebra is preferred over the query expressed in SQL? Explain with suitable example.

(b) What are the common database security failures? Give SQL commands, to grant permission for database access. Why statistical databases are more prone to security breach? Explain with the help of an example.

(c) What is OLAP? Discuss the different implementations of OLAP.

(d) Discuss the different types of structural diagrams in UML.

Q5. (a) Explain Centralized two-phase commit protocol in Distributed Environment. Give the algorithm for both coordinator and participants.

(b) Write short notes on the following:

(i) OLTP

(ii) Embedded SQL

(iii) Dynamic SQL

(iv) Clustering approaches

(v) Semantic databases

MCS – 43: ADVANCED DATABASE DESIGN
June, 2020

Note: *Question number **one** is **compulsory**. Answer **any three** questions from the rest.*

Q1. (a) What is join dependency? When a join dependency is referred as trivial? Explain with the help of an example.

(b) What are views in DBMS? Discuss their significance in DBMS. How views are created in SQL with the help of an example.

(c) What is query optimization? Discuss the role of query optimization in Oracle.

(d) What is Audit Trail? Give four benefits of Audit trail, in context of DBMS.

(e) What is shadow paging? Give advantages and disadvantages of shadow paging.

(f) What are "deadlocks? How are they detected? Explain with the help of an example.

(g) What is XML? What are the advantages of XML? Create and XML schema for list of students and their marks.

(h) What are mobile databases? List the characteristics and challenges in implementing mobile database.

Q2. (a) Discuss the following with the help of a suitable example for each:

(1) Triggers

(ii) Alerts

(iii) Cursors

(iv) Procedures

(v) Functions

(b) Compare and contrast Database Queries and Data Mining Queries. Give example for each.

(c) Explain K-means algorithm with the help of an example.

Q3. (a) Differentiate between clustering and classification in data mining.

(b) Explain log based recovery scheme, using deferred database modification approach with the help of an example.

(c) What is multilevel security? What are typical security levels?

(d) What is Data Dictionary in DBMS? Discuss its significance in DBMS. Give an example of a data dictionary entry.

Q4. (a) Explain the following:
(i) Dynamic SQL
(ii) OLAP and its types
(iii) Spatial Database
(iv) Semantic Database
(b) What are B-Tree Indexes? How do B-Tree Indexes differ from R-Tree Indexes? Give example for each.
(c) Compare and contrast the following:
(i) JDBC and ODBC
(ii) ER Diagram and EER Diagram.

Q5. (a) What are Exclusive mode locks? How do Exclusive mode locks differ from shared mode locks? How are these locks used in transaction management? Give suitable example to support your discussion.
(b) What is multivalued dependency? How is 4NF related to multivalued dependency? Is 4NF decomposition dependency preserving? Justify your answer.
(c) Explain the role of the following files in Oracle:
(i) Control Files
(ii) Data Files