easy-solutions

Mumbai University Paper Solutions

Strictly as per the New Revised Syllabus (Rev - 2016) of
Mumbai University w.e.f. academic year 2018-2019
(As per Choice Based Credit and Grading System)

MU

DATABASE MANAGEMENT SYSTEM

Semester V - Computer Engineering /
Electronics Engineering (Department Level Elective-I)

Chapterwise Paper Solution upto May 2019.

TechKnowledge™ Publications

easy – solutions

MUMBAI

Database Management System

Semester V - Computer Engineering (CSC502), Electronics Engineering (Department Level Elective-I)

Strictly as per the New Revised Syllabus (Rev- 2016) of Mumbai University w.e.f. academic year 2018-2019

TechKnowledge™ Publications

EM044A

Database Management System

(Semester V – Computer Engineering (CSC502), Electronics Engineering (Department Level Elective-I)) (MU)

Edition 2019

This edition is for sale in India, Bangladesh, Bhutan, Maldives, Nepal, Pakistan, Sri Lanka and designated countries in South-East Asia. Sale and purchase of this book outside of these countries is unauthorized by the publisher.

Printed at : 37/2, Ashtvinayak Industrial Estate, Near Pari Company,
Narhe, Pune, Maharashtra State India,
Pune – 411041

Published by

TechKnowledge Publications

Head Office : B/5, First floor, Maniratna Complex, Taware Colony, Aranyeshwar Corner,
Pune - 411 009. Maharashtra State, India
Ph : 91-20-24221234, 91-20-24225678.
Email : info@techknowledgebooks.com,
Website : www.techknowledgebooks.com

(Book Code : EMO44A)

INDEX

Chapter 1	:	Introduction Database Concepts
Chapter 2	:	Database Architecture
Chapter 3	:	Entity Relationship Data Model
Chapter 4	:	Relational Data Model
Chapter 5	:	Relational Algebra
Chapter 6	:	Structured Query Language
Chapter 7	:	SQL Security
Chapter 8	:	Trigger
Chapter 9	:	Relational Database Design
Chapter 10	:	Transaction
Chapter 11	:	Concurrency Control
Chapter 12	:	Recovery System

Table of Contents

SYLLABUS

Database Management System

Module No.	Unit No.	Topics
1.0		**Introduction Database Concepts**
	1.1	Introduction, Characteristics of databases File system v/s Database system Users of Database system
	1.2	Data Independence DBMS system architecture Database Administrator
2.0		**Entity–Relationship Data Model**
	2.1	The Entity-Relationship (ER) Model : Entity types : Weak and strong entity sets, Entity sets, Types of Attributes, Keys, Relationship constraints : Cardinality and Participation, Extended Entity-Relationship (EER) Model : Generalization, Specialization and Aggregation.
3.0		**Relational Model and relational Algebra**
	3.1	Introduction to the Relational Model, relational schema and concept of keys. Mapping the ER and EER Model to the Relational Model
	3.2	Relational Algebra – unary and set operations, Relational Algebra Queries.
4.0		**Structured Query Language (SQL)**
	4.1	Overview of SQL Data Definition Commands, Data Manipulation commands, Data Control commands, Transaction Control Commands.
	4.2	Set and string operations, aggregate function - group by, having. Views in SQL, joins, Nested and complex queries, Integrity constraints :- key constraints, Domain Constraints, Referential integrity, check constraints.
	4.3	Triggers
5.0		**Relational - Database Design**
	5.1	Pitfalls in Relational-Database designs, Concept of normalization Function Dependencies, First Normal Form, 2nd, 3rd, BCNF, multi valued dependencies, 4NF.
6.0		**Transactions Management and Concurrency**
	6.1	Transaction concept, Transaction states, ACID properties Concurrent Executions, Serializability - Conflict and View, Concurrency Control : Lock-based, Timestamp-based protocols.
	6.2	Recovery System : Failure Classification, Log based recovery, ARIES, Checkpoint, Shadow paging. Deadlock handling

□□□

Database Management System

Chapter 1 : Introduction Database Concepts

Q. 1 Explain the detailed concept of DBMS. **(5 Marks)**

Ans. :

Database Management System (DBMS)

- A **Database Management System (DBMS)** is a collection of software or programs which help user in creation and maintenance of a database (set of information). Hence it is also known as a computerized record-keeping system.
- DBMS is the software system that helps in the process of defining, constructing, manipulating the database.

Examples :

- MS Access, Fox Pro by Microsoft.
- Oracle by Oracle corp.
- SQL Server By Microsoft.
- Ingres, DB2 by IBM.

Q. 2 Explain the features of DBMS. **(5 Marks)**

Ans. : The various characteristics of the DBMS are :

1. Data integrity

- Integrity constraints provide a way of ensuring that changes made in the database by authorized users that do not result in the loss of data consistency and correctness.
- Database integrity is concerned with the correctness and completeness of data in the database.
- This objective can never be guaranteed, one cannot ensure that every entry made in database is accurate.

2. Data security

- A DBMS system always has a separate system for security which is responsible for protecting database against accidental or intentional loss, destruction or misuse.
- Data in database should be given to only authorized users. Only authorized users should be allowed to modify data.
- Authorized users are able to access data any time he wants.

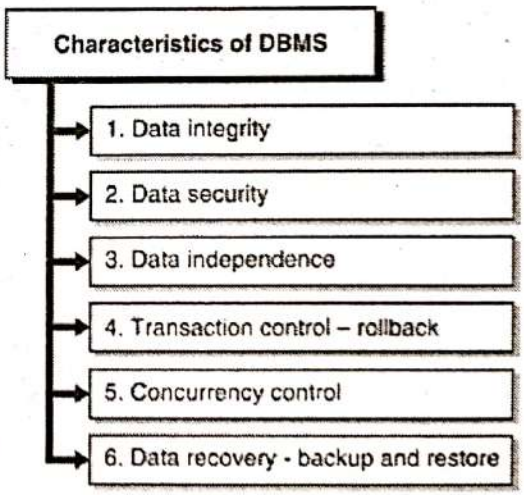


Fig. 1.1 : Characteristics of DBMS

Q. 3 Discuss different database Users. **(5 Marks)**

Ans. :

1. Naive users

- Naive users are users who interact with the system using application programs that have been developed previously.
- For example, Student wants to pay fees Rs.50 then accountant will invoke a program called fees_payment(). This program asks the accountant for the amount of fees to be paid.
- Naive users can read reports generated from the database.

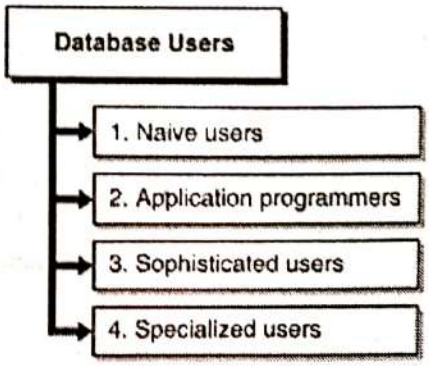


Fig. 1.2 : Database Users

2. Application programmers

- Application programmers responsible for writing application programs that use the database.
- Application programmers are developers or computer professionals who write application programs.
- Application programmers develop user interfaces using any preferred language.
- **Rapid Application Development (RAD)** tools are available nowadays that enable an application programmer to construct application without writing code.

3. Sophisticated users

- Sophisticated users interact with application without writing programs by using a database query language.
- This query will be solved by query processor.
- **Online Analytical Processing (OLAP)** tools is used to view summaries of data in different ways which helps analysts (e.g. sales of region, city etc.) with OLAP analysts can use data mining tools, which help them find certain kinds of patterns in data.

4. Specialized users

- Creates the actual database and implements technical controls needed to enforce various policy decisions.
- Specialized users are sophisticated users who develop database applications.

Chapter 2 : Database Architecture

Q. 1 Explain three-level architecture of DBMS. **(5 Marks)**

Ans. :

- The goal of the three-schema architecture is to separate the front end (user applications interface) and the back end (physical database).
- The three-schema architecture is a tool with which the user can visualize the schema levels in a DBMS. Many DBMS systems do not separate the three levels completely, but support the three schema architecture to some extent.

Database architecture

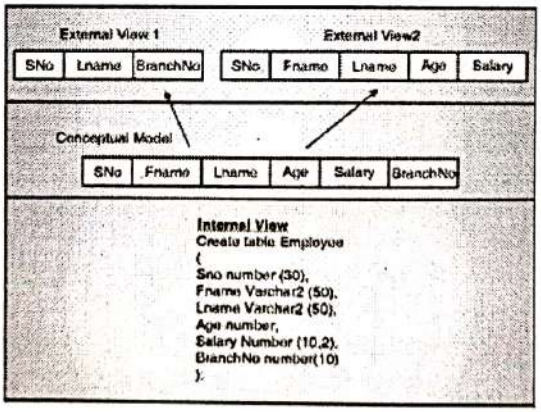


Fig. 2.1 : Database schema levels

(I) Internal Level (Physical Level)

- The internal level is very close to physical storage of data.
- The internal (or physical) database is stored on secondary storage devices, mainly the magnetic disk.
- Describes the complete details of data storage and various available access methods for the database.
  - At its ground level, it is stored in the form of bits with the physical addresses on the secondary storage device.
  - At its highest level, it can be viewed in the form of files and simple data structures.

Example :

```
Create table Employee
(   Sno         number (30),
    Fname       varchar2 (50),
    Lname       varchar2 (50),
    Age         number,
    Salary      number (10,2),
    BranchNo    number (10) );
```

(II) Conceptual level

- This level describes the structure of the whole database for a group of users.
- The conceptual model is also called as the data model or data model is used to describe the conceptual schema when a database system is implemented.
- The conceptual schema hides the internal details of physical storage and targets on describing entities, data types, relationships and constraints.

(III) External level (view level)

- The external level is the one closest to the user, i.e., it is related with the way data is viewed by individual end users.
- The external level includes a number of user views or external schemas.
- Each external schema describes the segment of the database that is required for a particular user group and hides the rest of the database from that user group.

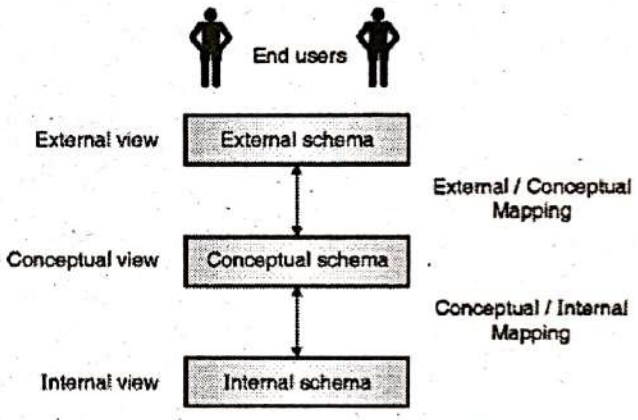


Fig. 2.2 : Three level schema architecture

(IV) Mapping

- The processes of transforming requests and results between various levels of architecture are called mappings.
- These mappings may be time-consuming, so small databases do not support external views.
- **External / conceptual mapping :** The DBMS must transform a request on an external schema into a request against the conceptual schema.
- **Conceptual / internal mapping :** A certain amount of mapping is necessary to transform requests between the conceptual and internal levels.

Q. 2 Define data independence and explain types of data independence. (5 Marks)

Ans. :

Definition : **Data Independence** can be defined as the capacity to change one level of schema without changing the schema at the next higher level.

Types

(a) Logical data independence

- Logical data independence is a capacity to change the conceptual schema without having any changes to external schemas. (or application programs)
- Separating the external views from the conceptual view enables us to change the conceptual view without affecting the external views. This separation is sometimes called logical data independence.

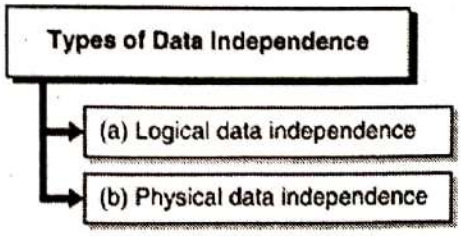


Fig. 2.3 : Types of Data Independence

Example :

Change the conceptual schema by removing a data item. In this case the external schemas that refer only to the remaining data should not be affected.

(b) Physical data independence

- Physical data independence is a capacity to change the internal schema without having any changes to conceptual schema.
- The separation of the conceptual view from the internal view enables us to provide a logical description of the database without the need to specify physical structures. This is often called physical data independence.

Example :

By creating additional access paths to improve the performance of retrieval. If the same data as before remains in the database, we should not have to change the conceptual schema.

Q. 3 Write a short note on : Responsibilities of Database Administrator. (3 Marks)

Ans. :

The various responsibilities of DBA are as follows,

- Designing overall Database schema
- The DBA selects the suitable DBMS software like Oracle, SQL Server or MySQL.
- Designing Authorization/Access Control
- Designing Recovery Procedures
- **Operations Management**
- The responsibilities include
  - Investigation of errors found in the data.
  - Supervision of restart and recovery procedures in the event of a failure.
  - Supervision of reorganization of databases.
  - Initiation and control of all periodic dumps of data.

Q. 4 List the important skills required to a Database Administrator (DBA). (3 Marks)

Ans. :

- The various programming and soft skills are required to DBA are as follows,
  - Good communication skills
  - Excellent knowledge of databases architecture and design and RDBMS
  - Knowledge of Structured Query Language (SQL).
- In addition, this aspect of database administration includes maintenance of data **security**, which involves maintaining security authorization tables, conducting periodic security audits, investigating all known security breaches.
- It should also have information about the source and destination of a data item and the flow of a data item as it is used by a system. This type of information is a great help to the DBA in maintaining centralized control of data.

Chapter 3 : Database Design Using ER Model

Q. 1 Explain Components of ER Model. (2 Marks)

Ans. ;

- ER diagram is the first step of database design to specify the desired components of the database system and the relationships among those components.
- ER model define data elements and relationships among various data elements for a specified system.

- The ER data model is based on perception of real world data that consists of set of entities (data items) and relationship among these entities.
- ER Diagrams having components,
  - Entity
  - Attributes
  - Relationships

Q. 2 What is strong entity and weak entity? Explain with example. **(3 Marks)**

Ans. :

An entity may be an object with a physical existence or it may have logical existence. Example, Department, Section, subject may have logical existence.

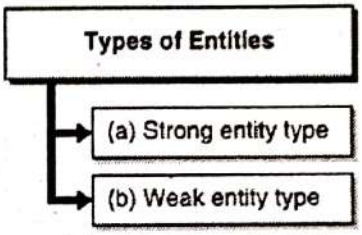


Fig. 3.1 : Types of entities

(a) Strong entity type

Definition : Entity type which has its own key attributes by which identify specific entity uniquely is called as **strong entity type.**

Example :

Strong entity type is represented by single rectangle.

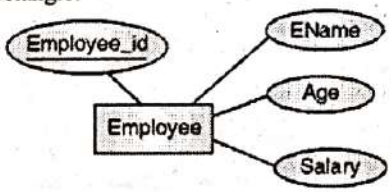


Fig. 3.2 : Employee entity

(b) Weak entity type

Definition : Entity type which cannot form distinct key from their attributes and takes help from corresponding strong entity is called as **weak entity type.**

Weak entity type is represented by double rectangle.

Example :

In case of "Dependent" entity depend on employee entity for primary key.

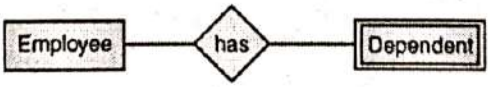


Fig. 3.3 : Weak entity "dependent"

Q. 3 Explain different types of attributes in ER Model. **(5 Marks)**

Ans. : Each entity has its own properties which describes that entity such properties are known as attributes.

Types of attribute are used in ER diagrams,

(a) Composite Attributes

The attributes which can be divided in multiple subparts.

Type	Notation
Composite attribute	

Example :

The Name attribute of Student table can be divided into First_Name and Last_Name.

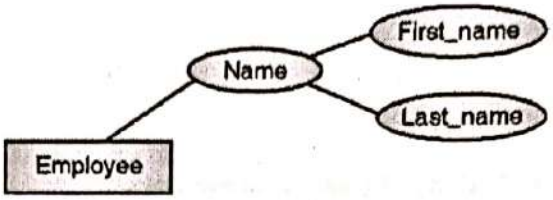
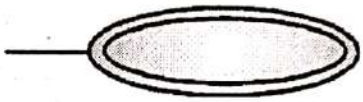


Fig. 3.4 : Composite attributes

(b) Multivalued Attributes

The attribute having more then one value for a same entity is called as multi-valued attribute.

Type	Notation
Multivalued Attribute	

Example :

A Single student can have multiple mobile numbers.

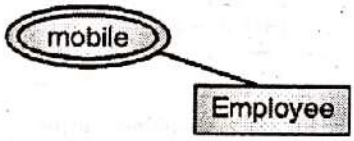
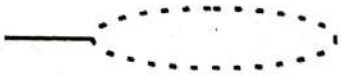


Fig. 3.5: Multi valued attributes

(c) Derived Attributes

Definition : The value of some attribute can be derived from the value of related stored attribute such attributes are known as **derived attributes.**

Type	Notation
Derived attribute	

Example :

Employee tenure can be calculated from stored attribute 'Date_of_joining' of employee by subtracting it from today's date.

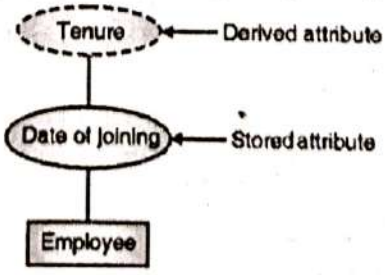


Fig. 3.6 : Derived attributes

(d) **Null Attribute :** This attribute can take NULL value when entity does not have value for it.

Example :

- The 'Net_Banking_Active_Bin' attribute gives weather particular customer having net banking facility activated or not activated.
- For bank which does not offer facility of net banking in customer table 'Net_Banking_Active_Bin' attribute is always null till Net banking facility is not activated as this attribute indicates Bank offers net banking facility or does not offers.

(e) **Key Attributes**

This is an attribute of an entity which must have a unique value by which any row can be identified is called as key attribute of entity.

Example : Emp_Id for employee.

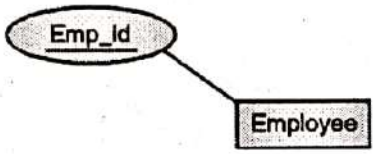
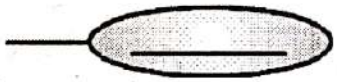


Fig. 3.7 : Key attributes

Type	Notation
Key attribute	

- The column value that uniquely identifies a single record in a table called as key of table.
- An attribute or set of attributes whose values uniquely identify each entity in an entity set is called a key for that entity set.
- ID is a key of student table. It is possible to have only one student with one ID (only one student **'Mahesh'** with ID = 1)

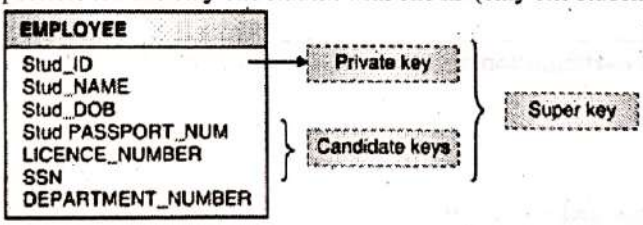


Fig. 3.8 : Key Concept

Types of Keys : The various types of keys in ER Diagrams are as follows,

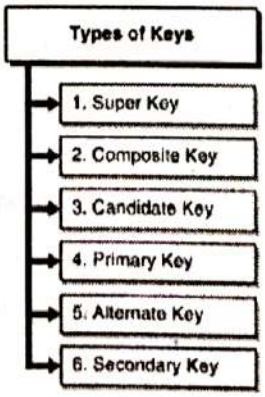


Fig. 3.9 : Types of keys

Q. 4 What is relationship set? **(2 Marks)**

Ans. :

- A relationship is an association among one or more than one entities.
- Use diamond shape to show relationship.

Type	Notation
Relationship	

Example :

Employee works for Department.

Fig. 3.10 : ER Diagram for Works_for

Relationship Set

Collection of all relationship of same type is relationship set. The many employees are working for different departments so it is relationship set of Works_For relationship.

Degree of Relationship

The degree of relationship type is number of participating entity types in a particular relation.

Types of Relationship based on degree are,

- Unary Relationship
- Binary Relationship
- Ternary Relationship

Q. 5 Explain the terms total participation and Partial participation with example. **(5 Marks)**

Ans. :

(i) Total participation

- In case of total participation every object in an entity must participate in a relationship.
- The total participation is indicated by a dark line or double line between entity and relationship.

Example :

Every department must have a manager.

Fig. 3.11 : Total participation

(ii) Partial participation

- In case of partial participation more than one object in an entity may participate in a relationship.
- The total participation is indicated by a single line between entity and relationship.

Example : Employees works for department.

Fig. 3.12 : Partial participation

Q. 7 Explain aggregation with the help of an example. (5 Marks)

Ans. :

- Aggregation is meant to represent a relationship between a whole object and its component parts.
- It is used when we have to model a relationship involving entity sets and a relationship set.

Example :

- A Project is sponsored by a department. This is a simple relationship.
- An Employee monitors this sponsorship (and not project or department). This is aggregation. Monitors are mapped to the table like any other relationship set.

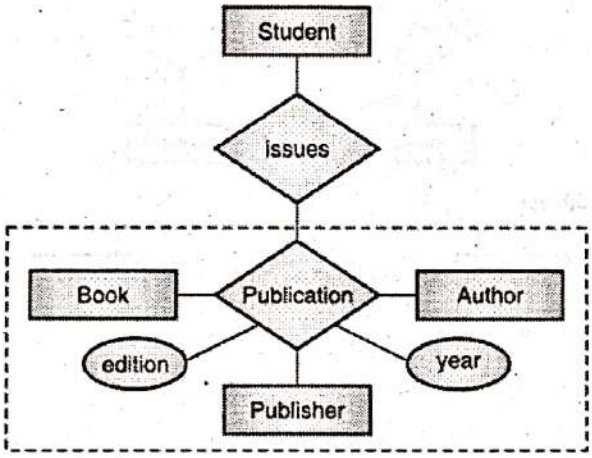


Fig. 3.15 : Aggregation

Q. 8 A publication may be a book or an article. Articles are published in Journals. Publication has title and location. Book having their title and category. Article includes title, Topic and date. Publication is written by Authors stores Name, address and mobile number. Publication also belongs to particular subject which has their names. (10 Marks)

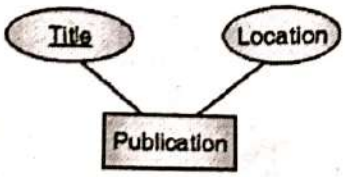
Ans. :

Step 1 : Identify entities

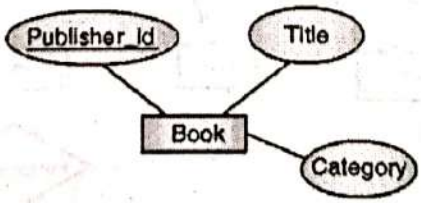
1. Publication 2. Book 3. Article
4. Journal 5. Subject 6. Author

Step 2 : Identify attributes

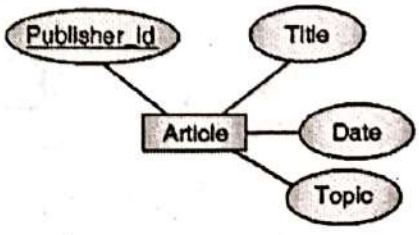
1. Publication (Title, Location)



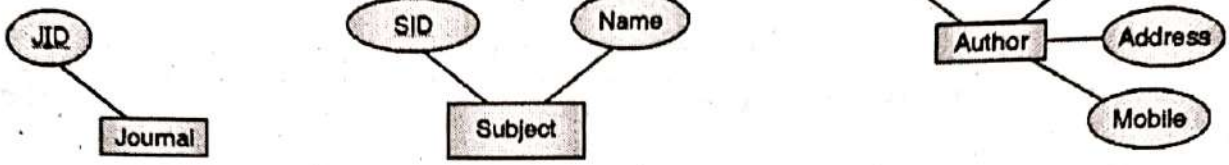
2. Book (Publisher_id, Title, Category)



3. Article (Publisher_id, Title, Date, Topic)

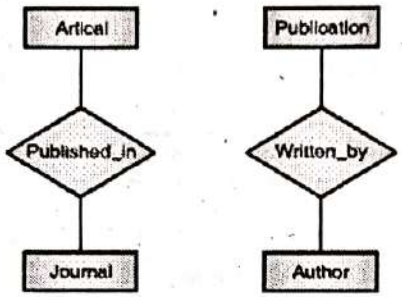


4. **Journal (JID)** 5. **Subject (SID, Name)** 6. **Author (Auth_id, Name, Address, Mobile)**



Step 3 : Identify relationships

1. Articles are published in Journal. 2. Publication is written by Author.

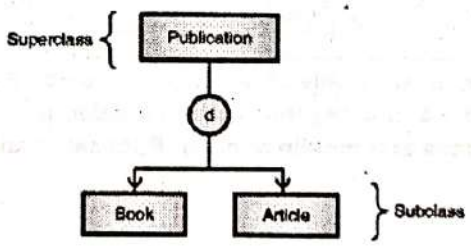


3. Publication belongs to a particular subject.

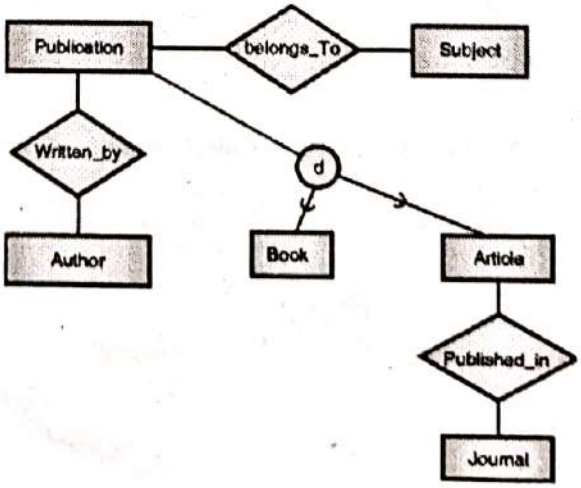
Step 4 : Identify inheritance relations

Publication can be

BOOK or ARTICLE.



Step 5 : Merging all above relations we will get final ER model



Q. 9 Construct on E-R diagram for a car-insurance company that has a set of customers each of whom owns one or more cars. Each car has associated with it zero to any number of recorded accidents. (10 Marks)

Ans. :

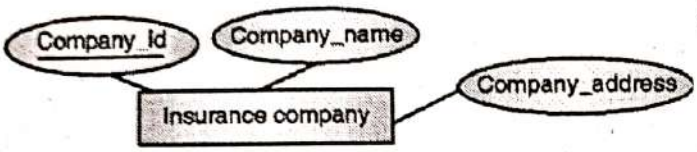
(1) Identify all entities

(a) Insurance company (b) Customer

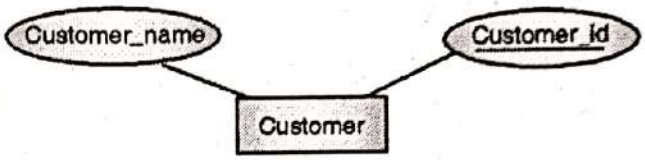
(c) Car (d) Accidents

(2) Identify all attributes

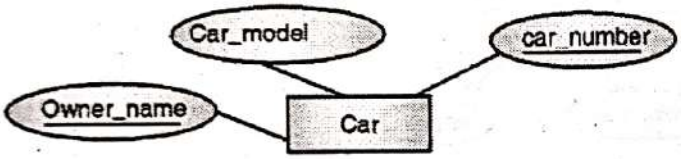
(a) Company entity



(b) Customer entity



(c) Car entity

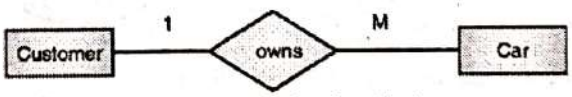


(d) Accidents

(3) Identify all relationship

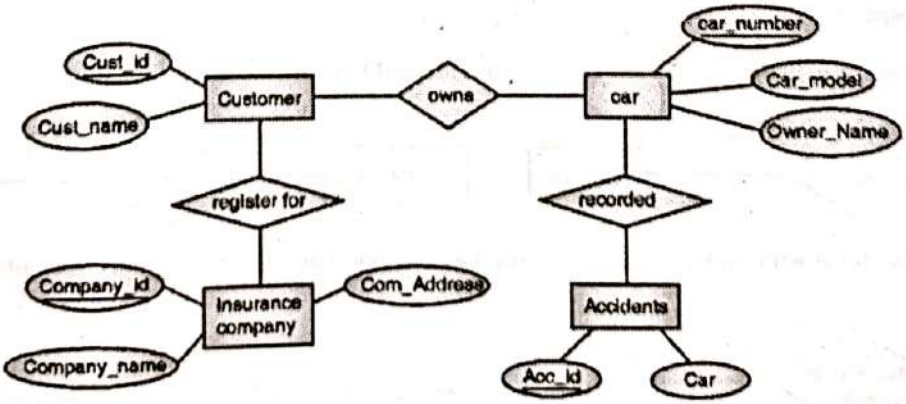
a) Car insurance company has a set of customers

b) Customer owns one or more car



c) Each car associated with zero or any number of accidents

(4) Construct ER diagram by merging all above relationships



Q. 10 Construct an ER diagram for a hospital with a set of patients and the set of medical doctors associated with each patient a record of various text and examination conducted. (10 Marks)

Ans. :

(1) Identify Entities

(i) Hospital

(ii) Patient

(iii) Doctors

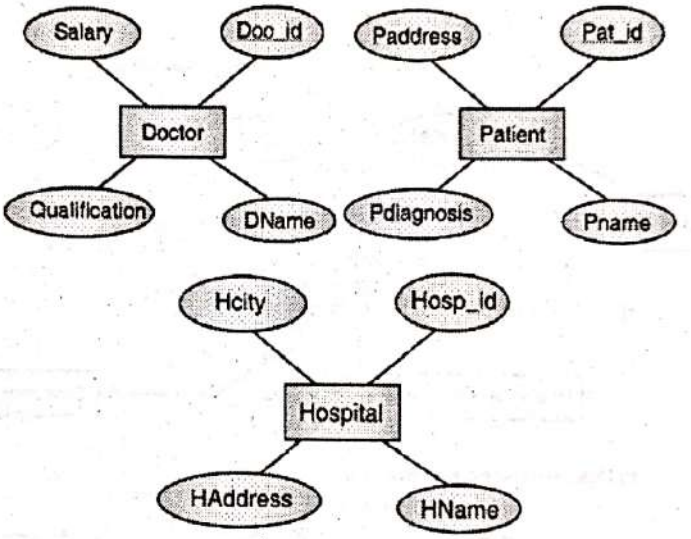
(iv) Medical-record (Record of various test and examination conducted)

(2) Identify Attributes

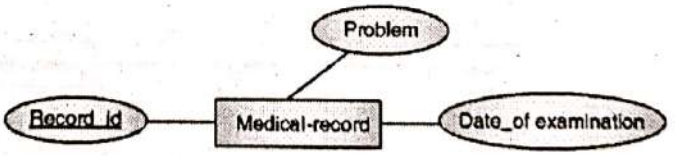
(i) Hospital (Hosp_id, HName, HAddress, Hcity)

(ii) Patient (Pat_id, Pname, Pdiagnosis, Padress)

(iii) Doctor (Doc_id, DName, Qualification, salary)



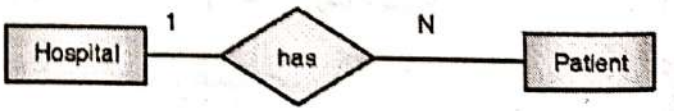
(iv) Medical_Record (Record_id, Date_of examination, Problem)



(3) Identify relationships

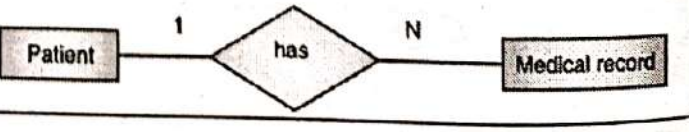
(a) Hospital has a set of patients

(b) Hospital has a set of doctors

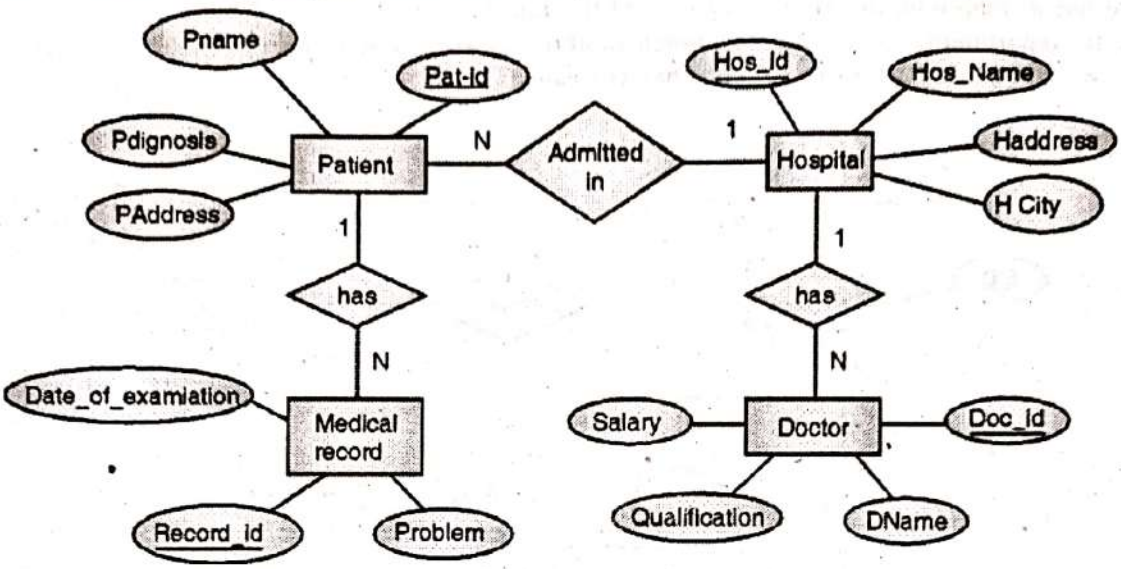


(c) Doctors are associated with each patient

(d) Each patient has record of various test and examination conducted

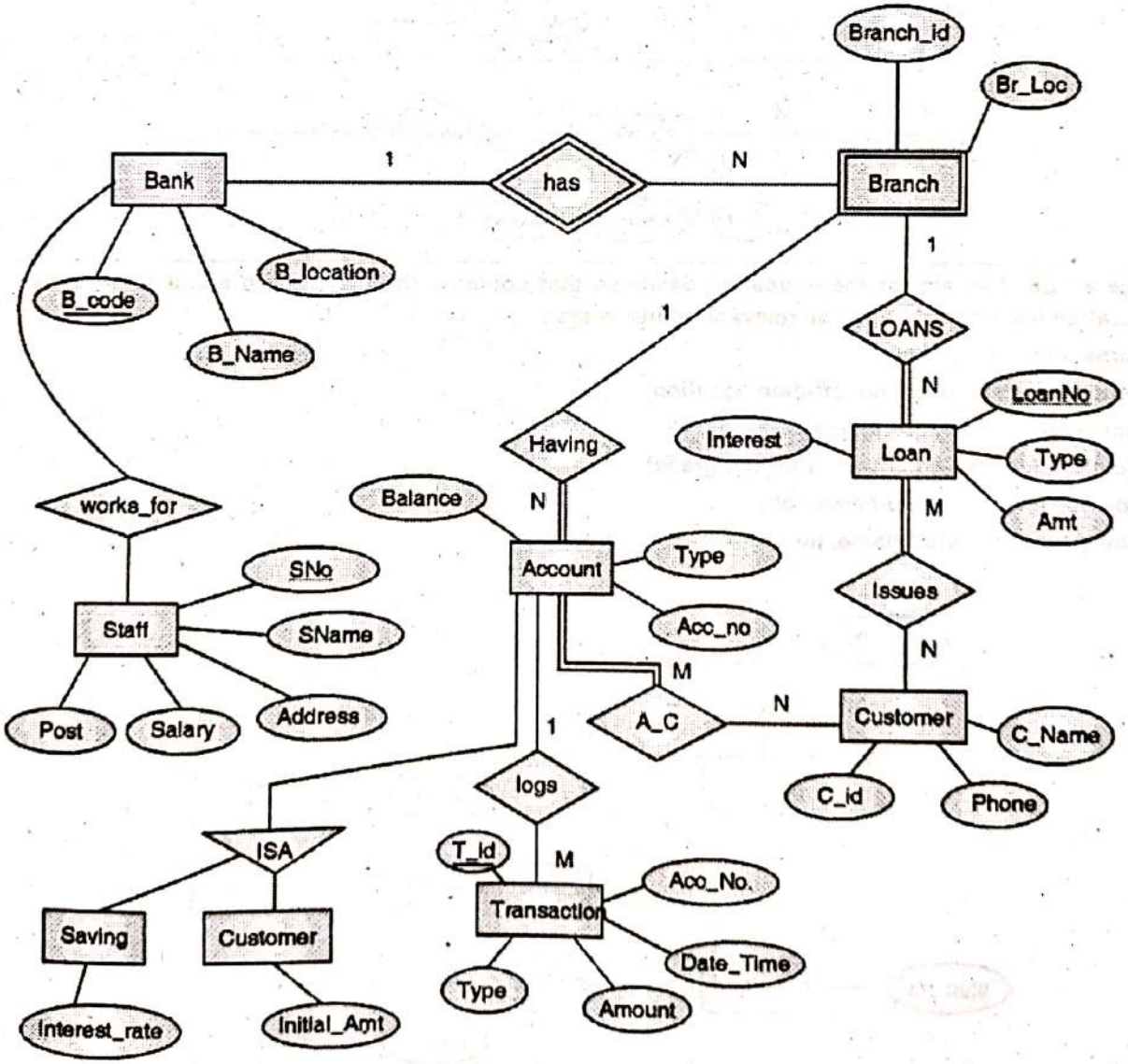


(4) Construct ER Model from merging all above relationship



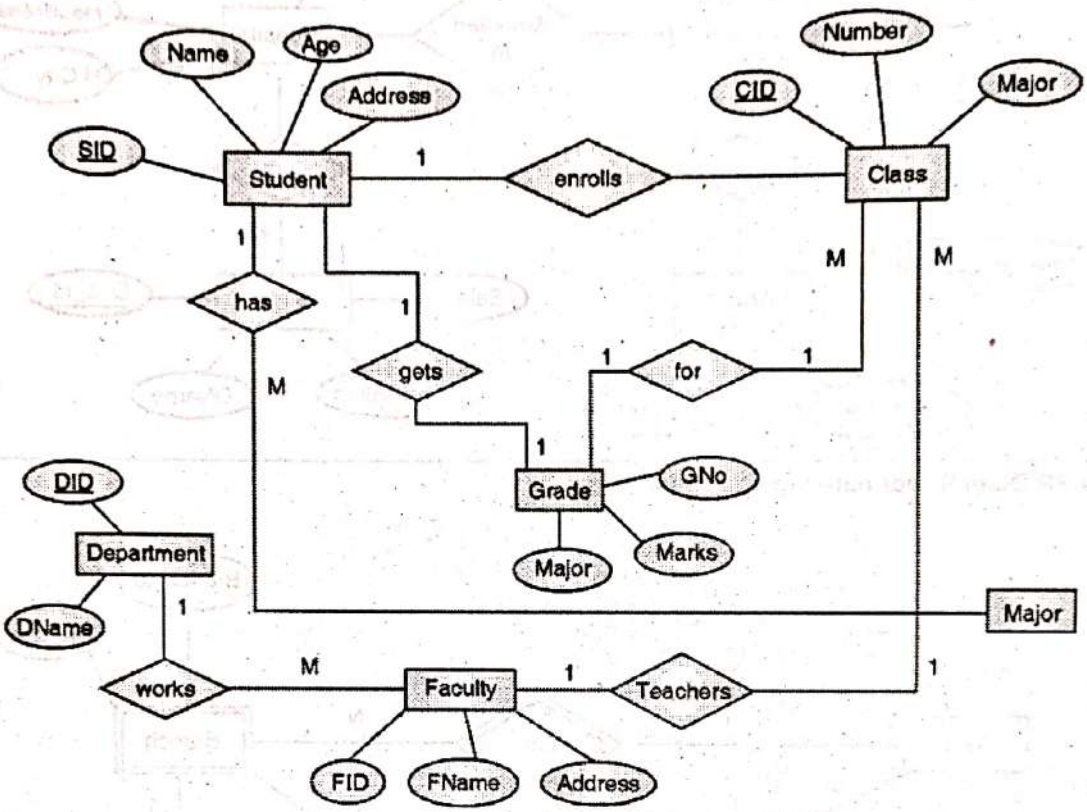
Q. 11 Draw ER Diagram for banking enterprise. (10 Marks)

Ans. :



Q. 12 Draw ER Diagram for University database consisting four Entities Student, Department, Class and Faculty. Student has a unique id, the student can enroll for multiple classes and has a most one major. Faculty must belong to department and faculty can teach multiple classes. Each class is taught by only faculty. Every student will get grade for the class he/she has enrolled. (10 Marks)

Ans. :



Q. 13 Draw an ER diagram for the education database that contains the information about an in house company education training scheme. The relevant relations are :

Course (course-no, title)

Offering (course-no, off-no, off-date, location)

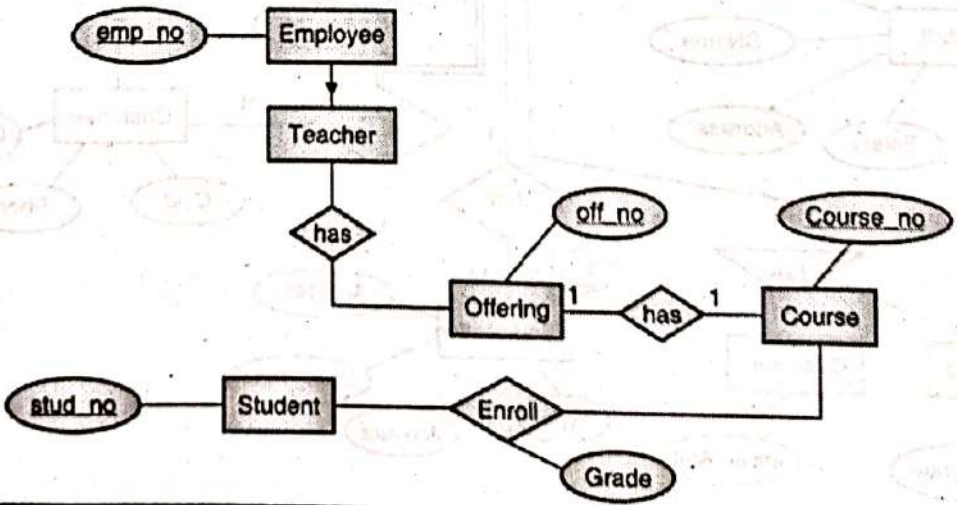
Teacher (course-no, off-no, emp-no)

Enrolment (course-no, off-no, stud-no, grade)

Employee (emp-no, emp-name, job)

Student (stud-no, stud-name, ph-no) (10 Marks)

Ans. :



Chapter 4 : Relational Data Model

Q.1 Define Relation. **(2 Marks)**

Ans. : Relation (Table)

Definition : Relations are a logical structure which is a collection of tables consisting horizontal rows also called as tuples and vertical columns also called as **Attributes.**

Characteristics of Relation

1. A table composed of rows and columns.
2. Each table in a database has its unique *table name.*
3. Each table row (tuple) represents a single entity occurrence within the entity set.
4. All values in a same column must conform to the same format of data.
5. Each table must have a single attribute or set of attributes that uniquely identifies each row.

Example :

The student relation can be,

Id	Name	Age	Class	Branch
105	Mahesh	25	BE	IT
106	Suhas	28	FE	CS
107	Jay	29	SE	CS
108	Sachin	30	TE	EXTC

Q.2 Write a note on Relational Database Schema. **(2 Marks)**

Ans. : The relational schema describes structure of relation (i.e. table) and relational database schema explains the structure of relational database.

Relational Database Schema : Relational database schema consists of a number of relation schemas associated with that database.

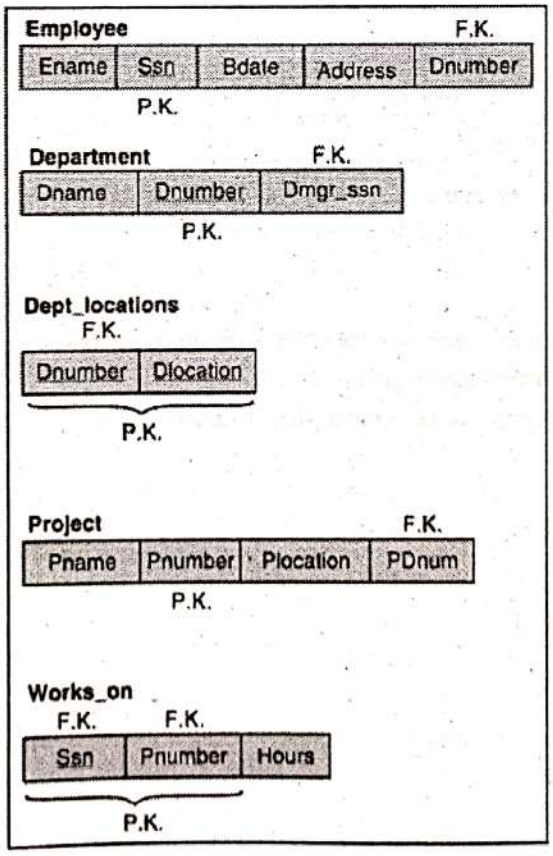


Fig. 4.1 : Sample Relational Database Schema

Q.3 Explain different Integrity constraints. **(10 Marks)**

Ans. : Relational Model Constraints

- Constraints makes sure that only authorised user will make modifications to database and changes should not lead to loss of data consistency and correctness.
- These concepts makes sure correctness and completeness of data stored in the database.
- This objective can never be guaranteed, one cannot ensure that every entry made in database is accurate.

Types of Relational Constraints

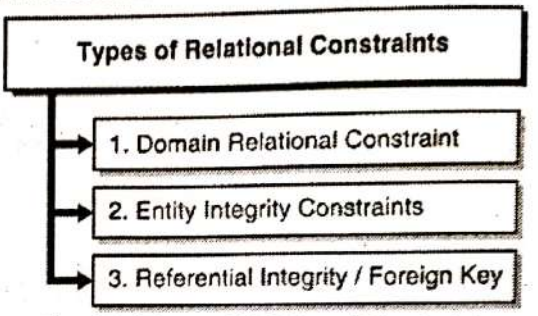


Fig. 4.2 : Types of Relational Constraint

1. Domain Relational Constraint

- Domain constraints allow us to test whether the values inserted into the database are correct or not.
- The CREATE TABLE Command may also include domain constraints which can check integrity of database.
- These domain constraints are the most basic form of integrity constraint.

Types of Domain Constraints

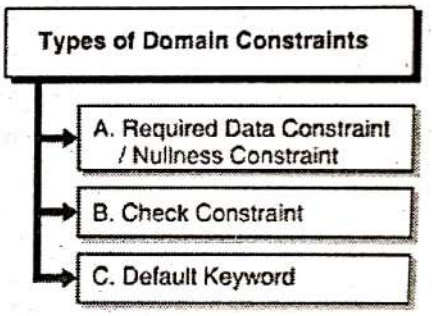


Fig. 4.3 : Types of Domain Constraints

A. Nullness Constraint

- A. Required Data Constraint / Nullness Constraint
- The database may have some attributes mandatory like user registration must have user email address.
- These attributes (columns) in a database are not allowed to contain NULL values or blanks.

Example :

In the student table, student must have an associated student name.

```
Student_Name varchar(100) NOT NULL
```

Therefore now, the Student_Name column in the STUDENT table is a required data column. It is not possible to insert Null value in Student_Name column of Student table.

B. Check Constraint

- The **check constraint** is used to ensure that attribute value satisfies specific condition as specified by data requirements or user.
- Suppose in Student Table, gender of student can be male or female only.
- The DBMS can prevent user from entering incorrect or other data in database table.

Example :

Table with student entity having gender which can be M or F.

Hence, attribute gender can take only two values either 'M' or 'F'.

```
Student_Gender varchar(1) CHECK (gender IN ('M','F'))
```

C. Default Keyword

- Default keyword is used to add a default specified value, if attribute value is not provided by user.
- It avoids the addition of NULL value to the database by inserting default value as specified by the developer while creating a table.

Example :

Table with customer entity having name and gender.

If name is not added for customer that will be taken as 'Unknown' if we specify DEFAULT value of NAME column to 'UNKNOWN'

```
Student_Name varchar(50) DEFAULT 'UNKNOWN'
```

2. Entity Integrity Constraints

- Entity constraints allow us to test whether the tuple (entity) inserted into the database are correct or not.
- The create table Command may also include entity constraints which can primary key of table.

Types of Entity Constraints

A. Unique Constraint / Unique Key

- In case of unique constraint no two tuples can have equal value for same attributes.
- This constraint says that attributes forms candidates key, which allows one Null value which is unique by itself.
- This UNIQUE constraint can be applicable to user defined domain declaration also.

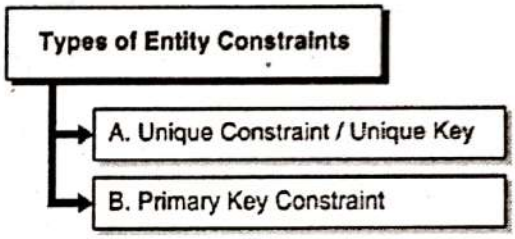


Fig. 4.4 : Types of Entity Constraints

Example :

```
EMAIL varchar(30) UNIQUE
```

B. Primary Key Constraint

- A table in a relational database has one column or combination of some columns whose values uniquely identifies a single row in the table. This column or combination of columns is called the **primary key** of the table.
- Primary key attribute is same as unique key constraint with NOT NULL constraints (Unique constraint+ Not Null constraint).
- For example, each row of the STUDENT table has a unique set of values in its STUDENT_ID column, which uniquely identifies the student represented by that row.
- Duplicate values are not allowed in primary key column, because they cause problems in distinguishing one entity from another (entity may be an employee).

```
STUDENT_ID char(10) PRIMARY KEY
```

3. Referential Integrity / Foreign Key

- A value appearing in a one relation (table) for a given set of attributes also appears for another set of attributes in another relation (table). This is called referential integrity.
- The referential integrity constraint is specified between two tables to maintain the consistency among tuples in the two tables.

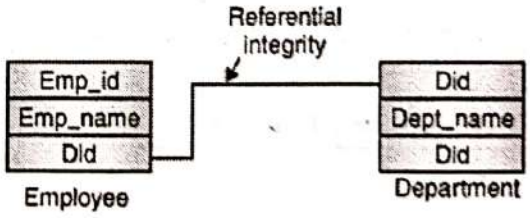


Fig. 4.5 : Referential integrity

Employee Table		
Emp_Id	**Emp_name**	**Did**
1	Sachin	20
2	Suhas	10
3	Jay	20
4	Om	10

Department Table	
Did	**Dept_name**
10	HR
20	TIS
30	L&D

- In the above example Employee table has Did as foreign key reference to Did column in Department table this is called as referential integrity.

Q. 4 Explain the algorithm to map ER and EER model to relational model in detail. **(10 Marks)**

Ans. :

(1) Regular entity types

- **Tables :** Regular entity sets can be represented as table in relational model.
- **Columns :** Attributes of entity set can be converted to the columns (attributes) of the tables in relational model.

Example :

Regular entity employee mapped as employee table in object model like 'Stud_id', 'Stud_Addr' etc. are shown as table columns.

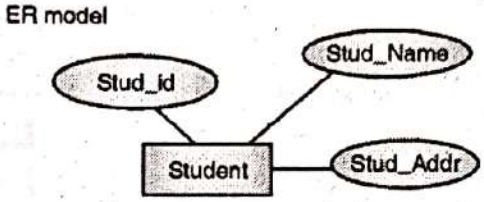


Fig. 4.6 : Regular entity

Stud_id	Stud_Name	Stud_Addr
1	Snehal	Mumbai
2	Pratiksha	Mumbai
3	Supriya	Mumbai
4	Tanmay	Goa

(2) Weak entity types

For each weak entity type with owner entity, create a table and include all simple attributes of weak entity type as columns of table, including foreign key attributes as the primary key attribute of the table that correspond to the owner entity type.

Example : Dependents (Weak entity) in Employee (Owner entity).

E_id	Ename	DName	DAge
1	Sachin	Jyoti	23
2	Suhas	Manju	22
3	Jayendra	Tanya	27

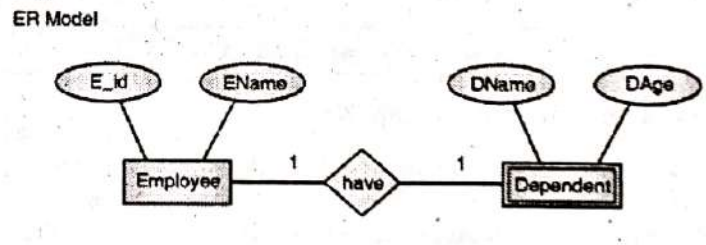


Fig. 4.7 : Weak entity

Q. 5 Explain how to convert attribute in ER to relational Table. **(5 Marks)**

Ans. :

(a) Simple attributes

Simple attribute can be directly converted to a column (Attribute) in relational model.

Example :

Employee 'Age' can be directly converted to column.

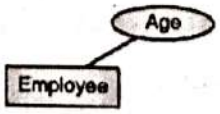


Fig. 4.8 : Simple attribute

Employee table

Eid	Age
1	23
2	24
3	43
4	28

(b) Composite attributes

These attributes need to be stored as set of simple component attributes (Columns) in relational model by avoiding actual attribute ('name' in below example).

Example :

In below example composite attribute 'Name' is converted to three columns in object model by avoiding actual attribute 'Name'.

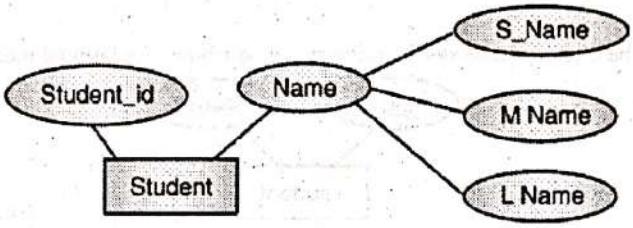


Fig. 4.9 : Composite attribute

Student table

Student_id	S_Name	MName	LName
1	Harshad	Rupali	Malar
2	Bipin	Anand	Shinde
3	Aanand	Ganesh	Panchal
4	Tushar	Bipin	Pimple

(c) Multi valued attributes

Multi valued attributes are mapped as a relation which includes combination of the primary key of table and multi valued attribute as a composite primary key as shown in Fig. 4.10.

Mobile table	
Sid	Mobile No.
1	9891295492
1	9821959241
2	8080918456
3	8095124890
3	9773112456

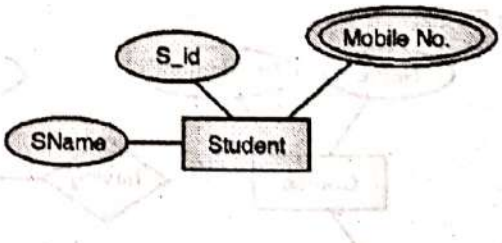


Fig. 4.10 : Multi valued attribute

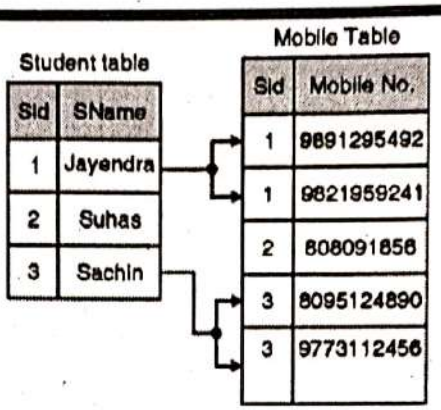


Fig. 4.11

(d) Derived attributes

There is no need to store such attribute in relational model. It will be calculated from stored attribute.

(e) Key attributes

Key attribute in ER Model can be directly converted to primary key attribute of relational model.

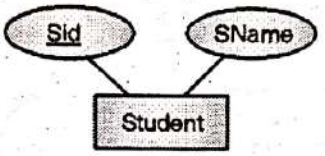


Fig. 4.12 : Key attributes

Student table

Sid	SName
1	Deepak
2	Vaibhav
3	Yogita
4	Bency

Q. 6 **Draw an E-R diagram and reduce it to relational database model for a university database for scheduling of classrooms for final exams. This database could be modelled using entities as exam (course_name, section_number, room_number, time);course. (name, department, C_number),room (r_number, capacity, building).Entity section is dependent on course.** **(10 Marks)**

Ans. :

Step 1 : ER diagram

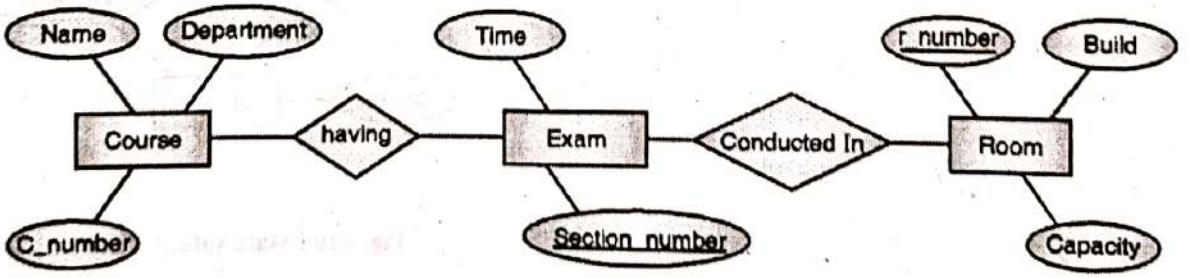


Fig. 4.13

Step 2 : Mapping to Tables

Course (C_number, name, department)

Exam (Section_number, time, C_number,)

Room (R_number, building, capacity, Section_number)

Q. 7 Draw an E-R diagram for a university database consisting of 4 entities,

(i) Student (ii) Department

(iii) Class (iv) Faculty

and convert it to tables. A student has a unique id, the student can enroll for multiple classes and has at most one major. Faculty must belong to department and faculty can take multiple classes.Every student will get a grade for the class he/she has enrolled. (10 Marks)

Ans. :

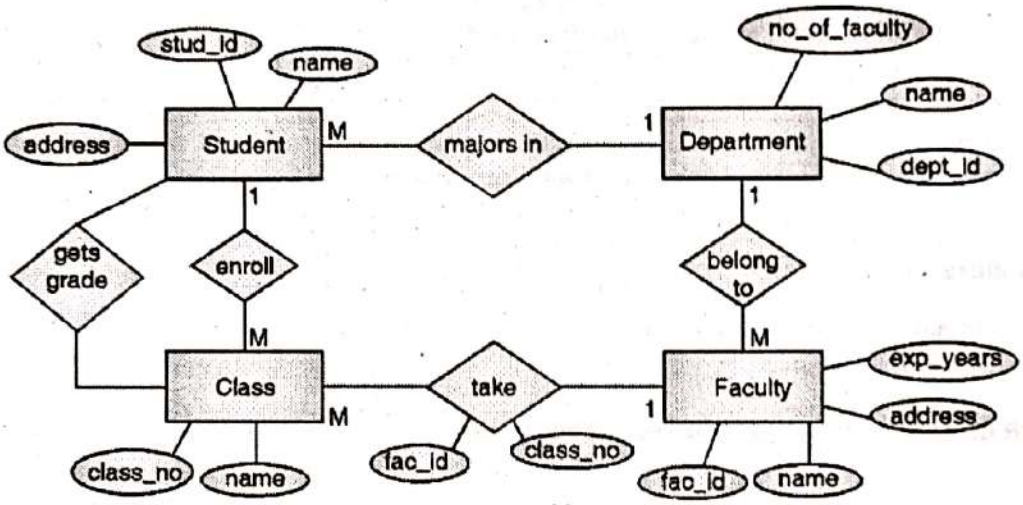


Fig. 4.14

Student	
Id	Primary Key
Name	
Address	
Dept_id	Foreign key references to dept_id column of Department table
Faculty	
Fac_id	Primary Key
Fac_name	
Exp_years	
Address	
Dept_id	Foreign key references to dept_id column of Department table
Class	
Class_ no	Primary Key
C_name	
Stud_class	
Class_ no	Foreign key references to dept_id column of Department table
Stud_ id	Foreign key references to dept_id column of Department table

take_class	
Fac_id	Foreign key references to fac_id column of Faculty table
Class_ no	Foreign key references to class_ no column of Class table
Department	
Dept_id	Primary Key
D_name	
No_of faculty	
Grade	
Stud_ id	Foreign key references to dept_id column of Department table
Class_ no	Foreign key references to dept_id column of Department table
Grade	

Q. 8 Construct on E-R diagram for a car-insurance company that has a set of customers each of whom owns one or more cars. Each car has associated with it zero to any number of recorded accidents. (10 Marks)

Ans. :

(1) Identify all entities

(a) Insurance company (b) Customer

(c) Car (d) Accidents

(2) Construct ER diagram by merging all above relationships

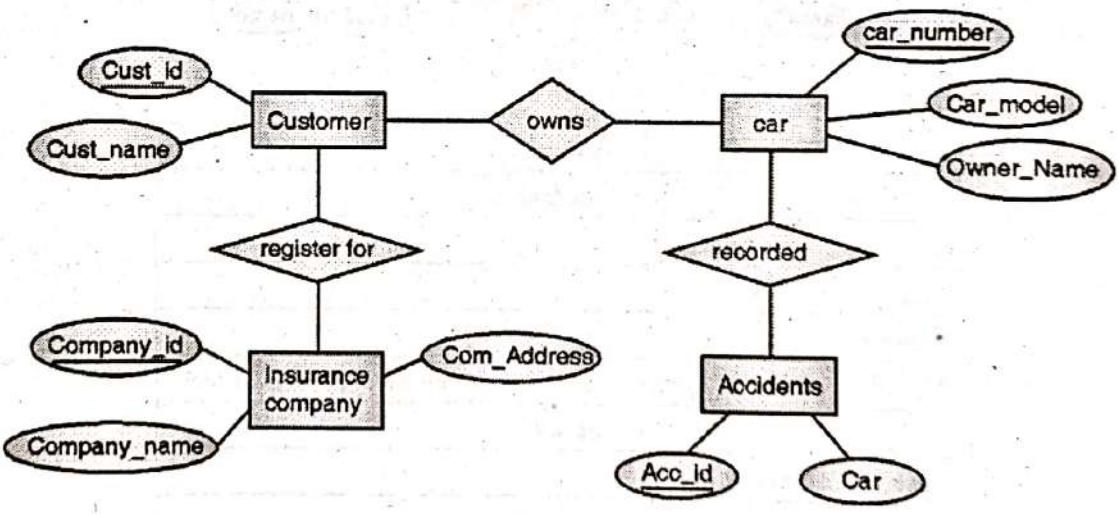


Fig. 4.15

(1) Mapping Entities

(a) Company (Company_id, Name, Address)

(b) Customer (Customer_id, Name, Address, phone)

(c) Car (Car_Number, Car_Model, Owner)

(d) Accidents (Accident_Id, Location, date, time)

(2) Mapping Relations

(a) Company (Company_id, Name, Address)

(b) Customer (Customer_id, Name, Address, phone, Insurance_Company)

Insurance_Company - refers to customers registered insurance company

(c) Car (Car_Number, Car_Model, Owner_Id)

Owner_Id - refers to customer id owns that car.

(d) Accidents (Accident_Id, Car_Number, Location, date, time)

Car_Number - refers to car involved in accident.

(3) Final Relational Schema

(a) Company (Company_id, Name, Address)

(b) Customer (Customer_id, Name, Address, phone, Insurance_Company)

(c) Car (Car_Number, Car_Model, Owner_Id)

(d) Accidents (Accident_Id, Car_Number, Location, date, time)

Q. 9 Construct ER diagram and convert into Relational Model for Company Which has several Employees working on different types of projects. Several Employees are working on one department. Every Employee has Manager. Several Employees are supervised by one Employee. (10 Marks)

Ans. :

1. Employee (Eid, Ename, mid, sup-id),Pid)
2. Company(Cid, Cname, Location)
3. Project (Pid, Pname, type)

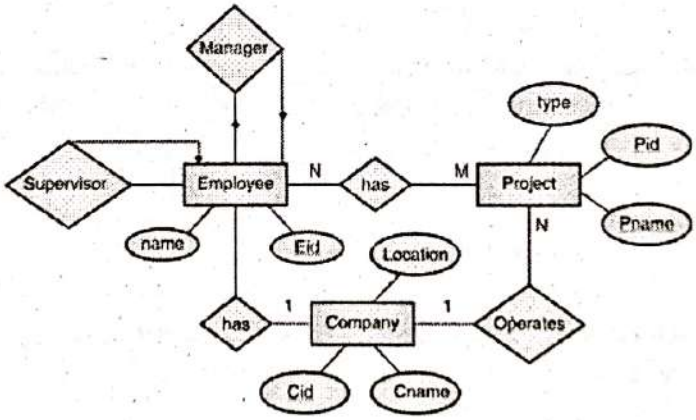


Fig. 4.16

Chapter 5 : Relational Algebra

Q. 1 Discuss fundamental operations in relational algebra. (10 Marks)

Ans. :Relational Algebra

- It is procedural language useful for representing query execution plan and relatively close to SQL.
- Relational algebra is set of operations which accept one relation and produces new relation as a result.

Fundamental operations of Relational Algebra

1.Selection Operation (σ)

- This operator is used to select some rows from table which satisfy particular selection condition given in selection operation.
- Selection operator selects a set of tuples that satisfy a selection predicate or condition.

- Output of query is exactly same as input schema of table.
- This is unary relational operator having only one input table.

Syntax

$\sigma_{<attribute\_name><comparison\_operator><constant\_value>}$ (Input_Table_Name)

2. Projection Operation (π)

- This operator is used for selecting some of many columns in table to displayed in result set.
- Projection operator can select a column or set of columns of table to be displayed in output of query.
- Select only few columns or all columns of a table as per requirements.
- This is unary relational operator having only one input table.

Syntax

$\pi_{<column\_list>}$ (Input_Table_Name)

3.Rename Operation (ρ)

- Give alternative name to any column (attribute) or any table of query expressions using operator called as RENAME operator.
- This operator is specially introduced to select specific column from joined table (set of two or more tables) containing multiple columns of same column name.
- Rename operator, denoted by the lowercase Greek letter rho (ρ).

Syntax

$\rho_{<New\_Name>}$ (Input_Table_Name)

4.SET Operation

- SQL SET operators allow combining results from two or more SELECT statements or combines result set of multiple queries.
- The results of two queries can be combined using the set operations union, intersection and difference.

Query_1 UNION [ALL] **Query_2**

Query_1 INTERSECT [ALL] **Query_2**

Query_1 EXCEPT [ALL] **Query_2**

SET Compatibility

- In order to apply SET operations the source tables must possess following Requirements,
- SELECT statement of both queries must retrieve the same number of columns.
- SELECT columns of both queries must be of same Data types.

Q. 3 Explain Set Union Relational algebra operators with suitable examples. **(3 Marks)**

Ans. : Union Operator

- This operator finds out all combined rows in table 1 and table 2.
- Union effectively appends the result of first query to the result of second query.
- It does not eliminate all duplicate rows and they are printed in result expression.

Syntax

(Query Expression 1) $\cup$ (Query Expression 2)

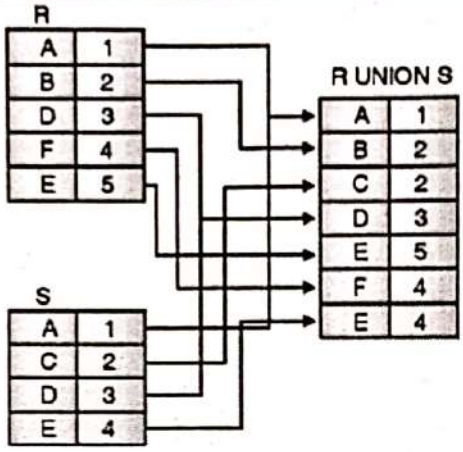


Fig. 5.1 : SET Operation

(c) Example :

(i) IT Employee table

Table Name : IT_Employee		
Eid	**Ename**	**Age**
11	Suhas	24
12	Jayendra	24
13	Sachin	25
14	Mahesh	23

(ii) Computer department Employee table

Table Name : COMP_Employee		
Eid	**Ename**	**Age**
21	Varsha	24
22	Bhavna	24
23	Geeta	25
24	Amrita	23

Query : Find all Employees in computer and IT departments.

Solution : (IT_Employee) ∪ (COMP_Employee)

Eid	Ename	Age
11	Suhas	24
12	Jayendra	24
13	Sachin	25
14	Mahesh	23
21	Varsha	24
22	Bhavna	24
23	Geeta	25
24	Amruta	23

Q. 4 Explain Set Intersection operation with suitable examples. **(2 Marks)**

Ans. : Intersect Operator

- This operator finds out all rows that are common in table 1 and table 2.
- If Intersect operator is applied on two queries then it will return all rows that are common in the result of Query 1 and Query 2.

Syntax

(Query Expression 1) ∩ (Query Expression 2)

R	
A	1
B	2
D	3
F	4
E	5

R INTERSECTION S	
A	1
D	3

S	
A	1
C	2
D	3
E	4

Example :

(i) All employees in IT department.

Table Name : IT_Employee		
Eid	Ename	Age
11	Suhas	24
12	Jayendra	24
13	Sachin	25
14	Mahesh	23

(ii) All employees in Vidya Engineering College.

Table Name : Vidya_Employee		
Eid	Ename	Age
11	Suhas	24
12	Jayendra	24
23	Geeta	25
24	Amruta	23
35	Sangita	21

Query : Find all Employees in IT department of Vidya Engineering College.

Solution : (IT_Employee) ∪ (Vidya_Employee)

Eid	Ename	Age
11	Suhas	24
12	Jayendra	24

Q. 4 Explain Set Difference Relational algebra operators with suitable examples. **(2 Marks)**

Ans. : Difference Operator

- This operator finds out all rows that are present in table 1 and not in table 2.
- If Intersect operator is applied on two queries then it will return all rows that are present in the result of Query 1 and not in Query 2.

Syntax

(Query Expression 1) - (Query Expression 2)

R	
A	1
B	2
D	3
F	4
E	5

S	
A	1
C	2
D	3
E	4

R DIFFERENCE S	
B	2
F	4
E	5

S DIFFERENCE R	
C	2
E	4

Example :

(i) All faculties in IT department of Vidya Engineering College.

Table Name : Vidya_Employee		
Eid	**Ename**	**Age**
11	Suhas	24
12	Jayendra	24
13	Sachin	25
14	Mahesh	23

(ii) All faculties in IT department of all colleges.

Table Name : IT_Employee		
Eid	**Ename**	**Age**
11	Suhas	24
12	Jayendra	24
13	Sachin	25
14	Mahesh	23
23	Geeta	25
24	Amruta	23
35	Sangita	21

Query : Find all Employees in IT department but not in Vidya Engineering College.
Solution : (IT_Employee) - (Vidya_Employee)

Eid	Ename	Age
23	Geeta	25
24	Amruta	23
35	Sangita	21

Q. 5 Explain Cartesian Product Relational algebra operation. **(2 Marks)**

Ans. : Cartesian Product

- A cross join performs relational product or Cartesian product of two tables specified in query.
- In this case every row in first table will be joined with every row in second table. So finally number of rows in result table will be equals to product of number of rows in table 1 and number of rows in table 2.
- That means all rows in the first table are joined to all rows in the second table.

Syntax

(Query Expression 1) × (Query Expression 2)

Example :

Employee		
Eid	**Ename**	**Did**
1	Mahesh	100
2	Suhas	200
3	Jayendra	100

Department	
Did	**Dname**
100	HR
200	TIS

Query : Find combination all Employees and departments.

Solution : (Employee) X (Department)

Eid	Ename	Did	Did	Dname
1	Mahesh	100	100	HR
1	Mahesh	100	200	TIS
2	Suhas	200	100	HR
2	Suhas	200	200	TIS
3	Jayendra	100	100	HR
3	Jayendra	100	200	TIS

Cartesian product

- A Cartesian product is formed when :

  A join condition is omitted.

  A join condition is invalid.

  To avoid a Cartesian product, always include a valid join condition in a WHERE clause.

Q. 6 Express Join in terms of basic relational algebra operations. **(6 Marks)**

Ans. : Join Operation ($\bowtie_\theta$)

- Join operator help to retrieve data from multiple tables or relations.
- Most common type of join is **Natural join ($\bowtie$)** in which column having same name in two table will be taken for joining tables.

Syntax

(< table_name>) $\bowtie$ < join_condition > (table_name)

Types of Joins

There are various types of joins possible in relational algebra.

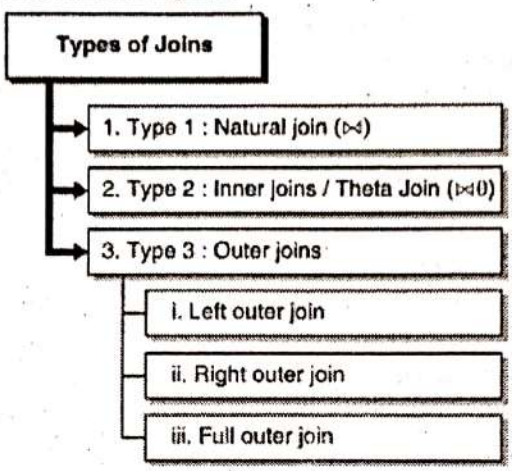


Fig. 5.2 : Types of Joins

Type 1 : Natural join (⋈)

- Natural join can join tables based on the common columns in the tables being joined.
- A natural join returns all rows by matching values in common columns having same name and data type of columns and that column should be present in both tables.

Example :

Employee		
Eid	**Ename**	**Did**
1	Amit	10
2	Nitin	30
3	Yogesh	50

Department	
Did	**Dname**
10	IT
30	HR
40	TIS

Query : Find all Employees and their respective departments.

Solution : (Employee) ⋈ (Department)

Eid	Ename	Did	Did	Dname
1	Amit	10	10	IT
2	Nitin	30	30	HR
Employee Data			Department Data	

Type 2 : Inner joins / Theta Join ($\bowtie_\theta$)

- Theta join will combines tuples from multiple relations if they satisfy the specified join condition.
- This join condition is also called as Theta and denoted by the symbol θ.
- The tables are joined according to join conditions.
- The only rows with matching v alues are combined using inner join.
- Inner join will ignore all tuple does not find matching tuple in other table.

Example :

Query : Find all Employees and their respective departments.

Solution : Employee ⋈ employee.did= Department.did Department

Eid	Ename	Did	Did	Dname
1	Amit	10	10	IT
2	Nitin	30	30	HR
Employee Data			Department Data	

Type 3 : Outer joins

- Outer join, joins two table although there is no match between two joining tables.
- Outer joins are useful when you are trying to determine which values in related tables cause referential integrity problem.

(i) Left outer join

- Table on left side of operator may contain null values.
- Left outer join takes all tuples in the left relation that did not match with any tuple in the right relation.

Example :

Query : Find all Employees and their respective department data.

Solution : Employee ⋈= employee.did = Department.did Department

Eid	Ename	Did	Did	Dname
1	Amit	10	10	IT
2	Nitin	30	30	HR
3	Yogesh	50	Null	Null
Employee Data			Department Data	

- In above example the employee data having did exactly same as department data are kept in result set.
- All left side non-matching tuples of cross join are also considered.

(ii) Right outer join

- Table on right side of operator may contain null values.
- Right outer join takes all tuples in the right relation that did not match with any tuple in the left relation.

Example :

Query : Find all departments with employee data.

Solution : Employee = ⋈employee.did = Department.did Department

Eid	Ename	Did	Did	Dname
1	Amit	10	10	IT
2	Nitin	30	30	HR
Null	Null	Null	40	TIS
Employee Data			Department Data	

- In above example the employee data having did exactly same as department data are kept in result set.
- All right side non-matching tuples of cross join are also considered.

(iii) Full outer join

Any table on both sides of operator may contain null values.

Example :

Query : Find all Employees and departments.

Solution : Employee $=\bowtie=_{employee.did=Department.did}$ Department

Eid	Ename	Did	Did	Dname
1	Amit	10	10	IT
2	Nitin	30	30	HR
3	Yogesh	50	Null	Null
Null	Null	Null	40	TIS
Employee Data			Department Data	

- In above example the employee data having did exactly same as department data are kept in result set.
- All right side non-matching tuples of cross join are also considered.

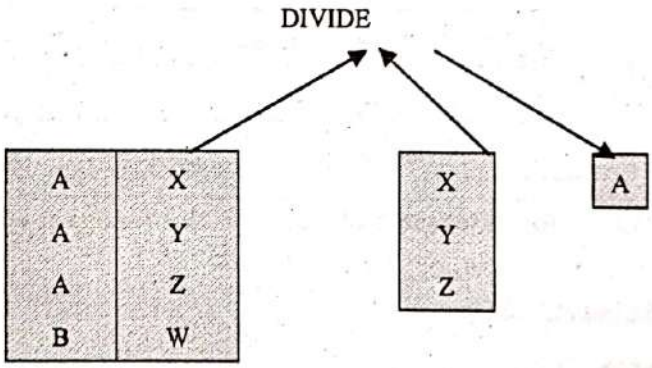
Q.7 Explain Division Relational algebra operators with suitable examples. (2 Marks)

Ans. :

- The relational divide operator (so called to distinguish it from mathematical division) returns the records in one record set that have values that match all the corresponding values in the second record set.
- The output of divide operation is one column in which values of common column in both tables match.

Syntax

(Query Expression 1) (Query Expression 2)



DIVIDE Operator

Example :

Student Table			
Stud_ID	Sname	Course_ID	Gender
1	Mahesh	100	M
2	Manish	100	M
3	Amruta	100	F
3	Amruta	200	F

Student Table			
Stud_ID	**Sname**	**Course_ID**	**Gender**
6	Neesha	100	F
3	Amruta	300	F
6	Neesha	300	F
6	Neesha	200	F

Course Table contains all courses that trainer 401 is taking.

Course_ID	Trainer_ID
100	401
200	401
300	401

Find female students take ALL the courses that 401 are taking.

Student ÷ Course

OR

π s.Stud_ID, s.Sname, s.Gender, s.Course_ID (ρ_s(Student) ÷ ρ_c(Course))

s.Stud_ID	s.Sname	s.Gender	s.Course_ID
3	Amruta	F	200
6	Neesha	F	100

Q. 8 Consider the following relations for database that keeps track of student enrollment in courses and books issued for each course.

STUDENT (Ssn, Name, Subject, DOB)

COURSE (Course_id, Name, Dept)

ENROLL (Ssn, Course_id, Semester, Grade)

Book_issued (Course_id, Semester, ISBN)

TEXT (ISBN, Title, Publisher, Author)

Write any 5 Queries in relational algebra. **(10 Marks)**

Ans. :

(1) A query to select all courses available in institute : $\Pi_{course\_id, CName, Dept}$ (COURSE)

(2) All student details registered for course id 10 : $\Pi_{Ssn, Name}$ ($\sigma_{course\_id = 10}$(ENROLL ⋈ STUDENT))

(3) Various book titles and authors for semester higher than 3 : $\Pi_{ISBN, Title, Author}$ ($\sigma_{semester > 3}$(Book_Issed ⋈ TEXT))

(4) All students belonging to IT Department.

(a) To find course_id of 'IT' Department

$$T_1 \leftarrow \Pi_{course\_id,}\ (\sigma_{Dept = 'IT'}\ (COURSE))$$

(b) To find all students enrolled for above course id.

$$T_2 \leftarrow \Pi_{ssn}\ , (ENROLL \bowtie T_1)$$

(c) To find student details having above Ssn.

$$Ans \leftarrow \Pi_{Ssn,\ Name,\ DOB}\ (STUDENT \bowtie T_2)$$

Q. 9 Consider the relations given below :

Dealer (Dealer-no, DealerName, address)

Part (Part-no, Part-name, color)

Assigned-to (Dealer-no, Part-no, cost)

Give an expression in relational algebra the following queries :

(i) Find the name of all dealers who supply 'Red' Parts.

(ii) Find the name of the dealers who supply both Yellow and Green Parts.

(iii) Find the name of the dealers who supply all the Parts.

(iv) List all dealer names. **(10 Marks)**

Ans. :

(i) The name of all dealers who supply 'Red' Parts.

$$\Pi_{Dealername}\ (\sigma_{course='red'}\ (Dealer \bowtie Part))$$

(ii) The name of the dealers who supply both Yellow and Green Parts

$$\Pi_{Dealername}\ (\sigma_{course='red'\ OR\ course='yellow'}\ (Dealer \bowtie Part))$$

(iii) The name of the dealers who supply all the Parts.

$$\Pi_{Dealername}\ (Dealer \bowtie Part))$$

(iv) The list of all dealer names

$$\Pi_{Dealername}\ (Dealer)$$

Chapter 6 : Structured Query Language

Q.1 Explain role of SQL with example. **(3 Marks)**

Ans. :

- SQL is an interactive query language which can be used to retrieve data from database. SQL is a database programming language which can be used along with programming language to access data from database.
- SQL is a database administration language which can be used to monitor and control data access by various users. SQL can be used as an Internet data access language.

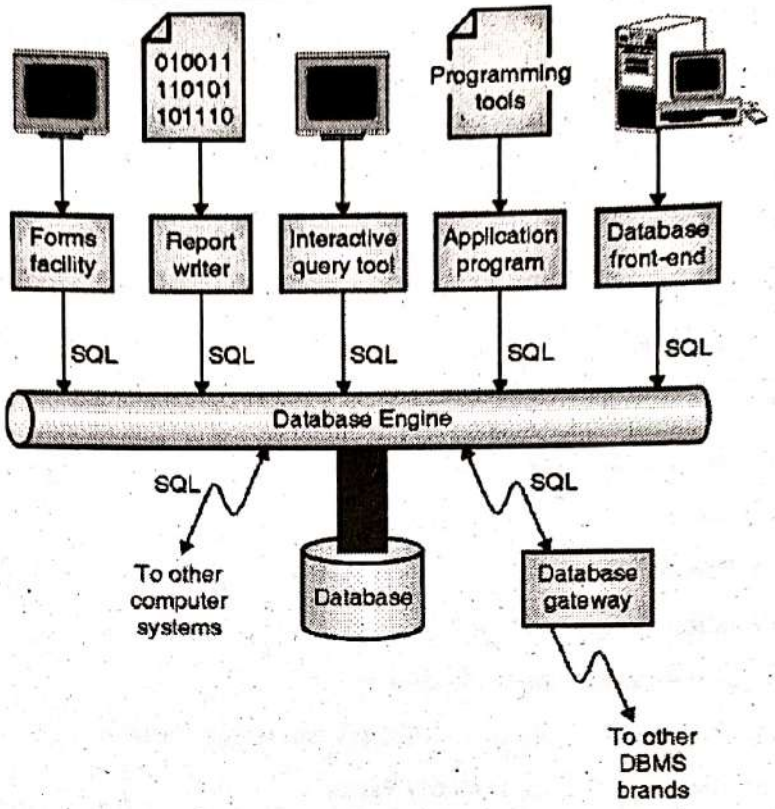


Fig. 6.1: Role of SQL in DBMS

Q.2 Explain various SQL data types with example. **(4 Marks)**

Ans. : Data types include numeric, character string, bit string, Boolean and date time.

1. **Numeric data types :** This data type is used to store a number values that can be decimal or floating point values.
   - (a) **Integer number of various size :** These types of system are used to store natural numbers which are not having any decimal values.
   - (b) **Floating point numbers of various precision :** This system is used for storing decimal numbers which may be of greater size than integers.
   - (c) **Formatted numbers :** This system used for storing some special numbers which may be of greater size than integers and floating-point numbers.

2. **Character string data type**

   This data type is used to store a character string which is combination of some alphabets and enclosed in single quotation marks.
   - (a) **Fixed length :** CHAR (n), Where n = number of characters
   - (b) **Varying length :** VARCHAR (n) Where n = maximum number of characters

3. **Date time data type**

(a) **Date**

- The DATE data type has ten positions, and its components are YEAR, MONTH and DAY in form YYYY-MM-DD.
- The length is 10.
- Generally not supported by SQL server.(Supported by DB2)

(b) **Time**

- The TIME data type has at least eight positions, and its components are HOUR, MINUTES and SECOND in form HH:MM:SS [.sF] where F is the fractional part of the SECOND value.

- Generally not supported by SQL server.
- If a second's precision is not specified, s defaults to 0. The length is 8 (or $9 + s$, if $s > 0$).

(c) Timestamp / date time

- The TIMESTAMP data type includes both date and time fields, plus a minimum of six positions and for decimal fractions of second and optimal with TIMEZONE Qualifier.
- Represented using the fields YEAR, MONTH, DAY, HOUR, MINUTE and SECOND in the format YYYY-MM-DD HH:MM:SS[.sF] where F is the fractional part of the SECOND value.
- If a second precision is not specified, s defaults to 6. The length is 26 (or 19, if s = 0 or $20 + s$, if $s > 0$).

(d) Interval

This specifies an interval a relative value that can be used to increment or decrement an absolute value of date, time or timestamp.
INTERVAL YEAR TO MONTH Datatype

Q.3 Explain Database Languages. **(2 Marks)**

Ans. :

- The set of DDL commands are as below,
  1. CREATE Statement : To create Database objects
  2. ALTER Statement : To modify structure of database objects
  3. DROP Statement : To remove database objects
  4. RENAME Statement : To Rename Database objects
  5. TRUNCATE Statement : To empty the database table
- When you execute a DDL statement, it takes effect immediately, as it is **Autocommited** into database. Hence no rollback operation (Undo) can be performed with these set of commands.
- Database objects are any data structure created in database.

Example : Table, View, Sequence etc.

Q.4 Explain CREATE command with example. **(4 Marks)**

Ans. :

CREATE statement is used to create new database objects like table, index and others. This statement used to create database object.

Syntax

```
CREATE TABLE <Table_Name>
        ( Column_1 datatype,
        Column_2 datatype,
        ....
        Column_n datatype
    );
```

Example :

```
SQL> CREATE TABLE Employee
        (   Eid   INT,
        Name      VARCHAR (20),
        Age       INT,
        Address   CHAR (25),
        Salary    DECIMAL (18, 2)
    );
Query OK, 0 rows affected (0.01 sec)
```

To view the structure of newly created table.

```
SQL> DESC Employee;
+---------+-----------------+------+------+---------+------+
| Field   | Type            | Null | Key  | Default | Extra |
+---------+-----------------+------+------+---------+------+
| EID     | int(10)         | YES  |      | NULL    |      |
| NAME    | varchar(20)     | YES  |      | NULL    |      |
| AGE     | int(11)         | YES  |      | NULL    |      |
| ADDRESS | char(25)        | YES  |      | NULL    |      |
| SALARY  | decimal(18,2)   | YES  |      | NULL    |      |
+---------+-----------------+------+------+---------+------+
5 rows in set (0.00 sec)
```

Q.5 Explain ALTER command with example. **(4 Marks)**

Ans. :

- Once database object is created in database, require ALTER command to update structure of database object.
- The ALTER TABLE statement can be used to add, delete, or modify columns in an existing table.
- The ALTER TABLE command can also be used to add and drop various constraints on an existing table.

Syntax

```
ALTER TABLE <Table_Name>
    ADD Column_1 datatype;
ALTER TABLE <Table_Name>
    Modify Column_1 New_datatype;
ALTER TABLE <Table_Name>
    DROP Column_1;
```

Example :

```
SQL> ALTER TABLE Employee
        ADD Address VARCHAR (100);
Query OK, 0 rows affected (0.01 sec)
```

To view the changed structure of table

```
SQL> DESCRIBE TABLE Employee;
+---------+-----------------+------+------+---------+------+
| Field   | Type            | Null | Key  | Default | Extra |
+---------+-----------------+------+------+---------+------+
| EID     | int(10)         | YES  |      | NULL    |      |
| NAME    | varchar(20)     | YES  |      | NULL    |      |
| AGE     | int(11)         | YES  |      | NULL    |      |
| ADDRESS | varchar(100)    | YES  |      | NULL    |      |
| SALARY  | decimal(18,2)   | YES  |      | NULL    |      |
+---------+-----------------+------+------+---------+------+
5 rows in set (0.00 sec)
```

Q.6 Explain RENAME command with example.

Ans. :

- It is possible to change name of table with or without data in it using simple RENAME command.
- Can rename any table object at any point of time.

Syntax

```
RENAME TABLE <Table_Name> To <New_Table_Name>;
```

Example :

```
SQL> RENAME TABLE Employee To EMP;
```

Q.7 Explain TRUNCATE command with example. **(4 Marks)**

Ans. :

- The TRUNCATE TABLE command is used to delete all data from an existing table.
- It is possible to do same action with DROP TABLE command but it would remove complete table structure from the database.
- A DELETE command will also remove all data from table but with DELETE data deletion can be rolled back and truncate acts as permanent data deletion with no roll back possible.
- If any delete triggers are defined on the table, then the triggers are not fired on truncate table.
- Truncate will de-allocates memory space. So that the free space can be used by other tables unlike DELETE command.

Syntax

```
TRUNCATE TABLE <Table_Name>;
```

Example :

```
SQL> TRUNCATE TABLE EMP;
```

Q.8 Explain DROP command with example. **(4 Marks)**

Ans. :

- Drop command can be used to remove database any objects from user database.
- The SQL DROP TABLE statement is used to remove a table definition and all related data like indexes, triggers, constraints and permission specifications for that table.
- The developer must be careful while running this command because once a table is dropped then all the information available in that table will also be lost forever and no roll back can be done.

Syntax

```
DROP TABLE <Table_Name>;
```

Example :

If want to permanently remove the Employee table that we created, we'd use the following command

```
SQL> DROP TABLE Employee;
Query OK, 0 rows affected (0.01 sec)
```

Q.9 Explain DML commands with syntax. **(10 Marks)**

Ans. :

- **Data Manipulation Language (DML)** statements are used for manipulating or managing data in database.
- DML commands are not auto-committed like DDL statements.
- It means changes done by DML command can be rolled back. Or in other words the DML statements do not implicitly commit the current transaction.

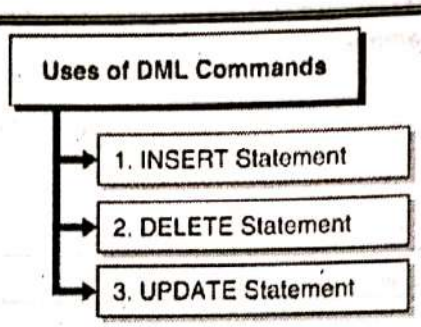


Fig. 6.2 : Uses of DML commands

1. INSERT Statement

- Insert statement used to add records to the existing table.
- To insert data into a table, SQL INSERT INTO command can be used.
- To insert few values in table as per columns names,can use generic syntax as,

```
INSERT INTO <Table_Name> (Column1, . . . , ColumnN)
VALUES (column1. . . , columnN);
```

- But, need to make sure the order of the values is in the same order as the columns in the table.

```
INSERT INTO <Table_Name>
VALUES (column1. . . , columnN);
```

2. DELETE Statement

- To delete data into a table, SQL DELETE command can be used.
- To delete all rows in table can use generic syntax as,

Syntax

```
DELETE
FROM <Table_Name>;
```

To delete selected rows from table , specify the WHERE condition.

```
DELETE
FROM <Table_Name>
WHERE <Condition>;
```

UPDATE Statement

- To update data in a table, SQL UPDATE command can be used.
- To update all rows in table ,can use generic syntax as,

```
UPDATE <Table_Name>
SET    column1 = new_value;
```

To update selected rows from table , specify the WHERE condition in Update statement.

```
UPDATE <Table_Name>
SET    column1=new_value
WHERE condition;
```

Q.10 Write a note on DCL. (4 Marks)

Ans. :

- **Data Control Language (DCL)** is used to control various user actions (or privileges) in Database.
- DCL is set of commands used to,
  - **Grant :** Gives some privilege to user for performing task on database.
  - **Revoke :** Take back permissions given from user.
- Privileges can be of many types,
  - **System Privileges :** creating a table is types of system privilege.
  - **Object Privileges :** To execute query on tables object privilege can be used.
  - **Ownership Privileges :** To execute query on tables created by same user.

Q. 11 Enlist the privileges with suitable example.(6 Marks)

Ans. :

- The set of actions that a user can perform on a database object are called the privileges.
- Privilege is right to execute particular SQL statement on database.

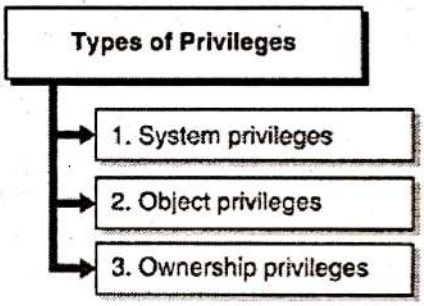


Fig. 6.3 : Types of Privileges

1. System privileges

- System privileges are rights and restriction that are implemented on databases to control which users can access how much data in the database.
- User requires system privileges to gain access to database.
- Few system privileges are as below,

System privileges	Authorized to
CREATE USER	Create number of users in DBMS
DROP USER	Drop any other users in DBMS
CREATE ANY TABLE	Create table object in any schema.
SELECT ANY TABLE	Query table object or view in any schema.
DROP ANY TABLE	Drop table object in any schema.

2. Object privileges

- Object privileges are rights and restrictions to change contents of database objects.
- User requires object privileges to manipulate the content of object within database.
- The user which has GRANT ANY PRIVILEGE system privilege granted to him then he can act like administrator to control database modifications.
- Few object privileges are as below,

Object privileges	Authorized to
SELECT	Select rows from table or view
INSERT	Add new rows to table or view
DELETE	Remove some rows from table or view
UPDATE	Modify content of rows from table or view
EXECUTE	To run procedure
REFERENCES	To reference a particular table using foreign key and check constraint

3. Ownership privileges

- Whenever you create a database object (like table or view) with the CREATE statement, you will become its owner and get full privileges for the table. (Like SELECT, INSERT, DELETE, UPDATE, and all other privileges).
- You as owner of database object can explicitly give grant privileges to any other user by using the GRANT statement.
- Whenever you are creating a view with the CREATE VIEW statement, you become the owner of that view, but you do not necessarily receive all privileges as you require the SELECT privilege on each of base tables on which view is defined.

Q. 12 Write syntax for GRANT privileges. **(2 Marks)**

Ans. : GRANT Privileges

- GRANT statement is used by owner of table or view to give other users access permissions.
- In SQL user accounts must present in system before grant privileges to him.

Syntax

```
Grant <ALL | privilege list>
ON <relation name or view name>
TO <user | role list | PUBLIC >
    [WITH GRANT OPTION]
```

Q. 13 Write a short note on Revoking of privileges. **(4 Marks)**

Ans. :

- Can reject the privileges given to particular user with help of revoke statement.
- To revoke an authorization, use the **revoke** statement.

Syntax

```
REVOKE    <ALL | privilege list>
ON  <relation name or view name>
FROM   <user | role list | PUBLIC>
[RESTRICT/ CASCADE]
```

- **CASCADE :** will revoke all privileges along with all dependent grant privileges
- **RESTRICT :** This will not revoke all related grants only removes that GRANT only.

Examples :

- The revocation of privileges from user or role may cause other user or roles also have to leave that privilege.
- This behaviour is called cascading of the revoke.

Q.14 For the given database, write SQL queries.
Employee (Eid. Name, Street, City)
Works (Eid. Cid. salary)
Manager (Eid, Manager_Name)
Company(Cid, Company_name, city) **(5 Marks)**

Ans. :

(i) Modify the database so that 'Jack' now lives in 'Newyork'.

```
MySQL> UPDATE Employee
       SET          City = 'Newyork'
       WHERE  Name = 'Jack';
```

(ii) Give all employees of 'ANZ corporation' a 10% raise in salary.

```
MySQL> UPDATE Works
SET    Salary = (salary+(0.1*salary))
WHERE  CID IN ( SELECT Cid
FROM   Company
WHERE  Company_name = 'ANZ corporation');
```

Q.15 For the following given database ? Write SQL queries

person (driver_Id #, name, address)
car (license, model, year)
accident (report_no, date, location)
owns (driver_Id #, license)
participated (driver_id, car, report_number, damage_amount) **(5 Marks)**

Ans. :

(i) Update the damage amount for car with licence number "Mum2022" in the accident with report number "AR2197" to Rs. 5000.

```
MySQL> Update   Participated
       SET      Damage_amount = 500
       WHERE    Report_number LIKE 'AR2197' AND Car = 'Mum2022';
```

Q. 16 For given database, write SQL queries.

Employee (EID, Name, Street, City)
Works (EID, CID, Salary)
Manager (EID, Manager_Name)
Company (CID, Company_name, City) **(5 Marks)**

Ans.:

(i) Modify the database so that 'TRATHAM' now lives in USA

```
MySQL>  UPDATE Employee
        SET    City = 'USA'
        WHERE  Name = 'TRATHAM';
```

(ii) Give all employees of 'SHARAYU Steel' a 10% raise in salary.

```
MySQL> UPDATE works
       SET          salary = (salary + (0.1 * salary))
       WHERE  cid IN (SELECT cid
                      FROM company
                      WHERE company_name = 'SHARAYU Steel');
```

Q. 17 Consider Insurance database given below and answer the following queries in SQL.

Person (driver_Id, name, address)
Car (license, model, year)
Accident (report_no, adate, location)
Owns (driver_Id, license)
Participated (driver_Id, license, report_no, damage_amount) **(5 Marks)**

Ans. :

(i) Add new accident to database.

```
MySQL> INSERT INTO Accident (report_no, adate, location)
       VALUES ('111', '01/01/2014','Pune');
```

(ii) Delete 'Santro' belonging to 'John Smith'.

```
MySQL> DELETE
       FROM    CAR
       WHERE  Model = 'SANTRO'
       AND
       License IN ( SELECT license
                    FROM  Owns
                    WHERE  Driver_id IN ( SELECT driver_id
                                          FROM person
                                          WHERE name = 'John Smith')
                  );
```

Q.18 Consider the following employee database.
Employee (empname, street, city, date_of_joining)
Works (empname, company _name, salary)
Company (company_name, city)
Manages (empname, manager_name).
Write SQL queries for the following statements :
(i) Modify the database so that 'John' now lives in'Mumbai'.
(ii) Give all employees of ABC Corporation' a 10% raise

(5 Marks)

Ans. :

(i) Modify the database so that 'John' now lives in 'Mumbai'.

```
MySQL> UPDATE Employee
       SET     City= 'Mumbai'
       WHERE  Empname= 'JOHN';
```

(ii) Give all employees of ABC Corporation' a 10% raise.

```
MySQL> UPDATE Works
       SET     Salary=0.1*salary
       WHERE  Company_name= 'ABC Corporation';
```

Q. 19 Employees (Empid, Fname, Lname, Email, Phoneno, Hiredate, Jobid, Salary, Mid, Did)
Departments (Did, Dname, Managerid, Locationid)
Locations (Locationid, Streetadd, Postalcode, City)
Write the SQL queries for the following.

1. **List the employees have a manager who works for a department based in the U.S.**
2. **Write a query to display the details of all employees in the Finance department.**
3. **Give 10% hike to all the employees working in Did 20.**
4. **Write a query to display all the information of the employees whose salary is within the range 1000 and 3000.**
5. **Display the information of all the employees whose first name starts with 'R' in descending order of their salary.**

Ans.: **(10 Marks)**

1. The employees have a manager who works for a department based in the U.S.

```
SELECT *
FROM Employees e
INNER JOIN Departments d ON e.did = d.did
INNER JOIN Locations l ON l.locationid=d.locationid
WHERE City ='US';
```

2. A query to display the details of all employees in the Finance department.

```
SELECT *
FROM Employees e
INNER JOIN Departments d ON e.did = d.did
WHERE Dname='Finance';
```

3. Give 10% hike to all the employees working in Did 20.

```
UPDATE employees
SET Salary = 1.1* Salary
WHERE did =20;
```

4. A query to display all the information of the employees whose salary is within the range 1000 and 3000.

```
SELECT *
FROM employees
WHERE Salary BETWEEN 1000 AND 3000;
```

5. Display the information of all the employees whose first name starts with R' in descending order of their salary.

```
SELECT *
FROM employees
WHERE Fname LIKE 'R%'
ORDER BY Salary DESC;
```

Q.20 **Employee(eid,ename,address,city)**
Works(eid,cid,salary)
Company(cid,cname,city)

(1) Modify database so that John now lives in Mumbai
(2) Find Employees who live in same city as the company for which they work.
(3) Give all employees of "AZ Corporation" where there is increase in salary by 15%
(4) Find the names of all employees, company name and city of residence such that Employee name begins with 'I'
(5) Delete all tuples in works relation for employees of small bank corporation. **(10 Marks)**

Ans. :

1. Update employee

```
Set city = 'Mumbai'
Where name = 'John';
```

2. Sectect e-ename

```
From   employee e, company c, works w
When   e.eid = w.eid
And    w.Cid = c.cid ,
And    e.city = C.city ;
```

3. update employee

```
Set    sal = 1.15 * sal
Where eid = (Sected eid from emp
             where Cid = (select eid
```

```
                from comp
                where name)
```

4. select e.ename e.city, c. cname

```
select  from e.city,
select where e.city
select  And e.city
And e name like 'J'
```

5. delete

```
from employ
When eid = (Sectect eid
        Form works
        Where Cid = (Sectect Cid from comp
```

Chapter 7 : SQL Security

Q.1 Write a note on various aggregate function. **(10 Marks)**

Ans. :

- Sometimes for decision making need summarize data from table like average, sum, minimum etc.
- SQL provides various aggregate functions which can summarize data of given table. The function operates on the table data produce a single output.

Example :

Table 7.1 : Exam_Marks

Sid	SName	Marks
1	Mahesh	90
2	Suhas	80
3	Jyendra	89
4	Sachin	99
5	Vishal	88
6	Payal	90

Types of aggregate functions

1. **COUNT ()**

- This function is used to calculate number of rows (or records) in a table selected by query.
- COUNT returns the number of rows in the table when the column value is not NULL.
- Column in the query must be numeric.

Example :

Find total number of students in above Table 7.1.

```
SELECT Count(Sid) as Count
FROM   Exam_Marks;
```

COUNT
6

2. **SUM ()**

- This function is used to calculate sum of column values in a table selected by query.
- Column in the query must be numeric.
- Value of the sum must be within the range of that data type.

Example :

Find total of marks scored by all students.

```
SELECT SUM(Marks) as Sum
FROM   Exam_Marks;
```

SUM
446

3. **AVG ()**

- This function is used to calculate Average of column values in a table selected by query.
- This function first calculates sum of column and then divide by total number of rows.
- AVG returns the average of all the values in the specified column.
- Column in the query must be numeric.

Example :

Find avarage marks of students.

```
SELECT AVG(Marks) as AVG
FROM   Exam_Marks;
```

AVG
89.33

4. **MIN()**

- This function is used to find minimum value out of column values in a table selected by query.
- Column in the query need not be numeric data type.

Example :

Find minimum marks scored by students.

```
SELECT MIN(Marks) as Min
FROM   Exam_Marks;
```

MIN
80

5. **MAX()**

- This function is used to find maximum value out of column values in a table selected by query.
- Column in the query need not be numeric data type.

Example :

Find maximum marks scored by students.

```
SELECT MAX(Marks) as Max
FROM   Exam_Marks;
```

MAX
80

Q. 2 Compare various Aggregate functions. **(5 Marks)**

Ans. :

Function	Description
AVG ([DISTINCT | ALL] n)	Average value of n, ignoring null values.
COUNT ((* | [DISTINCT | ALL] expr))	Number of rows, where *expr* evaluates to something other than null.
MAX ([DISTINCT | ALL] expr)	Maximum value of *expr*, ignoring null values.
MIN ([DISTINCT | ALL] expr)	Minimum value of *expr*, ignoring null values.
SUM ([DISTINCT | ALL] n)	Sum value of n, ignoring null values.

Q. 3 Describe view. **(10 Marks)**

Ans. :

Definition

A view is defined as a database object that allows us to create a virtual table in the database whose contents are defined by a query or taken from one or more tables.

1. **Base table**

   The table on which view is defined is called as Base table.

2. **View - As a window of entire table**

   Instead of showing entire table to a user , show a glimpsed of table to the user which is required for him

Example :

Consider a student table contains following columns,

STUDENT (Stud_Id, Stud_Name, Std, Div, Addr, Sports, Fees, Cultural_Activity)

- Now for a sports teacher requires only sports related data of students so create view called as Stud_Sports_View for teacher as below which will only depicts sports data of student to sports teacher.

  Stud_Sports_View (Stud_Id, Stud_Name, Sports)

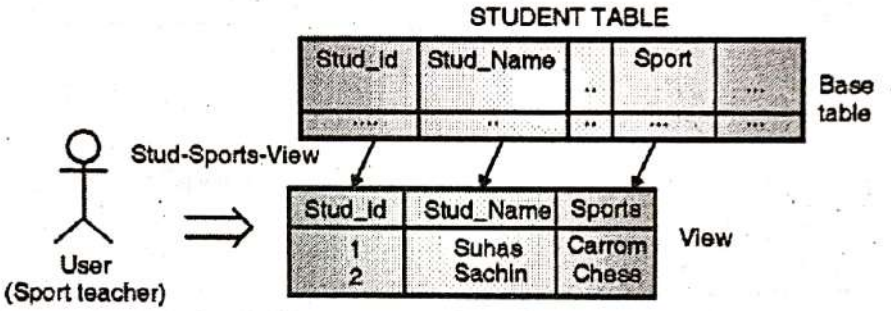


Fig. 7.1 : Overview of view

Types of views

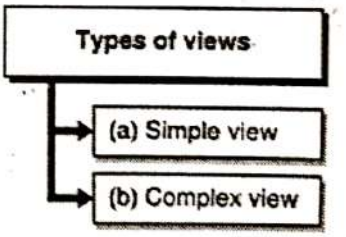


Fig. 7.2 : Types of Views

(a) Simple view

- The views which are based on only one table called as Simple view.
- Allow to perform DML **(Data Manipulation Language)** operations with some restrictions.
- Query defining simple view cannot have any join or grouping condition.

(b) Complex view

- The views which are based on more than one table called as complex view.
- Do not allow DML operations to be performed.
- Query defining complex view can have join or grouping condition.

Working of views

- When call view in SQL query it refers to database and finds definition of views which is already stored in database.
- Then the DBMS convert this call of view into equal request on the base tables of the view and carries out the operations written in view definition and returns result set to query from which view is called.

Q. 4 Explain syntax for creating views. **(4 Marks)**

Ans. :

- The CREATE statement assigns a name to the view and also gives the query which defines the view.
- To create the VIEW one should have privileges to access all of the base tables on which view is defined.

Syntax

```
CREATE [OR REPLACE] VIEW <view name>
AS
SUB QUERY
[WITH CHECK OPTION]
```

Q. 5 How to drop view ? Give Syntax. **(2 Marks)**

Ans. :

- To drop a view we use DROP VIEW statement.
- The DROP VIEW statement requires a name to the view.
- To DROP the VIEW one should have privileges to from DBA to DROP a view in database.

Syntax

```
DROP VIEW <View_name> [RESTRICT|CASCADE]
```

Example :

Remove a created view

```
SQL>    DROP VIEW JobBelow3K;
View Dropped;
```

Q. 6 Give syntax for altering views. **(2 Marks)**

Ans. :

- The CREATE OR REPLACE statement in view syntax is used to modify view.
- This statement is used by SQL to overwrite the old view definition with new definition without raising any error like existing view with same name.

Syntax

```
CREATE OR REPLACE VIEW <View_name>
AS
SUB QUERY
[WITH CHECK OPTION]
[WITH READ ONLY];
```

Q.7 What are advantages of views ? **(5 Marks)**

Ans. :

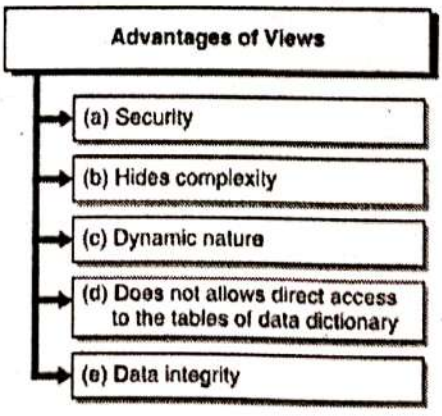


Fig. 7.3 : Advantages of Views

(a) Security

- In case of view only data that is given in view is accessible to user. So all data of base table is not accessible to user which will give you security of information.
- For example sports teacher can see data related to sports only and view preventing him from manipulating data pertaining to fees of students.

(b) Hides complexity

- The view may be result of very complex query. Hence instead of writing such complicated query again and again store such result to a view and access it whenever we want to access.
- So by writing query , hide the complexity of original query.

(c) Dynamic nature

- View definition remains unaffected although there is any change in structure of a table.
- This dynamic nature does not hold true in case if base table is dropped or the column selected by view is altered.

(d) Does not allows direct access to the tables of data dictionary

- This act like functionality of safeguard to data stored in the data dictionary.
- By this way user cannot change data dictionary to damage database.

(e) Data integrity

If data is accessed through a view, the DBMS can automatically check the data to check for specified integrity constraints.

Q. 8 What are disadvantages of views ? **(3 Marks)**

Ans. :

(a) Performance

- DBMS translates queries of view to queries on base table.
- As the complexity of query is hidden by view hence, users are not aware of how much complicated task the query is actually performing.

(b) View management

- The view should be created as per standard then it will simplify the job of DBA.
- This happens generally when views are references to the other views.
- Need to keep all information of all views in such case so as to it will become very difficult to manage views.

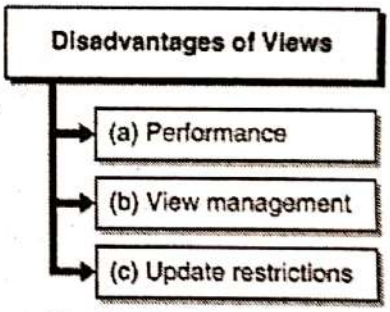


Fig. 7.4 : Disadvantages of views

(c) Update restrictions

- When a user tries to update a view, the DBMS must translate this query into an update on rows of the underlying base tables.
- Update is possible for simple views.
- Complex views cannot be updated as they are read-only type of views.

Chapter 8 : Trigger

Q.1 What are triggers? Explain with example. **(4 Marks)**

Ans. : Trigger

- A trigger is a procedure that is automatically invoked by the DBMS in response to specific alteration to the database or a table in database.
- Triggers are stored in database as a simple database object.
- A database that has a set of associated triggers is called an active database.
- A database trigger enables DBA (Database Administrators) to create additional relationships between separate databases.

Components of Trigger (E-C-A model)

- **Event (E)** - SQL statement that causes the trigger to fire (or activate). This event may be insert, update or delete operation database table.
- **Condition (C)** - A condition that must be satisfied for execution of trigger.
- **Action (A)** - This is code or statement that execute when triggering condition is satisfied and trigger is activated on database table.

Trigger syntax

```
CREATE [OR REPLACE] TRIGGER <Trigger_Name>
[<ENABLE | DISABLE>]
<BEFORE | AFTER>
<INSERT | UPDATE | DELETE>
ON <Table_Name>
      [FOR EACH ROW]
DECLARE
      <Variable_Definitions>;
BEGIN
      <Trigger_Code>;
END;
```

Trigger types

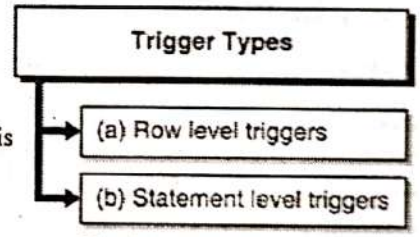


Fig. 8.1 : Trigger types

(a) Row level triggers

- A **row level trigger** is fired each time the table is affected by the triggering statement.
- For example, if an UPDATE statement changes multiple rows in a table, a row trigger is fired once for each row affected by the UPDATE statement.
- If a triggering statement do not affect any row then a row trigger will not run only.
- If FOR EACH ROW clause is written that means trigger is row level trigger.

(b) Statement level triggers

- A statement level trigger is fired only once on behalf of the triggering statement, irrespective of the number of rows in the table that are affected by the triggering statement.
- This trigger executes once even if no rows are affected.
- This is Default type, when FOR EACH ROW clause is not written in trigger that means trigger is statement level trigger.

Trigger example

Creating a trigger on employee table whenever new employee added a comment is written to EmpLog Table.

Example :

```
SQL> CREATE OR REPLACE TRIGGER AutoRecruit
  2  AFTER INSERT ON EMP
  3  FOR EACH ROW
  4  BEGIN
  5  Insert into EmpLog values ('Employee Inserted');
  6  END;
  7  /
```

```
Trigger created.
SQL> INSERT INTO EMP
2 VALUES
3 (1,'Mahesh','Manager','1-JAN-1986',3000,null,10);
1 row created.
SQL> SELECT * FROM EmpLog;
STATUS
----------
Employee Inserted
```

Trigger operations

(a) Data dictionary for triggers

Once triggers are created their definitions can be viewed by selecting it from system tables as shown below :

Syntax

```
MySQL>Select *
      From    User_Triggers
      Where   Trigger_Name = '<Trigger_Name>';
```

This statement will give you all properties of trigger including trigger code as well.

(b) Dropping Triggers

To remove trigger from database we use command DROP

Syntax

```
MySQL> Drop trigger <Trigger_Name>;
```

(c) Disabling Triggers

To deactivate trigger temporarily this can be activated again by enabling it.

Syntax

```
MySQL> Alter trigger <Trigger_Name> {disable | enable};
```

Chapter 9 : Relational Database Design

Q. 1 What is Normalization ? **(2 Marks)**

Ans. : Normalization

Normalization is a process of designing a consistent database by minimizing redundancy and ensuring data integrity through decomposition which is lossless.

Goals of Database Normalization

1. Ensures data integrity.
2. Prevents redundancy in data.
3. To avoid data anomaly
   (a) Update anomaly (b) An insertion anomaly (c) Deletion anomaly

Q. 2 Explain concept of functional dependency. (3 Marks)

Ans. :

- Functional Dependency (FD) provides a constraint between various attributes of a relation.
- Functional dependencies are restrictions imposed between two set of attributes in relation from a database.
- In a Relation R with attributes X and Y represented as R(X, Y), where Y is functionally dependent on other column X or X functionally determines Y.
- This dependency can be denoted with help of arrow (→)

$$X \rightarrow Y$$

- The data value in column Y must change when data value in another column X is modified.

All the attributes before arrow is called as **determinant** and attributes after arrow is called as determine.

Example :

Consider an employee table with columns as shown in Table 9.1

Table 9.1 : Employee Table

Employee_Id	Ename	Salary	Project_Id	Hours	Allowance
10	Mahesh	50000	E001	44	40000
12	Suresh	25000	B056	31	30000
15	Ganesh	26000	C671	23	20000
18	Mahesh	50000	E002	12	15000

Case 1 : (X→Y)

- Consider an Employee table for specific employee_Id there is one and only one Ename whereas for another employee_Id there can be other Ename.

  Employee_Id → Ename
- As per above constraint, it is possible to have multiple employees with same Ename and different Employee_Id. But it is not allowed to have two employees with same Employee_Id and different Ename.

Case 2 : (X→ YZ)

- In above Employee table using below given functional dependency, for specific employee_Id there is one and only one set of Ename and Salary whereas for another employee_Id there can be other values of Ename and salary.

  Employee_Id → Ename, Salary
- As per above constraint, it is possible to have multiple employees with same Ename and Salary.

Case 3 : (XY→ ZW)

- In above Employee table using below given functional dependency, for one employee_Id and Project_Id pair there is only one amount of time spent (Hours) and allowance given by company whereas for another pair there can be other values of Hours and Allowance.

  Employee_Id, Project_Id → Hours, Allowance
- As per above constraint, it is possible to have multiple employee_Id and Project_Id pairs with same values of Hours and Allowance.

Q. 3 List all functional dependencies satisfied by the relation. (5 Marks)

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

Ans. :

To find out all functional dependencies satisfied by the relation, we must remember one determinant (attributes before arrow) will give one and only one value of determine (attribute after arrow).

A	B	C	Tuple
a_1	b_1	c_1	Tuple 1
a_1	b_1	c_2	Tuple 2
a_2	b_1	c_1	Tuple 3
a_2	b_1	c_3	Tuple 4

Step 1 : Test for type 1 FD (X→Y)

Functional Dependency	Violated First by Tuple	Result
$A \rightarrow B$	--	Valid FD
$A \rightarrow C$	Tuple 2 $a_1 \rightarrow c_1$ $a_1 \rightarrow c_2$	Not Valid FD
$B \rightarrow A$	Tuple 3	Not Valid FD
$B \rightarrow C$	Tuple 2	Not Valid FD
$C \rightarrow A$	Tuple 3	Not Valid FD
$C \rightarrow B$	--	Valid FD

Step 2 : Test for type 1 FD (X→YZ)

Functional Dependency	Violated First by Tuple	Result
$A \rightarrow BC$	Tuple 2 $a_1 \rightarrow b_1c_1$ $a_1 \rightarrow b_1c_2$	Not Valid FD
$B \rightarrow AC$	Tuple 2	Not Valid FD
$C \rightarrow AB$	Tuple 3	Not Valid FD

Step 3 : Test for type 1 FD (XY→Y)

Functional Dependency	Violated First by Tuple	Result
$AB \rightarrow C$	Tuple 2 $a_1b_1 \rightarrow c_1$ $a_1b_1 \rightarrow c_2$	Not Valid FD
$AC \rightarrow B$	--	Valid FD
$BC \rightarrow A$	Tuple 3	Not Valid FD

Therefore, all FDs satisfied by relation are,

$A \rightarrow B$; $C \rightarrow B$; $AC \rightarrow B$

Q. 4 **Consider the following relation.**

A	B	C	Tuple #
10	b_1	c_1	#1
10	b_2	c_2	#2
11	b_4	c_1	#3
12	b_3	c_4	#4
13	b_1	c_1	#5
14	b_3	c_4	#6

Given the previous state which of the following dependencies may hold in the above relation ? If the dependency cannot hold explain why by specifying the tuples that cause the violation :

(i) $A \rightarrow B$ **(ii)** $B \rightarrow C$ **(iii)** $C \rightarrow B$

(iv) $B \rightarrow A$ **(v)** $C \rightarrow A$ **(5 Marks)**

Ans. :

To find out all functional dependencies satisfied by the relation, one determinant (attributes before arrow) will give one and only one value of determine (attribute after arrow).

(i) $A \rightarrow B$

- As in above relation to test $A \rightarrow B$, $\{10\} \rightarrow \{b1\}$ but in Tuple # 2 $\{10\} \rightarrow \{b2\}$
- This violates dependency.
- **Therefore,** Dependency cannot hold in above table values due to Tuple #2.

(ii) $B \rightarrow C$

- As in above relation to test $B \rightarrow C$, $\{b1\} \rightarrow \{c1\}$, $\{b2\} \rightarrow \{c2\}$, $\{b3\} \rightarrow \{c4\}$ for all tuples.
- **Therefore,** Dependency holds for all tuples in above table.

(iii) $C \rightarrow B$

- As in above relation to test $A \rightarrow B$, $\{c1\} \rightarrow \{b1\}$ but in Tuple #3 $\{c1\} \rightarrow \{b4\}$
- This violates dependency.
- **Therefore,** Dependency cannot hold in above table values due to Tuple #3.

(iv) $B \rightarrow A$

- As in above relation to test $A \rightarrow B$, $\{b1\} \rightarrow \{10\}$ but in Tuple #5 $\{b1\} \rightarrow \{13\}$
- Also $\{b3\} \rightarrow \{13\}$ but in Tuple #6 $\{b3\} \rightarrow \{14\}$
- This violates dependency.
- **Therefore,** Dependency cannot hold in above table values due to Tuple #5 and Tuple #6.

(v) $C \rightarrow A$

- As in above relation to test $C \rightarrow A$, In Tuple #1 $\{C1\} \rightarrow \{10,11,13\}$
- But in Tuple #3 $\{c1\} \rightarrow \{11\}$ and in Tuple #5 $\{c1\} \rightarrow \{13\}$
- This violates dependency.
- **Therefore,** Dependency cannot hold in above table values due to Tuple #3 and Tuple #5.

Q. 5 **List the Armstrong's axioms for functional dependencies. What do you understand by soundness and completeness of these axioms?** **(5 Marks)**

Ans. :

Axioms are nothing but rules of inference which provides a simple technique for reasoning about functional dependencies.

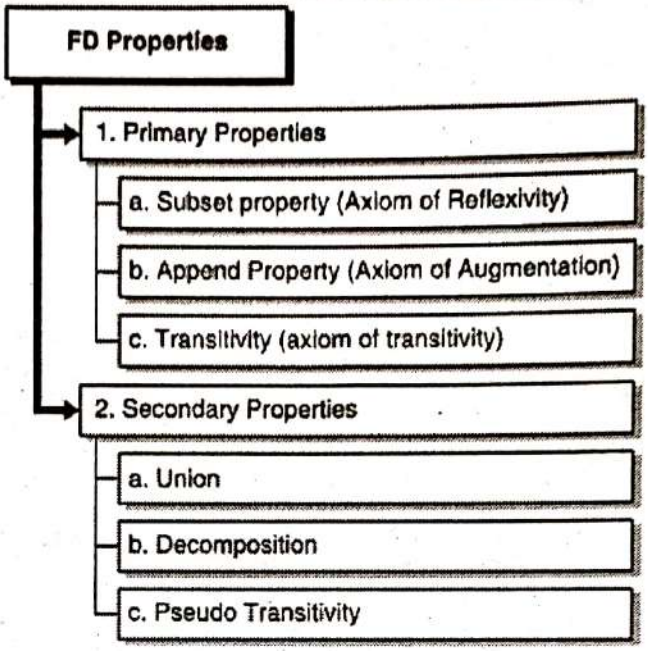


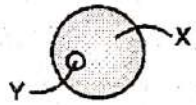
Fig. 9.1 : FD Properties

(1) Primary Properties

a. Subset property (Axiom of Reflexivity)

For given relation R(X, Y, Z, W),

If Y is a subset of X as shown in diagram,



Then **X → Y**

(Which can be referred as X is functionally dependent on Y)

b. Append Property (Axiom of Augmentation)

For given relation R(X, Y, Z, W),

If X → Y

Then XZ → YZ

It is possible to append attribute Z to both sides of FD provided that it is part of same table.

c. Transitivity (axiom of transitivity)

For given relation R(X, Y, Z, W),

If X → Y and Y → Z

Then X → Z

It is possible to use transitivity if attribute X, Y and Z are part of the same table.

(2) Secondary Properties

a. Union

For given relation R(X, Y, Z, W),

If X → Y and X → Z

Then **X → YZ**

b. Decomposition

For given relation R(X, Y, Z, W),

If X → YZ

Then $X \rightarrow Y$ and $X \rightarrow Z$

c. **Pseudo Transitivity**

For given relation R(X, Y, Z, W),

If $X \rightarrow Y$ and $YZ \rightarrow W$

Then $XZ \rightarrow W$

Q. 6 Consider relation R = (A, B, C, D, E, F) having set of FD's

$A \rightarrow B$ $A \rightarrow C$

$BC \rightarrow D$ $B \rightarrow E$

$BC \rightarrow F$ $AC \rightarrow F$

Calculate some members of Axioms as be below:

(I) $A \rightarrow E$ (II) $BC \rightarrow DF$

(III) $AC \rightarrow D$ (IV) $AC \rightarrow DF$ **(3 Marks)**

Ans. :

(i) $A \rightarrow E$

As $A \rightarrow B$ and $B \rightarrow E$

So using Transitive rule,

$\therefore$ **$A \rightarrow E$**

(ii) $BC \rightarrow DF$

As $BC \rightarrow D$ and $BC \rightarrow F$

So using union rule,

$\therefore$ **$BC \rightarrow DF$**

(iii) $AC \rightarrow D$

As $A \rightarrow B$ and $BC \rightarrow D$

So using pseudo transitivity,

$\therefore$ **$AC \rightarrow D$**

(iv) $AC \rightarrow DF$

As $AC \rightarrow D$ and $AC \rightarrow F$

So using union rule,

$\therefore$ **$AC \rightarrow DF$**

Q. 7 Consider relation R = (A, B, C, D, E, F) having set of FD's

$A \rightarrow B$ $A \rightarrow C$

$C \rightarrow D$ $B \rightarrow E$

$AC \rightarrow F$

Calculate some closures as $\{A\}^+$, $\{B\}^+$, $\{AC\}^+$ and also find key of above relation. **(3 Marks)**

Ans. :

(i)	$A \rightarrow B$, $A \rightarrow C$ So using union rule, $A \rightarrow BC$	$\{A\}^+ = \{A, B, C\}$
(ii)	$B \rightarrow E$	$\{B\}^+ = \{B, E\}$
(iii)	$\{A\}^+ = \{A, B, C\}$ and $C \rightarrow D$, $B \rightarrow E$ and $AC \rightarrow F$	$\{AC\}^+ = \{A, B, C, D, E, F\}$
(iv)	$\{AC\}^+ = \{A, B, C, D, E, F\}$	$\{AC\}$ can determine all attributes in relation R So, $\{AC\}$ is a key of Relation R

Q. 8 What is decomposition? (3 Mark

Ans. :

Decomposition

- If a relation is not in the normal form and we wish the relation to be normalised so that some of the anomalies can be eliminated, it necessary to decompose the relation in two or more relations.
- This is a process of dividing one table into multiple tables can be done using projection operator.
- Decomposed table can be reconstructed using join operation.

Q. 9 What are goals of decomposition? (4 Mark

Ans.:

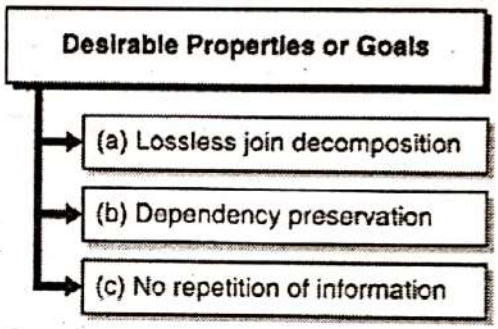


Fig. 9.2 : Desirable Properties of Goals

(a) Lossless join decomposition

- The original relation and relation reconstructed from joining decomposed relations must contain same number of tuples if number is increased or decreased then it is Losssy Join decomposition.
- Lossless join decomposition ensures that never get the situation where spurious tuple are generated in relation, for every value on the join attributes there will be a unique tuple in one of the relations.

Example :

- Employee (Employee_Id, Ename, Salary, Department_Id, Dname)
- Can be decomposed using lossless decomposition as,
- Employee_desc (Employee_Id, Ename, Salary, Department_Id)
- Department_desc (Department_Id, Dname)
- Alternatively the lossy decomposition would be as joining these tables is not possible so not possible to get back original data.
- Employee_desc (Employee_Id, Ename, Salary)
- Department_desc (Department_Id, Dname)

Rules for lossless decompositions are,

(i) The relations to be decomposed must have at least one common attribute in pair of relations.

i.e. $R_1 \cap R_2 \neq \Phi$

(ii) The attributes in common must be a key for one of the relation for decomposition to be lossless.

(b) Dependency preservation

- Dependency preservation is another important requirement since a dependency is a very important constraint on the database.
- As a result of any database updates, the database should not result in illegal relation being created. Hence, our design should allow t to check updates without natural joins.
- If $X \rightarrow Y$ holds then we know that the two (sets) attributes are closely related or functionally dependent and it would be useful both attributes in the same relation so that the dependency can be checked easily.

Example :

- Consider relation R(X, Y, Z, W) that has the following dependencies F,

$$X \rightarrow Y$$

$$Y \rightarrow ZW$$

- If decompose the above relation into R1 (X, Y) and R2 (X, Z, W) the dependency $Y \rightarrow ZW$ is not preserved.

- But, If we decompose the above relation into R1 (X, Y) and R2 (Y, Z, W) the all dependencies are preserved.

(c) No repetition of information

- Decomposition that we have done should not suffer from any repetition of information problem.
- It is desirable not to have any redundancy in database.

Q.10 Write a note on : Candidate Key, Secondary Key and Super Key. (4 Marks)

Ans. :

1. Super Key

- A **superkey** of a relation is a set of attributes $S \subseteq R$ with the property that no two tuples in relation will have same combination of attribute values in S.
- If we add additional attributes to above set of attributes S, the resulting combination would still uniquely identify a single record in a table. Such augmented keys are also called as **superkey.**

Example :

- STUDENT (Id, Name, Class, Branch, Age, Address, Mobile)
- The ID is a key attribute of STUDENT table, so ID and Name can be a superkey.

2. Candidates Key

- A candidate key (CK) is a superkey with the minimal attribute from super key.
- A minimal (irreducible) superkey is called as candidate's key.
- Minimum attributes of superkey by omitting unnecessary attributes of table which are sufficient for identifying entity (row/record) uniquely are called as **candidate keys.**
- Candidate key is also a potential primary key.

Example :

Consider a Relation R (A, B, C, D, E, F) with Functional dependencies,

$A \rightarrow BC$

$AC \rightarrow DE$

$E \rightarrow F$

To find out key of relation R,

$\{A\}^+ = \{A, B, C, D, E, F\}$

$\{AC\}^+ = \{A, B, C, D, E, F\}$

$\{ACD\}^+ = \{A, B, C, D, E, F\}$

So {A}, {AC}, {ACD} are all superkey and {A} is a candidate's key.

3. Secondary Key

- If a number of candidate keys in a relation schema then one is arbitrarily selected as **primary key** and other keys are called as **secondary keys.**
- Secondary key of a table is a column or combination of some columns used for data retrieval process.

Example :

Consider a Relation R (A, B, C, D, E, F) with Functional dependencies,

$AB \rightarrow C, BC \rightarrow DE, E \rightarrow AF$

To find out key of relation R,

$\{AB\}^+ = \{A, B, C, D, E, F\}$

$\{BC\}^+ = \{A, B, C, D, E, F\}$

So {AB} and {BC} are all candidates key.

If {AB} is selected as primary key then {BC} is called as secondary key.

Q. 11 Explain 1NF. **(3 Marks)**

Ans. :

First Normal Form (1NF)

Definition

1NF states that all attributes in relation must have atomic (indivisible) values and all attribute in a tuple must have a single value from the domain of that attribute.

- A relation is in 1NF, if every row contains exactly one value for each attribute.
- In short rules for data in 1NF is,
- A column in a table should contain only indivisible data.

Example :

- Consider an employee table with columns as shown in diagram,
- The relational schema not in 1 NF is represented as,

Employee Table

Employee_Id	Ename	Salary	Ecity

- The state of Employee relational schema is as given below and it contains the Ecity which is non atomic (divisible) domain.

Table 9.5: (Non-Normalised)Employee Table

Employee_Id	Ename	Salary	Ecity
10	Mahesh	50000	Mumbai, Pune
12	Suresh	25000	Mumbai
15	Ganesh	26000	Pune
18	Kasturi	50000	Mumbai, Delhi

- To convert relational schema in 1NF, the Ecity attribute is divided in atomic domains it may introduce some data redundancy.

Table 9.6 : 1NF Employee Table

Employee_Id	Ename	Salary	Ecity
10	Mahesh	50000	Mumbai
10	Mahesh	50000	Pune
12	Suresh	25000	Mumbai
15	Ganesh	26000	Pune
18	Kasturi	50000	Mumbai
18	Kasturi	50000	Delhi

Q. 12 Explain 2NF. **(3 Marks)**

Ans. :

Second Normal Form

Definition

A relation is in 2NF, if it is in 1NF and all non-key attributes in relation are fully functionally dependent on the primary key of the relation.

Example :

Employee (Employee_Id, Ename, Salary)

Employee_Id → Ename, Salary

Table 9.7 : 2NF Employee Table

Employee_Id	Ename	Salary
10	Mahesh	50000
12	Suresh	25000
15	Ganesh	26000
18	Mahesh	50000

Q.13 Explain 3NF. **(3 Marks)**

Ans. : Definition

- A relation R is in 3NF if all non prime attributes are,
  1. Full functionally dependent on primary key.
  2. Non-transitive dependent on every key.
- A relational schema R is in 3NF, if non-trivial functional dependency X→A holds true where X is a superkey and A is a prime attribute.

Example :

- Consider an Employee table with following FDs,

Employee (Employee_Id, Ename, Salary, Department_Id)

Employee_Id → Ename, Salary, Department_Id

Table 9.8 : 3NF Employee Table

Employee_Id	Ename	Salary	Department_Id
10	Mahesh	50000	C1
12	Suresh	25000	E2
15	Ganesh	26000	C1
18	Mahesh	50000	E2

Department (Department_Id, Dname)

Employee_Id → Dname

Table 9.9 : 3NF Department Table

Department_Id	Dname
C1	IT
E2	HR

Q. 14 Describe BCNF in detail. **(3 Marks)**

Ans. : Definition

- A relation R is said to be in BCNF, if and only if every determinant is a candidate key.
- A relational schema is in BCNF, if a non-trivial functional dependency X→A is true then X is a superkey of relation R.
- In 3NF definition A should be prime attribute, which is not the case in BCNF definition.

Example :

- Consider an employee table in which employee can work in more than one department,
- The relational schema **not in 2 NF** is represented as,
- Consider an Employee table with following FDs,

  Employee_Id → Ename, Salary, Department_Id

  Department_Id → Dname
- The state of Employee relational is,

Table 9.10 : Employee Table

Employee_Id	Ename	Department_Id	Dname	Dtype
10	Mahesh	C1	IT	Technical
12	Ganesh	E2	HR	Skill
12	Ganesh	C1	IT	Technical
10	Mahesh	E2	HR	Skill
13	Satish	E1	TS	Technical

To normalize above schema to **BCNF** , decompose tables as,

Employee (Employee_Id, Ename)

Employee_Id → Ename

The determinant Employee_Id is candidate key.

Q. 15 Consider the following relation,
CAR-SALE (Car#, Date-sold, Salesman#, commission%, Discount-amt)
Assume that {Car#, Salesman#} is the primary key.
Additional dependencies are,
Date-sold → Discount_amt
Salesman# → commission%
Based on the given primary key, is this relation in 1NF, 2NF or 3NF ?
Why or why not ?
How would you successively normalize it completely ? **(5 Marks)**

Ans. :

CAR-SALE (Car#, Salesman#, Date-sold, commission%, Discount-amt)

Assuming {Car#, Salesman#} is the primary key

Therefore,

Car#, Salesman# → Date-sold, commission%, Discount-amt

Additionally,

Date-sold → Discount_amt

Salesman# → commission%

- The above relation is in 1NF as all attributes of relation are atomic domains.
- The above relation is in 2NF as primary key is assumed and all non key attributes are full functionally depends on primary key.
- The above relation is in 3NF as all non prime attributes are non-transitively depending on primary key.
- Normalized Relation

CAR-SALE (Car#, Salesman#, Date-sold, commission%, Discount-amt)

Car#, Salesman# → Date-sold, commission%, Discount-amt

Date-sold → Discount_amt

Salesman# → commission%

Q.16 Relation R(A, B, C, D, E, F, G, H, I, J). Having following set of FD, show convert table to highest normal form. AB → C, C → EF, AD → GH, G → I, H → J (5 Marks)

Ans. :

(a) 1NF

Assuming all attributes are atomic domains so relation R is in 1NF.

(b) 2 NF

$\{A, B, D\}^+ \rightarrow \{\underline{A, B, D}, C, E, F, G, H, I, J\}$

{A, B, D} is candidate key for relation R.

No attribute in relation is full functionally dependent on above key.

Therefore, Relation R is not in 2NF.

Decomposition to 2NF

R_1 (A, B, C, E, F); AB → C, C → EF

R_2 (A, D, C, E, F); AD → GH, G → I, H → J

(c) 3NF

In relation R1, non-prime attributes E, F are transitively depends on key. So, relation is not in 3NF.

The relation R1 in 3NF can be written as,

R_{1a} (A, B, C); AB → C

R_{1a} (C, E, F); C → EF

In relation R2, non-prime attributes I, J are transitively depends on key. So, relation is not in 3NF.

The relation R1 in 3NF can be written as,

R_{2a} (A, D, G, H); AD → GH

R_{2b} (G, I); G → I

R_{2c} (H, J); H → J

(d) BCNF

Relation	FDs	Determinant	Key	BCNF?
R_{1a} (A, B, C)	AB → C	AB	AB	Yes
R_{1a} (C, E, F)	C → EF	C	C	Yes
R_{2a} (A, D, G, H)	AD → GH	AD	AD	Yes
R_{2b} (G, I)	G → I	G	G	Yes
R_{2c} (H, J)	H → J	H	H	Yes

So, relational schema in BCNF is as given below,

R_{1a} (A, B, C); AB → C

R_{1a} (C, E, F); C → EF

R_{2a} (A, D, G, H); AD → GH

R_{2b} (G, I); G → I

R_{2c} (H, J); H → J

Q. 17 We are given Relation R with Attributes A, B, C, D, E, F and the FDs as below, Find and explain which Armstrong's Axioms can be applied here to find closure, A → BC, B → E, CD → EF. (5 Marks)

Ans. :

1. $\{A\}^+$

(a) Axiom of Pseudo transitivity

A → BC and B → E ...(1)

∴ A → EC ...(2)

(c) Decomposition from Equations (1) and (2)

A → B, A → C, A → E

∴ $\{A\}^+$ = {A, B, C, E}

$\{A\}^+$ = { A, B, C, E}

2. {A, B} = {A, B, C, E}

3. {C, D} = {C, D, E, F}

4. {A, D} = {A, B, C, D, E, F}

Q. 18 Consider a dependency diagram of relation R and normalize it up to third normal form. (10 Marks)

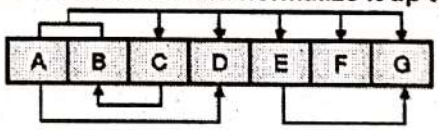


Fig. 9.3

Ans. :

Normalized Relation R (A, B, C, D, E, F, G) with set of FDs

AB → CDEFG

C → B

A → D

E → G

Q.19 Consider a dependency diagram of relation R and normalize it upto third normal form. (10 Marks)

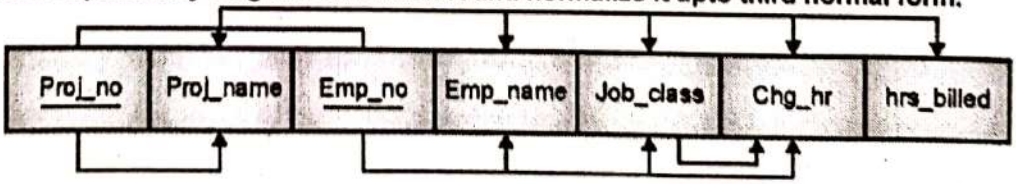


Fig. 9.4

Ans. :

– Normalized Relation,

- Employees (Proj_no, Emp_no, Proj_name, Emp_name, Job_Class, Chg_Hr, Hrs_Billed)
- With set of FDs

Proj_no, Emp_no → Proj_name, Emp_name, Job_Class, Chg_Hr, Hrs_Billed

Emp_no → Emp_name, Job_Class, Chg_Hr

Proj_no → Proj_name

Job_Class → Chg_Hr

Q. 20 Consider the following dependency diagram of relation R and normalize till 3NF form. (10 Marks)

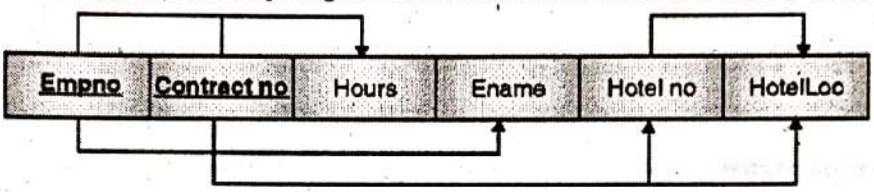


Fig. 9.5

Ans. :

Converting dependency diagram to dependencies,

- Empno,Contactno -> Hours
- Empno ->Ename
- Contactno ->Hotelno,HotelLoc
- Hotelno->HotelLoc

3NF schema is as given below,

- Emp(Empno,Contactno,Hours); Empno,Contactno -> Hours
- Emp_Details(Empno,Ename); Empno ->Ename
- Hotel_Contact(Contactno,Hotelno,HotelLoc); Contactno ->Hotelno,HotelLoc
- Hotel(Hotelno,HotelLoc); Hotelno ->HotelLoc

Chapter 10 : Transaction

Q. 1 What is transaction? Explain concept of transaction using example. (4 Marks)

Ans. :

Transaction

- Single SQL command is sent to database server as a query and server will reply with answer.
- Multiple SQL commands (DML,DRL etc.) are sent to database server which executed one after other (as shown in Fig. 10.1),

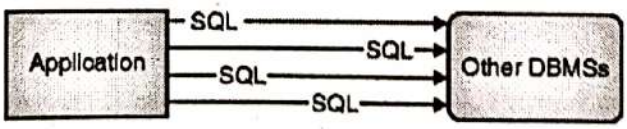


Fig. 10.1 : Executing single operation in DBMS

- In place of sending one by one SQL command to server combine multiple operations that are logically similar and send to serve as a single logical unit called **transaction**.

Example : Transferring Rs.100 from one account to other

1. Withdraw Rs.100 from account_1
2. Deposit Rs.100 to account_2

Fig. 10.2 : Executing transaction in DBMS

Q. 2 What are ACID properties ? **(5 Marks)**

Ans. :

1. Atomicity

- Transaction must be treated as a single unit of operation.
- That means when a sequence of operations is performed in a single transaction, they are treated as a single large operation.

Example : Making an airline reservation

(a) Check availability of seats in desired flight.

(b) Airline confirms your reservation

(c) Reduces number of available seats

(d) Charges your credit card (deduct amount from your balance)

(e) Increases number of meals loaded on flight (Sometimes)

2. Consistency

- Consistent state is a database state in which all valid data will be written to the database.
- If a transaction violates some consistency rules, the whole transaction will be rolled back and the database will be restored to its previous consistent state with those rules.
- Consistency means transaction will never leave your database in a half finished (inconsistent) state.
- If one part of the transaction fails, all of the pending changes made by that transaction are rolled back.

Example : Money transfer in above example

- Initially in total balance in accounts A is 1000 and B is 5000 so sum of balance in both accounts is 6000 and while carrying out above transaction some type of failure occurs after Write (A) but before Write (B) then system may lose 100 rupees in calculation.
- Sum of balance in both accounts is 5900 (which should be 6000) which is not a consistent result which introduces inconsistency in database.
- This means that during a transaction the database may not be consistent.

3. Isolation

- Isolation property ensures that each transaction must remain unaware of other concurrently executing transactions.
- Transaction isolation is generally configurable in a variety of modes. For example, in one mode, a transaction locks until the other transaction finishes.
- Even though many transactions may execute concurrently in the system. System must **Example : Money transfer in above example.**
- The database is temporarily inconsistent while above transaction is executing, with the deducted total written to A and the increased total is not written to account B. If some other concurrently running transaction reads balance of account A and B at this intermediate point and computes A+B, it will observe an inconsistent value (Rs. 5900). If that other transaction wants to perform updates on accounts A and B based on the inconsistent values (Rs. 5900) that it read, the database may be left database in an inconsistent state even after both transactions have completed.

4. Durability

- Once the execution of the transaction completes successfully, and the user will be notified that the transfer of amount has taken place, if there is no system failure in this transfer of funds.
- The durability property guarantees that, all the changes made by transaction on the database will be available permanently, although there is any type of failure after the transaction completes execution.

Q. 3 Explain transaction state diagram. (5 Marks)

Ans. : Transaction states

- A transaction must be in one of the following states :

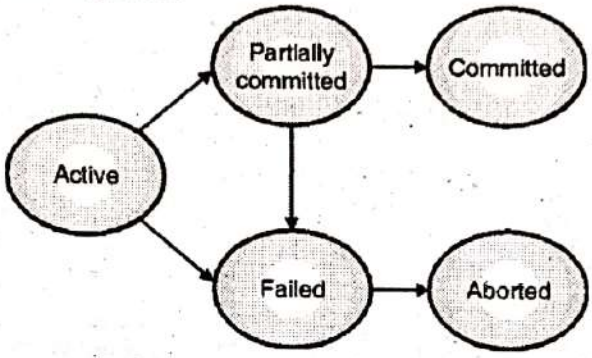


Fig. 10.4 : State diagram of a transaction

(a) Active

- This is initial state of transaction execution.
- As soon as transaction execution starts it is in active state.
- Transaction remains in this state till transaction finishes.

(b) Partially committed

- As soon as last operation in transaction is executed transaction goes to partially committed state.
- At this condition the transaction has completed its execution and ready to commit on database server. But, it is still possible that it may be aborted, as the actual output may be there in main memory, and thus a hardware failure may prohibit its successful completion.

(c) Failed

A transaction enters the failed state after the system determines that the transaction can no longer proceed with its normal execution.

Example : In case of Hardware or logical errors occurs while execution.

(d) Aborted

- Failed transaction must be rolled back. Then, it enters the aborted state.
- Transaction has been rolled back restoring into prior state.

(e) Committed

- When last data item is written out, the transaction enters into the committed state.
- This state occurs after successful completion of transaction.
- A transaction is said to have terminated if has either committed or aborted.

Q. 4 What are transaction schedules ? (5 Marks)

Ans. : Transactions Schedules

- Schedule is a sequence of instructions that specify the sequential order in which instructions of transactions are executed.

- A schedule for a set of transactions must consist of all instructions present in that transactions, it must save the order in which the instructions appear in each individual transaction.

Representation

- We denote this transaction T as,
- $R_T(X)$: Denotes read operation performed by the transaction T on object X.
- $W_T(X)$: Denotes write operation performed by the transaction T on object X.
- Each transaction must specify its final action as commit or abort.

Q. 5 Explain serial transaction with example. (5 Marks)

Ans. :

Serial Transaction

This is simple model in which transactions executed in a serial order that means after finishing first transaction second transaction starts its execution.

Example :

- Consider below two transactions T_1 and T_2.
- **Transaction T_1 :** deposits Rs.100 to both accounts A and B.
- **Transaction T_2 :** doubles the balance of accounts A and B.

T_1 :

Read(A)
A ← A + 100
Write(A)
Read(B)
B ← B + 100
Write(B)

T_2 :

Read(A)
A ← A*2
Write(A)
Read(B)
B ← B*2
Write(B)

- Serial schedule for above transaction can be represented as below,

Schedule A : A consistent serial schedule

- A consistent serial schedule is obtained by executing T_1 right after T_2.

T_1	T_2	A	B	Operations
		25	25	Initial Balance
	Read(A); A ← A*2			
	Write(A)	50	25	
	Read(B); B ← B*2			
	Write(B)	50	50	
Read(A); A ← A + 100				
Write(A)		150	50	
Read(B); B ← B + 100				
Write(B)		150	150	
		150	150	Final Balance

- In above Serial schedule for first execute transaction T_2 then T_1 which may results in some final values.

Representation : $T_1 \rightarrow T_2$

Schedule B : A consistent serial schedule

A serial schedule that is also consistent is obtained by executing T_2 right after T_1.

T_1	T_2	A	B	Operations
		25	25	Initial Balance
Read(A); A ← A + 100				
Write(A)		125	25	
Read(B); B ← B + 100				

T_1	T_2	A	B	Operations
Write(B)		25	125	
	Read(A); A ← A*2			
	Write(A)	250	125	
	Read(B); B ← B*2			
	Write(B)	250	250	
		250	250	Final Balance

Representation : $T_2 \rightarrow T_1$

- In above schedule T_1 is executed entirely then T_2 has started. Assume account A with Rs.25 and B with Rs.25. Then transaction T_1 will update A as 125 and B as 125. Now T_2 will read updated values of A and B. T_2 will update value of A as 250 and B as 250. The consistency constraint is A + B should remain unchanged. So at end of T_2, A + B.
- i.e. 250 + 250 = 500 remains unchanged so execution of this schedule keeps database in consistent state.

Q.6 Explain concurrent execution and give example. (5 Marks)

Ans.: Concurrent Execution

- Transactions executed concurrently, that means operating system executes one transaction for some time then context switches to second transaction and so on.
- Transaction processing can allows multiple transactions to be executed simultaneously on database server.

Example :

Consider below two transactions T_1 and T_2,

Transaction T_1 : deposits Rs.100 to both accounts A and B.

T_1 :

Read(A)
A ← A + 100
Write(A)
Read(B)
B ← B + 100
Write(B)

Transaction T_2 : doubles the balance of accounts A and B.

T_2 :

Read(A)
A ← A*2
Write(A)
Read(B)
B ← B*2
Write(B)

Above transaction can be executed concurrently as below,

Schedule C : A schedule that is not serial but is still consistent

Obtained by interleaving the actions of T_1 with those of T_2

Table 10.1

T_1	T_2	A	B	Operations
		25	25	Initial Balance
Read(A); A ← A + 100;				
Write(A);		125		
	Read(A); A ← A*2;			
	Write(A);	250		
Read(B); B ← B + 100;				
Write(B);			125	
	Read(B); B ← B*2;			
	Write(B);			
		250	250	Final Balance

- In above given schedule Part of T_1 is executed which updates A to 125. Then processor switches to T_2 and part of T_2 which updates A to 250 is executed.
- Then context switch to T_1 and remaining part of T_1 which updates B to 250 is executed. At the end remaining part of T_2 which reads B as 125 and updates it to 250 by multiplying value of B by two. This concurrent schedule also maintains consistency of database as ultimately A + B is 250 + 250 = 500 (unchanged).

Q. 7 Describe conflict serializability. **(5 Marks)**

Ans.: Conflict Serializability

The database system must control concurrent execution of transactions which ensure that the database state remains in consistent state.

Example :

A schedule is conflict serializable if it is conflict equivalent to a serial schedule.

T_1	T_2
Read(P)	
Write(P)	
	Read(P)
Read(Q)	
	Write(P)
Write(Q)	
	Read(Q)
	Write(Q)

Fig. 10.5 : Schedule S

Instruction WRITE(P) of T_1 and READ(P) of T_2 cannot be swapped as they conflict. (as shown in Fig. 10.6)

	T_1	T_2
	Read(P)	
Can not Be Swap	Write(P)	
		Read(P)
	Read(Q)	
		Write(P)
	Write(Q)	
		Read(Q)
		Write(Q)

Fig. 10.6

Instruction READ(Q) of T_1 and READ(P) of T_2 can be swapped as they operate on different data items so do not conflict. (as shown in Fig. 10.7)

Can be swapped

T_1	T_2
Read(P)	
Write(P)	
	Read(P)
Read(Q)	
	Write(P)
Write(Q)	
	Read(Q)
	Write(Q)
Before Swap	

T_1	T_2
Read(P)	
Write(P)	
Read(Q)	
	Read(P)
	Write(P)
Write(Q)	
	Read(Q)
	Write(Q)
After Swap	

Fig. 10.7

Instruction WRITE(Q) of T_1 and WRITE(P) of T_2 can be swapped as they operate on different data items so do not conflict.

Can be swapped

T_1	T_2
Read(P)	
Write(P)	
Read(Q)	
	Read(P)
	Write(P)
Write(Q)	
	Read(Q)
	Write(Q)

T_1	T_2
Read(P)	
Write(P)	
Read(Q)	
	Read(P)
Write(Q)	
	Write(P)
	Read(Q)
	Write(Q)

Fig. 10.8

Instruction WRITE (Q) of T_1 and READ (P) of T_2 can be swapped as they operate on different data items so do not conflict.

can be Swap As no conflict

T_1	T_2
Read(P)	
Write(P)	
Read(Q)	
	Read(P)
Write(Q)	
	Write(P)
	Read(Q)
	Write(Q)

T_1	T_2
Read(P)	
Write(P)	
Read(Q)	
Write(Q)	
	Read(P)
	Write(P)
	Read(Q)
	Write(Q)

Fig. 10.9 : Schedule R

If a schedule S can be transformed into a schedule R by a series of swap operations on non conflicting instructions, then **schedule S and R are Conflict equivalent.**

Q. 8 Explain view serializability, describe using example. **(5 Marks)**

Ans. :

View Serializability

View equivalence is less strict than conflict equivalence, but it is like conflict equivalence based on only the read and write operations of transactions.

Conditions for view equivalence

Let, D = Data item

S_1, S_2 = Transaction schedules

T_i, T_j = Database transaction

- Schedules S_1 and S_2 are view equivalent if they satisfy following conditions for each data item D,
  - (a) S_1 and S_2 must have same transactions included and also they are performing same operations on same data. If T_i reads initial value of D in S_1, then T_i also reads initial value of D in S_2.
  - (b) If T_i reads value of D written by T_j in S_1, then T_i also reads value of D written by T_j in S_2.
  - (c) If T_i writes final value of D in S_1, then T_i also writes final value of D in S_2.
- First 2 conditions ensure that transaction reads same value in both schedules.
- Condition 3 ensures that final consistent state.
- If a concurrent schedule is view equivalent to a serial schedule of same transactions then it is **View serializable.**

Example :

Schedules S and R are view equivalent,

Schedule S		
T_1	T_2	T_3
Read(P)		
	Write (P)	
Write (P)		
		Write (P)
Schedule R		
T_1	T_2	T_3
Read(P)		
Write (P)		
	Write (P)	
		Write (P)

Fig. 10.10

Q. 9 What is term precedence graph ? **(5 Marks)**

Ans. : Precedence Graph

- A particular transaction schedule can be serialized; draw a precedence graph.
- Precedence (or serializability) graph for schedule S is a graphical representation of transactions executed.
- A precedence graph is also known as conflict graph or serializability graph.
- Precedence graph is a graph of nodes and vertices, where the nodes are the transaction names and the vertices are attribute collisions.

Algorithm for precedence graph

(a) Add a node for each transaction.

(b) Add a directed edge from T_i to T_j , if T_j reads the value of an item written by T_i.

(c) Add a directed edge from T_j to T_i, if T_j writes a value in to an item after it has been read by T_i.

Example :

Consider Schedule C

T_1	T_2
Read(A); A ← A + 100;	
Write(A);	
	Read(A); A ← A*2;
	Write(A);
Read(B); B ← B + 100;	
Write(B);	
	Read(B); B ← B*2;
	Write(B);

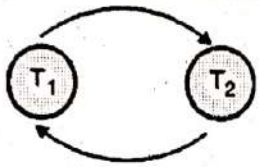


Fig.10.11

Q. 10 **A schedule has transactions T_1 and T_2 as given below;**

$r_1(x)$, $r_2(z)$, $r_1(z)$, $r_3(x)$, $r_3(y)$, $w_1(x)$, $w_3(y)$, $r_2(y)$, $w_2(z)$, $w_2(y)$

1. **Draw precedence graph.**
2. **Is schedule conflict serialisable or not ? Find respective serial schedule.**
3. **Is above Schedule view serialisable or not ?**

(5 Marks)

Ans. :

(1) Step 1 :

$r_1(x)$, $r_2(z)$, $r_1(z)$, $r_3(x)$, $r_3(y)$, $w_1(x)$, $w_3(y)$, $r_2(y)$, $w_2(z)$, $w_2(y)$,

Schedule S : Given Schedule

T_1	T_2	T_3
r(x)		
	r(z)	
r(z)		
		r(x)
		r(y)
w(x)		
		w(y)
	r(y)	
	w(z)	
	w(y)	

(2) **Step 2 :** Precedence graph

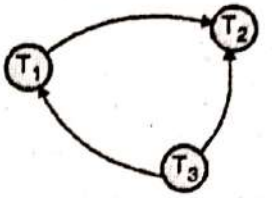


Fig. 10.12

∴ Above graph show no cycle

∴ Schedule s_1, is conflict serializable.

(3) **Step 3 :** Corresponding serial schedule

$T_3 \rightarrow T_1 \rightarrow T_2$

Schedule S_1 : equivalent serial schedule

T_1	T_2	T_3
		r(x)
		r(y)
		w(y)
r(x)		
r(z)		
w(x)		
	r(z)	
	r(y)	
	w(z)	
	w(y)	

∴ **Serial Schedule S_1 :** $r_3(x)$, $r_3(y)$, $w_3(y)$, $r_1(x)$, $r_1(z)$, $w_1(x)$,$r_2(z)$, $r_2(y)$, $w_2(z)$, $w_2(y)$,

(4) **Step 4 :** Above schedule is view serializable or not.

Conditions for view serializability

(1) S and S_1 must have same number of transaction and same number of read write operation.

(2) Initial read operation

X is read initially by T_3 in S as well as S_1

Y is read initially by T_3 in S and S_1 both.

Z is read initially by T_1 in both S and S_1

(3) If S reads value of X or Y written by T; then S_1 also reads value of x or y written by same transaction.

S_1 reads value Y written by T_3.

S also reading value of Y which is written by T_3.

(4) Final write operation

X-final write (x) done by T_1 in S as well as S_1

Y-final write (y) done by T_2 in S and S_2 both.

∴ S is view serializable.

Or S is view equivalent of S_1.

Chapter 11 : Concurrency Control

Q.1 Explain the concept of concurrency control. (5 Marks)

Ans. :

Concurrency Control

- The DBMS uses concurrency control to manage multi-user databases.
- Concurrency control is concerned with preventing loss of data integrity due to interference between users in a multi-user environment.

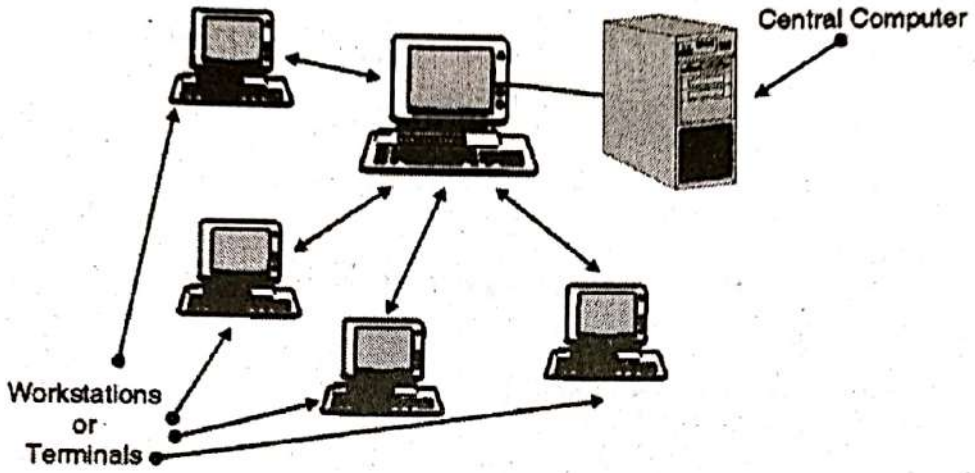


Fig. 11.1: Concurrent database access

- Concurrency control should provide a mechanism for avoiding and managing conflict between various database users operating same data item at same time.

Q. 2 Explain Conflicting Transactions with example. (4 Marks)

Ans. :

Conflicting Transactions

- A pair of consecutive database actions (reads, writes) is in conflict if changing they are accessing and changing same data simultaneously.
- The main conditions for conflict is transactions reading and/or writing the same data items.

Table 11.1

		Transaction T_i			
Transaction T_j		Read		Write	
		Accesses Same Data	Accesses Different Data	Accesses Same Data	Accesses Different Data
	Read	No Conflict	No Conflict	Conflict	No Conflict
	Write	Conflict	No Conflict	Conflict	No Conflict

- In Table 11.1 if transaction T_i and T_j accesses same data item for read(T_j) and write(T_j) then there is said to be **conflict** but if they are accessing different data items for read(T_j) and write(T_j) then there is said to be **no conflict.**

Q. 3 Explain various problems of concurrent executions. **(5 Marks)**

Ans.:

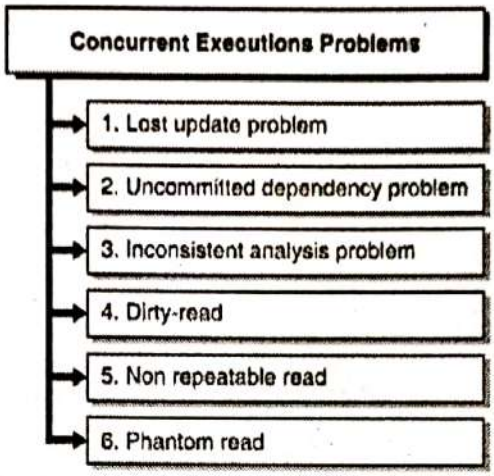


Fig. 11.2 : Concurrency execution problems

(1) Lost update problem

- Update made by one transaction is overwritten by other transaction or other user.
- This may loss updates of one transaction.

Example 1

Time	X	Y	Balance
t_1	-	BEGIN TRANSACTION	500
t2	BEGIN transaction	READ (A)	500
t3	READ (A)	A = A + 100	500
t4	A = A – 10	WRITE (A)	600
t5	WRITE (A)	COMMIT	490
t6	COMMIT	-	490

T_1 & T_2 - Transaction Y has read the value of Aas 500.

T_3 - Transaction Xhas read the value of Aas 500.

T_4 - TransactionY writes the new value of A as 200

- Transaction X has subtracted 10 from its value of Ato produce 490.

T_5 - Transaction X updates the value of Aon disc.

T_6 - The result of this operation T_4performed by transaction Y is lost.

Transaction X has overwritten the result of transaction Y.

The problem is avoided by not allowing only one transaction to update data at a time.

(2) Uncommitted dependency problem

- This problem occurs when user sees data coming from intermediate step of another ongoing transaction which is yet uncommitted.

Time	X	Y	A
t_1	–	begin transaction	100
t_2		read (A)	100
t_3		A=A + 100	100
t_4	begin transaction	Write (A)	200
t_5	read (A)		200
t_6	A=A – 10	Rollback	100
t_7	write (A)		190
t_8	commit		190

Example 2 : Transaction Y reads and updated the value of A (100 + 100 = 200) and writes the result at time t_4.

- Transaction X should be updating A = 100 because transaction Y has been rolled back and its changes undone.
- Transaction X has used the result of transaction Y but this result was incorrect as transaction Y failed.

This problem is avoided by not allowing transaction X to read the value of A until transaction Y either commits or rolls back.

(3) Inconsistent analysis problem

Transaction reads partial results of incomplete transaction update made by other transaction.

T_1	T_2
–	BEGIN Transaction
BEGIN Transaction	SUM=0
R (X)	R (X)
X = X – 20	SUM = SUM + X
W (X)	R (Y)
R (Z)	SUM = SUM + Y
Z = Z + 10	
W (Z)	
COMMIT	R (Z)
	SUM = SUM + Z
	COMMIT

Example 3 Transaction T_2 is adding the values of X, Y and Z.

- However, at the same time, transaction T_1 is transferring Rs.100 between X and Z.
- As transaction T_2 has used the old balances of X and Zits final result is incorrect.
- This problem can be solved by preventing transaction T_1 from transferring the amount, Before transaction T_2 has committed.

(4) Dirty-read

- Dirty data : Data, updated by a transaction, but not yet committed hence all users reading old data which is called as dirty data.
- Dirty read : A transaction reading dirty data is called as 'dirty read'.
- Because there is always a chance that the first transaction might rollback the change which causes the second transaction reads an invalid value.
- In short, dirty read is changes made during a Transaction are 'visible' to other parties.

(5) Non repeatable read

- A non-repeatable read occurs when a persistent object is read twice within a same transaction
- It is possible that between the reads, data is modified by another transaction, therefore the second read returns different values as compared to the first;

(6) Phantom read

- Phantom reads means insert or delete action is performed on a table row which referred by another transaction.
- Transaction T_2 inserts a row, T_1 rereads the query and if T_1 see the additional row, it is a ghost row to T_1 then this is called as **Phantom read**.

Q. 4 Explain various concurrency control schemes. (2 Marks)

Ans. :

Concurrency Control Schemes

- Fundamental property of transaction is isolation.
- Isolation property ensures that each transaction must remain unaware of other concurrently executing transactions.
- In case of concurrent transactions, when several transactions are executing simultaneously on a database may not preserve isolation property for long time.
- To implement concurrent system there must be interaction among various concurrent transactions.
- This can be done by using one of the concurrency control schemes.
- All the following schemes are based on serializability of schedules.

Q. 5 Explain concept of locking. (5 Marks)

Ans. :

Introduction (concept of locking) - Implementation of Isolation

- Whenever one transaction is accessing data, second transaction should not change data otherwise there may be dirty read problem. This can be done with help of locking concept.
- Transaction can access data if it is locked by that transaction.
- Locking is necessary in a concurrent environment to assure that one process should not retrieve or update a record which another process is updating.

Types of locks

(a) Shared locks

- This type of locking is used by the DBMS when a transaction wants to only **read** data without performing modification to it from the database.
- Shared locks are represented by S.
- If a transaction T_1 has obtained a shared lock on data item X, then transaction T_1 can only read data item X, but cannot write on data item X.

SQL Implementation :

```
LOCK TABLE customer IN
SHARED MODE;
```

(b) Exclusive locks

- This type of locking is used by the DBMS when a transaction wants to **read or write** (i.e. performing update) data in the database.
- When a transaction has an exclusive lock on some data, other transactions cannot acquire any type of lock (shared or exclusive) on the data.
- Exclusive locks are represented by X.

- SQL Implementation :

```
LOCK TABLE customer IN
EXCLUSIVE MODE;
```

(c) **Lock compatibility matrix**

	Lock by Transaction T_j		
Lock by Transaction T_i		S	X
	S	No Conflict	Conflict
	X	Conflict	Conflict

- Transaction can acquire shared lock although there is other transaction which currently has a shared or an exclusive lock on the data. As many transactions can READ data without any conflict.
- Transaction can acquire an exclusive lock only if no other transaction currently has a shared or an exclusive lock on the data.

Q. 6 How does locking protocol works ? **(5 Marks)**

Ans. : Working of Locking Protocol

- Whenever any user transaction want to access any data item (D) then he will issue a locking request to concurrency control manager.
- A lock manager is a process that receives **locking request messages** and sends response accordingly.
- The response given may be
  - Lock grant - On granting to request.
  - If there is deadlock then rollback transaction.
  - If lock is already held by other transaction then asking to wait.
  - On unlock just an acknowledgement is sent.
- Transaction can access data if it is locked by that transaction.

Example : Consider two transactions T_1 and T_2,

T_1 :	LOCK – X(Q)
	READ (Q)
	Q = Q – 100
	WRITE (Q)
	UNLOCK (Q)
	LOCK-X(P)
	READ (P)
	P = P + 100
	WRITE(P)
	UNLOCK(P)
T_2 :	LOCK-S(P)
	READ(P)
	UNLOCK(P)
	LOCK – S(Q)
	READ (Q)
	UNLOCK (Q)
	DISPLAY(P + Q)

Following is the schedule for T_1 and T_2, which is not scheduled properly because of granting locks at improper timings.

Table 11.2 : Schedule A

T_1	T_2	Concurrency Control Manager
LOCK – X(Q)		
		Grant - X(Q, T_1)
READ (Q)		
Q = Q – 100		
WRITE (Q)		
UNLOCK (Q)		
	LOCK-S(P)	
		Grant - S(P,T_2)
	READ(P)	
	UNLOCK(P)	
	LOCK – S(Q)	
		Grant - S(Q,T_2)
	READ (Q)	
	UNLOCK (Q)	
	DISPLAY(P + Q)	
LOCK-X(P)		
		Grant - X(P,T_1)
READ(P)		
P = P + 100		
WRITE(P)		
UNLOCK(P)		

- In above schedule, T_1 unlocks Q very early, because of which the inconsistent state is exposed to transaction T_2 which results in wrong value of P + Q.
- Assume P = 400, Q = 600 then serial schedule < T_1, T_2 > will show 1000 but above given schedule will result in 900 as T_1 has not updated P and before that only P + Q displayed by T_2.
- Both the transactions are holding a lock on two different data items, and waiting to acquire lock on each other's data item, this will never end up **waiting** of them and schedule cannot be executed further. This situation can be solved by roll back of one of the transactions. But this may lead to rollback of another transaction (such situation is called as cascaded rollback occurs in dependent transactions).

Q. 7 What is granting locks ? **(2 Marks)**

Ans. :

Granting Locks

- Lock is granted only when no other conflicting type of lock on it, is held by other transactions.
- Lock compatibility matrix shows which locks will be granted and which will be rejected.
- It may happen that a series of transactions holding shared lock on data item X one by one so transaction T has to wait for exclusive lock on data item X forever, this situation is called as starvation of T.
- To avoid starvation, Concurrency Control Manager should consider two things,

  (a) No other transaction is holding conflicting lock on it.

  (b) No other transaction is waiting on that data item for locking it.

Q. 8 What is rejecting locks ? **(2 Marks)**

Ans. :

Rejecting Locks

- Lock is rejected when other conflicting type of lock is held by other transactions.
- If any transaction has exclusive lock for writing on data item D then no other transaction can acquire any type of lock on data item D.

Lock compatibility matrix will show the locks that will be granted or rejected.

Q. 9 Explain advance locking concept using 2P locking **(5 Marks)**

Ans. : Two-Phase Locking (2PL)

- Two-Phase Locking (2PL) synchronizes reads and writes by explicitly detecting and preventing conflicts between concurrent operations.
- Before reading data item X, a transaction must "own" a read lock on X. Before writing into X, transaction must "own" a write lock on X.

The ownership of locks is governed by two rules :

(a) Different transactions cannot simultaneously own conflicting locks (i.e. WR).

(b) Once a transaction surrenders ownership of a lock, it may never obtain additional locks.

- For **'RW'** synchronization two locks conflict if

Both are locks on the same data item.

One is a read lock and the other is a write lock.

- For **'WW'** synchronization two locks conflict if.

Both are locks on the same data item.

Both are write locks.

The second lock ownership rule causes every transaction to obtain locks in a two phase manner.

Growing phase

In this phase, transaction may obtain n number of new locks but may not release any lock.

Shrinking phase

Transaction may release locks but cannot obtain any new Lock.

Example :

(A) No 2P (B) with 2P locking.

Q. 10 Explain different variant of 2PL. **(5 Marks)**

Ans.:

(1) Strict two-phase locking protocol

- Avoids cascaded rollbacks.
- It requires not only two-phase locking but also that all exclusive-locks held by transaction should be held until that transaction commits or abort.
- This property ensures that if data is being modified by one transaction (holding lock-X) then other transaction can't read it until first transaction commits.
- Strict schedule recoverability.
- Not deadlock free.

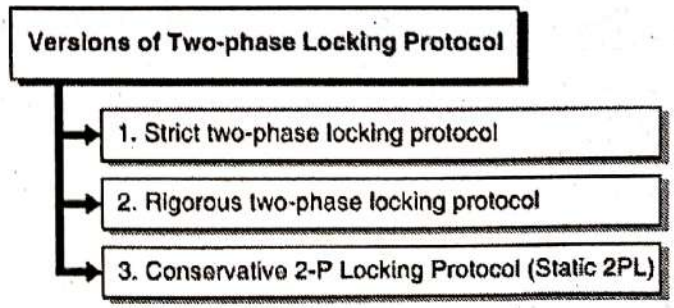


Fig. 11.4 : Versions of Two-phase Locking Protocol

(2) Rigorous two-phase locking protocol

- It also avoids cascading rollbacks.
- It requires that all share and exclusive locks to be held until the transaction commits.
- So transactions can be serialized in the sequence they commit.
- Most of the database systems implement strict or rigorous two phase locking protocol.

(3) Conservative 2-P Locking Protocol (Static 2PL)

- It is also called as static 2P locking protocol.
- This scheme requires locking all items needed to access before the transactions starts.
- It begins execution by declaration about read set and write Set of all data items needed in advance.

Read set

Write set

Q. 11 What do you mean by timestamp based protocol? Explain timestamp modeling protocol. (10 Marks)

Ans. : Concept of Timestamp

- A fixed timestamp is assigned at start of execution of the transaction. Every transaction T_j has been assigned a timestamp by database system denoted as TS (T_j).
- A transaction which has entered in system recently will have greater timestamp. If transaction T_j starts after T_i then,

$$TS(T_i) < TS(T_j)$$

Every data item X is with two timestamp values :

(a) W-timestamp (X)

- It denotes the largest timestamp of any transaction that executed WRITE (X) successfully on given data item X.
- That mean it is timestamp of recent WRITE (X) operation.
- On execution of next WRITE (X) operation (O_1), W-timestamps will be updated to new timestamp of O_1 i.e. TS (O_1).

(b) R-timestamp (X)

- Denotes the largest timestamp of any transaction that executed READ (X) successfully on data item (X).
- That mean it is timestamp of recent READ (X) operation.
- On execution of next READ (X) operation (O_1), R-timestamps will be updated to new timestamp of O_1 i.e. TS (O_1).

Timestamp - ordering Protocol

- This protocol ensures any conflicting READ or WRITE is executed in order of timestamp.
- If by any reasons, if transaction is aborted then on restarting, new timestamp is assigned.

Working

(a) For transaction T_i to execute READ (X) Operation,

If TS (T_i) < W - timestamp (X)

Then T_i is trying to read value of X that is overwritten by other transaction.

W - timestamp (X) = 149 (Recent Write)			
TS	**T_i**	**T_x**	**Operations**
148	READ (X)	...	TS (T_i) < W - timestamp (X) i.e. 148 < 149 (W-Timestamp)
149	Unlock (X)	WRITE (X)	Record W - timestamp (X) = 149

So this READ is rejected (as T_x is already performing write operation on data so no other operation can be performed by any other transaction) and T_i is rolled back.

If TS (T_i) ≥ W - timestamp (X)

Then READ executed and set

R-timestamp(X) = max {TS (T_i), R-timestamp}			
TS	**T_i**	**T_x**	**Operations**
148	....	WRITE (X)	Record W - timestamp (X) = 148 (recent write)
149	READ (X)	....	TS (T_i) ≥ W - timestamp (X) i.e. 149 ≥ 148 (W-Timestamp) Record R-timestamp(X) = 149
150	....	....	....
151	READ (X)	.....	TS (T_1) ≥ W - timestamp (X) i.e. 151 ≥ 148 (W-Timestamp) Record R-timestamp(X) = max{149,151} = 151

(b) If transaction T_i execute WRITE (X) Operation,

If TS (T_i) < R - timestamp (X)

Then T_i has produced value of X which is not needed now so rollback T_i.

TS	T_i	T_x	Operations
148	....	...	...
149	WRITE(X)	....	TS (T_1) < R - timestamp (X) i.e. 149 < 151 (R-Timestamp) Reject WRITE roll back T_i
150	....	....	....
151		READ (X).	Record R-timestamp(X) = 151 (Recent READ Operation)

If TS (T_i) < W - timestamp (X)

Then T_i is trying to write obsolete value of X, So rollback T_i

TS	T_i	T_x	Operations
148	....	...	...
149	WRITE (X)	....	TS (T_1) < W - timestamp (X) i.e. 149 < 151 (W-Timestamp) Cannot write as other transaction using data X for writing.
150	....	....	....
151		WRITE (X)	Record W-timestamp(X) = 151

(c) Otherwise, system executes WRITE (X) and sets

W-timestamp (X) = TS (T_i)

TS	T_i	T_x	Operations
150	....	....	....
151	WRITE (X)		Record W-timestamp(X) = 151

Q. 12 Explain Thomas write rule with algorithm. (5 Marks)

Ans. : Thomas' Write Rule

- Thomas Write Rule is also known as Modified timestamp protocol.
- Thomas Write Rule uses view serializability.
- It generates schedules which are not possible by other protocols.
- Generated schedules that are view equivalent to the serial schedule.

Example :

T_1	T_2
READ (D)	
	WRITE (D)
WRITE (D)	

- Rejects few write (D) operations by modifying check for WRITE(D).
- Suppose, T_i issues WRITE (D)

(a) If TS (T_i) < R - timestamp (D)

Then T_i is producing value of D, which needed previously not now Assuming it is never produced the system rejects WRITE and T_i is rollback.

TS	T_i	T_x	Operations
148	....	...	...
149	WRITE(X)	....	TS (T_1) < R - timestamp (X) i.e. 149 < 151 (R-timestamp) Reject WRITE roll back T_i
150	....	....	....
151		READ (X)	Record R-timestamp(X) = 151 (Recent READ Operation)

(b) If TS (T_i) < W-timestamp (D)

Then T_i is writing obsolete or outdated value of D so WRITE is ignored.

TS	T_i	T_x	Operations
148	....	...	...
149	WRITE (X)	....	TS (T_1) < W - timestamp (X) i.e. 149 < 151 (W-timestamp) Cannot write as other transaction using data X for writing.
150	....	....	....
151		WRITE (X)	Record W-timestamp(X) = 151

(c) Otherwise

WRITE (D) is executed and W-timestamp (D) = TS (T_j)

TS	T_i	T_x	Operations
150	....	....	....
151	WRITE (X)	....	Record W-timestamp(X) = 151

Compare to Basic Timestamp ordering

Unlike, timestamp protocol, Thomas write rule asks to ignore write operation if T_i issues WRITE (D) and TS (T_i) < W - timestamp (D).

Chapter 12 : Recovery System

Q. 1 Explain method of database recovery. **(5 Marks)**

Ans. :

Database Recovery

- Database recovery is the process of restoring the database to original (correct) state as it was before database failure occurs.
- The process of solving any type of database failures, quickly and without data loss and keep database available is called database recovery.
- The main element of database recovery is the most recent database backup. If you maintains database backup efficiently, then database recovery is very straight forward process.

Example :

To recover data from system having full backup option is as below,

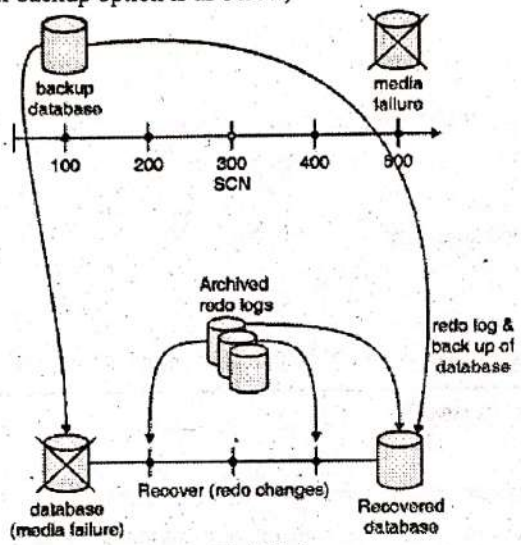


Fig. 12.1

Restoring entire system to a certain point may require time, depends on when last full backup taken and incremental backups which covers the period of time between the full backup and restore point.

Q. 2 Explain various types of system failure. **(5 Marks)**

Ans. :

Failure Classification

- Like any other real world database systems also suffer from failure from a variety of causes: disk crash, power outage, software error, a fire in the machine room even rain. The result of any failure is loss of information.
- Therefore, the database system must take actions in advance to ensure that the atomicity and durability properties of transactions are preserved.

1. Hardware Failure / System crash

- There is a hardware malfunction that causes the loss of the content of volatile storage, and brings transaction processing to a halt.
- The content of non volatile storage remains intact, and is not corrupted or changed.

2. Software Failure

The database software or the operating system may be corrupted or failed to work correctly, that may causes the loss of the content of volatile storage, and brings about database failure.

3. Media failure

- A disk block loses its content as a result of either a head crash or failure during a data-transfer operation.
- Copies of the data on other disks such as tapes, CDs are used to recover from the failure.

4. Network Failure

- The problem with network interface card can cause network failure.
- There may be problem with network connection.

5. Transaction failure

There are two types of errors that may cause a transaction to fail :

(a) Logical error

(b) System error

6. Application software error

- The problem with software accessing the data from database.
- This may cause database failure as data cannot be updated using such application to it..

7. Physical disasters

The problem caused due to flood, fire, earthquake etc..

8. Application software error

These are some logical errors in the program that is accessing database, which cause one or more transactions failure.

Q. 3 What is checkpoint ? What is its use ? **(5 Marks)**

Ans. : Checkpoint

- A database checkpoint is where all committed transactions are written to the redo/audit logs.
- The database administrator determines the frequency of the checkpoints based on volume of transactions.
- When a system fails, It check log to determine recovery action.
- Too frequent checkpoints can affect the performance.

Need of check points

- To record status of transaction execution, the system maintains the log, using one of the two techniques.
- The system periodically performs checkpoints, with following sequence of actions :
  1. Output all log records onto stable storage which are currently stored in main memory.
  2. Output to the disk.
  3. Output onto stable storage a log record <checkpoint>.
- Transactions are not allowed to perform any update actions, such as writing to a buffer block or writing a log record, while a checkpoint is in working state.
- The presence of a <checkpoint> record in the log allows the system to restructure its recovery procedure.

Q. 4 Explain concept of shadow paging. **(5 Marks)**

Ans. :

Shadow Paging

- It is not always convenient to maintain logs of all transactions for the purposes of recovery.
- An alternative is to use a system of shadow paging.
- This is where the database is divided into pages that may be stored in any order on the disk.
- In order to identify the location of any given page, we use something called a **page table.**

Method

- During the life of a transaction two page tables are maintained as below,
  (i) Shadow page table
  (ii) Current page table.
- When a transaction begins both of these page tables point to the same locations (are identical).

  During the lifetime of a transaction the shadow page table doesn't change at all.
- However during the lifetime of a transaction update values etc. may be changed.
- For pages updated by the transaction, two versions are kept. The old version is referenced by the shadow directory and the new version by the current directory.
- So whenever we update a page in the database we always write the updated page to a new location.
- This means that when we update our current page table it reflect the changes that have been made by that transaction.

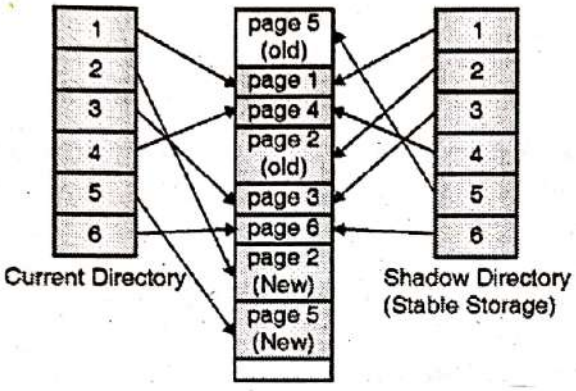


Fig. 12.3 : Pagetable-shadow paging

- The shadow page table shows the state of the database just prior to a transaction, and the current page table shows the state of the database during or after a transaction has been completed.

Q. 5 Explain ARIES algorithm with different steps. Enlist all advantages of ARIES algorithm. (5 Marks)

Ans. :

ARIES - Algorithm

- ARIES is a recovery algorithm that is designed for no force type of backup approach.
- Recovery manager is generally called when there is a crash.
- Restart can be proceeded in three different phases as below

Principles of ARIES algorithm

(a) Write-ahead logging

- Any change to a database object is first recorded in some log file.
- The record in the log must be written to any of stable storage before the change in database is written to disk.

(b) Repeating history during Redo

- ARIES finds all operations done by DBMS before the crash and restores system back to the same state that it was in at the time of the crash.
- Then, it aborts all the actions of transactions those are still there in active state at the time of the crash.

(c) Logging changes during Undo

- We make changes to the database during restoring database all transactions are logged in same order.
- So, it ensures same action is not repeated in the event of repeated restarts.

Phases of ARIES algorithm

(a) Analysis Phase

- It first finds dirty pages(data changes those are not committed to database) in the available buffer pool
- It also identifies all active transactions at the time of the system crash.
- Identify Redo LSN from which redo should start.

(b) Redo

- In order to restore database system will repeats all actions performed on database from start of log or from any selected point in log or from RedoLSN.
- Then it restores the database state to state at which it was at the time of the system crash.
- RecLSN and RedoLSN avoid redo action already reflected on page.

(c) Undo

- It reverts or undoes all operations of transactions which are not committed.
- So after above action now database only reflects actions which are committed transactions.

Q. 6 What is database deadlock? Explain various deadlock handling techniques. (5 Marks)

Ans. :

Concept of Deadlock

- A system is said to be in a state of deadlock if there exists a set of transactions such that every transaction in the set is waiting for another transaction to complete its execution.
- There are two principle methods to handle deadlock in system,

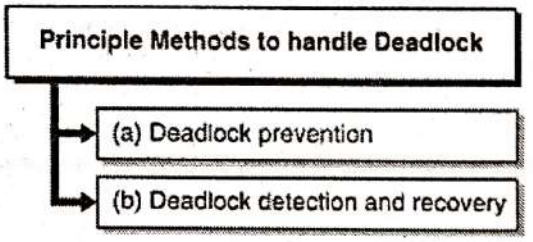


Fig. 12.4 : Principle Methods to handle Deadlock

(a) Deadlock prevention

- Use deadlock prevention techniques to ensure that the system will never enter a deadlock state.

(b) Deadlock detection and recovery

- Allow the system to enter a deadlock state and then detect such state by **deadlock detection techniques** and try to recover from deadlock state by using and **deadlock recovery scheme.**
- Both of above methods may result in transaction rollback.
- Deadlock prevention is commonly used if the probability that the deadlock occurs in system is high else detection and recovery are more efficient if probability of deadlock occurrence is less.

Q. 7 Explain dead lock prevention. (5 Marks)

Ans. :

Dead Lock Prevention

(1) Approach 1

A simplest form, in which a transaction acquires lock on all data items which will be required by transaction at the start of execution.

It is effective as other transactions can't hold lock on those data items till first unlocks data item.

Disadvantages

(i) It is difficult to know in advance which data items need to be locked.

(ii) Data utilization is very low as may be unused data items locked by transaction.

(2) Approach 2

This approach uses timestamp ordering of data items. It is something like tree protocol and every transaction have to access data item in given sequence only. The variation of this approach with two phase protocol assures deadlock prevention. Order of data items must be known to every transaction.

(i) To have concurrency control, two phase locking protocol is used which will ensure locks are requested in right order.

(ii) Timestamp ordering is determined by validation (T_i) i.e. TS (T_i) = validation (T_i) to achieve serializability.

(iii) The validation test for transaction T_j requires that for all transaction T_i with TS (T_i) < TS (T_j) then one of the following conditions must be hold.

(a) Finish (T_i) < start (T_j) as T_i finishes before T_j starts the serializability should be maintained.

(b) T_i completes its write phase before T_j starts its validation phase (Start (T_j) < finish (T_i) < Validation (T_j)).

(iv) This ensures writes of both transactions do not take place at same time.

(v) Read of T_j not affected by writes of T_j and T_j can't affect read of T_i so serializability is maintained.

(3) Approach 3 : Prevention and transaction rollbacks

Pre-emptive technique

(i) Pre-emption means, if transaction T_i wants to hold lock on data item hold by T_j. then system may preempt (UNLOCK all previous locks) T_i by rolling it back and granting lock to T_j on that data item.

(ii) To control this pre-emption every transaction is assigned a unique timestamp.

(iii) System uses this timestamp to decide whether to wait or rollback the transaction.

(iv) The transaction retains its old time stamp if it is rollback and restarted.

(v) Various deadlock prevention techniques using timestamps are as follows :

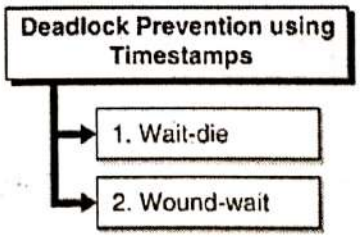


Fig. 12.5 : Deadlock Prevention using Timestamps

Q. 9 Explain deadlock detection and deadlock recovery. (5 Marks)

Ans. :

Deadlock Detection and Recovery

Deadlock detection

(i) Deadlock can be detected using directed graph called as **wait-for-graph.**

(ii) The graph G = (V, E) can be seen as V is of vertices i.e. set of transaction in execution concurrently and E is set of edges.

(iii) Such that edge $T_i \rightarrow T_j$ if $T_i \rightarrow T_j$ is present in graph it shows that, transaction T_j is waiting for transaction T_i to release a data item it needs.

(iv) If cycle is present in wait-for-graph then deadlock is present and transactions in cycle are deadlock.

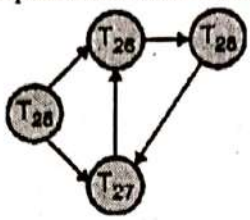


Fig. 12.6 : Wait-for graph with a cycle

(v) To detect deadlock, system must maintain wait-for-graph and periodically invoke an algorithm that searches cycle in the graph.

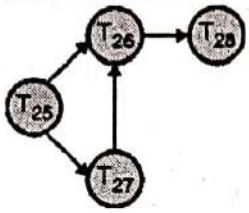


Fig. 12.7 : Wait-for graph with no cycle

(vi) If no cycle is present it mean no deadlock in system.

(vii) If deadlock occurs frequently, then detection algorithm should be invoked more frequently problem in this scenario is the data items locked by deadlock transaction will be unavailable until the deadlock is problem.

(viii) This may tend to more cycles in graph degrading capacity of granting lock requests.

Recovery from deadlock

- When deadlock is detected in the system, then system should be recovered from deadlock using recovery schemes.
- A most common solution is rollback one or more transaction to break the deadlock.
- Methods for recover from deadlock,

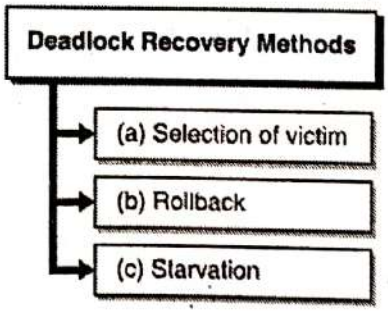


Fig. 12.8 : Deadlock Recovery Method

(a) Selection of victim

- If deadlock is detected then a transaction one or more transactions (victims) to be selected to break the deadlock.
- Transactions with minimum cost should be selected for rollbacks.
- Cost can be detected by following factors.
  - For how much time transaction has computed and how much time it needs to do.
  - How many data items it has locked.
  - How many data items it may need ahead.

- Number of transaction to be rollback.

(b) Rollback

- Once victims are decided then there are two ways to do so :
  - (i) Total rollback
  - (ii) Partial rollback
- But this approach needs to record some more information like state of running transactions, locks on data item held by them, and deadlock detection algorithm specifies points up to which transaction to be rollback and recovery method has to rollback.
- After sometime transaction resume; partial transaction.

(c) Starvation

- It may happen every time same transaction is selected as victim and this may lead to starvation of that transaction (minimum cost transaction it selected every time).
- System should take care that every time same transaction should not be selected as victim. So it will not be starved.

❑❑❑

Database Management Systems

Statistical Analysis

Chapter No.	Dec. 2018	May 2019
Chapter 1	10 Marks	5 Marks
Chapter 2	15 Marks	30 Marks
Chapter 3	15 Marks	12 Marks
Chapter 4	20 Marks	13 Marks
Chapter 5	10 Marks	15 Marks
Chapter 6	15 Marks	15 Marks
Chapter 7		
Chapter 8		5 Marks
Chapter 9	10 Marks	10 Marks
Chapter 10	15 Marks	15 Marks
Chapter 11		
Chapter 12	20 Marks	20 Marks

Dec. 2018

Chapter 1 : Introduction Database Concepts

Q. 2(b) List 5 Significant differences between file processing system and Database Management System.

(10 Marks)

Ans. :

Database Management System	File Processing System
Computerized record – keeping system is used in DBMS	Collection of individual files accessed by applications programs is called File Processing System
DBMS allows flexible access to data	File – Processing System is designed to allow predetermined access to data
It co-ordinates both the physical and logical	It co-ordinates only the physical access to data
DBMS provides multiple user interface	Data is isolated in the file system
Unauthorized access is restricted in DBMS	Unauthorized access cannot be restricted
Redundancy can be controlled	Redundancy cannot be controlled

Chapter 2 : Database Architecture

Q. 1(a) Define DBA. Discuss role of DBA **(5 Marks)**

Ans. :

Database Administrator (DBA)

- The database administrator is responsible for the overall planning of the company's data resources, for the design of data, and for the day-to-day operational aspects of data management.
- A database administrator is a person responsible for the installation, configuration, upgradation, maintenance and monitoring databases in an organization.

Roles of DBA

- The DBA needs to performs many roles to keep the database up and running,

System Administrator / Designer

- The database administrator need to manage DBMS software and server.
- He is also responsible for deciding on the storage and access methods.
- The DBA performs all data field updates or adding new fields into database.

Database Developer / Programmer

- The DBA writes the programs to design database and to design the means of reorganizing databases periodically.
- The DBA also determines and implement database searching strategies.

System Analyst

- The DBA needs to analyse the system performance and fine tune the DBMS activities.
- DBA needs to take care of system crashes by planning proper recovery procedures.
- He will also specify techniques for monitoring database performance.

Q.3 (a) Explain Overall Architecture of DBMS in detail. **(10 Marks)**

Ans. : DBMS Architecture

- A database system can be separated into two different modules that deal with all operations of the overall system.

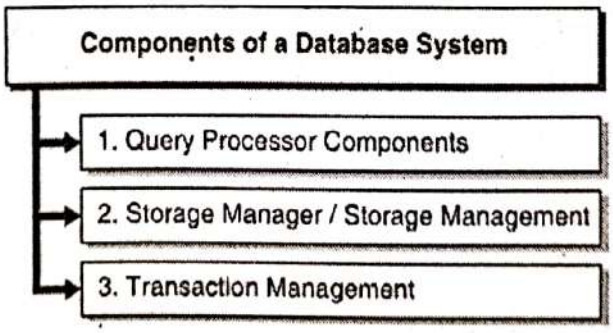


Fig. 1-Q.3 (a): Components of a Database System

- The storage manager is important because databases typically require a huge amount of storage space.

Query Processor Components

Parts of query processor

(i) DDL interpreter

(ii) DML compiler

(iii) Query evaluation engine

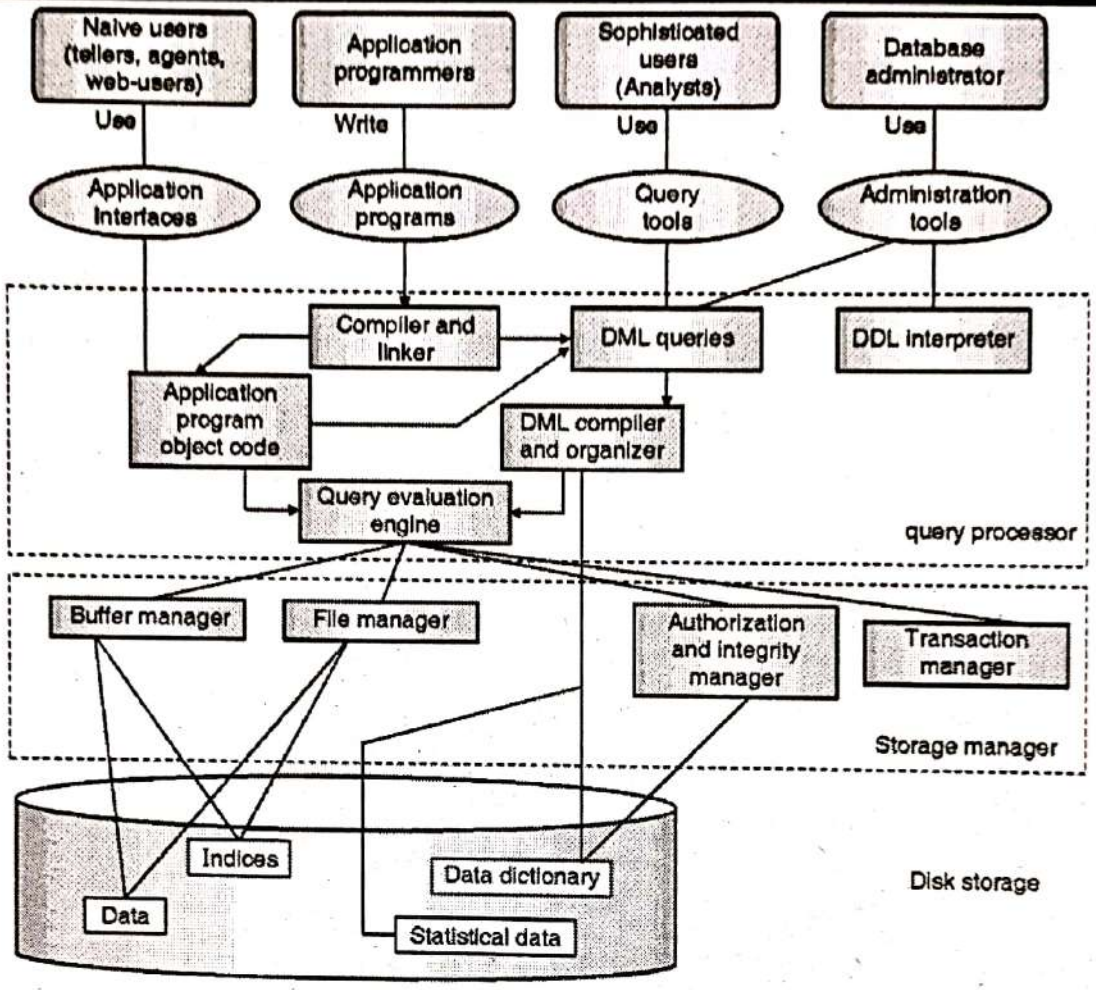


Fig. 2-Q.3 (a) : Components of DBMS

Storage Manager / Storage Management

- A **storage manager** is a program module which acts like interface between the data stored in the database and the application programs and queries submitted to the system.
- The data is stored on the disk using the file system.
- The storage manager is programme which is responsible for the interaction with the file manager.
- storage manager is responsible for storing, retrieving and updating data in the database.
- The storage manager components include :

Authorization and integrity manager Transaction manager

File manager

Buffer manage

Transaction Management

- A transaction is a series of small database operations that together form a single large operation.
- A transaction is started by issuing a BEGIN TRANSACTION command. Once this command is executed the DBMS starts monitoring the transaction.
- All operations executed after a BEGIN TRANSACTION command are treated as a single large operation.
- Application programs use transactions to execute sequences of operations when it is important that all the operations are successfully completed.
- Transaction management component will ensure the atomicity and durability properties.

Chapter 3 : Database Design Using ER Model

Q. 1b) Explain Components of ER Model **(5 Marks)**

Ans. :

ER Diagrams having components,

Entity

- Entity is anything in real world which may have physical or logical existence.
- An entity is anything in real world with its physical existence. Example, Student, faculty, subject having independent physical existence.

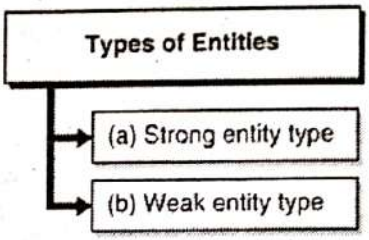
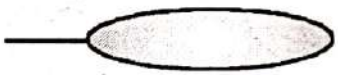


Fig. 1 - Q.1(b): Types of entities

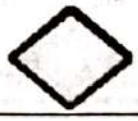
Attributes

- Each entity has its own properties which describes that entity such properties are known as attributes.
- The attribute value that describes each entity becomes a major part of data stored in database.

Type	Notation
Attribute (Simple/Single valued/Stored)	

Relationships

- A relationship is an association among one or more than one entities.
- Use diamond shape to show relationship.

Type	Notation
Relationship	

Types of Relationship based on degree are,

- Unary Relationship
- Binary Relationship
- Ternary Relationship

Q.6 (c) Write short notes on Specialization and Generalization. **(10 Marks)**

Ans. : Specialization

- Specialization is a process of defining a set of subclass of entity type, this entity type is called super **class of specialization.**
- The set of subclasses that forms a specialization is defined on the basis of some distinguishing characteristic of entity in super class.

Example :

Set of subclass (Saving_Account, Current_Account) are Specialization of super class Account.

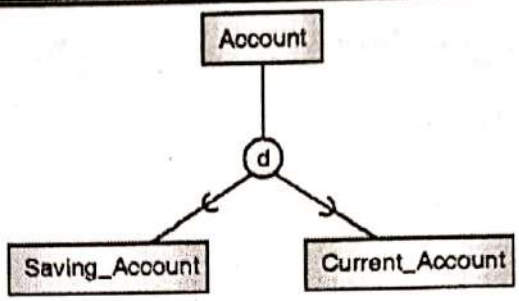


Fig. 1 - Q.6(c): Specialization

Generalization

Generalization is a process in which we differentiate among several entity types identifying there common features and generalizing them to a single super class of which original entity type are special subclass.

Example :

- Car and Bike all having several common attribute they can generalize to the super class vehicle.
- The attributes of higher and lower level entities created by specialization and generalizations are attributes inheritance.
- Abstraction through which relationship (aggregation) is treated as higher level entities.

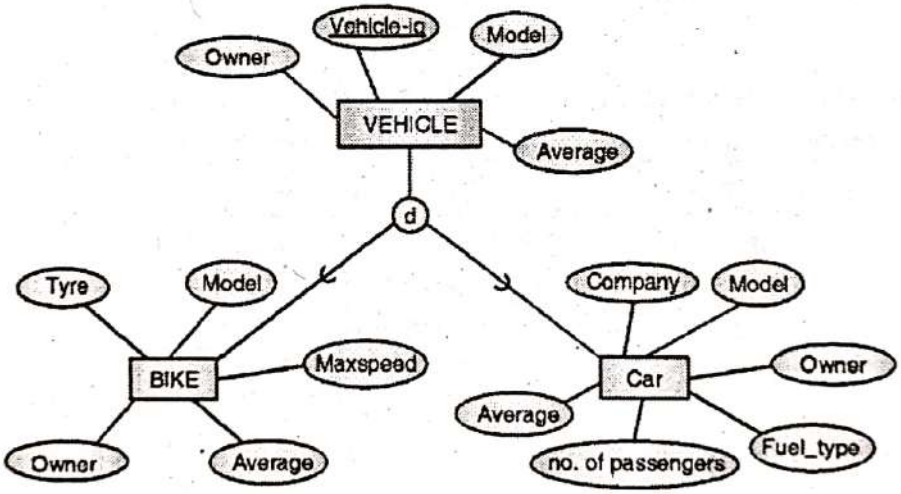


Fig. 2 - Q.6(c): Generalized VEHICLE entity

Chapter 4 : Relational Data Model

Q.3b) Construct ER diagram and convert into Relational Model for Company (10 Marks)
Which has several Employees working on different types of projects. Several Employees are working on one department. Every Employee has Manager. Several Employees are supervised by one Employee.

Ans. :

1. Employee (Eid, Ename, mid, sup-id),Pid)
2. Company(Cid, Cname, Location)
3. Project (Pid, Pname, type)

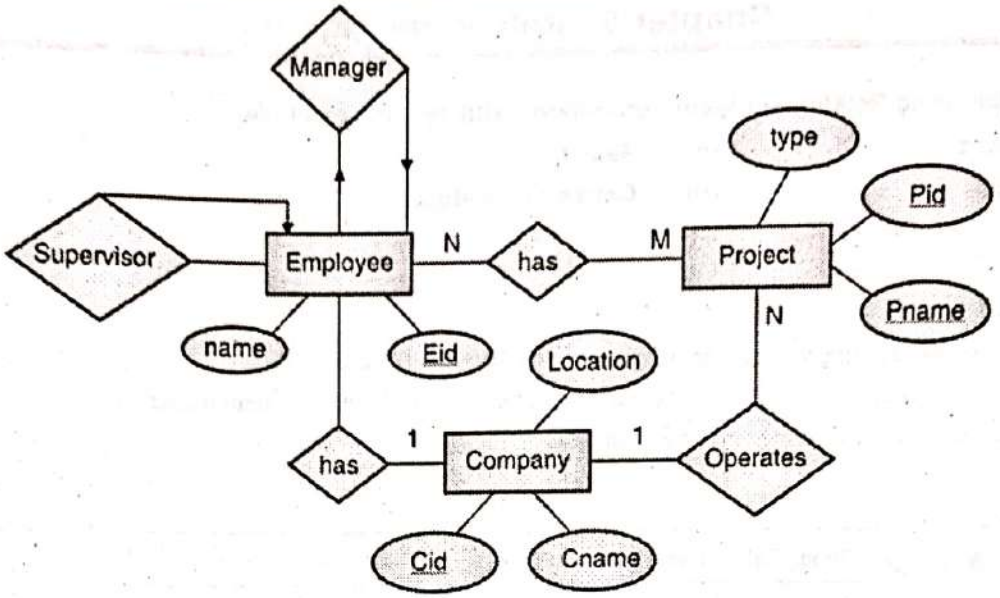


Fig.1-Q. 3(b)

Q. 6(b) Write short note on Constraints in SQL **(10 Marks)**

Ans. :

Constraints in SQL

(i) Domain Relational Constraint

- Domain constraints allow us to test whether the values inserted into the database are correct or not.
- The CREATE TABLE Command may also include domain constraints which can check integrity of database.
- These domain constraints are the most basic form of integrity constraint.

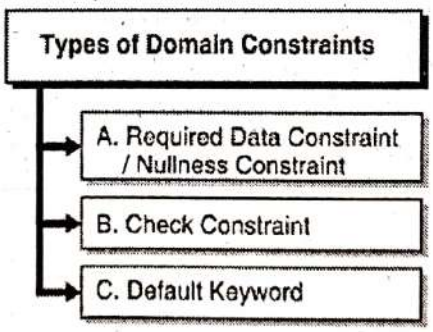


Fig. 1 - Q.6(b): Types of Domain Constraints

(ii) Entity Integrity Constraints

- Entity constraints allow us to test whether the tuple (entity) inserted into the database are correct or not.
- The create table Command may also include entity constraints which can primary key of table.
- Types of Entity Constraints

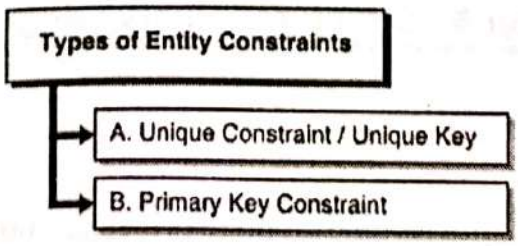


Fig. 2 - Q.6(b): Types of Entity Constraints

Chapter 5 : Relational Algebra

Q.4 b) Explain following Relational Algebra operations with suitable example (10 Marks)

(a) Project (b) Select

(c) Union (d) Cartesian Product

Ans. :

(a) Project

- This operator is used for selecting some of many columns in table to displayed in result set.
- Projection operator can select a column or set of columns of table to be displayed in output of query.
- This is unary relational operator having only one input table.

Syntax

$\pi_{<column\_list>}(Input\_Table\_Name)$

- (b) Select
- This operator is used to select some rows from table which satisfy particular selection condition given in selection operation.
- Selection operator selects a set of tuples that satisfy a selection predicate or condition.
- Output of query is exactly same as input schema of table.
- This is unary relational operator having only one input table.

Syntax

$\sigma_{<attribute\_name><comparison\_operator><constant\_value>}(Input\_Table\_Name)$

(c) Union

- This operator finds out all combined rows in table 1 and table 2.
- Union effectively appends the result of first query to the result of second query.
- It does not eliminate all duplicate rows and they are printed in result expression.

Syntax

(Query Expression 1) $\cup$ (Query Expression 2)

(d) Cartesian Product

- A cross join performs relational product or Cartesian product of two tables specified in query.
- In this case every row in first table will be joined with every row in second table. So finally number of rows in result table will be equals to product of number of rows in table 1 and number of rows in table 2.
- That means all rows in the first table are joined to all rows in the second table.

Syntax

(Query Expression 1) $\times$ (Query Expression 2)

Chapter 6 : Structured Query Language

Q. 1(d) Explain Database Languages (5 Marks)

Ans. : Data Definition Language (DDL)

- To create database schema and database objects like table **Data Definition Language (DDL)** can be used.
- DDL statements are used to build and modify the structure of your tables and other objects in the database.

- The set of DDL commands are as below,
  1. CREATE Statement : To create Database objects
  2. ALTER Statement : To modify structure of database objects
  3. DROP Statement : To remove database objects
  4. RENAME Statement : To Rename Database objects
  5. TRUNCATE Statement : To empty the database table

  Example : Table, View, Sequence etc.

Data Manipulation Language (DML)

- **DML** statements are used for manipulating or managing data in database.
- DML commands are not auto-committed like DDL statements.
- It means changes done by DML command can be rolled back. Or in other words the DML statements do not implicitly commit the current transaction.
- DML is set of commands used to,

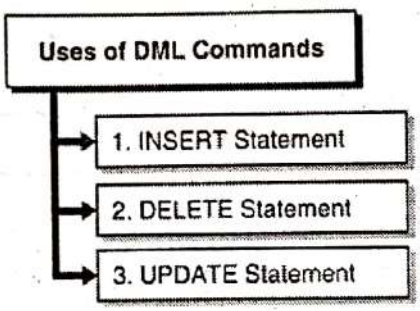


Fig. 1 - Q.1(d): Types of entities

Data Control Language (DCL)

- **Data Control Language (DCL)** is used to control various user actions (or privileges) in Database.
- To perform any operation in the database user needs **privileges** like creating tables, sequences or views.
- DCL is set of commands used to,
  - **Grant :** Gives some privilege to user for performing task on database.
  - **Revoke :** Take back permissions given from user.
- Privileges can be of many types,
  - **System Privileges :** creating a table is types of system privilege.
  - **Object Privileges :** To execute query on tables object privilege can be used.
  - **Ownership Privileges :** To execute query on tables created by same user.

Q. 5(a) **Employee (eid, ename, address, city)** **(10 Marks)**

Works (eid, cid, salary)

Company(cid, cname, city)

1. **Modify database so that John now lives in Mumbai**
2. **Find Employees who live in same city as the company for which they work.**
3. **Give all employees of "AZ Corporation" where there is increase in salary by 15%.**
4. **Find the names of all employees, company name and city of residence such that Employee name begins with 'I'**
5. **Delete all tuples in works relation for employees of small bank corporation.**

Ans. :

1. Update employee

```
Set city = 'Mumbai'
Where name = 'John';
```

2. Sectect e-ename

```
From    employee e, company c, works w
When    e.eid = w.eid
And     w.Cid = c.cid
And     e.city = C.city ;
```

3. update employee

```
Set     sal = 1.15 * sal
Where eid = (Sected eid from emp
             where Cid = (select eid
                          from comp
                          where name)
```

4. select e.ename e.city, c. cname

```
select  from e.city,
select where e.city
select  And e.city
And e name like 'J'
```

5. delete

```
from employ
When eid = (Sectect eid
        Form works
        Where Cid = (Sectect Cid from comp
```

Chapter 9 : Relational Database Design

Q. 5(b) Define Normalization. Discuss 1 NF, 2 NF and 3 NF In Detail. **(10 Marks)**

Ans. :

First Normal Form (1NF)

Definition

1NF states that all attributes in relation must have atomic (indivisible) values and all attribute in a tuple must have a single value from the domain of that attribute.

Example :

- Consider an employee table with columns as shown in diagram,
- The relational schema not in 1 NF is represented as,

Table 1 - Q.5(b) (Non-Normalised)Employee Table

Employee_Id	Ename	Salary	Ecity
10	Mahesh	50000	Mumbai, Pune
12	Suresh	25000	Mumbai
15	Ganesh	26000	Pune
18	Kasturi	50000	Mumbai, Delhi

- To convert relational schema in 1NF, the Ecity attribute is divided in atomic domains it may introduce some data redundancy.

Table 2 - Q.5(b) (1NF Employee Table

Employee_Id	Ename	Salary	Ecity
10	Mahesh	50000	Mumbai
10	Mahesh	50000	Pune
12	Suresh	25000	Mumbai
15	Ganesh	26000	Pune
18	Kasturi	50000	Mumbai
18	Kasturi	50000	Delhi

Second Normal Form (2NF)

Definition

A relation is in 2NF, if it is in 1NF and all non-key attributes in relation are fully functionally dependent on the primary key of the relation.

Example :

Consider an employee table with columns as shown in diagram,

Table 3 - Q.5(b) (: Non-2NF Employee Table

Employee_Id	Ename	Salary	Project_Id	Hours	Allowance
10	Mahesh	50000	E001	44	40000
12	Suresh	25000	B056	31	30000
15	Ganesh	26000	C671	23	20000
18	Mahesh	50000	E002	12	15000
15	Ganesh	26000	E001	24	20000
18	Mahesh	50000	B056	11	10000

- To normalize above schema to 2NF we can decompose tables as,

  Employee (Employee_Id, Ename, Salary)

  Employee_Id → Ename, Salary

Table 4 - Q.5(b) (: 2NF Employee Table

Employee_Id	Ename	Salary
10	Mahesh	50000
12	Suresh	25000
15	Ganesh	26000
18	Mahesh	50000

Third Normal Form (3NF)

A relation is in 3NF, if it is in 2NF and all non-prime attributes of the relation are non-transitively dependent on the every key.

Example :

- Consider an Employee table with following FDs,

  Employee_Id → Ename, Salary, Department_Id

  Department_Id → Dname

- The state of Employee relational is,

Table 5 - Q.5(b) : Employee Table

Employee_Id	Ename	Salary	Department_Id	Dname
10	Mahesh	50000	C1	IT
12	Suresh	25000	E2	HR
15	Ganesh	26000	C1	IT
18	Mahesh	50000	E2	HR

- To normalize above schema to 3NF , decompose tables as,

  Employee (Employee_Id, Ename, Salary, Department_Id)

  Employee_Id → Ename, Salary, Department_Id

Table 6 - Q.5(b) : 3NF Employee Table

Employee_Id	Ename	Salary	Department_id
10	Mahesh	50000	C1
12	Suresh	25000	E2
15	Ganesh	26000	C1
18	Mahesh	50000	E2

Table 7 - Q.5(b) : 1NF Employee Table

Employee_Id	Ename	Salary	Ecity
10	Mahesh	50000	Mumbai
10	Mahesh	50000	Pune
12	Suresh	25000	Mumbai
15	Ganesh	26000	Pune
18	Kasturi	50000	Mumbai
18	Kasturi	50000	Delhi

Chapter 10 : Transaction

Q. 1(c) Explain ACID Properties of transaction **(5 Marks)**

Ans. :

1. Atomicity

- Transaction must be treated as a single unit of operation.
- That means when a sequence of operations is performed in a single transaction, they are treated as a single large operation.

Examples :

(a) Withdrawing money from your account.

(b) Making an airline reservation.

- The term atomic means thing that cannot be divided in parts as in atomic physics.
- Execution of a transaction should be either complete or nothing should be executed at all.
- No partial transaction executions are allowed. (No half done transactions)

2. Consistency

- Consistent state is a database state in which all valid data will be written to the database.
- If a transaction violates some consistency rules, the whole transaction will be rolled back and the database will be restored to its previous consistent state with those rules.
- On the other hand, if a transaction is executed successfully then it will take the database from one consistent state to another consistent state.
- DBMS should handle an inconsistency and also ensure that the database is clean at the end of each transaction.
- Consistency means transaction will never leave your database in a half finished (inconsistent) state.
- If one part of the transaction fails, all of the pending changes made by that transaction are rolled back.

Example : Money transfer in above example

- Initially in total balance in accounts A is 1000 and B is 5000 so sum of balance in both accounts is 6000 and while carrying out above transaction some type of failure occurs after Write (A) but before Write (B) then system may lose 100 rupees in calculation.
- As now sum of balance in both accounts is 5900 (which should be 6000) which is not a consistent result which introduces inconsistency in database.
- This means that during a transaction the database may not be consistent.

3. Isolation

- Isolation property ensures that each transaction must remain unaware of other concurrently executing transactions.
- Isolation property keeps multiple transactions separated from each other until they are completed.
- Operations occurring in a transaction (example, insert or update statements) are invisible to other transactions until the transaction commits (on transaction success) or rolls back (on transaction fails).
- For example, when a transaction changes a bank account balance other transactions cannot see the new balance until the transaction commits.
- Different isolation levels can be set to modify this default behaviour.
- Transaction isolation is generally configurable in a variety of modes. For example, in one mode, a transaction locks until the other transaction finishes.
- Even though many transactions may execute concurrently in the system. System must guarantees that, for every transactions (T_i) all other transactions has finished before transactions (T_i) started, or other transactions are started execution after transactions (T_i) finished.
- That means, each transaction is unaware of other transactions executing in the system simultaneously.

Example : Money transfer in above example.

- The database is temporarily inconsistent while above transaction is executing, with the deducted total written to A and the increased total is not written to account B. If some other concurrently running transaction reads balance of account A and B at this intermediate point and computes A+B, it will observe an inconsistent value (Rs. 5900). If that other transaction wants to perform updates on accounts A and B based on the inconsistent values (Rs. 5900) that it read, the database may be left database in an inconsistent state even after both transactions have completed.
- A way to avoid the problem of concurrently executing transactions is to execute one transaction at a time.

4. Durability

- The results of a transaction that has been successfully committed to the database will remain unchanged even after database fails.
- Changes made during a transaction are permanent once the transaction commits.
- Even if the database server is fails in the between transaction, it will return to a consistent state when it is restarted.
- The database handles durability by transaction log.
- Once the execution of the above transaction completes successfully, and the user will be notified that the transfer of amount has taken place, if there is no system failure in this transfer of funds.
- The durability property guarantees that, all the changes made by transaction on the database will be available permanently, although there is any type of failure after the transaction completes execution.

Q. 4(a) Explain the concept of Serializability with its types. (10 Marks)

Ans. : Serializability / Serializable Schedule

- The database system must control above concurrent execution of transactions serializability will ensure that the database state remains in consistent state.

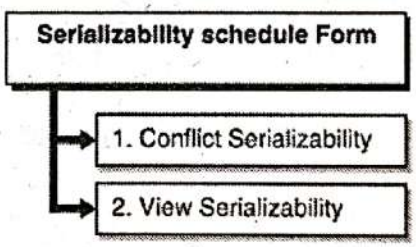


Fig.1 - Q.4(a) : Serializability schedule form

Conflict Serializability

Conflict

- A pair of consecutive database actions (reads, writes) is in conflict if changing their order would change the result of at least one of the transactions.

		Transaction T_j	
		Read(D)	Write(D)
Transaction T_i	Read(D)	No Conflict	Conflict
	Write(D)	Conflict	Conflict

- Consider schedule S has two consecutive instructions I_i and I_j from transactions T_i and T_j respectively.
- If I_i and I_j access to different data items then they will not conflict and can be swapped, without any problem.
- If I_i and I_j **access to same data item D** then consider following consequences :

  I_i = READ (D), I_j = READ (D) then **no conflict** as they only read value.

  This operation is called as non conflicting swap.

  I_i = READ (D), I_j = WRITE (D) then they **conflict** and cannot be swapped.

  I_i = WRITE (D), I_j = READ (D) then they **conflict** and cannot be swapped.

I_i = WRITE (D), I_j = WRITE (D) then they **conflict** and cannot be swapped.

- So we can say that instructions conflict if both consecutive instructions operate on same data item and from different transactions and one of them is WRITE operation.
- If I_i and I_j **access to different data item D** then consider following all consequences **no conflict** as they only read or writing different values.
- I_i = READ (D)/WRITE (D), I_j = READ (P) /WRITE (P) then **no conflict** as they only reading or writing different data.
- The following set of actions is conflicting :

  T_1:R(X), T_2:W(X), T_3:W(X)
- While the following sets of actions are not :

  T_1:R(X), T_2:R(X), T_3:R(X)

  T_1:R(X), T_2:W(Y), T_3:R(X)

View Serializability

- View equivalence is less strict than conflict equivalence, but it is like conflict equivalence based on only the read and write operations of transactions.

  Conditions for view equivalence

Let, D = Data item

S_1, S_2 = Transaction schedules

T_i, T_j = Database transaction

- Schedules S_1 and S_2 are view equivalent if they satisfy following conditions for each data item D,
  (a) S_1 and S_2 must have same transactions included and also they are performing same operations on same data. If T_i reads initial value of D in S_1, then T_i also reads initial value of D in S_2.
  (b) If T_i reads value of D written by T_j in S_1, then T_i also reads value of D written by T_j in S_2.
  (c) If T_i writes final value of D in S_1, then T_i also writes final value of D in S_2.
- First 2 conditions ensure that transaction reads same value in both schedules.
- Condition 3 ensures that final consistent state.
- If a concurrent schedule is view equivalent to a serial schedule of same transactions then it is **View serializable.**
- Consider following schedule S_1 with concurrent transactions $<T_1, T_2, T_3>$.
- In both the schedules S_1 and a serial schedule $S_2<T_1, T_2, T_3>$ T_1 reads initial value of D. Transaction T_3 writes final value of D. So schedule S_1 satisfies all three conditions and is view serializable to $<T_1, T_2, T_3>$.

Chapter 12 : Recovery System

Q. 2(a) Define Deadlock. Explain Deadlock Detection, Prevention and Recovery. (10 Marks)

Ans. : Deadlock

- A system is said to be in a state of deadlock if there exists a set of transactions such that every transaction in the set is waiting for another transaction to complete its execution.
- There are **two principle methods** to handle deadlock in system,

Approaches for Deadlock Prevention

Approach 1

- A simplest form, in which a transaction acquires lock on all data items which will be required by transaction at the start of execution. It is effective as other transactions can't hold lock on those data items till first unlocks data item.

Approach 2

This approach uses timestamp ordering of data items. It is something like tree protocol and every transaction have to access data item in given sequence only. The variation of this approach with two phase protocol assures deadlock prevention. Order of data items must be known to every transaction.

Approach 3 : Prevention and transaction rollbacks

Pre-emptive technique

(i) Pre-emption means, if transaction T_i wants to hold lock on data item hold by T_j, then system may preempt (UNLOCK all previous locks) T_i by rolling it back and granting lock to T_j on that data item.

(ii) To control this pre-emption every transaction is assigned a unique timestamp.

(iii) System uses this timestamp to decide whether to wait or rollback the transaction.

(iv) The transaction retains its old time stamp if it is rollback and restarted.

(v) Various deadlock prevention techniques using timestamps are as follows :

Deadlock Detection and Recovery

Deadlock detection

(i) Deadlock can be detected using directed graph called as **wait-for-graph.**

(ii) The graph G = (V, E) can be seen as V is of vertices i.e. set of transaction in execution concurrently and E is set of edges.

(iii) Such that edge $T_i \rightarrow T_j$ if $T_i \rightarrow T_j$ is present in graph it shows that, transaction T_j is waiting for transaction T_i to release a data item it needs.

(iv) If cycle is present in wait-for-graph then deadlock is present and transactions in cycle are deadlock.

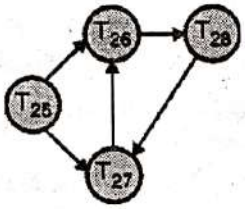


Fig. 1 - Q.2 (a): Types of entities : Wait-for graph with a cycle

(v) To detect deadlock, system must maintain wait-for-graph and periodically invoke an algorithm that searches cycle in the graph.

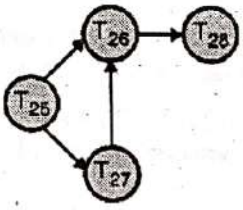


Fig. 2 - Q.2 (a): Wait-for graph with no cycle

(vi) If no cycle is present it mean no deadlock in system.

(vii) If deadlock occurs frequently, then detection algorithm should be invoked more frequently problem in this scenario is the data items locked by deadlock transaction will be unavailable until the deadlock is problem.

(viii) This may tend to more cycles in graph degrading capacity of granting lock requests.

Recovery from deadlock

- When deadlock is detected in the system, then system should be recovered from deadlock using recovery schemes.
- A most common solution is rollback one or more transaction to break the deadlock.

Deadlock Recovery Methods

(a) Selection of victim

- If deadlock is detected then a transaction one or more transactions (victims) to be selected to break the deadlock.
- Transactions with minimum cost should be selected for rollbacks.

- Cost can be detected by following factors.
- For how much time transaction has computed and how much time it needs to do.
- How many data items it has locked.
- How many data items it may need ahead.
- Number of transaction to be rollback.

(b) Starvation

- It may happen every time same transaction is selected as victim and this may lead to starvation of that transaction (minimum cost transaction it selected every time).
- System should take care that every time same transaction should not be selected as victim. So it will not be starved.

Q. 6(a) Write short note on Log Based Recovery **(10 Marks)**

Ans. :

Log Based Recovery

- There can be problem in accessing database due to any reason causes a database system failure.
- The most widely used structure for recording database modifications is **Transaction log (or log).**
- The log is a sequence of log records, recording all the update activities done on the database by all database users.

Transaction log

- Transaction log records are maintained to record various events during transaction processing.
- There are several types of log records.

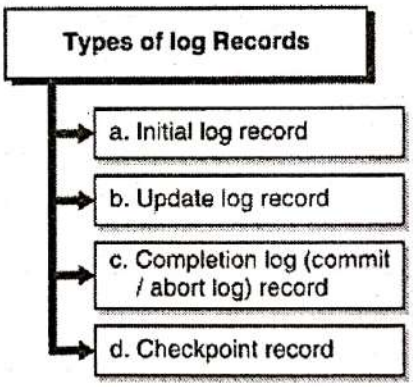


Fig. 1 - Q.6 : Types of log records

May 2019

Chapter 1 : Introduction Database Concepts [Total Marks - 05]

Q. 1(a) Differentiate between file system and database system with an example. **(5 Marks)**

Ans. : Please refer Q.2(b) of Dec. 2018.

Chapter 2 : Database Architecture [Total Marks - 30]

Q. 2(b) Construct an EER diagram and convert into Relational Model for a library Management System.

Specify 2 complex SQL queries on the above-one using Group by clause and the other using Join operation with an example. **(10 Marks)**

Ans. :

1. EER diagram and convert into Relational Model for a library Management System.

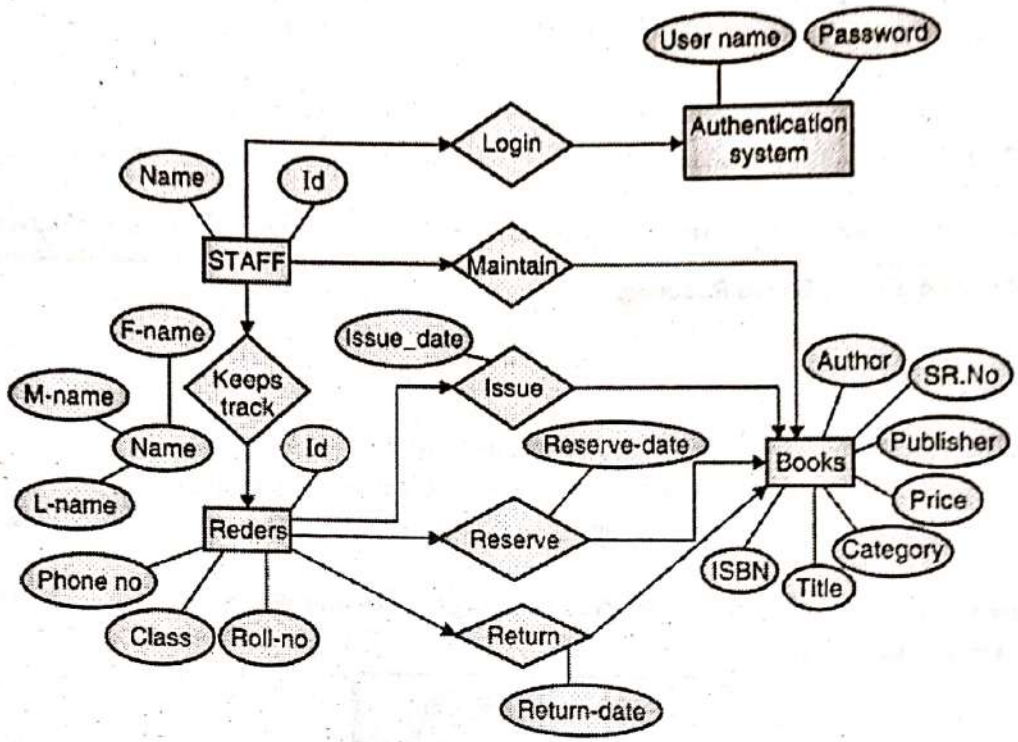


Fig.1-Q.2(b)

STAFF (Id,Name,Phono)

AUTH_Details (UserID, Pwd, Id)

READERS (Phono, Class, RollNo, Id, Name)

Books (ISBN, Title, Category, Price, Publisher, Srno, Author)

Reserve (Phono,ISBN)

Return (Phono,ISBN)

2. Two complex SQL queries on the above-one using Group by clause and the other using Join operation with an example.

To print publisher wise count of books.

```
SELECT ISBN, count(*)
FROM   Book
GROUP BY ISBN
Print user id and respective login details
SELECT UserID, Pwd, RollNo
FROM   AUTH_Details a,READERS r
WHERE a.id=r.id;
```

Q.3 (b) Explain the Overall Architecture of DBMS in detail. **(10 Marks)**

Ans. : Please refer Q.3(a) of Dec. 2018.

Q. 6(d) Write short note on Data Independence **(10 Marks)**

Ans. :

Data Independence

Definition : **Data Independence** can be defined as the capacity to change one level of schema without changing the schema at the next higher level.

Types

a) Logical data independence

- Logical data independence is a capacity to change the conceptual schema without having any changes to external schemas. (or application programs)
- Separating the external views from the conceptual view enables us to change the conceptual view without affecting the external views. This separation is sometimes called logical data independence.

Example :

We may change the conceptual schema by removing a data item. In this case the external schemas that refer only to the remaining data should not be affected.

(b) Physical data independence

- Physical data independence is a capacity to change the internal schema without having any changes to conceptual schema.
- The separation of the conceptual view from the internal view enables us to provide a logical description of the database without the need to specify physical structures. This is often called physical data independence.

Example :

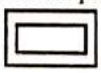
By creating additional access paths to improve the performance of retrieval. If the same data as before remains in the database, we should not have to change the conceptual schema.

Chapter 3 : Entity Relationship Data Model [Total Marks -12]

Q. 3(a)(iii)Explain the terms Weak Entity with an example . **(2 Marks)**

Ans. :

Weak Entity

- These types of entities are dependent on strong entity for primary key.
- For some weak entities we assign virtual primary key. Such virtual primary key of weak entity is called as '**discriminator**'.
- Weak entity type is represented by double rectangle. 

Example :

In case of "Dependent" entity depend on employee entity for primary key.

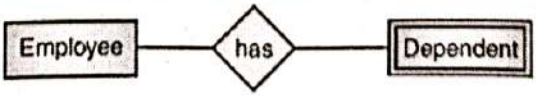


Fig. 3.2.3 : Weak entity "dependent"

Q.4 (b) Explain Specialization and Generalization with suitable example. **(10 Marks)**

Ans. : Please Refer Q.6(c) of Dec. 2018

Chapter 4 : Relational Data Model [Total Marks - 13]

Q. 1(b) Explain Referential Integrity with suitable example. **(5 Marks)**

Ans. :

Referential Integrity

- A value appearing in a one relation (table) for a given set of attributes also appears for another set of attributes in another relation (table). This is called referential integrity.
- The referential integrity constraint is specified between two tables to maintain the consistency among tuples in the two tables.
- The tuple in one relation refers only to an existing tuple in another relation.

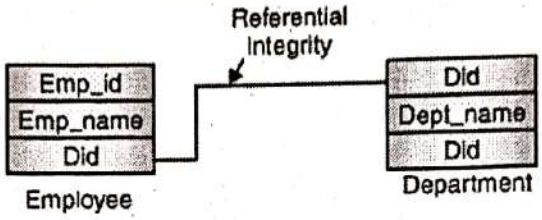


Fig. 1-Q.1(b) : Referential integrity

Employee Table		
Emp_Id	**Emp_name**	**Did**
1	Sachin	20
2	Suhas	10
3	Jay	20
4	Om	10

Department Table	
Did	**Dept_name**
10	HR
20	TIS
30	L&D

- In the above example Employee table has Did as foreign key reference to Did column in Department table this is called as referential integrity.

  Here we are forcing the database to check the value of Did column from the department table while inserting any value in Employee table. This helps to maintain data consistency.

Q. 1(c) List the steps required to map ER, EER model to relational model. **(5 Marks)**

Ans. :

(1) Regular entity types

- **Tables :** Regular entity sets can be represented as table in relational model.
- **Columns :** Attributes of entity set can be converted to the columns (attributes) of the tables in relational model.

Example :

Regular entity employee mapped as employee table in object model like 'Stud_id', 'Stud_Addr' etc. are shown as table columns.

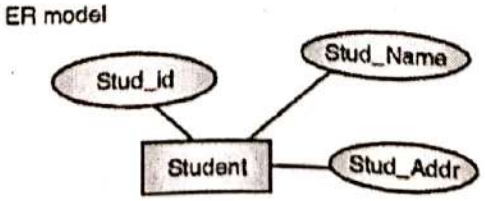


Fig. 1-Q.1(c) : Regular entity

Stud_Id	Stud_Name	Stud_Addr
1	Snehal	Mumbai
2	Pratiksha	Mumbai
3	Supriya	Mumbai
4	Tanmay	Goa

(2) Weak entity types

For each weak entity type with owner entity, create a table and include all simple attributes of weak entity type as columns of table, including foreign key attributes as the primary key attribute of the table that correspond to the owner entity type.

Example :

Dependents (Weak entity) in Employee (Owner entity).

ER Model

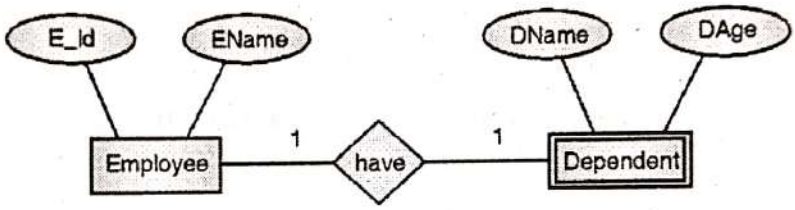


Fig. 2-Q.1(c) : Weak entity

E_id	Ename	DName	DAge
1	Sachin	Jyoti	23
2	Suhas	Manju	22
3	Jayendra	Tanya	27

Q. 3(a)(iv)Explain the terms Foreign Key with an example . (3 Marks)

Ans. : Foreign Key

- If any row in EMP table is added with 'Did' value which is not there in department table the insert statement will give foreign key violation error.

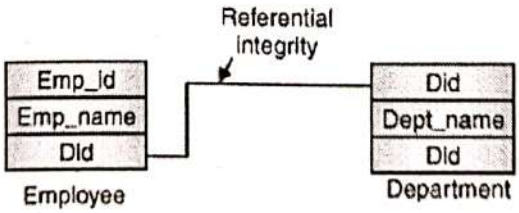


Fig. 1- Q. 3(a)(iv) : Referential integrity

Employee Table		
Emp_Id	Emp_name	Did
1	Sachin	20
2	Suhas	10
3	Jay	20
4	Om	10

Department Table	
Did	Dept_name
10	HR
20	TIS
30	L&D

- In above tables we will refer Department as parent table (as it is containing Primary key) and Employee table as Child table (as it is containing Foreign Key).

Chapter 5 : Relational Algebra [Total Marks - 15]

Q. 2(a) **Explain the following Relational Algebra operations with suitable example.** **(10 Marks)**

(I) Project (II) Select

(III) Union (Iv) Cartesian Product

Ans. : **Please refer Q. 4(b) of Dec. 2018.**

Q. 3(a) **Explain the following terms with an example :-** **(5 Marks)**

(I) Natural join (II) Set Intersection

Ans. :

(i) Natural join

- Natural join can join tables based on the common columns in the tables being joined.
- A natural join returns all rows by matching values in common columns having same name and data type of columns and that column should be present in both tables.

Prerequisites for Natural Join

Both table must have at list one common column with same column name and same data type.

Steps of Working

- The two table are joined using Cross join
- DBMS will look for a common column with same name and data type
- Tuples having exactly same values in common columns are kept in result.

(ii) Set Intersection

This operator finds out all rows that are common in table 1 and table 2.

If Intersect operator is applied on two queries then it will return all rows that are common in the result of Query 1 and Query 2.

Syntax

```
(Query Expression 1) ∩ (Query Expression 2)
```

Chapter 6 : Structured Query Language [Total Marks - 15]

Q. 5(a) **For the schema mentioned below.** **(10 Marks)**

Employee (eid, ename, address, city) Works (eid, cid, salary)

Company (cid, cname, city)

Create an ER diagram for the same and Specify the SQL queries for each of the statements given below.

1. **Modify database so that John now lives in Mumbai, assuming the database entry has John staying in Delhi.**
2. **Find employees who live in same city as the company for which they work.**
3. **Give all employees of "AZ Corporation" whose salary has increased by 15% in the year 2018-19.**

Ans. :

1. Update employee

```
Set city = 'Mumbai'
Where name = 'John';
```

2. Sectect e-ename

```
From    employee e, company c, works w
When    e.eid = w.eid
And     w.Cid = c.cid
And     e.city = C.city ;
```

3. update employee

```
Set     sal = 1.15 * sal
Where eid = (Sected eid from emp
             where Cid = (select eid
                          from comp
                          where name)
```

Q. 6(b) Write short note on Transaction Control Command **(5 Marks)**

Ans. :

Transaction Control Command

Any SQL query can be executed with two basic operations on the database objects :

1. Read 2. Write

- After executing SQL query we must specify its final action as commit (save data) or abort (or revert back changes).
- The **COMMIT** statement ends the operations and makes all changes made to the data permanent, on successful completion.
- ABORT terminates and undoes all the actions done so far.

Commit Transaction

- A query that is successful and has encountered no errors is committed by issuing commit. That is, all changes to the database are made permanent and become visible to other users of the database.
- The syntax is as follows :
- COMMIT [WORK]

Rollback Transaction

- A query that is unsuccessful and has encountered some type of error should be rolled back. That is, all changes to the database are undone and the database remains unchanged by the transaction.
- Transaction-processing systems ensure database integrity by recording intermediate states of the database as it is modified, then using these records to restore the database to a known state if a transaction cannot be committed.

Syntax

```
ROLLBACK
```

Chapter 8 : Trigger [Total Marks - 05]

Q. 6(b) Write short note on Triggers **(5 Marks)**

Ans. :

Triggers

- A trigger is a procedure that is automatically invoked by the DBMS in response to specific alteration to the database or a table in database.
- Triggers are stored in database as a simple database object.
- A database that has a set of associated triggers is called an active database.
- A database trigger enables DBA (Database Administrators) to create additional relationships between separate databases.

Example

Whenever there comes a new student add him to CS (Computer Science).

```
SQL>    CREATE TRIGGER CSAutoRecruit
AFTER INSERT ON Student
FOR EACH ROW
BEGIN
INSERT INTO Take VALUES (111, 'CS');
END;
```

Chapter 9 : Relational Database Design [Total Marks - 10]

Q. 5(b) Define the term Normalization as used in database design. Explain the various normal forms with an example. **(10 Marks)**

Ans. : Please Refer Q.5(b) of Dec. 2018.

Chapter 10 : Transaction [Total Marks - 15]

Q. 1(d) Explain the ACID properties of transactions. **(5 Marks)**

Ans. : Please Refer Q.1 (c) of Dec. 2018.

Q. 6(c) Write short notes on Conflict and View Serialiazability **(10 Marks)**

Ans. : Please Refer Q.4 (a) of Dec. 2018.

Chapter 12 : Recovery System [Total Marks - 20]

Q. 4(a) Define Deadlock. Explain how deadlock can be handled. **(10 Marks)**

Ans. : Please Refer Q.2 (a) of Dec. 2018.

Q. 6(a) Write short note on Log based recovery mechanism. **(10 Marks)**

Ans. : Please Refer Q. 6(a) of Dec. 2018.

□□□

Dec. 2018

Q. 1 (a) Define DBA. Discuss role of DBA **(5 Marks)**

(b) Explain Components of ER Model **(5 Marks)**

(c) Explain ACID Properties of transaction **(5 Marks)**

(d) Explain Database Languages **(5 Marks)**

Q. 2 (a) Define Deadlock. Explain Deadlock Detection, Prevention and Recovery. **(10 Marks)**

(b) List 5 Significant differences between file processing system and Database Management System. **(10 Marks)**

Q.3 (a) Explain Overall Architecture of DBMS in detail. **(10 Marks)**

b) Construct ER diagram and convert into Relational Model for Company **(10 Marks)**

Which has several Employees working on different types of projects. Several Employees are working on one department. Every Employee has Manager. Several Employees are supervised by one Employee.

Q. 4 (a) Explain the concept of Serializability with its types. **(10 Marks)**

(b) Explain following Relational Algebra operations with suitable example **(10 Marks)**

(a) Project (b) Select

(c) Union (d) Cartesian Product

Q. 5 (a) Employee (eid, ename, address, city) **(10 Marks)**

Works (eid, cid, salary)

Company(cid, cname, city)

1. Modify database so that John now lives in Mumbai
2. Find Employees who live in same city as the company for which they work.
3. Give all employees of "AZ Corporation" where there is increase in salary by 15%.
4. Find the names of all employees, company name and city of residence such that Employee name begins with 'I'
5. Delete all tuples in works relation for employees of small bank corporation.

(b) Define Normalization. Discuss 1 NF, 2 NF and 3 NF in Detail. **(10 Marks)**

Q. 6 Write short note on : **(10 Marks)**

(a) Log based recovery

(b) Constraints in SQL

(c) Specialization and Generalization. **(10 Marks)**

May 2019

Q. 1 (a) Differentiate between file system and database system with an example. **(5 Marks)**

(b) Explain Referential Integrity with suitable example. **(5 Marks)**

(c) List the steps required to map ER, EER model to relational model. **(5 Marks)**

(d) Explain the ACID properties of transactions. **(5 Marks)**

Q. 2 (a) Explain the following Relational Algebra operations with suitable example. (10 Marks)

(i) Project (ii) Select

(iii) Union (iv) Cartesian Product

(b) Construct an EER diagram and convert into Relational Model for a library Management System.

Specify 2 complex SQL queries on the above-one using Group by clause and the other using Join operation with an example. (10 Marks)

Q. 3 (a) Explain the following terms with an example . (10 Marks)

(i) Natural join (ii) Set Intersection

(iii) Weak Entity (iv) Foreign Key.

(b) Explain the Overall Architecture of DBMS in detail. (10 Marks)

Q. 4 (a) Define Deadlock. Explain how deadlock can be handled. (10 Marks)

(b) Explain Specialization and Generalization with suitable example. (10 Marks)

Q. 5 (a) For the schema mentioned below. (10 Marks)

Employee (eid, ename, address, city) Works (eid, cid, salary)

Company (cid, cname, city)

Create an ER diagram for the same and Specify the SQL queries for each of the statements given below.

1. Modify database so that John now lives in Mumbai, assuming the database entry has John staying in Delhi.
2. Find employees who live in same city as the company for which they work.

(b) Define the term Normalization as used in database design. Explain the various normal forms with an example.

Q. 6 Write short note on any two. (20 Marks)

(a) Log based recovery mechanism

(b) Triggers

(c) Conflict and View Serialiazability

(d) Data Independence

□□□

Your Success is Our Goal

Semester V - Computer Engineering

- **Computer Networks**
- **Database Management System**
- **MICROPROCESSOR**
- **Theory of Computer Science**
- **Multimedia System (Dept. Elective I)**
- **Advance Operating System (Dept. Elective I)**

Distributors

MUMBAI

Student's Agencies (I) Pvt. Ltd.
102, Konark Shram, Ground Floor, Behind Everest Building,156 Tardeo Road, Mumbai.
M :91672 90777.

Vidyarthi Sales Agencies
Shop. No. 5, Hendre Mansion, Khotachiwadi,157/159, J.S.S Road, Girgaum, Mumbai. M :98197 76110.

Bharat Sales Agency
Goregaonkar Lane, Behind Central Plaza Cinema, Charni Road, Mumbai. M :86572 92797

Ved Book Distributors - Mr. Sachin Waingade
(For Library Orders)
M : 80975 71421 / 92208 77214.
E : mumbai@techknowledgebooks.com

EMO44A Price ₹ 85/-

BOOKS ARE AVAILABLE AT ALL LEADING BOOKSELLERS !!

B-50