S.Y. Diploma : Sem. III
[CO/CM/IF/CD/CW]

Relational Database Management System

Time: 3 Hrs.] Prelim Question Paper Solution **[Marks : 100**

Q.1(a) Attempt any SIX of the following : [12]

Q.1(a) (i) List any four DBMS software. [2]

(A) List of DBMS software. [Each Software – ½ marks]

	DBMS	**Vendor**
1)	Access	Microsoft
2)	MySQL	Free
3)	DB2	IBM
4)	Oracle	Oracle

Q.1(a) (ii) List any four DDL commands. [2]

(A) DDL Commands : [Any four – ½ mark]

1) Create 2) Alter 3) Drop
4) Rename 5) Desc 6) Truncaste

Q.1(a) (iii) Explain group by, having clause of SQL with example. [2]

(A) The Group by and having clause

The group by clause arranges the data rows into a group according to the columns one specifies. [1 mark]

Example : Consider the following teachers database table.

SQL > Select * from teachers;

Name	Status	Gender	Salary
Pallavi	Visiting	F	8000
Raj	Permanent	M	15000
Ravi	Permanent	M	14000
Zarina	Visiting	F	8000

SQL > Select * from teachers **group by** status;

Name	Status	Gender	Salary
Pallavi	Visiting	F	8000
Zarina	Visiting	F	8000
Raj	Permanent	M	15000
Ravi	Permanent	M	14000

More than one column name can be used in the group by clause.

The having clause : The having clause is similar to the where clause. The having clause does for aggregate data what the where clause does for individual rows. The having clause is another search condition. In this case, the search is based on each group of a grouped table. The **difference between the where clause and having clause** is in the way the query is processed. In a where clause, the search condition on the row is performed before the rows are grouped. In the having clause the groups are formed first and then the search condition is applied to the group.

Its syntax is as follows : [1 mark]

Select select_list **from** table_list **[where** < condition >]
[**Group By** column1, col 2, … , col N] **[Having condition];**

Example : **Select * from** teacher **Group By** status, gender **Having** gender = 'F';

Output

Name	Status	Gender	Salary
Pallavi	Visiting	F	8000
Zarina	Visiting	F	8000

Q.1(a) (iv) What is index? List its types. **[2]**

(A) **INDEX:** They are optional structures associated with tables clustures that allows sqlstaments to execute more quickly against a table. Types are:

i) Unique index: Based on unique column, like national insurance number.
Syntax: Create unique index on <tablename><column name>;
Example - Create unique index idx on employee (ename);

ii) Simple index: An index created on single column of a table is called a Simple Index.
Syntax: Create index on <tablename><column name>;
Example - Create index idx on employee (ename);

iii) Composite (consented): Indexes that contain two or more columns from the same table which are useful for enforcing uniqueness in a table column where there's no single column that can uniquely identify a row.
Syntax: Create index on <tablename><column name1, Column name 2>;
Example - Create index idx on employee (ename, eno);

[Any 2 Types & Explanation – 2 marks each]

Q.1(a) (v) Define data independence. List its type. **[2]**

(A) Data Independence [Definition & type – 1 mark each]

The ability to modify a schema definition in one level without affecting a schema definition in the next higher level is called data independence. There are two levels of data independence.

1) *Physical data Independence* : It is the ability to modify the physical schema without causing application programs to be rewritten.
2) *Logical data Independence* : It is the ability to modify the logical schema without causing application programs to be rewritten. Logical data independence is more difficult to achieve than i.e. the physical data independence.

Q.1(a) (vi) Explain any two characteristics of DBMS? **[2]**

(A) Advantages of Database Management System [1 mark each]

1) *Centralized Management and Control over data* : Database administrator is the person having central control over the system.
2) *Reduction of Redundancies* : Centralized control of data by DBA avoids unnecessary duplication of data.

Q.1(a) (vii) Define DBMS. List any two applications of DBMS. **[2]**

(A) [Any 2 Application- ½ mark each, Any relevant applications can be considered]

DBMS [1 mark]

A database-management system is a collection of interrelated data and a set of programs to access those data.

Applications of DBMS:

1. Banking	2. Airlines and railways
3. Sales	4. Telecommunications
5. Universities.	6. Manufacturing
7. E-commerce	8. Credit card transactions.

Q.1(a) (viii) What is Primary Key? Give example. **[2]**

(A) [Definition Primary Key-1 mark, Example- 1 mark]

[Note: Any other example can be considered]

Primary key: Within a relation there is always one attribute which has values that are unique in a relation and thus can be used to identify tuple of that relation. Such a unique identifier is called the primary key. E.g. In the Student(Rollno,name,class,address) relation Rollno is the primary key.

Q.1(b) Attempt any TWO of the following : [8]

Q.1(b) (i) Explain any four Codd's Rules/Laws of RDBMS. [4]

(A) Codd's Rules/Laws of RDBMS [1 mark each rule]

1) **Foundation Rule :** A relational database management system must manage its stored data using only its relational capabilities.
2) **Information Rule :** All information in the database should be represented in one and only one way - as values in a table.
3) **Guaranteed Access Rule :** Each and every datum (atomic value) is guaranteed to be logically accessible by resorting to a combination of table name, primary key value and column name.
4) **Systematic Treatment of Null Values :** Null values (distinct from empty character string or a string of blank characters and distinct from zero or any other number) are supported in the fully relational DBMS for representing missing information in a systematic way, independent of data type.

Q.1(b) (ii) Explain ACID properties of Transaction. [4]

(A) ACID Properties : [Each property – 1 Mark]

- **Atomicity :** Either all operations of the transaction are reflected properly in the database, or none are.
- **Consistency:** Execution of a transaction in isolation (that is, with no other transaction executing concurrently) preserves the consistency of the database.
- **Isolation :** Even though multiple transactions may execute concurrently, the system guarantees that, for every pair of transactions T_i and T_j, it appears to T_i that either T_j finished execution before T_i started, or T_j started execution after T_i finished. Thus, each transaction is unaware of other transactions executing concurrently in the system.
- **Durability :** After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures.

Q.1(b) (iii) Write features of SQL. What is view? Give example. [4]

(A) [for features – 2 marks, View & example – 1 mark each]

SQL is a special-purpose programming language designed for managing data held in a relational database management system (RDBMS).

Originally based upon relational algebra and tuple relational calculus, SQL consists of a data definition language and a data manipulation language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. Although SQL is often described as, and to a great extent is, a declarative language (4GL), it also includes procedural elements.

Features :

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views

VIEWS IN SQL

A view in SQL terminology is a single table that is derived from other tables. The other tables could be base tables or previously defined views. A view does not necessarily exist in physical form; it is considered a virtual table in contrast to base tables whose tuples are actually stored in the database. Since the views don't exist physically there are certain limits for the possible update operations that can be applied to views but it does not provide any limitations on querying a view. Thus a view can be thought of as a way of specifying a table that need not exist physically.

We define a view in SQL by making use of the create view command. To define a view, we must give the view a name, and must state the query that computes the view. The form of the create view command is as follows.

Create view V as < query expression>
Where < query expression> is any legal query expression. The view name is represented by V. The query expression specifies the contents of a view.

A view is always updated if we modify the base tables on which the view is defined, the view automatically reflects these changes. It is the responsibility of DMBS and not the user to make sure that the view is up to date.

If we do not need a view any more, we can use the drop view command to dispose it off.
Syntax: Drop view view name;

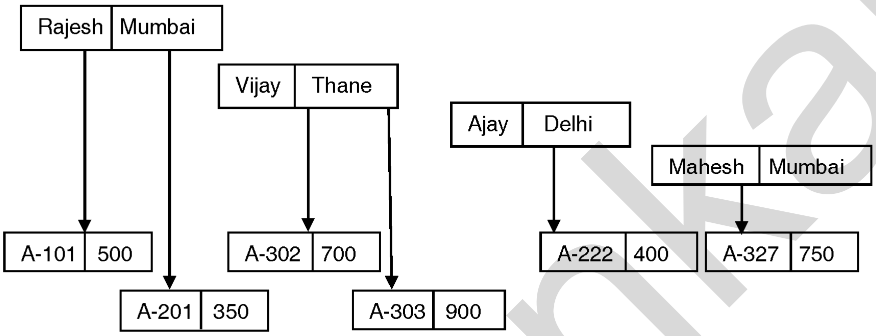
Q.2 Attempt any FOUR of the following : [16]

Q.2(a) Describe Hierarchical Data Model with suitable example. [4]

(A) Hierarchical Model [2 marks]

It is similar to Network model, in the sense that records and links represent data and relationship among data respectively. But in this model the records in databases are organised as collections of trees rather than arbitrary graphs.

[2 marks]



Q.2(b) Give syntax of UPDATE command. Demonstrate with suitable example. [4]

(A) UPDATE command [1 mark]

To modify the existing data in the database, the Update command is used. With this command, zero or more rows in a table can be updated.

The syntax of the update command is :

Syntax : Update table name **Set** column_name 1 = Expression 1,
[Column_name 2 = Expression 2], ... ,
[Column_name n = Expression n]
[Where condition]; [2 marks]

Table name is the name of the table that is to be updated. Column name is a column in the table that is to be updated. The expression is a valid SQL expression. The condition is a valid SQL condition. The update statement references a single table and assigns an expression to at least one column. The 'where' clause is optional. If an update statement doesn't contain a where clause, the assignment of a value to a column will, be applied to all rows in the table.

Example : Update branch **Set** balance = 4000; [1 mark]

The above statement will set the balance of all the branches to 4000.

Q.2(c) Consider following schema : [4]
ACCOUNT_HOLDER (Account_No, Name, Account_Type, PAN_Number, Balance).
Create a view on ACCOUNT_HOLDER having attributes (Account_No, Name, PAN_Number) where Balance is greater than 50,000.

(A) **Create view** ah_view **as** (**select** Account_No, Name, PAN_Number **from** ACCOUNT_HOLDER **where** Balance > 50000) [4 marks]

Q.2(d) What is CURSOR? Explain its types. [4]

(A) **Cursor** [2 Marks]
Oracle creates a memory area, known as context area, for processing an SQL statement, which contains all information needed for processing the statement, for example, number of rows processed, etc.

A cursor is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the cursor holds is referred to as the active set. You can name a cursor so that it could be referred to in a program to fetch and process the rows returned by the SQL statement, one at a time. There are two types of cursors :

- Implicit cursors
- Explicit cursors

- **Implicit Cursor** [1 mark]
  Implicit cursors are automatically created by Oracle whenever an SQL statement is executed, when there is no explicit cursor for the statement.

  Whenever a DML statement (INSERT, UPDATE and DELETE) is issued, an implicit cursor is associated with this statement. For INSERT operations, the cursor holds the data that needs to be inserted.
  For UPDATE and DELETE operations, the cursor identifies the rows that would be affected.

  In PL/SQL, you can refer to the most recent implicit cursor as the SQL cursor, which always has the attributes like %FOUND, %ISOPEN, %NOTFOUND, and %ROWCOUNT. The following table provides the description of the most used attributes.

Attribute	Description
%FOUND	Returns TRUE if an INSERT, UPDATE, or DELETE statement affected one or more rows or a SELECT INTO statement returned one or more rows. Otherwise, it returns FALSE.
%NOTFOUND	The logical opposite of %FOUND. It returns TRUE if an INSERT, UPDATE, or DELETE statement affected no rows, or a SELECT INTO statement returned no rows. Otherwise, it returns FALSE.
%ISOPEN	Always returns FALSE for implicit cursors, because Oracle closes the SQL cursor automatically after executing its associated SQL statement.
%ROWCOUNT	Returns the number of rows affected by an INSERT, UPDATE, or DELETE statement, or returned by a SELECT INTO statement.

- **Explicit cursor** [1 mark]
  Explicit cursors are programmer defined cursors for gaining more control over the context area. An explicit cursor should be defined in the declaration section of the PL/SQL Block. It is created on a SELECT Statement which returns more than one row.
  The syntax for creating an explicit cursor is :
  Cursor cursor_name IS select_statement;

  Working with an explicit cursor involves four steps:
  – Declaring the cursor for initializing in the memory
  – Opening the cursor for allocating memory
  – Fetching the cursor for retrieving data
  – Closing the cursor to release allocated memory

Q.2(e) List and explain four types of Database users. **[4]**

(A) List of DBMS user.

i) Naive users,	ii) Application programmers
iii) Sophisticated users	iv) Specialized users

Explanation : [1 mark each]

i) Naive User : Naïve users are unsophisticated users.
They are interact with the system through the application program.
They give data as input through application program or get output data which is generated by application program.
Example : Bank cashier.

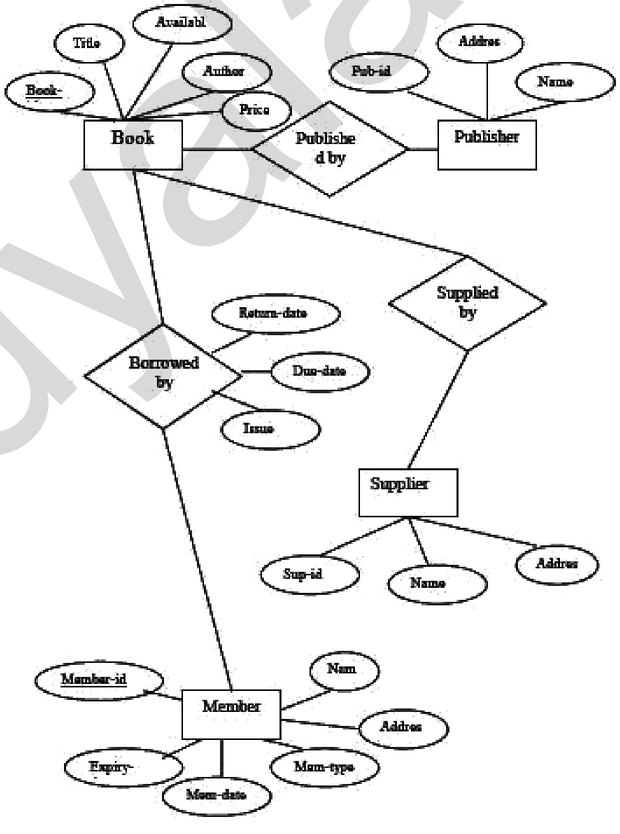
ii) **Application programmers :** Application programmers are the users who write the program.
These programmers use programming tools to develop the program.
RAD technology is used to write the program.

iii) **Sophisticated users :** Sophisticated users interact with the system by making the requests in the form of query language.
These queries are then submitted to the query processor.
Query processor converts the DML statements into lower level interactions which are understandable by storage manager.
Some sophisticated users can be analyst.

iv) **Specialized users :** These users are not traditional.
They write some special application programs which are not regular applications.
Example : such types of applications are CAD, knowledge based and expert system.

Q.2(f) Draw E-R diagram for library management system. The library should maintain the data about the staff, student and books. **[4]**

(A) E-R diagram for Library Management System [for diagram – 4 mark]



Q.3 Attempt any FOUR of the following : [16]

Q.3(a) Consider Following Schema [4]

Employee (ENO, ENAME, Department, Designation, DOJ, Salary, Dept_Location)

Solve the following query :

(i) List the employees having Designation as "Manager" and .Dept_Location as "Mumbai"

(ii) Set the salary as Rs.50,000/- having Designation as "Project Leader"

(iii) List ENO, ENAME, Salary of employees having Salary between Rs. 20,000/- to Rs.30,000/-

(iv) List Ename of employees having 2nd alphabet in the name as "A".

(A) (i) Select * from Employee where Designation = 'Manager' AND Dept_Location = 'Mumbai' [Each Query 1 Mark]

(ii) Update Employee Set Salary = 50000 where Designation = 'Project Leader'

(iii) Select ENO, ENAME, Salary from Employee where Salary >= 20000 AND Salary <= 30000

(iv) Select Ename from Employee where Ename LIKE '_A%'

Q.3(b) Give block-structure of PL/SQL and explain main components. [4]

(A) Declare [2 marks]
Declaration of memory variables
BEGIN(Mandatory)
SQL executable statements;
[Exceptions
Handling errors]
END; (Mandatory)

A block begins with a declarative section where variables are declared. This is followed by a section containing the procedural statements surrounded by BEGIN and END keywords. Each block must have Begin and End statements and may optionally include an exception section to handle errors. [2 marks]

Q.3(c) Describe in brief 2NF. [4]

(A) SECOND NORMAL FORM (2NF)

Definition : "A relation R is in second normal form (2NF) if and only if it is in INF and every non-key attribute (non prime attribute) is fully functionally dependent on the primary key." An attribute is a non-key or non-prime if it does not participate in the primary key. [1 mark]

By splitting the supplier relation of Table 1 into supplier details and product relation we can eliminate the insertion, deletion and Updation problems encountered in 1NF relation. The decomposition of the supplier relation is as shown in Table 2.

Table 1 : Supplier Relation

Supplierno	Status	City	Productno	Qty
S1	20	LATUR	P1	300
S1	20	LATUR	P2	200
S1	20	LATUR	P3	400
S1	20	LATUR	P4	200
S1	20	LATUR	P5	100
S1	20	LATUR	P6	100
S2	10	PUNE	P1	300
S2	10	PUNE	P2	400
S3	10	LATUR	P2	200
S4	20	LATUR	P2	200
S4	20	LATUR	P4	300
S4	20	LATUR	P5	400

[½ mark]

Table 2 : Supplier details Table.

Supplierno	**Status**	**City**
S1	20	LATUR
S2	10	PUNE
S3	10	PUNE
S4	20	LATUR
S5	30	AGRA

[½ mark]

Product Table

Supplierno	**Productno**	**Qty**
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

[1 mark]

Functional dependencies for the relations supplier details and products are as shown in Figure 1. Thus the problems encountered in 1NF supplier relation are solved as follows:

i) **Insertion:** Information regarding the supplier S5 can be added into the table supplier details even if he is not supplying any product.

ii) **Deleting:** The tuple (S3, P2) in the product table can be deleted without losing the information that S3 is located in Pune.

iii) **Updating:** Since redundancy is eliminated the problems during updation is less. However the relations supplier details and product cause certain problems in the 3 operations.

1) **Insertion Anomaly:** The information that a particular city has a particular status value cannot be inserted until some supplier is located in that city, because until a supplier exists there is no appropriate primary key value.

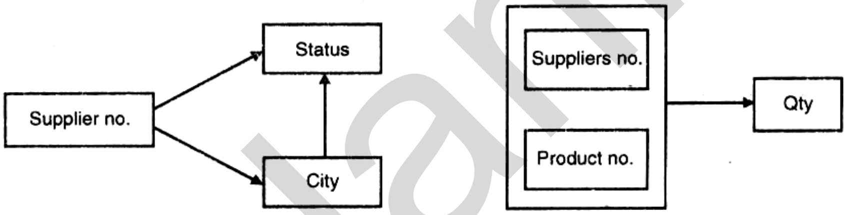


Fig. 1

2) **Deletion Anomaly:** If a tuple of the supplier details is deleted, then information regarding the supplier and the city is lost i.e. if the tuple pertaining to S5 is deleted then the status of Agra is also lost.

3) **Updation Anomaly:** Redundancy still exists in the supplier details relation because status for a city is still repeated in the supplier details relation. This leads to the same updation problem as in INF.

Q.3(d) State properties of Boycee Codd Normal form. **[4]**

(A) BOYCEE CODD NORMAL FORM

A database design is in BCNF if each, member of the set of relation schemas that constitutes the design is in BCNF. A relation schema R is in BCNF with respect to a set F of functional dependencies if for all functional dependencies in F+ of the form $\alpha \rightarrow \beta$ whereas $\alpha \subseteq R$ and $\beta \subseteq R$ at least one of the following holds : [1 mark]

i) $\alpha \rightarrow \beta$ is a trivial functional dependency (i.e. $\beta \subseteq \alpha$).

ii) α is a super key for schema R.

Consider the following relation schemas and their respective FD. [1 mark]

i) Customer_schema = (customer_name, customer_street, customer_city)
   Customer_name $\rightarrow$ customer_street, customer_city.
   This relation is in BCNF
   Customer_name $\rightarrow$ customer_street, customer_city is a non-trivial FD but the candidate key for this schema is customer_name.

ii) Branch_schema = (branch_name, assets, branch_city)
   Branch_name $\rightarrow$ assets branch_city is a non-trivial FD but branch_name is the candidate key.

iii) Loan_info schema = (branch_name, customer_name, loan_no, amount).
   Loan_no. $\rightarrow$ amount branch name.

This schema is not in BCNF, because loan_no is not a super key and loan_no amount branch name is not a trivial FD.
The loan_info schema is not in a desirable form since it suffers from repetition of information. To eliminate this redundancy the relation schema can be decomposed in those schemas, which are in BCNF. [1 mark]

Consider the decomposition of loan into schema into two schemas.
   Loan_schemas = (branch_name, loan_no., amount).
   Borrower_schema = (customer_name, loan_no.)

The decomposition is loss-less join decomposition because loan_no $\rightarrow$ amount branch name holds an loan_schema.

Loan_no is candidate key of loan_schema only trivial dependencies hold on borrower schema. These both schema of decomposition are in BCNF. [1 mark]

Q.3(e) What are different data types of SQL? Explain with example. [4]

(A) DATA TYPES IN SQL [any four, for each – 1 mark]

SQL Supports the following data types :

i) Char Data type : The char data type is used when fixed length character string is required. It can store alphanumeric values. If a user enters a value shorter than the specified length then the database blank-pads to the fixed length.

ii) Varchar2 Data type : The varchar2 data type supports a variable length character string. It also stores alphanumeric values. Using varchar2 saves disk space as compared to char. For e.g. consider a column assigned with varchar2 data type of size 30 bytes, if the user enters 10 bytes of character, then the length in that row would be 10 bytes and not 30 bytes. In the case of char, it would still occupy 30 bytes because the, remaining would be blank padded by SQL.

iii) Long Data type : This data type is used to store variable character length. The restriction on long data type is that only one column in a table Fan have long data type. This column moreover should not contain unique or primary constraint.

iv) Number Data type : The number data type can store positive numbers, negative numbers, zeros, fixed-point numbers and floating-point numbers. It can take up the following form :

(1) Column-name number (P) for integer data, where P is the number.
    For example : studentid number (5).

(2) Column-name (P, S) for floating point number where P is the total number of digits and S refers to the number of digits to the right of the decimal point.
    For example : Rate (4, 2) causes the value 44.20 to be stored exactly.

v) Date Data type : It is used to store date and time in a table. SQL database makes use of its own format to store date in a fixed length of 7 bytes each for century, month, day, year, hour, minute and second. Default date data type is "dd-mmm-yy".

Q.3(f) Explain Domain Integrity Constraint with suitable example. [4]

(A) DOMAIN INTEGRITY CONSTRAINTS [1 mark each]

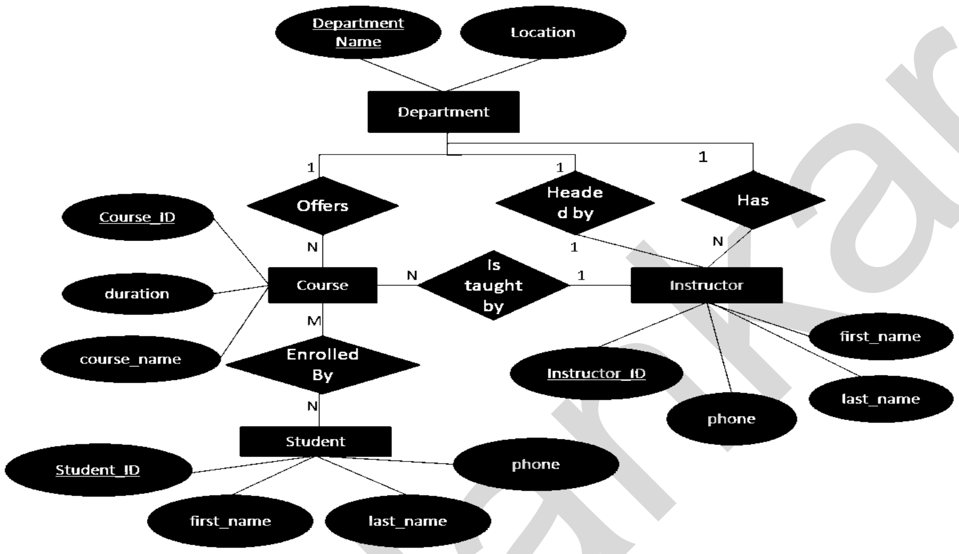
- There is a domain of possible values associated with every attribute. Domain constraints are the most elementary form for integrity constraint. It is possible for several attributes to have the same domain, while others will be distinctly different.
- The check clause in SQL-92 that will do that extension: create domain hourly-wage numeric(5,2) constraint wage- value test check(value >= 4.00)
- In this example, the hourly-wage is declared to be a decimal number with a total of five digits, two of which are placed after the decimal point and has a range of 4.00 to 999.99.
- The check clause can be used for things like value not null, or value in ("Checking", "Savings").

Q.4 Attempt any FOUR of the following : [16]

Q.4(a) Draw an E-R Diagram for College Management System. [4]

(A) E-R Diagram for College Management System

[Diagram – 4 marks, diagram should have Entities, attribute & relationship]



Q.4(b) Write PL/SQL program to display the Factorial of any number. [4]

(A) [Program – 4 marks, It should have correct syntax and logic]

```
DECLARE
   num number;
   factorial number;

  FUNCTION fact(x number)
  RETURN number
  IS
     f number;
  BEGIN
     IF x=0 THEN
        f := 1;
     ELSE
        f := x * fact(x-1);
     END IF;
  RETURN f;
  END;

  BEGIN
     num:= 6;
     factorial := fact(num);
   dbms_output.put_line(' Factorial '|| num || ' is ' || factorial);
END;
/
```

When the above code is executed at SQL prompt, it produces the following result:

```
Factorial 6 is 720
PL/SQL procedure successfully completed.
```

Q.4(c) Explain ALTER command. Demonstrate with any two options. [4]

(A) SQL ALTER TABLE Syntax [For alter add syntax and example – 2 marks]

To add a column in a table, use the following syntax:

```
ALTER TABLE table_name
ADD column_name datatype
```

To delete a column in a table, use the following syntax (notice that some database systems don't allow deleting a column):

```
ALTER TABLE table_name
DROP COLUMN column_name
```

Example :

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

Add Column Example :

```
ALTER TABLE Persons
ADD DateOfBirth date
```

P_Id	LastName	FirstName	Address	City	DateOfBirth
1	Hansen	Ola	Timoteivn 10	Sandnes	
2	Svendson	Tove	Borgvn 23	Sandnes	
3	Pettersen	Kari	Storgt 20	Stavanger	

DROP COLUMN Example [For alter drop syntax and example – 2 marks]

```
ALTER TABLE Persons
DROP COLUMN DateOfBirth
```

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

Q.4(d) What are sequences? Why it is used? Create sequence for "STUDENT" table. **[4]**

(A) In Oracle, you can create an autonumber field by using sequences. A sequence is an object in Oracle that is used to generate a number sequence. This can be useful when you need to create a unique number to act as a primary key. [1 mark]

The syntax for a sequence is: [2 marks]

```
CREATE SEQUENCE sequence_name
  MINVALUE value
  MAXVALUE value
  START WITH value
  INCREMENT BY value
  CACHE value;
```

For example: [1 mark]

```
CREATE SEQUENCE studentID_seq
  MINVALUE 1
  MAXVALUE 999999999999999999
  START WITH 1
  INCREMENT BY 1
  CACHE 20;
```

Q.4(e) What is data warehousing and data mining? **[4]**

(A) **DATA MINING** [2 marks]

Generally, data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases.

The **Knowledge Discovery in Databases (KDD)** process is commonly defined with the stages:

(i) Selection
(ii) Pre-processing
(iii) Transformation
(iv) Data Mining
(v) Interpretation/Evaluation.

DATA WAREHOUSING [2 marks]

In computing, a **data warehouse** or **enterprise data warehouse** (**DW**, **DWH**, or **EDW**) is a database used for reporting and data_analysis. It is a central repository of data which is created by integrating data from one or more disparate sources. Data warehouses store current as well as historical data and are used for creating trending reports for senior management reporting such as annual and quarterly comparisons.

Q.4(f) Describe the use of Grant and REVOKE privilege with the help of example. **[4]**

(A) GRANTING PERMISSIONS USING THE GRANT STATEMENT

The grant statement provides various types of access to database objects such as tables, views and sequences. [1 mark]

Syntax: **Grant** {object privileges}
ON object name
To < user list >
[With Grant Option]

Explanation of Syntax :

Object Privileges: Each object privilege that is granted-authorizes the grantee to perform some operation on the object. The user can grant all the privileges or grant only specific object privileges. [1 mark]

The list of object privileges is as follows: [1 mark]

(1) Alter : It allows the grantee to change the table definitions with the Alter Table command.
(2) Delete : It allows the grantee to remove the records from the table with the delete command.
(3) Index : It allows the grantee to create an index on the table with the create index command.
(4) Insert : It allows the grantee to add records to the table with the Insert command.
(5) Select : It allows the grantee to query the table with the select command.
(6) Update : It allows the grantee to modify the records in the table with the update command.

The Object name refers to a relation name or a view name.
The **with grant option** allows the grantee to grant object privileges to other users.

Examples : [1 mark]

1) To grant a privileges on the table product to the user Ajay:

   Grant ALL
   On Product
   To Ajay;

2) To grant select and update privileges on the table product to Vijay.

   Grant select, update
   On product
   To Vijay;

Q.5 Attempt any FOUR of the following : [16]

Q.5(a) What are snapshots? Create Snapshot for "Vehicle" table. [4]

(A) [Definition – 1 mark, Example – 3 marks]

```
USE master
CREATE DATABASE TEST
USE TEST
CREATE TABLE Vehicle(ID INT, Model VARCHAR(15),Price INT)
INSERT INTO Vehicle VALUES
(1,'Maruti',10000),
(2,'Fiat',20000),
(3,'Innova',30000)

USE master
CREATE DATABASE TEST_SS
ON
(NAME=TEST,
FILENAME='C:\Program Files\Microsoft SQL Server\
        MSSQL10_50.MSSQLSERVER\MSSQL\DATA\TEST_SS.ss'
)
AS SNAPSHOT OF TEST
```

Q.5(b) Explain two Locking Strategies. [4]

(A) Pessimistic Locking [Each Strategy – 2 marks]

Pessimistic locking is an approach where an entity is locked in the database for the entire time that it is in application memory (often in the form of an object). A lock either limits or prevents other users from working with the entity in the database. A write lock indicates that the holder of the lock intends to update the entity and disallows anyone from reading, updating, or deleting the entity. A read lock indicates that the holder of the lock does not want the entity to change while the hold the lock, allowing others to read the entity but not update or delete it. The scope of a lock might be the entire database, a table, a collection of rows, or a single row. These types of locks are called database locks, table locks, page locks, and row locks respectively.

The advantages of pessimistic locking are that it is easy to implement and guarantees that your changes to the database are made consistently and safely. The primary disadvantage is that this approach isn't scalable. When a system has many users, or when the transactions involve a greater number of entities, or when transactions are long lived, then the chance of having to wait for a lock to be released increases. Therefore this limits the practical number of simultaneous users that your system can support.

Optimistic Locking
With multi-user systems it is quite common to be in a situation where collisions are infrequent. Although the two of us are working with Customer objects, you're working with the Wayne Miller object while I work with the John Berg object and therefore we won't collide. When this is the case optimistic locking becomes a viable concurrency control strategy. The idea is that you accept the fact that collisions occur infrequently, and instead of trying to prevent them you simply choose to detect them and then resolve the collision when it does occur.

Overly Optimistic Locking
With the strategy you neither try to avoid nor detect collisions, assuming that they will never occur. This strategy is appropriate for single user systems, systems where the system of record is guaranteed to be accessed by only one user or system process at a time, or read-only tables. These situations do occur. It is important to recognize that this strategy is completely inappropriate for multi-user systems.

Q.5(c) Explain four different types of SET operators. **[4]**

(A) Set of operators in SQL with example [Each operator – 1 mark]

Set operators combine the results of two component queries into a single result. Queries containing set operators are called as compound queries. Set operators in SQL are represented with following special keywords as: Union, Union all, intersection & minus.

Consider data from two tables emp1 and emp2 as

emp1	emp2
ename	ename
abc	pqr
xyz	xyz
lmn	

i) **Union :** The Union of 2 or more sets contains all elements, which are present in either or both. Union works as or.
Example : select ename from emp1 union select ename from emp2;
The output considering above data is :

```
Ename
abc
xyz
lmn
pqr
```

ii) **Union all :** The Union of 2 or more sets contains all elements, which are present in both, including duplicates.
Example : select ename from emp1 union all select ename from emp2;
The output considering above data is :

```
Ename
abc
xyz
lmn
pqr
xyz
```

iii) **Intersection :** The intersection of 2 sets includes elements which are present in both.
Example : select ename from emp1 intersection select ename from emp2;
The output considering above data is:

```
Ename
Xyz
```

iv) **Minus :** The minus of 2 sets includes elements from set1 minus elements of set2.
Example : select ename from emp1 minus select ename from emp2;
The output considering above data is:

```
Ename
abc
lmn
```

Q.5(d) What is client server architecture? Draw diagram. [4]

(A) Client Server Architecture of the database [2 marks]

The architecture of a database system is generally influenced by the underlying computer system on which the database system runs. Networking of computers allow some tasks to be executed on the server system, and some tasks to be executed on client systems. This division of work has led to the development of client-server database systems. The general structure of a client –server system is shown in the figure drawn below :

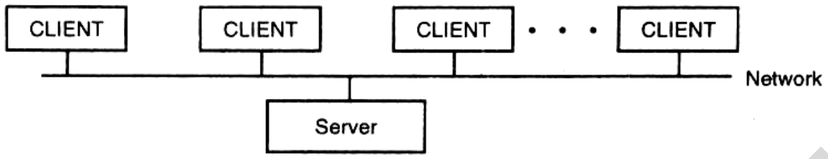
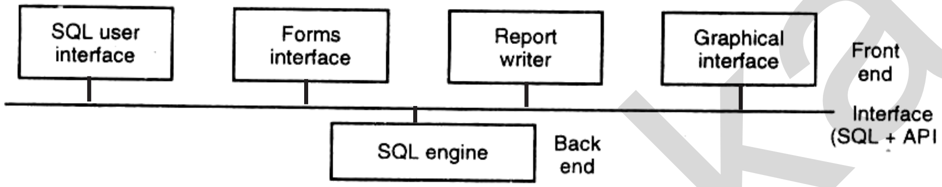


Fig. : General Structure of Client-Server System.



[2 marks]

Front-end and Back end Functionality

Database functionality can be broadly divided into two parts – the front end and the back end as shown in the above figure. The back end manages access structure, query evaluation and optimisation, concurrency control, and recovery. The front end of a database system consists of tools such as forms, report writers, and graphical user interface facilities. The interface between front-end and back-end is through SQL, or through an application program.

Q.5(e) What is data abstraction? What are the levels of abstraction? [4]

(A) [Data abstraction– 1 mark, Three levels – 3 marks]

Data Abstraction:

The need for efficiency has led designers to use complex data structures to represent data in the database. Since many database-systems users are not computer trained, developers hide the complexity from users through several levels of abstraction, to simplify users interactions with the system:

1. **Physical level**. The lowest level of abstraction describes how the data are actually stored. The physical level describes complex low-level data structures in detail.
2. **Logical level**. The next-higher level of abstraction describes what data are stored in the database, and what relationships exist among those data. The logical level thus describes the entire database in terms of a small number of relatively simple structures.
3. **View level**. The highest level of abstraction describes only part of the entire database.

   Many users of the database system do not need all this information; instead, they need to access only a part of the database. The view level of abstraction provides many views for the same database

Q.5(f) Explain following terms with suitable example (i) Entity (ii) Entity set. [4]

(A) **(i) Entity :** An entity is a thing or object in the real world that is distinguishable from all other objects. For example, Student is an entity, Account is an entity; Each Person in an enterprise is an entity. [2 marks]

(ii) Entity set : An entity set is a set of entities of the same type that share the same properties or attributes. For example, the customers at a given bank can be defined as the entity set Customer. [2 marks]

Q.6 Attempt any FOUR of the following : [16]

Q.6(a) List and explain any four advantages of DBMS over File processing system. [4]

(A) **Advantages of DBMS over File processing system** [For each – 1 mark]

1) Data Redundancy and Inconsistency

Since the data files and application programs are created by different programmers over a long period.

(i) The data files are likely to have different formats.
(ii) Programs may be written in several programming languages.
(iii) The same information may be duplicated in several plates. This results in data redundancy and inconsistency.

Consider following two data files :

(i) Savings account data file - stores information about customer (account-no, name, social-security, address, telephone-no.)
(ii) Checking account data file- stores information about customer (account-no, name, social-security, address, telephone-no.)

Fields (name, social-security, address and telephone-no.) are same in both the files, i.e. duplication of data is there which results in data redundancy. Data redundancy increases the cost of storing and retrieving the data.
If the values of these common fields are not matching for some records in both files, then it results in inconsistency of data.

2) Difficulty in accessing the data

Conventional file processing system does not allow needed data to be retrieved in a convenient and efficient manner. e.g. consider a data file Savings account data file with fields (acc-no, name, social-security, address, and balance) Application programs to access the data are written. But if user wants to display only those records for which balance is greater than Rs. 10,000. And if that program is not written, then it is difficult to access that data.

3) **Data Isolation**
Because data are scattered in various files and file may be in different formats, it is difficult to write new application programs to retrieve the appropriate data.

4) **Integrity Problems**
The data values stored in the database must satisfy certain types of consistency constraints. For example the balance of a bank account may never fall below a prescribed amount (say Rs.500). Developers enforce these constraints in the system by adding appropriate code in the various application programs. However, when a new constraint is to be added, it is difficult to change the programs to enforce the new constraint.

Q.6(b) Explain any four aggregate functions with example. [4]
(A) [Any four function – 1 mark, Meaning with syntax – ½ mark each, Example – ½ mark each]

Aggregate Functions : Avg, sum, min, max, count, variance and stddev.

i) **Avg(column_name)** : It is used to calculate average of values in column.
OR
Returns average value of column.
Example: select avg(salary) from Emp;

ii) **Sum(column)** : Used to calculate sum/addition of values of column.
OR
Returns addition of column of number data type.
Example : select sum(salary) from Emp;

iii) **Min(column)** : Used to find lowest value in column.
OR
Return lowest value from column.
Example : select min(salary) from Emp;

iv) **Max(column)** : Used to find highest value in column.
OR
Returns heighest value in column.
Example: select max(salary) from Emp;

v) **Count(column_name)** : Count the number of values in a column.
Example: select count(name) from Emp;

vi) **Count(*)** : count the total number of rows.
Example: select count(*) from Emp;

vii) Count(Distinct column_name) : count the total number of values without duplication.
Example: select count(distinct name) from Emp;

Q.6(c) Describe following keys : [4]
(i) Candidate Key (ii) Primary Key

(A) i) Candidate Key [2 marks]
Sometimes in a relation there are more than one attribute are having the unique identification property. Those attributes are known as candidate keys.
Example - Consider Student (RNO, NAME, PER, BRANCH), if RNO and NAME are unique then both are known as Candidate keys.

ii) Primary key [2 marks]
Within a relation there is always one attribute which has values that are unique in a relation and thus can be used to identify tuple of that relation. Such a unique identifier is called the primary key.
Example - In the Student relation RNO is the primary key.

Q.6(d) Describe Grant and Revoke commands. [4]

(A) GRANTING PERMISSIONS USING THE GRANT STATEMENT
The grant statement provides various types of access to database objects such as tables, views and sequences.

Syntax: **Grant** {object privileges} [1 mark]
ON object name
To < user list >
[With Grant Option]

The list of object privileges is as follows:
1) **Alter:** It allows the grantee to change the table definitions with the Alter Table command.
2) **Delete:** It allows the grantee to remove the records from the table with the delete command.
3) **Index:** It allows the grantee to create an index on the table with the create index command.
4) **Insert:** It allows the grantee to add records to the table with the Insert command.
5) **Select:** It allows the grantee to query the table with the select command.
6) **Update:** It allows the grantee to modify the records in the table with the update command.

Examples : [1 mark]

1) To grant a privileges on the table product to the user Ajay:

   Grant ALL
   On Product
   To Ajay;

2) To grant select and update privileges on the table product to Vijay.

   Grant select, update
   On product
   To Vijay;

REVOKING PERMISSIONS USING THE REVOKE STATEMENT

To revoke an authorization we use the revoke statement. It takes the following form:

Syntax: Revoke < object privilege list > **on** < Object name > [1 Mark]
from < user list> **[Restrict / Cascade];**

One cannot use the Revoke command to perform the following operation:

(1) Revoke the object privileges that one didn't grant to the revoke.
(2) Revoke the object privileges granted through the operating system.

E.g.: Revoke select **on** product **From** Ajay **cascade;** [1 mark]

The revoke statement may also specify restrict:

E.g.: Revoke select **on** branch **from** Ajay, Vijay **restrict;**

Q.6(e) Explain WHILE….LOOP in PL/SQL with example. **[4]**

(A) While Loop [2 Marks]

A WHILE LOOP is used when a set of statements has to be executed as long as a condition is true. The condition is evaluated at the beginning of each iteration. The iteration continues until the condition becomes false.

The General Syntax to write a WHILE LOOP is: [1 mark]

```
WHILE <condition>
LOOP statements;
END LOOP;
```

Important steps to follow when executing a while loop: [1 mark]

1) Initialise a variable before the loop body.
2) Increment the variable in the loop.
3) EXIT WHEN statement and EXIT statements can be used in while loops but it's not done oftenly.

Q.6(f) Explain join concept of SQL. State types of join. [4]

(A) JOINS [for concept & types – 2 marks each]

Joining of multiple tables (Equi-joins): Sometimes we require to treat more than one table as though it were a single entity. Then a single SQL sentence can manipulate data from all the tables as though the tables were not separate objects, but one single entity. To achieve this, we have to join tables. Tables are joined on columns that have the same data type and data width in the tables.

A join, which is based on equalities, is called equi-join. In equi-join the comparison operator equal to (=) is used to perform a join. It retrieves rows from tables having a common column. It is also called as simple join.

Syntax :
```
select table1.column, table 1. Column,
                        table2.column, table2.column, .......
from table1, table2
where tablel.column1 = table2.column2;
```

Example : Display order information like s_order_no, client name, s_order date and client_no for all the orders placed by the client. Here the data required is in two tables i.e. the sales order table and the client_table. These tables have to be accessed as though they were one entity.

For example : Sales order table.

Sales_order table

Sorder_no	Client_no	Sorder_date
101	1	12-oct-95
102	2	10-oct-95
103	3	11-oct-95
104	4	12-oct-95

Client table

Client_no	Client_name
1	Ajay
2	Vijay
3	Raj
4	Ravi

We will require using the following SQL statement to retrieve the required information.

```
SQL>Select sorder_no, sorderdate, name,
          client.client_no from sales_order, client
where client.client_no = sales_order.client_no;
```

In case of Joins we use the technique of specifying the table name prior the column name, separated by a period in the where condition because the column name in both the tables are identical.
The output will be :

Output

Sorder_no	Sorder_date	Name	Client.client_no
101	12-Oct.-95	Ajay	1
101	10-Oct.-98	Vijay	2
102	1l-Oct.-99	Raj	3
103	12-Oct.-99	Ravi	4

Types of Join

- Inner Join
- Outer Join
- Self-Join
- Non equi-join

❑ ❑ ❑ ❑ ❑