MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

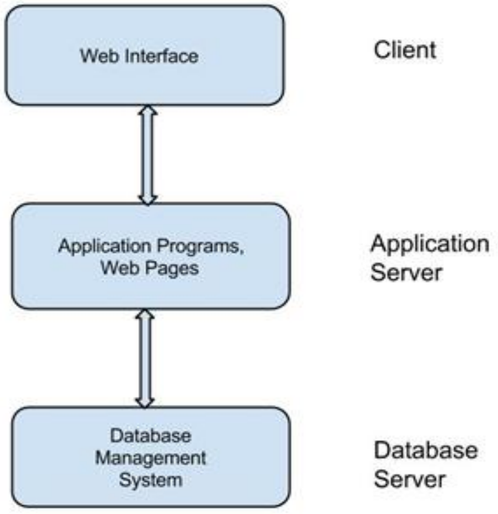
SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Relational Database Management System **Subject Code:** **17332**

Important Instructions to examiners:

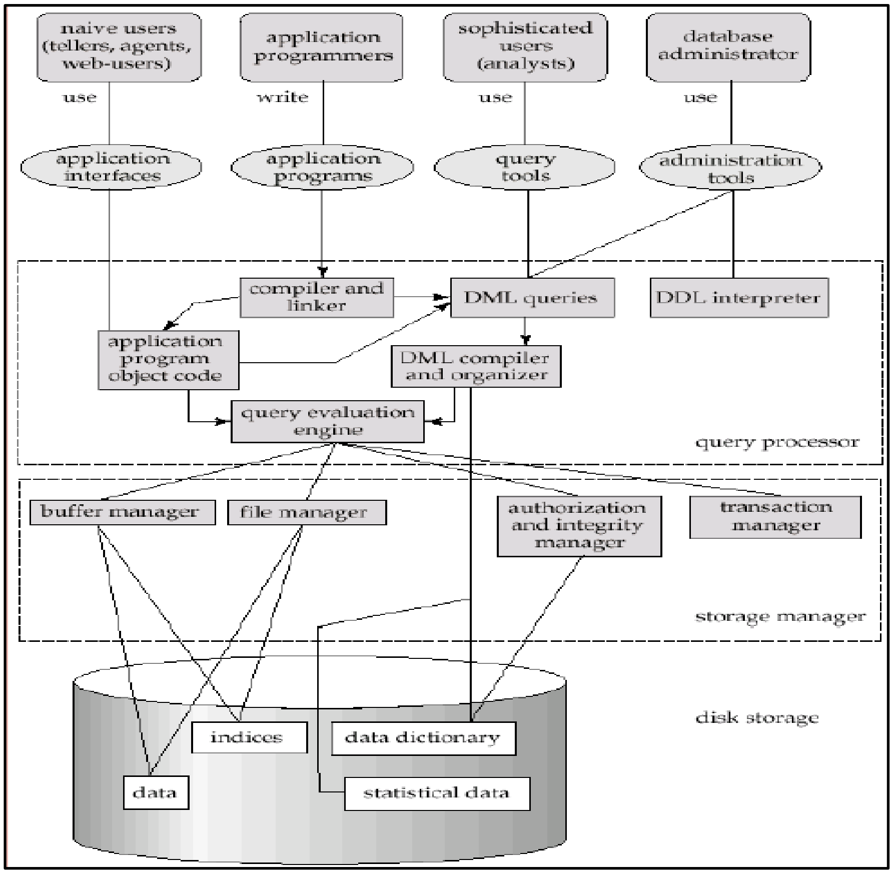
1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q.N.	Answer	Marking Scheme
1.	**(A)**	**Attempt any SIX of the following:**	**12**
	(a)	**List types of files in the computer system with example.**	**2M**
	Ans.	**The list of types of files in the computer system with example**	
		• **.doc** and **.docx** - Microsoft Word file • **.pdf** - PDF file • **.rtf** - Rich Text Format • **.txt** - Plain text file • **mp3** - MP3 audio file • **.pkg** - Package file • **.rar** - RAR file • **.zip** - Zip compressed file • **.csv** - Comma separated value file • **.dat** - Data file • **.db** or **.dbf** - Database file • **.log** - Log file • **.mdb** - Microsoft Access database file • **.c** - C and C++ source code file • **.class** - Java class file	***Any two type 1M each***

		• **.cpp** - C++ source code file • **.cs** - Visual C# source code file • **.java** - Java Source code file • **.sh** - Bash shell script • **.vb** - Visual Basic file	
	(b) **Ans.**	**Draw the neat labelled diagram of three-tier client/server architecture.**  Web Interface Client Application Programs, Web Pages Application Server Database Management System Database Server	**2M** **Correct** ***Diagram 2M***
	(c) **Ans.**	**List the functions of database admnistrator (four points).** The Functions of Database Administrator are the following: • Responsible for authorizing access to the database, • For coordinating and monitoring its use, • Acquiring software and hardware resources, • Controlling its use and monitoring efficiency of operations.	**2M** ***Any four points 1M each***
	(d) **Ans.**	**List any four DBMS software.** **List of DBMS software are the following:** i. Oracle RDBMS ii. IBM DB2 iii. Microsoft SQL Server iv. MySQL v. MS Access vi. SQLite	**2M** ***Any four $^{1/2}$M each***

		vii. PostgreSQL viii. MongoDB ix. SQL Developer x. SAP Sybase SE	
	(e) **Ans.**	**Enlist different types of data model.** Different types of Data model are the following: • Hierarchical database model. • Relational model. • Network model. • Object-oriented database model. • Entity-relationship model. • Enhanced Entity relationship model.	**2M** ***Any four $^{1/2}$M each***
	(f) **Ans.**	**Enlist properties of relational table.** The Properties of Relational model are the following: • In relational data model, relations are saved in the format of Tables. This format stores the relation among entities. A table has rows and columns, where rows represents records and columns represent the attributes. • **Tuple** − A single row of a table, which contains a single record for that relation is called a tuple. • **Relation key** − Each row has one or more attributes, known as relation key, which can identify the row in the relation (table) uniquely. • **Attribute domain** − Every attribute has some pre-defined value scope, known as attribute domain. • Every relation has some conditions that must hold for it to be a valid relation. These conditions are called **Relational Integrity Constraints**. There are three main integrity constraints −Key constraints, Domain constraints, Referential integrity constraints.	**2M** ***Any two properties 1M each***
	(g) **Ans.**	**State the characteristics of SQL (six points).** Characteristics of SQL are the following: • Allows users to access data in the relational database management systems. • Allows users to describe the data. • Allows users to define the data in a database and manipulate that data. • Allows to embed within other languages using SQL modules, libraries & pre-compilers.	**2M** ***Any six points 2M***

		• Allows users to create and drop databases and tables. • Allows users to create view, stored procedure, functions in a database. • Allows users to set permissions on tables, procedures and views.	
	(h) **Ans.**	**Enlist four properties of transaction.** The properties of Transaction are the following: 1. Atomicity: (all or nothing) 2. Consistency: (No violation of integrity constraints) 3. Isolation: (concurrent changes invisibles) 4. Durability: (committed update persist)	**2M** ***Any four*** $^{1/2}$***M each***
	(i) **Ans.**	**List four advantages of using snapshots.** List of four advantages of snapshot are the followings: • The user can create as many snapshots as he/she wants quickly in no amount of time. The user can schedule to take snapshots every hour. • Since the snapshot occupies less disk space, and are faster to create than a backup, they are a better option than backups in these cases. • The snapshots can be used in restore operations. • The corrupted or deleted data can be recovered from the snapshot to repair the primary database. • In case of user error, the administrator can revert back to the snapshot taken just before the error.	**2M** ***Any four advantages*** $^{1/2}$***M each***
1.	**(B)** **(a)** **Ans.**	**Attempt any TWO of the following:** **Explain DBMS architecture with neat labelled diagram.**	**8** **4M**



Correct diagram 3M

Explanation:

Query processor:

1. **DDL Interpreter:** It interprets DDL statements and records them in a set of tables containing metadata or data dictionary.
2. **DML Compiler**: It translates DML statements of high level query language into low level instructions that query evaluation engine understands.
3. **Compiler and linker** or **Embedded DML PreCompiler**: It converts DML statements embedded in application program to normal procedural calls in host language.
4. **Query Evaluation Engine**: It executes low level instructions generated by DML compiler and DDL interpreter.

Explanation 1M

		Storage Manager: **1. Authorization and integrity manager** Which tests for the satisfaction of integrity constraints and checks the authority of users to access data. **2. Transaction manager** Which ensures that the database remains in a consistent correct) state despite system failures, and that concurrent transaction executions proceed without conflicting. **3. File Manager** It manages the allocation of space on disk storage and the data structures used to represent information stored on disk. **4. Buffer manager** Which is responsible for fetching data from disk storage into main memory, and deciding what data to cache in main memory. **Disk Storage:** i. **Data files,** which store the database itself. ii. **Data dictionary,** which stores metadata about the structure of the database, in particular the schema of the database. iii. **Indices,** which provide fast access to data items that hold particular values. iv. **Statistical Data:** It stores statistical information about the data in the database.	
	(b) **Ans.**	**List types of join. Explain it with example in detail.** List of different types of joins are the following: 1. Equijoin 2. Inner Join 3. Outer Join • Left outer join • Right outer join • Full outer Join 4. Natural Join 5. Cross Join. 6. Self join The above joins are described as follows with example: 1. Equijoin -SQL EQUI JOIN performs a JOIN against equality or matching column(s) values of the associated tables. An equal sign (=) is used as comparison operator in the where clause to refer equality.	**4M** ***Types 1M***

Example-
Select * from Employee,Department where
Employee.Deptid=Department.Deptid;

Explanation of any 3 types 3M

2. Inner Join- The INNER JOIN selects all rows from both participating tables as long as there is a match between the columns. An SQL INNER JOIN is same as JOIN clause, combining rows from two or more tables.
Example-Select * from Employee Inner Join Department on Employee.Deptid=Department.Deptid;

3. Outer Join- The SQL OUTER JOIN returns all rows from both the participating tables which satisfy the join condition along with rows which do not satisfy the join condition.

- LEFT (OUTER) JOIN: Return all records from the left table, and the matched records from the right table
  Example- Select * from Employee left outer Join Department on
  Employee.Deptid=Department.Deptid
- RIGHT (OUTER) JOIN: Return all records from the right table, and the matched records from the left table
  Example- Select * from Employee right outer Join Department on
  Employee.Deptid=Department.Deptid;
- FULL (OUTER) JOIN: Return all records when there is a match in either left or right table
  Example- Select * from Employee full outer Join Department on
  Employee.Deptid=Department.Deptid;

4. Natural Join- The SQL NATURAL JOIN is a type of EQUI JOIN and is structured in such a way that, columns with the same name of associated tables will appear once only.
Example- Select * from Employee natural join Department;

5. Cross Join- The SQL CROSS JOIN produces a result set which is the number of rows in the first table multiplied by the number of rows in the second table if no WHERE clause is used along with CROSS JOIN. This kind of result is called as Cartesian Product.
Example- Select * from Employee cross join Department;

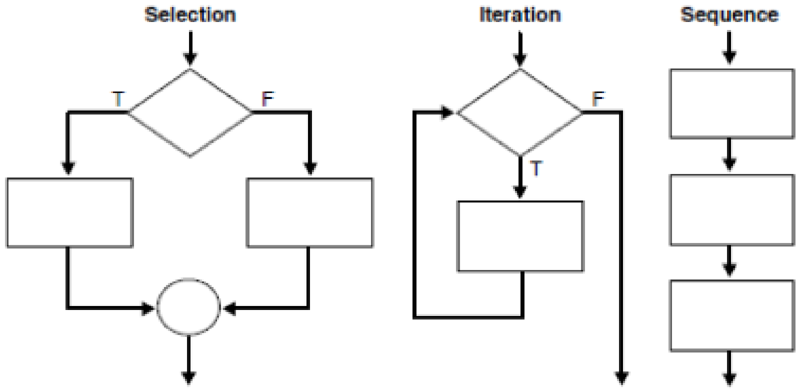
6. Self Join- A self join is a join in which a table is joined with itself

(which is also called Unary relationships), especially when the table has a FOREIGN KEY which references its own PRIMARY KEY. To join a table itself means that each row of the table is combined with itself and with every other row of the table.
Example- Select * from Employee self join Department;

(c) Ans.

Explain control structure in PL/SQL in brief. **4M**

The basic control structures in PL/SQL are the Following:



Explanation 4M

- The selection structure tests a condition, then executes one sequence of statements instead of another, depending on whether the condition is true or false. A condition is any variable or expression that returns a BOOLEAN value (TRUE or FALSE).
- The iteration structure executes a sequence of statements repeatedly as long as a condition holds true.
- The sequence structure simply executes a sequence of statements in the order in which they occur.

Testing Conditions: IF and CASE Statements

- The IF statement executes a sequence of statements depending on the value of a condition. There are three forms of IF statements: IF-THEN, IF-THEN-ELSE, and IF-THEN-ELSIF.
- The CASE statement is a compact way to evaluate a single condition and choose between many alternative actions.

Controlling Loop Iterations: LOOP and EXIT Statements

LOOP statements execute a sequence of statements multiple times.

		There are three forms of LOOP statements: LOOP, WHILE-LOOP, and FOR-LOOP. **Sequential Control: GOTO and NULL Statements** The GOTO statement is seldom needed. Occasionally, it can simplify logic enough to warrant its use. The NULL statement can improve readability by making the meaning and action of conditional statements clear	
2.	 **(a)** **Ans.**	**Attempt any FOUR of the following:** **List the characteristics of Database Administrator (4 points). Explain** **(i) Schema and physical organization modifications** **(ii) Granting of authorization for data access** The characteristics of a Database Administrator are the followings: 1. A good knowledge of the operating system(s) 2. A good knowledge of physical database design 3. Ability to perform both Oracle and also operating system performance monitoring and the necessary adjustments. 4. Be able to provide a strategic database direction for the organization. 5. Excellent knowledge of Oracle backup and recovery scenarios. 6. Good skills in all Oracle tools. 7. A good knowledge of Oracle security management. 8. A good knowledge of how Oracle acquires and manages resources. 9. A DBA should have sound communication skills with management, development teams, vendors, systems administrators and other related service providers. **Explanation:** **(i) Schema and physical organization modifications:** Schema modification refers to change in the structure of a database after its creation is done. It consists of adding or removing columns (attributes) to the table, adding constraints to the existing structure or removing any structure from the table. While physical organisation modification includes insertion, deletion or updating of records in the existing table or database.	**16** **4M** ***Any four points*** $^{1/2}$***M each*** ***Explanation 1M each***

(ii) Granting of authorization for data access:
Database Administrator has the authority to grant privileges or permissions to other users, public or a specific role to either execute a specific task within the database or to have access into the database to carry out some particular query.

(b) Explain following advantages of Relational Model: **4M**

(i) Data independence
(ii) Simplicity
(iii) Group data manipulation
(iv) Less overhead management

Ans.

(i) Data independence: The structure of a database can be changed without having to change any application.

Each point 1M

(ii) Simplicity: A relational data model is simpler than the hierarchical and network model.

(iii) Group data manipulation: A relational model is defined as a database that allows to group its data items into one or more independent tables that can be related to one another by using fields common to each related table.

(iv) Less overhead management: Since relational model is in the form of tables, less overhead is required for managing it.

(c) List different datatypes of SQL. Explain any two of them. **4M**

Ans.
Different types of datatypes of SQL are the following:

1. Numeric data types such as int, float, real etc.
2. Date and Time data types such as Date, Time, Datetime etc.
3. Character and String data types such as char, varchar, text etc.
4. Unicode character string data types, for example nchar, nvarchar, ntext etc.
5. Binary data types such as binary, varbinary etc.

List 2M

Numeric Types:

Data Type	**Description**
INTEGER	Represents an integer. The minimum and maximum values depend on the DBMS.

SMALLINT	Same as INTEGER type except that it might hold a smaller range of values, depending on the DBMS.
BIGINT	Same as INTEGER type except that it might hold a larger range of values, depending on the DBMS.
DECIMAL(p, s)	Exact numerical, precision p, scale s. A decimal number, that is a number that can have a decimal point in it. The size argument has two parts: precision and scale. The scale cannot exceed the precision. Precision comes first, and a comma must separate from the scale argument.
NUMERIC(p, s)	Exact numerical, precision p, scale s. The maximum precision depends on the DBMS.
FLOAT(p)	Approximate numerical, mantissa precision p. Precision is greater than or equal to 1 and the maximum precision depends on the DBMS.

Description of any 2 types 1M each

Character String Types:

Data Type	Description
CHARACTER	Character string, fixed length. A string of text in an implementer-defined format. The size argument is a single nonnegative integer that refers to the maximum length of the string. Values for this type must enclose in single quotes.
CHARACTER VARYING (VARCHAR)	Variable length character string, maximum length fixed.

Binary large object type:

A binary string is a sequence of octets that does not have either a character set or collation associated with it and is described by a binary data type descriptor.

Data Type	Description
BINARY LARGE OBJECT (BLOB).	BLOB stores a long sequence of bytes.

Datetime Types:

Data Type	Description
DATE	Represents a date. Format : yyyy-mm-dd
TIME WITHOUT TIME ZONE	Represents a time of day without time zone. Format : hh:mm:ss

(d) **Give difference between where clasuse and having clause (four points).** **4M**

Ans.

Sr. No.	Where clause	Having clause
1	The where clause satisfy the criteria which individual records must meet to be selected by query. It can be used without the Group by clause	The having clause cannot be used without Group by clause.
2	The Where clause select rows before grouping	The Having clause select rows after grouping
3	The Where clause cannot contain aggregate function.	The Having clause can contain aggregate function.
4	The Where clause is used to impose condition on select statement on single	The Having clause is used to impose on Group by function and is used after Group by

Any four points 1M each

	row function and is used before group by clause	clause in the query.
5	Example- Select ename, sal from emp where deptid=20;	Example- Select ename, Avg(sal) from emp group by deptid having deptid>20;

(e) Explain the following sequences in SQL with syntax and example: **4M**

(i) Creating sequneces

(ii) Dropping sequences

Ans.

(i) Creating sequneces:

```
CREATE SEQUENCE sequence_name
START WITH initial_value
INCREMENT BY increment_value
MINVALUE minimum value
MAXVALUE maximum value
CYCLE|NOCYCLE ;
```

Example-

```
CREATE SEQUENCE sequence_1
start with 1
increment by 1
minvalue 0
maxvalue 100
cycle;
```

Each syntax 1M

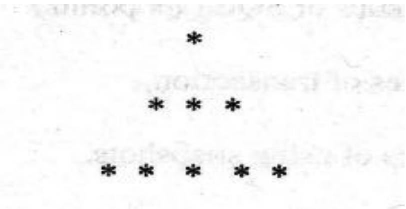
Each example 1M

(ii) Dropping sequences:

DROP sequence Sequence_name;

Example- DROP sequence Sequence_1;

(f) Write a program in PL/SQL to give following output: **4M**



```
    *
  * * *
* * * * *
```

Ans.

```
set serveroutput on;
declare
        n number:=5;
        i number:=1;
        j number;
        k number;
begin
        while i<n
        loop
                j:=1;
                while j<n-i
                loop
                        dbms_output.put('   ');
                        j:=j+2;
                end loop;
                for k in 1..i
                loop
                        dbms_output.put('*');
                end loop;
                        dbms_output.new_line;
                i:=i+2;
        end loop;
end;
/
```

Correct logic 2M

Correct syntax 2M

3. **Attempt any FOUR of the following:** **16**

(a) **List and explain two types of distributed database.** **4M**

Ans.

Distributed databases can be broadly classified into
1) homogeneous distributed database
2) heterogeneous distributed database

List 2M

Homogeneous Distributed Databases:

In a homogeneous distributed database, all the sites use identical DBMS and operating systems. Its properties are −

- The sites use very similar software.
- The sites use identical DBMS or DBMS from the same vendor.
- Each site is aware of all other sites and cooperates with other sites to process user requests.
- The database is accessed through a single interface as if it is a

Explanation of each 1M

single database.

Heterogeneous Distributed Databases:

In a heterogeneous distributed database, different sites have different operating systems, DBMS products and data models. Its properties are −

- Different sites use dissimilar schemas and software.
- The system may be composed of a variety of DBMSs like relational, network, hierarchical or object oriented.
- Query processing is complex due to dissimilar schemas.
- Transaction processing is complex due to dissimilar software.
- A site may not be aware of other sites and so there is limited co-operation in processing user requests.

(b) **Explain the steps for designing ER model.** **4M**

Ans.

1. Identify all the entities in the system. An entity should appear only once in a particular diagram. Create rectangles for all entities and name them properly.
2. Identify relationships between entities. And find the types of the relationships. Remove redundant or unnecessary relationships between entities.
3. Add attributes for entities. Identify their types so as to write them with appropriate symbols. Give meaningful attribute names so they can be understood easily.
4. Connect them using a line and add a diamond in the middle describing the relationship. Never connect a relationship to another relationship.

Each step 1M

(c) **Give difference between multi-valued dependencies and functional dependency (four points).** **4M**

Ans.

Sr. No.	Functional dependency	Multi-valued dependencies
1	This occurs when one attribute in a relation is unique	This occurs when more than one attributes are dependent on another.
2	It is represented as X-->Y	It is represented as X-->>Y
3	It plays a role in the 1NF	It plays a role in the 4NF

Any four points 1M each

	database normalization.	database normalization.
4	Example: EmpID -> EmpName, EmpName is functionally dependent on EmpID because EmpName can take only one value for the given value of EmpID	Example: Person->->mobile, Person->->food_likes This is described as: Persom multi determines "mobile" and "person multidetermines food_likes"

(d) **List four data integrity constraints. Explain each one of them in detail.** 4M

Ans. Integrity constraints:
1)Not null
2)check
3)unique
4)primary key
5)referential integrity constraints

List 2M

1) **Not Null:** By default, all columns in tables allows null values. When a NOT NULL Constraint is enforced on column or set of columns it will not allow null values.

Example

SQL> CREATE TABLE STUDENT (ROLL_NO NUMBER (5), NAME VARCHAR2 (20) NOT NULL);

Explanation of any 2 2M

2) **Check Constraint**: The constraint defines a condition that each row must satisfy. A single Column can have multiple check condition.

Example

SQL> CREATE TABLE EMP (ID NUMBER (5), NAME VARCHAR2 (10), SAL NUMBER (10) CONSTRINT CHK_SAL CHECK (SAL>15000));

3) **Primary Key constraint:** It is used to avoid redundant/duplicate value entry within the row of specified column in table. It restricts null values too.

Example

		SQL> CREATE TABLE EMP (ID NUMBER (5) CONSTRAINT ID_PK PRIMARY KEY, NAME VARCHAR2 (10), SAL NUMBER (10)); 3) **Unique Constraint**: The UNIQUE constraint uniquely identifies each record in a database table. The UNIQUE and PRIMARY KEY constraints both provide a guarantee for uniqueness of a column or set of columns. It allows null value. ***Example*** **SQL**>CREATE TABLE PERSONS (P_ID NUMBER CONSTRAINT P_UK UNIQUE, FIRSTNAME VARCHAR2(20), CITY VARCHAR2(20)); **4) Referential Integrity Constraint**: It is a relational database concept in which multiple tables share a relationship based on the data stored in the tables, and that relationship must remain consistent. A value of foreign key is derived from primary key which is defined in parent table. ***Example*** **SQL**>CREATE TABLE DEPARTMENT (EMP_ID NUMBER(5) REFERENCESEMP(EMP_ID), DNO NUMBER(3));	
	(e) **Ans.**	**Explain the following synonyms in SQL with syntax and example:** **(i) Creating synonyms** **(ii) Dropping synonyms** A **synonym** is an alternative name for objects such as tables, views, sequences, stored procedures, and other database objects. Synonyms are generally used granting access to an object from another schema and when you don't want the users to have to worry about knowing which schema owns the object. **(i) creating synonyms** ***Syntax:*** *The syntax to create a synonym is:* *CREATE [OR REPLACE] [PUBLIC] SYNONYM synonym_name FOR [tablename];* OR REPLACE Allows you to recreate the synonym (if it already exists) without having to issue a DROP synonym command.	**4M** ***Explanation 1M*** ***Syntax 1M***

PUBLIC
It means that the synonym is a public synonym and is accessible to all users. Remember though that the user must first have the appropriate privileges to the object to use the synonym.

Tablename
The name of the object for which you are creating the synonym. It can be one of the following:
Table,view,sequence,stored procedure,function,package,materialized view

Creating synonym example 1M

Example

CREATE PUBLIC SYNONYM suppliers FOR suppliers_table;

(ii) Dropping synonyms:

Syntax:
The syntax to drop a synonym in Oracle is:
DROP SYNONYM [synonym_name]

DROP SYNONYM suppliers;

Dropping synonym 1M

(f) List and explain predefined expections in PL/SQL **4M**

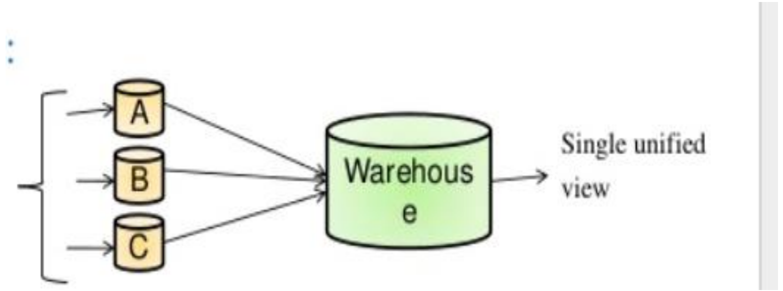
Ans.

Exception	**Raised when ...**
ACCESS_INTO_NULL	Your program attempts to assign values to the attributes of an uninitialized (atomically null) object.
CASE_NOT_FOUND	None of the choices in the WHEN clauses of a CASE statement is selected, and there is no ELSE clause.
COLLECTION_IS_NULL	Your program attempts to apply collection methods other than EXISTS to an uninitialized (atomically null) nested table or varray, or the program attempts to assign values to the elements of an uninitialized nested table or varray.
CURSOR_ALREADY_OPEN	Your program attempts to open an already open cursor. A cursor must be closed before it can be reopened. A cursor FOR loop automatically opens the cursor to

Any four predefined exception with explanation 1M each

	which it refers. So, your program cannot open that cursor inside the loop.
DUP_VAL_ON_INDEX	Your program attempts to store duplicate values in a database column that is constrained by a unique index.
INVALID_CURSOR	Your program attempts an illegal cursor operation such as closing an unopened cursor.
INVALID_NUMBER	In a SQL statement, the conversion of a character string into a number fails because the string does not represent a valid number. (In procedural statements, VALUE_ERROR is raised.) This exception is also raised when the LIMIT-clause expression in a bulk FETCH statement does not evaluate to a positive number.
LOGIN_DENIED	Your program attempts to log on to Oracle with an invalid username and/or password.
NO_DATA_FOUND	A SELECT INTO statement returns no rows, or your program references a deleted element in a nested table or an uninitialized element in an index-by table. SQL aggregate functions such as AVG and SUM always return a value or a null. So, a SELECT INTO statement that calls an aggregate function never raises NO_DATA_FOUND. The FETCH statement is expected to return no rows eventually, so when that happens, no exception is raised.
NOT_LOGGED_ON	Your program issues a database call without being connected to Oracle.
PROGRAM_ERROR	PL/SQL has an internal problem.
ROWTYPE_MISMATCH	The host cursor variable and PL/SQL cursor variable involved in an assignment have incompatible return types. For example, when an open host cursor

	variable is passed to a stored subprogram, the return types of the actual and formal parameters must be compatible.
SELF_IS_NULL	Your program attempts to call a MEMBER method on a null instance. That is, the built-in parameter SELF (which is always the first parameter passed to a MEMBER method) is null.
STORAGE_ERROR	PL/SQL runs out of memory or memory has been corrupted.
SUBSCRIPT_BEYOND_COUNT	Your program references a nested table or varray element using an index number larger than the number of elements in the collection.
SUBSCRIPT_OUTSIDE_LIMIT	Your program references a nested table or varray element using an index number (-1 for example) that is outside the legal range.
SYS_INVALID_ROWID	The conversion of a character string into a universal rowid fails because the character string does not represent a valid rowid.
TIMEOUT_ON_RESOURCE	A time-out occurs while Oracle is waiting for a resource.
TOO_MANY_ROWS	A SELECT INTO statement returns more than one row.
VALUE_ERROR	An arithmetic, conversion, truncation, or size-constraint error occurs. For example, when your program selects a column value into a character variable, if the value is longer than the declared length of the variable, PL/SQL aborts the assignment and raises VALUE_ERROR. In procedural statements, VALUE_ERROR is raised if the conversion of a character string into a number fails. (In SQL statements, INVALID_NUMBER is raised.)
ZERO_DIVIDE	Your program attempts to divide a number by zero.

4.		**Attempt any FOUR of the following:**	**16**
	(a) **Ans.**	**Explain properties of Dataware housing with neat labelled diagram.** ***(Note: Any relevant diagram can be considered)***  Data warehouses are repositories of high-volume information. They are centralized stores of all the data a company may generate, formed by relational databases and designed for query and analysis. Data warehouses allow for quick, accurate access to structured data via predefined queries. There are three prominent data warehouse characteristics: • **Integrated**: The way data is extracted and transformed is uniform, regardless of the original source. • **Time-variant**: Data is organized via time-periods (weekly, monthly, annually, etc.). • **Non-volatile**: A data warehouse is not updated in real-time. It is periodically updated via the uploading of data, protecting it from the influence of momentary change.	**4M** ***Diagram 1M*** ***Any 3 Properties 3M***
	(b) **Ans.**	**Explain four phases of database design in detail.** Four phases of database design are : **1. Conceptual design** When every data requirement is stored and analyzed, the next thing that we need to do is creating a conceptual database plan. Here, a highly leveled conceptual data model is used. This phase is called conceptual design. **2. Logical Design** The logical phase of database design is also called the data modeling mapping phase. This phase gives us a result of relation schemas. The	**4M** ***Four phases 1M each***

basis for these schemas is the ER or the Class Diagram.

3. Normalization

The main purpose of normalization is to remove redundancy and every other anomaly during the update.

Normalization in database design is a way to change the relation schema to reduce any excessive and redundant data. With every normalization phase, a new table is added to the database.

4. Physical Design

The last phase of database design is the physical design phase. In this phase, we implement the database design. Here, a DBMS (Database Management System) must be chosen to use.

(c) **List types of SQL operators. Explain each one of them in detail.** **4M**

Ans. **Types of SQL operators :**

List 2M

1) SQL Arithmetic Operators
2) SQL Comparison Operators
3) SQL Logical Operators

Arithmetic operators are used to perform arithmetic operations on numbers. They are +,-,*,/ and %.

Explanation 2M

Comparison operators are used in between two variables to compare their values. They are <,>,<=,>=,=,!=or <>,!< and !>.

Logical operators are used for the boolean results in sql queries for comparison of values from the attributes of the tables. Eg: Any, Exists, All, Like, Between, In etc.

(OR from each category given below, explanation of any 2 operators can be considered)

SQL Arithmetic Operators

Assume 'variable a' holds 10 and 'variable b' holds 20, then –

Operator	Description	Example
+ (Addition)	Adds values on either side of the operator.	a + b will give 30
- (Subtraction)	Subtracts right hand operand from left hand operand.	a - b will give -10
*	Multiplies values on either side	a * b will give

(Multiplication)	of the operator.	200
/ (Division)	Divides left hand operand by right hand operand.	b / a will give 2
% (Modulus)	Divides left hand operand by right hand operand and returns remainder.	b % a will give 0

SQL Comparison Operators

Assume 'variable a' holds 10 and 'variable b' holds 20, then –

Operator	Description	Example
=	Checks if the values of two operands are equal or not, if yes then condition becomes true.	(a = b) is not true.
!=	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(a != b) is true.
<>	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(a <> b) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(a > b) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(a < b) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(a >= b) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(a <= b) is true.

!<	Checks if the value of left operand is not less than the value of right operand, if yes then condition becomes true.	(a !< b) is false.
!>	Checks if the value of left operand is not greater than the value of right operand, if yes then condition becomes true.	(a !> b) is true.

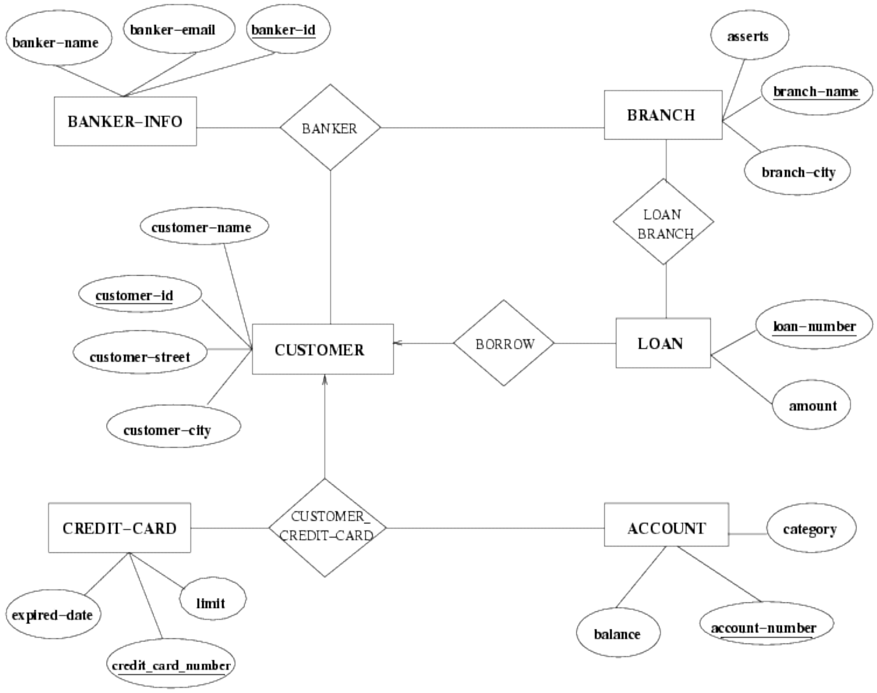
SQL Logical Operators

Sr. No.	Operator & Description
1	**ALL :** The ALL operator is used to compare a value to all values in another value set.
2	**AND :** The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause.
3	**ANY :** The ANY operator is used to compare a value to any applicable value in the list as per the condition.
4	**BETWEEN** : The BETWEEN operator is used to search for values that are within a set of values, given the minimum value and the maximum value.
5	**EXISTS :** The EXISTS operator is used to search for the presence of a row in a specified table that meets a certain criterion.
6	**IN :** The IN operator is used to compare a value to a list of literal values that have been specified.
7	**LIKE :** The LIKE operator is used to compare a value to similar values using wildcard operators.
8	**NOT :** The NOT operator reverses the meaning of the logical operator with which it is used. Eg: NOT EXISTS, NOT BETWEEN, NOT IN, etc. **This is a negate operator.**
9	**OR :** The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause.
10	**IS NULL :** The NULL operator is used to compare a value with a NULL value.
11	**UNIQUE :** The UNIQUE operator searches every row of a specified table for uniqueness (no duplicates).

	(d) Ans.	**Define views with its syntax and list four disadvantages of views.** View is the simply subset of table which are stored logically in a database. A view is a virtual table in the database whose contents are defined by a query. **Syntax:** CREATE VIEW view_name AS SELECT columns FROM tables [WHERE conditions]; view_name The name of the SQL VIEW that you wish to create. WHERE conditions Optional. The conditions that must be met for the records to be included in the VIEW. **Disadvantages of view** 1. The main disadvantage to using views rather than real tables is performance degradation. 2. views only create the appearance of a table, not a real table. 3. When a user tries to update rows of a view, the DBMS must translate the request into an update on rows of the underlying source table. This is possible for simple views, but more complicated views cannot be updated. 4. If the view is defined by a complex, multi-table query, even simple queries against the view become complicated joins. 5. If views are defined by complex queries, they can take a long time to complete.	**4M** ***Definition 1M*** ***Syntax 1M*** ***Four disadvantages $^{1/2}$M each***

	(e)
	
	Ans.
		Write a program using storage error predefined exception in PL/SQL.
		DECLARE
		temp varchar(20);
		BEGIN
		SELECT e_id into temp from emp where e_name='ABC';
		EXCEPTION
		WHEN storage_error THEN
		dbms_output.put_line('Out of Memory...');
		END;
			4M
			Correct syntax 2M
			
			Use of storage error exception 2M

	(f) Ans.	**Define cursors, list types of cursors and give purpose of using cursor.** **Definition :** A cursor is a temporary work area created in the system memory when a SQL statement is executed. **There are two types of cursors in PL/SQL as follows:** 1) Implicit Cursors 2) Explicit Cursors **Purpose of using cursor:** 1. A cursor contains information on a select statement and the rows of data accessed by it. This temporary work area is used to store the data retrieved from the database, and manipulate this data. 2. The major function of a cursor is to retrieve data, one row at a time, from a result set, unlike the SQL commands which operate on all the rows in the result set at one time. 3. Cursors are used when the user needs to update records in a singleton fashion or in a row by row manner, in a database table.	**4M** ***Definition 1M*** ***Types 1M*** ***Purpose 2M***
5.	(a) Ans.	**Attempt any FOUR of the following:** **Draw E-R diagram for bank management system (Assume suitable data).**	**16** **4M**

		 banker-name banker-email banker-id asserts branch-name BANKER-INFO BANKER BRANCH branch-city LOAN BRANCH customer-name customer-id customer-street CUSTOMER BORROW LOAN loan-number amount customer-city CREDIT-CARD CUSTOMER_ CREDIT-CARD ACCOUNT category expired-date limit credit_card_number balance account-number	*2M for proper notations* *2M for logic*
	(b) **Ans.**	**Explain the following string functions of SQL with syntax in detail with example:** **(i) REPLACE** **(ii) TRANSLATE** **(i) REPLACE:** REPLACE(string, old_substring, new_substring); The REPLACE() function replaces all occurrences of a substring within a string, with a new substring select replace('Click','Cl','T') from dual; output: Tick **(ii) TRANSLATE:** Translate(string,from_string,to_string)_ The Oracle TRANSLATE() function returns a string with all occurrences of each character in a string replaced by its corresponding character in another string Select translate('1val23','123','456') from dual; Output:4val56	**4M** ***Each explanation 1M*** ***Each example 1M***

	(c) Ans.	**Explain ACID properties of transactions.** A transaction is a single unit of execution. It can have different steps. Every transaction in the DBMS must follow the ACID properties. The ACID properties are: 1. A-Atomicity 2. C-Consistency 3. I-Isolation 4. D-Durability **Atomicity:** This property states that every transaction should be treated as an atomic unit that is, either the entire transaction should be completed totally or it should not be done at all. It also states that under no condition should a transaction be partially completed. **Consistency:** The database must remain in consistent state after any transaction. The execution of a transaction should not result in inconsistency of the database. **Isolation:** in systems where more than one transaction execute simultaneously and in parallel, all transaction will be carried out and each transaction should feel that it is the only transaction happening. If Ti, Tj are two transactions, then Ti should feel that it is the only transaction happening while it is executing, either Tj should have completed execution Indexes or will execute once Ti completes. **Durability:** The database should be durable enough to hold all its latest updates even if the system fails or restarts. If a transaction updates data in a database and commits, then the database will hold the modified data. If a transaction commits but the system fails before the data could be written on to the disk, then that data will be updated once the system starts.	**4M** ***Each property 1M***
	(d) Ans.	**Explain how indexes can be used for both single column and multiple column.** Indexes are structures that the database search engine can use to speed up data retrieval i.e, an index is a pointer to data in a table. An index helps to speed up SELECT queries. Creating an index involves the CREATE INDEX statement, which allows to name the index, to specify the table and which column or columns to index,	**4M**

		and to indicate whether the index is in an ascending or descending order **Single-Column Indexes** A single-column index is created based on only one table column. Create index index_name on table_name(column_name) **Unique Indexes** Unique indexes are used not only for performance, but also for data integrity. A unique index does not allow any duplicate values to be inserted into the table. Create unique index index_name on table_name(Column_name) **Composite Indexes** A composite index is an index on two or more columns of a table. Its basic syntax is as follows. Create index index_name on table_name(column_name1,column_name2)	***Explanation of single column 2M*** ***Multiple column 2M***
	(e) **Ans.**	**Define locking. List two types of locks and explain any one locking protocol.** A lock is a mechanism to control concurrent access to a data item. Lock requests are made to concurrency-control manager. Transaction can proceed only after request is granted. A transaction may be granted a lock on an item if the requested lock is compatible with locks already held on the item by other transactions Data items can be locked in two modes : 1. exclusive(X) mode. Data item can be both read as well as written. X-lock is requested using lock-X instruction. 2. shared(S) mode. Data item can only be read. S-lock is requested using lock-S instruction. Any number of transactions can hold shared locks on an item, but if any transaction holds an exclusive on the item no other transaction may hold any lock on the item. If a lock cannot be granted, the requesting transaction is made to wait till all incompatible locks held by other transactions have been released. The lock is then granted.	**4M** ***Definition and types 2M*** ***Explanation of any 1 type 2M***

Two phase locking protocol:
This is a protocol which ensures conflict-serializable schedules.

Phase 1: Growing Phase

transaction may obtain locks
transaction may not release locks

Phase 2: Shrinking Phase

transaction may release locks
transaction may not obtain locks

The protocol assures serializability. It can be proved that the transactions can be serialized in the order of their lock points, i.e. the point where a transaction acquired its final lock. Two-phase locking does not ensure freedom from deadlocks. Cascading roll-back is possible under two-phase locking. To avoid this, a modified protocol called strict two-phase locking is followed. Here a transaction must hold all its exclusive locks till it commits/aborts

Time stamp based protocol:
Each transaction is issued a timestamp when it enters the system. If an old transaction Ti has time-stamp TS(Ti), a new transaction Tj is assigned time-stamp TS(Tj) such that TS(Ti) <TS(Tj). The protocol manages concurrent execution such that the time-stamps determine the serializability order. In order to assure such behaviour, the protocol maintains for each data Q two timestamp values:
W-timestamp(Q) is the largest time-stamp of any transaction that executed write(Q) successfully.
R-timestamp(Q) is the largest time-stamp of any transaction that executed read(Q) successfully.

The timestamp ordering protocol ensures that any conflicting read and write operations are executed in timestamp order.

Validation based protocol:
Execution of transaction Ti is done in three phases.

1. Read and execution phase: Transaction Ti writes only to temporary local variables
2. Validation phase: Transaction Ti performs a validation test to determine if local variables can be written without violating serializability.

		3. Write phase: If Ti is validated, the updates are applied to the database; otherwise, Ti is rolled back. The three phases of concurrently executing transactions can be interleaved, but each transaction must go through the three phases in that order. Each transaction Ti has 3 timestamps Start(Ti) : the time when Ti started its execution Validation(Ti): the time when Ti entered its validation phase Finish(Ti) : the time when Ti finished its write phase Serializability order is determined by timestamp given at validation time, to increase concurrency.	
	(f) **Ans**	**Explain triggers. When it is used?** **Trigger:** A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. A trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated. **General syntax:** create trigger [trigger_name] [before | after] insert | update | delete] on [table_name] [for each row] [trigger_body] Triggers are written to be executed in response to any of the following events − • A **database manipulation (DML)** statement (DELETE, INSERT, or UPDATE) • A **database definition (DDL)** statement (CREATE, ALTER, or DROP). • A **database operation** (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN). Triggers can be defined on the table, view, schema, or database with which the event is associated. Triggers can be written for the following purposes −	**4M** ***Trigger explanation with general syntax 2M*** ***Usage 2M***

		• Generating some derived column values automatically • Enforcing referential integrity • Event logging and storing information on table access • Auditing • Synchronous replication of tables • Imposing security authorizations • Preventing invalid transactions	
6.	(a) Ans.	**Attempt any FOUR of the following:** **Explain Relational Algebra with example for** **(i) Set difference operation** **(ii) Natural join** Relational algebra is a widely used procedural query language. It collects instances of relations as input and gives occurrences of relations as output. It uses various operation to perform this action. **(i) Set difference operation(-):** It is denoted by - symbol. The result of A - B, is a relation which includes all tuples that are in A but not in B. • The attribute name of A has to match with the attribute name in B. • The two-operand relations A and B should be either compatible or Union compatible. • It should be defined relation consisting of the tuples that are in relation A, but not in B. π_{cust_id}(account)- π_{cust_id}(loan) – here all the customers who has account but not loan will be displayed. **(ii) Natural join(⋈):**Natural join is used to join more than one table. Natural join can only be performed if there is a common attribute (column) between the relations. The name and type of the attribute must be same. Consider the following tables:	16 4M ***Each Explanation 1M*** ***Example 1M***

A

num	square
2	4
3	9

B

num	cube
2	8
3	27

$A \bowtie B$

num	square	cube
2	4	8
3	9	27

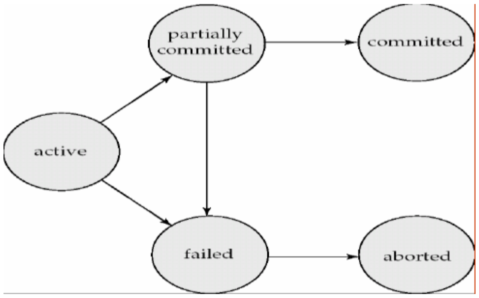
(b) Enlist features of
(i) tuple relational calculus (four points)
(ii) domain relational calculus

4M

Ans.

(i) tuple relational calculus:
Tuple Relational Calculus is a **non-procedural query language**.
Tuple Calculus provides only the description of the query but it does not provide the methods to solve it. Thus, it explains what to do but not how to do.
In Tuple Calculus, a query is expressed as
{t|P(t)}
Where t represents the resulting tuples.
P(t) – is called a predicate which gives the condition to get t
Thus, it generates set of all tuples t, such that Predicate P(t) is true for

Each explanation 2M

		t. P(t) may have various conditions logically combined with OR (∨), AND (∧), NOT(¬). It also uses quantifiers: $\exists\ t \in r\ (Q(t))$ = ”there exists” a tuple in t in relation r such that predicate Q(t) is true. *Eg:* $\{t \vert t\in \text{Loan} \wedge t[\text{amount}]>=10000\}$ **(ii) domain relational calculus:** Domain relational calculus uses list of attribute to be selected from the relation based on the condition. It is same as Tuple Relational Calculus, but differs by selecting the attributes rather than selecting whole tuples. It is denoted as below: $\{< a_1, a_2, a_3, \dots a_n > \vert P(a_1, a_2, a_3, \dots a_n)\}$ Where $a_1, a_2, a_3, \dots a_n$ are attributes of the relation and P is the condition. *Eg:* $\{<\text{name}>,<\text{age}>\vert\ \in \text{Student} \wedge \text{age}>17\}$	
	(c) **Ans.**	**Draw and explain different steps of transactions.**  partially committed committed active failed aborted Active –the initial state; the transaction stays in this state while it is executing Partially committed –after the final statement has been executed. Failed - after the discovery that normal execution can no longer proceed. Aborted – after the transaction has been rolled back and the database restored to its state prior to the start of the transaction. Two options after it has been aborted: restart the transaction - can be done only if no internal logical error kill the transaction Committed –after successful completion.	**4M** ***Diagram 2M*** ***Explanation 2M***

	(d) Ans.	**Define serializability and explain different types of schedule.** Each transaction preserves database consistency. The serial execution of a set of transactions preserves database consistency. A concurrent schedule is serializable if it is equivalent to a serial schedule. Different forms of schedule equivalence give rise to two forms of serializability 1.conflict serializability 2. view serializability **Schedule:** A sequences of instructions that specify the chronological order in which instructions of concurrent transactions are execute. A schedule for a set of transactions must consist of all instructions of those transactions. It must preserve the order in which the instructions appear in each individual transaction. **The different types of schedules are:** **Serial schedule:** Schedules in which the transactions are executed non-interleaved, i.e., a serial schedule is one in which no transaction starts until a running transaction has ended are called serial schedules. **Recoverable Schedules:** Schedules in which transactions commit only after all transactions whose changes they read commit are called recoverable schedules. In other words, if some transaction T_j is reading value updated or written by some other transaction T_i, then the commit of T_j must occur after the commit of T_i. **Cascade less Schedules:** Schedules in which transactions read values only after all transactions whose changes they are going to read commit are called cascade less schedules. It avoids that a single transaction abort leads to a series of transaction rollbacks. A strategy to prevent cascading aborts is to disallow a transaction from reading uncommitted changes from another transaction in the same schedule **Strict Schedule:** In Strict schedule, if the write operation of a transaction precedes a conflicting operation (Read or Write operation) of another transaction then the commit or abort operation of such transaction should also precede the conflicting operation of other	4M *Definition 2M* *Any 2 types explanation 1M each*

transaction.

(e) **Give difference between stored procedure and PL/SQL function (four points).** **4M**

Ans.

Sr. No.	Stored procedure	PL/SQL function
1	Stored procedure may return a value	Function must return a value
2	It is compiled every time it is called	It is precompiled
3	Can have only input parameters	Can have input and output parameters
4	Can be called from stored procedures	Cannot be called from functions
5	Allows only select	Allows any DML query and select query
6	Return type is not must	Return type is must

Any four points 1M each

(f) **Explain fetching record from cursor with example.** **4M**

Ans.

```
Eg:
Declare
enumemp.eno%type;
enemp.ename%type;
Cursor cur is select eno, ename from emp where jobname = ‘mgr’;
Begin
Open cur;
Loop
Fetch cur into enum,en;
Exit when cur%NOTFOUND;
Dbms_output.put_line(‘emp num ’||enum||’ emp name ‘||en);
End loop;
Close cur;
End;
/
```

Example 2M

The example shows fetching multiple records using cursor.

A cursor is a temporary work area created in system memory when

Explana

		a SQL statement is executed. A cursor is a set of rows together with a pointer that identifies a current row. In the example, the cursor is defined to hold the rows as defined by the select query. Once the cursor is defined, the next step is to open the cursor. When the cursor is opened, it is ready to retrieve the rows. This is done using the fetch statement. Since there are many rows, a loop is used to display the values of all the rows. Once the rows are fetched, the cursor should be closed.	***tion 2M***