

1. A relational database consists of a collection of

- a) Tables
- b) Fields
- c) Records
- d) Keys

**Answer: a**

2. A \_\_\_\_\_ in a table represents a relationship among a set of values.

- a) Column
- b) Key
- c) Row
- d) Entry

**Answer: c**

3. The term \_\_\_\_\_ is used to refer to a row.

- a) Attribute
- b) Tuple
- c) Field
- d) Instance

**Answer: b**

4. The term attribute refers to a \_\_\_\_\_ of a table.

- a) Record
- b) Column
- c) Tuple
- d) Key

**Answer: b**

5. For each attribute of a relation, there is a set of permitted values, called the \_\_\_\_\_ of that attribute.

- a) Domain

- b) Relation
- c) Set
- d) Schema

**Answer: a**

6. Database \_\_\_\_\_ which is the logical design of the database, and the database \_\_\_\_\_ which is a snapshot of the data in the database at a given instant in time.

- a) Instance, Schema
- b) Relation, Schema
- c) Relation, Domain
- d) Schema, Instance

**Answer: d**

7. Course(course\_id,sec\_id,semester)

Here the course\_id,sec\_id and semester are \_\_\_\_\_ and course is a \_\_\_\_\_

- a) Relations, Attribute
- b) Attributes, Relation
- c) Tuple, Relation
- d) Tuple, Attributes

**Answer: b**

8. Department (dept name, building, budget) and Employee (employee\_id, name, dept name, salary)

Here the dept\_name attribute appears in both the relations. Here using common attributes in relation schema is one way of relating \_\_\_\_\_ relations.

- a) Attributes of common
- b) Tuple of common
- c) Tuple of distinct
- d) Attributes of distinct

**Answer: c**

9. A domain is atomic if elements of the domain are considered to be \_\_\_\_\_ units.

- a) Different

- b) Indivisible
- c) Constant
- d) Divisible

**Answer: b**

10. The tuples of the relations can be of \_\_\_\_\_ order.

- a) Any
- b) Same
- c) Sorted
- d) Constant

**Answer: a**

1. Which one of the following is a set of one or more attributes taken collectively to uniquely identify a record?

- a) Candidate key
- b) Sub key
- c) Super key
- d) Foreign key

**Answer: c**

2. Consider attributes ID, CITY and NAME. Which one of this can be considered as a super key?

- a) NAME
- b) ID
- c) CITY
- d) CITY, ID

**Answer: b**

3. The subset of a super key is a candidate key under what condition?

- a) No proper subset is a super key
- b) All subsets are super keys
- c) Subset is a super key
- d) Each subset is a super key

**Answer: a**

4. A \_\_\_\_ is a property of the entire relation, rather than of the individual tuples in which each tuple is unique.

- a) Rows
- b) Key
- c) Attribute
- d) Fields

**Answer: b**

5. Which one of the following attribute can be taken as a primary key?

- a) Name
- b) Street
- c) Id
- d) Department

**Answer: c**

6. Which one of the following cannot be taken as a primary key?

- a) Id
- b) Register number
- c) Dept\_id
- d) Street

**Answer: d**

7. An attribute in a relation is a foreign key if the \_\_\_\_\_ key from one relation is used as an attribute in that relation.

- a) Candidate
- b) Primary
- c) Super
- d) Sub

**Answer: b**

8. The relation with the attribute which is the primary key is referenced in another relation. The relation which has the attribute as a primary key is called \_\_\_\_\_

- a) Referential relation
- b) Referencing relation
- c) Referenced relation
- d) Referred relation

**Answer: c**

9. The \_\_\_\_\_ is the one in which the primary key of one relation is used as a normal attribute in another relation.

- a) Referential relation
- b) Referencing relation
- c) Referenced relation
- d) Referred relation

**Answer: c**

10. A \_\_\_\_\_ integrity constraint requires that the values appearing in specified attributes of any tuple in the referencing relation also appear in specified attributes of at least one tuple in the referenced relation.

- a) Referential
- b) Referencing
- c) Specific
- d) Primary

**Answer: a**

1. Using which language can a user request information from a database?

- a) Query
- b) Relational
- c) Structural
- d) Compiler

**Answer: a**

2. Student(ID, name, dept name, tot\_cred)  
In this query which attributes form the primary key?

- a) Name
- b) Dept

- c) Tot\_cred
- d) ID

**Answer: d**

3. Which one of the following is a procedural language?

- a) Domain relational calculus
- b) Tuple relational calculus
- c) Relational algebra
- d) Query language

**Answer: c**

4. The\_\_\_\_\_ operation allows the combining of two relations by merging pairs of tuples, one from each relation, into a single tuple.

- a) Select
- b) Join
- c) Union
- d) Intersection

**Answer: b**

5. The result which operation contains all pairs of tuples from the two relations, regardless of whether their attribute values match.

- a) Join
- b) Cartesian product
- c) Intersection
- d) Set difference

**Answer: b**

6. The \_\_\_\_\_operation performs a set union of two “similarly structured” tables

- a) Union
- b) Join
- c) Product
- d) Intersect

**Answer: a**

7. The most commonly used operation in relational algebra for projecting a set of tuple from a relation is

- a) Join
- b) Projection
- c) Select
- d) Union

**Answer: c**

8. The \_\_\_\_\_ operator takes the results of two queries and returns only rows that appear in both result sets.

- a) Union
- b) Intersect
- c) Difference
- d) Projection

**Answer: b**

9. A \_\_\_\_\_ is a pictorial depiction of the schema of a database that shows the relations in the database, their attributes, and primary keys and foreign keys.

- a) Schema diagram
- b) Relational algebra
- c) Database diagram
- d) Schema flow

**Answer: a**

10. The \_\_\_\_\_ provides a set of operations that take one or more relations as input and return a relation as an output.

- a) Schematic representation
- b) Relational algebra
- c) Scheme diagram
- d) Relation flow

**Answer: b**

1. What is a superkey in the context of the relational model?

- a) A minimal set of attributes that uniquely identifies a tuple
- b) A set of attributes that uniquely identifies all tuples in a relation
- c) A candidate key with the fewest attributes
- d) A primary key for a relation

**Answer: b) A set of attributes that uniquely identifies all tuples in a relation**

2. In a relational database, what does the term "functional dependency" describe?

- a) A relationship between tables in a database
- b) A set of attributes that determines another set of attributes
- c) A way to link records from different relations
- d) A type of data integrity constraint

**Answer: b) A set of attributes that determines another set of attributes**

3. Which of the following is not a normal form in the context of database design?

- a) First Normal Form (1NF)
- b) Second Normal Form (2NF)
- c) Third Normal Form (3NF)
- d) Unique Normal Form (UNF)

**Answer: d) Unique Normal Form (UNF)**

4. Consider two relations,  $R(A, B)$  and  $S(B, C)$ . What is the result of the natural join  $R \bowtie S$ ?

- a)  $R$
- b)  $S$
- c)  $R \bowtie S = R \times S$



d) The set of all combinations of tuples from R and S where B matches

**Answer: d) The set of all combinations of tuples from R and S where B matches**

5. Which of the following SQL clauses is used to eliminate duplicate rows from a result set?

- a) SELECT DISTINCT
- b) GROUP BY
- c) HAVING
- d) ORDER BY

**Answer: a) SELECT DISTINCT**

6. What is the cardinality of a relation?

- a) The number of tuples in a relation
- b) The number of attributes in a relation
- c) The number of unique values in an attribute
- d) The degree of the relation

**Answer: a) The number of tuples in a relation**

7. In the relational model, what is the purpose of the foreign key?

- a) It ensures that a relation does not have duplicate rows.
- b) It enforces referential integrity by linking one table to another.
- c) It defines the primary key of a relation.
- d) It is used for indexing columns in a table.

**Answer: b) It enforces referential integrity by linking one table to another.**

8. What is the Boyce-Codd Normal Form (BCNF) in database design?

- a) A stronger form of Third Normal Form (3NF)
- b) A way to define primary keys for a relation
- c) A method for designing non-relational databases
- d) A form of data encryption

**Answer: a) A stronger form of Third Normal Form (3NF)**

9. Which SQL command is used to add a new attribute (column) to an existing table?

- a) ALTER TABLE
- b) ADD COLUMN
- c) INSERT INTO
- d) UPDATE TABLE

**Answer: a) ALTER TABLE**

10. What is the purpose of an index in a relational database?

- a) To define the primary key of a table
- b) To improve the performance of queries by providing quick access to data
- c) To store historical data for auditing purposes
- d) To define constraints on data integrity

**Answer: b) To improve the performance of queries by providing quick access to data**

11. What is a functional dependency in a relation?

- a) A unique key that identifies a tuple
- b) A constraint that ensures no null values in a column
- c) A constraint specifying how one set of attributes determines another

d) A type of join operation in SQL

Answer: c) A constraint specifying how one set of attributes determines another

12. Which of the following statements about candidate keys is correct?

- a) A relation can have multiple candidate keys.
- b) A candidate key is always a primary key.
- c) A candidate key cannot contain composite attributes.
- d) Candidate keys are optional in a relational database.

Answer: a) A relation can have multiple candidate keys.

13. What is the purpose of the SQL command "JOIN" in relational databases?

- a) To create a new table by combining two existing tables.
- b) To remove rows from a table that do not match a specified condition.
- c) To add new columns to a table.
- d) To retrieve data from multiple tables based on a related column.

Answer: d) To retrieve data from multiple tables based on a related column.

14. In the context of relational algebra, what does the operator  $\sigma$  (sigma) represent?

- a) Union
- b) Projection
- c) Selection
- d) Join

Answer: c) Selection

15. Which of the following is an example of a unary relationship in a relational model?

- a) One-to-one relationship
- b) Many-to-many relationship
- c) One-to-many relationship
- d) A relationship with a single entity or table

Answer: d) A relationship with a single entity or table

16. What is the purpose of referential integrity constraints in a relational database?

- a) To enforce unique constraints on attributes.
- b) To ensure that foreign keys have the same data type as primary keys.
- c) To maintain the consistency of relationships between tables.
- d) To improve query performance.

Answer: c) To maintain the consistency of relationships between tables.

17. Which of the following normal forms eliminates partial dependencies and is stricter than 3NF?

- a) First Normal Form (1NF)
- b) Second Normal Form (2NF)
- c) Boyce-Codd Normal Form (BCNF)
- d) Fourth Normal Form (4NF)

Answer: c) Boyce-Codd Normal Form (BCNF)

18. In a relational database, what does a Cartesian product (×) operation between two tables result in?

- a) A new table with all possible combinations of rows from the two tables.
- b) A table that contains the common rows between the two tables.
- c) A table that combines rows based on a specified join condition.
- d) A new table with only distinct values from the two tables.

Answer: a) A new table with all possible combinations of rows from the two tables.

19. Which SQL clause is used to group rows that have the same values in specified columns?

- a) HAVING
- b) ORDER BY
- c) GROUP BY
- d) DISTINCT

Answer: c) GROUP BY

20. What is the purpose of the "UNION" operator in SQL?

- a) To combine rows from multiple tables into a single result set.
- b) To remove duplicates from a result set.
- c) To update data in a table.
- d) To select specific columns from a table.

Answer: a) To combine rows from multiple tables into a single result set.

21. In a relational database, what is the purpose of the "ON DELETE CASCADE" constraint in a foreign key relationship?

- a) It enforces that foreign key values cannot be null.
- b) It prevents the deletion of a record in the parent table if child records exist.
- c) It automatically deletes related records in the child table when the parent record is deleted.
- d) It ensures that only one record in the child table is linked to a parent record.

Answer: c) It automatically deletes related records in the child table when the parent record is deleted.

22. Which normal form allows multivalued dependencies to be removed?

- a) First Normal Form (1NF)
- b) Second Normal Form (2NF)
- c) Third Normal Form (3NF)
- d) Fourth Normal Form (4NF)

Answer: d) Fourth Normal Form (4NF)

23. In the context of relational databases, what is a surrogate key?

- a) A key derived from the attributes of a relation.
- b) A temporary key used for data migration.
- c) A synthetic key created for the sole purpose of uniquely identifying records.
- d) A primary key consisting of multiple attributes.

Answer: c) A synthetic key created for the sole purpose of uniquely identifying records.

24. Which of the following SQL statements is used to add a new row to a table?

- a) INSERT INTO
- b) UPDATE
- c) ALTER TABLE
- d) DELETE

Answer: a) INSERT INTO

25. In the relational model, what is a self-join?

- a) A join operation involving two unrelated tables.
- b) A join operation between a table and itself.
- c) A join operation that combines multiple tables into a single table.
- d) A join operation between two tables with no common attributes.

Answer: b) A join operation between a table and itself.

26. What is the purpose of the "LIKE" operator in SQL?

- a) To perform arithmetic operations on numeric values.
- b) To compare two columns for equality.
- c) To search for a specified pattern in a string.
- d) To concatenate two strings.

Answer: c) To search for a specified pattern in a string.

27. Which type of join returns all rows from both tables, with NULL values in columns where there is no match?

- a) INNER JOIN
- b) LEFT JOIN (or LEFT OUTER JOIN)
- c) RIGHT JOIN (or RIGHT OUTER JOIN)
- d) FULL JOIN (or FULL OUTER JOIN)

Answer: d) FULL JOIN (or FULL OUTER JOIN)

28. What is a transaction in the context of a relational database?

- a) A set of SQL commands executed simultaneously.
- b) A procedure that returns a result set.
- c) A single SQL statement.
- d) A sequence of SQL statements treated as a single unit of work.

Answer: d) A sequence of SQL statements treated as a single unit of work.

29. Which of the following is an advantage of using views in a relational database?

- a) Views allow for faster data retrieval.
- b) Views reduce data redundancy.
- c) Views provide a physical representation of the data.
- d) Views automatically create primary keys.

Answer: b) Views reduce data redundancy.

30. What is the purpose of the "ROLLBACK" statement in SQL?

- a) To commit a transaction and save changes.
- b) To release locks on database objects.



- c) To undo changes made during the current transaction.
- d) To retrieve data from a database.

Answer: c) To undo changes made during the current transaction.

1. Which one of the following is used to define the structure of the relation, deleting relations and relating schemas?

- a) DML(Data Manipulation Language)
- b) DDL(Data Definition Language)
- c) Query
- d) Relational Schema

Answer: b

2. Which one of the following provides the ability to query information from the database and to insert tuples into, delete tuples from, and modify tuples in the database?

- a) DML(Data Manipulation Language)
- b) DDL(Data Definition Language)
- c) Query
- d) Relational Schema

Answer: a

3.

```
CREATE TABLE employee (name VARCHAR, id INTEGER)
```

What type of statement is this?

- a) DML
- b) DDL
- c) View
- d) Integrity constraint

Answer: b

4.

```
SELECT * FROM employee
```

What type of statement is this?

- a) DML
- b) DDL
- c) View
- d) Integrity constraint

Answer: a

5. The basic data type char(n) is a \_\_\_\_ length character string and varchar(n) is \_\_\_\_ length character.

- a) Fixed, equal
- b) Equal, variable
- c) Fixed, variable
- d) Variable, equal

Answer: c

6. An attribute A of datatype varchar(20) has the value "Avi". The attribute B of datatype char(20) has value "Reed". Here attribute A has \_\_\_\_ spaces and attribute B has \_\_\_\_ spaces.

- a) 3, 20
- b) 20, 4
- c) 20, 20
- d) 3, 4

Answer: a

7. To remove a relation from an SQL database, we use the \_\_\_\_ command.

- a) Delete
- b) Purge
- c) Remove
- d) Drop table

Answer: d

8.

```
DELETE FROM r;  //r - relation
```

This command performs which of the following action?

- a) Remove relation
- b) Clear relation entries
- c) Delete fields
- d) Delete rows

Answer: b.

9.

```
INSERT INTO instructor VALUES (10211, 'Smith', 'Biology', 66000);
```

What type of statement is this?

- a) Query
- b) DML
- c) Relational
- d) DDL

Answer: b

10. Updates that violate \_\_\_\_\_ are disallowed.

- a) Integrity constraints
- b) Transaction control
- c) Authorization
- d) DDL constraints

Answer: a

1.

Name
Annie
Bob
Callie
Derek

Which of these query will display the the table given above ?

- a) Select employee from name
- b) Select name
- c) Select name from employee
- d) Select employee

Answer: c

2. Here which of the following displays the unique values of the column?

```
SELECT _____ dept_name  
FROM instructor;
```

- a) All
- b) From
- c) Distinct
- d) Name

Answer: c

3. The \_\_\_\_\_ clause allows us to select only those rows in the result relation of the \_\_\_\_\_ clause that satisfy a specified predicate.

- a) Where, from
- b) From, select
- c) Select, from
- d) From, where

Answer: a

4. The query given below will not give an error. Which one of the following has to be replaced to get the desired output?

```
SELECT ID, name, dept name, salary * 1.1  
WHERE instructor;
```

- a) Salary\*1.1
- b) ID
- c) Where
- d) Instructor

Answer: c

5. The \_\_\_\_\_ clause is used to list the attributes desired in the result of a query.

- a) Where
- b) Select
- c) From
- d) Distinct

Answer: b

6. This Query can be replaced by which one of the following?

```
SELECT name, course_id
FROM instructor, teaches
WHERE instructor_ID= teaches_ID;
```

- a) Select name,course\_id from teaches,instructor where instructor\_id=course\_id;
- b) Select name, course\_id from instructor natural join teaches;
- c) Select name, course\_id from instructor;
- d) Select course\_id from instructor join teaches;

Answer: b

7.

```
SELECT * FROM employee WHERE salary>10000 AND dept_id=101;
```

Which of the following fields are displayed as output?

- a) Salary, dept\_id
- b) Employee
- c) Salary
- d) All the field of employee relation

Answer: d

8.

Employee_id	Name	Salary
1001	Annie	6000

1009	Ross	4500
1018	Zeith	7000

This is Employee table.

Which of the following employee\_id will be displayed for the given query?

```
SELECT * FROM employee WHERE employee_id>1009;
```

- a) 1009, 1001, 1018
- b) 1009, 1018
- c) 1001
- d) 1018

Answer: d

9. Which of the following statements contains an error?

- a) Select \* from emp where empid = 10003;
- b) Select empid from emp where empid = 10006;
- c) Select empid from emp;
- d) Select empid where empid = 1009 and lastname = 'GELLER';

Answer: d

10. In the given query which of the keyword has to be inserted?

```
INSERT INTO employee _____ (1002,Joey,2000);
```

- a) Table
- b) Values
- c) Relation
- d) Field

Answer: b

1.

```
SELECT name ____ instructor name, course id
FROM instructor, teaches
WHERE instructor.ID= teaches.ID;
```

Which keyword must be used here to rename the field name?

- a) From
- b) Rename
- c) As
- d) Join

Answer: c

2.

```
SELECT * FROM employee WHERE dept_name="Comp Sci";
```

In the SQL given above there is an error . Identify the error.

- a) Dept\_name
- b) Employee
- c) "Comp Sci"
- d) From

Answer: c

3.

```
SELECT emp_name  
FROM department  
WHERE dept_name LIKE ' ____ Computer Science';
```

Which one of the following has to be added into the blank to select the dept\_name which has Computer Science as its ending string?

- a) %
- b) \_
- c) ||
- d) \$

Answer: a

4. '\_\_\_' matches any string of \_\_\_\_\_ three characters. '\_\_\_%' matches any string of at \_\_\_\_\_ three characters.

- a) Atleast, Exactly
- b) Exactly, Atleast
- c) Atleast, All

d) All, Exactly

Answer: b

5.

```
SELECT name
FROM instructor
WHERE dept name = 'Physics'
ORDER BY name;
```

By default, the order by clause lists items in \_\_\_\_\_ order.

- a) Descending
- b) Any
- c) Same
- d) Ascending

Answer: d

6.

```
SELECT *
FROM instructor
ORDER BY salary ____, name ____;
```

To display the salary from greater to smaller and name in ascending order which of the following options should be used?

- a) Ascending, Descending
- b) Asc, Desc
- c) Desc, Asc
- d) Descending, Ascending

Answer: c

7.

```
SELECT name
FROM instructor
WHERE salary <= 100000 AND salary >= 90000;
```

This query can be replaced by which of the following ?



a)

```
SELECT name
FROM instructor
WHERE salary BETWEEN 90000 AND 100000;
```

b)

```
SELECT name
FROM employee
WHERE salary <= 90000 AND salary >= 100000;
```

c)

```
SELECT name
FROM employee
WHERE salary BETWEEN 90000 AND 100000;
```

d)

```
SELECT name
FROM instructor
WHERE salary BETWEEN 100000 AND 90000;
```

Answer: a

8.

```
SELECT instructor.*
FROM instructor, teaches
WHERE instructor.ID= teaches.ID;
```

This query does which of the following operation?

- a) All attributes of instructor and teaches are selected
- b) All attributes of instructor are selected on the given condition
- c) All attributes of teaches are selected on given condition
- d) Only the some attributes from instructed and teaches are selected

Answer: b

9. In SQL the spaces at the end of the string are removed by \_\_\_\_\_ function.

- a) Upper
- b) String
- c) Trim
- d) Lower

Answer: c

10. \_\_\_\_\_ operator is used for appending two strings.

- a) &
- b) %
- c) ||
- d) \_

Answer: c

1. The union operation is represented by

- a)  $\cap$
- b)  $\cup$
- c) -
- d) \*

Answer: b

2. The intersection operator is used to get the \_\_\_\_\_ tuples.

- a) Different
- b) Common
- c) All
- d) Repeating

Answer: b

3. The union operation automatically \_\_\_\_\_ unlike the select clause.

- a) Adds tuples
- b) Eliminates unique tuples
- c) Adds common tuples
- d) Eliminates duplicate

Answer: d

4. If we want to retain all duplicates, we must write \_\_\_\_\_ in place of union.

- a) Union all
- b) Union some
- c) Intersect all
- d) Intersect some

Answer: a

5.

Check this: [Programming Books](#) | [DBMS Books](#)

```
(SELECT course id
FROM SECTION
WHERE semester = 'Fall' AND YEAR= 2009)
EXCEPT
(SELECT course id
FROM SECTION
WHERE semester = 'Spring' AND YEAR= 2010);
```

This query displays

- a) Only tuples from second part
- b) Only tuples from the first part which has the tuples from second part
- c) Tuples from both the parts
- d) Tuples from first part which do not have second part

Answer: d

6. For like predicate which of the following is true.

- i) % matches zero **OF** more characters.
- ii) \_ matches exactly one **CHARACTER**.

- a) i-only
- b) ii-only
- c) i & ii
- d) None of the mentioned

Answer: c

7. The number of attributes in relation is called as its

- a) Cardinality
- b) Degree
- c) Tuples
- d) Entity

Answer: b

8. \_\_\_\_ clause is an additional filter that is applied to the result.

- a) Select
- b) Group-by
- c) Having
- d) Order by

Answer: c

9. \_\_\_\_\_ joins are SQL server default

- a) Outer
- b) Inner
- c) Equi
- d) None of the mentioned

Answer: b

10. The \_\_\_\_\_ is essentially used to search for patterns in target string.

- a) Like Predicate
- b) Null Predicate
- c) In Predicate
- d) Out Predicate

Answer: a

1. The union operation is represented by

- a)  $\cap$
- b)  $\cup$
- c)  $-$
- d)  $*$

Answer: b

2. The intersection operator is used to get the \_\_\_\_ tuples.

- a) Different
- b) Common
- c) All
- d) Repeating

Answer: b

3. The union operation automatically \_\_\_\_\_ unlike the select clause.

- a) Adds tuples
- b) Eliminates unique tuples
- c) Adds common tuples
- d) Eliminates duplicate

Answer: d

4. If we want to retain all duplicates, we must write \_\_\_\_\_ in place of union.

- a) Union all
- b) Union some
- c) Intersect all
- d) Intersect some

Answer: a.

5.

Check this: [Programming Books](#) | [DBMS Books](#)

```
(SELECT course id
FROM SECTION
WHERE semester = 'Fall' AND YEAR= 2009)
EXCEPT
(SELECT course id
FROM SECTION
WHERE semester = 'Spring' AND YEAR= 2010);
```

This query displays

- a) Only tuples from second part
- b) Only tuples from the first part which has the tuples from second part
- c) Tuples from both the parts
- d) Tuples from first part which do not have second part

Answer: d

6. For like predicate which of the following is true.

```
i) % matches zero OF more characters.  
ii) _ matches exactly one CHARACTER.
```

- a) i-only
- b) ii-only
- c) i & ii
- d) None of the mentioned

Answer: c

7. The number of attributes in relation is called as its

- a) Cardinality
- b) Degree
- c) Tuples
- d) Entity

Answer: b

8. \_\_\_\_ clause is an additional filter that is applied to the result.

- a) Select
- b) Group-by
- c) Having
- d) Order by

Answer: c

9. \_\_\_\_\_ joins are SQL server default

- a) Outer
- b) Inner

- c) Equi
- d) None of the mentioned

Answer: b

10. The \_\_\_\_\_ is essentially used to search for patterns in target string.

- a) Like Predicate
- b) Null Predicate
- c) In Predicate
- d) Out Predicate

Answer: a

1. A \_\_\_\_ indicates an absent value that may exist but be unknown or that may not exist at all.

- a) Empty tuple
- b) New value
- c) Null value
- d) Old value

Answer: c

2. If the attribute phone number is included in the relation all the values need not be entered into the phone number column. This type of entry is given as

- a) 0
- b) -
- c) Null
- d) Empty space

Answer: c

3. The predicate in a where clause can involve Boolean operations such as and. The result of true and unknown is \_\_\_\_\_ false and unknown is \_\_\_\_\_ while unknown and unknown is \_\_\_\_\_

- a) Unknown, unknown, false
- b) True, false, unknown
- c) True, unknown, unknown

d) Unknown, false, unknown

Answer: d

4.

```
SELECT name
FROM instructor
WHERE salary IS NOT NULL;
Selects
```

- a) Tuples with null value
- b) Tuples with no null values
- c) Tuples with any salary
- d) All of the mentioned

Answer: b

5. In an employee table to include the attributes whose value always have some value which of the following constraint must be used?

- a) Null
- b) Not null
- c) Unique
- d) Distinct

Answer: b

6. Using the \_\_\_\_\_ clause retains only one copy of such identical tuples.

- a) Null
- b) Unique
- c) Not null
- d) Distinct

Answer: d

7.

```
CREATE TABLE employee (id INTEGER, name VARCHAR(20), salary NOT NULL);
INSERT INTO employee VALUES (1005, Rach, 0);
INSERT INTO employee VALUES (1007, Ross, );
```



```
INSERT INTO employee VALUES (1002,Joey,335);
```

Some of these insert statements will produce an error. Identify the statement.

- a) Insert into employee values (1005,Rach,0);
- b) Insert into employee values (1002,Joey,335);
- c) Insert into employee values (1007,Ross, );
- d) None of the mentioned

Answer: c

8. The primary key must be

- a) Unique
- b) Not null
- c) Both Unique and Not null
- d) Either Unique or Not null

Answer: c

9. You attempt to query the database with this command:

```
SELECT nvl (100 / quantity, NONE)  
FROM inventory;
```

Why does this statement cause an error when QUANTITY values are null?

- a) The expression attempts to divide by a null value
- b) The data types in the conversion function are incompatible
- c) The character string none should be enclosed in single quotes ( ' ' )
- d) A null value used in an expression cannot be converted to an actual value

Answer: a

10. The result of \_\_\_\_unknown is unknown.

- a) Xor
- b) Or
- c) And
- d) Not

Answer: d

1. Aggregate functions are functions that take a \_\_\_\_\_ as input and return a single value.

- a) Collection of values
- b) Single value
- c) Aggregate value
- d) Both Collection of values & Single value

Answer: a

2.

```
SELECT _____  
FROM instructor  
WHERE dept name= 'Comp. Sci.';
```

Which of the following should be used to find the mean of the salary ?

- a) Mean(salary)
- b) Avg(salary)
- c) Sum(salary)
- d) Count(salary)

Answer: b

3.

Take [Database Management System Tests](#) Now!

```
SELECT COUNT (____ ID)  
FROM teaches  
WHERE semester = 'Spring' AND YEAR = 2010;
```

If we do want to eliminate duplicates, we use the keyword \_\_\_\_\_in the aggregate expression.

- a) Distinct
- b) Count
- c) Avg
- d) Primary key

Answer: a

4. All aggregate functions except \_\_\_\_\_ ignore null values in their input collection.

- a) Count(attribute)
- b) Count(\*)
- c) Avg
- d) Sum

Answer: b

5. A Boolean data type that can take values true, false, and\_\_\_\_\_

- a) 1
- b) 0
- c) Null
- d) Unknown

Answer: d

6. The \_\_\_\_ connective tests for set membership, where the set is a collection of values produced by a select clause. The \_\_\_\_ connective tests for the absence of set membership.

- a) Or, in
- b) Not in, in
- c) In, not in
- d) In, or

Answer: c

7. Which of the following should be used to find all the courses taught in the Fall 2009 semester but not in the Spring 2010 semester .

a)

```
SELECT DISTINCT course id
FROM SECTION
WHERE semester = 'Fall' AND YEAR= 2009 AND
course id NOT IN (SELECT course id
FROM SECTION
WHERE semester = 'Spring' AND YEAR= 2010);
```

b)

```
SELECT DISTINCT course_id
FROM instructor
```

```
WHERE name NOT IN ('Fall', 'Spring');
```

c)

```
(SELECT course id  
FROM SECTION  
WHERE semester = 'Spring' AND YEAR= 2010)
```

d)

```
SELECT COUNT (DISTINCT ID)  
FROM takes  
WHERE (course id, sec id, semester, YEAR) IN (SELECT course id, sec id, semester, YEAR  
FROM teaches  
WHERE teaches.ID= 10101);
```

Answer: a

8. The phrase “greater than at least one” is represented in SQL by \_\_\_\_

- a) < all
- b) < some
- c) > all
- d) > some

Answer: d

9. Which of the following is used to find all courses taught in both the Fall 2009 semester and in the Spring 2010 semester .

a)

```
SELECT course id  
FROM SECTION AS S  
WHERE semester = 'Fall' AND YEAR= 2009 AND  
EXISTS (SELECT *  
FROM SECTION AS T  
WHERE semester = 'Spring' AND YEAR= 2010 AND  
S.course id= T.course id);
```

b)

```
SELECT name  
FROM instructor  
WHERE salary > SOME (SELECT salary
```

```
FROM instructor
WHERE dept name = 'Biology');
```

c)

```
SELECT COUNT (DISTINCT ID)
FROM takes
WHERE (course id, sec id, semester, YEAR) IN (SELECT course id, sec id, semester, YEAR
FROM teaches
WHERE teaches.ID= 10101);
```

d)

```
(SELECT course id
FROM SECTION
WHERE semester = 'Spring' AND YEAR= 2010)
```

Answer: a

10. We can test for the nonexistence of tuples in a subquery by using the \_\_\_\_ construct.

- a) Not exist
- b) Not exists
- c) Exists
- d) Exist

Answer: b

1.

```
SELECT dept_name, ID, avg (salary)
FROM instructor
GROUP BY dept_name;
This statement IS erroneous because
```

- a) Avg(salary) should not be selected
- b) Dept\_id should not be used in group by clause
- c) Misplaced group by clause
- d) Group by clause is not valid in this query

Answer: b

2. SQL applies predicates in the \_\_\_\_\_ clause after groups have been formed, so aggregate functions may be used.

- a) Group by
- b) With
- c) Where
- d) Having

Answer: d

3. Aggregate functions can be used in the select list or the \_\_\_\_\_ clause of a select statement or subquery. They cannot be used in a \_\_\_\_\_ clause.

- a) Where, having
- b) Having, where
- c) Group by, having
- d) Group by, where

Answer: b

4. The \_\_\_\_\_ keyword is used to access attributes of preceding tables or subqueries in the from clause.

- a) In
- b) Lateral
- c) Having
- d) With

Answer: b

5. Which of the following creates a temporary relation for the query on which it is defined?

- a) With
- b) From
- c) Where
- d) Select

Answer: a

6.

```
WITH max_budget (VALUE) AS  
(SELECT MAX(budget)  
FROM department)  
SELECT budget  
FROM department, max_budget  
WHERE department.budget = MAX budget.value;
```

In the query given above which one of the following is a temporary relation?

- a) Budget
- b) Department
- c) Value
- d) Max\_budget

Answer: d

7. Subqueries cannot:

- a) Use group by or group functions
- b) Retrieve data from a table different from the one in the outer query
- c) Join tables
- d) Appear in select, update, delete, insert statements.

Answer: c

8. Which of the following is not an aggregate function?

- a) Avg
- b) Sum
- c) With
- d) Min

Answer: c

9. The EXISTS keyword will be true if:

- a) Any row in the subquery meets the condition only
- b) All rows in the subquery fail the condition only
- c) Both of these two conditions are met
- d) Neither of these two conditions is met

Answer: a

10. How can you find rows that do not match some specified condition?

- a) EXISTS
- b) Double use of NOT EXISTS
- c) NOT EXISTS
- d) None of the mentioned

Answer: b

1. A Delete command operates on \_\_\_\_\_ relation.

- a) One
- b) Two
- c) Several
- d) Null

Answer: a

2.

Delete from r where P;

The above command

- a) Deletes a particular tuple from the relation
- b) Deletes the relation
- c) Clears all entries from the relation
- d) All of the mentioned

Answer: a

3. Which one of the following deletes all the entries but keeps the structure of the relation.

- a) Delete from r where P;
- b) Delete from instructor where dept name= 'Finance';
- c) Delete from instructor where salary between 13000 and 15000;
- d) Delete from instructor;

Answer: d



4. Which of the following is used to insert a tuple from another relation?

a)

Take [Database Management System Tests](#) Now!

```
INSERT INTO course (course id, title, dept name, credits)
VALUES ('CS-437', 'DATABASE Systems', 'Comp. Sci.', 4);
```

b)

```
INSERT INTO instructor
SELECT ID, name, dept name, 18000
FROM student
WHERE dept name = 'Music' AND tot cred > 144;
```

c)

```
INSERT INTO course VALUES ('CS-437', 'DATABASE Systems', 'Comp. Sci.', 4);
```

d) Not possible

Answer: b

5. Which of the following deletes all tuples in the instructor relation for those instructors associated with a department located in the Watson building which is in department relation.

a)

```
DELETE FROM instructor
WHERE dept_name IN 'Watson';
```

b)

```
DELETE FROM department
WHERE building='Watson';
```

c)

```
DELETE FROM instructor
WHERE dept_name IN (SELECT dept name
                    FROM department
                    WHERE building = 'Watson');
```

d) None of the mentioned

Answer: c.

6.

```
UPDATE instructor
  _____ salary= salary * 1.05;
```

Fill in with correct keyword to update the instructor relation.

- a) Where
- b) Set
- c) In
- d) Select

Answer: b

7. \_\_\_\_\_ are useful in SQL update statements, where they can be used in the set clause.

- a) Multiple queries
- b) Sub queries
- c) Update
- d) Scalar subqueries

Answer: d

8. The problem of ordering the update in multiple updates is avoided using

- a) Set
- b) Where
- c) Case
- d) When

Answer: c

9. Which of the following is the correct format for case statements.

a)

```
CASE
WHEN pred1 ... result1
WHEN pred2 ... result2
. . .
WHEN predn ... resultn
ELSE result0
END
```

b)

```

CASE
WHEN pred1 THEN result1
WHEN pred2 THEN result2
. . .
WHEN predn THEN resultn
ELSE result0
END

```

c)

```

CASE
WHEN pred1 THEN result1
WHEN pred2 THEN result2
. . .
WHEN predn THEN resultn
ELSE result0

```

d) All of the mentioned

Answer: b

10. Which of the following relation updates all instructors with salary over \$100,000 receive a 3 percent raise, whereas all others receive a 5 percent raise.

a)

```

UPDATE instructor
SET salary = salary * 1.03
WHERE salary > 100000;
UPDATE instructor
SET salary = salary * 1.05
WHERE salary <= 100000;

```

b)

```

UPDATE instructor
SET salary = salary * 1.05
WHERE salary < (SELECT avg (salary)
FROM instructor);

```

c)

```

UPDATE instructor
SET salary = CASE
WHEN salary <= 100000 THEN salary * 1.03
ELSE salary * 1.05
END

```

d) None of the mentioned

Answer: a

1. The\_\_\_\_condition allows a general predicate over the relations being joined.

- a) On
- b) Using
- c) Set
- d) Where

Answer: a

2. Which of the join operations do not preserve non matched tuples?

- a) Left outer join
- b) Right outer join
- c) Inner join
- d) Natural join

Answer: c

3.

**SELECT \***

**FROM** student **JOIN** takes **USING** (ID);

The above query is equivalent to

a)

**Check this: [Programming Books](#) | [RDBMS MCQ](#)**

**SELECT \***

**FROM** student **INNER JOIN** takes **USING** (ID);

b)

**SELECT \***

**FROM** student **OUTER JOIN** takes **USING** (ID);

c)

**SELECT \***

```
FROM student LEFT OUTER JOIN takes USING (ID);
```

d) None of the mentioned

Answer: a

4. What type of join is needed when you wish to include rows that do not have matching values?

- a) Equi-join
- b) Natural join
- c) Outer join
- d) All of the mentioned

Answer: c

5. How many tables may be included with a join?

- a) One
- b) Two
- c) Three
- d) All of the mentioned

Answer: d

6. Which are the join types in join condition:

- a) Cross join
- b) Natural join
- c) Join with USING clause
- d) All of the mentioned

Answer: d

7. How many join types in join condition:

- a) 2
- b) 3
- c) 4
- d) 5

Answer: d

8. Which join refers to join records from the right table that have no matching key in the left table are include in the result set:

- a) Left outer join
- b) Right outer join
- c) Full outer join
- d) Half outer join

Answer: b

9. The operation which is not considered a basic operation of relational algebra is

- a) Join
- b) Selection
- c) Union
- d) Cross product

Answer: a

10. In SQL the statement select \* from R, S is equivalent to

- a) Select \* from R natural join S
- b) Select \* from R cross join S
- c) Select \* from R union join S
- d) Select \* from R inner join S

Answer: b

1. Which of the following creates a virtual relation for storing the query?

- a) Function
- b) View
- c) Procedure
- d) None of the mentioned

Answer: b

2. Which of the following is the syntax for views where v is view name?

- a) Create view v as "query name";
- b) Create "query expression" as view;
- c) Create view v as "query expression";

d) Create view "query expression";

Answer: c

3.

```
SELECT course_id  
FROM physics_fall_2009  
WHERE building= 'Watson';
```

Here the tuples are selected from the view. Which one denotes the view.

- a) Course\_id
- b) Watson
- c) Building
- d) physics\_fall\_2009

Answer: c

4. Materialised views make sure that

- a) View definition is kept stable
- b) View definition is kept up-to-date
- c) View definition is verified for error
- d) View is deleted after specified time

Answer: b

5. Updating the value of the view

- a) Will affect the relation from which it is defined
- b) Will not change the view definition
- c) Will not affect the relation from which it is defined
- d) Cannot determine

Answer: a

6. SQL view is said to be updatable (that is, inserts, updates or deletes can be applied on the view) if which of the following conditions are satisfied by the query defining the view?

- a) The from clause has only one database relation
- b) The query does not have a group by or having clause

- c) The select clause contains only attribute names of the relation and does not have any expressions, aggregates, or distinct specification  
d) All of the mentioned

Answer: d

7. Which of the following is used at the end of the view to reject the tuples which do not satisfy the condition in where clause?

- a) With  
b) Check  
c) With check  
d) All of the mentioned

Answer: c

8. Consider the two relations instructor and department  
Instructor:

ID	Name	Dept_name
1001	Ted	Finance
1002	Bob	Music
1003	Ron	Physics

Department:

Dept_name	Building	Budg
Biology	Watson	400
Chemistry	Painter	300
Music	Taylor	500

Which of the following is used to create view for these relations together?

a)

```
CREATE VIEW instructor_info AS
SELECT ID, name, building
FROM instructor, department
```



```
WHERE instructor.dept name= department.dept name;
```

b)

```
CREATE VIEW instructor_info  
SELECT ID, name, building  
FROM instructor, department;
```

c)

```
CREATE VIEW instructor_info AS  
SELECT ID, name, building  
FROM instructor;
```

d)

```
CREATE VIEW instructor_info AS  
SELECT ID, name, building  
FROM department;
```

Answer: a

9. For the view Create view instructor\_info as

```
SELECT ID, name, building  
FROM instructor, department  
WHERE instructor.dept name= department.dept name;
```

If we insert tuple into the view as insert into instructor\_info values ('69987', 'White', 'Taylor');

What will be the values of the other attributes in instructor and department relations?

- a) Default value
- b) Null
- c) Error statement
- d) 0

Answer: b

10.

```
CREATE VIEW faculty AS  
SELECT ID, name, dept name  
FROM instructor;
```

Find the error in this query.

- a) Instructor
- b) Select
- c) View ...as
- d) None of the mentioned

Answer: d

1. A \_\_\_\_\_ consists of a sequence of query and/or update statements.

- a) Transaction
- b) Commit
- c) Rollback
- d) Flashback

Answer: a

2. Which of the following makes the transaction permanent in the database?

- a) View
- b) Commit
- c) Rollback
- d) Flashback

Answer: b

3. In order to undo the work of transaction after last commit which one should be used?

- a) View
- b) Commit
- c) Rollback
- d) Flashback

Answer: c

4. Consider the following action:

```
TRANSACTION.....  
Commit;  
ROLLBACK;
```

What does Rollback do?

- a) Undoes the transactions before commit
- b) Clears all transactions
- c) Redoes the transactions before commit
- d) No action

Answer: d

5. In case of any shut down during transaction before commit which of the following statement is done automatically?

- a) View
- b) Commit
- c) Rollback
- d) Flashback

Answer: c

6. In order to maintain the consistency during transactions, database provides

- a) Commit
- b) Atomic
- c) Flashback
- d) Retain

Answer: b

7. Transaction processing is associated with everything below except

- a) Conforming an action or triggering a response
- b) Producing detail summary or exception report
- c) Recording a business activity
- d) Maintaining a data

Answer: a

8. A transaction completes its execution is said to be

- a) Committed
- b) Aborted
- c) Rolled back

d) Failed

Answer: a

9. Which of the following is used to get back all the transactions back after rollback?

- a) Commit
- b) Rollback
- c) Flashback
- d) Redo

Answer: c

10. \_\_\_\_\_ will undo all statements up to commit?

- a) Transaction
- b) Flashback
- c) Rollback
- d) Abort

Answer: c

1. To include integrity constraint in an existing relation use :

- a) Create table
- b) Modify table
- c) Alter table
- d) Drop table

Answer: c

2. Which of the following is not an integrity constraint?

- a) Not null
- b) Positive
- c) Unique
- d) Check 'predicate'

Answer: b

3.

```
CREATE TABLE Employee(Emp_id NUMERIC NOT NULL, Name VARCHAR(20), dept_name VARCHAR(20),  
Salary NUMERIC UNIQUE(Emp_id,Name));  
INSERT INTO Employee VALUES(1002, Ross, CSE, 10000)  
INSERT INTO Employee VALUES(1006,Ted,Finance, );  
INSERT INTO Employee VALUES(1002,Rita,Sales,20000);
```

What will be the result of the query?

- a) All statements executed
- b) Error in create statement
- c) Error in insert into Employee values(1006,Ted,Finance, );
- d) Error in insert into Employee values(1008,Ross,Sales,20000);

Answer: d

4.

```
CREATE TABLE Manager(ID NUMERIC,Name VARCHAR(20),budget NUMERIC,Details VARCHAR(30));
```

Inorder to ensure that the value of budget is non-negative which of the following should be used?

- a) Check(budget>0)
- b) Check(budget<0)
- c) Alter(budget>0)
- d) Alter(budget<0)

Answer: a

5. Foreign key is the one in which the \_\_\_\_\_ of one relation is referenced in another relation.

- a) Foreign key
- b) Primary key
- c) References
- d) Check constraint

Answer: b

6.

```
CREATE TABLE course
( . . .
FOREIGN KEY (dept name) REFERENCES department
. . . );
```

Which of the following is used to delete the entries in the referenced table when the tuple is deleted in course table?

- a) Delete
- b) Delete cascade
- c) Set null
- d) All of the mentioned

Answer: b

7. Domain constraints, functional dependency and referential integrity are special forms of \_\_\_\_\_

- a) Foreign key
- b) Primary key
- c) Assertion
- d) Referential constraint

Answer: c

8. Which of the following is the right syntax for the assertion?

- a) Create assertion 'assertion-name' check 'predicate';
- b) Create assertion check 'predicate' 'assertion-name';
- c) Create assertions 'predicates';
- d) All of the mentioned

Answer: a

9. Data integrity constraints are used to:

- a) Control who is allowed access to the data
- b) Ensure that duplicate records are not entered into the table
- c) Improve the quality of data entered for a specific property (i.e., table column)
- d) Prevent users from changing the values stored in the table

Answer: c

10. Which of the following can be addressed by enforcing a referential integrity constraint?

- a) All phone numbers must include the area code
- b) Certain fields are required (such as the email address, or phone number) before the record is accepted
- c) Information on the customer must be known before anything can be sold to that customer
- d) When entering an order quantity, the user must input a number and not some text (i.e., 12 rather than 'a dozen')

Answer: c

1. Dates must be specified in the format

- a) mm/dd/yy
- b) yyyy/mm/dd
- c) dd/mm/yy
- d) yy/dd/mm

Answer: b

2. A \_\_\_\_\_ on an attribute of a relation is a data structure that allows the database system to find those tuples in the relation that have a specified value for that attribute efficiently, without scanning through all the tuples of the relation.

- a) Index
- b) Reference
- c) Assertion
- d) Timestamp

Answer: a

3.

Create index studentID\_index on student(ID);

Here which one denotes the relation for which index is created?

- a) StudentID\_index
- b) ID

- c) StudentID
- d) Student

Answer: d

4. Which of the following is used to store movie and image files?

- a) Clob
- b) Blob
- c) Binary
- d) Image

Answer: b

5. The user defined data type can be created using

- a) Create datatype
- b) Create data
- c) Create definetype
- d) Create type

Answer: d

6. Values of one type can be converted to another domain using which of the following?

- a) Cast
- b) Drop type
- c) Alter type
- d) Convert

Answer: a

7.

```
CREATE DOMAIN YearlySalary NUMERIC(8,2)
CONSTRAINT salary VALUE test _____;
```

In order to ensure that an instructor's salary domain allows only values greater than a specified value use:

- a) Value>=30000.00
- b) Not null;



- c) Check(value >= 29000.00);
- d) Check(value)

Answer: c

8. Which of the following closely resembles Create view?

- a) Create table . . .like
- b) Create table . . . as
- c) With data
- d) Create view as

Answer: b

9. In contemporary databases, the top level of the hierarchy consists of \_\_\_\_\_ each of which can contain \_\_\_\_\_

- a) Catalogs, schemas
- b) Schemas, catalogs
- c) Environment, schemas
- d) Schemas, Environment

Answer: a

10. Which of the following statements creates a new table temp instructor that has the same schema as an instructor.

- a) create table temp\_instructor;
- b) Create table temp\_instructor like instructor;
- c) Create Table as temp\_instructor;
- d) Create table like temp\_instructor;

Answer: b

1. The database administrator who authorizes all the new users, modifies the database and takes grants privilege is

- a) Super user
- b) Administrator
- c) Operator of operating system
- d) All of the mentioned

Answer: d

2. Which of the following is a basic form of grant statement?

a)

```
GRANT 'privilege list'  
ON 'relation name or view name'  
TO 'user/role list';
```

b)

**Note: Join free Sanfoundry classes at [Telegram](#) or [Youtube](#)**

```
GRANT 'privilege list'  
ON 'user/role list'  
TO 'relation name or view name';
```

c)

**Take Database Management System Mock Tests - Chapterwise!**

**Start the Test Now: [Chapter 1](#), [2](#), [3](#), [4](#), [5](#), [6](#), [7](#), [8](#), [9](#), [10](#)**

```
GRANT 'privilege list'  
TO 'user/role list'
```

d)

```
GRANT 'privilege list'  
ON 'relation name or view name'  
ON 'user/role list';
```

Answer: a

3. Which of the following is used to provide privilege to only a particular attribute?

- a) Grant select on employee to Amit
- b) Grant update(budget) on department to Raj
- c) Grant update(budget,salary,Rate) on department to Raj
- d) Grant delete to Amit

Answer: b

4. Which of the following statement is used to remove the privilege from the user Amir?

- a) Remove update on department from Amir
- b) Revoke update on employee from Amir
- c) Delete select on department from Raj
- d) Grant update on employee from Amir

Answer: b

5. Which of the following is used to provide delete authorization to instructor?

a)

```
CREATE ROLE instructor ;  
GRANT DELETE TO instructor;
```

b)

```
CREATE ROLE instructor;  
GRANT SELECT ON takes  
TO instructor;
```

c)

```
CREATE ROLE instructor;  
GRANT DELETE ON takes  
TO instructor;
```

d) All of the mentioned

Answer: c

6. Which of the following is true regarding views?

- a) The user who creates a view cannot be given update authorization on a view without having update authorization on the relations used to define the view
- b) The user who creates a view cannot be given update authorization on a view without having update authorization on the relations used to define the view
- c) If a user creates a view on which no authorization can be granted, the system will allow the view creation request
- d) A user who creates a view receives all privileges on that view

Answer: c

7. If we wish to grant a privilege and to allow the recipient to pass the privilege on to other users, we append the \_\_\_\_\_ clause to the appropriate grant command.

- a) With grant
- b) Grant user
- c) Grant pass privilege
- d) With grant option

Answer: d

8. In authorization graph, if DBA provides authorization to u1 which in turn gives to u2 which of the following is correct?

- a) If DBA revokes authorization from u1 then u2 authorization is also revoked
- b) If u1 revokes authorization from u2 then u2 authorization is revoked
- c) If DBA & u1 revokes authorization from u1 then u2 authorization is also revoked
- d) If u2 revokes authorization then u1 authorization is revoked

Answer: c

9. Which of the following is used to avoid cascading of authorizations from the user?

- a) Granted by current role
- b) Revoke select on department from Amit, Satoshi restrict;
- c) Revoke grant option for select on department from Amit;
- d) Revoke select on department from Amit, Satoshi cascade;

Answer: b

10. The granting and revoking of roles by the user may cause some confusions when that user role is revoked. To overcome the above situation

- a) The privilege must be granted only by roles
- b) The privilege is granted by roles and users
- c) The user role cannot be removed once given
- d) By restricting the user access to the roles

Answer: a

1. Which of the following is used to access the database server at the time of executing the program and get the data from the server accordingly?

- a) Embedded SQL
- b) Dynamic SQL
- c) SQL declarations
- d) SQL data analysis

Answer: b

2. Which of the following header must be included in java program to establish database connectivity using JDBC ?

- a) Import java.sql.\*;
- b) Import java.sql.odbc.jdbc.\*;
- c) Import java.jdbc.\*;
- d) Import java.sql.jdbc.\*;

Answer: a

3. DriverManager.getConnection(\_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_)

What are the two parameters that are included?

- a) URL or machine name where server runs, Password, User ID
- b) URL or machine name where server runs, User ID, Password
- c) User ID, Password, URL or machine name where server runs
- d) Password, URL or machine name where server runs, User ID

Answer: b

4. Which of the following invokes functions in sql?

- a) Prepared Statements
- b) Connection statement
- c) Callable statements
- d) All of the mentioned

Answer: c.

5. Which of the following function is used to find the column count of the particular result set?

- a) getMetaData()
- b) Metadata()
- c) getColumn()

d) get Count()

Answer: a

6. Which of the following is a following statement is a prepared statements?

- a) Insert into department values(?,?,?)
- b) Insert into department values(x,x,x)
- c) SQLSetConnectOption(conn, SQL AUTOCOMMIT, 0)
- d) SQLTransact(conn, SQL ROLLBACK)

Answer: a

7. Which of the following is used as the embedded SQL in COBOL?

- a) EXEC SQL <embedded SQL statement >;
- b) EXEC SQL <embedded SQL statement > END-EXEC
- c) EXEC SQL <embedded SQL statement >
- d) EXEC SQL <embedded SQL statement > END EXEC;

Answer: b

8. Which of the following is used to distinguish the variables in SQL from the host language variables?

- a) .
- b) –
- c) :
- d) ,

Answer: c

9. The update statement can be executed in host language using

- a) EXEC SQL update c;
- b) EXEC SQL update c into :si, :sn;
- c)

```
EXEC SQL
UPDATE instructor
SET salary = salary + 100
WHERE CURRENT OF c;
```

d) EXEC SQL update END-SQL

Answer: c

10. Which of the following is used to access large objects from a database ?

- a) setBlob()
- b) getBlob()
- c) getClob()
- d) all of the mentioned

Answer: d

1.

```
Create function dept count(dept_name varchar(20))
```

```
begin
```

```
declare d count integer;
```

```
select count(*) into d count
```

```
from instructor
```

```
where instructor.dept_name= dept_name
```

```
return d count;
```

```
end
```

Find the error in the the above statement.

- a) Return type missing
- b) Dept\_name is mismatched
- c) Reference relation is not mentioned
- d) All of the mentioned

Answer: a

2. For the function created in Question 1, which of the following is a proper select statement ?

a)

```
SELECT dept name, budget
FROM instructor
WHERE dept COUNT() > 12;
```

b

```
SELECT dept name, budget
FROM instructor
WHERE dept COUNT(dept name) > 12;
```

c)

```
SELECT dept name, budget
WHERE dept COUNT(dept name) > 12;
```

d)

```
SELECT dept name, budget
FROM instructor
WHERE dept COUNT(budget) > 12;
```

Answer: b

3. Which of the following is used to input the entry and give the result in a variable in a procedure?

- a) Put and get
- b) Get and put
- c) Out and In
- d) In and out

Answer: d

4.

Create procedure dept\_count proc(in dept name varchar(20),

out d count integer)

begin



```
select count(*) into d_count
```

```
from instructor
```

```
where instructor.dept_name = dept_count_proc.dept_name
```

```
end
```

Which of the following is used to call the procedure given above ?

a)

```
Declare d_count integer;
```

b)

```
Declare d_count integer;
```

```
call dept_count_proc('Physics', d_count);
```

c)

```
Declare d_count integer;
```

```
call dept_count_proc('Physics');
```

d)

```
Declare d_count;
```

```
call dept_count_proc('Physics', d_count);
```

Answer: b

5. The format for compound statement is

a) Begin ..... end

b) Begin atomic..... end

c) Begin ..... repeat

d) Both Begin ..... end and Begin atomic..... end

Answer: d

6.

```
Repeat  
sequence of statements;
```

```
end repeat
```

Fill in the correct option :

- a) While Condition
- b) Until variable
- c) Until boolean expression
- d) Until 0

Answer: c

7. Which of the following is the correct format for if statement?

a)

If boolean expression

then statement or compound statement

elseif boolean expression

then statement or compound statement

else statement or compound statement

end if

b)

If boolean expression

then statement or compound statement

elseif boolean expression

then statement or compound statement

else statement or compound statement

end if

c)

If boolean expression

then statement or compound statement

elif boolean expression

then statement or compound statement

else statement or compound statement

end if

d)

If boolean expression

then statement or compound statement

else

statement or compound statement

else statement or compound statement

end if

Answer: a

8. A stored procedure in SQL is a\_\_\_\_\_

a) Block of functions

b) Group of Transact-SQL statements compiled into a single execution plan.

c) Group of distinct SQL statements.

d) None of the mentioned

Answer: b

9. Temporary stored procedures are stored in \_\_\_\_\_ database.

- a) Master
- b) Model
- c) User specific
- d) Tempdb

Answer: d

10. Declare out of classroom seats condition

```
DECLARE exit handler FOR OUT OF classroom seats  
BEGIN  
SEQUENCE OF statements  
END
```

The above statements are used for

- a) Calling procedures
- b) Handling Exception
- c) Handling procedures
- d) All of the mentioned

Answer: b

1. A \_\_\_\_\_ is a special kind of a store procedure that executes in response to certain action on the table like insertion, deletion or updation of data.

- a) Procedures
- b) Triggers
- c) Functions
- d) None of the mentioned

Answer: b

2. Triggers are supported in

- a) Delete
- b) Update
- c) Views
- d) All of the mentioned

Answer: c

3. The CREATE TRIGGER statement is used to create the trigger. THE \_\_\_\_\_ clause specifies the table name on which the trigger is to be attached. The \_\_\_\_\_ specifies that this is an AFTER INSERT trigger.

- a) for insert, on
- b) On, for insert
- c) For, insert
- d) None of the mentioned

Answer: b

4. What are the after triggers?

- a) Triggers generated after a particular operation
- b) These triggers run after an insert, update or delete on a table
- c) These triggers run after an insert, views, update or delete on a table
- d) All of the mentioned

Answer: b

5. The variables in the triggers are declared using

- a) -
- b) @
- c) /
- d) /@

Answer: b

6. The default extension for an Oracle SQL\*Plus file is:

- a) .txt
- b) .pls
- c) .ora
- d) .sql

Answer: d

7. Which of the following is NOT an Oracle-supported trigger?

- a) BEFORE
- b) DURING
- c) AFTER
- d) INSTEAD OF

Answer: b

8. What are the different in triggers?

- a) Define, Create
- b) Drop, Comment
- c) Insert, Update, Delete
- d) All of the mentioned

Answer: c

9. Triggers \_\_\_\_\_ enabled or disabled

- a) Can be
- b) Cannot be
- c) Ought to be
- d) Always

Answer: a

10. Which prefixes are available to Oracle triggers?

- a) : new only
- b) : old only
- c) Both :new and : old
- d) Neither :new nor : old

Answer: c

1. Any recursive view must be defined as the union of two subqueries: a \_\_\_\_\_ query that is nonrecursive and a \_\_\_\_\_ query.

- a) Base, recursive
- b) Recursive, Base
- c) Base, Redundant

d) View, Base

Answer: a

2. Ranking of queries is done by which of the following?

- a) Group by
- b) Order by
- c) Having
- d) Both Group by and Order by

Answer: b

3. In rank() function if one value is shared by two tuples then

- a) The rank order continues as counting numbers
- b) The rank order continues by leaving one rank in the middle
- c) The user specifies the order
- d) The order does not change

Answer: b

4. The \_\_\_\_\_ function that does not create gaps in the ordering.

- a) Intense\_rank()
- b) Continue\_rank()
- c) Default\_rank()
- d) Dense\_rank()

Answer: d

5.

```
SELECT ID, GPA
FROM student_grades
ORDER BY GPA
_____;
```

In order to give only 10 rank on the whole we should use

- a) Limit 10
- b) Upto 10
- c) Only 10

d) Max 10

Answer: a

6. If there are  $n$  tuples in the partition and the rank of the tuple is  $r$ , then its \_\_\_\_\_ is defined as  $(r - 1)/(n - 1)$ .

- a) Ntil()
- b) Cum\_rank
- c) Percent\_rank
- d) rank()

Answer: c.

7. Inorder to simplify the null value confusion in the rank function we can specify

- a) Not Null
- b) Nulls last
- c) Nulls first
- d) Either Nulls last or first

Answer: d

8. Suppose we are given a view tot credits (year, num credits) giving the total number of credits taken by students in each year. The query that computes averages over the 3 preceding tuples in the specified sort order is

a)

```
SELECT YEAR, avg(num credits)
OVER (ORDER BY YEAR ROWS 3 preceding)
AS avg total credits
FROM tot credits;
```

b)

```
SELECT YEAR, avg(num credits)
OVER (ORDER BY YEAR ROWS 3 unbounded preceding)
AS avg total credits
FROM tot credits;
```

- c) All of the mentioned
- d) None of the mentioned

Answer: a



9. The functions which construct histograms and use buckets for ranking is

- a) Rank()
- b) Newtil()
- c) Ntil()
- d) None of the mentioned

Answer: c

10. The command \_\_\_\_\_ such tables are available only within the transaction executing the query and are dropped when the transaction finishes.

- a) Create table
- b) Create temporary table
- c) Create view
- d) Create label view

Answer: b

1. OLAP stands for

- a) Online analytical processing
- b) Online analysis processing
- c) Online transaction processing
- d) Online aggregate processing

Answer: a

2. Data that can be modeled as dimension attributes and measure attributes are called \_\_\_\_\_ data.

- a) Multidimensional
- b) Singledimensional
- c) Measured
- d) Dimensional

Answer: a

3. The generalization of cross-tab which is represented visually is \_\_\_\_\_ which is also called as data cube.

- a) Two dimensional cube
- b) Multidimensional cube
- c) N-dimensional cube
- d) Cuboid

Answer: a

4. The process of viewing the cross-tab (Single dimensional) with a fixed value of one attribute is

- a) Slicing
- b) Dicing
- c) Pivoting
- d) Both Slicing and Dicing

Answer: a

5. The operation of moving from finer-granularity data to a coarser granularity (by means of aggregation) is called a \_\_\_\_\_

- a) Rollup
- b) Drill down
- c) Dicing
- d) Pivoting

Answer: a

6. In SQL the cross-tabs are created using

- a) Slice
- b) Dice
- c) Pivot
- d) All of the mentioned

Answer: a

7.

```
{ (item name, color, clothes size), (item name, color), (item name, clothes size), (color, clothes size), (item name), (color), (clothes size), () }
```

This can be achieved by using which of the following ?

- a) group by rollup
- b) group by cubic
- c) group by
- d) none of the mentioned

Answer: d

8. What do data warehouses support?

- a) OLAP
- b) OLTP
- c) OLAP and OLTP
- d) Operational databases

Answer: a

9.

```
SELECT item name, color, clothes SIZE, SUM(quantity)
FROM sales
GROUP BY rollup(item name, color, clothes SIZE);
```

How many grouping is possible in this rollup?

- a) 8
- b) 4
- c) 2
- d) 1

Answer: b

10. Which one of the following is the right syntax for DECODE?

- a) DECODE (search, expression, result [, search, result]... [, default])
- b) DECODE (expression, result [, search, result]... [, default], search)
- c) DECODE (search, result [, search, result]... [, default], expression)
- d) DECODE (expression, search, result [, search, result]... [, default])

Answer: d

1. In SQL, which clause is used to filter rows based on a specified condition?

- a) SELECT
- b) WHERE
- c) FROM
- d) HAVING

**\*\*Answer:\*\*** b) WHERE

2. What does SQL stand for?

- a) Structured Query Language
- b) Simple Text Query Language
- c) Sequential Table Query Language
- d) Standardized Transaction Query Language

**\*\*Answer:\*\*** a) Structured Query Language

3. Which SQL statement is used to modify existing records in a table?

- a) INSERT
- b) DELETE
- c) UPDATE
- d) ALTER

**\*\*Answer:\*\*** c) UPDATE

4. What SQL clause is used to sort the result set in ascending or descending order?

- a) SORT BY
- b) ORDER BY
- c) GROUP BY
- d) ARRANGE BY

**\*\*Answer:\*\*** b) ORDER BY

5. In SQL, which aggregate function returns the number of rows in a result set?

- a) SUM
- b) AVG
- c) COUNT
- d) MAX

**\*\*Answer:\*\*** c) COUNT

6. What SQL command is used to remove a table from a database?

- a) DELETE TABLE
- b) DROP TABLE
- c) REMOVE TABLE
- d) ERASE TABLE

**\*\*Answer:\*\*** b) DROP TABLE

7. Which SQL statement is used to retrieve data from one or more tables?

- a) EXTRACT
- b) PULL

- c) SELECT
- d) FETCH

**\*\*Answer:\*\*** c) SELECT

8. In SQL, what is the purpose of the "GROUP BY" clause?

- a) To filter rows based on a specified condition.
- b) To join multiple tables.
- c) To create a new table.
- d) To group rows with identical values into summary rows.

**\*\*Answer:\*\*** d) To group rows with identical values into summary rows.

9. What SQL statement is used to add new rows to a table?

- a) INSERT
- b) ADD
- c) APPEND
- d) CREATE

**\*\*Answer:\*\*** a) INSERT

10. In SQL, what is the purpose of the "HAVING" clause?

- a) To filter rows based on a specified condition.
- b) To group rows with identical values into summary rows.
- c) To filter rows after grouping by using aggregate functions.
- d) To sort the result set in ascending or descending order.

**\*\*Answer:\*\*** c) To filter rows after grouping by using aggregate functions.

11. In SQL, what does the "LIKE" operator do?

- a) Compares two values for equality
- b) Filters rows based on a specified condition
- c) Searches for a specified pattern in a string
- d) Calculates the average value in a column

**\*\*Answer:\*\*** c) Searches for a specified pattern in a string

12. Which SQL clause is used to retrieve unique values from a column?

- a) UNIQUE
- b) DISTINCT
- c) UNIQUEVALUES
- d) FILTER

**\*\*Answer:\*\*** b) DISTINCT

13. In SQL, what is the purpose of the "INNER JOIN" clause?

- a) To return all rows from both tables with NULL values where there is no match
- b) To combine rows from two tables based on a specified condition
- c) To filter rows based on a specified condition
- d) To order the result set in ascending or descending order

**\*\*Answer:\*\*** b) To combine rows from two tables based on a specified condition

14. What SQL command is used to create a new table?

- a) BUILD TABLE
- b) ADD TABLE
- c) CREATE TABLE
- d) DEFINE TABLE

**\*\*Answer:\*\*** c) CREATE TABLE

15. In SQL, what does the "AND" operator do when used in a WHERE clause?

- a) Combines two or more conditions, and all conditions must be true
- b) Combines two or more conditions, and at least one condition must be true
- c) Combines two or more columns into one
- d) Orders the result set based on multiple columns

**\*\*Answer:\*\*** a) Combines two or more conditions, and all conditions must be true

16. What SQL command is used to remove data from a table?

- a) ERASE
- b) DELETE
- c) REMOVE
- d) WASTE

**\*\*Answer:\*\*** b) DELETE

17. In SQL, which operator is used to retrieve data from one or more tables based on a related column?



- a) MATCH
- b) JOIN
- c) CONNECT
- d) RELATE

**\*\*Answer:\*\*** b) JOIN

18. What SQL statement is used to change the structure of an existing table?

- a) MODIFY TABLE
- b) UPDATE TABLE
- c) ALTER TABLE
- d) ADJUST TABLE

**\*\*Answer:\*\*** c) ALTER TABLE

19. In SQL, what is the purpose of the "AS" keyword?

- a) To create a new column in a table
- b) To assign a value to a variable
- c) To rename a column or table
- d) To filter rows based on a specified condition

**\*\*Answer:\*\*** c) To rename a column or table

20. What SQL clause is used to restrict the number of rows returned by a query?

- a) LIMIT

- b) TOP
- c) ROWNUM
- d) RESTRICT

**\*\*Answer:\*\*** a) LIMIT

21. In SQL, what is the purpose of the "GROUP BY" clause?

- a) To filter rows based on a specified condition.
- b) To join multiple tables.
- c) To create a new table.
- d) To group rows with identical values into summary rows.

**\*\*Answer:\*\*** d) To group rows with identical values into summary rows.

22. What SQL statement is used to add new rows to a table?

- a) INSERT
- b) ADD
- c) APPEND
- d) CREATE

**\*\*Answer:\*\*** a) INSERT

23. In SQL, what is the purpose of the "HAVING" clause?

- a) To filter rows based on a specified condition.
- b) To group rows with identical values into summary rows.
- c) To filter rows after grouping by using aggregate functions.

d) To sort the result set in ascending or descending order.

**\*\*Answer:\*\*** c) To filter rows after grouping by using aggregate functions.

24. What SQL clause is used to retrieve unique values from a column?

a) UNIQUE

b) DISTINCT

c) UNIQUEVALUES

d) FILTER

**\*\*Answer:\*\*** b) DISTINCT

25. In SQL, what does the "LIKE" operator do?

a) Compares two values for equality

b) Filters rows based on a specified condition

c) Searches for a specified pattern in a string

d) Calculates the average value in a column

**\*\*Answer:\*\*** c) Searches for a specified pattern in a string

26. In SQL, what does the "ORDER BY" clause allow you to do?

a) Group rows with identical values.

b) Filter rows based on a specified condition.

c) Sort the result set in ascending or descending order.

d) Combine rows from multiple tables.

**\*\*Answer:\*\*** c) Sort the result set in ascending or descending order.

27. What SQL statement is used to create a copy of an existing table?

- a) COPY TABLE
- b) CLONE TABLE
- c) DUPLICATE TABLE
- d) CREATE TABLE AS

**\*\*Answer:\*\*** d) CREATE TABLE AS

28. In SQL, what is the purpose of the "LEFT JOIN" clause?

- a) To return all rows from both tables with NULL values where there is no match.
- b) To combine rows from two tables based on a specified condition.
- c) To filter rows based on a specified condition.
- d) To order the result set in ascending or descending order.

**\*\*Answer:\*\*** a) To return all rows from both tables with NULL values where there is no match.

29. What SQL command is used to remove data from a table?

- a) ERASE
- b) DELETE
- c) REMOVE
- d) WASTE

**\*\*Answer:\*\*** b) DELETE

30. In SQL, what does the "COUNT" function return?

- a) The sum of values in a column.
- b) The average value in a column.
- c) The number of rows in a result set.
- d) The maximum value in a column.

**\*\*Answer:\*\*** c) The number of rows in a result set.

31. In SQL, what is the purpose of the "JOIN" clause?

- a) To create a new table.
- b) To filter rows based on a specified condition.
- c) To retrieve data from one or more tables based on a related column.
- d) To sort the result set in ascending or descending order.

**\*\*Answer:\*\*** c) To retrieve data from one or more tables based on a related column.

32. What SQL statement is used to modify the structure of an existing table?

- a) MODIFY TABLE
- b) UPDATE TABLE
- c) ALTER TABLE
- d) ADJUST TABLE

**\*\*Answer:\*\*** c) ALTER TABLE

33. In SQL, what does the "DISTINCT" keyword do when used in a SELECT statement?

- a) Filters rows based on a specified condition.
- b) Returns unique values from a column.
- c) Orders the result set in ascending or descending order.
- d) Combines rows from multiple tables.

**\*\*Answer:\*\*** b) Returns unique values from a column.

34. What SQL clause is used to filter rows based on a specified condition?

- a) WHERE
- b) FILTER
- c) HAVING
- d) GROUP BY

**\*\*Answer:\*\*** a) WHERE

35. In SQL, which operator is used to compare values for equality?

- a) = (Equal)
- b) <> (Not Equal)
- c) > (Greater Than)
- d) < (Less Than)

**\*\*Answer:\*\*** a) = (Equal)

36. What SQL command is used to remove a table structure from a database?

- a) DROP TABLE

- b) DELETE TABLE
- c) ERASE TABLE
- d) REMOVE TABLE

**\*\*Answer:\*\*** a) DROP TABLE

37. In SQL, what is the purpose of the "AS" keyword in a SELECT statement?

- a) To assign a value to a variable.
- b) To filter rows based on a specified condition.
- c) To create a new table.
- d) To rename columns or tables.

**\*\*Answer:\*\*** d) To rename columns or tables.

38. Which SQL command is used to retrieve all rows from a table?

- a) PULL ALL
- b) FETCH ALL
- c) SELECT ALL
- d) SHOW ALL

**\*\*Answer:\*\*** c) SELECT ALL

39. In SQL, what does the "GROUP BY" clause do?

- a) To filter rows based on a specified condition.
- b) To join multiple tables.
- c) To create a new table.

d) To group rows with identical values into summary rows.

**\*\*Answer:\*\*** d) To group rows with identical values into summary rows.

40. What SQL statement is used to change data in a table?

a) UPDATE

b) MODIFY

c) ALTER

d) CHANGE

**\*\*Answer:\*\*** a) UPDATE

41. In SQL, what does the "UNION" operator do when used between two SELECT statements?

a) Combines rows from multiple tables based on a common column.

b) Returns the intersection of rows from two SELECT statements.

c) Combines the result sets of two SELECT statements, removing duplicates.

d) Performs a cross-product of rows from two SELECT statements.

**\*\*Answer:\*\*** c) Combines the result sets of two SELECT statements, removing duplicates.

42. What SQL statement is used to remove a row from a table?

a) DELETE

b) DROP

c) ERASE

d) REMOVE



**\*\*Answer:\*\*** a) DELETE

43. In SQL, what does the "GROUP BY" clause with the "HAVING" clause allow you to do?

- a) To filter rows based on a specified condition.
- b) To combine rows from multiple tables.
- c) To aggregate data after grouping based on a condition.
- d) To sort the result set in ascending or descending order.

**\*\*Answer:\*\*** c) To aggregate data after grouping based on a condition.

44. Which SQL function is used to find the highest value in a column?

- a) AVG
- b) MAX
- c) MIN
- d) SUM

**\*\*Answer:\*\*** b) MAX

45. In SQL, what is the purpose of the "AS" keyword when used with column aliases?

- a) To change the data type of the column.
- b) To create a new column.
- c) To specify a name for the column in the result set.
- d) To filter rows based on a condition.

**\*\*Answer:\*\*** c) To specify a name for the column in the result set.

46. Which SQL statement is used to create a new database?

- a) CREATE DATABASE
- b) NEW DATABASE
- c) INITIATE DATABASE
- d) BUILD DATABASE

**\*\*Answer:\*\*** a) CREATE DATABASE

47. In SQL, what does the "BETWEEN" operator do?

- a) Filters rows based on a specified condition.
- b) Checks if a value is within a range.
- c) Combines rows from multiple tables.
- d) Calculates the average value in a column.

**\*\*Answer:\*\*** b) Checks if a value is within a range.

48. What SQL command is used to add a new column to an existing table?

- a) APPEND COLUMN
- b) ALTER TABLE
- c) MODIFY TABLE
- d) ADD COLUMN

**\*\*Answer:\*\*** d) ADD COLUMN

49. In SQL, what is the purpose of the "LIMIT" clause?

- a) To filter rows based on a specified condition.
- b) To group rows with identical values into summary rows.
- c) To restrict the number of rows returned by a query.
- d) To sort the result set in ascending or descending order.

**\*\*Answer:\*\*** c) To restrict the number of rows returned by a query.

50. Which SQL function is used to find the average value in a column?

- a) AVG
- b) MAX
- c) MIN
- d) SUM

**\*\*Answer:\*\*** a) AVG

1. Relational Algebra is a \_\_\_\_\_ query language that takes two relations as input and produces another relation as an output of the query.

- a) Relational
- b) Structural
- c) Procedural
- d) Fundamental

Answer: c

2. Which of the following is a fundamental operation in relational algebra?

- a) Set intersection
- b) Natural join
- c) Assignment
- d) None of the mentioned

Answer: d

3. Which of the following is used to denote the selection operation in relational algebra?

- a) Pi (Greek)
- b) Sigma (Greek)
- c) Lambda (Greek)
- d) Omega (Greek)

Answer: b

4. For select operation the \_\_\_\_\_ appear in the subscript and the \_\_\_\_\_ argument appears in the paranthesis after the sigma.

- a) Predicates, relation
- b) Relation, Predicates
- c) Operation, Predicates
- d) Relation, Operation

Answer: a

5. The \_\_\_\_\_ operation, denoted by  $-$ , allows us to find tuples that are in one relation but are not in another.

- a) Union
- b) Set-difference
- c) Difference
- d) Intersection

Answer: b

6. Which is a unary operation:

- a) Selection operation
- b) Primitive operation
- c) Projection operation
- d) Generalized selection

Answer: d

7. Which is a join condition contains an equality operator:

- a) Equijoins
- b) Cartesian
- c) Natural
- d) Left

Answer: a

8. In precedence of set operators, the expression is evaluated from

- a) Left to left
- b) Left to right
- c) Right to left
- d) From user specification

Answer: b

9. Which of the following is not outer join?

- a) Left outer join
- b) Right outer join
- c) Full outer join
- d) All of the mentioned

Answer: d

10. The assignment operator is denoted by

- a) ->
- b) <-
- c) =
- d) ==

Answer: b

1. An \_\_\_\_\_ is a set of entities of the same type that share the same properties, or attributes.

- a) Entity set
- b) Attribute set

- c) Relation set
- d) Entity model

Answer: a

2. Entity is a \_\_\_\_\_

- a) Object of relation
- b) Present working model
- c) Thing in real world
- d) Model of relation

Answer: c

3. The descriptive property possessed by each entity set is \_\_\_\_\_

- a) Entity
- b) Attribute
- c) Relation
- d) Model

Answer: b

4. The function that an entity plays in a relationship is called that entity's \_\_\_\_\_

- a) Participation
- b) Position
- c) Role
- d) Instance

Answer: c

5. The attribute *name* could be structured as an attribute consisting of first name, middle initial, and last name. This type of attribute is called

- a) Simple attribute
- b) Composite attribute
- c) Multivalued attribute
- d) Derived attribute

Answer: b

6. The attribute AGE is calculated from DATE\_OF\_BIRTH. The attribute AGE is

- a) Single valued
- b) Multi valued
- c) Composite
- d) Derived

Answer: d

7. Not applicable condition can be represented in relation entry as

- a) NA
- b) 0
- c) NULL
- d) Blank Space

Answer: c

8. Which of the following can be a multivalued attribute?

- a) Phone\_number
- b) Name
- c) Date\_of\_birth
- d) All of the mentioned

Answer: a

9. Which of the following is a single valued attribute

- a) Register\_number
- b) Address
- c) SUBJECT\_TAKEN
- d) Reference

Answer: a

10. In a relation between the entities the type and condition of the relation should be specified. That is called as\_\_\_\_\_attribute.

- a) Descriptive
- b) Derived
- c) Recursive
- d) Relative

Answer: a

**Question 1:** In Relational Algebra, the operation that is used to select rows from a relation that satisfy a given condition is:

- A. Projection
- B. Union
- C. Join
- D. Selection

**Answer:** D. Selection

**Question 2:** Consider two relations R and S. If R has n tuples and S has m tuples, what is the maximum number of tuples in the result of the natural join ( $R \bowtie S$ )?

- A.  $n + m$
- B.  $n * m$
- C.  $\min(n, m)$
- D.  $\max(n, m)$

**Answer:** B.  $n * m$

**Question 3:** Which of the following relational algebra operations is equivalent to the SQL `INNER JOIN`?



- A.  $\sigma$  (Selection)
- B.  $\pi$  (Projection)
- C.  $\bowtie$  (Natural Join)
- D.  $\cup$  (Union)

**\*\*Answer:\*\*** C.  $\bowtie$  (Natural Join)

**\*\*Question 4:\*\*** In Relational Algebra, the Cartesian product of two relations with 'm' and 'n' tuples will result in:

- A.  $m + n$  tuples
- B.  $m * n$  tuples
- C.  $m - n$  tuples
- D.  $m / n$  tuples

**\*\*Answer:\*\*** B.  $m * n$  tuples

**\*\*Question 5:\*\*** The relational algebra operation that combines two relations and eliminates duplicate rows is:

- A. Union
- B. Intersection
- C. Set Difference
- D. Natural Join

**\*\*Answer:\*\*** A. Union

**Question 6:** In Relational Algebra, what is the result of the following expression:  
 $\pi(A, B)(R)$  where R is a relation with attributes A, B, and C?

- A. A and B from relation R
- B. A from relation R
- C. B from relation R
- D. A, B, and C from relation R

**Answer:** A. A and B from relation R

**Question 7:** Which of the following is a unary operation in Relational Algebra?

- A. Join
- B. Union
- C. Projection
- D. Division

**Answer:** C. Projection

**Question 8:** Which of the following operations is used to eliminate duplicates from a relation?

- A. Projection
- B. Selection
- C. Union
- D. Difference

**Answer:** A. Projection

**Question 9:** The result of the relational algebra operation  $\rho(X/Y)(R)$  will:

- A. Rename the attribute X to Y in relation R
- B. Project the attribute X from relation R
- C. Select rows where attribute X equals attribute Y in relation R
- D. None of the above

**Answer:** A. Rename the attribute X to Y in relation R

**Question 10:** In Relational Algebra, the operation that combines two relations but retains only the common rows is:

- A. Union
- B. Intersection
- C. Set Difference
- D. Natural Join

**Answer:** B. Intersection

**Question 11:** Which of the following operations in Relational Algebra corresponds to the SQL `SELECT DISTINCT` statement?

- A. Projection
- B. Selection
- C. Difference
- D. Set Difference

**Answer:** A. Projection

**Question 12:** Given two relations  $R(A, B)$  and  $S(B, C)$ , what is the result of the natural join ( $R \bowtie S$ )?

- A. A, B, C
- B. A, B
- C. B
- D. A, C

**Answer:** A. A, B, C

**Question 13:** Which of the following is an example of a binary operation in Relational Algebra?

- A. Projection
- B. Selection
- C. Cross Product
- D. Intersection

**Answer:** C. Cross Product

**Question 14:** The relational algebra operation that selects distinct rows from a relation is:

- A.  $\pi$  (Projection)
- B.  $\sigma$  (Selection)
- C.  $\rho$  (Renaming)
- D.  $\delta$  (Duplicate Elimination)

**Answer:** D.  $\delta$  (Duplicate Elimination)

**Question 15:** In Relational Algebra, which operation is used to combine two relations and include all rows from both, filling in missing columns with NULL values?

- A. Union
- B. Intersection
- C. Set Difference
- D. Outer Join

**Answer:** D. Outer Join

**Question 16:** What does the relational algebra operation  $\rho(A/B)(R)$  do to relation R?

- A. Rename attribute B to A
- B. Rename attribute A to B
- C. Select rows where attribute A equals attribute B
- D. Project attribute A from relation R

**Answer:** B. Rename attribute A to B

**Question 17:** The operation in Relational Algebra that returns all tuples from one relation that are not present in another relation is:

- A. Join
- B. Intersection
- C. Set Difference
- D. Union

**\*\*Answer:\*\* C. Set Difference**

**\*\*Question 18:\*\*** In Relational Algebra, the operation that combines two relations and includes all rows from both relations, filling in missing columns with NULL values, is known as:

- A. Union
- B. Intersection
- C. Set Difference
- D. Full Outer Join

**\*\*Answer:\*\* D. Full Outer Join**

**\*\*Question 19:\*\*** What is the result of the relational algebra operation  $\pi(A, B) (\sigma(A > 5) (R))$  where R is a relation with attributes A, B, and C?

- A. A and B from relation R where A is greater than 5
- B. A from relation R where A is greater than 5
- C. B from relation R where A is greater than 5
- D. A, B, and C from relation R where A is greater than 5

**\*\*Answer:\*\* A. A and B from relation R where A is greater than 5**

**\*\*Question 20:\*\*** Which of the following relational algebra operations is equivalent to the SQL `LEFT JOIN`?

- A. Semi-Join
- B. Theta Join

- C. Equi-Join
- D. Left Outer Join

**\*\*Answer:\*\*** D. Left Outer Join

1. \_\_\_\_\_ express the number of entities to which another entity can be associated via a relationship set.

- a) Mapping Cardinality
- b) Relational Cardinality
- c) Participation Constraints
- d) None of the mentioned

Answer: a

2. An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A. This is called as

- a) One-to-many
- b) One-to-one
- c) Many-to-many
- d) Many-to-one

Answer: b

3. An entity in A is associated with at most one entity in B. An entity in B, however, can be associated with any number (zero or more) of entities in A.

- a) One-to-many
- b) One-to-one
- c) Many-to-many
- d) Many-to-one

Answer: d

4. Data integrity constraints are used to:

- a) Control who is allowed access to the data
- b) Ensure that duplicate records are not entered into the table
- c) Improve the quality of data entered for a specific property

d) Prevent users from changing the values stored in the table

Answer: c

5. Establishing limits on allowable property values, and specifying a set of acceptable, predefined options that can be assigned to a property are examples of:

- a) Attributes
- b) Data integrity constraints
- c) Method constraints
- d) Referential integrity constraints

Answer: b

6. Which of the following can be addressed by enforcing a referential integrity constraint?

- a) All phone numbers must include the area code
- b) Certain fields are required (such as the email address, or phone number) before the record is accepted
- c) Information on the customer must be known before anything can be sold to that customer
- d) Then entering an order quantity, the user must input a number and not some text (i.e., 12 rather than 'a dozen')

Answer: c

7. \_\_\_\_\_ is a special type of integrity constraint that relates two relations & maintains consistency across the relations.

- a) Entity Integrity Constraints
- b) Referential Integrity Constraints
- c) Domain Integrity Constraints
- d) Domain Constraints

Answer: b

8. Which one of the following uniquely identifies the elements in the relation?

- a) Secondary Key
- b) Primary key



- c) Foreign key
- d) Composite key

Answer: b

9. Drop Table cannot be used to drop a table referenced by a \_\_\_\_\_ constraint .

- a) Local Key
- b) Primary Key
- c) Composite Key
- d) Foreign Key

Answer: d

10. \_\_\_\_\_ is preferred method for enforcing data integrity

- a) Constraints
- b) Stored Procedure
- c) Triggers
- d) Cursors

Answer: a

1. Which of the following gives a logical structure of the database graphically?

- a) Entity-relationship diagram
- b) Entity diagram
- c) Database diagram
- d) Architectural representation

Answer: a

2. The entity relationship set is represented in E-R diagram as

- a) Double diamonds
- b) Undivided rectangles
- c) Dashed lines
- d) Diamond

Answer: d

3. The Rectangles divided into two parts represents

- a) Entity set
- b) Relationship set
- c) Attributes of a relationship set
- d) Primary key

Answer: a

4. Consider a directed line(->) from the relationship set advisor to both entity sets instructor and student. This indicates \_\_\_\_\_ cardinality

- a) One to many
- b) One to one
- c) Many to many
- d) Many to one

Answer: b

5. We indicate roles in E-R diagrams by labeling the lines that connect \_\_\_\_\_ to \_\_\_\_\_

- a) Diamond , diamond
- b) Rectangle, diamond
- c) Rectangle, rectangle
- d) Diamond, rectangle

Answer: d

6. An entity set that does not have sufficient attributes to form a primary key is termed a \_\_\_\_\_

- a) Strong entity set
- b) Variant set
- c) Weak entity set
- d) Variable set

Answer: c

7. For a weak entity set to be meaningful, it must be associated with another entity set, called the

- a) Identifying set
- b) Owner set
- c) Neighbour set
- d) Strong entity set

Answer: a

8. Weak entity set is represented as

- a) Underline
- b) Double line
- c) Double diamond
- d) Double rectangle

Answer: d

9. If you were collecting and storing information about your music collection, an album would be considered a(n) \_\_\_\_\_

- a) Relation
- b) Entity
- c) Instance
- d) Attribute

Answer: b

10. What term is used to refer to a specific record in your music database; for instance; information stored about a specific album?

- a) Relation
- b) Instance
- c) Table
- d) Column

Answer: b

Consider the following relational schemas and answer the questions below

The *section* relation

Course_id	Sec_id	Semester	Year
BIO-101	1	Spring	2010
CS-102	4	Summer	2009
EE-201	3	Fall	2010
FIN-301	1	Spring	2011

The *teaches* relation

Id	Course_id	Sec_id	Semester	Year
1001	CS-101	1	Fall	2009
1002	EE-201	2	Spring	2010
1003	FIN-301	3	Fall	2009
1004	BIO-101	1	Summer	2011

1. Which one of the following can be treated as a primary key in teaches relation?

- a) Id
- b) Semester
- c) Sec\_id
- d) Year

Answer: a

2. The primary key in the section relation is

- a) Course\_id
- b) Sec\_id
- c) Both Course\_id and Sec\_id
- d) All the attributes

Answer: a

3.

```
SELECT * FROM teaches WHERE Sec_id = 'CS-101';
```

Which of the following Id is selected for the following query?

- a) 1003
- b) 1001
- c) None
- d) Error message appears

Answer: d

4.

```
SELECT Id, Course_id, Building FROM SECTION s AND teaches t WHERE t.year=2009;
```

Which of the following Id are displayed?

- a) 1003
- b) 1001
- c) Both 1003 and 1001
- d) Error message appears

Answer: c

5. The query which selects the Course\_id 'CS-101' from the section relation is

- a) Select Course\_id from section where Building = 'Richard';
- b) Select Course\_id from section where Year = '2009';
- c) Select Course\_id from teaches where Building = 'Packyard';
- d) Select Course\_id from section where Sec\_id = '3';

Answer: b

6.

```
CREATE TABLE SECTION
(Course_id VARCHAR (8),
Sec_id VARCHAR (8),
Semester VARCHAR (6),
YEAR NUMERIC (4,0),
Building NUMERIC (15),
PRIMARY KEY (course id, sec id, semester, YEAR),
FOREIGN KEY (course id) REFERENCES course);
```

Which of the following has an error in the above create table for the relation section

- a) Primary key (course id, sec id, semester, year)
- b) Foreign key (course id) references course
- c) Year numeric (4,0)
- d) Building numeric (15)

Answer: d

7. The relation with primary key can be created using

- a) Create table instructor (Id, Name)
- b) Create table instructor (Id, Name, primary key(name))
- c) Create table instructor (Id, Name, primary key (Id))
- d) Create table instructor ( Id unique, Name )

Answer: c

8. How can the values in the relation teaches be deleted?

- a) Drop table teaches;
- b) Delete from teaches;
- c) Purge table teaches;
- d) Delete from teaches where Id ='Null';

Answer: b

9. In the above teaches relation " Select \* from teaches where Year = '2010'" displays how many rows?

- a) 2
- b) 4

- c) 5
- d) 1

Answer: a

10. The relation changes can be got back using \_\_\_\_\_ command.

- a) Flashback
- b) Purge
- c) Delete
- d) Getback

Answer: a

1. Let us consider *phone\_number* ,which can take single or several values .  
Treating *phone\_number* as an \_\_\_\_\_ permits instructors to have several phone numbers (including zero) associated with them.

- a) Entity
- b) Attribute
- c) Relation
- d) Value

Answer: a.

2. The total participation by entities is represented in E-R diagram as

- a) Dashed line
- b) Double line
- c) Double rectangle
- d) Circle

Answer: b

3. Given the basic ER and relational models, which of the following is INCORRECT?

- a) An attribute of an entity can have more than one value
- b) An attribute of an entity can be composite
- c) In a row of a relational table, an attribute can have more than one value
- d) In a row of a relational table, an attribute can have exactly one value or a NULL value

Answer: c

4. Which of the following indicates the maximum number of entities that can be involved in a relationship?

- a) Minimum cardinality
- b) Maximum cardinality
- c) ERD
- d) Greater Entity Count

Answer: b

5. In E-R diagram generalization is represented by

- a) Ellipse
- b) Dashed ellipse
- c) Rectangle
- d) Triangle

Answer: d

6. What is a relationship called when it is maintained between two entities?

- a) Unary
- b) Binary
- c) Ternary
- d) Quaternary

Answer: b

7. Which of the following is a low level operator?

- a) Insert
- b) Update
- c) Delete
- d) Directory

Answer: d

8. Key to represent relationship between tables is called



- a) Primary key
- b) Secondary Key
- c) Foreign Key
- d) None of the mentioned

Answer: c

9. A window into a portion of a database is

- a) Schema
- b) View
- c) Query
- d) Data dictionary

Answer: b

10. A primary key is combined with a foreign key creates

- a) Parent-Child relation ship between the tables that connect them
- b) Many to many relationship between the tables that connect them
- c) Network model between the tables that connect them
- d) None of the mentioned

Answer: a

1. The entity set person is classified as student and employee. This process is called

\_\_\_\_\_

- a) Generalization
- b) Specialization
- c) Inheritance
- d) Constraint generalization

Answer: b

2. Which relationship is used to represent a specialization entity?

- a) ISA
- b) AIS
- c) ONIS

d) WHOIS

Answer: a

3. The refinement from an initial entity set into successive levels of entity subgroupings represents a \_\_\_\_\_ design process in which distinctions are made explicit.

- a) Hierarchy
- b) Bottom-up
- c) Top-down
- d) Radical

Answer: c

4. There are similarities between the instructor entity set and the secretary entity set in the sense that they have several attributes that are conceptually the same across the two entity sets: namely, the identifier, name, and salary attributes. This process is called

- a) Commonality
- b) Specialization
- c) Generalization
- d) Similarity

Answer: c

5. If an entity set is a lower-level entity set in more than one ISA relationship, then the entity set has

- a) Hierarchy
- b) Multilevel inheritance
- c) Single inheritance
- d) Multiple inheritance

Answer: d

6. A \_\_\_\_\_ constraint requires that an entity belong to no more than one lower-level entity set.

- a) Disjointness
- b) Uniqueness
- c) Special
- d) Relational

Answer: a

7. Consider the employee work-team example, and assume that certain employees participate in more than one work team. A given employee may therefore appear in more than one of the team entity sets that are lower level entity sets of employee. Thus, the generalization is \_\_\_\_\_

- a) Overlapping
- b) Disjointness
- c) Uniqueness
- d) Relational

Answer: a

8. The completeness constraint may be one of the following: Total generalization or specialization, Partial generalization or specialization. Which is the default?

- a) Total
- b) Partial
- c) Should be specified
- d) Cannot be determined

Answer: b

9. Functional dependencies are a generalization of

- a) Key dependencies
- b) Relation dependencies
- c) Database dependencies
- d) None of the mentioned

Answer: a

10. Which of the following is another name for a weak entity?

- a) Child
- b) Owner
- c) Dominant
- d) All of the mentioned

Answer: a

```
CREATE TABLE department
(dept_name VARCHAR (20),
building VARCHAR (15),
budget NUMBER,
PRIMARY KEY (dept_name));

CREATE TABLE course
(course_id VARCHAR (7),
title VARCHAR (50),
dept_name VARCHAR (20),
credits NUMERIC (2,0),
PRIMARY KEY (course_id),
FOREIGN KEY (dept_name) _____ department);

CREATE TABLE instructor
(ID VARCHAR (5),
name VARCHAR (20) NOT NULL,
dept_name VARCHAR (20),
salary NUMERIC (8,2),
FOREIGN KEY (dept_name) _____ department);

CREATE TABLE SECTION
(course_id VARCHAR (8),
sec_id VARCHAR (8),
semester VARCHAR (6),
YEAR NUMERIC (4,0),
building VARCHAR (15),
room_number VARCHAR (7),
time_slot id VARCHAR (4),
PRIMARY KEY (course_id, sec_id, semester, YEAR),
FOREIGN KEY (_____) _____ course);

CREATE TABLE teaches
(ID VARCHAR (5),
course_id VARCHAR (8),
sec_id VARCHAR (8),
semester VARCHAR (6),
```

```
YEAR NUMERIC (4,0),  
PRIMARY KEY (ID, course_id, sec_id, semester, YEAR),  
FOREIGN KEY (course_id, sec_id, semester, YEAR) REFERENCES SECTION,  
FOREIGN KEY (ID) _____ instructor);
```

Answer questions based on the above commands

1. Which is the main relation which is used in the university database which is referenced by all other relation of the university?

- a) Teaches
- b) Course
- c) Department
- d) Section

Answer: c

2. The department relation has the an entry budget whose type has to be replaced by

- a) Varchar (20)
- b) Varchar2 (20)
- c) Numeric (12,2)
- d) Numeric

Answer: c

3. In the course relation, the title field should throw an error in case of any missing title. The command to be added in title is

- a) Unique
- b) Not null
- c) 0
- d) Null

Answer: b

4. In the above DDL command the foreign key entries are got by using the keyword

- a) References
- b) Key reference
- c) Relating

d) None of the mentioned

Answer: a

5. Identify the error in the section relation

- a) No error
- b) Year numeric (4,0)
- c) Building varchar (15)
- d) Sec\_id varchar (8)

Answer: a

6. The following entry is given in to the instructor relation .

```
(100202, Drake, Biology, 30000)
```

Identify the output of the query given

- a) Row(s) inserted
- b) Error in ID of insert
- c) Error in Name of insert
- d) Error in Salary of the insert

Answer: b

7. Which of the following can be used as a primary key entry of the instructor relation.

- a) DEPT\_NAME
- b) NAME
- c) ID
- d) All of the mentioned

Answer: c

8. In the section relation which of the following is used as a foreign key?

- a) Course\_id
- b) Course\_id, sec\_id
- c) Room\_number
- d) Course\_id, sec\_id, room\_number

Answer: a

9. In order to include an attribute Name to the teaches relation which of the following command is used?

- a) Alter table teaches include Name;
- b) Alter table teaches add Name;
- c) Alter table teaches add Name varchar;
- d) Alter table teaches add Name varchar(20);

Answer: d

10. To replace the relation section with some other relation the initial step to be carried out is

- a) Delete section;
- b) Drop section;
- c) Delete from section;
- d) Replace section new\_table ;

Answer: b

This set of Database Questions & Answers focuses on “Querying database part-1 DML”

The *instructor* relation

ID	Name	Dept_name	Salary
10101	Hayley	Comp.Sci.	65000
12121	Jackson	Finance	90000
15151	Nathan	Music	87000
22222	April	Biology	73000
34345	Crick	Comp.Sci.	100000

The *course* relation

Course_id	Title	Dept_name	Credits
CS-101	Robotics	Comp.Sci.	5
BIO-244	Genetics	Biology	4
PHY-333	Physical Principles	Physics	3
MUS-562	Music Video Production	Music	2
FIN-101	Investment Banking	Finance	3

Answer the questions based on the above relations

1. Which of the following command is used to display the departments of the instructor relation?

- a) Select \* from instructor where Dept\_name = Finance;
- b) Select \* from instructor ;
- c) Select dept\_name from instructor;
- d) Select dept\_name for instructor where Name=Jackson;

Answer: c

2. How can we select the elements which have common Dept\_name in both the relation ?

- a) Select \* from instructor i , course c where i.Dept\_name=c.Dept\_name;
- b) Select Dept name from instructor ,Course ;
- c) Select \* from instructor i , course c ;
- d) Select Dept\_name from instructor where Dept\_name = NULL;

Answer: a

3. Select distinct Dept\_name from instructor ;  
How many row(s) are displayed ?



- a) 4
- b) 3
- c) 5
- d) Error

Answer: a

4. Suppose the Authority want to include a new instructor for the title Neuroscience what command should be inserted ?

- a) Insert into instructor values(12111,Emma,NeuroScience,200000);
- b) Insert into course values(12111,Introduction,NeuroScience,2);
- c)

Insert into instructor values(12111,Emma,Biology,200000);

Insert into course values(BIO-112,Introduction to Neuro Science,NeuroScience,2);

- d) Insert into course values(12111,Emma,NeuroScience,200000);

Answer: c

5. If a person all the people in Music department gets fired which of the following has to be performed on the instructor relation ?

- a) Delete Dept\_name=Music in instructor;
- b) Delete from instructor where Dept\_name=Music;
- c) Remove Dept\_name= Music
- d) All of the mentioned

Answer: b

6.

```
SELECT DISTINCT T.name
FROM instructor AS T, instructor AS S
WHERE T.salary > S.salary AND S.dept name = 'Comp.Sci.';
```

What will be displayed as the value of name for the above query?

- a) Hayley
- b) Jackson

- c) Hayley and Crick
- d) Crick

Answer: d

7.

```
SELECT Name
FROM instructor
WHERE salary > SOME (SELECT salary FROM instructor WHERE dept_name = 'Comp.Sci.');
```

How many rows are selected ?

- a) 3
- b) 4
- c) 2
- d) 1

Answer: d

8. How will you select the Names whose first letter is E ?

a)

```
SELECT Name
FROM instructor
WHERE Name LIKE 'A%;
```

b)

```
SELECT Name
FROM course
WHERE Name LIKE 'A%;
```

c)

```
SELECT Dept_name
FROM instructor
WHERE Name LIKE 'A%;
```

d)

```
SELECT Name
FROM instructor
WHERE Dept_name LIKE 'A%;
```

Answer: a

9. Which function is used to find the count of distinct departments?

- a) Dist
- b) Distinct
- c) Count
- d) Count,Dist

Answer: a

10. Which function is used to identify the title with Least scope?

- a) Min(Credits)
- b) Max(Credits)
- c) Min(title)
- d) Min(Salary)

Answer: a

This set of Database Multiple Choice Questions & Answers (MCQs) focuses on "Atomic Domains".

1. A domain is \_\_\_\_\_ if elements of the domain are considered to be indivisible units.

- a) Atomic
- b) Subatomic
- c) Substructure
- d) Subset

Answer: a

2. Identify the composite attributes

- a) Salary
- b) Credits
- c) Section\_id
- d) None of the mentioned

Answer: d

3. Consider the relation given below and find the maximum normal form applicable to them

- i.  $R(A, B)$  WITH productions  $\{ A \twoheadrightarrow B \}$
- ii.  $R(A, B)$  WITH productions  $\{ B \twoheadrightarrow A \}$
- iii.  $R(A, B)$  WITH productions  $\{ A \rightarrow B, B \twoheadrightarrow A \}$
- iv.  $R(A, B, C)$  WITH productions  $\{ A \twoheadrightarrow B, B \twoheadrightarrow A, AB \twoheadrightarrow C \}$

- a) i, ii and iii are in 3NF and iv is in BCNF
- b) i and ii are in BCNF and iii and iv are in 3NF
- c) All are in 3NF
- d) All are in BCNF

Answer: d

4. Which one is based on multi-valued dependency:

- a) First
- b) Second
- c) Third
- d) Fourth

Answer: d

5. If a relation is in BCNF, then it is also in

- a) 1 NF
- b) 2 NF
- c) 3 NF
- d) All of the mentioned

Answer: d

6. If every non-key attribute is functionally dependent primary key, then the relation will be in

- a) First normal form
- b) Second normal form
- c) Third form
- d) Fourth normal form

Answer: b

7. If an attribute of a composite key is dependent on an attribute of the other composite key, a normalization called \_\_\_\_\_ is needed.

- a) DKNF
- b) BCNF
- c) Fourth
- d) Third

Answer: b

8. The term for information that describes what type of data is available in a database is:

- a) Data dictionary
- b) data repository
- c) Index data
- d) Metadata

Answer: d

9. A data type that creates unique numbers for key columns in Microsoft Access is:

- a) Autonumber
- b) Boolean
- c) Sequential key
- d) Sequential number

Answer: a

10. A dependency exist between two columns when

- a) Together they constitute a composite key for the table
- b) Knowing the value in one column determines the value stored in another column
- c) The table is in 3NF
- d) Together they constitute a foreign key

Answer: a

### Employee Table:

- Entity Name: Employee
- Attributes:
  - Emp\_ID (Primary Key)
  - Emp\_Name
  - Emp\_Expertise
  - Emp\_Designation

### Project Table:

- Entity Name: Project
- Attributes:
  - Project\_ID (Primary Key)
  - Project\_Name
  - Project\_Category

### Work Table:

- Entity Name: Work
- Attributes:
  - Work\_Code (Primary Key)
  - Emp\_ID (Foreign Key referencing Employee.Emp\_ID)
  - Project\_ID (Foreign Key referencing Project.Project\_ID)
  - Technology

### Doctor entity:

- Attributes
  - Dr\_id (Primary Key)
  - Dr\_Name
  - Dr\_specialization

### Patient entity:

- Attributes
  - P\_id (Primary Key)
  - P\_Name
  - P\_Disease
  - P\_Allergies
  - P\_City

### Test entity:

- Attributes
  - Test\_id (Primary Key)
  - P\_id (Foreign Key referring to Patient)
  - Test\_Name

### Treatment entity:

- Attributes
  - Treatment\_id (Primary Key)
  - P\_id (Foreign Key referring to Patient)
  - Dr\_id (Foreign Key referring to Doctor)
  - Admit\_Date

Q 1. Retrieve the projects with their names and the total years of experience of all employees working on them.

- A. `SELECT Project_Name, AVG(Emp_Expertiese) AS Total_Experience FROM Project INNER JOIN Work ON Project.Project_ID = Work.Project_ID INNER JOIN Employee ON Work.Emp_ID = Employee.Emp_ID GROUP BY Project_Name`
- B. `SELECT Project_Name, MAX(Emp_Expertiese) AS Total_Experience FROM Project INNER JOIN Work ON Project.Project_ID = Work.Project_ID INNER JOIN Employee ON Work.Emp_ID = Employee.Emp_ID GROUP BY Project_Name`
- C. `SELECT Project_Name, SUM(Emp_Expertiese) AS Total_Experience FROM Project INNER JOIN Work ON Project.Project_ID = Work.Project_ID INNER JOIN Employee ON Work.Emp_ID = Employee.Emp_ID GROUP BY Project_Name`
- D. `SELECT Project_Name, MIN(Emp_Expertiese) AS Total_Experience FROM Project INNER JOIN Work ON Project.Project_ID = Work.Project_ID INNER JOIN Employee ON Work.Emp_ID = Employee.Emp_ID GROUP BY Project_Name`

Q 2. Update the designation of employees to 'Senior Developer' if they have more than 5 years of experience, 'Junior Developer' if they have more than 2 years of experience, and 'Intern' otherwise.

- A. `UPDATE Employee SET Emp_Designation = CASE WHEN Emp_Expertiese > 2 THEN 'Intern' WHEN Emp_Expertiese > 5 THEN 'Senior Developer' ELSE 'Junior Developer' END`
- B. `UPDATE Employee SET Emp_Designation = CASE WHEN Emp_Expertiese > 2 THEN 'Intern' WHEN Emp_Expertiese > 5 THEN 'Junior Developer' ELSE 'Senior Developer' END`
- C. `UPDATE Employee SET Emp_Designation = CASE WHEN Emp_Expertiese > 5 THEN 'Senior Developer' WHEN Emp_Expertiese > 2 THEN 'Junior Developer' ELSE 'Intern' END`
- D. `UPDATE Employee SET Emp_Designation = CASE WHEN Emp_Expertiese > 5 THEN 'Junior Developer' WHEN Emp_Expertiese > 2 THEN 'Intern' ELSE 'Senior Developer' END`

Q 3. Find the doctors who have treated the most patients and show the total number of patients they have treated.

- A. `SELECT D.Dr_Name, COUNT(T.P_id) AS Total_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr_id = T.Dr_id GROUP BY D.Dr_id, D.Dr_Name HAVING COUNT(T.P_id) = (SELECT MAX(Total_Patients) FROM (SELECT D.Dr_id, COUNT(T.P_id) AS Total_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr_id = T.Dr_id GROUP BY D.Dr_id) AS Subquery);`
- B. `SELECT D.Dr_Name, COUNT(T.P_id) AS Total_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr_id = T.Dr_id GROUP BY D.Dr_id, D.Dr_Name HAVING COUNT(T.P_id) =`

(SELECT AVG(Total\_Patients) FROM (SELECT D.Dr\_id, COUNT(T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id GROUP BY D.Dr\_id) AS Subquery);

C. SELECT D.Dr\_Name, COUNT(T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(T.P\_id) = (SELECT MIN(Total\_Patients) FROM (SELECT D.Dr\_id, COUNT(T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id GROUP BY D.Dr\_id) AS Subquery);

D. SELECT D.Dr\_Name, COUNT(T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(T.P\_id) = (SELECT SUM(Total\_Patients) FROM (SELECT D.Dr\_id, COUNT(T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id GROUP BY D.Dr\_id) AS Subquery);

Q 4. List the doctors who have treated patients with 'Heart Disease' and 'Hypertension,' and show their names and the total number of such patients.

A. SELECT D.Dr\_Name, COUNT(DISTINCT T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease IN ('Heart Disease', 'Hypertension') GROUP BY D.Dr\_Name HAVING COUNT(DISTINCT T.P\_id) >= 2;

B. SELECT D.Dr\_Name, COUNT(DISTINCT T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease IN ('Heart Disease', 'Hypertension') GROUP BY D.Dr\_Name HAVING COUNT(DISTINCT T.P\_id) > 2;

C. SELECT D.Dr\_Name, COUNT(DISTINCT T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease IN ('Cancer', 'Diabetes') GROUP BY D.Dr\_Name HAVING COUNT(DISTINCT T.P\_id) > 2;

D. SELECT D.Dr\_Name, COUNT(DISTINCT T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease IN ('Cancer', 'Diabetes') GROUP BY D.Dr\_Name HAVING COUNT(DISTINCT T.P\_id) >= 2;

Q 5. Find the doctors who have treated patients with 'Asthma' and 'Allergies,' and show their names and the total number of such patients.

A. SELECT D.Dr\_Name, COUNT(DISTINCT T.P\_id) AS Total\_Patients FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE



P.P\_Disease IN ('Asthma', 'Allergies') GROUP BY D.Dr\_id, D.Dr\_Name HAVING  
COUNT(DISTINCT T.P\_id) = 1;

B. SELECT D.Dr\_Name, COUNT(DISTINCT T.P\_id) AS Total\_Patients FROM Doctor D INNER  
JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE  
P.P\_Disease IN ('Asthma', 'Allergies') GROUP BY D.Dr\_id, D.Dr\_Name HAVING  
COUNT(DISTINCT T.P\_id) = 2;

C. SELECT D.Dr\_Name, COUNT(DISTINCT T.P\_id) AS Total\_Patients FROM Doctor D INNER  
JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE  
P.P\_Disease IN ('Cancer', 'Hypertension') GROUP BY D.Dr\_id, D.Dr\_Name HAVING  
COUNT(DISTINCT T.P\_id) = 1;

D. SELECT D.Dr\_Name, COUNT(DISTINCT T.P\_id) AS Total\_Patients FROM Doctor D INNER  
JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE  
P.P\_Disease IN ('Cancer', 'Hypertension') GROUP BY D.Dr\_id, D.Dr\_Name HAVING  
COUNT(DISTINCT T.P\_id) = 2;

Q 6. Retrieve the names of doctors who have not treated patients with 'Cancer' and have an  
experience of more than 10 years.

A. SELECT D.Dr\_Name FROM Doctor D WHERE D.Dr\_specialization IN (SELECT DISTINCT  
D.Dr\_specialization FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER  
JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease = 'Cancer') AND D.Dr\_Experience <= 10;

B. SELECT D.Dr\_Name FROM Doctor D WHERE D.Dr\_specialization IN (SELECT DISTINCT  
D.Dr\_specialization FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER  
JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease = 'Cancer') AND D.Dr\_Experience > 10;

C. SELECT D.Dr\_Name FROM Doctor D WHERE D.Dr\_specialization NOT IN (SELECT DISTINCT  
D.Dr\_specialization FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER  
JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease = 'Cancer') AND D.Dr\_Experience <= 10;

D. SELECT D.Dr\_Name FROM Doctor D WHERE D.Dr\_specialization NOT IN (SELECT DISTINCT  
D.Dr\_specialization FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER  
JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease = 'Cancer') AND D.Dr\_Experience > 10;

Q 7. Calculate the average age of patients who are treated by doctors specializing in 'Pediatrics.'

A. SELECT AVG(YEAR(CURRENT\_DATE) - YEAR(P.P\_Birthdate)) AS Avg\_Age FROM Patient P  
INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id  
WHERE D.Dr\_specialization = 'Oncology';

B. SELECT AVG(YEAR(CURRENT\_DATE) - YEAR(P.P\_Birthdate)) AS Avg\_Age FROM Patient P  
INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id  
WHERE D.Dr\_specialization = 'Pediatrics';

C. SELECT AVG(YEAR(CURRENT\_DATE) - YEAR(P.P\_Birthdate)) AS Avg\_Age FROM Patient P  
INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id  
WHERE D.Dr\_specialization = 'Cardiology';

D. SELECT AVG(YEAR(CURRENT\_DATE) - YEAR(P.P\_Birthdate)) AS Avg\_Age FROM Patient P  
INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id  
WHERE D.Dr\_specialization = 'Surgery';

Q 8. Retrieve the names of employees and their corresponding designations, but if the designation is 'Manager,' display it as 'MGR,' 'Senior Developer' as 'Sr. Dev,' and 'Junior Developer' as 'Jr. Dev.'

A. SELECT Emp\_Name, CASE WHEN Emp\_Designation = 'Manager' THEN 'MGR' WHEN  
Emp\_Designation = 'Junior Developer' THEN 'Jr. Dev' WHEN Emp\_Designation = 'Senior  
Developer' THEN 'Sr. Dev' ELSE Emp\_Designation END FROM Employee

B. SELECT Emp\_Name, CASE Emp\_Designation WHEN 'Manager' THEN 'MGR' WHEN 'Senior  
Developer' THEN 'Sr. Dev' WHEN 'Junior Developer' THEN 'Jr. Dev' ELSE Emp\_Designation  
END FROM Employee

C. SELECT Emp\_Name, CASE WHEN Emp\_Designation = 'Manager' THEN 'MGR' WHEN  
Emp\_Designation = 'Senior Developer' THEN 'Sr. Dev' WHEN Emp\_Designation = 'Junior  
Developer' THEN 'Jr. Dev' ELSE Emp\_Designation END FROM Employee

D. SELECT Emp\_Name, CASE Emp\_Designation WHEN 'Manager' THEN 'MGR' WHEN 'Junior  
Developer' THEN 'Jr. Dev' WHEN 'Senior Developer' THEN 'Sr. Dev' ELSE Emp\_Designation  
END FROM Employee

Q 9. List the employees who have worked on projects in multiple project categories and classify them as 'Versatile' employees.

A. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work  
GROUP BY Emp\_ID HAVING COUNT(DISTINCT Project\_id) > 1)

B. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work  
GROUP BY Emp\_ID HAVING COUNT(Project\_id) > 1)

- C. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work GROUP BY Emp_ID HAVING COUNT(DISTINCT Project_id) = 1)`
- D. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work GROUP BY Emp_ID HAVING COUNT(Project_id) = 1)`

Q 10. Find the doctors who have treated patients from 'Chicago' and have performed at least one 'CT Scan' test.

- A. `SELECT D.Dr_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr_id = T.Dr_id INNER JOIN Patient P ON T.P_id = P.P_id INNER JOIN Test TS ON P.P_id = TS.P_id WHERE P.P_City = 'New York City' AND TS.Test_Name = 'MRI';`
- B. `SELECT D.Dr_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr_id = T.Dr_id INNER JOIN Patient P ON T.P_id = P.P_id INNER JOIN Test TS ON P.P_id = TS.P_id WHERE P.P_City = 'Chicago' AND TS.Test_Name = 'CT Scan';`
- C. `SELECT D.Dr_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr_id = T.Dr_id INNER JOIN Patient P ON T.P_id = P.P_id INNER JOIN Test TS ON P.P_id = TS.P_id WHERE P.P_City = 'New York City' AND TS.Test_Name = 'CT Scan';`
- D. `SELECT D.Dr_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr_id = T.Dr_id INNER JOIN Patient P ON T.P_id = P.P_id INNER JOIN Test TS ON P.P_id = TS.P_id WHERE P.P_City = 'Chicago' AND TS.Test_Name = 'MRI';`

Q 11. Update the salaries of employees by adding 10% for employees with more than 3 years of experience, 15% for employees with more than 5 years, and 20% for employees with more than 10 years.

- A. `UPDATE Employee SET Emp_Salary = Emp_Salary * 1.2 WHERE Emp_Expertiese > 3`
- B. `UPDATE Employee SET Emp_Salary = Emp_Salary * 1.15 WHERE Emp_Expertiese > 5`
- C. `UPDATE Employee SET Emp_Salary = Emp_Salary * 1.2 WHERE Emp_Expertiese > 10`
- D. `UPDATE Employee SET Emp_Salary = Emp_Salary * 1.1 WHERE Emp_Expertiese > 3`

Q 12. List the projects and their categories, but if the project category is 'Development,' display it as 'DEV,' 'Research' as 'RSR,' and 'Management' as 'MGT.'

- A. SELECT Project\_Name, CASE Project\_Categoery WHEN 'Development' THEN 'DEV' WHEN 'Management' THEN 'MGT' WHEN 'Research' THEN 'RSR' ELSE Project\_Categoery END FROM Project
- B. SELECT Project\_Name, CASE Project\_Categoery WHEN 'Development' THEN 'MGT' WHEN 'Research' THEN 'DEV' WHEN 'Management' THEN 'RSR' ELSE Project\_Categoery END FROM Project
- C. SELECT Project\_Name, CASE WHEN Project\_Categoery = 'Development' THEN 'DEV' WHEN Project\_Categoery = 'Research' THEN 'RSR' WHEN Project\_Categoery = 'Management' THEN 'MGT' ELSE Project\_Categoery END FROM Project
- D. SELECT Project\_Name, CASE Project\_Categoery WHEN 'Development' THEN 'DEV' WHEN 'Research' THEN 'RSR' WHEN 'Management' THEN 'MGT' ELSE Project\_Categoery END FROM Project

Q 13. Retrieve the projects and their categories with the count of employees working on each project.

- A. SELECT Project\_Name, Project\_Categoery, COUNT(Emp\_ID) AS Employee\_Count FROM Project LEFT JOIN Work ON Project.Project\_ID = Work.Project\_ID GROUP BY Project\_Name, Project\_Categoery
- B. SELECT Project\_Name, Project\_Categoery, MAX(Emp\_ID) AS Employee\_Count FROM Project RIGHT JOIN Work ON Project.Project\_ID = Work.Project\_ID GROUP BY Project\_Name, Project\_Categoery
- C. SELECT Project\_Name, Project\_Categoery, SUM(Emp\_ID) AS Employee\_Count FROM Project INNER JOIN Work ON Project.Project\_ID = Work.Project\_ID GROUP BY Project\_Name, Project\_Categoery
- D. SELECT Project\_Name, Project\_Categoery, AVG(Emp\_ID) AS Employee\_Count FROM Project LEFT JOIN Work ON Project.Project\_ID = Work.Project\_ID GROUP BY Project\_Name, Project\_Categoery

Q 14. List the projects that have more than two employees assigned to them.

- A. SELECT Project\_Name FROM Project WHERE Project\_ID IN (SELECT Project\_ID FROM Work GROUP BY Project\_id HAVING COUNT(Emp\_ID) = 2)
- B. SELECT Project\_Name FROM Project WHERE Project\_ID IN (SELECT Project\_ID FROM Work GROUP BY Project\_id HAVING COUNT(Emp\_ID) < 2)

- C. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_ID FROM Work GROUP BY Project_id HAVING COUNT(Emp_ID) > 2)`
- D. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_ID FROM Work GROUP BY Project_id HAVING COUNT(Emp_ID) = 3)`

Q 15. List all employees who have worked on projects in the 'Development' category.

- A. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id = (SELECT Project_ID FROM Project WHERE Project_Categoery <> 'Development'))`
- B. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id IN (SELECT Project_ID FROM Project WHERE Project_Categoery <> 'Development'))`
- C. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id = (SELECT Project_ID FROM Project WHERE Project_Categoery = 'Development'))`
- D. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id IN (SELECT Project_ID FROM Project WHERE Project_Categoery = 'Development'))`

Q 16. Identify employees who have worked on projects with a budget exceeding \$75,000, and if they have less than 5 years of experience, grant them a bonus of 5% of the project's budget.

- A. `UPDATE Employee SET Emp_Bonus = Project_Budget * 0.05 FROM Employee INNER JOIN Work ON Employee.Emp_ID = Work.Emp_ID INNER JOIN Project ON Work.Project_ID = Project.Project_ID WHERE Project_Budget >= 75000 AND Emp_Expertiese >= 5`
- B. `UPDATE Employee SET Emp_Bonus = Project_Budget * 0.05 FROM Employee INNER JOIN Work ON Employee.Emp_ID = Work.Emp_ID INNER JOIN Project ON Work.Project_ID = Project.Project_ID WHERE Project_Budget >= 75000 AND Emp_Expertiese < 5`
- C. `UPDATE Employee SET Emp_Bonus = Project_Budget * 0.05 FROM Employee INNER JOIN Work ON Employee.Emp_ID = Work.Emp_ID INNER JOIN Project ON Work.Project_ID = Project.Project_ID WHERE Project_Budget > 75000 AND Emp_Expertiese >= 5`
- D. `UPDATE Employee SET Emp_Bonus = Project_Budget * 0.05 FROM Employee INNER JOIN Work ON Employee.Emp_ID = Work.Emp_ID INNER JOIN Project ON Work.Project_ID = Project.Project_ID WHERE Project_Budget > 75000 AND Emp_Expertiese < 5`

Q 17. Retrieve the names of patients who are allergic to 'Penicillin' and are treated by doctors specializing in 'Allergy.'

- A. `SELECT P.P_Name FROM Patient P WHERE P.P_Allergies LIKE '%Penicillin%' AND P.P_id IN (SELECT T.P_id FROM Treatment T INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE D.Dr_specialization = 'Allergy')`
- B. `SELECT P.P_Name FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE P.P_Allergies LIKE '%Penicillin%' AND D.Dr_specialization = 'Allergy'`
- C. `SELECT P.P_Name FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE P.P_Allergies LIKE '%Penicillin%' AND D.Dr_specialization = 'Allergy'`
- D. `SELECT P.P_Name FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE P.P_Allergies LIKE '%Penicillin%' AND P.P_id IN (SELECT T.P_id FROM Treatment T INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE D.Dr_specialization = 'Allergy')`

Q 18. Find the doctors who have treated patients from 'San Francisco' and have had at least one patient from 'New York City.'

- A. `SELECT D.Dr_Name FROM Doctor D INNER JOIN Treatment T1 ON D.Dr_id = T1.Dr_id INNER JOIN Patient P1 ON T1.P_id = P1.P_id WHERE P1.P_City = 'San Francisco' AND EXISTS (SELECT 1 FROM Treatment T2 INNER JOIN Patient P2 ON T2.P_id = P2.P_id WHERE T2.Dr_id = D.Dr_id AND P2.P_City = 'New York City');`
- B. `SELECT D.Dr_Name FROM Doctor D INNER JOIN Treatment T1 ON D.Dr_id = T1.Dr_id INNER JOIN Patient P1 ON T1.P_id = P1.P_id WHERE P1.P_City = 'San Francisco' AND NOT EXISTS (SELECT 1 FROM Treatment T2 INNER JOIN Patient P2 ON T2.P_id = P2.P_id WHERE T2.Dr_id = D.Dr_id AND P2.P_City = 'Los Angeles');`
- C. `SELECT D.Dr_Name FROM Doctor D INNER JOIN Treatment T1 ON D.Dr_id = T1.Dr_id INNER JOIN Patient P1 ON T1.P_id = P1.P_id WHERE P1.P_City = 'New York City' AND EXISTS (SELECT 1 FROM Treatment T2 INNER JOIN Patient P2 ON T2.P_id = P2.P_id WHERE T2.Dr_id = D.Dr_id AND P2.P_City = 'San Francisco');`
- D. `SELECT D.Dr_Name FROM Doctor D INNER JOIN Treatment T1 ON D.Dr_id = T1.Dr_id INNER JOIN Patient P1 ON T1.P_id = P1.P_id WHERE P1.P_City = 'Los Angeles' AND NOT EXISTS (SELECT 1 FROM Treatment T2 INNER JOIN Patient P2 ON T2.P_id = P2.P_id WHERE T2.Dr_id = D.Dr_id AND P2.P_City = 'Chicago');`

Q 19. List the patients who have been treated by doctors from the same city as the patient and show the names of the doctors.

- A. `SELECT P.P_Name, D.Dr_Name FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE P.P_City = D.Dr_City AND P.P_id = D.Dr_id;`
- B. `SELECT P.P_Name, D.Dr_Name FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE P.P_City <> D.Dr_City AND P.P_id != D.Dr_id;`
- C. `SELECT P.P_Name, D.Dr_Name FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE P.P_City <> D.Dr_City AND P.P_id = D.Dr_id;`
- D. `SELECT P.P_Name, D.Dr_Name FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE P.P_City = D.Dr_City AND P.P_id != D.Dr_id;`

Q 20. Find the projects with the highest total budget. If multiple projects have the same highest budget, choose the one with the most employees.

- A. `SELECT Project_Name FROM Project ORDER BY Project_Budget DESC, (SELECT COUNT(*) FROM Work WHERE Project.Project_ID = Work.Project_ID) DESC LIMIT 1`
- B. `SELECT Project_Name FROM Project ORDER BY Project_Budget DESC, (SELECT COUNT(*) FROM Work WHERE Project.Project_ID = Work.Project_ID) ASC LIMIT 1`
- C. `SELECT Project_Name FROM Project ORDER BY Project_Budget ASC, (SELECT COUNT(*) FROM Work WHERE Project.Project_ID = Work.Project_ID) DESC LIMIT 1`
- D. `SELECT Project_Name FROM Project ORDER BY Project_Budget ASC, (SELECT COUNT(*) FROM Work WHERE Project.Project_ID = Work.Project_ID) ASC LIMIT 1`

Q 21. Retrieve the names of doctors who have treated the same patient more than once, and show the patient's name and the number of treatments.

- A. `SELECT D.Dr_Name, P.P_Name, COUNT(T.Treatment_id) AS Total_Treatments FROM Doctor D INNER JOIN Treatment T ON D.Dr_id = T.Dr_id INNER JOIN Patient P ON T.P_id =`

P.P\_id WHERE T.P\_id IN (SELECT T.P\_id FROM Treatment T GROUP BY T.P\_id HAVING COUNT(Treatment\_id) = 1) GROUP BY D.Dr\_Name, P.P\_Name;

B. SELECT D.Dr\_Name, P.P\_Name, COUNT(T.Treatment\_id) AS Total\_Treatments FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE T.P\_id IN (SELECT T.P\_id FROM Treatment T GROUP BY T.P\_id HAVING COUNT(T.Treatment\_id) > 1) GROUP BY D.Dr\_Name, P.P\_Name;

C. SELECT D.Dr\_Name, P.P\_Name, COUNT(T.Treatment\_id) AS Total\_Treatments FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE T.P\_id IN (SELECT T.P\_id FROM Treatment T GROUP BY T.P\_id HAVING COUNT(Treatment\_id) = 2) GROUP BY D.Dr\_Name, P.P\_Name;

D. SELECT D.Dr\_Name, P.P\_Name, COUNT(T.Treatment\_id) AS Total\_Treatments FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE T.P\_id NOT IN (SELECT T.P\_id FROM Treatment T GROUP BY T.P\_id HAVING COUNT(Treatment\_id) > 1) GROUP BY D.Dr\_Name, P.P\_Name;

Q 22. Calculate the average project duration in days, but if a project has a duration of more than 90 days, categorize it as 'Long-Term,' between 30 and 90 days as 'Medium-Term,' and less than 30 days as 'Short-Term.'

A. SELECT AVG(Project\_Duration), CASE WHEN AVG(Project\_Duration) >= 30 THEN 'Medium-Term' WHEN AVG(Project\_Duration) > 90 THEN 'Long-Term' ELSE 'Short-Term' END FROM Project

B. SELECT AVG(Project\_Duration), CASE WHEN AVG(Project\_Duration) >= 30 THEN 'Long-Term' WHEN AVG(Project\_Duration) > 90 THEN 'Medium-Term' ELSE 'Short-Term' END FROM Project

C. SELECT AVG(Project\_Duration), CASE WHEN AVG(Project\_Duration) > 90 THEN 'Long-Term' WHEN AVG(Project\_Duration) >= 30 THEN 'Medium-Term' ELSE 'Short-Term' END FROM Project

D. SELECT AVG(Project\_Duration), CASE WHEN AVG(Project\_Duration) > 90 THEN 'Short-Term' WHEN AVG(Project\_Duration) >= 30 THEN 'Medium-Term' ELSE 'Long-Term' END FROM Project

Q 23. Update the designation of employees with more than 10 years of experience to 'Senior Developer.'

A. UPDATE Employee SET Emp\_Designation = 'Senior Developer' WHERE Emp\_Expertiese < 10



- B. UPDATE Employee SET Emp\_Designation = 'Senior Developer' WHERE Emp\_Expertiese > 10
- C. UPDATE Employee SET Emp\_Designation = 'Senior Developer' WHERE Emp\_Expertiese >= 10
- D. UPDATE Employee SET Emp\_Designation = 'Senior Developer' WHERE Emp\_Expertiese = 10

Q 24. Find the projects that have at least one employee with expertise in 'Java.'

- A. SELECT Project\_Name FROM Project WHERE Project\_ID IN (SELECT Project\_ID FROM Work WHERE Emp\_ID IN (SELECT Emp\_ID FROM Employee WHERE Emp\_Expertiese = 'Java'))
- B. SELECT Project\_Name FROM Project WHERE Project\_ID IN (SELECT Project\_ID FROM Work WHERE Emp\_ID IN (SELECT Emp\_ID FROM Employee WHERE Emp\_Expertiese = 'Java'))
- C. SELECT Project\_Name FROM Project WHERE Project\_ID IN (SELECT Project\_ID FROM Work WHERE Emp\_ID IN (SELECT Emp\_ID FROM Employee WHERE Emp\_Expertiese LIKE '%Java%'))
- D. SELECT Project\_Name FROM Project WHERE Project\_ID IN (SELECT Project\_ID FROM Work WHERE Emp\_ID IN (SELECT Emp\_ID FROM Employee WHERE Emp\_Expertiese IN ('Java', 'Java Script')))

Q 25. Identify employees who have worked on projects with a budget exceeding \$100,000 and have more than 5 years of experience. Update their designation to 'Project Lead.'

- A. UPDATE Employee SET Emp\_Designation = 'Project Lead' WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work INNER JOIN Project ON Work.Project\_ID = Project.Project\_ID WHERE Project\_Budget >= 100000 AND Emp\_Expertiese >= 5)
- B. UPDATE Employee SET Emp\_Designation = 'Project Lead' WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work INNER JOIN Project ON Work.Project\_ID = Project.Project\_ID WHERE Project\_Budget >= 100000 AND Emp\_Expertiese > 5)
- C. UPDATE Employee SET Emp\_Designation = 'Project Lead' WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work INNER JOIN Project ON Work.Project\_ID = Project.Project\_ID WHERE Project\_Budget > 100000 AND Emp\_Expertiese >= 5)
- D. UPDATE Employee SET Emp\_Designation = 'Project Lead' WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work INNER JOIN Project ON Work.Project\_ID = Project.Project\_ID WHERE Project\_Budget > 100000 AND Emp\_Expertiese > 5)

Q 26. Find the projects that have employees with expertise in both 'Java' and 'Python.'

- A. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_ID FROM Work WHERE Emp_ID IN (SELECT Emp_ID FROM Employee WHERE Emp_Expertiese = 'Java')) AND Project_ID IN (SELECT Project_ID FROM Work WHERE Emp_ID IN (SELECT Emp_ID FROM Employee WHERE Emp_Expertiese = 'Python'))`
- B. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_ID FROM Work WHERE Emp_ID IN (SELECT Emp_ID FROM Employee WHERE Emp_Expertiese = 'Java') OR Emp_ID IN (SELECT Emp_ID FROM Employee WHERE Emp_Expertiese = 'Python'))`
- C. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_ID FROM Work WHERE Emp_ID IN (SELECT Emp_ID FROM Employee WHERE Emp_Expertiese = 'Java') AND Project_ID IN (SELECT Project_ID FROM Work WHERE Emp_ID IN (SELECT Emp_ID FROM Employee WHERE Emp_Expertiese = 'Python')))`
- D. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_ID FROM Work WHERE Emp_ID IN (SELECT Emp_ID FROM Employee WHERE Emp_Expertiese = 'Java') AND Project_ID IN (SELECT Project_ID FROM Work WHERE Emp_ID IN (SELECT Emp_ID FROM Employee WHERE Emp_Expertiese = 'Python')))`

Q 27. Calculate the average salary for employees in each department, but if the average salary is below \$50,000, categorize the department as 'Low Pay,' between \$50,000 and \$75,000 as 'Average Pay,' and above \$75,000 as 'High Pay.'

- A. `SELECT Department, AVG(Emp_Salary), CASE WHEN AVG(Emp_Salary) < 50000 THEN 'Average Pay' WHEN AVG(Emp_Salary) >= 50000 AND AVG(Emp_Salary) <= 75000 THEN 'Low Pay' ELSE 'High Pay' END FROM Employee GROUP BY Department`
- B. `SELECT Department, AVG(Emp_Salary), CASE WHEN AVG(Emp_Salary) < 50000 THEN 'Low Pay' WHEN AVG(Emp_Salary) >= 50000 AND AVG(Emp_Salary) <= 75000 THEN 'Average Pay' ELSE 'High Pay' END FROM Employee GROUP BY Department`
- C. `SELECT Department, CASE WHEN AVG(Emp_Salary) < 50000 THEN 'High Pay' WHEN AVG(Emp_Salary) >= 50000 AND AVG(Emp_Salary) <= 75000 THEN 'Low Pay' ELSE 'Average Pay' END FROM Employee GROUP BY Department`
- D. `SELECT Department, CASE WHEN AVG(Emp_Salary) < 50000 THEN 'Low Pay' WHEN AVG(Emp_Salary) >= 50000 AND AVG(Emp_Salary) <= 75000 THEN 'High Pay' ELSE 'Average Pay' END FROM Employee GROUP BY Department`

Q 28. Calculate the total bonus for employees who have more than 3 years of experience. If the bonus is above \$2,000, categorize it as 'High Bonus,' between \$1,000 and \$2,000 as 'Medium Bonus,' and below \$1,000 as 'Low Bonus.'

- A. SELECT Emp\_Name, SUM(Emp\_Bonus), CASE WHEN SUM(Emp\_Bonus) > 2000 THEN 'High Bonus' WHEN SUM(Emp\_Bonus) >= 1000 THEN 'Medium Bonus' ELSE 'Low Bonus' END FROM Employee WHERE Emp\_Experience > 3 GROUP BY Emp\_Name
- B. SELECT Emp\_Name, SUM(Emp\_Bonus), CASE WHEN SUM(Emp\_Bonus) >= 1000 THEN 'Medium Bonus' WHEN SUM(Emp\_Bonus) > 2000 THEN 'Low Bonus' ELSE 'High Bonus' END FROM Employee WHERE Emp\_Experience > 3 GROUP BY Emp\_Name
- C. SELECT Emp\_Name, SUM(Emp\_Bonus), CASE WHEN SUM(Emp\_Bonus) > 1000 THEN 'Low Bonus' WHEN SUM(Emp\_Bonus) >= 2000 THEN 'Medium Bonus' ELSE 'High Bonus' END FROM Employee WHERE Emp\_Experience > 3 GROUP BY Emp\_Name
- D. SELECT Emp\_Name, SUM(Emp\_Bonus), CASE WHEN SUM(Emp\_Bonus) >= 1000 THEN 'Low Bonus' WHEN SUM(Emp\_Bonus) > 2000 THEN 'Medium Bonus' ELSE 'High Bonus' END FROM Employee WHERE Emp\_Experience > 3 GROUP BY Emp\_Name

Q 29. Identify the patients who have been treated by doctors from the same city and specialization as the patient, and show the names of the doctors.

- A. SELECT P.P\_Name, D.Dr\_Name FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_City <> D.Dr\_City AND P.P\_Specialization <> D.Dr\_Specialization GROUP BY P.P\_Name, D.Dr\_Name;
- B. SELECT P.P\_Name, D.Dr\_Name FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_City = D.Dr\_City AND P.P\_Specialization <> D.Dr\_Specialization GROUP BY P.P\_Name, D.Dr\_Name;
- C. SELECT P.P\_Name, D.Dr\_Name FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_City = D.Dr\_City AND P.P\_Specialization = D.Dr\_Specialization GROUP BY P.P\_Name, D.Dr\_Name;
- D. SELECT P.P\_Name, D.Dr\_Name FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_City <> D.Dr\_City AND P.P\_Specialization = D.Dr\_Specialization GROUP BY P.P\_Name, D.Dr\_Name;

Q 30. List the employees who have not been assigned to any projects yet.

- A. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IS NULL
- B. SELECT Emp\_Name FROM Employee WHERE Emp\_ID NOT IN (SELECT Emp\_ID FROM Work)
- C. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work)
- D. SELECT Emp\_Name FROM Employee WHERE Emp\_ID = NULL

Q 31. List the doctors who have treated patients from 'New York City' and have performed more than 3 different tests.

- A. SELECT DISTINCT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'New York City' HAVING COUNT(DISTINCT TS.Test\_Name) <= 3;
- B. SELECT DISTINCT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'New York City' GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(DISTINCT TS.Test\_Name) <= 3;
- C. SELECT DISTINCT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'New York City' HAVING COUNT(DISTINCT TS.Test\_Name) > 3;
- D. SELECT DISTINCT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'New York City' GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(DISTINCT TS.Test\_Name) > 3;

Q 32. Retrieve the names of patients who have not had any tests and are treated by doctors with less than 5 years of experience.

- A. SELECT P.P\_Name FROM Patient P WHERE P.P\_id NOT IN (SELECT T.P\_id FROM Test T) AND P.P\_id NOT IN (SELECT D.P\_id FROM Doctor D WHERE D.Dr\_Experience < 5);
- B. SELECT P.P\_Name FROM Patient P WHERE P.P\_id IN (SELECT T.P\_id FROM Test T) AND P.P\_id NOT IN (SELECT D.P\_id FROM Doctor D WHERE D.Dr\_Experience < 5);
- C. SELECT P.P\_Name FROM Patient P WHERE P.P\_id IN (SELECT T.P\_id FROM Test T) AND P.P\_id IN (SELECT D.P\_id FROM Doctor D WHERE D.Dr\_Experience < 5);
- D. SELECT P.P\_Name FROM Patient P WHERE P.P\_id NOT IN (SELECT T.P\_id FROM Test T) AND P.P\_id IN (SELECT D.P\_id FROM Doctor D WHERE D.Dr\_Experience < 5);

Q 33. Find the employees who have the same designation as their project category.

- A. `SELECT Emp_Name FROM Employee WHERE Emp_Designation = (SELECT Project_Categoery FROM Project WHERE Project_ID = Work.Project_ID) AND Emp_ID = Work.Emp_ID`
- B. `SELECT Emp_Name FROM Employee WHERE Emp_Designation = (SELECT Project_Categoery FROM Project WHERE Project_ID = Work.Project_ID) AND Emp_ID = (SELECT Emp_ID FROM Work WHERE Emp_Designation = Project_Categoery)`
- C. `SELECT Emp_Name FROM Employee WHERE Emp_Designation = (SELECT Project_Categoery FROM Project WHERE Project_ID = Work.Project_ID)`
- D. `SELECT Emp_Name FROM Employee WHERE Emp_Designation = (SELECT Project_Categoery FROM Project WHERE Project_ID = Work.Project_ID) AND Emp_ID = (SELECT Emp_ID FROM Work WHERE Emp_Designation = Project_Categoery)`

Q 34. Find the total number of patients admitted by each doctor, but only if the doctor has treated more than 10 patients.

- A. `SELECT D.Dr_Name, COUNT(T.P_id) AS Total_Patients FROM Doctor D LEFT JOIN Treatment T ON D.Dr_id = T.Dr_id GROUP BY D.Dr_id, D.Dr_Name HAVING COUNT(T.P_id) >= 10;`
- B. `SELECT D.Dr_Name, COUNT(T.P_id) AS Total_Patients FROM Doctor D LEFT JOIN Treatment T ON D.Dr_id = T.Dr_id GROUP BY D.Dr_id, D.Dr_Name HAVING COUNT(T.P_id) <= 10;`
- C. `SELECT D.Dr_Name, COUNT(T.P_id) AS Total_Patients FROM Doctor D LEFT JOIN Treatment T ON D.Dr_id = T.Dr_id GROUP BY D.Dr_id, D.Dr_Name HAVING COUNT(T.P_id) > 10;`
- D. `SELECT D.Dr_Name, COUNT(T.P_id) AS Total_Patients FROM Doctor D LEFT JOIN Treatment T ON D.Dr_id = T.Dr_id GROUP BY D.Dr_id, D.Dr_Name HAVING COUNT(T.P_id) < 10;`

Q 35. Find the patients who have been treated by doctors from the 'Bay Area' and have had both 'X-ray' and 'MRI' tests.

- A. SELECT P.P\_Name, P.P\_id FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id  
INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE  
P.P\_City = 'Bay Area' AND TS.Test\_Name IN ('X-ray', 'MRI') GROUP BY P.P\_id, P.P\_Name  
HAVING COUNT(DISTINCT TS.Test\_Name) = 2;
- B. SELECT P.P\_Name, P.P\_id FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id  
INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE  
P.P\_City = 'Bay Area' AND TS.Test\_Name IN ('X-ray', 'CT Scan') GROUP BY P.P\_id, P.P\_Name  
HAVING COUNT(DISTINCT TS.Test\_Name) = 2;
- C. SELECT P.P\_Name, P.P\_id FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id  
INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE  
P.P\_City = 'New York City' AND TS.Test\_Name IN ('X-ray', 'MRI') GROUP BY P.P\_id, P.P\_Name  
HAVING COUNT(DISTINCT TS.Test\_Name) = 2;
- D. SELECT P.P\_Name, P.P\_id FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id  
INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE  
P.P\_City = 'Bay Area' AND TS.Test\_Name IN ('X-ray', 'MRI') GROUP BY P.P\_id, P.P\_Name  
HAVING COUNT(DISTINCT TS.Test\_Name) = 1;

Q 36. Calculate the total cost of treatment for patients who have been admitted more than once and are treated by doctors specializing in 'Surgery.'

- A. SELECT T.P\_id, SUM(Treatment\_Cost) AS Total\_Cost FROM Treatment T INNER JOIN  
Doctor D ON T.Dr\_id = D.Dr\_id WHERE D.Dr\_specialization = 'Oncology' AND T.P\_id IN  
(SELECT P\_id FROM Treatment GROUP BY P\_id HAVING COUNT(Treatment\_id) <= 1) GROUP  
BY T.P\_id;
- B. SELECT T.P\_id, SUM(Treatment\_Cost) AS Total\_Cost FROM Treatment T INNER JOIN  
Doctor D ON T.Dr\_id = D.Dr\_id WHERE D.Dr\_specialization = 'Surgery' AND T.P\_id IN (SELECT  
P\_id FROM Treatment GROUP BY P\_id HAVING COUNT(Treatment\_id) <= 1) GROUP BY T.P\_id;
- C. SELECT T.P\_id, SUM(Treatment\_Cost) AS Total\_Cost FROM Treatment T INNER JOIN  
Doctor D ON T.Dr\_id = D.Dr\_id WHERE D.Dr\_specialization = 'Surgery' AND T.P\_id IN (SELECT  
P\_id FROM Treatment GROUP BY P\_id HAVING COUNT(Treatment\_id) > 1) GROUP BY T.P\_id;
- D. SELECT T.P\_id, SUM(Treatment\_Cost) AS Total\_Cost FROM Treatment T INNER JOIN  
Doctor D ON T.Dr\_id = D.Dr\_id WHERE D.Dr\_specialization = 'Oncology' AND T.P\_id IN  
(SELECT P\_id FROM Treatment GROUP BY P\_id HAVING COUNT(Treatment\_id) > 1) GROUP BY  
T.P\_id;

Q 37. Find the doctors who have not treated patients with 'Diabetes' and specialize in 'Cardiology' or 'Endocrinology.'

- A. `SELECT D.Dr_Name, D.Dr_id FROM Doctor D WHERE D.Dr_specialization IN ('Cardiology', 'Endocrinology') AND D.Dr_id IN (SELECT DISTINCT T.Dr_id FROM Treatment T INNER JOIN Patient P ON T.P_id = P.P_id WHERE P.P_Disease = 'Diabetes');`
- B. `SELECT D.Dr_Name, D.Dr_id FROM Doctor D WHERE D.Dr_specialization IN ('Cardiology', 'Endocrinology') AND D.Dr_id NOT IN (SELECT DISTINCT T.Dr_id FROM Treatment T INNER JOIN Patient P ON T.P_id = P.P_id WHERE P.P_Disease = 'Diabetes');`
- C. `SELECT D.Dr_Name, D.Dr_id FROM Doctor D WHERE D.Dr_specialization IN ('Oncology', 'Surgery') AND D.Dr_id NOT IN (SELECT DISTINCT T.Dr_id FROM Treatment T INNER JOIN Patient P ON T.P_id = P.P_id WHERE P.P_Disease = 'Diabetes');`
- D. `SELECT D.Dr_Name, D.Dr_id FROM Doctor D WHERE D.Dr_specialization IN ('Oncology', 'Surgery') AND D.Dr_id IN (SELECT DISTINCT T.Dr_id FROM Treatment T INNER JOIN Patient P ON T.P_id = P.P_id WHERE P.P_Disease = 'Diabetes');`

Q 38. Retrieve the projects with the highest number of employees working on them.

- A. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_id FROM Work GROUP BY Project_id ORDER BY COUNT(Emp_ID) DESC LIMIT 1)`
- B. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_id FROM Work GROUP BY Project_id HAVING COUNT(Emp_ID) = SUM(COUNT(Emp_ID)))`
- C. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_id FROM Work GROUP BY Project_id HAVING COUNT(Emp_ID) = MIN(COUNT(Emp_ID)))`
- D. `SELECT Project_Name FROM Project WHERE Project_ID IN (SELECT Project_id FROM Work GROUP BY Project_id HAVING COUNT(Emp_ID) = MAX(COUNT(Emp_ID)))`

Q 39. List the employees who are working on projects with a specific technology, e.g., 'SQL Server.'

- A. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Technology = 'SQL Server')`
- B. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Technology LIKE '%SQL Server%')`
- C. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Technology IN ('SQL Server', 'SQL'))`

D. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work WHERE Technology = 'MySQL')

Q 40. Retrieve the employee names who are working on projects that are not in the 'Research' category.

A. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work WHERE Project\_id IN (SELECT Project\_ID FROM Project WHERE Project\_Categoery = 'Development'))

B. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work WHERE Project\_id IN (SELECT Project\_ID FROM Project WHERE Project\_Categoery <> 'Research'))

C. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work WHERE Project\_id IN (SELECT Project\_ID FROM Project WHERE Project\_Categoery = 'Development'))

D. SELECT Emp\_Name FROM Employee WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work WHERE Project\_id NOT IN (SELECT Project\_ID FROM Project WHERE Project\_Categoery = 'Research'))

Q 41. List the doctors who have not treated patients from 'Los Angeles' and have performed more than 5 tests in total.

A. SELECT D.Dr\_Name FROM Doctor D LEFT JOIN Treatment T ON D.Dr\_id = T.Dr\_id LEFT JOIN Patient P ON T.P\_id = P.P\_id LEFT JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'Los Angeles' GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(DISTINCT TS.Test\_id) <= 5;

B. SELECT D.Dr\_Name FROM Doctor D LEFT JOIN Treatment T ON D.Dr\_id = T.Dr\_id LEFT JOIN Patient P ON T.P\_id = P.P\_id LEFT JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City <> 'Los Angeles' OR P.P\_id IS NULL GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(DISTINCT TS.Test\_id) > 5;

C. SELECT D.Dr\_Name FROM Doctor D LEFT JOIN Treatment T ON D.Dr\_id = T.Dr\_id LEFT JOIN Patient P ON T.P\_id = P.P\_id LEFT JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'Los Angeles' GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(DISTINCT TS.Test\_id) > 5;

D. SELECT D.Dr\_Name FROM Doctor D LEFT JOIN Treatment T ON D.Dr\_id = T.Dr\_id LEFT JOIN Patient P ON T.P\_id = P.P\_id LEFT JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City <> 'Los Angeles' OR P.P\_id IS NULL GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(DISTINCT TS.Test\_id) <= 5;



Q 42. Find the projects where the total budget exceeds \$100,000, and categorize them as 'High Budget.' If the budget is between \$50,000 and \$100,000, categorize them as 'Medium Budget,' and below \$50,000 as 'Low Budget.'

A. SELECT Project\_Name, CASE WHEN Project\_Budget >= 50000 AND Project\_Budget <= 100000 THEN 'Low Budget' WHEN Project\_Budget > 100000 THEN 'High Budget' ELSE 'Medium Budget' END FROM Project

B. SELECT Project\_Name, CASE WHEN Project\_Budget >= 50000 AND Project\_Budget <= 100000 THEN 'High Budget' WHEN Project\_Budget > 100000 THEN 'Medium Budget' ELSE 'Low Budget' END FROM Project

C. SELECT Project\_Name, CASE WHEN Project\_Budget > 100000 THEN 'High Budget' WHEN Project\_Budget >= 50000 AND Project\_Budget <= 100000 THEN 'Medium Budget' ELSE 'Low Budget' END FROM Project

D. SELECT Project\_Name, CASE WHEN Project\_Budget >= 50000 AND Project\_Budget <= 100000 THEN 'Medium Budget' WHEN Project\_Budget > 100000 THEN 'High Budget' ELSE 'Low Budget' END FROM Project

Q 43. List the patients who have had multiple treatments and show the names of their treating doctors for each treatment.

A. SELECT P.P\_Name, T.P\_id, D.Dr\_Name, T.Dr\_id FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE T.P\_id IN (SELECT T.P\_id FROM Treatment T GROUP BY T.P\_id HAVING COUNT(Treatment\_id) > 1);

B. SELECT P.P\_Name, T.P\_id, D.Dr\_Name, T.Dr\_id FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE T.P\_id IN (SELECT T.P\_id FROM Treatment T GROUP BY T.P\_id HAVING COUNT(Treatment\_id) >= 2);

C. SELECT P.P\_Name, T.P\_id, D.Dr\_Name, T.Dr\_id FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE T.P\_id IN (SELECT T.P\_id FROM Treatment T GROUP BY T.P\_id HAVING COUNT(Treatment\_id) <= 1);

D. SELECT P.P\_Name, T.P\_id, D.Dr\_Name, T.Dr\_id FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE T.P\_id IN (SELECT T.P\_id FROM Treatment T GROUP BY T.P\_id HAVING COUNT(Treatment\_id) < 2);

Q 44. Retrieve a list of employees with their project details (Emp\_Name, Project\_Name).

- A. SELECT Emp\_Name, Project\_Name FROM Employee RIGHT JOIN Work ON Employee.Emp\_ID = Work.Emp\_ID RIGHT JOIN Project ON Work.Project\_ID = Project.Project\_ID
- B. SELECT Emp\_Name, Project\_Name FROM Employee INNER JOIN Work ON Employee.Emp\_ID = Work.Emp\_ID INNER JOIN Project ON Work.Project\_ID = Project.Project\_ID
- C. SELECT Emp\_Name, Project\_Name FROM Employee LEFT JOIN Work ON Employee.Emp\_ID = Work.Emp\_ID LEFT JOIN Project ON Work.Project\_ID = Project.Project\_ID
- D. SELECT Emp\_Name, Project\_Name FROM Employee CROSS JOIN Work CROSS JOIN Project

Q 45. Retrieve the names of employees who have more than 5 years of experience.

- A. SELECT Emp\_Name FROM Employee WHERE Emp\_Expertiese = 5
- B. SELECT Emp\_Name FROM Employee WHERE Emp\_Expertiese < 5
- C. SELECT Emp\_Name FROM Employee WHERE Emp\_Expertiese > 5
- D. SELECT Emp\_Name FROM Employee WHERE Emp\_Expertiese >= 5

Q 46. Calculate the total salary of each employee, including a bonus of \$500 for employees with the designation 'Manager' and \$200 for employees with the designation 'Senior Developer.'

- A. SELECT Emp\_Name, Emp\_Salary + CASE WHEN Emp\_Desigation = 'Manager' THEN 500 WHEN Emp\_Desigation = 'Senior Developer' THEN 200 ELSE 0 END FROM Employee
- B. SELECT Emp\_Name, Emp\_Salary + CASE WHEN Emp\_Desigation = 'Senior Developer' THEN 200 WHEN Emp\_Desigation = 'Manager' THEN 500 ELSE 0 END FROM Employee
- C. SELECT Emp\_Name, Emp\_Salary + CASE WHEN Emp\_Desigation = 'Manager' THEN 200 WHEN Emp\_Desigation = 'Senior Developer' THEN 500 ELSE 0 END FROM Employee
- D. SELECT Emp\_Name, Emp\_Salary + CASE WHEN Emp\_Desigation = 'Senior Developer' THEN 500 WHEN Emp\_Desigation = 'Manager' THEN 200 ELSE 0 END FROM Employee

Q 47. Retrieve the projects with their respective statuses, but if a project started before 2022, categorize it as 'Old Project,' otherwise as 'Recent Project.'

- A. `SELECT Project_Name, CASE WHEN Project_Start_Date < '2022-01-01' THEN 'Old Project' ELSE 'Recent Project' END FROM Project`
- B. `SELECT Project_Name, CASE WHEN Project_Start_Date < '2022-01-01' THEN 'Recent Project' ELSE 'Old Project' END FROM Project`
- C. `SELECT Project_Name, CASE WHEN Project_Start_Date > '2022-01-01' THEN 'Old Project' ELSE 'Recent Project' END FROM Project`
- D. `SELECT Project_Name, CASE WHEN Project_Start_Date > '2022-01-01' THEN 'Recent Project' ELSE 'Old Project' END FROM Project`

Q 48. List the employees who are assigned to projects with the word 'Database' in their project name.

- A. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id IN (SELECT Project_ID FROM Project WHERE Project_Name LIKE '%Database%'))`
- B. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id = (SELECT Project_ID FROM Project WHERE Project_Name LIKE 'Database'))`
- C. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id = (SELECT Project_ID FROM Project WHERE Project_Name = 'Database'))`
- D. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id = (SELECT Project_ID FROM Project WHERE Project_Name IN ('Database', 'Data Storage')))`

Q 49. Calculate the average treatment cost for patients who have been admitted more than once and are treated by doctors specializing in 'Cardiology.'

- A. `SELECT AVG(Treatment_Cost) AS Avg_Treatment_Cost FROM Treatment T INNER JOIN Patient P ON T.P_id = P.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE P.P_id IN (SELECT P_id FROM Treatment GROUP BY P_id HAVING COUNT(Treatment_id) > 1) AND D.Dr_Specialization = 'Cardiology';`

B. SELECT AVG(Treatment\_Cost) AS Avg\_Treatment\_Cost FROM Treatment T INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_id IN (SELECT P\_id FROM Treatment GROUP BY P\_id HAVING COUNT(Treatment\_id) <= 1) AND D.Dr\_Specialization <> 'Cardiology';

C. SELECT AVG(Treatment\_Cost) AS Avg\_Treatment\_Cost FROM Treatment T INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_id IN (SELECT P\_id FROM Treatment GROUP BY P\_id HAVING COUNT(Treatment\_id) <= 1) AND D.Dr\_Specialization = 'Cardiology';

D. SELECT AVG(Treatment\_Cost) AS Avg\_Treatment\_Cost FROM Treatment T INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_id IN (SELECT P\_id FROM Treatment GROUP BY P\_id HAVING COUNT(Treatment\_id) > 1) AND D.Dr\_Specialization <> 'Cardiology';

Q 50. Retrieve the names of doctors who have treated patients from the 'Bay Area' and have performed at least one 'MRI' test.

A. SELECT DISTINCT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'New York City' AND TS.Test\_Name = 'CT Scan';

B. SELECT DISTINCT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'Bay Area' AND TS.Test\_Name = 'MRI';

C. SELECT DISTINCT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'New York City' AND TS.Test\_Name = 'MRI';

D. SELECT DISTINCT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_City = 'Bay Area' AND TS.Test\_Name = 'CT Scan';

Q 51. Calculate the average treatment duration for patients admitted by doctors specializing in 'Oncology' and who have had more than one treatment.

A. SELECT AVG(Days) AS Avg\_Duration FROM (SELECT P\_id, D.Dr\_id, DATEDIFF(Admit\_Date, MIN(Admit\_Date)) AS Days FROM Treatment T INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE D.Dr\_specialization = 'Oncology' GROUP BY T.P\_id, D.Dr\_id HAVING COUNT(Treatment\_id) > 1) AS Subquery;

B. SELECT AVG(Days) AS Avg\_Duration FROM (SELECT P\_id, D.Dr\_id, DATEDIFF(Admit\_Date, MIN(Admit\_Date)) AS Days FROM Treatment T INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE D.Dr\_specialization = 'Cardiology' GROUP BY T.P\_id, D.Dr\_id HAVING COUNT(Treatment\_id) > 1) AS Subquery;

C. SELECT AVG(Days) AS Avg\_Duration FROM (SELECT P\_id, D.Dr\_id, DATEDIFF(Admit\_Date, MIN(Admit\_Date)) AS Days FROM Treatment T INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE D.Dr\_specialization = 'Cardiology' GROUP BY T.P\_id, D.Dr\_id HAVING COUNT(Treatment\_id) <= 1) AS Subquery;

D. SELECT AVG(Days) AS Avg\_Duration FROM (SELECT P\_id, D.Dr\_id, DATEDIFF(Admit\_Date, MIN(Admit\_Date)) AS Days FROM Treatment T INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE D.Dr\_specialization = 'Oncology' GROUP BY T.P\_id, D.Dr\_id HAVING COUNT(Treatment\_id) <= 1) AS Subquery;

Q 52. Identify the patients with the highest number of admissions, and show the details of their treating doctors.

A. SELECT P.P\_Name, P.P\_id, D.Dr\_Name, D.Dr\_id FROM Patient P INNER JOIN (SELECT T.P\_id, T.Dr\_id, COUNT(Treatment\_id) AS Admissions FROM Treatment T GROUP BY T.P\_id, T.Dr\_id HAVING Admissions = (SELECT MAX(Admissions) FROM (SELECT P\_id, Dr\_id, COUNT(Treatment\_id) AS Admissions FROM Treatment T GROUP BY P\_id, Dr\_id) AS Subquery)) AS Subquery ON P.P\_id = Subquery.P\_id INNER JOIN Doctor D ON Subquery.Dr\_id = D.Dr\_id;

B. SELECT P.P\_Name, P.P\_id, D.Dr\_Name, D.Dr\_id FROM Patient P INNER JOIN (SELECT T.P\_id, T.Dr\_id, COUNT(Treatment\_id) AS Admissions FROM Treatment T GROUP BY T.P\_id, T.Dr\_id HAVING Admissions = (SELECT MIN(Admissions) FROM (SELECT P\_id, Dr\_id, COUNT(Treatment\_id) AS Admissions FROM Treatment T GROUP BY P\_id, Dr\_id) AS Subquery)) AS Subquery ON P.P\_id = Subquery.P\_id INNER JOIN Doctor D ON Subquery.Dr\_id = D.Dr\_id;

C. SELECT P.P\_Name, P.P\_id, D.Dr\_Name, D.Dr\_id FROM Patient P INNER JOIN (SELECT T.P\_id, T.Dr\_id, COUNT(Treatment\_id) AS Admissions FROM Treatment T GROUP BY T.P\_id, T.Dr\_id HAVING Admissions = (SELECT MAX(Admissions) FROM (SELECT P\_id, Dr\_id, COUNT(Treatment\_id) AS Admissions FROM Treatment T GROUP BY P\_id, Dr\_id) AS Subquery)) AS Subquery ON P.P\_id = Subquery.P\_id INNER JOIN Doctor D ON Subquery.Dr\_id = D.Dr\_id;

D. SELECT P.P\_Name, P.P\_id, D.Dr\_Name, D.Dr\_id FROM Patient P INNER JOIN (SELECT T.P\_id, T.Dr\_id, COUNT(Treatment\_id) AS Admissions FROM Treatment T GROUP BY T.P\_id, T.Dr\_id HAVING Admissions = (SELECT AVG(Admissions) FROM (SELECT P\_id, Dr\_id, COUNT(Treatment\_id) AS Admissions FROM Treatment T GROUP BY P\_id, Dr\_id) AS Subquery)) AS Subquery ON P.P\_id = Subquery.P\_id INNER JOIN Doctor D ON Subquery.Dr\_id = D.Dr\_id;

Q 53. Calculate the average number of tests performed on patients who are allergic to 'Peanuts' and are treated by doctors with a specialization of 'Allergy.'

- A. `SELECT AVG(Num_Tests) AS Avg_Tests FROM (SELECT P.P_id, COUNT(TS.Test_Name) AS Num_Tests FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id INNER JOIN Test TS ON P.P_id = TS.P_id WHERE P.P_Allergies LIKE '%Peanuts%' AND D.Dr_specialization = 'Orthopedics' GROUP BY P.P_id) AS Subquery;`
- B. `SELECT AVG(Num_Tests) AS Avg_Tests FROM (SELECT P.P_id, COUNT(TS.Test_Name) AS Num_Tests FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id INNER JOIN Test TS ON P.P_id = TS.P_id WHERE P.P_Allergies LIKE '%Peanuts%' AND D.Dr_specialization = 'Pediatrics' GROUP BY P.P_id) AS Subquery;`
- C. `SELECT AVG(Num_Tests) AS Avg_Tests FROM (SELECT P.P_id, COUNT(TS.Test_Name) AS Num_Tests FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id INNER JOIN Test TS ON P.P_id = TS.P_id WHERE P.P_Allergies LIKE '%Peanuts%' AND D.Dr_specialization = 'Cardiology' GROUP BY P.P_id) AS Subquery;`
- D. `SELECT AVG(Num_Tests) AS Avg_Tests FROM (SELECT P.P_id, COUNT(TS.Test_Name) AS Num_Tests FROM Patient P INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id INNER JOIN Test TS ON P.P_id = TS.P_id WHERE P.P_Allergies LIKE '%Peanuts%' AND D.Dr_specialization = 'Allergy' GROUP BY P.P_id) AS Subquery;`

Q 54. Find the projects where the project category is 'Development.'

- A. `SELECT Project_Name FROM Project WHERE Project_Categoery IN ('Development', 'Software')`
- B. `SELECT Project_Name FROM Project WHERE Project_Categoery = 'Research'`
- C. `SELECT Project_Name FROM Project WHERE Project_Categoery LIKE 'Develop%'`
- D. `SELECT Project_Name FROM Project WHERE Project_Categoery = 'Development'`

Q 55. Calculate the average project cost, but if a project cost is over \$50,000, categorize it as 'Expensive,' between \$20,000 and \$50,000 as 'Moderate,' and below \$20,000 as 'Inexpensive.'

- A. `SELECT AVG(Project_Cost), CASE WHEN AVG(Project_Cost) > 50000 THEN 'Expensive' WHEN AVG(Project_Cost) >= 20000 THEN 'Moderate' ELSE 'Inexpensive' END FROM Project`

- B. `SELECT AVG(Project_Cost), CASE WHEN AVG(Project_Cost) > 50000 THEN 'Inexpensive' WHEN AVG(Project_Cost) >= 20000 THEN 'Moderate' ELSE 'Expensive' END FROM Project`
- C. `SELECT AVG(Project_Cost), CASE WHEN AVG(Project_Cost) > 50000 THEN 'Moderate' WHEN AVG(Project_Cost) >= 20000 THEN 'Expensive' ELSE 'Inexpensive' END FROM Project`
- D. `SELECT AVG(Project_Cost), CASE WHEN AVG(Project_Cost) > 50000 THEN 'Moderate' WHEN AVG(Project_Cost) >= 20000 THEN 'Inexpensive' ELSE 'Expensive' END FROM Project`

Q 56. Find the projects and their categories along with the employee assigned to each project.

- A. `SELECT Project_Name, Project_Categoery, Emp_Name FROM Project CROSS JOIN Work CROSS JOIN Employee`
- B. `SELECT Project_Name, Project_Categoery, Emp_Name FROM Project RIGHT JOIN Work ON Project.Project_ID = Work.Project_ID RIGHT JOIN Employee ON Work.Emp_ID = Employee.Emp_ID`
- C. `SELECT Project_Name, Project_Categoery, Emp_Name FROM Project INNER JOIN Work ON Project.Project_ID = Work.Project_ID INNER JOIN Employee ON Work.Emp_ID = Employee.Emp_ID`
- D. `SELECT Project_Name, Project_Categoery, Emp_Name FROM Project LEFT JOIN Work ON Project.Project_ID = Work.Project_ID LEFT JOIN Employee ON Work.Emp_ID = Employee.Emp_ID`

Q 57. Retrieve the names of patients who have had more than 5 different tests and are treated by doctors with more than 15 years of experience.

- A. `SELECT P.P_Name FROM Patient P INNER JOIN (SELECT P_id, COUNT(DISTINCT Test_Name) AS Num_Tests FROM Test GROUP BY P_id HAVING Num_Tests > 5) AS Subquery ON P.P_id = Subquery.P_id INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE D.Dr_Experience <= 15;`
- B. `SELECT P.P_Name FROM Patient P INNER JOIN (SELECT P_id, COUNT(DISTINCT Test_Name) AS Num_Tests FROM Test GROUP BY P_id HAVING Num_Tests <= 5) AS Subquery ON P.P_id = Subquery.P_id INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE D.Dr_Experience > 15;`
- C. `SELECT P.P_Name FROM Patient P INNER JOIN (SELECT P_id, COUNT(DISTINCT Test_Name) AS Num_Tests FROM Test GROUP BY P_id HAVING Num_Tests > 5) AS Subquery ON P.P_id = Subquery.P_id INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE D.Dr_Experience > 15;`

D. `SELECT P.P_Name FROM Patient P INNER JOIN (SELECT P_id, COUNT(DISTINCT Test_Name) AS Num_Tests FROM Test GROUP BY P_id HAVING Num_Tests <= 5) AS Subquery ON P.P_id = Subquery.P_id INNER JOIN Treatment T ON P.P_id = T.P_id INNER JOIN Doctor D ON T.Dr_id = D.Dr_id WHERE D.Dr_Experience <= 15;`

Q 58. List the employees who have worked on projects with 'Database' in the project name and are designated as 'Developer.'

A. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id IN (SELECT Project_ID FROM Project WHERE Project_Name LIKE '%Database%')) AND Emp_Designation = 'Developer'`

B. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id IN (SELECT Project_ID FROM Project WHERE Project_Name LIKE '%Database%')) AND Emp_Designation = 'Developer'`

C. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id IN (SELECT Project_ID FROM Project WHERE Project_Name = 'Database') AND Emp_Designation = 'Developer')`

D. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work WHERE Project_id IN (SELECT Project_ID FROM Project WHERE Project_Name = 'Database')) AND Emp_Designation = 'Developer'`

Q 59. List all employees who have worked on projects with a total duration of more than 180 days and have designations of 'Manager' or 'Senior Developer.'

A. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work GROUP BY Emp_ID HAVING SUM(Project_Duration) > 180) AND (Emp_Designation = 'Junior Developer' OR Emp_Designation = 'Manager')`

B. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work GROUP BY Emp_ID HAVING SUM(Project_Duration) > 180) AND (Emp_Designation = 'Senior Developer' OR Emp_Designation = 'Junior Developer')`

C. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work GROUP BY Emp_ID HAVING SUM(Project_Duration) > 180) AND (Emp_Designation = 'Senior Developer' OR Emp_Designation = 'Manager')`

D. `SELECT Emp_Name FROM Employee WHERE Emp_ID IN (SELECT Emp_ID FROM Work GROUP BY Emp_ID HAVING SUM(Project_Duration) > 180) AND (Emp_Designation = 'Manager' OR Emp_Designation = 'Junior Developer')`



Q 60. Find the employees who have the same expertise as another employee with Emp\_ID 101.

- A. `SELECT Emp_Name FROM Employee WHERE Emp_Expertiese IN (SELECT DISTINCT Emp_Expertiese FROM Employee WHERE Emp_ID = 101) AND Emp_ID <> 101`
- B. `SELECT Emp_Name FROM Employee WHERE Emp_Expertiese IN (SELECT Emp_Expertiese FROM Employee WHERE Emp_ID = 101) AND Emp_ID <> 101`
- C. `SELECT Emp_Name FROM Employee WHERE Emp_Expertiese = (SELECT Emp_Expertiese FROM Employee WHERE Emp_ID = 101) AND Emp_ID <> 101`
- D. `SELECT Emp_Name FROM Employee WHERE Emp_Expertiese = (SELECT DISTINCT Emp_Expertiese FROM Employee WHERE Emp_ID = 101) AND Emp_ID <> 101`

Q 61. Calculate the average years of experience for all employees and categorize them as 'Junior,' 'Intermediate,' or 'Senior.'

- A. `SELECT AVG(Emp_Expertiese) AS Average_Experience, CASE WHEN AVG(Emp_Expertiese) < 5 THEN 'Junior' WHEN AVG(Emp_Expertiese) >= 5 AND AVG(Emp_Expertiese) <= 10 THEN 'Intermediate' ELSE 'Senior' END AS Experience_Category FROM Employee`
- B. `SELECT AVG(Emp_Expertiese) AS Average_Experience, CASE WHEN AVG(Emp_Expertiese) < 5 THEN 'Junior' WHEN AVG(Emp_Expertiese) >= 5 AND AVG(Emp_Expertiese) <= 9 THEN 'Intermediate' ELSE 'Senior' END AS Experience_Category FROM Employee`
- C. `SELECT AVG(Emp_Expertiese) AS Average_Experience, CASE WHEN AVG(Emp_Expertiese) < 5 THEN 'Junior' WHEN AVG(Emp_Expertiese) >= 5 AND AVG(Emp_Expertiese) <= 15 THEN 'Intermediate' ELSE 'Senior' END AS Experience_Category FROM Employee`
- D. `SELECT AVG(Emp_Expertiese) AS Average_Experience, CASE WHEN AVG(Emp_Expertiese) < 6 THEN 'Junior' WHEN AVG(Emp_Expertiese) >= 6 AND AVG(Emp_Expertiese) <= 10 THEN 'Intermediate' ELSE 'Senior' END AS Experience_Category FROM Employee`

Q 62. List the patients who have had more tests than their age and show the total number of tests they've undergone.

- A. `SELECT P.P_Name, COUNT(T.Test_id) AS Total_Tests FROM Patient P INNER JOIN Test T ON P.P_id = T.P_id WHERE COUNT(T.Test_id) >= YEAR(CURRENT_DATE) - YEAR(P.P_Birthdate) GROUP BY P.P_Name;`

B. SELECT P.P\_Name, COUNT(T.Test\_id) AS Total\_Tests FROM Patient P INNER JOIN Test T ON P.P\_id = T.P\_id WHERE COUNT(T.Test\_id) <= YEAR(CURRENT\_DATE) - YEAR(P.P\_Birthdate) GROUP BY P.P\_Name;

C. SELECT P.P\_Name, COUNT(T.Test\_id) AS Total\_Tests FROM Patient P INNER JOIN Test T ON P.P\_id = T.P\_id WHERE COUNT(T.Test\_id) > YEAR(CURRENT\_DATE) - YEAR(P.P\_Birthdate) GROUP BY P.P\_Name;

D. SELECT P.P\_Name, COUNT(T.Test\_id) AS Total\_Tests FROM Patient P INNER JOIN Test T ON P.P\_id = T.P\_id WHERE COUNT(T.Test\_id) = YEAR(CURRENT\_DATE) - YEAR(P.P\_Birthdate) GROUP BY P.P\_Name;

Q 63. Find the doctors who have treated patients with 'Hypertension' and have performed more 'CT Scan' tests than 'X-ray' tests.

A. SELECT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_Disease = 'Hypertension' AND TS.Test\_Name = 'CT Scan' GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(TS.Test\_id) > (SELECT COUNT(TS.Test\_id) FROM Test TS WHERE TS.Test\_Name = 'X-ray');

B. SELECT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_Disease = 'Asthma' AND TS.Test\_Name = 'CT Scan' GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(TS.Test\_id) > (SELECT COUNT(TS.Test\_id) FROM Test TS WHERE TS.Test\_Name = 'X-ray');

C. SELECT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_Disease = 'Hypertension' AND TS.Test\_Name = 'X-ray' GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(TS.Test\_id) > (SELECT COUNT(TS.Test\_id) FROM Test TS WHERE TS.Test\_Name = 'CT Scan');

D. SELECT D.Dr\_Name FROM Doctor D INNER JOIN Treatment T ON D.Dr\_id = T.Dr\_id INNER JOIN Patient P ON T.P\_id = P.P\_id INNER JOIN Test TS ON P.P\_id = TS.P\_id WHERE P.P\_Disease = 'Asthma' AND TS.Test\_Name = 'X-ray' GROUP BY D.Dr\_id, D.Dr\_Name HAVING COUNT(TS.Test\_id) > (SELECT COUNT(TS.Test\_id) FROM Test TS WHERE TS.Test\_Name = 'CT Scan');

Q 64. Calculate the total treatment cost for patients who have been admitted on or after '2023-01-01' and have 'Diabetes.'

A. SELECT T.P\_id, SUM(Treatment\_Cost) AS Total\_Cost FROM Treatment T INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease = 'Cancer' AND T.Admit\_Date < '2023-01-01' GROUP BY T.P\_id;

B. SELECT T.P\_id, SUM(Treatment\_Cost) AS Total\_Cost FROM Treatment T INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease = 'Diabetes' AND T.Admit\_Date < '2023-01-01' GROUP BY T.P\_id;

C. SELECT T.P\_id, SUM(Treatment\_Cost) AS Total\_Cost FROM Treatment T INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease = 'Cancer' AND T.Admit\_Date >= '2023-01-01' GROUP BY T.P\_id;

D. SELECT T.P\_id, SUM(Treatment\_Cost) AS Total\_Cost FROM Treatment T INNER JOIN Patient P ON T.P\_id = P.P\_id WHERE P.P\_Disease = 'Diabetes' AND T.Admit\_Date >= '2023-01-01' GROUP BY T.P\_id;

Q 65. Identify employees who have worked on more than 3 projects and have designations of either 'Senior Developer' or 'Manager,' and grant them a performance bonus of \$1,000.

A. UPDATE Employee SET Emp\_Bonus = 1000 WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work GROUP BY Emp\_ID HAVING COUNT(Project\_ID) > 3) AND (Emp\_Designation = 'Junior Developer' OR Emp\_Designation = 'Manager')

B. UPDATE Employee SET Emp\_Bonus = 1000 WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work GROUP BY Emp\_ID HAVING COUNT(Project\_ID) > 3) AND (Emp\_Designation = 'Senior Developer' OR Emp\_Designation = 'Junior Developer')

C. UPDATE Employee SET Emp\_Bonus = 1000 WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work GROUP BY Emp\_ID HAVING COUNT(Project\_ID) = 3) AND (Emp\_Designation = 'Senior Developer' OR Emp\_Designation = 'Manager')

D. UPDATE Employee SET Emp\_Bonus = 1000 WHERE Emp\_ID IN (SELECT Emp\_ID FROM Work GROUP BY Emp\_ID HAVING COUNT(Project\_ID) > 3) AND (Emp\_Designation = 'Senior Developer' OR Emp\_Designation = 'Manager')

Q 66. List the patients who have been treated by doctors from 'San Francisco' and have had both 'Blood Test' and 'X-ray' tests.

A. SELECT P.P\_Name FROM Patient P WHERE P.P\_City = 'San Francisco' AND P.P\_id IN (SELECT P\_id FROM Test WHERE Test\_Name = 'Blood Test') AND P.P\_id IN (SELECT P\_id FROM Test WHERE Test\_Name = 'X-ray');

B. SELECT P.P\_Name FROM Patient P WHERE P.P\_City = 'San Francisco' AND P.P\_id NOT IN (SELECT P\_id FROM Test WHERE Test\_Name = 'Blood Test') AND P.P\_id NOT IN (SELECT P\_id FROM Test WHERE Test\_Name = 'X-ray');

C. SELECT P.P\_Name FROM Patient P WHERE P.P\_City = 'New York City' AND P.P\_id IN (SELECT P\_id FROM Test WHERE Test\_Name = 'Blood Test') AND P.P\_id IN (SELECT P\_id FROM Test WHERE Test\_Name = 'X-ray');

D. SELECT P.P\_Name FROM Patient P WHERE P.P\_City = 'New York City' AND P.P\_id NOT IN (SELECT P\_id FROM Test WHERE Test\_Name = 'Blood Test') AND P.P\_id NOT IN (SELECT P\_id FROM Test WHERE Test\_Name = 'X-ray');

Q 67. List the projects and their categories, but if the project category is 'Management,' display it as 'MGT,' 'Development' as 'DEV,' and 'Research' as 'RSR.'

A. SELECT Project\_Name, CASE Project\_Categoery WHEN 'Management' THEN 'DEV' WHEN 'Development' THEN 'MGT' WHEN 'Research' THEN 'RSR' ELSE Project\_Categoery END FROM Project

B. SELECT Project\_Name, CASE Project\_Categoery WHEN 'Development' THEN 'DEV' WHEN 'Management' THEN 'RSR' WHEN 'Research' THEN 'MGT' ELSE Project\_Categoery END FROM Project

C. SELECT Project\_Name, CASE Project\_Categoery WHEN 'Management' THEN 'MGT' WHEN 'Development' THEN 'DEV' WHEN 'Research' THEN 'RSR' ELSE Project\_Categoery END FROM Project

D. SELECT Project\_Name, CASE Project\_Categoery WHEN 'Development' THEN 'RSR' WHEN 'Management' THEN 'MGT' WHEN 'Research' THEN 'DEV' ELSE Project\_Categoery END FROM Project

Q 68. Calculate the average number of tests performed on patients from 'Los Angeles' who have had treatments by doctors with more than 10 years of experience.

A. SELECT AVG(COUNT(T.Test\_id)) AS Avg\_Tests FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_City = 'New York City' AND D.Dr\_Experience <= 10 GROUP BY P.P\_id;

B. SELECT AVG(COUNT(T.Test\_id)) AS Avg\_Tests FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_City = 'Los Angeles' AND D.Dr\_Experience > 10 GROUP BY P.P\_id;

C. SELECT AVG(COUNT(T.Test\_id)) AS Avg\_Tests FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_City = 'New York City' AND D.Dr\_Experience > 10 GROUP BY P.P\_id;

D. SELECT AVG(COUNT(T.Test\_id)) AS Avg\_Tests FROM Patient P INNER JOIN Treatment T ON P.P\_id = T.P\_id INNER JOIN Doctor D ON T.Dr\_id = D.Dr\_id WHERE P.P\_City = 'Los Angeles' AND D.Dr\_Experience <= 10 GROUP BY P.P\_id;

Q 69. Identify employees who have the highest salary in their respective departments and update their designations to 'Department Head.'

A. UPDATE Employee SET Emp\_Designation = 'Department Head' WHERE Emp\_Salary = (SELECT MAX(Emp\_Salary) FROM Employee)

B. UPDATE Employee SET Emp\_Designation = 'Department Head' WHERE Emp\_Salary IN (SELECT MAX(Emp\_Salary) FROM Employee)

C. UPDATE Employee SET Emp\_Designation = 'Department Head' WHERE (Department, Emp\_Salary) IN (SELECT Department, MAX(Emp\_Salary) FROM Employee GROUP BY Department)

D. UPDATE Employee SET Emp\_Designation = 'Department Head' WHERE Emp\_Salary IN (SELECT MAX(Emp\_Salary) FROM Employee GROUP BY Department)

Q 70. Update the designation of employees to 'Manager' if they have more than 5 years of experience, 'Senior Developer' if they have more than 10 years of experience, and 'Junior Developer' otherwise.

A. UPDATE Employee SET Emp\_Designation = CASE WHEN Emp\_Experience > 5 THEN 'Manager' WHEN Emp\_Experience > 10 THEN 'Senior Developer' ELSE 'Junior Developer' END

B. UPDATE Employee SET Emp\_Designation = CASE WHEN Emp\_Experience > 10 THEN 'Senior Developer' WHEN Emp\_Experience > 5 THEN 'Manager' ELSE 'Junior Developer' END

C. UPDATE Employee SET Emp\_Designation = CASE WHEN Emp\_Experience > 5 THEN 'Senior Developer' WHEN Emp\_Experience > 10 THEN 'Manager' ELSE 'Junior Developer' END

D. UPDATE Employee SET Emp\_Designation = CASE WHEN Emp\_Experience > 10 THEN 'Manager' WHEN Emp\_Experience > 5 THEN 'Senior Developer' ELSE 'Junior Developer' END

S.no	Correct answer
------	----------------

1	C
---	---

2	C
---	---

3	A
---	---

4	A
---	---

5	B
---	---

6	D
---	---

7	B
---	---

8	C
---	---

9	A
---	---

10	B
----	---

11	D
----	---

12	C
----	---

13	A
----	---

14	C
----	---

15	D
----	---

16	D
----	---

17	C
----	---

18	A
----	---

19	D
----	---

20	A
----	---

21	B
----	---

22	C
----	---

23	C
----	---

24	B
----	---

25	D
----	---

26	A
----	---

27	B
----	---

28	A
----	---

29	C
----	---

30	B
----	---

31	D
----	---

32	D
----	---

33	B
----	---

34	C
----	---

35	A
----	---

36	C
----	---

37	B
----	---

38	A
----	---

39	A
----	---

40	D
----	---

41	B
----	---

42	C
----	---

43	A
----	---

44	B
----	---

45	C
----	---

46	A
----	---

47 A  
48 A  
49 A  
50 B  
51 A  
52 C  
53 D  
54 D  
55 A  
56 C  
57 C  
58 B  
59 C  
60 B  
61 C  
62 C  
63 A  
64 D  
65 D  
66 A  
67 C  
68 B  
69 C  
70 B

## 1 mark each->

Here are 10 multiple-choice questions (MCQs) related to SQL queries in the context of a bank management database. Each question is followed by the correct answer (indicated in parentheses).

1. What does the SQL statement SELECT do in the context of a bank management database?
  - a. Insert new records into a table.
  - b. Retrieve data from one or more tables. (Correct)
  - c. Update existing records in a table.
  - d. Delete records from a table.
  
2. Which SQL clause is used to filter the results of a SELECT query?
  - a. SET
  - b. WHERE (Correct)
  - c. JOIN
  - d. GROUP BY
  
3. To calculate the total balance of all savings accounts, which SQL function would you use?
  - a. SUM (Correct)
  - b. AVG
  - c. COUNT
  - d. MAX
  
4. In a bank management system, if you want to list the customers with account balances greater than \$10,000, which SQL query should you use?
  - a. SELECT FROM customers WHERE balance = 10000
  - b. SELECT FROM customers WHERE balance > 10000 (Correct)
  - c. SELECT FROM customers WHERE balance < 10000
  - d. SELECT FROM customers WHERE balance = 10000 OR balance > 10000
  
5. Which SQL statement is used to add a new record to a table in a bank management database?
  - a. ADD



- b. CREATE
- c. INSERT INTO (Correct)
- d. UPDATE

6. To retrieve a list of transactions made by a specific customer named "John Doe," you would use the SQL statement:

- a. SELECT FROM transactions WHERE customer = 'John Doe'
- b. SELECT FROM transactions WHERE customer\_id = 'John Doe'
- c. SELECT FROM transactions WHERE customer\_name = 'John Doe'
- d. SELECT FROM transactions WHERE customer\_id = (SELECT customer\_id FROM customers WHERE name = 'John Doe') (Correct)

7. Which SQL clause is used to combine rows from two or more tables in a SELECT query in a bank management database?

- a. WHERE
- b. HAVING
- c. JOIN (Correct)
- d. FROM

8. If you want to retrieve a unique list of branch names from a bank's branches table, which SQL keyword should you use?

- a. DISTINCT (Correct)
- b. UNIQUE
- c. UNIQUEKEY
- d. UNIQUENAME

9. To update the account balance of a specific customer in a bank management database, which SQL statement should you use?

- a. UPDATE customer SET balance = 10000 WHERE name = 'John Doe'
- b. MODIFY customer SET balance = 10000 WHERE name = 'John Doe'
- c. ALTER customer UPDATE balance = 10000 WHERE name = 'John Doe'
- d. UPDATE customers SET balance = 10000 WHERE name = 'John Doe' (Correct)

10. Which SQL clause is used to group rows that have the same values in specified columns, such as calculating the total balance per branch in a bank management system?

- a. SORT BY
- b. GROUP BY (Correct)
- c. MERGE
- d. COMBINE

2 marks each->

Here are 5 multiple-choice questions (MCQs) related to SQL queries based on a bank management system, along with the correct answers:

Customers Table:

CustomerID	Name	AccountType
1	John Doe	Savings
2	Jane Smith	Checking
3	Mike Johnson	Savings
4	Sarah Lee	Checking
5	David White	Savings

Transactions Table:

TransactionID	CustomerID	Amount	TransactionDate
1	1	500	2023-01-15
2	2	1000	2023-01-20
3	3	800	2023-01-25
4	4	1200	2023-02-05
5	1	600	2023-02-10

Question 1: What will the following SQL query retrieve?

```
```sql  
SELECT C.Name, SUM(T.Amount) AS TotalBalance  
FROM Customers C  
JOIN Transactions T ON C.CustomerID = T.CustomerID  
GROUP BY C.Name  
HAVING TotalBalance > 1000;  
```
```

- a. The total balance of each customer.
- b. The names of customers with a total balance greater than \$1000. (Correct)
- c. All transactions with an amount greater than \$1000.
- d. The total balance of all customers.

Question 2: Which SQL statement is used to find the customer who made the largest single transaction (by amount) in January 2023?

- a. `SELECT CustomerID, MAX(Amount) FROM Transactions WHERE YEAR(TransactionDate) = 2023 AND MONTH(TransactionDate) = 1;`
- b. `SELECT Name, MAX(Amount) FROM Customers JOIN Transactions ON Customers.CustomerID = Transactions.CustomerID WHERE YEAR(TransactionDate) = 2023 AND MONTH(TransactionDate) = 1;` (Correct)
- c. `SELECT Name, MAX(Amount) FROM Transactions WHERE YEAR(TransactionDate) = 2023 AND MONTH(TransactionDate) = 1;`
- d. `SELECT CustomerID, Name, MAX(Amount) FROM Customers JOIN Transactions ON Customers.CustomerID = Transactions.CustomerID WHERE YEAR(TransactionDate) = 2023 AND MONTH(TransactionDate) = 1;`

Question 3: What is the result of this SQL query?

```
```sql  
SELECT AccountType, COUNT() AS NumberOfCustomers
```

```
FROM Customers
GROUP BY AccountType
HAVING COUNT() > 1;
...
```

- a. The total number of customers for each account type.
- b. The number of customers who have more than one account. (Correct)
- c. The number of accounts for each customer type.
- d. The number of customers for each account type.

Question 4: How can you retrieve the names of customers who have both savings and checking accounts?

- a. ``SELECT Name FROM Customers WHERE AccountType = 'Savings' AND AccountType = 'Checking';``
- b. ``SELECT Name FROM Customers WHERE AccountType = 'Savings' OR AccountType = 'Checking';``
- c. ``SELECT Name FROM Customers WHERE CustomerID IN (SELECT CustomerID FROM Customers WHERE AccountType = 'Savings') AND CustomerID IN (SELECT CustomerID FROM Customers WHERE AccountType = 'Checking');`` (Correct)
- d. ``SELECT Name FROM Customers WHERE EXISTS (SELECT 1 FROM Customers AS C1 WHERE C1.CustomerID = Customers.CustomerID AND C1.AccountType = 'Savings') AND EXISTS (SELECT 1 FROM Customers AS C2 WHERE C2.CustomerID = Customers.CustomerID AND C2.AccountType = 'Checking');``

Question 5: What does the following SQL query do?

```
```sql
SELECT Name, SUM(Amount) AS TotalBalance
FROM Customers
LEFT JOIN Transactions ON Customers.CustomerID = Transactions.CustomerID
GROUP BY Name;
...
```

- a. It lists the total balance for each customer, including those with no transactions. (Correct)

- b. It lists the total balance for each customer but excludes customers with no transactions.
- c. It lists the total balance for customers with only checking accounts.
- d. It lists the total balance for customers with only savings accounts.

Certainly! Here are 5 more multiple-choice questions (MCQs) related to SQL queries based on a bank management system, along with the correct answers:

Customers Table:

CustomerID	Name	AccountType
1	John Doe	Savings
2	Jane Smith	Checking
3	Mike Johnson	Savings
4	Sarah Lee	Checking
5	David White	Savings

Transactions Table:

TransactionID	CustomerID	Amount	TransactionDate
1	1	500	2023-01-15
2	2	1000	2023-01-20
3	3	800	2023-01-25
4	4	1200	2023-02-05
5	1	600	2023-02-10

Question 6: What does the following SQL query retrieve?

```
```sql
```

```
SELECT CustomerID, AVG(Amount) AS AvgTransaction
```

```
FROM Transactions
```

```
GROUP BY CustomerID
```

```
HAVING AvgTransaction > 800;
```

```
```
```

- a. The average transaction amount for each customer.
- b. The total balance for each customer.
- c. The names of customers with an average transaction amount greater than \$800. (Correct)
- d. The names of customers with more than \$800 in their accounts.

Question 7: Which SQL statement can be used to find the customer who has made the largest number of transactions?

- a. ``SELECT CustomerID, MAX(NumberOfTransactions) FROM (SELECT CustomerID, COUNT(TransactionID) AS NumberOfTransactions FROM Transactions GROUP BY CustomerID);``
- b. ``SELECT CustomerID, MAX(TransactionCount) FROM Customers LEFT JOIN (SELECT CustomerID, COUNT(TransactionID) AS TransactionCount FROM Transactions GROUP BY CustomerID) AS Subquery;`` (Correct)
- c. ``SELECT CustomerID, MAX(TransactionCount) FROM Customers JOIN (SELECT CustomerID, COUNT(TransactionID) AS TransactionCount FROM Transactions GROUP BY CustomerID) AS Subquery ON Customers.CustomerID = Subquery.CustomerID;``
- d. ``SELECT CustomerID, MAX(TransactionCount) FROM Customers JOIN (SELECT CustomerID, COUNT(TransactionID) AS TransactionCount FROM Transactions GROUP BY CustomerID) AS Subquery ON Customers.CustomerID = Subquery.CustomerID;``

Question 8: What does the following SQL query do?

```
```sql
```

```
SELECT AccountType, COUNT() AS NumberOfCustomers
```

```
FROM Customers
```

```
GROUP BY AccountType
```

HAVING COUNT() > 1;

...

- a. The total number of customers for each account type.
- b. The number of customers who have more than one account. (Correct)
- c. The number of accounts for each customer type.
- d. The number of customers for each account type.

Question 9: How can you retrieve the names of customers who have the highest total balance?

- a. `'SELECT Name FROM Customers WHERE TotalBalance = MAX(TotalBalance);'`
- b. `'SELECT Name FROM Customers WHERE TotalBalance = (SELECT MAX(TotalBalance) FROM Customers);'` (Correct)
- c. `'SELECT Name FROM Customers GROUP BY Name HAVING TotalBalance = MAX(TotalBalance);'`
- d. `'SELECT Name FROM Customers WHERE TotalBalance = (SELECT MAX(TotalBalance) FROM Customers) GROUP BY Name;'`

Question 10: What is the result of the following SQL query?

```
```sql
```

```
SELECT T.CustomerID, C.Name, COUNT(T.TransactionID) AS NumberOfTransactions
```

```
FROM Customers C
```

```
LEFT JOIN Transactions T ON C.CustomerID = T.CustomerID
```

```
GROUP BY T.CustomerID, C.Name
```

```
HAVING NumberOfTransactions = 0;
```

```
```
```

- a. The names of customers with no transactions.
- b. The total number of transactions for each customer.
- c. The names of customers with more than one transaction.
- d. The names of customers with no transactions. (Correct)

These questions and answers continue to provide a challenge with more advanced SQL queries in the context of a bank management database.

11. To retrieve a list of customers who have made the highest transaction amount within each branch, you should use which SQL statement?

- a. `SELECT FROM transactions WHERE amount = MAX(amount) GROUP BY branch_id`
- b. `SELECT MAX(amount), customer_id FROM transactions GROUP BY branch_id`
- c. `SELECT customer_id, MAX(amount) FROM transactions GROUP BY branch_id HAVING MAX(amount)`
- d. `SELECT branch_id, MAX(amount) FROM transactions GROUP BY branch_id` (Correct)

12. In the context of a bank management database, which SQL command is used to delete all transactions older than one year?

- a. REMOVE
- b. `DELETE FROM transactions WHERE transaction_date < DATEADD(YEAR, -1, GETDATE())`
- c. `DELETE FROM transactions WHERE DATEDIFF(YEAR, transaction_date, GETDATE()) > 1` (Correct)
- d. DROP

13. You want to find the top 5 customers with the highest account balances. Which SQL statement should you use?

- a. `SELECT FROM customers ORDER BY balance DESC LIMIT 5`
- b. `SELECT FROM customers ORDER BY balance DESC FETCH FIRST 5 ROWS ONLY`
- c. `SELECT FROM customers ORDER BY balance DESC OFFSET 0 ROWS FETCH NEXT 5 ROWS ONLY`
- d. `SELECT FROM customers ORDER BY balance DESC LIMIT 5` (Correct)

14. In a bank management system, how would you list all customers who have both savings and checking accounts?



- a. `SELECT FROM customers WHERE account_type = 'savings' AND account_type = 'checking'`
- b. `SELECT FROM customers WHERE account_type = 'savings' OR account_type = 'checking'`
- c. `SELECT FROM customers WHERE customer_id IN (SELECT customer_id FROM accounts WHERE account_type = 'savings') AND customer_id IN (SELECT customer_id FROM accounts WHERE account_type = 'checking')` (Correct)
- d. `SELECT FROM customers JOIN accounts ON customers.customer_id = accounts.customer_id WHERE account_type = 'savings' AND account_type = 'checking'`

15. To find the total number of transactions made by each branch, which SQL statement should you use?

- a. `SELECT branch_id, COUNT(transaction_id) FROM transactions GROUP BY branch_id`
- b. `SELECT COUNT(transaction_id) FROM transactions WHERE branch_id = DISTINCT branch_id`
- c. `SELECT COUNT(transaction_id) AS total_transactions, branch_id FROM transactions GROUP BY branch_id`
- d. `SELECT branch_id, SUM(transaction_id) FROM transactions GROUP BY branch_id` (Correct)

16. You want to retrieve the last transaction for each customer. What SQL statement would you use?

- a. `SELECT FROM transactions WHERE transaction_id = MAX(transaction_id) GROUP BY customer_id`
- b. `SELECT FROM transactions WHERE transaction_id = (SELECT MAX(transaction_id) FROM transactions GROUP BY customer_id)`
- c. `SELECT FROM transactions WHERE transaction_id = (SELECT MAX(transaction_id) FROM transactions) GROUP BY customer_id`
- d. `SELECT FROM transactions WHERE transaction_id = (SELECT MAX(transaction_id) FROM transactions WHERE customer_id = transactions.customer_id)` (Correct)

17. To calculate the average balance of customers who have made at least three transactions, which SQL query should you use?

- a. `SELECT AVG(balance) FROM customers HAVING COUNT(SELECT transaction_id FROM transactions WHERE transactions.customer_id = customers.customer_id) >= 3`
- b. `SELECT AVG(balance) FROM customers WHERE (SELECT COUNT(transaction_id) FROM transactions WHERE transactions.customer_id = customers.customer_id) >= 3`
- c. `SELECT AVG(balance) FROM customers WHERE customer_id IN (SELECT customer_id FROM transactions GROUP BY customer_id HAVING COUNT(transaction_id) >= 3)` (Correct)
- d. `SELECT AVG(balance) FROM customers WHERE (SELECT COUNT() FROM transactions WHERE transactions.customer_id = customers.customer_id) >= 3`

18. In a bank management system, you want to retrieve a list of customers who have both a checking account and a savings account at the same branch. What SQL query would you use?

a. `SELECT FROM customers WHERE account_type = 'checking' AND account_type = 'savings'`  
`GROUP BY branch_id`

b. `SELECT FROM customers WHERE customer_id IN (SELECT customer_id FROM accounts WHERE account_type = 'checking' AND branch_id IN (SELECT branch_id FROM accounts WHERE account_type = 'savings'))`

c. `SELECT FROM customers WHERE EXISTS (SELECT 1 FROM accounts AS a1 WHERE a1.customer_id = customers.customer_id AND a1.account_type = 'checking') AND EXISTS (SELECT 1 FROM accounts AS a2 WHERE a2.customer_id = customers.customer_id AND a2.account_type = 'savings')` (Correct)

d. `SELECT FROM customers JOIN accounts ON customers.customer_id = accounts.customer_id WHERE account_type = 'checking' AND account_type = 'savings'`

19. To find the total interest earned by the bank in a given year on all fixed deposits, what SQL statement should you use?

a. `SELECT SUM(interest_earned) FROM transactions WHERE transaction_type = 'fixed deposit' AND YEAR(transaction_date) = [year]`

b. `SELECT SUM(interest_earned) FROM transactions WHERE transaction_type = 'fixed deposit' AND EXTRACT(YEAR FROM transaction_date) = [year]` (Correct)

c. `SELECT SUM(interest_earned) FROM transactions WHERE transaction_type = 'fixed deposit' AND DATEPART(YEAR, transaction_date) = [year]`

d. `SELECT SUM(interest_earned) FROM transactions WHERE transaction_type = 'fixed deposit' AND YEAR = [year]`

20. To list all customers who have made transactions on weekdays (Monday to Friday), what SQL statement should you use?

a. `SELECT FROM customers WHERE WEEKDAY(transaction_date) BETWEEN 0 AND 4`

b. `SELECT FROM customers WHERE DATEPART(WEEKDAY, transaction_date) BETWEEN 1 AND 5` (Correct)

c. `SELECT FROM customers WHERE DAYOFWEEK(transaction_date) BETWEEN 1 AND 5`

d. `SELECT FROM customers WHERE EXTRACT(DAY FROM transaction_date) BETWEEN 1 AND 5`

| S<br>R | Questions                                                                                              | Option 1                                            | Option 2                                                                                | Option 3                                                                           | Option 4                                           | Ans |
|--------|--------------------------------------------------------------------------------------------------------|-----------------------------------------------------|-----------------------------------------------------------------------------------------|------------------------------------------------------------------------------------|----------------------------------------------------|-----|
| 1      | Which of the following is NOT a benefit of using transactions?                                         | Data integrity                                      | High availability                                                                       | Data consistency                                                                   | Data durability                                    | B   |
| 2      | A transaction that violates the consistency property is considered to be:                              | Serializable                                        | Inconsistent                                                                            | Isolate                                                                            | Error                                              | B   |
| 3      | Can you change the parameter values of a cursor after it has been declared and opened?                 | Yes, parameter values can be modified at any time.  | No, parameter values are fixed once the cursor is declared and opened.                  | Parameter values can only be changed during cursor declaration.                    | Cursors cannot have parameter values.              | B   |
| 4      | Can you declare a cursor without specifying the SELECT statement immediately?                          | No, a SELECT statement must always be specified.    | Yes, a SELECT statement can be added later in the code.                                 | Cursors cannot be declared in PL/SQL.                                              | Cursors are automatically generated in PL/SQL.     | A   |
| 5      | Can you declare multiple cursors with the same name but different parameters in the same PL/SQL block? | Yes, as long as the cursor names are unique.        | No, cursor names must be unique regardless of the parameters.                           | Multiple cursors are not allowed in the same PL/SQL block.                         | Cursors with parameters cannot have the same name. | B   |
| 6      | Can you declare multiple cursors within the same PL/SQL block? If so, how do you differentiate them?   | No, only one cursor is allowed per block.           | Yes, multiple cursors can be declared, and they are differentiated by their data types. | Yes, multiple cursors can be declared, and they are differentiated by their names. | Multiple cursors cannot be used in PL/SQL.         | C   |
| 7      | Can you fetch data from a cursor into individual variables or into a record type? Explain.             | Data can only be fetched into individual variables. | Data can only be fetched into a record type.                                            | Data can be fetched into both individual variables and a record type.              | Data cannot be fetched from a cursor.              | C   |

|   |                                                                                            |                                              |                                                                                                            |                                                             |                                                         |   |
|---|--------------------------------------------------------------------------------------------|----------------------------------------------|------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------|---------------------------------------------------------|---|
| 8 | Can you nest a Cursor FOR Loop inside another Cursor FOR Loop? If so, why might you do so? | No, nesting Cursor FOR Loops is not allowed. | Yes, you can nest Cursor FOR Loops to perform complex data processing and handle related data hierarchies. | Cursor FOR Loops can only be used individually, not nested. | Nesting Cursor FOR Loops results in performance issues. | B |
|---|--------------------------------------------------------------------------------------------|----------------------------------------------|------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------|---------------------------------------------------------|---|

|    |                                                                                         |                                                                                                     |                                                                                                                                                                            |                                                                                                       |                                                                                                     |   |
|----|-----------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------|---|
| 9  | Can you use a Cursor FOR Loop to update or delete records in a database table? Explain. | No, Cursor FOR Loops are read- only.                                                                | Yes, Cursor FOR Loops can update or delete records using the UPDATE and DELETE statements.                                                                                 | Cursor FOR Loops can only insert records, not update or delete them.                                  | Cursor FOR Loops can only be used for reporting purposes.                                           | B |
| 10 | Describe the differences between an implicit cursor and an explicit cursor in PL/SQL.   | Implicit cursors are used for data modeling, while explicit cursors are used for data manipulation. | Implicit cursors are automatically created for DML statements, while explicit cursors are user-defined.                                                                    | Implicit cursors are used for database connections, while explicit cursors are used for loop control. | Implicit cursors are used for hardware design, while explicit cursors are used for web development. | B |
| 11 | Describe the purpose of PL/SQL collections, and provide examples of their types.        | PL/SQL collections are used for defining variables.                                                 | PL/SQL collections are used for database connections.                                                                                                                      | PL/SQL collections are used for storing multiple values of the same data type.                        | PL/SQL collections are used for creating triggers.                                                  | C |
| 12 | Explain how cursor parameters can be used to create dynamic cursors.                    | Cursor parameters have no role in creating dynamic cursors.                                         | By allowing parameterization of the WHERE clause in the cursor's SELECT statement, you can create dynamic cursors that retrieve specific data based on different criteria. | Cursor parameters can only be used with static cursors.                                               | Cursor parameters can be used to create triggers.                                                   | B |

|        |                                                                                     |                                                                            |                                                                            |                                                                                            |                                             |   |
|--------|-------------------------------------------------------------------------------------|----------------------------------------------------------------------------|----------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|---------------------------------------------|---|
| 1<br>3 | Explain the concept of triggers in a database context. How are they used in PL/SQL? | Triggers are used for creating web applications.                           | Triggers are used for hardware design.                                     | Triggers are used for automatically executing PL/SQL code in response to database events.  | Triggers are used for data modeling.        | C |
| 1<br>4 | Explain the difference between declaring a cursor and opening a cursor.             | Declaring a cursor retrieves data; opening a cursor defines its structure. | Declaring a cursor defines its structure; opening a cursor retrieves data. | Declaring a cursor and opening a cursor are the same.                                      | Declaring a cursor is not a PL/SQL concept. | B |
| 1<br>5 | Explain the importance of transactions in PL/SQL and how they are managed.          | Transactions are used for web development.                                 | Transactions are used for data modeling.                                   | Transactions ensure data consistency and are managed using COMMIT and ROLLBACK statements. | Transactions are not supported in PL/SQL.   | C |

|        |                                                                               |                                                                  |                                                                                         |                                                      |                                                        |   |
|--------|-------------------------------------------------------------------------------|------------------------------------------------------------------|-----------------------------------------------------------------------------------------|------------------------------------------------------|--------------------------------------------------------|---|
| 1<br>6 | Explain the purpose of a PL/SQL package and its components.                   | PL/SQL packages are used for web development.                    | PL/SQL packages are used for encapsulating procedures and functions.                    | PL/SQL packages are used for data modeling.          | PL/SQL packages are used for hardware design.          | B |
| 1<br>7 | How can you pass parameters to a PL/SQL procedure or function?                | Parameters are passed using the CALL statement.                  | Parameters are not supported in PL/SQL.                                                 | Parameters are passed as input and output variables. | Parameters are passed using the DECLARE statement.     | C |
| 1<br>8 | How can you resolve a deadlock in a database system?                          | By terminating one of the transactions involved in the deadlock. | By rolling back all transactions involved in the deadlock.                              | By increasing the isolation level.                   | Deadlocks cannot be resolved.                          | A |
| 1<br>9 | How do you create and manipulate PL/SQL associative arrays (index-by tables)? | Associative arrays are created using the ARRAY keyword.          | Associative arrays are created using the INDEX keyword.                                 | Associative arrays are not supported in PL/SQL.      | Associative arrays are created using the TYPE keyword. | D |
| 2<br>0 | How do you declare a cursor, and what are the required components?            | Cursors are automatically declared in PL/SQL.                    | Cursors are declared using the DECLARE CURSOR statement and require a SELECT statement. | Cursors are declared using the DECLARE keyword.      | Cursors are declared using the OPEN statement.         | B |

|        |                                                                                               |                                                                                           |                                                                                        |                                                                                                                                      |                                                                   |   |
|--------|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|---|
| 2<br>1 | How do you declare a variable in PL/SQL, and what are the data types supported for variables? | Variables are declared using the DECLARE keyword, and PL/SQL supports only one data type. | Variables are declared using the VAR keyword, and PL/SQL supports multiple data types. | Variables are declared using the VARIABLE keyword, and PL/SQL supports multiple data types.                                          | Variables are not supported in PL/SQL.                            | B |
| 2<br>2 | How do you define and use PL/SQL records and record types?                                    | Records are used for creating tables in PL/SQL.                                           | Records are defined using the DECLARE RECORD statement.                                | Records are used to hold data in a structured format.                                                                                | Records are not supported in PL/SQL.                              | C |
| 2<br>3 | How do you ensure that you've fetched all available data from a cursor?                       | By using the CLOSE statement.                                                             | By using the OPEN statement.                                                           | By checking the cursor attribute %NOTFOUND.                                                                                          | Cursors automatically fetch all available data.                   | C |
| 2<br>4 | How do you handle database connections and transactions in PL/SQL?                            | Database connections and transactions are automatically managed by the PL/SQL engine.     | Database connections and transactions are not supported in PL/SQL.                     | Database connections are established using the CONNECT statement, and transactions are managed using COMMIT and ROLLBACK statements. | Database connections are established using the DECLARE statement. | C |

Given this Restaurant schema Answer the following Questions

Tables:

a. Customers:

- customer\_id (Primary Key)
- first\_name
- last\_name
- email
- phone\_number
- address
- ...

b. Employees:

- employee\_id (Primary Key)
- first\_name
- last\_name
- email
- phone\_number
- position
- hire\_date
- ...

c. Menu Items:

- item\_id (Primary Key)
- name
- description
- price
- category
- ...

d. Orders:

- order\_id (Primary Key)
- customer\_id (Foreign Key to Customers)
- order\_date
- total\_amount
- ...

e. Order Items:

- order\_item\_id (Primary Key)
- order\_id (Foreign Key to Orders)
- item\_id (Foreign Key to Menu Items)
- quantity
- subtotal
- ...

f. Reservations:

- reservation\_id (Primary Key)
- customer\_id (Foreign Key to Customers)
- reservation\_date
- table\_number
- party\_size

- ...

#### g. Tables:

- table\_number (Primary Key)
- seating\_capacity
- description
- status (e.g., available, occupied, reserved)
- ...

#### h. Reviews:

- review\_id (Primary Key)
- customer\_id (Foreign Key to Customers)
- rating
- comments
- review\_date
- ...

#### i. Payments:

- payment\_id (Primary Key)
- order\_id (Foreign Key to Orders)
- payment\_date
- payment\_amount
- payment\_method
- ...

**\*\*Question 1:\*\*** In the restaurant database schema, which table would likely store information about the dishes and drinks available for customers to order?

- A. Customers
- B. Employees
- C. Menu Items
- D. Orders

**\*\*Question 2:\*\*** What is the primary key for the "Employees" table in the restaurant database schema?

- A. employee\_id
- B. order\_id
- C. customer\_id
- D. table\_number

**\*\*Question 3:\*\*** If you want to find the total number of reservations made by a specific customer with the first name "John," which SQL statement would you use?

- A. SELECT COUNT(\*) FROM Reservations WHERE first\_name = 'John';
- B. SELECT COUNT(\*) FROM Reservations WHERE customer\_id = 'John';
- C. SELECT COUNT(\*) FROM Reservations WHERE reservation\_date = 'John';
- D. SELECT COUNT(\*) FROM Reservations WHERE party\_size = 'John';

**\*\*Question 4:\*\*** What is the foreign key in the "Order Items" table in the restaurant database schema?



- A. order\_id
- B. order\_item\_id
- C. customer\_id
- D. item\_id

**\*\*Question 5:\*\*** Which SQL statement would you use to find the average rating of all reviews in the "Reviews" table?

- A. SELECT AVG(rating) FROM Reviews;
- B. SELECT AVG(rating) FROM Reviews WHERE rating IS NOT NULL;
- C. SELECT AVERAGE(rating) FROM Reviews;
- D. SELECT AVG(rating) FROM Reviews GROUP BY rating;

**\*\*Question 6:\*\*** What data type is typically used for the "price" field in the "Menu Items" table to represent monetary values?

- A. INT
- B. FLOAT
- C. VARCHAR
- D. DECIMAL

**\*\*Question 7:\*\*** Which SQL clause is used to filter rows in a query result, based on a specified condition?

- A. GROUP BY
- B. HAVING
- C. WHERE
- D. ORDER BY

**\*\*Question 8:\*\*** If you want to retrieve all the menu items with a price less than \$10, which SQL statement would you use?

- A. SELECT \* FROM "Menu Items" WHERE price < 10;
- B. SELECT \* FROM "Menu Items" WHERE price > 10;
- C. SELECT \* FROM "Menu Items" HAVING price < 10;
- D. SELECT \* FROM "Menu Items" ORDER BY price ASC;

**\*\*Question 9:\*\*** What SQL clause is used to sort the result set in ascending or descending order?

- A. WHERE
- B. SORT
- C. ORDER BY
- D. GROUP BY

**\*\*Question 10:\*\*** In the "Tables" table of the restaurant schema, what field indicates whether a table is currently available for seating?

- A. seating\_capacity
- B. description
- C. status
- D. table\_number

**\*\*Question 11:\*\*** Which SQL statement would you use to find the highest total amount spent on an order?

- A. SELECT MAX(total\_amount) FROM Orders;
- B. SELECT MIN(total\_amount) FROM Orders;
- C. SELECT SUM(total\_amount) FROM Orders;
- D. SELECT AVG(total\_amount) FROM Orders;

**\*\*Question 12:\*\*** In the "Payments" table, which field would typically store the payment method used for a particular order?

- A. payment\_id
- B. order\_id
- C. payment\_date
- D. payment\_method

**\*\*Question 13:\*\*** What is the purpose of the SQL GROUP BY clause?

- A. To filter rows based on a condition.
- B. To join two or more tables.
- C. To aggregate data and perform calculations on groups of rows.
- D. To sort rows in a result set.

**\*\*Question 14:\*\*** Which SQL statement is used to add a new customer to the "Customers" table?

- A. INSERT INTO Customers (first\_name, last\_name) VALUES ('John', 'Doe');
- B. UPDATE Customers SET first\_name = 'John', last\_name = 'Doe' WHERE customer\_id = 1;
- C. DELETE FROM Customers WHERE first\_name = 'John' AND last\_name = 'Doe';
- D. SELECT \* FROM Customers WHERE first\_name = 'John' AND last\_name = 'Doe';

**\*\*Question 15:\*\*** In the "Reservations" table, what does the "party\_size" field represent?

- A. The reservation ID
- B. The number of people in the reservation party
- C. The table number
- D. The reservation date and time

**\*\*Question 16:\*\*** What SQL statement is used to delete a specific order with order\_id = 12345 from the "Orders" table?

- A. DELETE FROM Orders WHERE order\_id = 12345;
- B. UPDATE Orders SET order\_status = 'Canceled' WHERE order\_id = 12345;
- C. SELECT \* FROM Orders WHERE order\_id = 12345;
- D. INSERT INTO Orders (order\_id) VALUES (12345);

**\*\*Question 17:\*\*** In the "Customers" table, what is the primary key for uniquely identifying each customer?

- A. order\_id
- B. customer\_id
- C. employee\_id
- D. menu\_item\_id

**\*\*Question 18:\*\*** Which SQL clause is used to combine rows from two or more tables based on a related column between them?

- A. WHERE
- B. JOIN
- C. GROUP BY
- D. HAVING

**\*\*Question 19:\*\*** If you want to find the names and email addresses of customers who have made a reservation, which SQL statement would you use?

- A. SELECT first\_name, email FROM Customers;
- B. SELECT first\_name, email FROM Reservations;
- C. SELECT first\_name, email FROM Customers WHERE customer\_id IN (SELECT customer\_id FROM Reservations);
- D. SELECT first\_name, email FROM Reservations WHERE customer\_id IN (SELECT customer\_id FROM Customers);

**\*\*Question 20:\*\*** In the "Menu Items" table, which SQL constraint should ensure that the "name" field contains unique values for each menu item?

- A. PRIMARY KEY
- B. FOREIGN KEY
- C. UNIQUE
- D. CHECK

**\*\*Answers:\*\***

- 1. C. Menu Items
- 2. A. employee\_id
- 3. A. SELECT COUNT(\*) FROM Reservations WHERE first\_name = 'John';
- 4. A. order\_id
- 5. A. SELECT AVG(rating) FROM Reviews;
- 6. D. DECIMAL
- 7. C. WHERE
- 8. A. SELECT \* FROM "Menu Items" WHERE price < 10;
- 9. C. ORDER BY
- 10. C. status
- 11. A. SELECT MAX(total\_amount) FROM Orders;
- 12. D. payment\_method
- 13. C. To aggregate data and perform calculations on groups of rows.
- 14. A. INSERT INTO Customers (first\_name, last\_name) VALUES ('John', 'Doe');
- 15. B. The number of people in the reservation party
- 16. A. DELETE FROM Orders WHERE order\_id = 12345;
- 17. B. customer\_id
- 18. B. JOIN
- 19. C. SELECT first\_name, email FROM Customers WHERE customer\_id IN (SELECT customer\_id FROM Reservations);
- 20. C. UNIQUE

**\*\*Question 1:\*\*** You want to retrieve the names of customers who have placed orders but have not made any reservations. Which SQL query should you use?

- A. `SELECT c.first_name, c.last_name FROM Customers c WHERE c.customer_id IN (SELECT o.customer_id FROM Orders o) AND c.customer_id NOT IN (SELECT r.customer_id FROM Reservations r);`
- B. `SELECT c.first_name, c.last_name FROM Customers c JOIN Orders o ON c.customer_id = o.customer_id WHERE c.customer_id NOT IN (SELECT r.customer_id FROM Reservations r);`
- C. `SELECT c.first_name, c.last_name FROM Customers c WHERE c.customer_id NOT IN (SELECT r.customer_id FROM Reservations r) GROUP BY c.customer_id HAVING COUNT(o.customer_id) > 0;`
- D. `SELECT c.first_name, c.last_name FROM Customers c LEFT JOIN Reservations r ON c.customer_id = r.customer_id WHERE r.customer_id IS NULL;`

**\*\*Question 2:\*\*** In the restaurant database schema, which SQL statement would you use to find the customer who has spent the most on orders?

- A. `SELECT MAX(total_amount) FROM Orders;`
- B. `SELECT c.first_name, c.last_name  
FROM Customers c  
JOIN Orders o ON c.customer_id = o.customer_id  
GROUP BY c.customer_id  
HAVING MAX(o.total_amount);`
- C. `SELECT c.first_name, c.last_name  
FROM Customers c  
WHERE c.customer_id = (SELECT customer_id FROM Orders WHERE total_amount = (SELECT  
MAX(total_amount) FROM Orders));`
- D. `SELECT c.first_name, c.last_name  
FROM Customers c  
WHERE c.customer_id = (SELECT customer_id FROM Orders GROUP BY customer_id HAVING  
MAX(total_amount));`

**\*\*Question 3:\*\*** What is the result of the following SQL query?

```
```sql
SELECT COUNT(*)
FROM Orders o
JOIN Customers c ON o.customer_id = c.customer_id
WHERE o.order_date > (SELECT MAX(reservation_date) FROM Reservations WHERE customer_id =
o.customer_id);
```
```

- A. It counts the number of orders placed after the last reservation date for each customer.
- B. It counts the number of orders placed by each customer.

C. It counts the number of customers who have placed orders after making reservations.

D. It returns an error because the subquery is not properly correlated.

**\*\*Question 4:\*\*** To find the average number of orders placed by customers, which SQL statement should you use?

A. `SELECT AVG(COUNT(order_id)) FROM Orders;`

B. `SELECT AVG(order_count) FROM (SELECT customer_id, COUNT(order_id) as order_count FROM Orders GROUP BY customer_id) AS subquery;`

C. `SELECT AVG(COUNT(order_id)) FROM Customers c JOIN Orders o ON c.customer_id = o.customer_id GROUP BY c.customer_id;`

D. `SELECT AVG(order_count) FROM (SELECT customer_id, COUNT(order_id) as order_count FROM Orders GROUP BY customer_id) subquery;`

**\*\*Question 5:\*\*** You want to find the names of employees who have processed at least one payment and also placed an order. Which SQL statement would you use?

A. `SELECT e.first_name, e.last_name  
FROM Employees e  
JOIN Payments p ON e.employee_id = p.employee_id  
WHERE e.employee_id IN (SELECT employee_id FROM Orders);`

B. `SELECT e.first_name, e.last_name  
FROM Employees e  
WHERE e.employee_id IN (SELECT employee_id FROM Orders)  
AND e.employee_id IN (SELECT employee_id FROM Payments);`

C. `SELECT e.first_name, e.last_name  
FROM Employees e  
JOIN Payments p ON e.employee_id = p.employee_id  
JOIN Orders o ON e.employee_id = o.employee_id;`

D. `SELECT e.first_name, e.last_name  
FROM Employees e  
WHERE e.employee_id IN (SELECT employee_id FROM Orders)  
UNION  
SELECT e.first_name, e.last_name  
FROM Employees e  
WHERE e.employee_id IN (SELECT employee_id FROM Payments);`

**\*\*Question 6:\*\*** Which SQL statement would you use to find the names of customers who have not placed any orders and have not made any reservations?

A. `SELECT c.first_name, c.last_name  
FROM Customers c  
WHERE c.customer_id NOT IN (SELECT customer_id FROM Orders)  
AND c.customer_id NOT IN (SELECT customer_id FROM Reservations);`

B. `SELECT c.first_name, c.last_name  
FROM Customers c  
LEFT JOIN Orders o ON c.customer_id = o.customer_id  
LEFT JOIN Reservations r ON c.customer_id = r.customer_id  
WHERE o.customer_id IS NULL AND r.customer_id IS NULL;`

C. `SELECT c.first_name, c.last_name  
FROM Customers c  
JOIN Orders o ON c.customer_id = o.customer_id  
JOIN Reservations r ON c.customer_id = r.customer_id  
WHERE o.customer_id IS NULL AND r.customer_id IS NULL;`

D. It is not possible to find such customers with a single SQL query.

**\*\*Question 7:\*\*** In the "Payments" table, which SQL constraint would ensure that the "payment\_date" is not null?

A. PRIMARY KEY

B. FOREIGN KEY

C. NOT NULL

D. CHECK

**\*\*Question 8:\*\*** You want to find the menu items with the highest price in each category. Which SQL statement would you use?

A. `SELECT MAX(price), category FROM "Menu Items" GROUP BY category;`

B. `SELECT category, MAX(price) FROM "Menu Items" GROUP BY category;`

C. `SELECT category, price FROM "Menu Items" WHERE price = MAX(price) GROUP BY category;`

D. `SELECT category, price FROM "Menu Items" WHERE price IN (SELECT MAX(price) FROM "Menu Items" GROUP BY category);`

**\*\*Question 9:\*\*** What does the following SQL query do?

```
```sql
SELECT c.first_name, c.last_name
FROM Customers c
WHERE c.customer_id NOT IN (
    SELECT o.customer_id
    FROM Orders o
    WHERE o.order_date >= '2023-01-01'
);
```
```

A. It selects the names of customers who have placed orders on or after January 1, 2023.

B. It selects the names of customers who have never placed an order.

- C. It selects the names of customers who have placed orders before January 1, 2023.
- D. It selects the names of customers who have placed orders before or on January 1, 2023.

**\*\*Question 10:\*\*** In the "Reviews" table, you want to find the average rating given by customers for each menu item. Which SQL statement should you use?

- A. `SELECT AVG(rating), item_id FROM Reviews GROUP BY item_id;`
- B. `SELECT AVG(rating), menu_item_id FROM Reviews GROUP BY menu_item_id;`
- C. `SELECT AVG(rating), menu_item_id FROM Menu Items GROUP BY menu_item_id;`
- D. `SELECT AVG(rating), item_id FROM Menu Items GROUP BY item_id;`

**\*\*Question 11:\*\*** To find the total amount spent by each customer on their orders, which SQL statement should you use?

- A. `SELECT customer_id, SUM(total_amount) FROM Orders;`
- B. `SELECT c.first_name, c.last_name, SUM(o.total_amount)  
FROM Customers c  
JOIN Orders o ON  
c.customer_id = o.customer_id  
GROUP BY c.customer_id;`
- C. `SELECT SUM(total_amount) FROM Orders GROUP BY customer_id;`
- D. `SELECT c.first_name, c.last_name, total_amount FROM Customers c JOIN Orders o ON c.customer_id = o.customer_id;`

**\*\*Question 12:\*\*** What is the result of the following SQL query?

```
```sql
SELECT c.first_name, c.last_name
FROM Customers c
WHERE c.customer_id IN (
    SELECT o.customer_id
    FROM Orders o
    WHERE o.total_amount > 50
);
```
```

- A. It selects the names of customers who have placed orders with a total amount greater than 50.
- B. It selects the names of customers who have never placed an order.
- C. It selects the names of customers who have placed orders with a total amount less than or equal to 50.

D. It returns an error because the subquery is not properly correlated.

**\*\*Question 13:\*\*** To find the names of customers who have placed at least two orders, which SQL statement should you use?

- A. 

```
SELECT c.first_name, c.last_name
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id
HAVING COUNT(o.order_id) >= 2;
```
- B. 

```
SELECT c.first_name, c.last_name
FROM Customers c
WHERE c.customer_id IN (SELECT customer_id FROM Orders GROUP BY customer_id HAVING
COUNT(order_id) >= 2);
```
- C. 

```
SELECT c.first_name, c.last_name
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
WHERE COUNT(o.order_id) >= 2;
```
- D. 

```
SELECT c.first_name, c.last_name
FROM Customers c
WHERE c.customer_id IN (SELECT customer_id FROM Orders WHERE COUNT(order_id) >= 2);
```

**\*\*Question 14:\*\*** What does the following SQL query do?

```
``sql
SELECT m.name
FROM "Menu Items" m
WHERE m.price = (
    SELECT MAX(price)
    FROM "Menu Items"
    WHERE category = 'Appetizers'
);
...

```

- A. It selects the names of the most expensive menu items in the "Appetizers" category.
- B. It selects the names of all menu items in the "Appetizers" category.
- C. It selects the names of the most expensive menu items in all categories.
- D. It returns an error because the subquery is not properly correlated.

**\*\*Question 15:\*\*** You want to find the names of customers who have placed orders, made reservations, and left a review. Which SQL statement would you use?

- A. 

```
SELECT c.first_name, c.last_name
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
```



JOIN Reservations r ON c.customer\_id = r.customer\_id  
JOIN Reviews rv ON c.customer\_id = rv.customer\_id;

B. SELECT c.first\_name, c.last\_name  
FROM Customers c  
WHERE c.customer\_id IN (SELECT customer\_id FROM Orders)  
AND c.customer\_id IN (SELECT customer\_id FROM Reservations)  
AND c.customer\_id IN (SELECT customer\_id FROM Reviews);

C. SELECT c.first\_name, c.last\_name  
FROM Customers c  
JOIN Orders o ON c.customer\_id = o.customer\_id  
JOIN Reservations r ON c.customer\_id = r.customer\_id  
WHERE c.customer\_id IN (SELECT customer\_id FROM Reviews);

D. It is not possible to find such customers with a single SQL query.

**\*\*Question 16:\*\*** You want to find the total number of orders placed on tables with a seating capacity of 4 or more. Which SQL query should you use?

A. SELECT COUNT(\*) FROM Orders o JOIN "Tables" t ON o.table\_number = t.table\_number WHERE t.seating\_capacity >= 4;

B. SELECT COUNT(\*) FROM Orders o WHERE o.table\_number IN (SELECT table\_number FROM "Tables" WHERE seating\_capacity >= 4);

C. SELECT COUNT(\*) FROM Orders o JOIN "Tables" t ON o.table\_number = t.table\_number WHERE t.seating\_capacity <= 4;

D. SELECT COUNT(\*) FROM Orders o WHERE o.table\_number IN (SELECT table\_number FROM "Tables" WHERE seating\_capacity <= 4);

**\*\*Question 17:\*\*** To find the menu items that have never been reviewed, which SQL statement should you use?

A. SELECT name FROM "Menu Items" WHERE item\_id NOT IN (SELECT item\_id FROM Reviews);

B. SELECT name FROM "Menu Items" WHERE item\_id IN (SELECT item\_id FROM Reviews) HAVING COUNT(\*) = 0;

C. SELECT name FROM "Menu Items" WHERE NOT EXISTS (SELECT \* FROM Reviews WHERE Reviews.item\_id = "Menu Items".item\_id);

D. SELECT name FROM "Menu Items" LEFT JOIN Reviews ON "Menu Items".item\_id = Reviews.item\_id WHERE Reviews.item\_id IS NULL;

**\*\*Question 18:\*\*** You want to find the names of customers who have placed orders on or after the date of their last reservation. Which SQL query should you use?

A. SELECT c.first\_name, c.last\_name  
FROM Customers c  
JOIN Orders o ON c.customer\_id = o.customer\_id

```
JOIN Reservations r ON c.customer_id = r.customer_id
WHERE o.order_date >= r.reservation_date;
```

B. SELECT c.first\_name, c.last\_name  
FROM Customers c  
WHERE c.customer\_id IN (  
SELECT customer\_id  
FROM Orders  
WHERE order\_date >= (SELECT MAX(reservation\_date) FROM Reservations WHERE customer\_id =  
c.customer\_id)  
);

C. SELECT c.first\_name, c.last\_name  
FROM Customers c  
JOIN Orders o ON c.customer\_id = o.customer\_id  
JOIN Reservations r ON c.customer\_id = r.customer\_id  
WHERE o.order\_date >= (SELECT MAX(reservation\_date) FROM Reservations WHERE customer\_id =  
c.customer\_id);

D. It is not possible to achieve this with a single SQL query.

**\*\*Question 19:\*\*** You want to find the employees who have processed payments for orders with a total amount greater than \$1000. Which SQL query should you use?

A. SELECT e.first\_name, e.last\_name  
FROM Employees e  
JOIN Payments p ON e.employee\_id = p.employee\_id  
WHERE p.order\_id IN (SELECT order\_id FROM Orders WHERE total\_amount > 1000);

B. SELECT e.first\_name, e.last\_name  
FROM Employees e  
JOIN Payments p ON e.employee\_id = p.employee\_id  
WHERE p.order\_id IN (SELECT order\_id FROM Orders WHERE total\_amount > 1000);

C. SELECT e.first\_name, e.last\_name  
FROM Employees e  
JOIN Orders o ON e.employee\_id = o.employee\_id  
JOIN Payments p ON o.order\_id = p.order\_id  
WHERE o.total\_amount > 1000;

D. SELECT e.first\_name, e.last\_name  
FROM Employees e  
WHERE e.employee\_id IN (  
SELECT p.employee\_id  
FROM Payments p  
WHERE p.order\_id IN (SELECT order\_id FROM Orders WHERE total\_amount > 1000)  
);

**\*\*Question 20:\*\*** To find the number of customers who have both placed orders and made reservations, which SQL query should you use?

A. SELECT COUNT(DISTINCT c.customer\_id)  
FROM Customers c  
JOIN Orders o ON c.customer\_id = o.customer\_id  
JOIN Reservations

r ON c.customer\_id = r.customer\_id;

B. SELECT COUNT(\*) FROM Customers c WHERE c.customer\_id IN (SELECT customer\_id FROM Orders) AND  
c.customer\_id IN (SELECT customer\_id FROM Reservations);

C. SELECT COUNT(\*) FROM Customers c WHERE c.customer\_id IN (SELECT customer\_id FROM Orders)  
INTERSECT SELECT customer\_id FROM Reservations;

D. It is not possible to find such customers with a single SQL query.

**\*\*Answers:\*\***

1. B. SELECT c.first\_name, c.last\_name FROM Customers c JOIN Orders o ON c.customer\_id = o.customer\_id  
WHERE c.customer\_id NOT IN (SELECT r.customer\_id FROM Reservations r);

2. C. SELECT c.first\_name, c.last\_name FROM Customers c WHERE c.customer\_id = (SELECT customer\_id  
FROM Orders WHERE total\_amount = (SELECT MAX(total\_amount) FROM Orders));

3. A. It counts the number of orders placed after the last reservation date for each customer.

4. B. SELECT AVG(order\_count) FROM (SELECT customer\_id, COUNT(order\_id) as order\_count FROM Orders  
GROUP BY customer\_id) AS subquery;

5. A. SELECT e.first\_name, e.last\_name FROM Employees e JOIN Payments p ON e.employee\_id =  
p.employee\_id WHERE e.employee\_id IN (SELECT employee\_id FROM Orders);

6. B. SELECT c.first\_name, c.last\_name FROM Customers c LEFT JOIN Orders o ON c.customer\_id =  
o.customer\_id LEFT JOIN Reservations r ON c.customer\_id = r.customer\_id WHERE o.customer\_id IS NULL AND  
r.customer\_id IS NULL;

7. C. NOT NULL

8. B. SELECT category, MAX(price) FROM "Menu Items" GROUP BY category;

9. A. It selects the names of customers who have placed orders on or after January 1, 2023.

10. B. SELECT AVG(rating), menu\_item\_id FROM Reviews GROUP BY menu\_item\_id;

11. B. SELECT c.first\_name, c.last\_name, SUM(o.total\_amount) FROM Customers c JOIN Orders o ON  
c.customer\_id = o.customer\_id GROUP BY c.customer\_id;

12. A. It selects the names of customers who have placed orders with a total amount greater than 50.

13. A. SELECT c.first\_name, c.last\_name FROM Customers c JOIN Orders o ON c.customer\_id = o.customer\_id  
GROUP BY c.customer\_id HAVING COUNT(o.order\_id) >= 2;

14. A. It selects the names of the most expensive menu items in the "Appetizers" category.

15. B. SELECT c.first\_name, c.last\_name FROM Customers c WHERE c.customer\_id IN (SELECT customer\_id  
FROM Orders) AND c.customer\_id IN (SELECT customer\_id FROM Reservations) AND c.customer\_id IN (SELECT  
customer\_id FROM Reviews);

16. A. SELECT COUNT(\*) FROM Orders o JOIN "Tables" t ON o.table\_number = t.table\_number WHERE  
t.seating\_capacity >= 4;

17. D. SELECT name FROM "Menu Items" LEFT JOIN Reviews ON "Menu Items".item\_id = Reviews.item\_id  
WHERE Reviews.item\_id IS NULL;

18. B. SELECT c.first\_name, c.last\_name FROM Customers c WHERE c.customer\_id IN (SELECT customer\_id  
FROM Orders WHERE order\_date >= (SELECT MAX(reservation\_date) FROM Reservations WHERE customer\_id  
= c.customer\_id));

19. D. SELECT e.first\_name, e.last\_name FROM Employees e WHERE e.employee\_id IN (SELECT p.employee\_id  
FROM Payments p WHERE p.order\_id IN (SELECT order\_id FROM Orders WHERE total\_amount > 1000));

20. C. SELECT COUNT(\*) FROM Customers c WHERE c.customer\_id IN (SELECT customer\_id FROM Orders)  
INTERSECT SELECT customer\_id FROM Reservations;

A schema related to a flight system can be quite complex, as it involves various components and systems to ensure the safe and efficient operation of an aircraft. Below is a simplified schema outlining some of the key components and their interconnections in a typical commercial aircraft flight system:

1. **Aircraft Structure**

- **Fuselage**: The main body of the aircraft, which houses passengers, cargo, and some of the critical systems.
- **Wings**: Generate lift and house fuel tanks.
- **Tail Section**: Includes the horizontal and vertical stabilizers and control surfaces.

2. **Aircraft Power**

- **Engines**: Responsible for generating thrust to propel the aircraft.
- **Fuel System**: Stores and distributes fuel to the engines.
- **APU (Auxiliary Power Unit)**: Provides power on the ground and acts as a backup power source.

3. **Flight Control Systems**

- **Cockpit**: Where pilots control the aircraft.
- **Control Surfaces**: Ailerons, elevators, rudders, and flaps for maneuvering.
- **Fly-by-Wire System**: Electronic control of flight surfaces.
- **AutoPilot**: For automated navigation and control.

4. **Navigation and Communication**

- **Avionics**: Electronics for navigation, communication, and monitoring.
- **GPS**: Provides precise location data.
- **Radar and Transponders**: Used for traffic and weather monitoring.

5. **Hydraulics and Landing Gear**

- **Hydraulic Systems**: Control landing gear, flaps, and other systems.
- **Landing Gear**: Wheels and struts for takeoff and landing.

6. **Electrical Systems**

- **Power Distribution**: Provides power to various systems.
- **Lighting**: Interior and exterior lighting.
- **Entertainment Systems**: In-flight entertainment for passengers.

7. **Environmental Control**

- **Air Conditioning and Pressurization**: Maintains a comfortable cabin environment

1. **Question:** In the flight system schema, which table would store information about passengers on a particular flight?

- A. `Aircraft\_Structure`
- B. `Flight\_Control\_Systems`
- C. `Navigation\_Communication`
- D. `Passengers`

**Answer:** D

2. **Question:** What SQL statement is used to retrieve the names of all airports from the `Airports` table in the flight system schema?

- A. `SELECT \* FROM Airports`
- B. `SELECT Airport\_Name FROM Airports`
- C. `GET Airport\_Name FROM Airports`
- D. `SHOW NAMES FROM Airports`

**Answer:** B

3. **Question:** Which SQL command would you use to find the total number of engines on all aircraft in the `Aircraft\_Power` table?

- A. `SELECT COUNT(\*) FROM Aircraft\_Power`
- B. `SUM(Engines) FROM Aircraft\_Power`
- C. `TOTAL(Engines) FROM Aircraft\_Power`
- D. `COUNT(Engines) FROM Aircraft\_Power`

**Answer:** A

4. **Question:** To retrieve a list of all airports with ICAO codes starting with 'K' from the `Airports` table, which SQL statement should you use?

- A. `SELECT \* FROM Airports WHERE ICAO\_Code LIKE 'K%'`
- B. `SELECT \* FROM Airports WHERE ICAO\_Code = 'K%'`
- C. `SELECT \* FROM Airports WHERE ICAO\_Code STARTS WITH 'K'`
- D. `SELECT \* FROM Airports WHERE ICAO\_Code = 'K'`

**Answer:** A

5. **Question:** What SQL command would you use to find the average passenger capacity of all aircraft in the `Aircraft\_Structure` table?

- A. `SELECT AVG(Passenger\_Capacity) FROM Aircraft\_Structure`
- B. `AVERAGE(Passenger\_Capacity) FROM Aircraft\_Structure`
- C. `SUM(Passenger\_Capacity) FROM Aircraft\_Structure`
- D. `AVG(Passenger\_Capacity) FROM Aircraft\_Structure`

**Answer:** A

6. **Question:** Which SQL statement would retrieve the flight numbers and destinations for flights that use the `AutoPilot` system from the `Flight\_Control\_Systems` table?

- A. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes'`
  - B. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE AutoPilot = 1`
  - C. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE AutoPilot = 'Enabled'`
  - D. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE AutoPilot = 'On'`
- \*\*Answer:\*\* A**

7. **\*\*Question:\*\*** To retrieve the names of all passengers whose last name is 'Smith' from the `Passengers` table, which SQL statement should you use?

- A. `SELECT First\_Name FROM Passengers WHERE Last\_Name = 'Smith'`
- B. `SELECT Last\_Name, First\_Name FROM Passengers WHERE Last\_Name = 'Smith'`
- C. `SELECT First\_Name FROM Passengers WHERE Last\_Name LIKE 'Smith'`
- D. `SELECT Last\_Name, First\_Name FROM Passengers WHERE First\_Name = 'Smith'`

**\*\*Answer:\*\* B**

8. **\*\*Question:\*\*** What SQL command would you use to find the distinct ICAO codes of all airports in the `Airports` table?

- A. `SELECT DISTINCT ICAO\_Code FROM Airports`
- B. `SELECT UNIQUE ICAO\_Code FROM Airports`
- C. `SELECT ICAO\_Code FROM Airports GROUP BY ICAO\_Code`
- D. `SELECT DISTINCT ICAO\_Code FROM Airports GROUP BY ICAO\_Code`

**\*\*Answer:\*\* A**

9. **\*\*Question:\*\*** To retrieve the flight numbers and destinations for flights that have more than 200 passengers on board, which SQL statement should you use?

- A. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE Passenger\_Count > 200`
- B. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems HAVING Passenger\_Count > 200`
- C. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE Passenger\_Count < 200`
- D. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems HAVING Passenger\_Count < 200`

**\*\*Answer:\*\* A**

10. **\*\*Question:\*\*** What SQL statement would you use to retrieve the names of all passengers and their corresponding flight numbers from the `Passengers` and `Flight\_Control\_Systems` tables, respectively, where the passenger is on a flight that uses the `AutoPilot` system?

- A. `SELECT P.Passenger\_Name, F.Flight\_Number FROM Passengers P, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.AutoPilot = 'Yes'`

- B. `SELECT Passenger\_Name, Flight\_Number FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes')`
  - C. `SELECT Passenger\_Name, Flight\_Number FROM Passengers JOIN Flight\_Control\_Systems ON Passengers.Flight\_ID = Flight\_Control\_Systems.Flight\_ID WHERE AutoPilot = 'Yes'`
  - D. `SELECT Passenger\_Name, Flight\_Number FROM Passengers WHERE Flight\_ID = (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes')`
- \*\*Answer:\*\* C**



1. **Question:** What SQL query retrieves the flight numbers and destinations for flights that have the highest passenger capacity from the `Flight\_Control\_Systems` table?

- A. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE Passenger\_Capacity = MAX(Passenger\_Capacity)`
- B. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE Passenger\_Capacity = (SELECT MAX(Passenger\_Capacity) FROM Flight\_Control\_Systems)`
- C. `SELECT MAX(Flight\_Number), MAX(Destination) FROM Flight\_Control\_Systems`
- D. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems ORDER BY Passenger\_Capacity DESC LIMIT 1`

**Answer:** B

2. **Question:** Which SQL query would you use to find the airports where at least two flights from the `Flight\_Control\_Systems` table are scheduled to land?

- A. `SELECT Airport\_Name FROM Airports WHERE Airport\_Code IN (SELECT Destination FROM Flight\_Control\_Systems HAVING COUNT(\*) >= 2)`
- B. `SELECT Airport\_Name FROM Airports WHERE Airport\_Code IN (SELECT Destination FROM Flight\_Control\_Systems GROUP BY Destination HAVING COUNT(\*) >= 2)`
- C. `SELECT Airport\_Name FROM Airports WHERE Airport\_Code IN (SELECT DISTINCT Destination FROM Flight\_Control\_Systems HAVING COUNT(\*) >= 2)`
- D. `SELECT Airport\_Name FROM Airports WHERE Airport\_Code IN (SELECT Destination FROM Flight\_Control\_Systems WHERE Flight\_Number IN (SELECT Flight\_Number FROM Flight\_Control\_Systems GROUP BY Flight\_Number HAVING COUNT(\*) >= 2))`

**Answer:** B

3. **Question:** How can you retrieve the passenger names and their corresponding flight destinations from the `Passengers` and `Flight\_Control\_Systems` tables, respectively, for passengers who are on flights with more than 300 passengers?

- A. `SELECT P.Passenger\_Name, F.Destination FROM Passengers P, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Passenger\_Count > 300`
- B. `SELECT Passenger\_Name, Destination FROM Passengers JOIN Flight\_Control\_Systems ON Passengers.Flight\_ID = Flight\_Control\_Systems.Flight\_ID WHERE Flight\_Control\_Systems.Passenger\_Count > 300`
- C. `SELECT Passenger\_Name, Destination FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count > 300)`
- D. `SELECT P.Passenger\_Name, F.Destination FROM Passengers P JOIN Flight\_Control\_Systems F ON P.Flight\_ID = F.Flight\_ID WHERE F.Passenger\_Count > 300`

**Answer:** B

4. **Question:** Which SQL query retrieves the flight numbers and destinations of flights using the `AutoPilot` system with passenger capacities greater than the average passenger capacity of all flights in the `Flight\_Control\_Systems` table?

- A. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes' AND Passenger\_Capacity > AVG(Passenger\_Capacity)`

- B. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes' AND Passenger\_Capacity > (SELECT AVG(Passenger\_Capacity) FROM Flight\_Control\_Systems)`
- C. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE AutoPilot = 'Enabled' AND Passenger\_Capacity > (SELECT AVG(Passenger\_Capacity) FROM Flight\_Control\_Systems)`
- D. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE AutoPilot = 'On' AND Passenger\_Capacity > AVG(Passenger\_Capacity) GROUP BY Flight\_Number, Destination`

**\*\*Answer:\*\* B**

5. **\*\*Question:\*\*** To find the total number of passengers on flights that use the `AutoPilot` system and have a passenger capacity greater than 250, which SQL query should you use?

- A. `SELECT SUM(Passenger\_Count) FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes' AND Passenger\_Capacity > 250`
- B. `SELECT COUNT(\*) FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes' AND Passenger\_Capacity > 250)`
- C. `SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes' AND Passenger\_Capacity > 250`
- D. `SELECT SUM(Passenger\_Count) FROM Flight\_Control\_Systems WHERE AutoPilot = 'Yes' HAVING Passenger\_Capacity > 250`

**\*\*Answer:\*\* A**

6. **\*\*Question:\*\*** What SQL query retrieves the airport names from the `Airports` table where the number of flights departing from them is greater than the number of flights arriving at them according to the `Flight\_Control\_Systems` table?

- A. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code)`
- B. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) GROUP BY Airport\_Name`
- C. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) HAVING COUNT(\*) > 0`
- D. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code)`

**\*\*Answer:\*\* B**

7. **\*\*Question:\*\*** To find the names of passengers who are on flights to the same destination as another passenger, which SQL query would you use?

- A. `SELECT DISTINCT P1.Passenger\_Name, P2.Passenger\_Name FROM Passengers P1, Passengers P2, Flight\_Control\_Systems F WHERE P1.Destination = P2.Destination AND P1.Flight\_ID = F.Flight\_ID AND P2.Flight\_ID = F.Flight\_ID`
- B. `SELECT P1.Passenger\_Name, P2.Passenger\_Name FROM Passengers P1, Passengers P2 WHERE P1.Destination = P2.Destination`
- C. `SELECT DISTINCT P.Passenger\_Name FROM Passengers P WHERE P.Flight\_ID IN (SELECT Flight\_ID FROM Passengers WHERE Destination = P.Destination)`
- D. `SELECT P1.Passenger\_Name, P2.Passenger\_Name FROM Passengers P1, Passengers P2 WHERE P1.Destination = P2.Destination AND P1.Flight\_ID = P2.Flight\_ID`

**\*\*Answer:\*\* A**

8. **\*\*Question:\*\*** What SQL query retrieves the names of passengers who are on flights that have the lowest passenger count according to the `Flight\_Control\_Systems` table?

- A. `SELECT Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = MIN(Passenger\_Count))`
- B. `SELECT Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = (SELECT MIN(Passenger\_Count) FROM Flight\_Control\_Systems))`
- C. `SELECT Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = (SELECT MIN(Passenger\_Count) FROM Flight\_Control\_Systems) GROUP BY Flight\_ID)`
- D. `SELECT Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = MIN(SELECT Passenger\_Count FROM Flight\_Control\_Systems))`

**\*\*Answer:\*\* B**

9. **\*\*Question:\*\*** Which SQL query retrieves the passenger names and the corresponding destination airports from the `Passengers` and `Airports` tables, respectively, for passengers who are on flights departing from airports with ICAO codes starting with 'K'?

- A. `SELECT P.Passenger\_Name, A.Airport\_Name FROM Passengers P, Airports A, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Departure = A.Airport\_Code AND A.ICAO\_Code LIKE 'K%'`
- B. `SELECT P.Passenger\_Name, A.Airport\_Name FROM Passengers P, Airports A, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Departure = A.Airport\_Code AND A.ICAO\_Code = 'K%'`
- C. `SELECT P.Passenger\_Name, A.Airport\_Name FROM Passengers P, Airports A, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Departure = A.Airport\_Code AND A.ICAO\_Code LIKE 'K%'`
- D. `SELECT P.Passenger\_Name, A.Airport\_Name FROM Passengers P, Airports A, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Departure = A.Airport\_Code AND A.ICAO\_Code = 'K'`

**\*\*Answer:\*\* A**

10. **Question:** To retrieve the flight numbers and destinations of flights that have more than twice the passenger capacity of the average passenger capacity of all flights in the `Flight\_Control\_Systems` table, which SQL query should you use?

- A. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE Passenger\_Capacity > 2 \* AVG(Passenger\_Capacity)`
- B. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE Passenger\_Capacity > 2 \* (SELECT AVG(Passenger\_Capacity) FROM Flight\_Control\_Systems)`
- C. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems WHERE Passenger\_Capacity > AVG(Passenger\_Capacity) \* 2`
- D. `SELECT Flight\_Number, Destination FROM Flight\_Control\_Systems GROUP BY Flight\_Number, Destination HAVING Passenger\_Capacity > AVG(Passenger\_Capacity) \* 2`

**Answer:** B

11. **Question:** How can you retrieve the passenger names and their corresponding flight destinations from the `Passengers` and `Flight\_Control\_Systems` tables, respectively, for passengers who are on flights with the maximum passenger capacity?

- A. `SELECT P.Passenger\_Name, F.Destination FROM Passengers P, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Passenger\_Capacity = MAX(F.Passenger\_Capacity)`
- B. `SELECT Passenger\_Name, Destination FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Capacity = (SELECT MAX(Passenger\_Capacity) FROM Flight\_Control\_Systems))`
- C. `SELECT P.Passenger\_Name, F.Destination FROM Passengers P JOIN Flight\_Control\_Systems F ON P.Flight\_ID = F.Flight\_ID WHERE F.Passenger\_Capacity = (SELECT MAX(Passenger\_Capacity) FROM Flight\_Control\_Systems)`
- D. `SELECT P.Passenger\_Name, F.Destination FROM Passengers P, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Passenger\_Capacity = (SELECT MAX(Passenger\_Capacity) FROM Flight\_Control\_Systems) GROUP BY P.Passenger\_Name, F.Destination`

**Answer:** B

12. **Question:** What SQL query retrieves the airport names from the `Airports` table where the number of flights departing from them is equal to the number of flights arriving at them according to the `Flight\_Control\_Systems` table?

- A. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code)`
- B. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) GROUP BY Airport\_Name`

- C. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) HAVING COUNT(\*) > 0`

- D. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code)`

**\*\*Answer:\*\* B**

13. **\*\*Question:\*\*** Which SQL query retrieves the passenger names who are on flights to the same destination as another passenger and are on flights with the maximum passenger capacity according to the `Flight\_Control\_Systems` table?

- A. `SELECT DISTINCT P1.Passenger\_Name, P2.Passenger\_Name FROM Passengers P1, Passengers P2, Flight\_Control\_Systems F WHERE P1.Destination = P2.Destination AND P1.Flight\_ID = F.Flight\_ID AND P2.Flight\_ID = F.Flight\_ID AND F.Passenger\_Capacity = (SELECT MAX(Passenger\_Capacity) FROM Flight\_Control\_Systems)`

- B. `SELECT P1.Passenger\_Name, P2.Passenger\_Name FROM Passengers P1, Passengers P2 WHERE P1.Destination = P2.Destination AND P1.Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Capacity = (SELECT MAX(Passenger\_Capacity) FROM Flight\_Control\_Systems))`

- C. `SELECT DISTINCT P.Passenger\_Name FROM Passengers P WHERE P.Flight\_ID IN (SELECT Flight\_ID FROM Passengers WHERE Destination = P.Destination) AND P.Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Capacity = (SELECT MAX(Passenger\_Capacity) FROM Flight\_Control\_Systems))`

- D. `SELECT P1.Passenger\_Name, P2.Passenger\_Name FROM Passengers P1, Passengers P2 WHERE P1.Destination = P2.Destination AND P1.Flight\_ID = P2.Flight\_ID AND P1.Flight\_ID IN (SELECT Flight\_ID FROM

Flight\_Control\_Systems WHERE Passenger\_Capacity = (SELECT MAX(Passenger\_Capacity) FROM Flight\_Control\_Systems))`

**\*\*Answer:\*\* A**

14. **\*\*Question:\*\*** What SQL query retrieves the passenger names who are on flights that have the highest passenger count and are departing from airports with ICAO codes starting with 'K'?

- A. `SELECT DISTINCT P.Passenger\_Name FROM Passengers P, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Departure IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%') AND F.Passenger\_Count = (SELECT MAX(Passenger\_Count) FROM Flight\_Control\_Systems)`

- B. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = (SELECT MAX(Passenger\_Count) FROM Flight\_Control\_Systems) AND Departure IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%'))`

- C. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Departure IN (SELECT Airport\_Code FROM

Airports WHERE ICAO\_Code LIKE 'K%') AND Passenger\_Count = (SELECT MAX(Passenger\_Count) FROM Flight\_Control\_Systems))`

- D. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = (SELECT MAX(Passenger\_Count) FROM Flight\_Control\_Systems) AND Departure IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%'))`

**\*\*Answer:\*\* A**

15. **\*\*Question:\*\*** How can you retrieve the airport names from the `Airports` table where the number of flights departing from them is equal to the number of flights arriving at them and the passenger count is greater than 200 according to the `Flight\_Control\_Systems` table?

- A. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 200)`

- B. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT DISTINCT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 200)`

- C. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 200)`

- D. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 200) HAVING COUNT(\*) > 0`

**\*\*Answer:\*\* B**

16. **\*\*Question:\*\*** Which SQL query retrieves the passenger names who are on flights that have the lowest passenger count and are departing from airports with ICAO codes starting with 'K'?

- A. `SELECT P.Passenger\_Name FROM Passengers P, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Departure IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%') AND F.Passenger\_Count = (SELECT MIN(Passenger\_Count) FROM Flight\_Control\_Systems)`

- B. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = (SELECT MIN(Passenger\_Count) FROM Flight\_Control\_Systems) AND Departure IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%'))`

- C. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Departure IN (SELECT Airport\_Code FROM

Airports WHERE ICAO\_Code LIKE 'K%') AND Passenger\_Count = (SELECT MIN(Passenger\_Count) FROM Flight\_Control\_Systems))`

- D. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = (SELECT MIN(Passenger\_Count) FROM Flight\_Control\_Systems) AND Departure IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%'))`

**\*\*Answer:\*\* A**

17. **\*\*Question:\*\*** What SQL query retrieves the airport names from the `Airports` table where the number of flights departing from them is greater than the number of flights arriving at them and the passenger count is greater than 100 according to the `Flight\_Control\_Systems` table?

- A. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 100)`

- B. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT DISTINCT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 100)`

- C. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 100)`

- D. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 100) HAVING COUNT(\*) > 0`

**\*\*Answer:\*\* B**

18. **\*\*Question:\*\*** How can you retrieve the airport names from the `Airports` table where the number of flights departing from them is equal to the number of flights arriving at them and the passenger count is greater than 50 according to the `Flight\_Control\_Systems` table?

- A. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 50)`

- B. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code

IN (SELECT DISTINCT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 50)`

- C. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 50)`

- D. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) = (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 50) HAVING COUNT(\*) > 0`

**\*\*Answer:\*\* B**

19. **\*\*Question:\*\*** Which SQL query retrieves the passenger names who are on flights that have the highest passenger count and are arriving at airports with ICAO codes starting with 'K'?

- A. `SELECT P.Passenger\_Name FROM Passengers P, Flight\_Control\_Systems F WHERE P.Flight\_ID = F.Flight\_ID AND F.Destination IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%') AND F.Passenger\_Count = (SELECT MAX(Passenger\_Count) FROM Flight\_Control\_Systems)`

- B. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = (SELECT MAX(Passenger\_Count) FROM Flight\_Control\_Systems) AND Destination IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%'))`

- C. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Destination IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%') AND Passenger\_Count = (SELECT MAX(Passenger\_Count) FROM Flight\_Control\_Systems))`

- D. `SELECT P.Passenger\_Name FROM Passengers WHERE Flight\_ID IN (SELECT Flight\_ID FROM Flight\_Control\_Systems WHERE Passenger\_Count = (SELECT MAX(Passenger\_Count) FROM Flight\_Control\_Systems) AND Destination IN (SELECT Airport\_Code FROM Airports WHERE ICAO\_Code LIKE 'K%'))`

**\*\*Answer:\*\* A**

20. **\*\*Question:\*\*** What SQL query retrieves the airport names from the `Airports` table where the number of flights departing from them is greater than the number of flights arriving at them and the passenger count is greater than 75 according to the `Flight\_Control\_Systems` table?

- A. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 75)`

- B. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT DISTINCT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 75)`



- C. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(\*) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 75)`

- D. `SELECT Airport\_Name FROM Airports WHERE (SELECT COUNT(Departure) FROM Flight\_Control\_Systems WHERE Departure = Airport\_Code) > (SELECT COUNT(Destination) FROM Flight\_Control\_Systems WHERE Destination = Airport\_Code) AND Airport\_Code IN (SELECT Departure FROM Flight\_Control\_Systems WHERE Passenger\_Count > 75) HAVING COUNT(\*) > 0`

**\*\*Answer:\*\*** B

### MCQ based on cursor, function and procedure in PL/SQL

1. What is PL/SQL?

- a. A database management system
- b. A programming language for Oracle databases
- c. A web development framework
- d. A data visualization tool

Answer: b. A programming language for Oracle databases

2. What is the purpose of a cursor in PL/SQL?

- a. To declare variables
- b. To define user-defined functions
- c. To retrieve and process rows from a result set
- d. To create database tables

Answer: c. To retrieve and process rows from a result set

3. Which PL/SQL construct is used to return a single value from a function?

- a. Cursor
- b. Procedure
- c. Trigger
- d. Return statement

Answer: d. Return statement

4. Which keyword is used to create a new procedure in PL/SQL?

- a. DECLARE
- b. BEGIN
- c. CREATE
- d. PROCEDURE

Answer: d. PROCEDURE

5. Which of the following is true about a PL/SQL function?

- a. It cannot return a value.
- b. It can return single values.
- c. It cannot accept parameters.
- d. It cannot contain SQL statements.

Answer: b. It can return single values.

6. What is the primary purpose of EXCEPTION handling in PL/SQL?

- a. To define user roles
- b. To declare variables
- c. To handle errors and exceptions gracefully
- d. To create database triggers

Answer: c. To handle errors and exceptions gracefully

7. Which keyword is used to pass parameters to a PL/SQL procedure?

- a. PARAM
- b. VALUE
- c. IN
- d. OUT

Answer: c. IN

8. Which PL/SQL construct is used to iterate through the rows of a result set returned by a query?

- a. FOR loop
- b. WHILE loop
- c. CASE statement
- d. IF statement

Answer: a. FOR loop

9. What is the primary difference between a function and a procedure in PL/SQL?

- a. A function returns a value, while a procedure does not.
- b. A procedure returns a value, while a function does not.
- c. A function can accept parameters, while a procedure cannot.
- d. A procedure can be called from SQL queries, while a function cannot.

Answer: a. A function returns a value, while a procedure does not.

10. Which PL/SQL construct is used to handle runtime errors explicitly?

- a. DECLARE
- b. EXCEPTION
- c. BEGIN
- d. RETRY

Answer: b. EXCEPTION

here are 10 challenging multiple-choice questions (MCQs) related to PL/SQL programming with a focus on a "Student" table:

Assume we have a "Student" table with the following columns: "StudentID," "FirstName," "LastName," "Age," and "GPA."

1. Which PL/SQL construct is commonly used to retrieve data from the "Student" table?

- a. Procedure
- b. Cursor
- c. Function

d. Trigger

Answer: b. Cursor

2. What is the purpose of the following PL/SQL block?

```
``psql
DECLARE
    total_students NUMBER;
BEGIN
    SELECT COUNT() INTO total_students FROM Student;
    DBMS_OUTPUT.PUT_LINE('Total students: ' || total_students);
END;
``
```

- a. Deletes all records from the "Student" table.
- b. Calculates the total number of students in the "Student" table and displays it.
- c. Updates the "Age" column of all students in the "Student" table.
- d. Inserts a new student record into the "Student" table.

Answer: b. Calculates the total number of students in the "Student" table and displays it.

3. What is the purpose of the following PL/SQL block?

```
``psql
DECLARE
    student_name VARCHAR2(50);
BEGIN
    SELECT FirstName || ' ' || LastName INTO student_name FROM Student WHERE StudentID = 101;
    DBMS_OUTPUT.PUT_LINE('Student name: ' || student_name);
END;
```

...

- a. Inserts a new student record into the "Student" table.
- b. Updates the "Age" column of all students in the "Student" table.
- c. Retrieves the full name of the student with StudentID 101 and displays it.
- d. Deletes the student with StudentID 101 from the "Student" table.

Answer: c. Retrieves the full name of the student with StudentID 101 and displays it.

4. Which PL/SQL construct is used to handle exceptions that may occur during the execution of a PL/SQL program?

- a. Cursor
- b. Function
- c. Exception handling block
- d. Trigger

Answer: c. Exception handling block

5. What will the following PL/SQL block do?

```
```sql
BEGIN
    DELETE FROM Student WHERE Age < 18;
    COMMIT;
END;
...

```

- a. Deletes all records from the "Student" table.
- b. Deletes students who are 18 years or older from the "Student" table and saves the changes permanently.
- c. Rolls back all changes made to the "Student" table.
- d. Updates the "Age" column of all students in the "Student" table.

Answer: b. Deletes students who are 18 years or older from the "Student" table and saves the changes permanently.

6. Which PL/SQL construct is commonly used to update records in the "Student" table based on specific conditions?

- a. Procedure
- b. Cursor
- c. Function
- d. Trigger

Answer: a. Procedure

7. What is the primary purpose of the following PL/SQL block?

```
```sql
DECLARE
    avg_gpa NUMBER;
BEGIN
    SELECT AVG(GPA) INTO avg_gpa FROM Student;
    DBMS_OUTPUT.PUT_LINE('Average GPA: ' || avg_gpa);
END;
```
```

- a. Inserts a new student record into the "Student" table.
- b. Updates the "Age" column of all students in the "Student" table.
- c. Calculates the average GPA of all students in the "Student" table and displays it.
- d. Deletes all records from the "Student" table.

Answer: c. Calculates the average GPA of all students in the "Student" table and displays it.

8. Which PL/SQL construct is used to execute a set of statements repeatedly until a condition is met?

- a. FOR loop
- b. WHILE loop
- c. CASE statement
- d. IF statement

Answer: b. WHILE loop

9. What is the purpose of the following PL/SQL block?

```
``psql
BEGIN
    UPDATE Student SET Age = Age + 1;
    COMMIT;
END;
...
```

- a. Deletes all records from the "Student" table.
- b. Updates the "Age" column of all students in the "Student" table by incrementing it by 1 and saves the changes permanently.
- c. Rolls back all changes made to the "Student" table.
- d. Inserts a new student record into the "Student" table.

Answer: b. Updates the "Age" column of all students in the "Student" table by incrementing it by 1 and saves the changes permanently.

10. What will the following PL/SQL block do?

```
``psql
BEGIN
    INSERT INTO Student (StudentID, FirstName, LastName, Age, GPA)
    VALUES (102, 'Alice', 'Smith', 20, 3.8);
    COMMIT;
```



END;

...

- a. Deletes all records from the "Student" table.
- b. Inserts a new student record with the given details into the "Student" table and saves the changes permanently.
- c. Rolls back all changes made to the "Student" table.
- d. Updates the "Age" column of all students in the "Student" table.

Answer: b. Inserts a new student record with the given details into the "Student" table and saves the changes permanently.

1. What is a stored program unit in a database?

- a. Table
- b. Procedure
- c. Trigger
- d. View

Answer: b. Procedure

2. Which of the following is not a part of a procedure in a database?

- a. Declaration section
- b. Exception section
- c. Body section
- d. Trigger section

Answer: d. Trigger section

3. In parameter modes for procedures, which mode allows you to pass values from the calling program to the procedure and vice versa?

- a. IN
- b. OUT
- c. IN OUT
- d. DEFAULT

Answer: c. IN OUT

4. What is one of the advantages of using procedures in a database?

- a. Simplified data modeling
- b. Enhanced security
- c. Dynamic table creation
- d. Reduced data redundancy

Answer: b. Enhanced security

5. Which of the following is not a type of trigger in a database?

- a. Before Trigger
- b. After Trigger
- c. During Trigger
- d. Instead Of Trigger

Answer: c. During Trigger

6. What is the syntax for creating a trigger in SQL?

- a. CREATE PROCEDURE
- b. CREATE FUNCTION
- c. CREATE TRIGGER
- d. CREATE VIEW

Answer: c. CREATE TRIGGER

7. Which type of trigger is fired before the execution of a DML statement in SQL?

- a. Before Trigger
- b. After Trigger
- c. Instead Of Trigger
- d. Concurrent Trigger

Answer: a. Before Trigger

8. What is the purpose of a package specification in a database?

- a. Contains the implementation details of a package
- b. Declares the public interface of a package
- c. Stores data for the package
- d. Executes procedures in the package

Answer: b. Declares the public interface of a package

9. Which part of a package in a database contains the actual code and implementation details?

- a. Package specification
- b. Package body
- c. Package header
- d. Package declaration

Answer: b. Package body

10. What is a "bodiless package" in a database?

- a. A package without a package specification
- b. A package without a package body
- c. A package without any code or implementation
- d. A package without parameters

Answer: b. A package without a package body

11. Which of the following is not an advantage of using packages in a database?

- a. Encapsulation of code
- b. Improved performance
- c. Simplified maintenance
- d. Increased data redundancy

Answer: d. Increased data redundancy

12. What does a "PRAGMA AUTONOMOUS\_TRANSACTION" do in a trigger?

- a. Executes the trigger automatically
- b. Is used to create a trigger
- c. Starts a new transaction within the trigger

d. Stops a running transaction

Answer: c. Starts a new transaction within the trigger

13. Which of the following is a valid parameter mode for a procedure in SQL?

- a. IN OUT INCREMENT
- b. OUT DECREMENT
- c. INCREMENT OUT
- d. OUT IN

Answer: a. IN OUT INCREMENT

14. What type of trigger is used to replace a DML statement with another statement?

- a. Before Trigger
- b. After Trigger
- c. Instead Of Trigger
- d. Around Trigger

Answer: c. Instead Of Trigger

15. What is the primary purpose of a trigger in a database?

- a. To define data types
- b. To declare variables
- c. To enforce data integrity and automate actions
- d. To create views

Answer: c. To enforce data integrity and automate actions

16. Which of the following is not a valid part of a procedure in SQL?

- a. Parameter list
- b. Declaration section

- c. Exception section
- d. Trigger section

Answer: d. Trigger section

17. What is the primary purpose of a package in SQL?

- a. To store data
- b. To define triggers
- c. To group related procedures, functions, and variables
- d. To create views

Answer: c. To group related procedures, functions, and variables

18. Which parameter mode allows a procedure to receive values from the calling program but not return values back?

- a. IN
- b. OUT
- c. IN OUT
- d. DEFAULT

Answer: a. IN

19. Which of the following is not a type of package in SQL?

- a. Standalone package
- b. Bodiless package
- c. Composite package
- d. Nested package

Answer: c. Composite package

20. What is the advantage of using triggers in a database?

- a. Improved code encapsulation
- b. Enhanced security
- c. Simplified package creation
- d. Dynamic table creation

Answer: b. Enhanced security

1. What is the primary purpose of a transaction in a DBMS?

- A. To retrieve data from the database
- B. To update the database
- C. To manage database schemas
- D. To organize data into tables

Answer: B

2. Which of the following is not a property of a transaction in DBMS?

- A. Isolation
- B. Atomicity
- C. Consistency
- D. Transparency

Answer: D

3. Which property of transactions ensures that a transaction's changes are permanent and will survive system failures?

- A. Atomicity
- B. Consistency
- C. Isolation
- D. Durability

Answer: D

4. What does the ACID acronym stand for in the context of transactions?

- A. Atomicity, Completeness, Isolation, Durability
- B. Atomicity, Consistency, Isolation, Durability
- C. Availability, Consistency, Isolation, Durability
- D. Allotment, Concurrency, Integration, Durability

Answer: B

5. Which of the following is not a common concurrency problem in DBMS?

- A. Deadlock
- B. Dirty Read
- C. Lost Update
- D. Normalization

Answer: D

6. What is the primary goal of a concurrency control mechanism in a DBMS?

- A. To ensure that transactions are executed concurrently without any restrictions
- B. To improve the performance of the database system
- C. To prevent conflicts and maintain data consistency in a multi-user environment
- D. To reduce the storage space required for the database

Answer: C

7. Which of the following is a common method for achieving isolation in DBMS?

- A. Two-Phase Locking
- B. Time-based Synchronization
- C. Data Duplication
- D. Data Sharding



Answer: A

8. What is a serializable schedule in the context of concurrency control?

- A. A schedule in which transactions are executed one after the other
- B. A schedule that preserves the original order of transactions
- C. A schedule that produces the same result as if transactions were executed serially
- D. A schedule that allows concurrent execution without any restrictions

Answer: C

9. Which of the following is a benefit of using serializability in a DBMS?

- A. Improved system performance
- B. Reduced data consistency
- C. Enhanced data integrity
- D. Faster query processing

Answer: C

10. What is the primary purpose of a lock in a DBMS?

- A. To restrict access to specific data items
- B. To encrypt data for security
- C. To optimize query execution
- D. To permanently delete data

Answer: A

11. In a DBMS, what is a shared lock used for?

- A. To allow multiple transactions to write to the same data item
- B. To prevent multiple transactions from reading the same data item simultaneously
- C. To allow multiple transactions to read the same data item simultaneously
- D. To break deadlocks

Answer: C

12. Which concurrency control technique uses a timeout mechanism to resolve conflicts?

- A. Two-Phase Locking
- B. Timestamp-Based Protocol
- C. Optimistic Concurrency Control
- D. Strict Two-Phase Locking

Answer: C

13. What is a deadlock in the context of concurrency control?

- A. A situation where two transactions are waiting for each other to release locks
- B. A situation where a transaction is permanently blocked
- C. A situation where transactions cannot be rolled back
- D. A situation where transactions cannot be committed

Answer: A

14. What is the purpose of a deadlock detection mechanism in a DBMS?

- A. To prevent deadlocks from occurring
- B. To identify and resolve deadlocks
- C. To escalate conflicts between transactions
- D. To increase the isolation level

Answer: B

15. Which of the following is not a common deadlock prevention technique?

- A. Wait-Die
- B. Wound-Wait
- C. Timeout

D. Rollback

Answer: C

16. What is the purpose of an intent lock in a DBMS?

- A. To indicate the intention of a transaction to acquire a shared lock
- B. To indicate the intention of a transaction to acquire an exclusive lock
- C. To prevent deadlocks
- D. To release all locks

Answer: B

17. In a DBMS, what is the purpose of a transaction log?

- A. To record all user queries
- B. To store database schema information
- C. To maintain a record of all committed and uncommitted transactions
- D. To store backup copies of data

Answer: C

18. What is the primary goal of a checkpoint in a DBMS?

- A. To initiate a transaction rollback
- B. To recover from system crashes
- C. To release all locks held by a transaction
- D. To optimize query execution

Answer: B

19. Which of the following is an example of a conflict-serializable schedule in a DBMS?

- A. Schedule S1:  $T1 \rightarrow T2 \rightarrow T3$
- B. Schedule S2:  $T2 \rightarrow T1 \rightarrow T3$

C. Schedule S3: T1 → T3 → T2

D. Schedule S4: T3 → T1 → T2

Answer: A

20. What does the isolation level "Serializable" in a DBMS ensure?

A. It allows dirty reads

B. It provides the highest level of isolation

C. It allows transactions to write to the same data simultaneously

D. It does not allow any concurrency

Answer: B

21. Which of the following is a benefit of using a lower isolation level, such as "Read Uncommitted," in a DBMS?

A. Improved data integrity

B. Higher isolation between transactions

C. Reduced contention for locks

D. Faster query performance

Answer: D

22. In the context of locking, what is a lock mode?

A. The time duration for which a lock is held

B. The type of lock (shared or exclusive) and its compatibility with other locks

C. The order in which locks are acquired

D. The number of transactions waiting for a lock

Answer: B

23. Which of the following is a drawback of using a high isolation level, such as "Serializable," in a DBMS?

- A. Increased likelihood of deadlocks
- B. Improved data consistency
- C. Lower transaction throughput
- D. Reduced data integrity

Answer: C

24. What is the purpose of a transaction manager in a DBMS?

- A. To optimize query execution
- B. To manage database schemas
- C. To ensure the ACID properties of transactions
- D. To store backup copies of data

Answer: C

25. What is the primary goal

of a deadlock prevention technique like "Wait-Die"?

- A. To escalate conflicts between transactions
- B. To prevent transactions from waiting indefinitely
- C. To improve query performance
- D. To increase data redundancy

Answer: B

26. In a DBMS, what is a transaction's isolation level?

- A. The number of locks acquired by the transaction
- B. The duration for which a transaction is active
- C. The level of visibility a transaction has into other transactions' changes
- D. The number of concurrent transactions

Answer: C

27. Which of the following is a disadvantage of using optimistic concurrency control?

- A. Higher contention for locks
- B. Increased likelihood of deadlocks
- C. Slower query performance
- D. Limited data consistency

Answer: D

28. What is the purpose of a timestamp in a DBMS?

- A. To record the time when a transaction started
- B. To ensure data encryption
- C. To prevent conflicts between transactions
- D. To optimize query execution

Answer: A

29. What is a conflict-serializable schedule?

- A. A schedule that contains conflicts between transactions
- B. A schedule in which transactions are executed serially
- C. A schedule that preserves the original order of transactions
- D. A schedule that is equivalent to a serial schedule with the same transactions

Answer: D

30. Which of the following is not a common concurrency control mechanism in a DBMS?

- A. Two-Phase Commit
- B. Optimistic Concurrency Control
- C. Strict Two-Phase Locking
- D. Timestamp-Based Protocol

Answer: A

31. What is the purpose of a transaction ID in a DBMS?

- A. To identify the user who initiated the transaction
- B. To indicate the transaction's priority
- C. To uniquely identify and track each transaction
- D. To store backup copies of data

Answer: C

32. Which of the following is a benefit of using a higher isolation level, such as "Serializable," in a DBMS?

- A. Improved query performance
- B. Reduced likelihood of deadlocks
- C. Higher data consistency
- D. Lower transaction throughput

Answer: C

33. What is the primary goal of a timeout-based deadlock prevention technique?

- A. To prevent transactions from waiting indefinitely
- B. To prioritize transactions based on their importance
- C. To escalate conflicts between transactions
- D. To increase the isolation level

Answer: A

34. What does a "lost update" refer to in the context of concurrency control?

- A. A situation where a transaction is permanently blocked
- B. A situation where a transaction is rolled back

- C. A situation where one transaction overwrites the changes made by another transaction
- D. A situation where transactions cannot be committed

Answer: C

35. Which of the following is not a common technique for deadlock detection in a DBMS?

- A. Wait-Die
- B. Wound-Wait
- C. Timeout
- D. Rollback

Answer: D

36. What is the primary purpose of a deadlock prevention technique like "Wound-Wait"?

- A. To escalate conflicts between transactions
- B. To prevent transactions from waiting indefinitely
- C. To improve query performance
- D. To increase data redundancy

Answer: A

37. Which of the following statements about the "Repeatable Read" isolation level in a DBMS is true?

- A. It allows dirty reads.
- B. It allows lost updates.
- C. It prevents phantom reads.
- D. It has the lowest isolation level.

Answer: C

38. What is the primary goal of a transaction recovery manager in a DBMS?

- A. To escalate conflicts between transactions



- B. To optimize query execution
- C. To ensure the durability of transactions
- D. To manage database schemas

Answer: C

39. Which of the following is not a common cause of deadlocks in a DBMS?

- A. Circular Wait
- B. Resource Preemption
- C. Hold and Wait
- D. No Concurrency

Answer: D

40. What is the purpose of a data dictionary in a DBMS?

- A. To store user data
- B. To maintain a log of transactions
- C. To manage database schemas and metadata
- D. To perform data encryption

Answer: C

**Assuming we have a "Bank" table with the following sample data:**

| account_number | account_holder | balance |
|----------------|----------------|---------|
| 1001           | John Doe       | 5000.00 |
| 1002           | Jane Smith     | 7500.50 |
| 1003           | Alice Johnson  | 3000.25 |

```
```sql
```

```
CREATE TABLE Bank (  
    account_number NUMBER PRIMARY KEY,  
    account_holder VARCHAR2(100),  
    balance NUMBER(10, 2)  
);
```

Here's a PL/SQL code snippet based on this data:

PL/SQL Code Snippet:

```
```plsql
```

```
DECLARE  
    v_balance NUMBER;  
BEGIN  
    SELECT balance  
    INTO v_balance  
    FROM Bank  
    WHERE account_holder = 'John Doe';  
    DBMS_OUTPUT.PUT_LINE('John Doe\'s Balance: $' || v_balance);  
END;  
```
```

**Based on above details gives the following question's answer**

1. What is the primary purpose of the PL/SQL code snippet?

- a) Updates John Doe's account balance
- b) Deletes John Doe's account record
- c) Retrieves and displays John Doe's account balance
- d) Inserts a new account record for John Doe

Answer: c) Retrieves and displays John Doe's account balance

2. What is the data type of the "balance" column in the "Bank" table?

- a) String
- b) Date
- c) Number
- d) Boolean

Answer: c) Number

3. Which SQL operation is performed in the PL/SQL code?

- a) INSERT
- b) DELETE
- c) SELECT
- d) UPDATE

Answer: c) SELECT

4. What is the purpose of the `INTO` clause in the code?

- a) To indicate the end of the PL/SQL block
- b) To declare a new variable
- c) To specify the source of data for the SELECT statement
- d) To define a cursor

Answer: c) To specify the source of data for the SELECT statement

5. What does the `DBMS\_OUTPUT.PUT\_LINE` statement do in the code?

- a) Updates the database records
- b) Deletes database records
- c) Retrieves data from the database
- d) Displays a message in the console

Answer: d) Displays a message in the console

here are 10 multiple-choice questions (MCQs) based on a PL/SQL code snippet

PL/SQL Code Snippet:

```
```sql
DECLARE
    v_employee_count NUMBER;
BEGIN
    SELECT COUNT() INTO v_employee_count
    FROM Employees;
    DBMS_OUTPUT.PUT_LINE('Total Employees: ' || v_employee_count);
END;
```
```

**MCQs:**

1. What is the primary purpose of the PL/SQL code snippet?

- a) Updates employee records
- b) Deletes employee records
- c) Retrieves and displays the total number of employees
- d) Inserts a new employee record

Answer: c) Retrieves and displays the total number of employees

2. In the code snippet, what is the value stored in the `v\_employee\_count` variable?

- a) Employee names
- b) Employee IDs
- c) Total number of employees
- d) Employee salaries

Answer: c) Total number of employees

3. Which SQL operation is performed in the PL/SQL code?

- a) INSERT
- b) DELETE
- c) SELECT
- d) UPDATE

Answer: c) SELECT

4. What is the purpose of the `INTO` clause in the code?

- a) To indicate the end of the PL/SQL block
- b) To declare a new variable
- c) To specify the source of data for the SELECT statement
- d) To define a cursor

Answer: c) To specify the source of data for the SELECT statement

5. What does the `DBMS\_OUTPUT.PUT\_LINE` statement do in the code?

- a) Updates the database records
- b) Deletes database records
- c) Retrieves data from the database
- d) Displays a message in the console

Answer: d) Displays a message in the console

6. Which PL/SQL construct allows you to handle exceptions in a structured manner?

- a) TRY-CATCH
- b) EXCEPTION
- c) ERROR-HANDLER
- d) ON-ERROR

Answer: b) EXCEPTION

7. In PL/SQL, what is the primary purpose of a cursor?

- a) To define variables
- b) To loop through a result set

- c) To declare procedures
- d) To manage transactions

Answer: b) To loop through a result set

8. What is the expected output of the code snippet if there are 100 employees in the "Employees" table?

- a) Total Employees: 100
- b) Total Employees: 0
- c) Total Employees: 1
- d) Total Employees: 99

Answer: a) Total Employees: 100

9. What type of variable is `v\_employee\_count` in the code snippet?

- a) String
- b) Date
- c) Number
- d) Boolean

Answer: c) Number

10. In PL/SQL, how can you pass a parameter to a stored procedure?

- a) Using a RETURN statement
- b) Using a SELECT statement
- c) Using an IN parameter
- d) Using a WHERE clause

Answer: c) Using an IN parameter

**Here's a PL/SQL package with a "College" table and some basic code snippets :**

```
```sql
```

```
-- Create the College table
```

```
CREATE TABLE College (  
    student_id NUMBER PRIMARY KEY,  
    student_name VARCHAR2(50),  
    major VARCHAR2(50)  
);
```

```
+-----+-----+-----+  
| student_id | student_name | major |  
+-----+-----+-----+  
| 1 | John Smith | Computer Science |  
| 2 | Jane Doe | Biology |  
| 3 | Alice Johnson | History |  
| 4 | Bob Brown | Mathematics |  
| 5 | Eva Williams | Chemistry |  
+-----+-----+-----+
```

```
```plsql
```

```
-- Create a PL/SQL package
```

```
CREATE OR REPLACE PACKAGE College_Package AS
```

```
    -- Function to retrieve student count by major
```

```
    FUNCTION getStudentCountByMajor(major IN VARCHAR2) RETURN NUMBER;
```

```
    FUNCTION mcq1 RETURN VARCHAR2;
```

```
    FUNCTION mcq2 RETURN NUMBER;
```

```
END College_Package;
```

```
/
```

```
CREATE OR REPLACE PACKAGE BODY College_Package AS
```

```

-- Function to retrieve student count by major
FUNCTION getStudentCountByMajor(major IN VARCHAR2) RETURN NUMBER IS
    cnt NUMBER;
BEGIN
    SELECT COUNT() INTO cnt FROM College WHERE major = major;
    RETURN cnt;
END;

FUNCTION mcq1 RETURN VARCHAR2 IS
BEGIN
    RETURN 'student_id';
END;

FUNCTION mcq2 RETURN NUMBER IS

    biology_count NUMBER;

BEGIN

    biology_count := getStudentCountByMajor('Biology');

    RETURN biology_count;

END;

END College_Package;

/

'''

```

MCQ 1: Which column is used to uniquely identify students?

- A) student\_id
- B) student\_name
- C) major
- D) None of the above

Answer: A) student\_id

MCQ 2: How many students are majoring in Computer Science?

- A) 1
- B) 2



C) 3

D) 0

Answer: A) 1

MCQ 3: What is the data type of the "student\_name" column in the College table?

A) NUMBER

B) VARCHAR2

C) DATE

D) BOOLEAN

Answer: B) VARCHAR2

MCQ 4: Which PL/SQL construct is used to loop through records in a result set?

A) FOR loop

B) IF statement

C) WHILE loop

D) CASE statement

Answer: A) FOR loop

MCQ 5: How many students are majoring in Chemistry?

A) 1

B) 2

C) 3

D) 0

Answer: D) 0

MCQ 6: Which PL/SQL keyword is used to declare a variable?

A) DEFINE

B) DECLARE

C) VARIABLE

D) SET

Answer: B) DECLARE

MCQ 7: What is the output of the following PL/SQL code?

```
``psql
DECLARE
    total_students NUMBER;
BEGIN
    total_students := College_Package.getStudentCountByMajor('Computer Science');
    DBMS_OUTPUT.PUT_LINE('Total students in Computer Science: ' || total_students);
END;
``
```

- A) Total students in Computer Science: 1
- B) Total students in Computer Science: 2
- C) Total students in Computer Science: 3
- D) Total students in Computer Science: 0

Answer: A) Total students in Computer Science: 1

MCQ 8: Which PL/SQL statement is used to raise an exception?

- A) RAISE
- B) THROW
- C) EXCEPTION
- D) ERROR

Answer: A) RAISE

MCQ 9: What is the purpose of the PRIMARY KEY constraint in the College table?

- A) It enforces unique values in the "student\_name" column.
- B) It enforces unique values in the "major" column.
- C) It ensures that the "student\_id" column is not null.
- D) It uniquely identifies each row in the table.

Answer: D) It uniquely identifies each row in the table.

MCQ 10: Which PL/SQL construct is used to handle exceptions in a controlled manner?

A) TRY...CATCH block

B) EXCEPTION block

C) ERROR block

D) HANDLE block

Answer: B) EXCEPTION block

**Here's a PL/SQL code snippet for a hypothetical "hospital" table, along with 10 multiple-choice questions (MCQs)**

Let's create a PL/SQL trigger for the "hospital" table. This trigger updates the "patient\_count" column in a separate "hospital\_stats" table whenever a new patient is inserted into the "hospital" table.

```
```sql
```

```
-- Create the hospital_stats table to store statistics.
```

```
CREATE TABLE hospital_stats (  
    total_patients NUMBER  
);
```

```
-- Create a sequence to generate unique IDs for each patient.
```

```
CREATE SEQUENCE patient_id_seq START WITH 1;
```

```
-- Create the hospital table.
```

```
CREATE TABLE hospital (  
    patient_id NUMBER PRIMARY KEY,  
    patient_name VARCHAR2(50),  
    admission_date DATE,  
    discharge_date DATE  
);
```

```
-- Create the trigger to update patient count in hospital_stats.
```

```
CREATE OR REPLACE TRIGGER update_patient_count
AFTER INSERT ON hospital
FOR EACH ROW
BEGIN
    UPDATE hospital_stats
    SET total_patients = total_patients + 1;
END;
/
'''
```

### Multiple-Choice Questions (MCQs):

1. What is the purpose of the "update\_patient\_count" trigger in the "hospital" table?

- a) To automatically update all patient records.
- b) To update the total count of patients in the "hospital\_stats" table when a new patient is inserted.
- c) To prevent new records from being inserted.
- d) To calculate the average length of stay for all patients.

Correct Answer: b

2. In which event(s) will the "update\_patient\_count" trigger execute?

- a) Before inserting a new patient record.
- b) After deleting a patient record.
- c) Before updating an existing patient record.
- d) After inserting a new patient record.

Correct Answer: d

3. What does 'AFTER INSERT ON hospital' mean in the trigger definition?

- a) The trigger fires before a new patient record is inserted.
- b) The trigger fires after a patient record is deleted.

- c) The trigger fires after a new patient record is inserted.
- d) The trigger fires before an existing patient record is updated.

Correct Answer: c

4. What is the purpose of the "hospital\_stats" table in the code snippet?

- a) To store patient names.
- b) To store admission and discharge dates.
- c) To store statistics related to the hospital, such as the total number of patients.
- d) To store the patient IDs.

Correct Answer: c

5. How is the "patient\_id" assigned in the "hospital" table?

- a) Manually entered by the user.
- b) Generated automatically using a sequence.
- c) Copied from the "patient\_id" in the "hospital\_stats" table.
- d) Set to a constant value.

Correct Answer: b

6. What happens if you attempt to insert a new patient record without specifying values for "patient\_name," "admission\_date," and "discharge\_date"?

- a) The trigger inserts default values.
- b) The trigger raises an error.
- c) The trigger inserts NULL values.
- d) The trigger generates random values.

Correct Answer: b

7. Which keyword is used to specify the trigger action timing in PL/SQL?

- a) WHEN
- b) BEFORE
- c) AFTER
- d) TRIGGER

Correct Answer: c

8. What is the primary purpose of the ``UPDATE hospital_stats SET total_patients = total_patients + 1;`` statement in the trigger?

- a) To delete a patient record.
- b) To insert a new patient record.
- c) To update the "patient\_count" column in the "hospital\_stats" table.
- d) To calculate the average length of stay for all patients.

Correct Answer: c

9. Can you have multiple triggers with the same timing (e.g., AFTER INSERT) on the same table?

- a) No, only one trigger is allowed per table.
- b) Yes, but they must have different names.
- c) Yes, and they execute in a random order.
- d) No, it will result in an error.

Correct Answer: b

10. What does the ``CREATE SEQUENCE patient_id_seq START WITH 1;`` statement do in the code snippet?

- a) It creates a new table.
- b) It defines a new trigger.
- c) It creates a sequence for generating unique patient IDs.
- d) It initializes the patient ID to 1.

Correct Answer: c

A PL/SQL procedure that takes two numbers as input parameters, adds them together, and then displays the result using dbms\_output:

```
```sql
CREATE OR REPLACE PROCEDURE add_numbers (
    p_num1 IN NUMBER,
    p_num2 IN NUMBER
) AS
    v_result NUMBER;
BEGIN
    -- Perform the addition
    v_result := p_num1 + p_num2;

    -- Display the result
    DBMS_OUTPUT.PUT_LINE('The sum of ' || p_num1 || ' and ' || p_num2 || ' is ' || v_result);
END add_numbers;
/
```
```

Here's an example of how to call this procedure:

```
```sql
DECLARE
    num1 NUMBER := 10;
```

```
    num2 NUMBER := 20;
BEGIN
    add_numbers(num1, num2);
END;
/
'''
```

This will call the `add\_numbers` procedure with `num1` and `num2` as arguments and display the sum.

**Based on given pl sql answer the following mcq:**

1. What is the purpose of the PL/SQL procedure mentioned in the code snippet?

- A. To subtract two numbers.
- B. To add two numbers and display the result.
- C. To multiply two numbers.
- D. To divide two numbers.

Answer: B

2. How many input parameters does the `add\_numbers` procedure have?

- A. None
- B. One
- C. Two
- D. Three

Answer: C

3. What data type are the input parameters `p\_num1` and `p\_num2` in the `add\_numbers` procedure?

- A. VARCHAR2
- B. DATE
- C. NUMBER



D. BOOLEAN

Answer: C

4. What is the purpose of the `DBMS\_OUTPUT.PUT\_LINE` statement in the procedure?

- A. It calculates the sum of two numbers.
- B. It displays the result of the addition.
- C. It defines a new variable.
- D. It retrieves data from the database.

Answer: B

5. How is the result of the addition operation displayed in the output?

- A. Using the PRINT statement
- B. Using the RETURN statement
- C. Using the DBMS\_OUTPUT.PUT\_LINE statement
- D. Using the DISPLAY statement

Answer: C

6. What should you do to call the `add\_numbers` procedure with specific numbers as arguments?

- A. Use the CALL statement.
- B. Use the SELECT statement.
- C. Use the DECLARE block.
- D. Use the EXECUTE statement.

Answer: C

7. In the example provided for calling the procedure, what are the values of `num1` and `num2`?

- A. num1 = 20, num2 = 10
- B. num1 = 10, num2 = 30
- C. num1 = 10, num2 = 20
- D. num1 = 30, num2 = 10

Answer: C

8. What is the result of calling the `add\_numbers` procedure with `num1` and `num2` as arguments in the example?

- A. 10
- B. 20
- C. The sum of 10 and 20 is 30
- D. There will be no output.

Answer: C

9. Which SQL statement is used to create a PL/SQL procedure?

- A. CREATE PROCEDURE
- B. DECLARE PROCEDURE
- C. EXECUTE PROCEDURE
- D. CALL PROCEDURE

Answer: A

10. What is the purpose of the `DECLARE` block in the example?

- A. To define a new variable.
- B. To execute SQL statements.
- C. To declare and initialize variables before calling the procedure.
- D. To declare a function.

Answer: C

Ques1 - Consider the following PL/SQL function: (Difficulty level – Easy)

```
CREATE OR REPLACE FUNCTION calculate_total(price NUMBER,  
quantity NUMBER)  
RETURN NUMBER IS  
    total NUMBER;  
BEGIN  
    total := price * quantity;  
    RETURN total;  
END;
```

What does the PL/SQL function `calculate\_total` do?

- A. It calculates the average of `price` and `quantity`.
- B. It calculates the sum of `price` and `quantity`.
- C. It calculates the product of `price` and `quantity`.
- D. It calculates the difference between `price` and `quantity`.

Correct Option: C

---

Ques2 - Consider the following PL/SQL function: (Difficulty level – Easy)

```
CREATE OR REPLACE FUNCTION greet(name VARCHAR2)
RETURN VARCHAR2 IS
    greeting VARCHAR2(100);
BEGIN
    greeting := 'Hello, ' || name || '!';
    RETURN greeting;
END;
```

**\*\*What does the PL/SQL function `greet` do?\*\***

- A. It calculates the length of the input string `name`.
- B. It calculates the square of a numeric input.
- C. It generates a greeting message with the input `name`.
- D. It calculates the factorial of a numeric input.

**\*\*Correct Option:\*\* C**

---

Ques3 - Consider the following PL/SQL function: (Difficulty level – Easy)

``plsqli

```
CREATE OR REPLACE FUNCTION is_even(num NUMBER)
RETURN BOOLEAN IS
BEGIN
    IF MOD(num, 2) = 0 THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
```

```
END;
```

```
---
```

**\*\*What does the PL/SQL function `is\_even` do?\*\***

- A. It checks if the input `num` is an even number and returns `TRUE` if it is, `FALSE` otherwise.
- B. It checks if the input `num` is a positive number and returns `TRUE` if it is, `FALSE` otherwise.
- C. It checks if the input `num` is a prime number and returns `TRUE` if it is, `FALSE` otherwise.
- D. It calculates the factorial of the input `num`.

**\*\*Correct Option:\*\* A**

```
---
```

Ques4 - Consider the following PL/SQL function: (Difficulty level – Easy)

```
```sql
```

```
CREATE OR REPLACE FUNCTION get_employee_salary(emp_id NUMBER)
RETURN NUMBER IS
    salary NUMBER;
BEGIN
    -- Retrieve the salary of the employee with the given
    emp_id
    SELECT salary INTO salary FROM employees WHERE employee_id
    = emp_id;
    RETURN salary;
END;
```

```
```
```

What does the PL/SQL function `get\_employee\_salary` do?

- A. It calculates the average salary of all employees.
- B. It retrieves the salary of the employee with the specified `emp\_id`.
- C. It calculates the total salary of all employees.
- D. It retrieves the highest salary among all employees.

**\*\*Correct Option:\*\* B**

```
---
```

Ques5 - Consider the following PL/SQL function: (Difficulty level – Easy)

```
CREATE OR REPLACE FUNCTION convert_to_uppercase(text
VARCHAR2)
RETURN VARCHAR2 IS
    upper_text VARCHAR2(100);
BEGIN
    upper_text := UPPER(text);
    RETURN upper_text;
END;
```

What does the PL/SQL function `convert\_to\_uppercase` do?

- A. It calculates the length of the input `text`.
- B. It calculates the square of a numeric input.
- C. It converts the input `text` to uppercase.
- D. It calculates the factorial of a numeric input.

Correct Option: C

**\*\*Ques6 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION calculate_tax(income NUMBER)
RETURN NUMBER IS
    tax NUMBER;
BEGIN
    IF income <= 50000 THEN
        tax := income * 0.1;
    ELSE
        tax := 50000 * 0.1 + (income - 50000) * 0.2;
    END IF;
    RETURN tax;
END;
```

```

**\*\*What does the PL/SQL function `calculate\_tax` do?\*\***

- A. It calculates the total income after applying a tax rate.
- B. It calculates the square root of the input number `income`.
- C. It calculates the factorial of the input number `income`.
- D. It calculates the tax amount based on the input income.

**\*\*Correct Option:\*\* D**

---

**\*\*Ques7 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

``psql

```
CREATE OR REPLACE FUNCTION reverse_and_uppercase(input_str
VARCHAR2)
RETURN VARCHAR2 IS
    reversed_upper VARCHAR2(255);
BEGIN
    reversed_upper := UPPER(REVERSE(input_str));
    RETURN reversed_upper;
END;
```

``

**\*\*What does the PL/SQL function `reverse\_and\_uppercase` do?\*\***

- A. It calculates the length of the input string `input\_str`.
- B. It calculates the square of the input number `input\_str`.
- C. It reverses the characters in the input string `input\_str` and converts them to uppercase.
- D. It calculates the factorial of the input number `input\_str`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques8 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

``psql

```
CREATE OR REPLACE FUNCTION find_largest
(numbers VARCHAR2)
RETURN NUMBER IS
    largest NUMBER := NULL;
    num_list VARCHAR2(255);
    num_str VARCHAR2(10);
BEGIN
    num_list := TRIM(BOTH ',' FROM numbers);
    LOOP
        EXIT WHEN LENGTH(num_list) = 0;
        num_str := TRIM(SUBSTR(num_list, 1, INSTR(num_list,
',') - 1));
        num_list := SUBSTR(num_list, INSTR(num_list, ',') + 1);
        IF TO_NUMBER(num_str) > largest OR largest IS NULL THEN
```

```

        largest := TO_NUMBER(num_str);
    END IF;
END LOOP;
RETURN largest;
END;

```

**\*\*What does the PL/SQL function `find\_largest` do?\*\***

- A. It calculates the square root of the input string `numbers`.
- B. It calculates the sum of all numbers in the input string `numbers`.
- C. It retrieves the largest number from a comma-separated list of numbers in the input string `numbers`.
- D. It calculates the factorial of all numbers in the input string `numbers`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques9 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```plsqli

```

CREATE OR REPLACE FUNCTION generate_invoice(total_amount
NUMBER, customer_id NUMBER)
RETURN VARCHAR2 IS
    invoice_text VARCHAR2(500);
    customer_name VARCHAR2(255);
BEGIN
    -- Retrieve the customer's name based on the customer_id
    SELECT name INTO customer_name FROM customers WHERE
customer_id = customer_id;
    invoice_text := 'Invoice for ' || customer_name || ':
Total Amount - $' || total_amount;
    RETURN invoice_text;
END;

```

```

**\*\*What does the PL/SQL function `generate\_invoice` do?\*\***

- A. It generates an invoice text for a customer with the specified `customer\_id` and total amount.
- B. It calculates the average total amount for all customers.
- C. It retrieves the customer's name based on the customer\_id.
- D. It calculates the total amount for a customer with the specified `customer\_id`.

**\*\*Correct Option:\*\* A**

**\*\*Ques10 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE product_recommendations AS
    FUNCTION recommend_products(customer_id NUMBER) RETURN
    VARCHAR2;
    FUNCTION get_product_rating(product_id NUMBER) RETURN
    NUMBER;
    FUNCTION get_product_reviews(product_id NUMBER) RETURN
    NUMBER;
END product_recommendations;
```

/

**\*\*What does the PL/SQL package `product\_recommendations` contain?\*\***

- A. It contains three PL/SQL functions, `recommend\_products`, `get\_product\_rating`, and `get\_product\_reviews`, for providing product recommendations and retrieving product ratings and reviews.
- B. It contains two PL/SQL triggers, `recommend\_products`, `get\_product\_rating`, and `get\_product\_reviews`, for providing product recommendations and retrieving product ratings and reviews.
- C. It contains four PL/SQL procedures, `recommend\_products`, `get\_product\_rating`, and `get\_product\_reviews`, for providing product recommendations and retrieving product ratings and reviews.
- D. It contains one PL/SQL function, `product\_recommendations`, and one PL/SQL procedure, `product\_recommendations`, for providing product recommendations and retrieving product ratings and reviews.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques11 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE inventory_management AS
    FUNCTION check_stock_availability(product_id NUMBER,
    warehouse_id NUMBER) RETURN BOOLEAN;
    FUNCTION transfer_product(product_id NUMBER,
    source_warehouse_id NUMBER, destination_warehouse_id NUMBER,
    quantity NUMBER) RETURN NUMBER;
    FUNCTION get_product_location(product_id NUMBER) RETURN
    VARCHAR2;
END inventory_management;
```



/

...

**\*\*What does the PL/SQL package `inventory\_management` contain?\*\***

A. It contains four PL/SQL functions, `check\_stock\_availability`, `transfer\_product`, and `get\_product\_location`, for managing inventory and retrieving product locations.

B. It contains three PL/SQL triggers, `check\_stock\_availability`, `transfer\_product`, and `get\_product\_location`, for managing inventory and retrieving product locations.

C. It contains two PL/SQL

procedures, `check\_stock\_availability`, `transfer\_product`, and `get\_product\_location`, for managing inventory and retrieving product locations.

D. It contains one PL/SQL function, `inventory\_management`, and one PL/SQL procedure, `inventory\_management`, for managing inventory and retrieving product locations.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques12 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE customer_order_history AS
    FUNCTION get_order_count(customer_id NUMBER) RETURN
    NUMBER;
    FUNCTION get_average_order_value(customer_id NUMBER)
    RETURN NUMBER;
    FUNCTION get_last_order_date(customer_id NUMBER) RETURN
    DATE;
END customer_order_history;
```

/

...

**\*\*What does the PL/SQL package `customer\_order\_history` contain?\*\***

A. It contains three PL/SQL functions, `get\_order\_count`, `get\_average\_order\_value`, and `get\_last\_order\_date`, for retrieving customer order history statistics.

B. It contains three PL/SQL triggers, `get\_order\_count`, `get\_average\_order\_value`, and `get\_last\_order\_date`, for retrieving customer order history statistics.

C. It contains three PL/SQL procedures, `get\_order\_count`, `get\_average\_order\_value`, and `get\_last\_order\_date`, for retrieving customer order history statistics.

D. It contains one PL/SQL function, `customer\_order\_history`, and one PL/SQL procedure, `customer\_order\_history`, for retrieving customer order history statistics.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques13 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE employee_performance AS
    FUNCTION calculate_performance_rating(employee_id NUMBER,
    year NUMBER) RETURN NUMBER;
    FUNCTION get_top_performing_employee(year NUMBER) RETURN
    VARCHAR2;
END employee_performance;
```

/

```

**\*\*What does the PL/SQL package `employee\_performance` contain?\*\***

A. It contains two PL/SQL functions, `calculate\_performance\_rating` and `get\_top\_performing\_employee`, for calculating employee performance ratings and identifying the top-performing employee.

B. It contains one PL/SQL triggers, `calculate\_performance\_rating` and `get\_top\_performing\_employee`, for calculating employee performance ratings and identifying the top-performing employee.

C. It contains three PL/SQL procedures, `calculate\_performance\_rating` and `get\_top\_performing\_employee`, for calculating employee performance ratings and identifying the top-performing employee.

D. It contains four PL /SQL function, `employee\_performance`, and one PL/SQL procedure, `employee\_performance`, for calculating employee performance ratings and identifying the top-performing employee.

**\*\*Correct Option:\*\* A**

**\*\*Ques14 - Consider the following PL/SQL package specification: (Difficulty level – Easy)\*\***

```sql

```
CREATE OR REPLACE PACKAGE employee_info AS
    FUNCTION get_employee_name(emp_id NUMBER) RETURN VARCHAR2;
    FUNCTION get_employee_salary(emp_id NUMBER) RETURN NUMBER;
END employee_info;
```

/

```

**\*\*What does the PL/SQL package `employee\_info` contain?\*\***

A. It contains four PL/SQL functions, `get\_employee\_name` and `get\_employee\_salary`.

- B. It contains PL/SQL triggers, `get\_employee\_name` and `get\_employee\_salary`.
- C. It contains two PL/SQL procedures, `get\_employee\_name` and `get\_employee\_salary`.
- D. It contains one PL/SQL function, `employee\_info`, and one PL/SQL procedure, `employee\_info`.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques15 - Consider the following PL/SQL package specification: (Difficulty level – Easy)\*\***

```sql

```
CREATE OR REPLACE PACKAGE math_operations AS
    FUNCTION add_numbers(num1 NUMBER, num2 NUMBER) RETURN
NUMBER;
    FUNCTION subtract_numbers(num1 NUMBER, num2 NUMBER) RETURN
NUMBER;
    END math_operations;
/
```

```

**\*\*What does the PL/SQL package `math\_operations` contain?\*\***

- A. It contains two PL/SQL functions, `add\_numbers` and `subtract\_numbers`, for performing mathematical operations.
- B. It contains two PL/SQL triggers, `add\_numbers` and `subtract\_numbers`, for performing mathematical operations.
- C. It contains two PL/SQL procedures, `add\_numbers` and `subtract\_numbers`, for performing mathematical operations.
- D. It contains one PL/SQL function, `math\_operations`, and one PL/SQL procedure, `math\_operations`, for performing mathematical operations.

**\*\*Correct Option:\*\* A**

**\*\*Ques1 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```sql

```
CREATE OR REPLACE FUNCTION calculate_area(length NUMBER,
width NUMBER)
RETURN NUMBER IS
    area NUMBER;
```

```
BEGIN
    area := length * width;
    RETURN area;
END;
```

```

**\*\*What does the PL/SQL function `calculate\_area` do?\*\***

- A. It calculates the perimeter of a rectangle.
- B. It calculates the area of a rectangle.
- C. It calculates the volume of a rectangle.
- D. It calculates the diagonal length of a rectangle.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques7 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION get_grade(score NUMBER)
RETURN VARCHAR2 IS
    grade VARCHAR2(2);
BEGIN
    IF score >= 90 THEN
        grade := 'A';
    ELSIF score >= 80 THEN
        grade := 'B';
    ELSIF score >= 70 THEN
        grade := 'C';
    ELSE
        grade := 'D';
    END IF;
    RETURN grade;
END;
```

```

**\*\*What does the PL/SQL function `get\_grade` do?\*\***

- A. It calculates the square root of the input `score`.
- B. It calculates the average of multiple scores.
- C. It assigns a grade ('A', 'B', 'C', or 'D') based on the input `score`.
- D. It calculates the factorial of the input `score`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques2 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION is_positive(num NUMBER)
RETURN BOOLEAN IS
BEGIN
    IF num > 0 THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
```

```

**\*\*What does the PL/SQL function `is\_positive` do?\*\***

- A. It checks if the input `num` is a positive number and returns `TRUE` if it is, `FALSE` otherwise.
- B. It checks if the input `num` is an even number and returns `TRUE` if it is, `FALSE` otherwise.
- C. It calculates the square of the input `num`.
- D. It calculates the factorial of the input `num`.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques3 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION reverse_string(input_str VARCHAR2)
RETURN VARCHAR2 IS
    reversed_str VARCHAR2(255);
BEGIN
    SELECT REVERSE(input_str) INTO reversed_str FROM DUAL;
    RETURN reversed_str;
END;
```

```

**\*\*What does the PL/SQL function `reverse\_string` do?\*\***

- A. It calculates the length of the input string `input\_str`.
- B. It calculates the square root of the input number `input\_str`.

- C. It reverses the characters in the input string `input\_str`.
- D. It calculates the factorial of the input number `input\_str`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques4 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

``psql

```
CREATE OR REPLACE FUNCTION find_maximum(a NUMBER, b NUMBER)
RETURN NUMBER IS
    max_val NUMBER;
BEGIN
    IF a > b THEN
        max_val := a;
    ELSE
        max_val := b;
    END IF;
    RETURN max_val;
END;
```

``

**\*\*What does the PL/SQL function `find\_maximum` do?\*\***

- A. It calculates the average of two numbers.
- B. It calculates the sum of two numbers.
- C. It calculates the maximum value between two numbers.
- D. It calculates the factorial of two numbers.

**\*\*Correct Option:\*\* C**

**\*\*Ques5 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

``psql

```
CREATE OR REPLACE FUNCTION calculate_discount(amount NUMBER)
RETURN NUMBER IS
    discount NUMBER;
BEGIN
    IF amount >= 1000 THEN
        discount := 0.1 * amount;
    ELSE
        discount := 0;
    END IF;
END;
```

```
END IF;  
RETURN discount;  
END;
```

---

**\*\*What does the PL/SQL function `calculate\_discount` do?\*\***

- A. It calculates the total cost after applying a discount of 10%.
- B. It calculates the total cost without any discount.
- C. It calculates the total cost after applying a discount of 1%.
- D. It calculates the total cost after applying a discount of 5%.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques6 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION is_vowel(character CHAR)  
RETURN BOOLEAN IS  
BEGIN  
    IF character IN ('A', 'E', 'I', 'O', 'U', 'a', 'e', 'i',  
    'o', 'u') THEN  
        RETURN TRUE;  
    ELSE  
        RETURN FALSE;  
    END IF;  
END;
```

```

**\*\*What does the PL/SQL function `is\_vowel` do?\*\***

- A. It checks if the input character is a consonant and returns `TRUE` if it is, `FALSE` otherwise.
- B. It checks if the input character is a digit and returns `TRUE` if it is, `FALSE` otherwise.
- C. It checks if the input character is a vowel and returns `TRUE` if it is, `FALSE` otherwise.
- D. It calculates the square root of the input character.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques7 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```sql

```
CREATE OR REPLACE FUNCTION find_length(input_str VARCHAR2)
RETURN NUMBER IS
    length NUMBER;
BEGIN
    SELECT LENGTH(input_str) INTO length FROM DUAL;
    RETURN length;
END;
```

```

**\*\*What does the PL/SQL function `find\_length` do?\*\***

- A. It calculates the factorial of the length of the input string `input\_str`.
- B. It calculates the square root of the length of the input string `input\_str`.
- C. It retrieves the length of the input string `input\_str`.
- D. It checks if the length of the input string `input\_str` is even and returns `TRUE` if it is, `FALSE` otherwise.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques8 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```sql

```
CREATE OR REPLACE FUNCTION calculate_average(num1 NUMBER,
num2 NUMBER)
RETURN NUMBER IS
    average NUMBER;
BEGIN
    average := (num1 + num2) / 2;
    RETURN average;
END;
```

```

**\*\*What does the PL/SQL function `calculate\_average` do?\*\***

- A. It calculates the sum of two numbers.
- B. It calculates the product of two numbers.
- C. It calculates the average of two numbers.
- D. It calculates the square root of two numbers.

**\*\*Correct Option:\*\* C**



---

**\*\*Ques9 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION is_positive_or_zero(num NUMBER)
RETURN BOOLEAN IS
BEGIN
    IF num >= 0 THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
```

```

**\*\*What does the PL/SQL function `is\_positive\_or\_zero` do?\*\***

- A. It checks if the input `num` is a positive number and returns `TRUE` if it is, `FALSE` otherwise.
- B. It checks if the input `num` is an even number and returns `TRUE` if it is, `FALSE` otherwise.
- C. It checks if the input `num` is a non-negative number and returns `TRUE` if it is, `FALSE` otherwise.
- D. It calculates the factorial of the input `num`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques10 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION generate_greeting(name VARCHAR2)
RETURN VARCHAR2 IS
    greeting VARCHAR2(100);
BEGIN
    greeting := 'Hi there, ' || name || '!';
    RETURN greeting;
END;
```

```

**\*\*What does the PL/SQL function `generate\_greeting` do?\*\***

- A. It calculates the length of the input string `name`.
- B. It calculates the square of a numeric input.
- C. It generates a friendly greeting message with the input `name`.

D. It calculates the factorial of a numeric input.

**\*\*Correct Option:\*\* C**

**\*\*Ques11 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

``psql

```
CREATE OR REPLACE FUNCTION calculate_factorial(n NUMBER)
RETURN NUMBER IS
    result NUMBER := 1;
BEGIN
    IF n < 0 THEN
        RETURN NULL;
    ELSIF n = 0 THEN
        RETURN 1;
    ELSE
        FOR i IN 1..n LOOP
            result := result * i;
        END LOOP;
    END IF;
    RETURN result;
END;
```

``

**\*\*What does the PL/SQL function `calculate\_factorial` do?\*\***

A. It calculates the factorial of a non-negative integer `n`.

B. It calculates the square root of the input number `n`.

C. It calculates the average of multiple numbers.

D. It calculates the sum of all integers from 1 to `n`.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques12 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

``psql

```
CREATE OR REPLACE FUNCTION calculate_fibonacci(n NUMBER)
RETURN NUMBER IS
    a NUMBER := 0;
    b NUMBER := 1;
    result NUMBER := 0;
```

```

BEGIN
  IF n <= 0 THEN
    RETURN 0;
  ELSIF n = 1 THEN
    RETURN 1;
  ELSE
    FOR i IN 2..n LOOP
      result := a + b;
      a := b;
      b := result;
    END LOOP;
  END IF;
  RETURN result;
END;

```

**\*\*What does the PL/SQL function `calculate\_fibonacci` do?\*\***

- A. It calculates the sum of the first `n` Fibonacci numbers.
- B. It calculates the square root of the input number `n`.
- C. It calculates the factorial of the input number `n`.
- D. It calculates the `n`-th Fibonacci number.

**\*\*Correct Option:\*\* D**

---

**\*\*Ques13 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

**```sql**

```

CREATE OR REPLACE FUNCTION calculate_power(base NUMBER,
exponent NUMBER)
RETURN NUMBER IS
  result NUMBER := 1;
BEGIN
  IF exponent < 0 THEN
    RETURN NULL;
  ELSE
    FOR i IN 1..exponent LOOP
      result := result * base;
    END LOOP;
  END IF;
  RETURN result;
END;

```

**```**

**\*\*What does the PL/SQL function `calculate\_power` do?\*\***

- A. It calculates the product of `base` and `exponent`.
- B. It calculates the square root of `base` raised to the power of `exponent`.
- C. It calculates the factorial of `exponent`.
- D. It calculates `base` raised to the power of `exponent`.

**\*\*Correct Option:\*\* D**

---

**\*\*Ques14 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```sql

```
CREATE OR REPLACE FUNCTION is_palindrome(word VARCHAR2)
RETURN BOOLEAN IS
    reversed_word VARCHAR2(255);
BEGIN
    reversed_word := REVERSE(word);
    IF word = reversed_word THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
```

```

**\*\*What does the PL/SQL function `is\_palindrome` do?\*\***

- A. It checks if the input string `word` is a palindrome (reads the same forwards and backwards) and returns `TRUE` if it is, `FALSE` otherwise.
- B. It calculates the length of the input string `word`.
- C. It calculates the square root of the input number `word`.
- D. It checks if the input string `word` contains any digits and returns `TRUE` if it does, `FALSE` otherwise.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques25 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```sql

```
CREATE OR REPLACE FUNCTION get_employee_salary(emp_id NUMBER)
RETURN NUMBER IS
```

```

    salary NUMBER;
BEGIN
    -- Retrieve the salary of the employee with the given
emp_id
    SELECT salary INTO salary FROM employees WHERE employee_id
= emp_id;
    IF SQL%FOUND THEN
        RETURN salary;
    ELSE
        RETURN NULL;
    END IF;
END;

```

---

**\*\*What does the PL/SQL function `get\_employee\_salary` do?\*\***

- A. It calculates the average salary of all employees.
- B. It retrieves the salary of the employee with the specified `emp\_id`.
- C. It calculates the total salary of all employees.
- D. It retrieves the highest salary among all employees.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques15 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```plsqli

```

CREATE OR REPLACE FUNCTION count_words(sentence VARCHAR2)
RETURN NUMBER IS
    word_count NUMBER := 0;
BEGIN
    FOR i IN 1..LENGTH(sentence) LOOP
        IF SUBSTR(sentence, i, 1) = ' ' THEN
            word_count := word_count + 1;
        END IF;
    END LOOP;
    -- Add one to count the last word
    word_count := word_count + 1;
    RETURN word_count;
END;

```

---

**\*\*What does the PL/SQL function `count\_words` do?\*\***

- A. It calculates the number of characters in the input sentence.

- B. It calculates the number of words in the input sentence.
- C. It calculates the number of vowels in the input sentence.
- D. It calculates the number of digits in the input sentence.

**\*\*Correct Option:\*\* B**

---

2 mark questions –

**\*\*Ques1 - Consider the following PL/SQL package specification: (Difficulty level – Medium)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE product_discounts AS
    FUNCTION calculate_discount(product_id NUMBER, quantity
NUMBER) RETURN NUMBER;
    FUNCTION apply_discount_to_order(order_id NUMBER) RETURN
BOOLEAN;
END product_discounts;
```

/

**\*\*What does the PL/SQL package `product\_discounts` contain?\*\***

- A. It contains two PL/SQL functions, `calculate\_discount` and `apply\_discount\_to\_order`, for calculating and applying product discounts.
- B. It contains two PL/SQL triggers, `calculate\_discount` and `apply\_discount\_to\_order`, for calculating and applying product discounts.
- C. It contains two PL/SQL procedures, `calculate\_discount` and `apply\_discount\_to\_order`, for calculating and applying product discounts.
- D. It contains one PL/SQL function, `product\_discounts`, and one PL/SQL procedure, `product\_discounts`, for calculating and applying product discounts.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques2 - Consider the following PL/SQL package specification: (Difficulty level – Medium)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE order_processing AS
    FUNCTION process_order(order_id NUMBER) RETURN BOOLEAN;
```

```
FUNCTION validate_payment(order_id NUMBER) RETURN BOOLEAN;  
END order_processing;
```

/

**\*\*What does the PL/SQL package `order\_processing` contain?\*\***

- A. It contains two PL/SQL functions, `process\_order` and `validate\_payment`, for processing orders and validating payments.
- B. It contains two PL/SQL triggers, `process\_order` and `validate\_payment`, for processing orders and validating payments.
- C. It contains two PL/SQL procedures, `process\_order` and `validate\_payment`, for processing orders and validating payments.
- D. It contains one PL/SQL function, `order\_processing`, and one PL/SQL procedure, `order\_processing`, for processing orders and validating payments.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques3 - Consider the following PL/SQL package specification: (Difficulty level – Medium)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE employee_management AS  
    FUNCTION hire_employee(name VARCHAR2, salary NUMBER)  
    RETURN NUMBER;  
    FUNCTION terminate_employee(employee_id NUMBER) RETURN  
    BOOLEAN;  
END employee_management;
```

/

**\*\*What does the PL/SQL package `employee\_management` contain?\*\***

- A. It contains two PL/SQL functions, `hire\_employee` and `terminate\_employee`, for hiring and terminating employees.
- B. It contains two PL/SQL triggers, `hire\_employee` and `terminate\_employee`, for hiring and terminating employees.
- C. It contains two PL/SQL procedures, `hire\_employee` and `terminate\_employee`, for hiring and terminating employees.
- D. It contains one PL/SQL function, `employee\_management`, and one PL/SQL procedure, `employee\_management`, for hiring and terminating employees.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques4 - Consider the following PL/SQL package specification: (Difficulty level – Medium)\*\***

```sql

```
CREATE OR REPLACE PACKAGE order_management AS
    FUNCTION create_order(customer_id NUMBER, total_amount
NUMBER) RETURN NUMBER;
    FUNCTION cancel_order(order_id NUMBER) RETURN BOOLEAN;
END order_management;
```

/

```

**\*\*What does the PL/SQL package `order\_management` contain?\*\***

- A. It contains two PL/SQL functions, `create\_order` and `cancel\_order`, for creating and canceling orders.
- B. It contains two PL/SQL triggers, `create\_order` and `cancel\_order`, for creating and canceling orders.
- C. It contains two PL/SQL procedures, `create\_order` and `cancel\_order`, for creating and canceling orders.
- D. It contains one PL/SQL function, `order\_management`, and one PL/SQL procedure, `order\_management`, for creating and canceling orders.

**\*\*Correct Option:\*\* A**

**\*\*Ques5 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE order_tracking AS
    FUNCTION track_order(order_id NUMBER) RETURN VARCHAR2;
    FUNCTION estimate_delivery_time(order_id NUMBER) RETURN
NUMBER;
    FUNCTION get_order_status(order_id NUMBER) RETURN
VARCHAR2;
END order_tracking;
```

/

```

**\*\*What does the PL/SQL package `order\_tracking` contain?\*\***

- A. It contains three PL/SQL functions, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.



B. It contains three PL/SQL triggers, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.

C. It contains three PL/SQL procedures, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.

D. It contains one PL/SQL function, `order\_tracking`, and one PL/SQL procedure, `order\_tracking`, for tracking orders and estimating delivery times.

**\*\*Correct Option:\*\* A**

---

---

**\*\*Ques6 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE project_management AS
    FUNCTION allocate_resources(project_id NUMBER, resource_id
NUMBER, hours NUMBER) RETURN BOOLEAN;
    FUNCTION get_project_status(project_id NUMBER) RETURN
VARCHAR2;
END project_management;
```

/

```

**\*\*What does the PL/SQL package `project\_management` contain?\*\***

A. It contains two PL/SQL functions, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.

B. It contains two PL/SQL triggers, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.

C. It contains two PL/SQL procedures, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.

D. It contains one PL/SQL function, `project\_management`, and one PL/SQL procedure, `project\_management`, for resource allocation and project status retrieval.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques7 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE student_grading AS
    FUNCTION calculate_final_grade(student_id NUMBER,
course_id NUMBER) RETURN CHAR;
    FUNCTION get_student_ranking(course_id NUMBER) RETURN
NUMBER;
END student_grading;
```

/

**\*\*What does the PL/SQL package `student\_grading` contain?\*\***

- A. It contains two PL/SQL functions, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- B. It contains two PL/SQL triggers, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- C. It contains two PL/SQL procedures, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- D. It contains one PL/SQL function, `student\_grading`, and one PL/SQL procedure, `student\_grading`, for calculating student grades and retrieving student rankings in a course.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques8 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE medical_records AS
    FUNCTION get_patient_history(patient_id NUMBER) RETURN
CLOB;
    FUNCTION analyze_patient_data(patient_id NUMBER) RETURN
CLOB;
END medical_records;
/
```

```

**\*\*What does the PL/SQL package `medical\_records` contain?\*\***

- A. It contains two PL/SQL functions, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- B. It contains two PL/SQL triggers, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- C. It contains two PL/SQL procedures, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.

D. It contains one PL/SQL function, `medical\_records`, and one PL/SQL procedure, `medical\_records`, for retrieving patient medical history and analyzing patient data.

**\*\*Correct Option:\*\* A**

**\*\*Ques9 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE order_tracking AS
    FUNCTION track_order(order_id NUMBER) RETURN VARCHAR2;
    FUNCTION estimate_delivery_time(order_id NUMBER) RETURN
NUMBER;
    FUNCTION get_order_status(order_id NUMBER) RETURN
VARCHAR2;
END order_tracking;
```

/

```

**\*\*What does the PL/SQL package `order\_tracking` contain?\*\***

A. It contains three PL/SQL functions, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.

B. It contains three PL/SQL triggers, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.

C. It contains three PL/SQL procedures, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.

D. It contains one PL/SQL function, `order\_tracking`, and one PL/SQL procedure, `order\_tracking`, for tracking orders and estimating delivery times.

**\*\*Correct Option:\*\* A**

---

---

**\*\*Ques10 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE project_management AS
    FUNCTION allocate_resources(project_id NUMBER, resource_id
NUMBER, hours NUMBER) RETURN BOOLEAN;
    FUNCTION get_project_status(project_id NUMBER) RETURN
VARCHAR2;
```

```
END project_management;
```

```
/
```

**\*\*What does the PL/SQL package `project\_management` contain?\*\***

- A. It contains two PL/SQL functions, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- B. It contains two PL/SQL triggers, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- C. It contains two PL/SQL procedures, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- D. It contains one PL/SQL function, `project\_management`, and one PL/SQL procedure, `project\_management`, for resource allocation and project status retrieval.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques11 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

**```plsqli**

```
CREATE OR REPLACE PACKAGE student_grading AS
    FUNCTION calculate_final_grade(student_id NUMBER,
course_id NUMBER) RETURN CHAR;
    FUNCTION get_student_ranking(course_id NUMBER) RETURN
NUMBER;
END student_grading;
```

```
/
```

**\*\*What does the PL/SQL package `student\_grading` contain?\*\***

- A. It contains two PL/SQL functions, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- B. It contains two PL/SQL triggers, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- C. It contains two PL/SQL procedures, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- D. It contains one PL/SQL function, `student\_grading`, and one PL/SQL procedure, `student\_grading`, for calculating student grades and retrieving student rankings in a course.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques12 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE PACKAGE medical_records AS
    FUNCTION get_patient_history(patient_id NUMBER) RETURN
    CLOB;
    FUNCTION analyze_patient_data(patient_id NUMBER) RETURN
    CLOB;
END medical_records;
/
```

```

**\*\*What does the PL/SQL package `medical\_records` contain?\*\***

- A. It contains two PL/SQL functions, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- B. It contains two PL/SQL triggers, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- C. It contains two PL/SQL procedures, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- D. It contains one PL/SQL function, `medical\_records`, and one PL/SQL procedure, `medical\_records`, for retrieving patient medical history and analyzing patient data.

**\*\*Correct Option:\*\* A**

**\*\*Ques13 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
    emp_cursor CURSOR FOR
        SELECT employee_name FROM employees;
```

```

**\*\*What does the SQL cursor `emp\_cursor` do?\*\***

- A. It retrieves all columns from the `employees` table.
- B. It retrieves the `employee\_name` column from the `employees` table.
- C. It updates the `employee\_name` column in the `employees` table.
- D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques14 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
    product_cursor CURSOR FOR
        SELECT product_name, product_price FROM products;
```

```

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_name` and `product\_price` columns from the `products` table.
- C. It updates the `product\_name` and `product\_price` columns in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques15 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
    order_cursor CURSOR FOR
        SELECT order_id, order_date FROM orders;
```

```

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_id` and `order\_date` columns from the `orders` table.
- C. It updates the `order\_id` and `order\_date` columns in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques1 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
```

```
customer_cursor CURSOR FOR
  SELECT customer_name FROM customers;
```

---

**\*\*What does the SQL cursor `customer\_cursor` do?\*\***

- A. It retrieves all columns from the `customers` table.
- B. It retrieves the `customer\_name` column from the `customers` table.
- C. It updates the `customer\_name` column in the `customers` table.
- D. It deletes records from the `customers` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques2 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
  employee_cursor CURSOR FOR
    SELECT employee_id, employee_name FROM employees;
```

---

**\*\*What does the SQL cursor `employee\_cursor` do?\*\***

- A. It retrieves all columns from the `employees` table.
- B. It retrieves the `employee\_id` and `employee\_name` columns from the `employees` table.
- C. It updates the `employee\_id` and `employee\_name` columns in the `employees` table.
- D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques3 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
  product_cursor CURSOR FOR
    SELECT product_id FROM products WHERE product_price >
100;
```

---

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_id` column from the `products` table for products with a price greater than 100.
- C. It updates the `product\_id` column in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques4 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

**```sql**

```
DECLARE
    order_cursor CURSOR FOR
        SELECT order_date FROM orders WHERE order_status =
'Shipped';
```

**```**

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_date` column from the `orders` table for orders with a status of 'Shipped'.
- C. It updates the `order\_date` column in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques5 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

**```sql**

```
DECLARE
    customer_cursor CURSOR FOR
        SELECT customer_id FROM customers WHERE
registration_date >= '2023-01-01';
```

**```**

**\*\*What does the SQL cursor `customer\_cursor` do?\*\***

- A. It retrieves all columns from the `customers` table.



B. It retrieves the `customer\_id` column from the `customers` table for customers registered on or after January 1, 2023.

C. It updates the `customer\_id` column in the `customers` table.

D. It deletes records from the `customers` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques6 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
    employee_cursor CURSOR FOR
        SELECT department_id, COUNT(*) FROM employees GROUP BY
            department_id;
```

```

**\*\*What does the SQL cursor `employee\_cursor` do?\*\***

A. It retrieves all columns from the `employees` table.

B. It retrieves the `department\_id` and the count of employees in each department from the `employees` table.

C. It updates the `department\_id` and employee counts in the `employees` table.

D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques7 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
    product_cursor CURSOR FOR
        SELECT product_name, product_category FROM products
        WHERE product_category = 'Electronics';
```

```

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

A. It retrieves all columns from the `products` table.

B. It retrieves the `product\_name` and `product\_category` columns from the `products` table for products in the 'Electronics' category.

C. It updates the `product\_name` and `product\_category` columns in the `products` table.

D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques8 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
    employee_cursor CURSOR FOR
        SELECT employee_id, employee_name, department_id
        FROM employees
        WHERE salary > (SELECT AVG(salary) FROM employees);
```

```

**\*\*What does the SQL cursor `employee\_cursor` do?\*\***

A. It retrieves all columns from the `employees` table.

B. It retrieves the `employee\_id`, `employee\_name`, and `department\_id` columns from the `employees` table for employees with salaries above the average salary in the company.

C. It updates the `employee\_id`, `employee\_name`, and `department\_id` columns in the `employees` table.

D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques9 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
    order_cursor CURSOR FOR
        SELECT order_id, customer_id, order_date
        FROM orders
        WHERE EXISTS (SELECT 1 FROM order_items WHERE
            order_items.order_id = orders.order_id);
```

```

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_id`, `customer\_id`, and `order\_date` columns from the `orders` table for orders that have associated order items.
- C. It updates the `order\_id`, `customer\_id`, and `order\_date` columns in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques10 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
  customer_cursor CURSOR FOR
    SELECT customer_id, COUNT(*) AS order_count
    FROM orders
    GROUP BY customer_id
    HAVING COUNT(*) > 5;
```

```

**\*\*What does the SQL cursor `customer\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `customer\_id` and the count of orders placed by each customer from the `orders` table for customers who have placed more than 5 orders.
- C. It updates the `customer\_id` and order counts in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques11 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
  product_cursor CURSOR FOR
    SELECT product_id, product_name, product_price
    FROM products
    WHERE product_id IN (SELECT product_id FROM order_items
    GROUP BY product_id HAVING COUNT(*) >= 10);
```

```

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_id`, `product\_name`, and `product\_price` columns from the `products` table for products that have been ordered at least 10 times.
- C. It updates the `product\_id`, `product\_name`, and `product\_price` columns in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques12 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

**```sql**

```
DECLARE
    order_cursor CURSOR FOR
        SELECT order_id, order_date, SUM(order_total) AS
total_amount
        FROM orders
        WHERE order_status = 'Shipped'
        GROUP BY order_id, order_date
        HAVING SUM(order_total) > 1000;
```

**```**

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_id`, `order\_date`, and total order amount columns from the `orders` table for shipped orders with a total amount greater than 1000.
- C. It updates the `order\_id`, `order\_date`, and total order amount columns in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques13 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

**```sql**

```
CREATE OR REPLACE TRIGGER update_salary_trigger
BEFORE UPDATE ON employees
```

```

FOR EACH ROW
BEGIN
    IF :NEW.salary > :OLD.salary THEN
        INSERT INTO salary_history (employee_id, old_salary,
new_salary, change_date)
            VALUES (:OLD.employee_id, :OLD.salary, :NEW.salary,
SYSDATE);
    END IF;
END;

```

/

**\*\*What does the SQL trigger `update\_salary\_trigger` do?\*\***

- A. It updates the salary of all employees in the `employees` table.
- B. It inserts a record into the `salary\_history` table whenever an employee's salary is increased.
- C. It deletes records from the `employees` table whenever an employee's salary is updated.
- D. It calculates the average salary of all employees.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques14 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```sql

```

CREATE OR REPLACE TRIGGER audit_employee_delete
AFTER DELETE ON employees
FOR EACH ROW
BEGIN
    INSERT INTO audit_log (event_type, event_date, username,
details)
        VALUES ('Employee Deletion', SYSDATE, USER, 'Employee ID:
' || :OLD.employee_id);
END;

```

/

**\*\*What does the SQL trigger `audit\_employee\_delete` do?\*\***

- A. It updates employee records in the `employees` table.
- B. It inserts a record into the `audit\_log` table whenever an employee is deleted.
- C. It inserts a record into the `employees` table whenever an employee is deleted.
- D. It calculates the total number of employees in the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques15 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE TRIGGER calculate_avg_salary
AFTER INSERT OR DELETE ON employees
FOR EACH ROW
BEGIN
    DECLARE
        total_salary NUMBER;
        num_employees NUMBER;
    BEGIN
        SELECT SUM(salary), COUNT(*) INTO total_salary,
num_employees FROM employees;
        IF num_employees > 0 THEN
            INSERT INTO salary_stats (average_salary,
total_employees, calculation_date)
VALUES (total_salary / num_employees, num_employees,
SYSDATE);
        END IF;
    END;
END;
```

/
```

**\*\*What does the SQL trigger `calculate\_avg\_salary` do?\*\***

- A. It updates the salary of all employees in the `employees` table.
- B. It calculates the average salary and total number of employees whenever a new employee is inserted or an employee is deleted.
- C. It inserts a record into the `salary\_stats` table whenever an employee is deleted.
- D. It calculates the total number of employees in the `employees` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques1 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE TRIGGER prevent_salary_reduction
BEFORE UPDATE ON employees
FOR EACH ROW
BEGIN
```

```

    IF :NEW.salary < :OLD.salary THEN
        RAISE_APPLICATION_ERROR (-20001, 'Salary reduction is
not allowed.');
```

```
    END IF;
```

```
END;
```

```

/
---
```

**\*\*What does the SQL trigger `prevent\_salary\_reduction` do?\*\***

- A. It updates the salary of all employees in the `employees` table.
- B. It prevents any attempt to reduce an employee's salary and raises a custom application error if such an update is detected.
- C. It inserts a record into the `salary\_history` table whenever an employee's salary is increased.
- D. It calculates the average salary of all employees.

**\*\*Correct Option:\*\* B**

```
---
```

**\*\*Ques2 - Consider the following SQL trigger: (Difficulty level – Medium)\*\***

```
```sql
```

```

CREATE OR REPLACE TRIGGER audit_table_changes
AFTER INSERT OR UPDATE OR DELETE ON employees
DECLARE
    change_description VARCHAR2(500);
BEGIN
    change_description := 'Table "employees" was ';
    IF INSERTING THEN
        change_description := change_description || 'inserted
into.';
    ELSIF UPDATING THEN
        change_description := change_description || 'updated.';
    ELSIF DELETING THEN
        change_description := change_description || 'deleted
from.';
    END IF;
    INSERT INTO audit_log (event_type, event_date, details)
VALUES ('Table Change', SYSDATE, change_description);
END;
/
---
```

**\*\*What does the SQL trigger `audit\_table\_changes` do?\*\***

- A. It updates the `employees` table whenever a change is made to it.

B. It inserts a record into the `audit\_log` table whenever a change (insert, update, or delete) is made to the `employees` table, including a description of the change.

C. It calculates the total number of employees in the `employees` table.

D. It deletes records from the `employees` table whenever a change is made to it.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques3 - Consider the following SQL trigger: (Difficulty level – Easy)\*\***

```sql

```
CREATE OR REPLACE TRIGGER enforce_manager_approval
BEFORE INSERT ON purchase_orders
FOR EACH ROW
BEGIN
    IF :NEW.total_amount > 1000 AND :NEW.manager_approval IS
    NULL THEN
        RAISE_APPLICATION_ERROR (-20002, 'Manager approval is
        required for purchase orders over $1000.');
```

```

**\*\*What does the SQL trigger `enforce\_manager\_approval` do?\*\***

A. It inserts records into the `purchase\_orders` table.

B. It updates records in the `purchase\_orders` table.

C. It prevents the insertion of purchase orders with a total amount over \$1000 if they don't have manager approval, raising a custom application error if such an insert is attempted.

D. It calculates the total amount of all purchase orders.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques4 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE TRIGGER calculate_total_order_amount
AFTER INSERT OR UPDATE ON order_items
```



```

FOR EACH ROW
DECLARE
    total_amount NUMBER;
BEGIN
    total_amount := 0;
    SELECT SUM(quantity * unit_price) INTO total_amount FROM
order_items WHERE order_id = :NEW.order_id;
    UPDATE orders SET total_amount = total_amount WHERE
order_id = :NEW.order_id;
END;
/

```

**\*\*What does the SQL trigger `calculate\_total\_order\_amount` do?\*\***

- A. It inserts records into the `order\_items` table.
- B. It updates records in the `order\_items` table.
- C. It calculates the total order amount for an order whenever a new order item is inserted or an existing order item is updated, and updates the `total\_amount` in the `orders` table.
- D. It calculates the average order amount.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques5 - Consider the following SQL trigger: (Difficulty level –Easy)\*\***

```sql

```

CREATE OR REPLACE TRIGGER prevent_duplicate_records
BEFORE INSERT ON employees
FOR EACH ROW
BEGIN
    IF EXISTS (SELECT 1 FROM employees WHERE employee_id =
:NEW.employee_id) THEN
        RAISE_APPLICATION_ERROR (-20003, 'Employee ID must be
unique. ');
    END IF;
END;
/

```

```

**\*\*What does the SQL trigger `prevent\_duplicate\_records` do?\*\***

- A. It inserts records into the `employees` table.
- B. It updates records in the `employees` table.

C. It prevents the insertion of duplicate employee records with the same `employee\_id`, raising a custom application error if such an insert is attempted.

D. It calculates the total number of employees in the `employees` table.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques6 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE TRIGGER calculate_sales_bonus
AFTER INSERT OR UPDATE ON sales
FOR EACH ROW
BEGIN
    DECLARE
        bonus_amount NUMBER;
    BEGIN
        IF :NEW.sale_amount > 10000 THEN
            bonus_amount := :NEW.sale_amount * 0.05;
            UPDATE sales SET bonus = bonus_amount WHERE sale_id
= :NEW.sale_id;
        END IF;
    END;
END;
/
```

```

**\*\*What does the SQL trigger `calculate\_sales\_bonus` do?\*\***

A. It inserts records into the `sales` table.

B. It updates records in the `sales` table.

C. It calculates a sales bonus for sales with an amount over \$10,000 and updates the `bonus` field in the `sales` table whenever a new sale is inserted or an existing sale is updated.

D. It calculates the average sale amount.

**\*\*Correct Option:\*\* C**

**\*\*Ques7 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION get_last_name(full_name VARCHAR2)
RETURN VARCHAR2 IS
```

```

    last_name VARCHAR2(50);
BEGIN
    last_name := SUBSTR(full_name, INSTR(full_name, ' ')+1);
    RETURN last_name;
END;

```

**\*\*What does the PL/SQL function `get\_last\_name` do?\*\***

- A. It calculates the average length of all words in the input `full\_name`.
- B. It calculates the length of the last word in the input `full\_name`.
- C. It retrieves the last name from the input `full\_name`.
- D. It checks if the input `full\_name` contains any digits and returns `TRUE` if it does, `FALSE` otherwise.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques8 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

**```plsql**

```

CREATE OR REPLACE FUNCTION square_number(num NUMBER)
RETURN NUMBER IS
    square NUMBER;
BEGIN
    square := num * num;
    RETURN square;
END;

```

**```**

**\*\*What does the PL/SQL function `square\_number` do?\*\***

- A. It calculates the square root of the input number `num`.
- B. It calculates the sum of two numbers.
- C. It calculates the square of the input number `num`.
- D. It calculates the factorial of the input number `num`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques9 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

``psql

```
CREATE OR REPLACE FUNCTION is_prime(number NUMBER)
RETURN BOOLEAN IS
BEGIN
    IF number <= 1 THEN
        RETURN FALSE;
    END IF;
    FOR i IN 2..number-1 LOOP
        IF MOD(number, i) = 0 THEN
            RETURN FALSE;
        END IF;
    END LOOP;
    RETURN TRUE;
END;
```

``

**\*\*What does the PL/SQL function `is\_prime` do?\*\***

- A. It checks if the input `number` is a prime number and returns `TRUE` if it is, `FALSE` otherwise.
- B. It calculates the square root of the input `number`.
- C. It calculates the factorial of the input `number`.
- D. It checks if the input `number` is even and returns `TRUE` if it is, `FALSE` otherwise.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques10 - Consider the following PL/**

**SQL function: (Difficulty level – Easy)\*\***

``psql

```
CREATE OR REPLACE FUNCTION get_day_of_week(date_value DATE)
RETURN VARCHAR2 IS
    day_of_week VARCHAR2(15);
BEGIN
    SELECT TO_CHAR(date_value, 'Day') INTO day_of_week FROM
DUAL;
    RETURN day_of_week;
END;
```

``

**\*\*What does the PL/SQL function `get\_day\_of\_week` do?\*\***

- A. It calculates the day of the week for the input `date\_value` and returns it as a string.
- B. It calculates the square root of the input `date\_value`.

- C. It calculates the average of multiple dates.
- D. It retrieves the month of the input `date\_value`.

**\*\*Correct Option:\*\* A**

**\*\*Ques11 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
    product_cursor CURSOR FOR
        SELECT product_id, product_name
        FROM products
        WHERE product_id NOT IN (SELECT DISTINCT product_id
        FROM order_items);
```

```

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_id` and `product\_name` columns from the `products` table for products that have not been ordered.
- C. It updates the `product\_id` and `product\_name` columns in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques12 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
    customer_cursor CURSOR FOR
        SELECT customer_id, MAX(order_date) AS last_order_date
        FROM orders
        GROUP BY customer_id
        HAVING MAX(order_date) < TO_DATE('2023-01-01', 'YYYY-MM-DD');
```

```

**\*\*What does the SQL cursor `customer\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.

B. It retrieves the `customer\_id` and the last order date columns from the `orders` table for customers whose last order date is before January 1, 2023.

C. It updates the `customer\_id` and last order date columns in the `orders` table.

D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques13 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
    employee_cursor CURSOR FOR
        SELECT employee_id, employee_name, department_id
        FROM employees
        WHERE department_id = (SELECT department_id FROM
        departments WHERE department_name = 'Engineering');
```

```

**\*\*What does the SQL cursor `employee\_cursor` do?\*\***

A. It retrieves all columns from the `employees` table.

B. It retrieves the `employee\_id`, `employee\_name`, and `department\_id` columns from the `employees` table for employees in the 'Engineering' department.

C. It updates the `employee\_id`, `employee\_name`, and `department\_id` columns in the `employees` table.

D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques14 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
    product_cursor CURSOR FOR
        SELECT product_id, product_name
        FROM products
        WHERE product_price = (SELECT MAX(product_price) FROM
        products);
```

```

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_id` and `product\_name` columns from the `products` table for products with the highest product price.
- C. It updates the `product\_id` and `product\_name` columns in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques15 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
  order_cursor CURSOR FOR
    SELECT order_id, order_date
    FROM orders
    WHERE order_id = (SELECT MAX(order_id) FROM orders);
```

```

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_id` and `order\_date` columns from the `orders` table for the order with the highest order ID.
- C. It updates the `order\_id` and `order\_date` columns in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

SR	Questions	Option 1	Option 2	Option 3	Option 4	Correct Answer
1	Which of the following is NOT a benefit of using transactions?	Data integrity	High availability	Data consistency	Data durability	B
2	A transaction that violates the consistency property is considered to be:	Serializable	Inconsistent	Isolate	Error	B
3	Can you change the parameter values of a cursor after it has been declared and opened?	Yes, parameter values can be modified at any time.	No, parameter values are fixed once the cursor is declared and opened.	Parameter values can only be changed during cursor declaration.	Cursors cannot have parameter values.	B
4	Can you declare a cursor without specifying the SELECT statement immediately?	No, a SELECT statement must always be specified.	Yes, a SELECT statement can be added later in the code.	Cursors cannot be declared in PL/SQL.	Cursors are automatically generated in PL/SQL.	A
5	Can you declare multiple cursors with the same name but different parameters in the same PL/SQL block?	Yes, as long as the cursor names are unique.	No, cursor names must be unique regardless of the parameters.	Multiple cursors are not allowed in the same PL/SQL block.	Cursors with parameters cannot have the same name.	B
6	Can you declare multiple cursors within the same PL/SQL block? If so, how do you differentiate them?	No, only one cursor is allowed per block.	Yes, multiple cursors can be declared, and they are differentiated by their data types.	Yes, multiple cursors can be declared, and they are differentiated by their names.	Multiple cursors cannot be used in PL/SQL.	C
7	Can you fetch data from a cursor into individual variables or into a record type? Explain.	Data can only be fetched into individual variables.	Data can only be fetched into a record type.	Data can be fetched into both individual variables and a record type.	Data cannot be fetched from a cursor.	C
8	Can you nest a Cursor FOR Loop inside another Cursor FOR Loop? If so, why might you do so?	No, nesting Cursor FOR Loops is not allowed.	Yes, you can nest Cursor FOR Loops to perform complex data processing and handle related data hierarchies.	Cursor FOR Loops can only be used individually, not nested.	Nesting Cursor FOR Loops results in performance issues.	B
9	Can you use a Cursor FOR Loop to update or delete records in a database table? Explain.	No, Cursor FOR Loops are read-only.	Yes, Cursor FOR Loops can update or delete records using the UPDATE and DELETE statements.	Cursor FOR Loops can only insert records, not update or delete them.	Cursor FOR Loops can only be used for reporting purposes.	B
10	Describe the differences between an implicit cursor and an explicit cursor in PL/SQL.	Implicit cursors are used for data modeling, while explicit cursors are used for data manipulation.	Implicit cursors are automatically created for DML statements, while explicit cursors are user-defined.	Implicit cursors are used for database connections, while explicit cursors are used for loop control.	Implicit cursors are used for hardware design, while explicit cursors are used for web development.	B
11	Describe the purpose of PL/SQL collections, and provide examples of their types.	PL/SQL collections are used for defining variables.	PL/SQL collections are used for database connections.	PL/SQL collections are used for storing multiple values of the same data type.	PL/SQL collections are used for creating triggers.	C
12	Explain how cursor parameters can be used to create dynamic cursors.	Cursor parameters have no role in creating dynamic cursors.	By allowing parameterization of the WHERE clause in the cursor's SELECT statement, you can create dynamic cursors that retrieve specific data based on different criteria.	Cursor parameters can only be used with static cursors.	Cursor parameters can be used to create triggers.	B



13	Explain the concept of triggers in a database context. How are they used in PL/SQL?	Triggers are used for creating web applications.	Triggers are used for hardware design.	Triggers are used for automatically executing PL/SQL code in response to database events.	Triggers are used for data modeling.	C
14	Explain the difference between declaring a cursor and opening a cursor.	Declaring a cursor retrieves data; opening a cursor defines its structure.	Declaring a cursor defines its structure; opening a cursor retrieves data.	Declaring a cursor and opening a cursor are the same.	Declaring a cursor is not a PL/SQL concept.	B
15	Explain the importance of transactions in PL/SQL and how they are managed.	Transactions are used for web development.	Transactions are used for data modeling.	Transactions ensure data consistency and are managed using COMMIT and ROLLBACK statements.	Transactions are not supported in PL/SQL.	C
16	Explain the purpose of a PL/SQL package and its components.	PL/SQL packages are used for web development.	PL/SQL packages are used for encapsulating procedures and functions.	PL/SQL packages are used for data modeling.	PL/SQL packages are used for hardware design.	B
17	How can you pass parameters to a PL/SQL procedure or function?	Parameters are passed using the CALL statement.	Parameters are not supported in PL/SQL.	Parameters are passed as input and output variables.	Parameters are passed using the DECLARE statement.	C
18	How can you resolve a deadlock in a database system?	By terminating one of the transactions involved in the deadlock.	By rolling back all transactions involved in the deadlock.	By increasing the isolation level.	Deadlocks cannot be resolved.	A
19	How do you create and manipulate PL/SQL associative arrays (index-by tables)?	Associative arrays are created using the ARRAY keyword.	Associative arrays are created using the INDEX keyword.	Associative arrays are not supported in PL/SQL.	Associative arrays are created using the TYPE keyword.	D
20	How do you declare a cursor, and what are the required components?	Cursors are automatically declared in PL/SQL.	Cursors are declared using the DECLARE CURSOR statement and require a SELECT statement.	Cursors are declared using the DECLARE keyword.	Cursors are declared using the OPEN statement.	B

21	How do you declare a variable in PL/SQL, and what are the data types supported for variables?	Variables are declared using the DECLARE keyword, and PL/SQL supports only one data type.	Variables are declared using the VAR keyword, and PL/SQL supports multiple data types.	Variables are declared using the VARIABLE keyword, and PL/SQL supports multiple data types.	Variables are not supported in PL/SQL.	B
22	How do you define and use PL/SQL records and record types?	Records are used for creating tables in PL/SQL.	Records are defined using the DECLARE RECORD statement.	Records are used to hold data in a structured format.	Records are not supported in PL/SQL.	C
23	How do you ensure that you've fetched all available data from a cursor?	By using the CLOSE statement.	By using the OPEN statement.	By checking the cursor attribute %NOTFOUND.	Cursors automatically fetch all available data.	C
24	How do you handle database connections and transactions in PL/SQL?	Database connections and transactions are automatically managed by the PL/SQL engine.	Database connections and transactions are not supported in PL/SQL.	Database connections are established using the CONNECT statement, and transactions are managed using COMMIT and ROLLBACK statements.	Database connections are established using the DECLARE statement.	C
25	How do you handle exceptions in PL/SQL? Provide an example.	Exceptions are handled using the IF-ELSE statement.	Exceptions are handled using the TRY-CATCH block.	Exceptions are handled using the EXCEPTION block.	Exceptions are not supported in PL/SQL.	C
26	How do you handle exceptions that may occur when working with cursors that have parameters?	By using the FETCH statement.	By ignoring exceptions and proceeding with the cursor operations.	By using exception handling techniques such as WHEN OTHERS and specific exception handlers for cursor-related errors.	Cursors with parameters do not raise exceptions.	C
27	How do you name a cursor, and what are some best practices for naming conventions?	Cursors are named automatically.	Cursors can be named using any random string.	Cursors should have meaningful names following naming conventions such as prefixing with CUR_.	Cursors cannot have names in PL/SQL.	C
28	How do you open a cursor to make it ready for data retrieval?	Use the DECLARE CURSOR statement.	Use the OPEN CURSOR statement.	Use the FETCH statement.	Cursors are automatically opened in PL/SQL.	B
29	How do you pass values to the cursor parameters when opening the cursor?	Use the FETCH statement to provide parameter values.	Use the SET PARAMETER statement.	Use a separate ASSIGN statement to assign values to parameters before opening the cursor.	Cursor parameters do not require values when opening.	C
30	How does a Cursor FOR Loop handle exceptions compared to explicit cursor processing?	Cursor FOR Loops do not support exception handling.	Cursor FOR Loops handle exceptions more gracefully by providing built-in error handling mechanisms.	Exception handling in Cursor FOR Loops is the same as in explicit cursor processing.	Cursor FOR Loops handle exceptions less efficiently than explicit cursors.	B
31	How does a cursor with parameters differ from a cursor without parameters in terms of flexibility?	Cursors with parameters are less flexible.	Cursors with parameters are more flexible because they can retrieve data based on varying conditions.	There is no difference in flexibility between the two types of cursors.	Cursors with parameters are slower.	B

32	How is the declaration of a cursor different from a regular SQL query?	Cursors cannot be used to retrieve data.	Cursors have a SELECT statement, while regular SQL queries are standalone.	Regular SQL queries cannot be used in PL/SQL.	There is no difference; they are the same.	B
33	In a multi-user database system, what does optimistic concurrency control aim to achieve?	It aims to prevent transactions from running concurrently.	It aims to avoid blocking and allow transactions to proceed concurrently, only checking for conflicts at the end.	It aims to lock all records to avoid conflicts.	It aims to roll back all transactions.	B
34	In database recovery, what is the difference between forward recovery and backward recovery?	Forward recovery restores the database to a previous state, while backward recovery recovers the database to its current state.	Forward recovery is the same as database backup, while backward recovery restores the database to a previous state.	Forward recovery involves log analysis, while backward recovery involves restoring database backups.	There is no difference; the terms are used interchangeably.	C
35	In database recovery, what is the purpose of a database log file?	To store user data.	To record changes made to the database for recovery purposes.	To create database backups.	To store database metadata.	B
36	used?	View	Commit	Rollback	Flashback	C
37	In SQL, what is the role of the ROLLBACK statement?	To save pending changes.	To begin a new transaction.	To undo all changes made during the current transaction.	To release locks on database records.	C
38	Is it necessary to declare a cursor inside a PL/SQL block, or can it be declared globally in a package?	Cursors can only be declared globally.	Cursors can only be declared inside a PL/SQL block.	Cursors can be declared both globally and inside a PL/SQL block.	Cursors are not supported in PL/SQL.	C
39	What are database triggers, and when might you use them in PL/SQL?	Database triggers are used for hardware design.	Database triggers are used for declaring variables.	Database triggers are used to automatically respond to database events and can be used for auditing or enforcing business rules.	Database triggers are used for creating tables.	C

40	What are some common use cases for using cursor parameters in PL/SQL?	Cursor parameters are rarely used in practice.	Common use cases include generating reports with different filter criteria, processing data based on user inputs, and customizing data retrieval based on changing conditions.	Cursor parameters are mainly used for database administration tasks.	Cursor parameters are only used in triggers.	B
41	What are the advantages of using explicit cursors over implicit cursors in PL/SQL?	Explicit cursors are faster in performance.	Implicit cursors are more flexible.	Explicit cursors are easier to use and provide more control.	Implicit cursors are automatically managed by the database.	C
42	What are the benefits of using PL/SQL for database programming compared to using SQL alone?	PL/SQL allows for creating web applications.	PL/SQL provides procedural capabilities for better control and encapsulation of logic in the database.	PL/SQL is used for hardware design.	PL/SQL is primarily used for data modeling.	B
43	What does the isolation level READ COMMITTED mean?	Reads data as it was when the transaction started.	Reads uncommitted changes made by other transactions.	Prevents any reads until the transaction is committed.	Reads data from committed transactions only.	A
44	What does the SAVEPOINT statement do in SQL?	Marks a point in a transaction to be rolled back to later.	Commits the transaction.	Opens a new transaction.	Locks the database.	A
45	What happens when you fetch data from a cursor that has no more rows to retrieve?	An error occurs.	The cursor is automatically closed.	The cursor remains open and ready for the next fetch.	Cursors always have more rows to retrieve.	A
46	What is a cursor in PL/SQL, and why is it used?	A cursor is a database table.	A cursor is used for looping through query results.	A cursor is a data type in PL/SQL.	A cursor is used for creating triggers.	B
47	What is a database checkpoint?	A physical location where the database is stored.	A marker indicating the point in time up to which transactions are considered safe and can be recovered.	A log file containing SQL statements.	A password for accessing the database.	B
48	What is a database lock in the context of concurrency control?	A mechanism to block all database transactions.	A mechanism to prevent data corruption.	A mechanism to prevent multiple transactions from accessing the same data simultaneously.	A mechanism to unlock databases.	C
49	What is a database restore operation?	A process that erases all data from the database.	A process that removes the database log files.	A process that brings a database back to a previous state by applying database backups and log files.	A process that upgrades the database to a new version.	C
50	What is a database transaction?	A single SQL statement.	A sequence of related SQL statements that are executed as a unit.	A database schema.	A database table.	B

51		A situation where a transaction is rolled back.	A situation where two or more transactions are waiting for each other to release locks.	A situation where a transaction is terminated.	A situation where a transaction is committed.	B
	What is a deadlock in the context of concurrency control?					
52		A transaction that involves multiple databases.	A transaction that is committed automatically.	A transaction with a large number of SQL statements.	A transaction without a COMMIT.	A
	What is a distributed transaction in database management?					
53		A backup that includes only a subset of the database.	A backup that includes all the data and structures in the database.	A backup that contains only log files.	A backup that is encrypted for security.	B
	What is a full database backup?					
54		A transaction inside another transaction.	A transaction without any nested SQL statements.	A transaction that cannot be rolled back.	A transaction with a SAVEPOINT.	A
	What is a nested transaction in SQL?					
55		A function is used for controlling database transactions.	A function is used for encapsulating reusable logic and returns a value.	A procedure is used for data modeling.	A procedure is used for creating tables.	B
	What is a PL/SQL function, and how does it differ from a procedure?					
56		Managing multiple database transactions simultaneously.	Controlling access to the database using passwords.	Rolling back transactions in case of errors.	Creating indexes for database tables.	A
	What is concurrency control in database systems?					
57		Cursor positioning determines the cursor's name.	Cursor positioning is the process of opening a cursor.	Cursor positioning refers to the current position of the cursor relative to the result set, affecting the next fetch operation.	Cursor positioning is not relevant in PL/SQL.	C
	What is cursor positioning, and how does it relate to fetching data?					
58		Backing up the database to prevent data loss.	The process of restoring a database to a previous state after a failure.	Increasing the database size to accommodate more data.	Encrypting database files for security.	B
	What is database recovery in the context of database management systems?					
59		Dynamic SQL is used for creating triggers in PL/SQL.	Dynamic SQL allows you to generate and execute SQL statements at runtime.	Dynamic SQL is used for web development.	Dynamic SQL is used for hardware design.	B
	What is dynamic SQL, and why might you use it in PL/SQL?					
60		PL/SQL is a markup language for web development.	PL/SQL is a procedural extension of SQL.	PL/SQL is a data modeling language.	PL/SQL is a hardware description language.	B
	What is PL/SQL, and how does it differ from SQL?					
61		Atomicity, Consistency, Isolation, Durability	Aggregation, Continuity, Integrity, Durability	Affinity, Consistency, Isolation, Durability	Atomicity, Cancellation, Isolation, Division	A
	What is the ACID property in the context of database transactions?					
62		Cursor FOR Loops are slower than traditional cursors.	Cursor FOR Loops offer less control.	Cursor FOR Loops simplify cursor processing by handling cursor declaration, opening, fetching, and closing automatically.	Cursor FOR Loops are not recommended in PL/SQL.	C
	What is the advantage of using a Cursor FOR Loop over traditional cursor processing?					

63	What is the benefit of using cursor parameters when working with data retrieval?	Cursor parameters make cursor declaration simpler.	Cursor parameters allow for dynamic queries and customization of data retrieval based on varying conditions.	Cursor parameters improve cursor performance.	Cursor parameters are not useful in PL/SQL.	B
64	What is the default behavior of a Cursor FOR Loop if there are no rows to process?	It raises an error.	It skips the loop and continues with the next statement.	It automatically exits the loop.	It waits for rows to be available.	C
65	What is the difference between a PL/SQL procedure and a PL/SQL function?	A procedure returns a value, while a function does not.	A procedure does not return a value, while a function does.	A procedure and a function are the same.	A procedure and a function are not supported in PL/SQL.	B
66	What is the opposite of a COMMIT statement in SQL?	ROLLBACK	BEGIN TRANSACTION	SAVEPOINT	LOCK	A
67	What is the primary drawback of using pessimistic concurrency control in a database system?	It can lead to data inconsistency.	It can result in excessive locking and reduced concurrency.	It is slower than optimistic concurrency control.	It requires frequent COMMIT statements.	B
68	What is the purpose of a differential backup in database recovery?	To recover the database to a specific point in time.	To restore only the data that has changed since the last full backup, reducing the recovery time.	To make a copy of the entire database.	To compress the database backup files.	B
69	What is the purpose of declaring a cursor in PL/SQL?	To insert data into a table.	To define a variable in PL/SQL.	To retrieve and manipulate query results in a controlled manner.	To create a trigger in PL/SQL.	C
70	What is the purpose of fetching data from a cursor in PL/SQL?	To insert data into a table.	To define a variable in PL/SQL.	To retrieve and manipulate query results row by row.	To create a trigger in PL/SQL.	C
71	What is the purpose of the COMMIT statement in SQL?	To roll back a transaction.	To save all pending changes permanently to the database.	To lock database records.	To create a new transaction.	B
72	What is the purpose of the FOR loop in PL/SQL, and how is it used?	The FOR loop is used for declaring variables.	The FOR loop is used for defining exceptions.	The FOR loop is used for iterative processing.	The FOR loop is used for database connections.	C
73	What is the purpose of the LOCK TABLE statement in SQL?	To unlock a table.	To create a new table.	To specify the locking mode for a table explicitly.	To commit a transaction.	C
74	What is the purpose of the WHEN OTHERS exception handler in PL/SQL?	It is used for declaring variables.	It is used for defining custom exceptions.	It is used to catch and handle unexpected exceptions.	It is used for database connections.	C
75	What is the role of a database backup in recovery?	It serves as a temporary storage location.	It records changes made to the database.	It provides a copy of the database that can be used to restore data in case of data loss or corruption.	It ensures data consistency during concurrent transactions.	C
76	What is the role of the FETCH statement in cursor processing?	It defines the cursor's name.	It specifies the number of rows to fetch.	It retrieves rows from the cursor into variables or records.	It opens the cursor for data retrieval.	C

77		It defines the cursor's name.	It specifies the number of rows the cursor can fetch.	It defines the structure of the result set the cursor will hold.	It determines the data type of cursor variables.	C
	What is the significance of the %ROWTYPE attribute when declaring a cursor?					
78		It defines the maximum number of recovery points allowed.	It specifies the desired point in time to which a database should be recovered after a failure.	It defines the number of database logs to retain.	It measures the database's performance.	B
	What is the significance of the recovery point objective (RPO) in database recovery planning?					
79	Which ACID property ensures that a transaction is completed in its entirety or not at all?	Atomicity	Consistency	Isolation	Durability	A

80	Which concurrency control technique allows conflicts to be detected and resolved only at the commit time?	Validation-based protocol	Timestamp ordering	Two-phase locking	Three-phase locking	A
81	Which database recovery model allows for point-in-time recovery to any arbitrary moment?	Simple recovery model	Full recovery model	Bulk-logged recovery model	Incremental recovery model	B
82	Which isolation level in SQL provides the highest level of isolation but can lead to concurrency issues?	READ COMMITTED	SERIALIZABLE	READ UNCOMMITTED	REPEATABLE READ	B
83	Which of the following is not a concurrency control mechanism in DBMS?	Locking	Timestamp ordering	Multiversion concurrency control	Rollback and recovery	D
84	Which of the following recovery techniques is based on maintaining multiple copies of the database at different points in time?	Replication	Deferred update	Redo logging	Undo logging	A
85	Which SQL statement is used to set a SAVEPOINT within a transaction?	BEGIN SAVEPOINT	SAVEPOINT	SET SAVEPOINT	CREATE SAVEPOINT	B
86	Which technique allows concurrent transactions to access different parts of a database without conflicts?	Pessimistic concurrency control	Optimistic concurrency control	Exclusive locking	Distributed transactions	B
87	[ON table_name] specifies the name of the table associated with the trigger, some events occur.	Yes	No	Can be yes or no	None of the above	A
88		Procedure	Triggers	Collection	Transaction	B
89	A _____ consists of a sequence of query and/or update statements.	Transaction	Commit	Rollback	Flashback	A
90	A _____ is a special kind of a store procedure that executes in response to certain action on the table like insertion, deletion or updation of data.	Procedures	Triggers	Functions	None of the mentioned	B
91	A stored procedure in SQL is a _____	Block of functions	Group of Transact-SQL statements compiled into a single execution plan.	Group of distinct SQL statements.	None of the mentioned	B
92	A view is actually a?	composition of a table	decomposition of a table	associated to a table	None of the above	A
93	All objects placed in the specification are called _____ objects.	private	protected	public	None of the above	B
94	Any subprogram not in the package specification but coded in the package body is called a _____ object.	protected	private	self	public	B
95	Boyce-Codd Normal Form (BCNF) is an extension of which normal form?	A) First Normal Form (1NF)	B) Second Normal Form (2NF)	C) Third Normal Form (3NF)	D) Fourth Normal Form (4NF)	B
96	Consider the following action: TRANSACTION..... Commit; ROLLBACK; What does Rollback do?	Undoes the transactions before commit	Clears all transactions	Redoes the transactions before commit	No action	D
97	Consider the following cursor declaration in SQL: DECLARE cursor1 CURSOR FOR SELECT FirstName, LastName FROM Employees WHERE Department = 'Sales' If you want to open and fetch rows from this cursor, what SQL statement should you use?	FETCH NEXT FROM cursor1;	OPEN cursor1;	CLOSE cursor1;	DECLARE cursor1 CURSOR FOR ...	B
98	Consider the following schedule involving two transactions T1 and T2: S = r2(X); r1(X); r2(Y); w1(X); r1(Y); w2(X); a1; a2 where r2(Z) denotes a read operation by transaction T1 on a variable Z, w1(Z) denotes a write operation by T1 on a variable Z and a1 denotes an abort by transaction T1. Which one of the following statements about the above schedule is TRUE?	S is non-recoverable	S is recoverable, but has a cascading abort	S does not have a cascading abort	S is strict	C
99	Consider the following stored procedure in SQL: CREATE PROCEDURE CalculateTotalPrice @ProductID INT, @Quantity INT AS BEGIN DECLARE @Price DECIMAL(10, 2) SELECT @Price = UnitPrice FROM Products WHERE ProductID = @ProductID PRINT Total Price: + CAST(@Price * @Quantity AS VARCHAR) END If you call this stored procedure with @ProductID = 101 and @Quantity = 5, what will be the output?	Total Price: 505	Total Price: 25	Total Price: 101	Total Price: 5	B
100	Create function dept count(dept_name varchar(20)) begin declare d count integer; select count(*) into d count from instructor where instructor.dept_name= dept_name return d count; end Find the error in the above statement.	Return type missing	Dept_name is mismatched	Reference relation is not mentioned	All of the mentioned	A
101	CREATE OR REPLACE FUNCTION calculate_gpa( student_id NUMBER ) RETURN NUMBER IS total_points NUMBER := 0; total_credits NUMBER := 0; gpa NUMBER; BEGIN -- Calculate GPA for a student -- Assume the grade scale: A=4, B=3, C=2, D=1, F=0 -- Credits per subject: 3 credits -- GPA = (Total Points) / (Total Credits) RETURN gpa; END;	Deletes students by age.	Updates student's name.	Calculates a student's GPA based on their grades.	Retrieves students in a subject above average.	C
102	CREATE OR REPLACE FUNCTION calculate_student_gpa( student_id NUMBER ) RETURN NUMBER IS gpa NUMBER; BEGIN -- Calculate GPA for a student based on their grades and credit hours RETURN gpa; END;	Deletes students by age.	Updates student's name.	Calculates a student's GPA based on their grades and credit hours.	Transfers students from one batch to another.	C
103	CREATE OR REPLACE FUNCTION calculate_subject_average( subject_name VARCHAR2 ) RETURN NUMBER IS avg_grade NUMBER; BEGIN SELECT AVG(grade) INTO avg_grade FROM student WHERE subject = subject_name; RETURN avg_grade; END;	Deletes students by age.	Updates student information.	Calculates the average grade in a specific subject.	Returns a list of students with above-average grades in a subject.	C
104	CREATE OR REPLACE FUNCTION calculate_total_students RETURN NUMBER IS total_students NUMBER; BEGIN SELECT COUNT(*) INTO total_students FROM student; RETURN total_students; END;	Deletes students by name.	Updates student information.	Calculates the total number of students.	Returns a list of students by age range.	C



105	<pre> CREATE OR REPLACE FUNCTION count_students_by_age(     age NUMBER ) RETURN NUMBER IS     student_count NUMBER; BEGIN     SELECT COUNT(*) INTO student_count     FROM student     WHERE age = age;     RETURN student_count; END;</pre>	Retrieves students by age.	Returns the total count of students by age.	Enrolls students in multiple subjects.	Deletes students by subject.	B
106	<pre> CREATE OR REPLACE FUNCTION count_students_in_subject(     subject_name VARCHAR2 ) RETURN NUMBER IS     student_count NUMBER; BEGIN     SELECT COUNT(*) INTO student_count     FROM student     WHERE subject = subject_name;     RETURN student_count; END;</pre>	Retrieves students by subject count.	Returns a list of students with the maximum number of subjects.	Deletes students by age.	Updates a student's batch.	A
107	<pre> SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE age = (SELECT MAX(age) FROM student);     RETURN students_cursor; END;</pre>	Retrieves students with the lowest age.	Returns a list of students with the highest age.	Deletes students by age.	Awards scholarships to deserving students.	B
108	<pre> SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM (             SELECT student_id, COUNT(DISTINCT subject) AS subject_count             FROM student             GROUP BY student_id             ORDER BY subject_count DESC         )         WHERE ROWNUM = 1;     RETURN students_cursor; END;</pre>	Retrieves students with the highest grades.	Returns a list of subjects with above-average grades.	Enrolls students in multiple subjects.	Deletes students by subject.	B
109	<pre> CREATE OR REPLACE FUNCTION find_students_with_subject_count(     subject_count NUMBER ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM (             SELECT student_id, COUNT(DISTINCT subject) AS subject_count             FROM student             GROUP BY student_id         )         WHERE subject_count = subject_count;     RETURN students_cursor; END;</pre>	Retrieves students by subject count.	Returns a list of students with a specific subject count.	Deletes students by age.	Updates a student's subject and grade.	A
110	<pre> SYS_REFCURSOR IS     subjects_cursor SYS_REFCURSOR; BEGIN     OPEN subjects_cursor FOR         SELECT subject         FROM (             SELECT subject, COUNT(*) AS student_count             FROM student             GROUP BY subject             ORDER BY student_count ASC         )         WHERE ROWNUM = 1;     RETURN subjects_cursor; END;</pre>	Retrieves subjects with the highest grades.	Returns a list of subjects with the lowest number of students.	Enrolls students in multiple subjects.	Deletes students by subject.	B
111	<pre> CREATE OR REPLACE FUNCTION get_highest_grade_by_subject(     subject_name VARCHAR2 ) RETURN NUMBER IS     highest_grade NUMBER; BEGIN     SELECT MAX(grade) INTO highest_grade     FROM student     WHERE subject = subject_name;     RETURN highest_grade; END;</pre>	Returns the highest grade of all students.	Enrolls students in a subject.	Deletes students by subject.	Returns the highest grade in a specific subject.	D
112	<pre> CREATE OR REPLACE FUNCTION get_student_count_by_subject(     subject_name VARCHAR2 ) RETURN NUMBER IS     student_count NUMBER; BEGIN     SELECT COUNT(*) INTO student_count     FROM student     WHERE subject = subject_name;     RETURN student_count; END;</pre>	Deletes a student by subject.	Returns the count of students in a specific subject.	Enrolls a student in a subject.	Updates the student's subject.	B
113	<pre> subject_name VARCHAR2 ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE subject = subject_name AND grade &gt; (SELECT AVG(grade) FROM student         WHERE subject = subject_name);     RETURN students_cursor; END;</pre>	Retrieves all students.	Returns a list of students with above-average grades in a specific subject.	Enrolls students in a subject.	Deletes students by subject.	B
114	<pre> CREATE OR REPLACE FUNCTION get_students_by_age_range(     min_age NUMBER,     max_age NUMBER ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE age BETWEEN min_age AND max_age;     RETURN students_cursor; END;</pre>	Retrieves students by age range.	Returns a list of students with above-average grades in a specific subject.	Enrolls students in multiple subjects.	Deletes students by subject.	A
115	<pre> CREATE OR REPLACE FUNCTION get_students_by_grade_range(     min_grade NUMBER,     max_grade NUMBER ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE grade BETWEEN min_grade AND max_grade;     RETURN students_cursor; END;</pre>	Retrieves students by grade range.	Returns the total count of students by grade range.	Awards scholarships to deserving students.	Transfers students from one batch to another.	A

116	<pre> CREATE OR REPLACE FUNCTION get_students_by_subject_and_grade(     subject_name VARCHAR2,     min_grade NUMBER ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE subject = subject_name AND grade &gt;= min_grade;     RETURN students_cursor; END; </pre>	Retrieves students by age range.	Returns a list of students with a specific subject and minimum grade.	Enrolls students in multiple subjects.	Deletes students by subject.	B
117	<pre> IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE batch_id IS NOT NULL;     RETURN students_cursor; END; </pre>	Retrieves students in a specific batch.	Returns a list of students with the highest age.	Enrolls students in multiple subjects.	Deletes students by subject.	A
118	<pre> subject_name VARCHAR2 ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE subject = subject_name         AND grade &gt; (SELECT AVG(grade) FROM student WHERE subject =         subject_name);     RETURN students_cursor; END; </pre>	Retrieves all students.	Returns a list of students with above-average grades in a specific subject.	Enrolls students in a subject.	Updates student information.	B
119	<pre> SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE grade = (SELECT MAX(grade) FROM student);     RETURN students_cursor; END; </pre>	Retrieves students with the lowest grade.	Returns students with the highest grade.	Enrolls students in a batch.	Deletes students by age.	B
120	<pre> SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM student         WHERE grade = (SELECT MIN(grade) FROM student);     RETURN students_cursor; END; </pre>	Retrieves students with the highest grade.	Returns students with the lowest grade.	Enrolls students in a batch.	Updates student information.	B
121	<pre> RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR         SELECT student_id         FROM (             SELECT student_id, MAX(grade) AS max_grade             FROM student             GROUP BY student_id         );     RETURN students_cursor; END; </pre>	Retrieves students with the subject wise lowest grades.	Returns students with the subject wise highest grades.	Enrolls students in multiple subjects.	Deletes students by subject.	B
122	<pre> CREATE OR REPLACE FUNCTION get_subjects_by_student(     student_id NUMBER ) RETURN SYS_REFCURSOR IS     subjects_cursor SYS_REFCURSOR; BEGIN     OPEN subjects_cursor FOR         SELECT DISTINCT subject         FROM student         WHERE student_id = student_id;     RETURN subjects_cursor; END; </pre>	Deletes a student record.	Updates a student's information.	Returns a list of distinct subjects for a specific student.	Enrolls students in a subject.	C
123	<pre> SYS_REFCURSOR IS     subjects_cursor SYS_REFCURSOR; BEGIN     OPEN subjects_cursor FOR         SELECT subject         FROM (             SELECT subject, MAX(grade) AS max_grade             FROM student             GROUP BY subject         );     RETURN subjects_cursor; END; </pre>	Retrieves subjects with the highest grades.	Returns a list of subjects with above-average grades.	Enrolls students in multiple subjects.	Deletes students by subject.	A
124	<pre> RETURN SYS_REFCURSOR IS     subjects_cursor SYS_REFCURSOR; BEGIN     OPEN subjects_cursor FOR         SELECT subject         FROM (             SELECT subject, AVG(grade) AS avg_grade             FROM student             GROUP BY subject         )         WHERE avg_grade &gt; (SELECT AVG(grade) FROM student);     RETURN subjects_cursor; END; </pre>	Retrieves subjects with students above average grades.	Returns a list of subjects with students below average grades.	Enrolls students in multiple subjects.	Deletes students by subject.	A
125	<pre> CREATE OR REPLACE PROCEDURE archive_student_records IS BEGIN     -- Write logic to archive old student records END; </pre>	Deletes students by age.	Updates student information.	Archives old student records.	Awards scholarships to deserving students.	C
126	<pre> CREATE OR REPLACE PROCEDURE assign_student_grade(     student_id NUMBER,     subject_name VARCHAR2,     grade NUMBER ) IS BEGIN     -- Write logic to assign a specific grade to a student in a subject END; </pre>	Deletes students by age.	Updates a student's name.	Assigns a specific grade to a student.	Promotes students to the next grade.	C
127	<pre> CREATE OR REPLACE PROCEDURE assign_student_subjects_and_grades(     student_id NUMBER,     subject_grades SYS.ODCINUMBERLIST ) IS BEGIN     -- Write logic to assign subjects and grades to a student END; </pre>	Deletes students by age.	Updates student information.	Assigns subjects and grades to a student.	Promotes students to the next grade.	C
128	<pre> CREATE OR REPLACE PROCEDURE delete_student(     student_id NUMBER ) IS BEGIN     DELETE FROM student     WHERE student_id = student_id;     COMMIT; END; </pre>	Retrieve student details by ID.	Update student information.	Enroll a new student.	Delete a student record.	D
129	<pre> CREATE OR REPLACE PROCEDURE delete_students_by_age(     max_age NUMBER ) IS BEGIN     DELETE FROM student     WHERE age &gt; max_age;     COMMIT; END; </pre>	Deletes students by name.	Updates student age.	Deletes students older than a specified age.	Calculates students' GPA.	C

130	CREATE OR REPLACE PROCEDURE delete_students_by_batch( batch_id NUMBER )IS BEGIN DELETE FROM student WHERE batch_id = batch_id; COMMIT; END;	Deletes students by age.	Updates student information.	Deletes students by batch.	Assigns subjects and grades to a student.	C
131	CREATE OR REPLACE PROCEDURE delete_students_by_subject( subject_name VARCHAR2 )IS BEGIN DELETE FROM student WHERE subject = subject_name; COMMIT; END;	Deletes students by age.	Updates student information.	Deletes students by subject.	Enrolls students in multiple subjects.	C
132	student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER )IS BEGIN INSERT INTO student(student_id, name, age, subject, grade) VALUES(student_sequence.NEXTVAL, student_name, student_age, subject, student_grade); COMMIT; END;	Deletes a student record.	Updates a student's information.	Enrolls a new student with provided details.	Returns the count of students in a specific subject.	C
133	CREATE OR REPLACE PROCEDURE enroll_student_in_multiple_subjects( student_name VARCHAR2, student_age NUMBER, subjects VARCHAR2, student_grades VARCHAR2 )IS BEGIN -- Write logic to enroll a student in multiple subjects with corresponding grades END;	Retrieve student details by ID.	Update student information.	Enroll students in multiple subjects.	Delete students by subject.	C
134	CREATE OR REPLACE PROCEDURE increase_student_grades( grade_increase NUMBER )IS BEGIN UPDATE student SET grade = grade + grade_increase; COMMIT; END;	Deletes students by age.	Updates student information.	Increases student grades.	Awards scholarships to deserving students.	C
135	CREATE OR REPLACE PROCEDURE print_student_details( student_id NUMBER )IS student_name VARCHAR2(100); student_age NUMBER; subject_name VARCHAR2(50); student_grade NUMBER; BEGIN SELECT name, age, subject, grade INTO student_name, student_age, subject_name, student_grade FROM student WHERE student_id = student_id;  DBMS_OUTPUT.PUT_LINE('Student ID: '    student_id); DBMS_OUTPUT.PUT_LINE('Name: '    student_name); DBMS_OUTPUT.PUT_LINE('Age: '    student_age);	Deletes a student record.	Updates student information.	Prints details of a student by ID.	Returns the count of students in a specific subject.	C
136	CREATE OR REPLACE PROCEDURE print_student_transcript( student_id NUMBER )IS BEGIN -- Write logic to print the student's transcript END;	Deletes students by age.	Updates student information.	Prints a student's transcript.	Returns students with the lowest grade.	C
137	student_id NUMBER )IS BEGIN FOR rec IN (SELECT subject, grade FROM student WHERE student_id = student_id) LOOP DBMS_OUTPUT.PUT_LINE('Subject: '    rec.subject    ', Grade: '    rec.grade); END LOOP; END;	Deletes a student record.	Updates student information.	Prints subjects and grades of a student by ID.	Returns a list of subjects in which a student is enrolled.	C
138	CREATE OR REPLACE PROCEDURE transfer_student( student_id NUMBER, new_batch_id NUMBER )IS BEGIN -- Write logic to transfer a student to a new batch END;	Deletes students by age.	Updates student information.	Transfers a student to a new batch.	Calculates the average grade in a subject.	C
139	CREATE OR REPLACE PROCEDURE update_student_age( student_id NUMBER, new_age NUMBER )IS BEGIN UPDATE student SET age = new_age WHERE student_id = student_id; COMMIT; END; /	Deletes a student record.	Updates a student's age.	Enrolls a new student.	Returns the count of students in a specific subject.	B
140	CREATE OR REPLACE PROCEDURE update_student_name( student_id NUMBER, new_name VARCHAR2 )IS BEGIN UPDATE student SET name = new_name WHERE student_id = student_id; COMMIT; END;	Deletes students by age.	Updates a student's name.	Enrolls a new student.	Returns the count of students in a specific subject.	B
141	CREATE OR REPLACE PROCEDURE update_student_subject_and_grade( student_id NUMBER, subject_name VARCHAR2, new_grade NUMBER )IS BEGIN -- Write logic to update a student's subject and grade END;	Deletes a student's subject and grade.	Updates a student's subject and grade.	Enrolls a new student.	Returns a student's subject and grade.	B
142	Create procedure dept_count proc(in dept name varchar(20), out d count integer) begin select count(*) into d count from instructor where instructor.dept name= dept count proc.dept name end Which of the following is used to call the procedure given above ?	Declare d_count integer;	Declare d_count integer; call dept_count proc('Physics', d_count);	Declare d_count integer; call dept_count proc('Physics');	Declare d_count; call dept_count proc('Physics', d_count);	B
143	DECLARE exit handler FOR OUT OF classroom seats BEGIN SEQUENCE OF statements END The above statements are used for	Calling procedures	Handling Exception	Handling procedures	All of the mentioned	B
144	internal database error.	Yes	No	1 or 2	All of the above	A

145	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to calculate the average grade for each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN FOR subject_rec IN subject_cursor LOOP -- Calculate and print the average grade for each subject END LOOP; END; </pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; avg_grade NUMBER; BEGIN FOR subject_rec IN subject_cursor LOOP -- Calculate and store the average grade for each subject in avg_grade DBMS_OUTPUT.PUT_ LINE('Subject: '    subject_rec.subject    ', Avg Grade: '    avg_grade); END LOOP; END; </pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_rec student% ROWTYPE; BEGIN FOR subject_rec IN subject_cursor LOOP -- Calculate and print the average grade for each subject END LOOP; END; </pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN FOR subject_rec IN subject_cursor LOOP -- Calculate and print the average grade for each subject END LOOP; END; </pre>	B
146	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to fetch all students who have a grade greater than or equal to 90 in the subject "Mathematics." Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR high_math_grades IS SELECT student_id FROM student WHERE subject = 'Mathematics' AND grade &gt;= 90; BEGIN FOR rec IN high_math_grades LOOP DBMS_OUTPUT.PUT_ LINE('Student ID: '    rec.student_id); END LOOP; END; </pre>	<pre> DECLARE CURSOR high_math_grades IS SELECT student_id FROM student WHERE subject = 'Mathematics' AND grade &gt;= 90; student_id NUMBER; BEGIN OPEN high_math_grades; LOOP FETCH high_math_grades INTO student_id; EXIT WHEN high_math_grades% NOTFOUND; DBMS_OUTPUT.PUT_ LINE('Student ID: '    student_id); END LOOP; CLOSE high_math_grades; END; </pre>	<pre> DECLARE CURSOR high_math_grades IS SELECT student_id FROM student WHERE subject = 'Mathematics' AND grade &gt;= 90; student_id NUMBER; BEGIN OPEN high_math_grades; LOOP FETCH high_math_grades INTO student_id; EXIT WHEN high_math_grades% NOTFOUND; DBMS_OUTPUT.PUT_ LINE('Student ID: '    student_id); END LOOP; CLOSE high_math_grades; END; </pre>	<pre> DECLARE CURSOR high_math_grades(student_id NUMBER) IS SELECT student_id FROM student WHERE subject = 'Mathematics' AND grade &gt;= 90; BEGIN FOR rec IN high_math_grades(90) LOOP DBMS_OUTPUT.PUT_ LINE('Student ID: '    rec.student_id); END LOOP; END; </pre>	B
147	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the student with the highest grade for each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN FOR subject_rec IN subject_cursor LOOP -- Find and print the student with the highest grade for each subject END LOOP; END; </pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; highest_grade NUMBER; BEGIN FOR subject_rec IN subject_cursor LOOP -- Calculate and store the highest grade for each subject in highest_grade DBMS_OUTPUT.PUT_ LINE('Subject: '    subject_rec.subject    ', Highest Grade: '    highest_grade); END LOOP; END; </pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; student_rec student% ROWTYPE; BEGIN FOR subject_rec IN subject_cursor LOOP -- Find and print the student with the highest grade for each subject END LOOP; END; </pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN FOR subject_rec IN subject_cursor LOOP -- Find and print the student with the highest grade for each subject END LOOP; END; </pre>	A
148	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have improved their grades in at least one subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have improved their grades END; </pre>	<pre> DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; improved_students VARCHAR2(4000); BEGIN -- Calculate and store the list of improved students in improved_students DBMS_OUTPUT.PUT_ LINE('Improved Students: '    improved_students); END; </pre>	<pre> DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_rec student% ROWTYPE; BEGIN -- Find and print students who have improved their grades END; </pre>	<pre> DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have improved their grades END; </pre>	B

149	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; not_improved_stude nts VARCHAR2(4000); BEGIN -- Calculate and store the list of students who have not improved their grades in not_improved_stude nts); DBMS_OUTPUT.PUT LINE('Students Who Have Not Improved Their Grades: '    not_improved_stude nts); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	B
150	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	B
151	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; not_improved_stude nts VARCHAR2(4000); BEGIN -- Calculate and store the list of students who have not improved their grades in not_improved_stude nts); DBMS_OUTPUT.PUT LINE('Students Who Have Not Improved Their Grades: '    not_improved_stude nts); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	B
152	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; not_improved_stude nts VARCHAR2(4000); BEGIN -- Calculate and store the list of students who have not improved their grades in not_improved_stude nts); DBMS_OUTPUT.PUT LINE('Students Who Have Not Improved Their Grades: '    not_improved_stude nts); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	B

153	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; not_improved_stude nts VARCHAR2(4000); BEGIN -- Calculate and store the list of students who have not improved their grades in not_improved_stude nts DBMS_OUTPUT.PUT LINE('Students Who Have Not Improved Their Grades: '    not_improved_stude nts); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordsstudent %ROWTYPE; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	B
154	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; not_improved_stude nts VARCHAR2(4000); BEGIN -- Calculate and store the list of students who have not improved their grades in not_improved_stude nts DBMS_OUTPUT.PUT LINE('Students Who Have Not Improved Their Grades: '    not_improved_stude nts); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordsstudent %ROWTYPE; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	B
155	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; not_improved_stude nts VARCHAR2(4000); BEGIN -- Calculate and store the list of students who have not improved their grades in not_improved_stude nts DBMS_OUTPUT.PUT LINE('Students Who Have Not Improved Their Grades: '    not_improved_stude nts); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordsstudent %ROWTYPE; BEGIN -- Find and print students who have not improved their grades END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have not improved their grades END;</pre>	B
156	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in at least one subject and the lowest grade in at least one subject. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students with highest and lowest grades in subjects END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; students_with_extre me_grades VARCHAR2(4000); BEGIN -- Calculate and store the list of students with extreme grades in students_with_extre me_grades DBMS_OUTPUT.PUT LINE('Students with Extreme Grades: '    students_with_extre me_grades); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordsstudent %ROWTYPE; BEGIN -- Find and print students with highest and lowest grades in subjects END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students with highest and lowest grades in subjects END;</pre>	B

157	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; highest_grade NUMBER; BEGIN -- Calculate and store the highest grade for each subject in highest_grade DBMS_OUTPUT.PUT_ LINE('Students with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; student_recordsstudent %ROWTYPE; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	B
158	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; highest_grade NUMBER; BEGIN -- Calculate and store the highest grade for each subject in highest_grade DBMS_OUTPUT.PUT_ LINE('Students with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; student_recordsstudent %ROWTYPE; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	B
159	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; highest_grade NUMBER; BEGIN -- Calculate and store the highest grade for each subject in highest_grade DBMS_OUTPUT.PUT_ LINE('Students with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; student_recordsstudent %ROWTYPE; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	B
160	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; highest_grade NUMBER; BEGIN -- Calculate and store the highest grade for each subject in highest_grade DBMS_OUTPUT.PUT_ LINE('Students with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; student_recordsstudent %ROWTYPE; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	B

161	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; highest_grade NUMBER; BEGIN -- Calculate and store the highest grade for each subject in highest_grade DBMS_OUTPUT.PUT_ LINE('Students with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the highest grade in each subject END;</pre>	B
162	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the lowest grade in each subject. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the lowest grade in each subject END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; lowest_grade NUMBER; BEGIN -- Calculate and store the lowest grade for each subject in lowest_grade DBMS_OUTPUT.PUT_ LINE('Students with Lowest Grade in Each Subject: '    lowest_grade); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students with the lowest grade in each subject END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the lowest grade in each subject END;</pre>	B
163	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the lowest grade in each subject. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the lowest grade in each subject END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; lowest_grade NUMBER; BEGIN -- Calculate and store the lowest grade for each subject in lowest_grade DBMS_OUTPUT.PUT_ LINE('Students with Lowest Grade in Each Subject: '    lowest_grade); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students with the lowest grade in each subject END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print students with the lowest grade in each subject END;</pre>	B
164	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the same grade in all subjects. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; students_with_same _grade VARCHAR2(4000); BEGIN -- Calculate and store the list of students with the same grade in all subjects in students_with_same _grade END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students who have scored the same grade in all subjects END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students who have scored the same grade in all subjects END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have scored the same grade in all subjects END;</pre>	B



165	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the students who have scored the same grade in all subjects. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have scored the same grade in all subjects END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; students_with_same _grade VARCHAR2(4000); BEGIN -- Calculate and store the list of students with the same grade in all subjects in students_with_same _grade DBMS_OUTPUT.PUT_ LINE('Students with Same Grade in All Subjects: '    students_with_same _grade); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_records %ROWTYPE; BEGIN -- Find and print students who have scored the same grade in all subjects END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have scored the same grade in all subjects END;</pre>	B
166	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subject with the highest overall grades (sum of grades for all students). Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print the subject with the highest overall grades END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; highest_subject VARCHAR2(100); BEGIN -- Calculate and store the subject with the highest overall grades in highest_subject DBMS_OUTPUT.PUT_ LINE('Subject with Highest Overall Grades: '    highest_subject); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_records %ROWTYPE; BEGIN -- Find and print the subject with the highest overall grades END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print the subject with the highest overall grades END;</pre>	B
167	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subject with the lowest average grade. Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print the subject with the lowest average grade END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; lowest_avg_grade NUMBER; BEGIN -- Calculate and store the lowest average grade for a subject in lowest_avg_grade DBMS_OUTPUT.PUT_ LINE('Subject with Lowest Average Grade: '    lowest_avg_grade); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_records %ROWTYPE; BEGIN -- Find and print the subject with the lowest average grade END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print the subject with the lowest average grade END;</pre>	B
168	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which all students have scored above a specified threshold (e.g., 60). Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_all_hig h_scores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with all high scores in subjects_with_all_hig h_scores DBMS_OUTPUT.PUT_ LINE('Subjects with All High Scores: '    subjects_with_all_hig h_scores); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_records %ROWTYPE; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	B

169	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which all students have scored above a specified threshold (e.g., 70). Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_all_high_h_scores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with all high scores in subjects_with_all_high_h_scores END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	B
170	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 40). Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_low_s_cores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with at least one low score in subjects_with_low_s_cores END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	A
171	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 50). Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_low_s_cores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with at least one low score in subjects_with_low_s_cores END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	B
172	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 50). Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_low_s_cores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with at least one low score in subjects_with_low_s_cores END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	A

173	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 55). Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_low_s cores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with at least one low score in subjects_with_low_s cores DBMS_OUTPUT.PUT_ LINE('Subjects with Low Scores: '    subjects_with_low_s cores); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	B
174	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 60). Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_low_s cores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with at least one low score in subjects_with_low_s cores DBMS_OUTPUT.PUT_ LINE('Subjects with Low Scores: '    subjects_with_low_s cores); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with at least one score below the threshold END;</pre>	B
175	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which every student has scored above a specified threshold (e.g., 85). Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_all_hig h_scores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with all high scores in subjects_with_all_hig h_scores DBMS_OUTPUT.PUT_ LINE('Subjects with All High Scores: '    subjects_with_all_hig h_scores); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	B
176	<p>Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which every student has scored above a specified threshold (e.g., 90). Which of the following code snippets accomplishes this task?</p>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_all_hig h_scores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with all high scores in subjects_with_all_hig h_scores DBMS_OUTPUT.PUT_ LINE('Subjects with All High Scores: '    subjects_with_all_hig h_scores); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with all scores above the threshold END;</pre>	B

177	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which no student has scored below a specified threshold (e.g., 60). Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with no scores below the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_with_no_lo w_scores VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with no low scores in subjects_with_no_lo w_scores DBMS_OUTPUT.PUT_ LINE('Subjects with No Low Scores: '    subjects_with_no_lo w_scores); END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with no scores below the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with no scores below the threshold END;</pre>	B
178	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which the average grade is above a specified threshold (e.g., 75). Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_above_thre shold VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with an average grade above the threshold in subjects_above_thre shold DBMS_OUTPUT.PUT_ LINE('Subjects with Average Grade Above Threshold: '    subjects_above_thre shold); END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	B
179	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which the average grade is above a specified threshold (e.g., 75). Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_above_thre shold VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with an average grade above the threshold in subjects_above_thre shold DBMS_OUTPUT.PUT_ LINE('Subjects with Average Grade Above Threshold: '    subjects_above_thre shold); END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	B
180	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which the average grade is above a specified threshold (e.g., 80). Which of the following code snippets accomplishes this task?	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_above_thre shold VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with an average grade above the threshold in subjects_above_thre shold DBMS_OUTPUT.PUT_ LINE('Subjects with Average Grade Above Threshold: '    subjects_above_thre shold); END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	<pre> DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with an average grade above the threshold END;</pre>	B

181	Given a "student" table with columns 'student_id', 'subject', and 'grade', write a PL/SQL block that uses a cursor to find and print the subjects in which the average grade is below a specified threshold (e.g., 70). Which of the following code snippets accomplishes this task?	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with an average grade below the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subjects_below_thre shold VARCHAR2(4000); BEGIN -- Calculate and store the list of subjects with an average grade below the threshold in subjects_below_thre shold; DBMS_OUTPUT.PUT_ LINE('Subjects with Average Grade Below Threshold: '    subjects_below_thre shold); END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; subject_recordstudent% ROWTYPE; BEGIN -- Find and print subjects with an average grade below the threshold END;</pre>	<pre>DECLARE CURSOR subject_cursor IS SELECT DISTINCT subject FROM student; BEGIN -- Find and print subjects with an average grade below the threshold END;</pre>	B
182	Given a "student" table with columns 'student_id', 'subject', and 'grade',write a PL/SQL block that uses a cursor to find and print the students who have scored the same grade in all subjects. Which of the following code snippets accomplishes this task?	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; students_with_same _grade VARCHAR2(4000); BEGIN -- Calculate and store the list of students with the same grade in all subjects in students_with_same _grade DBMS_OUTPUT.PUT_ LINE('Students with Same Grade in All Subjects: '    students_with_same _grade); END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; student_recordstudent %ROWTYPE; BEGIN -- Find and print students who have scored the same grade in all subjects END;</pre>	<pre>DECLARE CURSOR student_cursor IS SELECT DISTINCT student_id FROM student; BEGIN -- Find and print students who have scored the same grade in all subjects END;</pre>	B	
183	How can concurrent access to shared data lead to data inconsistency in a DBMS?	A) By preventing data updates	B) By enforcing data integrity rules	C) By allowing simultaneous updates	D) By reducing query performance	C
184	How can you remove a package from the database in PL/SQL?	By using the DROP PACKAGE statement	By removing all the procedures and functions from the package body	By using the DELETE PACKAGE statement	By using the TRUNCATE PACKAGE statement	A
185	How many mandatory parts packages has?	1	2	3	4	B
186	In a DBMS, a package is typically used to:	Store and organize tables and views	Group related procedures, functions, and variables	Define user roles and permissions	Execute ad-hoc SQL queries	B
187	In a DBMS, what is a benefit of using stored procedures for business logic?	It allows for dynamic SQL execution.	It centralizes and secures the business logic.	It simplifies data retrieval operations.	It eliminates the need for data validation.	B
188	In a DBMS, what is a database trigger?	A procedure that runs when the database is created	A statement that rolls back database changes	A set of rules for data validation	A piece of code that automatically executes in response to a specific event	D
189	In a DBMS, what is the main purpose of a database trigger?	To enforce referential integrity constraints	To encapsulate business logic for data processing	To automatically generate primary keys	To record changes to data in response to events	D
190	In a DBMS, what is the primary purpose of a "SERIALIZABLE" isolation level?	To maximize concurrency	To prevent transactions from acquiring locks	To ensure that transactions execute in a specific order	To provide the highest level of data consistency and isolation	D
191	In a DBMS, what is the primary purpose of a database package body?	A) To define package variables	B) To provide information about the package's procedures and functions	C) To specify package triggers	D) To implement the actual code for package procedures	D
192	In a DBMS, what is the primary purpose of using "SELECT FOR UPDATE"?	To retrieve data for reporting purposes	To lock rows, preventing other transactions from modifying them	To retrieve data without acquiring locks	To roll back a transaction	B

193	In a DBMS, what is the primary purpose of using a "READ COMMITTED" isolation level?	To minimize data redundancy	To ensure that transactions execute in a specific order	To prevent transactions from acquiring locks	<b>To balance data consistency and concurrency</b>	D
194	In a DBMS, what is the primary purpose of using an "AFTER UPDATE" trigger?	To prevent any updates from occurring	To execute before any update operation occurs	<b>To log changes made to a table after an update</b>	To execute after a row is updated in a table	C
195	In a DBMS, what is the primary purpose of using triggers?	To simplify data retrieval operations	<b>To enforce data integrity constraints</b>	To create temporary tables	To improve query performance	B
196	In a DBMS, what is the purpose of a "SAVE TRANSACTION" statement?	To save a transaction for later execution	To create a new transaction	<b>To save the current state of a transaction as a savepoint</b>	To commit the transaction	C
197	In a DBMS, what is the purpose of a savepoint?	To commit a transaction	To roll back a transaction	<b>To create a point within a transaction to which you can later roll back</b>	To lock a table temporarily	C
198	In a DBMS, what is the purpose of the "COMMIT" statement?	To start a new transaction	<b>To save all changes made within the current transaction</b>	To roll back all changes made within the current transaction	To create a new savepoint	B
199	In a DBMS, what is the purpose of the "ROLLBACK TO SAVEPOINT" statement?	To roll back an entire transaction	To create a new savepoint	<b>To roll back to a specific savepoint within a transaction</b>	To commit a transaction	C
200	In a DBMS, what is the purpose of the "SET TRANSACTION" statement?	<b>To define transaction isolation levels</b>	To start a new transaction	To commit a transaction	To roll back a transaction	A
201	In a DBMS, what is the purpose of the procedure header?	A) To declare the procedure's input and output variables	B) To specify the operations to be performed by the procedure	C) To implement the actual code for the procedure	D) To define the package specification	A
202	In a DBMS, what is the term for a package that only contains the package specification without a package body?	A) A minimal package	B) A comprehensive package	C) A bodiless package	D) A complete package	C
203	In a DBMS, what is the typical process of developing a package?	A) Write the package body first, then the specification	B) Write the package specification first, then the body	C) Develop procedures and triggers independently of packages	D) Develop triggers before procedures	B
204	In a DBMS, which clause is used to specify the action to be taken when a trigger event occurs?	EXECUTE	FOR EACH ROW	<b>WHEN</b>	INSTEAD OF	C
205	In a DBMS, which type of cursor is used to fetch and process one row at a time?	Static cursor	Dynamic cursor	<b>Forward-only cursor</b>	Scrollable cursor	C
206	In a locking-based concurrency control system, what does a "lock" prevent other transactions from doing?	A) Accessing the locked data	B) Aborting the transaction	C) Executing queries	D) Creating database triggers	A
207	In a multi-user DBMS, what issue can occur without proper concurrency control?	A) Faster query execution	B) Data consistency problems	C) Reduced code duplication	D) Increased code modularity	B
208	In a multi-user DBMS, what problem can arise when transactions are executed concurrently without control?	A) Faster data retrieval	B) Enhanced data consistency	C) Reduced code duplication	D) Increased code modularity	B
209	In Oracle PL/SQL, can a package contain both procedures and functions?	Yes, but they must all have the same name.	No, a package can only contain either procedures or functions, not both.	<b>Yes, a package can contain a mix of procedures and functions.</b>	Yes, but they must all be stored in separate packages.	C
210	In Oracle PL/SQL, what is a benefit of using packages over standalone procedures?	Packages are easier to write and debug.	<b>Packages allow you to encapsulate related code and data.</b>	Packages execute faster than standalone procedures.	Packages are not dependent on any database schema.	B
211	In Oracle PL/SQL, which statement is used to raise an exception explicitly within a stored procedure?	RAISE EXCEPTION	SIGNAL	<b>RAISE_APPLICATION_ERROR</b>	THROW	C

212		To create a place in the database for storage of scheme objects, rollback segments, and naming the data files to comprise the tablespace	To create a database trigger	To add/rename data files, to change storage	All of the above	A
	In PL/SQL, the CREATE TABLESPACE is used					
213	In PL/SQL, which of the following statements accurately describes a view?	A view is a virtual table based on the result of a SELECT query.	A view is a physical table that stores data permanently.	A view is a temporary table used for transactional purposes.	A view is a table used only for indexing purposes.	A
214	In PL/SQL, which trigger event is fired when a column value is updated to NULL?	AFTER UPDATE NULL	BEFORE UPDATE NULL	AFTER UPDATE OF column_name	BEFORE UPDATE OF column_name	C
215	In the context of a DBMS, what is the purpose of a stored procedure?	To store data in the database	To define the structure of a table	To encapsulate a series of SQL statements for reuse	To retrieve data from external sources	C
216	In the context of database transactions, what is a deadlock?	A situation where two or more transactions are waiting for each other to release locks, preventing progress.	A situation where a transaction reads uncommitted data.	A situation where a transaction violates integrity constraints.	A situation where a transaction is aborted due to a rollback request.	A
217	In the context of isolation levels in a DBMS, what does the "Read Uncommitted" isolation level allow?	Transactions can read uncommitted changes made by other transactions.	Transactions cannot read any data until all changes are committed.	Transactions can only read committed data.	Transactions can read their own uncommitted changes.	A
218	In the context of isolation levels in a DBMS, what does the "Read Uncommitted" isolation level allow?	Transactions can read uncommitted changes made by other transactions.	Transactions cannot read any data until all changes are committed.	Transactions can only read committed data.	Transactions can read their own uncommitted changes.	A
219	In the context of transactions, what does "serializability" mean?	A) The ability to serialize data	B) The ability to execute transactions concurrently	C) The ability to recover from failures	D) The ability to perform queries efficiently	A
220	OLD and NEW references are not available for table-level triggers.	TRUE	FALSE	Can be true or false	None of the above	A
221	and subprograms.	Yes	No	Can be yes or no	none of the above	A
222	PL/SQL controls the context area through a cursor.	TRUE	FALSE	Can be true or false	All of the above	A
223	the program	Try	Throw	Catch	Exception	D
224	Repeat sequence of statements; _____ end repeat Fill in the correct option :	While Condition	Until variable	Until boolean expression	Until 0	C
225	Repeat sequence of statements; _____ end repeat Fill in the correct option :	While Condition	Until variable	Until boolean expression	Until 0	C
226	Suppose a database system crashes again while recovering from a previous crash. Assume checkpointing is not done by the database either during the transactions or during recovery. Which of the following statements is/are correct?	The same undo and redo list will be used while recovering again.	The database will become inconsistent.	All the transactions that are already undone and redone will not be recovered again.	The system cannot recover any further.	A
227	Temporary stored procedures are stored in _____ database.	Master	Model	User specific	Tempdb	D
228	The _____ Statement is used for creating the package body.	CREATE	CREATE PACKAGE	CREATE BODY	CREATE PACKAGE BODY	D
229	The _____ Statement is used for creating the package body.	CREATE PACKAGE BODY	CREATE	CREAT BODY	CREATE BODY	A
230	The constructs of a procedure, function or a package are _____ .	Variables and Constants	Cursors	Exceptions	All of the above	D
231	The CREATE TRIGGER statement is used to create the trigger. THE _____ clause specifies the table name on which the trigger is to be attached. The _____ specifies that this is an AFTER INSERT trigger.	for insert, on	On, for insert	For, insert	None of the mentioned	B
232	_____ specifies the table name on which the trigger is to be attached. The _____ specifies that this is an AFTER INSERT trigger.	for insert, on	On, for insert	For, insert	None of the mentioned	A
233	The format for compound statement is	Begin ..... end	Begin atomic..... end	Begin ..... repeat	Both Begin ..... end and Begin atomic..... end	D
234	The package specification is the interface to the package.	TRUE	FALSE	Nither TRUE NOR FALSE	none of the above	A

235	The parameters can be passed as default also to the procedures and the functions.	TRUE	FALSE	Neither TRUE NOR FALSE	None of the above	A
236	The property of a schedule that states that the result of executing concurrent transactions is the same as executing them serially is known as:	Consistency	Atomicity	Serializability	Durability	C
237	The technique used to detect and resolve conflicts among concurrent transactions is called:	Two-phase locking	Timestamp ordering	Deadlock detection	Deadlock prevention	C
238	Triggers can be defined on the?	table	view	schema	All of the above	D
239	Triggers can be defined on the?	DOL	DML	Database Operation	All of the above	D
240	What does "recoverability" encompass in the context of transactions and concurrency control?	A) The ability to recover from system failures	B) The ability to execute transactions concurrently	C) The ability to lock database tables	D) The ability to perform efficient queries	A
241	What does DBA stand for in the context of databases?	A) Database Backup Administrator	B) Data Business Analyst	C) Database Architect	D) Database Administrator	D
242	What does the concept of "recoverability" in concurrency control refer to?	A) The ability to lock data	B) The ability to recover from system failures	C) The ability to perform database recovery	D) The ability to execute queries efficiently	B
243	What does the concept of "serializability" in concurrency control refer to?	A) The ability to lock data	B) The ability to execute transactions in parallel	C) The ability to perform database recovery	D) The ability to execute queries efficiently	B
244	What does the term "serializability" imply in the context of transaction execution?	A) Transactions occur in sequence	B) Transactions can execute concurrently	C) Transactions are aborted	D) Transactions are isolated	A
245	What does the term "transaction isolation" refer to in the context of concurrency control?	A) A transaction's lifespan	B) A transaction's ability to update data	C) A transaction's isolation level	D) A transaction's recovery	C
246	What is a "bodiless" package in a DBMS context?	A) A package without a body	B) A package with excessive code	C) A package with only triggers	D) A package with minimal documentation	A
247	What is a "bodiless" package in a DBMS context?	A) A package without a body	B) A package with excessive code	C) A package with only triggers	D) A package with minimal documentation	A
248	What is a "transaction" in the context of a database management system (DBMS)?	A) A data dictionary	B) A single unit of work	C) A database schema	D) A database connection	B
249	What is a common approach to resolving deadlocks in a DBMS?	Rolling back one of the transactions involved in the deadlock	Killing all transactions to release locks	Preventing transactions from acquiring locks	Using deadlock detection and resolution algorithms	D
250	What is a common drawback of "pessimistic" locking in concurrency control systems?	A) Increased code modularity	B) Reduced data consistency	C) Reduced query performance	D) Optimized query execution	C
251	What is a common use of a database trigger in a DBMS?	A) To define package specifications	B) To encapsulate related procedures and functions	C) To monitor and respond to database events	D) To create database packages	C
252	What is a database schema?	A) A collection of tables in a database	B) A diagram representing the structure of a database	C) A set of rules that define the database structure	D) A description of the database structure, including tables, fields, and relationships	D
253	What is a package body in a DBMS?	A) A part of a package that contains package variables	B) A part of a package that specifies the package's procedures and functions	C) A part of a package that defines package triggers	D) A part of a package that implements the actual code for package procedures	D
254	What is a package specification in a DBMS?	A) A part of a package that contains package variables	B) A part of a package that specifies the package's procedures and functions	C) A part of a package that defines package triggers	D) A part of a package that implements the actual code for package procedures	D
255	What is a parameterized stored procedure in a DBMS?	<b>A procedure that accepts parameters and returns a result set</b>	A procedure that uses a parameter as its name	A procedure that cannot accept any input parameters	A procedure that can only accept integer parameters	A
256	What is a transaction in a DBMS?	A database schema	A series of SQL statements	<b>A logical unit of work that is either fully completed or fully undone</b>	A data dictionary	C



257		<pre>CREATE OR REPLACE TRIGGER trg_department_update BEFORE UPDATE OF department ON employee FOR EACH ROW BEGIN     IF :NEW.department =     'Management' THEN         :NEW.salary :=         80000;     END IF; END; /</pre>	<pre>CREATE OR REPLACE TRIGGER trg_department_update BEFORE UPDATE OF department ON employee FOR EACH ROW IF :NEW.department = 'Management' THEN :NEW.salary := 80000; END IF; END; /</pre>	<pre>CREATE OR REPLACE TRI trg_department_update BEFORE UPDATE OF department ON employee FOR EACH BEGIN     IF :NEW.department =     'Management' THEN         :NEW.salary :=         80000;     END IF; END; /</pre>	<pre>CREATE OR REPLACE TRIGGER trg_department_update UPDATE OF department ON employee ROW BEGIN     IF :NEW.department = 'Management' THEN         :NEW.salary := 80000;     END IF; END; /</pre>	A	What is correct a PL/SQL program that create Trigger to update the "salary" of an employee to 80000 if the "department" is changed to 'Management'?
258		<pre>CREATE OR REPLACE PROCEDURE total_salary_by_deptme nt(p_department_name IN VARCHAR2 DEFAULT HR) AS v_total_salary NUMBER; BEGIN     SELECT SUM(salary) INTO v_total_salary FROM employee WHERE department = p_department_name;  DBMS_OUTPUT.PUT_LIN E('Total Salary for Department '    p_department_name    ': '    v_total_salary); EXCEPTION     WHEN OTHERS THEN         DBMS_OUTPUT.PUT_LIN E('An error occurred.');</pre>	<pre>CREATE OR REPLACE PROCEDURE total_salary_by_deptme nt(p_department_name IN VARCHAR2 ) AS v_total_salary NUMBER; BEGIN     SELECT SUM(salary) INTO v_total_salary FROM employee WHERE department = p_department_name;  DBMS_OUTPUT.PUT_LIN E('Total Salary for Department '    p_department_name    ': '    v_total_salary); EXCEPTION     WHEN OTHERS THEN         DBMS_OUTPUT.PUT_LIN E('An error occurred.');</pre>	<pre>CREATE OR REPLACE PROCEDURE total_salary_by_deptme nt(p_department_name IN VARCHAR2 DEFAULT HR) AS v_total_salary NUMBER; BEGIN     SELECT SUM(salary) INTO v_total_salary FROM employee WHERE department = p_department_name;  DBMS_OUTPUT.PUT_LIN E('Total Salary for Department '    p_department_name    ': '    v_total_salary); EXCEPTION     WHEN OTHERS THEN         DBMS_OUTPUT.PUT_LIN E('An error occurred.');</pre>	<pre>CREATE OR REPLACE PROCEDURE total_salary_by_department(DEFAULT HR) AS v_total_salary NUMBER; BEGIN     SELECT SUM(salary) INTO v_total_salary FROM employee WHERE department = p_department_name;  DBMS_OUTPUT.PUT_LINE('Total Salary for Department '    p_department_name    ': '    v_total_salary); EXCEPTION     WHEN OTHERS THEN         DBMS_OUTPUT.PUT_LINE('An error occurred.');</pre>	A	What is correct a procedure that calculates and displays the total salary of employees in a given department. The department name is an optional parameter with a default value of 'HR'.
259	What is one of the advantages of using procedures in a DBMS?	A) Increased code duplication	B) Slower query performance	C) Enhanced security vulnerabilities	D) Reduced code redundancy	D	
260	What is one of the advantages of using triggers in a DBMS?	A) Increased code modularity	B) Reduced control over data changes	C) Enhanced query performance	D) Automated enforcement of data integrity rules	D	
261	<p>What is Output- Insert sample records into the "employee" table</p> <pre>INSERT INTO employee (employee_id, first_name, last_name, department, salary) VALUES (1, 'John', 'Doe', 'HR', 50000);  INSERT INTO employee (employee_id, first_name, last_name, department, salary) VALUES (2, 'Jane', 'Smith', 'Finance', 60000);  INSERT INTO employee (employee_id, first_name, last_name, department, salary) VALUES (3, 'Michael', 'Johnson', 'IT', 70000);  INSERT INTO employee (employee_id, first_name, last_name, department, salary) VALUES (4, 'Mary', 'Agarwal', 'IT', 50000);</pre> <pre>CREATE OR REPLACE PROCEDURE delete_employee_by_id(p_employee_id NUMBER) AS BEGIN     DELETE FROM employee     WHERE employee_id = p_employee_id;      IF SQL%ROWCOUNT &gt; 0 THEN         DBMS_OUTPUT.PUT_LINE('Employee deleted successfully.');</pre>	An error occurred.	Employee ID not found. No employee deleted.	Employee deleted successfully.	EXECUTE delete_employee_by_id(2);	B	
262	What is the ACID property that ensures that transactions are performed correctly and completely?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	A	
263		Stored procedures can return multiple values, while functions can only return a single value	Stored procedures can be executed by users, while functions can only be executed by the database administrator	Stored procedures are used for data manipulation, while functions are used for data retrieval	Stored procedures can be called from within other procedures, while functions cannot	A	
264	What is the key difference between stored procedures and functions in DBMS?						
264	What is the Oracle Error Code for ACCESS_INTO_NULL?	6502	6531	1722	6530	D	
265	<p>What is the output of the following program?</p> <pre>DECLARE A NUMBER :=2; BEGIN FOR I IN 1..3 LOOP A := A*2; END LOOP; DBMS_OUTPUT.PUT_LINE(A); END;</pre>	4	8	16	32	D	
266	What is the primary advantage of using packages in a DBMS?	improved query performance	Enhanced data security	Better code organization and reusability	Simplified database design	D	

267	What is the primary difference between a row-level trigger and a statement-level trigger?	Row-level triggers are executed before statement-level triggers.	Row-level triggers are fired once for each affected row, while statement-level triggers are fired once for each SQL statement.	Statement-level triggers can be defined on tables, whereas row-level triggers cannot.	Row-level triggers can be recursive, while statement-level triggers cannot.	B
268	What is the primary drawback of "optimistic" concurrency control in a DBMS?	A) Increased code modularity	B) Slower query performance	C) Risk of transaction conflicts	D) Enhanced data integrity	C
269	What is the primary goal of concurrency control in a DBMS?	To maximize data redundancy	To minimize database access	To ensure data consistency and integrity in a multi-user environment	To eliminate the need for indexing	C
270	What is the primary goal of concurrency control in a DBMS?	To maximize data redundancy	To minimize database access	<b>To ensure data consistency and integrity in a multi-user environment</b>	To eliminate the need for indexing	C
271	What is the primary purpose of "deadlock detection" mechanisms in a DBMS that uses locking for concurrency control?	A) To prevent transaction conflicts	B) To optimize query performance	C) To eliminate transactions	D) To create database triggers	C
272	What is the primary purpose of a "BEFORE DELETE" trigger in a DBMS?	<b>To execute before any delete operation occurs</b>	To execute after a row is deleted from a table	To prevent any delete operations from taking place	To execute only if a delete operation fails	A
273	What is the primary purpose of a cursor in a stored procedure?	To store the results of a query	<b>To iterate through the records returned by a query</b>	To create a temporary table	To enforce data integrity constraints	B
274	What is the primary purpose of a database package in a DBMS?	A) To define database triggers	B) To encapsulate and group related procedures, functions, and variables	C) To establish database connections	D) To optimize query performance	B
275	What is the primary purpose of a stored procedure in DBMS?	To store and organize data in a database	To retrieve data from the database	To define the structure of the database	To encapsulate a series of database operations	D
276	What is the primary purpose of an "AFTER INSERT" trigger in a DBMS?	To execute before any insert operation occurs	<b>To execute after a new row is inserted into a table</b>	To prevent any insert operations from taking place	To execute only if an insert operation fails	B
277	What is the primary purpose of locking in concurrency control?	A) To eliminate transactions	B) To optimize query performance	C) To manage data access	D) To create database triggers	C
278	What is the primary purpose of transaction management in a database system?	A) To optimize query performance	B) To ensure data consistency	C) To define database triggers	D) To establish database connections	B
279	What is the primary purpose of using packages in a DBMS?	To store data in tables	<b>To organize related procedures, functions, and variables</b>	To manage user permissions	To enforce data integrity constraints	B
280	What is the primary syntax for creating a trigger in a DBMS?	A) CREATE PROCEDURE	B) CREATE FUNCTION	C) CREATE TRIGGER	D) CREATE TABLE	C
281	What is the purpose of "recoverability" in the context of database transactions?	A) To ensure all transactions recover	B) To prevent data recovery issues	C) To recover from system failures	D) To lock database tables	C
282	What is the purpose of declaring a cursor in SQL?	To define a new table in the database	To specify the database connection string	<b>To define a result set for query execution</b>	To create a new user account	C
283	What is the purpose of the "ROLLBACK" statement in a stored procedure?	To commit all changes made within the procedure	<b>To undo all changes made within the procedure</b>	To restart the execution of the procedure from the beginning	To create a new savepoint within the procedure	B

284		A) To define the logical view of the database for users	B) To specify the access controls and security settings for the database	C) To represent the physical storage structure of the database	D) To define the user views and queries for the database	C
	What is the purpose of the internal schema in a database system?					
285		A) To add new records to a table	B) To remove records from a table	C) To combine data from multiple tables based on a related column	D) To modify existing records in a table	C
	What is the purpose of the JOIN operation in a relational database?					
286	What is the purpose of the SAVEPOINT statement in DBMS?	To define the start of a transaction	To create a temporary table	To define a point within a transaction to which you can roll back	To release a lock on a database object	C
287	What is the role of a "transaction log" in a DBMS with respect to concurrency control?	A) To manage database locks	B) To record transaction history	C) To optimize query performance	D) To create database triggers	B
288		A) To design the database	B) To manage database connections	C) To coordinate transaction execution	D) To create database triggers	C
	What is the role of a "transaction manager" in a DBMS?					
289		A) To define package variables	B) To provide information about the package's procedures and functions	C) To specify package triggers	D) To establish database connections	B
	What is the role of a package specification in a database package?					
290	What is the role of the FETCH statement in SQL cursor operations?	It declares a new cursor.	It retrieves rows from the result set and moves the cursor to the next row.	It closes an open cursor.	It defines the structure of a cursor.	B
291	What is the role of the FETCH statement in SQL cursor operations?	It declares a new cursor.	It retrieves rows from the result set and moves the cursor to the next row.	It closes an open cursor.	It defines the structure of a cursor.	B
292		DECLARE my-exception EXCEPTION;	DECLARE EXCEPTION;	DECLARE my-exception;	EXCEPTION;	A
	What is the syntax of User-defined exceptions?					
293		A) Develop triggers first, then procedures	B) Develop procedures and package specification simultaneously	C) Write the package specification first, then the package body	D) Write the package body first, then the specification	C
	What is the typical sequence of steps for developing a package in a DBMS?					
294		A) Develop triggers first, then procedures	B) Develop procedures and package specification simultaneously	C) Write the package specification first, then the package body	D) Write the package body first, then the specification	C
	What is the typical sequence of steps for developing a package in a DBMS?					
295	What part of a procedure in a DBMS is responsible for declaring the input and output variables?	A) Procedure header	B) Procedure specification	C) Procedure body	D) Procedure parameters	B
296	What type of trigger is executed automatically after the triggering event?	A) After Trigger	B) Before Trigger	C) Instead of Trigger	D) Compound Trigger	A
297	What type of trigger is executed automatically before a specific event, such as an INSERT or UPDATE operation?	A) After Trigger	B) Before Trigger	C) Instead of Trigger	D) Compound Trigger	B
298	When executing a stored procedure, what keyword is commonly used to return a result set to the calling application?	RESULT	RESULTSET	OUTPUT	RETURN	C
299	and the data doesn't get _____ back, another transaction tries to access the updated database item.	Rollid	Committed	Aborted	None	A
300	Which ACID property ensures that a transaction's effects on the database are permanent?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	D
301	Which attribute is used to raise exception?	Open	Select	Raise	Try	C
302	Which attribute returns TRUE if an INSERT, UPDATE, or DELETE statement affected one or more rows?	%NOTFOUND	%ISOPEN	%ROWCOUNT	%FOUND	D
303	Which clause is used to create trigger on a view?	BEFORE	INSTEAD OF	AFTER	None of the above	B
304	Which component of a package in DBMS defines the interface and public entities?	Package body	Package Signature	Package Constructor	Package specification	D
305	Which concurrency control technique allows multiple transactions to read data simultaneously but enforces write locks to prevent data conflicts?	Two-Phase Locking (2PL)	Time-stamp Ordering	Multi-Version Concurrency Control (MVCC)	Optimistic Concurrency Control	A
306	Which concurrency control technique allows multiple transactions to read data simultaneously but enforces write locks to prevent data conflicts?	Two-Phase Locking (2PL)	Time-stamp Ordering	Multi-Version Concurrency Control (MVCC)	Optimistic Concurrency Control	A
307	Which cursor attribute can be used to determine the total number of rows returned by a cursor in PL/SQL?	%ROWCOUNT	%FOUND	%ISOPEN	%NOTFOUND	A

308	Which database system component is responsible for managing transactions and ensuring data integrity?	A) Database schema	B) Data dictionary	C) Concurrency control manager	D) Query optimizer	C
309	Which isolation level allows only committed data to be read?	Read Uncommitted	Read committed	Serializable	Read Update	B
310	Which isolation level allows the highest concurrency but may result in non-repeatable reads?	Read Uncommitted	Read Committed	Repeatable Read	Serializable	A
311	Which isolation level provides the highest level of data consistency but can lead to reduced concurrency?	Read Uncommitted	Read Committed	Repeatable Read	Serializable	D
312	Which isolation level provides the highest level of data consistency but can lead to reduced concurrency?	Read Uncommitted	Read Committed	Repeatable Read	<b>Serializable</b>	D
313	Which keyword is used to create a new package body in PL/SQL?	BODY	IMPLEMENTATION	DEFINE	CREATE	A
314	Which keyword is used to define an exception handler in PL/SQL?	EXCEPTION	CATCH	TRY	HANDLE	A
315	Which of the following actions can be performed by a "BEFORE INSERT" trigger in a DBMS?	Rollback the entire transaction	<b>Modify data before it is inserted</b>	Create a new table	Terminate the DBMS session	B
316	Which of the following are benefits of Triggers?	Generating some derived column values automatically	Enforcing referential integrity	Event logging and storing information on table access	All of the above	D
317	Which of the following are the advantages of PL/SQL Packages?	Modularity	Easier Application Design	Information Hiding	All of the above	D
318	Which of the following clause does not comes in the syntax while raising an exception?	DECLARE	WHEN	CLOSE	END	C
319	Which of the following database languages is used to define the structure and organization of a database?	A) Data Manipulation Language (DML)	B) Data Definition Language (DDL)	C) Data Control Language (DCL)	D) Data Query Language (DQL)	B
320	Which of the following describes the ACID properties of transactions?	Atomicity, Consistency, Isolation, Durability	Atomicity, Consistency, Isolation, Dileasing	Atomicity, Consistency, Isolate, Durability	Atomic, Consistency, Isolation, Durability	A
321	Which of the following is a benefit of using procedures in a DBMS?	A) Increased code duplication	B) Reduced security	C) Improved code organization and maintenance	D) Limited code reuse	C
322	Which of the following is a benefit of using the "SAVEPOINT" statement in a DBMS?	It allows you to commit a transaction.	It allows you to roll back the entire database.	It enables you to create nested transactions.	<b>It provides a way to roll back to a specific point within a transaction.</b>	D
323	Which of the following is a benefit of using the "SAVEPOINT" statement in a DBMS?	It allows you to commit a transaction.	It allows you to roll back the entire database.	It enables you to create nested transactions.	<b>It provides a way to roll back to a specific point within a transaction.</b>	D
324	Which of the following is a characteristic of an "AUTOCOMMIT" transaction mode in a DBMS?	Each SQL statement is treated as a separate transaction.	<b>Transactions are automatically committed after each SQL statement.</b>	Transactions cannot be rolled back.	It is a read-only mode.	B
325	Which of the following is a commonly used technique for concurrency control in a DBMS?	A) Optimistic concurrency control	B) Serializability control	C) Slower query performance	D) Increased code duplication	B
326	Which of the following is a drawback of strict two-phase locking (S2PL)?	Increased concurrency	Increased deadlock probability	Reduced consistency	Reduced durability	B
327	Which of the following is a property of a transaction in a database system?	A) Increased code duplication	B) Slower query performance	C) Atomicity	D) Reduced code redundancy	C
328	Which of the following is an advantage of using packages in a DBMS?	A) Limited code organization	B) Increased code redundancy	C) Enhanced code isolation	D) Improved code modularity and reusability	D
329	Which of the following is an advantage of using procedures in a DBMS?	A) Increased code redundancy	B) Slower execution of queries	C) Improved security vulnerabilities	D) Code reusability and maintainability	D
330	Which of the following is an advantage of using stored procedures in a DBMS?	Reduced security	Increased data redundancy	<b>Improved performance</b>	Decreased data integrity	C
331	Which of the following is an advantage of using triggers in a DBMS?	A) Limited code organization	B) Increased code redundancy	C) Enhanced code isolation	D) Improved code modularity and reusability	D
332	Which of the following is an example of a parameterized trigger in a DBMS?	A trigger that fires after any insert operation	A trigger that fires after a specific date and time	A trigger that fires when a specific condition is met	<b>A trigger that accepts parameters passed from the calling application</b>	D
333	Which of the following is an example of an optimistic concurrency control technique in a DBMS?	Two-Phase Locking (2PL)	<b>Multi-Version Concurrency Control (MVCC)</b>	Time-stamp Ordering	Rollback Segments	B
334	Which of the following is not a level of data abstraction in a database system?	A) Physical level	B) Logical level	C) External level	D) Semantic level	D
335	Which of the following is NOT a property of a transaction in DBMS?	Atomicity	Consistency	Durability	Isolation	B
336	Which of the following is not a type of trigger in DBMS?	Insert trigger	Update trigger	Delete trigger	Search trigger	D

337	Which of the following is not an advantage of trigger?	Various column values are automatically generated by triggers	Maintains the integrity of referential	Tables are replicated asynchronously	Validating transactions and preventing them from being invalid	C
338	Which of the following is NOT an Oracle-supported trigger?	BEFORE	DURING	AFTER	INSTEAD OF	B
339	Which of the following is the correct format for if statement?	If boolean expression then statement or compound statement elseif boolean expression then statement or compound statement else statement or compound statement end if	If boolean expression then statement or compound statement elseif boolean expression then statement or compound statement else statement or compound statement end if	If boolean expression then statement or compound statement elif boolean expression then statement or compound statement else statement or compound statement end if	If boolean expression then statement or compound statement else statement or compound statement else statement or compound statement end if	A
340	Which of the following is true about compound triggers?	They can only be defined for tables, not views	They are fired once for each row affected by the triggering event	They cannot contain any SQL statements	They are not supported in DBMS	B
341	Which of the following is true about recursive triggers?	They are triggered by other triggers	They can only be fired once per event	They can cause an infinite loop if not handled properly	They are not supported in DBMS	C
342	Which of the following is true about stored procedures?	They can only return a single scalar value.	They can contain control-of-flow statements like IF and LOOP.	They cannot accept input parameters.	They are always automatically executed when the database starts.	B
343	Which of the following is TRUE about User-defined exceptions?	Users can explicitly raise an exception by using a RAISE statement	RAISE_APPLICATION_ERROR can be used to raise a user-defined exception explicitly	both 1 and 2	None of the above	C
344	Which of the following is used to input the entry and give the result in a variable in a procedure?	Put and get	Get and put	Out and In	In and out	D
345	Which of the following makes the transaction permanent in the database?	View	Commit	Rollback	Flashback	B
346	Which of the following specifies when the trigger will be executed?	BEFORE	AFTER	INSTEAD OF	All of the above	D
347	Which of the following statements best defines database recovery in DBMS?	The process of restoring data from backup tapes	The process of ensuring that the database remains secure	The process of restoring the database to a consistent state after a failure	The process of recovering deleted data from the Recycle Bin	C
348	Which of the following statements is true about First Normal Form (1NF)?	A) It allows for multivalued dependencies.	B) It allows for partial dependencies.	C) It eliminates repeating groups and ensures atomicity of data.	D) It enforces referential integrity constraints.	C
349	Which of the following statements is true about stored procedures?	Stored procedures cannot have input parameters	Stored procedures cannot return values	Stored procedures can be reused and shared by multiple applications	Stored procedures can only be executed by the database administrator	C
350	Which of the following statements is true about the Two-tier architecture?	A) It allows for better scalability than the Three-tier architecture.	B) It is easier to maintain and modify compared to the Three-tier architecture.	C) It requires less network traffic than the Three-tier architecture.	D) It provides better security and data isolation compared to the Three-tier architecture.	C
351	Which of the following statements is true regarding stored procedures?	Stored procedures always return a single value.	Stored procedures are not allowed to contain conditional statements.	<b>Stored procedures are precompiled and stored in the database for reuse.</b>	Stored procedures can only be executed by database administrators.	C
352	Which package lets PL/SQL programs read and write operating system (OS) text files?	UTL_HTTP	UTL_FILE	UTL_SMTP	UTL_FMT	B
353	Which part of a procedure in a DBMS is responsible for specifying the operations to be performed?	A) Procedure header	B) Procedure specification	C) Procedure body	D) Procedure parameters	C

354	Which property of a transaction ensures that it does not interfere with other transactions while executing?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	C
355	Which property of a transaction ensures that it does not interfere with other transactions while executing?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	C
356	Which property of a transaction ensures that it does not violate integrity constraints?	Isolation	Atomicity	Consistency	Durability	C
357	Which property of a transaction ensures that it either completes in its entirety or has no effect at all?	A) Atomicity	B) Optimistic concurrency control	C) Slower query performance	D) Data redundancy	A
358	Which property of a transaction ensures that the database remains in a consistent state after transaction execution?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	B
359	Which recovery technique uses backward recovery to undo the changes made by a failed transaction?	Undo logging	Redo logging	Deferred update	Immediate update	A
360	Which specifies the column name that will be updated?	For col_name	ON col_name	OF col_name	WHEN col_name	C
361	Which type of database constraint ensures that a foreign key value matches a primary key value in another table?	A) Unique constraint	B) Primary key constraint	C) Foreign key constraint	D) Not null constraint	C
362	Which type of database trigger in SQL is executed before the triggering event occurs?	AFTER trigger	INSTEAD OF trigger	BEFORE trigger	FOR EACH ROW trigger	C
363	Which type of error occurs when the database crashes while a transaction is being executed?	System error	Media error	Transaction error	Operator error	A
364	Which type of trigger in a DBMS can be used to prevent changes to a table?	BEFORE trigger	AFTER trigger	<b>INSTEAD OF trigger</b>	FOR EACH ROW trigger	C
365	Which type of trigger in a DBMS is fired after a triggering event and can be used for auditing purposes?	BEFORE trigger	<b>AFTER trigger</b>	INSTEAD OF trigger	FOR EACH ROW trigger	B
366	Which type of view in PL/SQL allows you to update data directly through the view?	Materialized View	Read-Only View	Updatable View	Join View	C
367	Why is "concurrency control" important in a multi-user database environment?	A) To increase query performance	B) To ensure data consistency	C) To eliminate transactions	D) To optimize database storage	B
368	Why is "concurrency" a concern in a multi-user DBMS environment?	A) To simplify data retrieval	B) To ensure data consistency	C) To reduce query performance	D) To create redundant data	B
369	Why is concurrency control needed in a database management system (DBMS)?	A) To increase data redundancy	B) To slow down query execution	C) To ensure data consistency	D) To reduce code duplication	C
370	Write a PL/SQL function named 'get_student_average_grade' that takes a student ID as input and returns the average grade of the specified student across all subjects. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_average_grade(student_id NUMBER) RETURN NUMBER IS BEGIN     -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_average_grade(student_id NUMBER) IS BEGIN     -- Procedure logic     here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_average_grade(student_id NUMBER, avg_grade OUT NUMBER) IS BEGIN     -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_average_grade(student_id NUMBER) RETURN TABLE IS BEGIN     -- Function logic here END;</pre>	A
371	Write a PL/SQL function named 'get_student_count_by_subject' that takes a subject as input and returns the count of students enrolled in that subject. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_count_by_subject(subject VARCHAR2) RETURN NUMBER IS BEGIN     -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_count_by_subject(subject VARCHAR2) RETURN NUMBER IS student_count NUMBER; BEGIN     -- Function logic     here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_count_by_subject(subject VARCHAR2, student_count OUT NUMBER) IS BEGIN     -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_count_by_subject(subject VARCHAR2) IS BEGIN     -- Procedure logic here END;</pre>	a
372	Write a PL/SQL function named 'get_student_grade_in_subject' that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN     -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) IS BEGIN     -- Procedure logic     here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN     -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN     -- Function logic here END;</pre>	A

373	Write a PL/SQL function named 'get_student_grade_in_subject' that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A
374	Write a PL/SQL function named 'get_student_grade_in_subject' that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A
375	Write a PL/SQL function named 'get_student_grade_in_subject' that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A
376	Write a PL/SQL function named 'get_student_grade_in_subject' that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A
377	Write a PL/SQL function named 'get_student_grade' that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade(student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade(student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade(student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A

378	Write a PL/SQL function named 'get_student_info' that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;</pre>	C
379	Write a PL/SQL function named 'get_student_info' that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;</pre>	C
380	Write a PL/SQL function named 'get_student_info' that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;</pre>	C
381	Write a PL/SQL function named 'get_student_info' that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;</pre>	C



382	Write a PL/SQL function named 'get_student_info' that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;</pre>	C
383	Write a PL/SQL function named 'get_student_subject_scores' that takes a student ID as input and returns a cursor containing the subject and grade for all subjects in which the student is enrolled. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_subject_scores(student_id NUMBER) RETURN CURSOR IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subject_scores(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_subject_scores(student_id NUMBER, subject_scores OUT SYS_REFCURSOR) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_subject_scores(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;</pre>	C
384	Write a PL/SQL function named 'get_student_subjects' that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER, subject OUT VARCHAR2) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER, subject OUT VARCHAR2) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	A
385	Write a PL/SQL function named 'get_student_subjects' that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER, subject OUT VARCHAR2) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER, subject OUT VARCHAR2) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	A
386	Write a PL/SQL function named 'get_student_subjects' that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER, subject OUT VARCHAR2) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER, subject OUT VARCHAR2) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	A
387	Write a PL/SQL function named 'get_student_subjects' that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	<pre>CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER, subject OUT VARCHAR2) IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER, subject OUT VARCHAR2) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER) IS BEGIN -- Procedure logic here END;</pre>	A

388	Write a PL/SQL function named 'get_student_subjects' that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_subjects(stu dent_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_subjects (student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(st udent_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER) IS BEGIN -- Procedure logic here END;	A
389	Write a PL/SQL function named 'get_subjects_by_student' that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_subjects_by_student (student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_subjects_by_stu dent(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_subjects_by_studen t(student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_subjects_by_student(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	A
390	Write a PL/SQL procedure named 'add_student_subject' that takes a student ID, subject, and grade as input and adds a new subject enrollment record for the specified student in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
391	Write a PL/SQL procedure named 'add_student_subject' that takes a student ID, subject, and grade as input and adds a new subject enrollment record for the specified student in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
392	Write a PL/SQL procedure named 'calculate_avg_grade' that takes a student ID as input and calculates the average grade of that student across all subjects. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE PROCEDURE calculate_avg_grade(stu dent_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE calculate_avg_grade( student_id NUMBER, avg_grade OUT NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE calculate_avg_grade(stu dent_id NUMBER) AS avg_grade NUMBER; BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION calculate_avg_grade(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	B
393	Write a PL/SQL procedure named 'calculate_student_average' that calculates the average grade for a specific student identified by their 'student_id' and stores it in a variable. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE PROCEDURE calculate_student_aver age(student_id NUM BER, avg_grade OUT NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE calculate_student_av erage(student_id NUMBER, avg_grade OUT NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION calculate_student_avera ge(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION calculate_student_average(student_id NUMBER, avg_grade OUT NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	B

394	Write a PL/SQL procedure named 'delete_student_record' that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(st udent_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_recor d(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
395	Write a PL/SQL procedure named 'delete_student_record' that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(st udent_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_recor d(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
396	Write a PL/SQL procedure named 'delete_student_record' that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(st udent_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_recor d(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
397	Write a PL/SQL procedure named 'delete_student_record' that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(st udent_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_recor d(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
398	Write a PL/SQL procedure named 'delete_student_record' that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(st udent_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_recor d(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
399	Write a PL/SQL procedure named 'delete_student' that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student(student_ id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student(studi nt_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student(student _id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
400	Write a PL/SQL procedure named 'enroll_student_in_subject' that takes a student ID, subject, and grade as input and inserts a new enrollment record for the specified student in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION enroll_student_in_subje ct( student_id NUMBER, subject VARCHAR2, grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE enroll_student_in_su bject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE enroll_student_in_subj ect( student_id NUMBER, subject VARCHAR2, grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE enroll_student_in_subje ct( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B

401	<p>Write a PL/SQL procedure named 'enroll_student' that takes student details (name, age, subject, grade) as input and inserts a new record into the "student" table. Which of the following code snippets correctly defines this procedure?</p>	<pre>CREATE OR REPLACE FUNCTION enroll_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE enroll_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE enroll_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) AS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE enroll_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;</pre>	b
402	<p>Write a PL/SQL procedure named 'insert_student_record' that takes student details (name, age, subject, grade) as input and inserts a new record into the "student" table. Additionally, it should return the newly generated student ID. Which of the following code snippets correctly defines this procedure?</p>	<pre>CREATE OR REPLACE FUNCTION insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE insert_student_recor d( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) AS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS student_id NUMBER; BEGIN -- Function logic here END;</pre>	B
403	<p>Write a PL/SQL procedure named 'insert_student_record' that takes student details (name, age, subject, grade) as input and inserts a new record into the "student" table. Additionally, it should return the newly generated student ID. Which of the following code snippets correctly defines this procedure?</p>	<pre>CREATE OR REPLACE FUNCTION insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE insert_student_recor d( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) AS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS student_id NUMBER; BEGIN -- Function logic here END;</pre>	B
404	<p>Write a PL/SQL procedure named 'insert_student' that takes student details (name, age, subject, grade) as input and inserts a new record into the "student" table. Additionally, it should return the newly generated student ID. Which of the following code snippets correctly defines this procedure?</p>	<pre>CREATE OR REPLACE FUNCTION insert_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE insert_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) IS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE PROCEDURE insert_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) AS BEGIN -- Procedure logic here END;</pre>	<pre>CREATE OR REPLACE FUNCTION insert_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS student_id NUMBER; BEGIN -- Function logic here END;</pre>	B

405	Write a PL/SQL procedure named 'update_student_grade' that takes a student ID, subject, and a new grade as input and updates the grade of the specified student for the given subject. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
406	Write a PL/SQL procedure named 'update_student_record' that takes a student ID as input and updates the student's name, age, and grade in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_record( student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_record( student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
407	Write a PL/SQL procedure named 'update_student_record' that takes a student ID as input and updates the student's name, age, and grade in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_record( student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_record( student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
408	Write a PL/SQL procedure named 'update_student_record' that takes a student ID as input and updates the student's name, age, and grade in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_record( student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_record( student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
409	Write a PL/SQL procedure named 'update_student_record' that takes a student ID as input and updates the student's name, age, and grade in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_record( student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_record( student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record( student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
410	Write a PL/SQL procedure named 'update_student_subject_grade' that takes a student ID, subject, and a new grade as input and updates the grade of the specified student for the given subject. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_subject_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_subject_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_subject_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_subject_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
411	You have a stored procedure named UpdateEmployeeSalary that accepts an employee ID and a salary value as parameters and updates the employee's salary in the database. Which SQL statement would you use to execute this stored procedure with employee ID 101 and a new salary of 55000?	EXEC UpdateEmployeeSalary 101, 55000;	CALL UpdateEmployeeSalary( 101, 55000);	RUN UpdateEmployeeSalary( 101, 55000);	UPDATE EmployeeSalary(101, 55000);	A

412	You have a stored procedure that calculates the average salary of employees in a specific department. Which SQL statement do you use to execute this stored procedure and retrieve the result?	<b>EXEC</b> <b>GetAverageSalaryForDepartment 101;</b>	<b>CALL</b> GetAverageSalaryForDepartment(101);	<b>EXEC</b> GetAverageSalaryForDepartment @DepartmentID = 101;	<b>EXEC</b> GetAverageSalaryForDepartment @DepartmentID = 101, @Result = OUTPUT;	A
413	want to create a trigger that updates the "LastPurchaseDate" column to the current date whenever a new purchase is made by a customer. What type of trigger should you create?	<b>AFTER INSERT</b>	<b>BEFORE INSERT</b>	<b>AFTER UPDATE</b>	<b>INSTEAD OF INSERT</b>	A
414	You have a table named "Inventory" with a "Quantity" column. You want to create a trigger that automatically updates the "Quantity" column to zero when a product is sold. What type of trigger should you create?	<b>AFTER UPDATE</b>	<b>BEFORE UPDATE</b>	<b>AFTER INSERT</b>	<b>INSTEAD OF UPDATE</b>	A
415	You want to create a stored procedure that inserts a new customer record into the "Customers" table. The customer's name, email, and phone number will be passed as parameters. Which SQL statement creates this stored procedure?	<b>CREATE PROCEDURE</b> <b>InsertCustomer</b> <b>@Name VARCHAR(50),</b> <b>@Email VARCHAR(100),</b> <b>@Phone VARCHAR(20)</b> <b>AS</b> <b>BEGIN</b> <b>INSERT INTO Customers</b> <b>(Name, Email, Phone)</b> <b>VALUES (@Name,</b> <b>@Email, @Phone)</b> <b>END</b>	<b>CREATE PROCEDURE</b> <b>InsertCustomer AS</b> <b>BEGIN INSERT INTO</b> <b>Customers (Name,</b> <b>Email, Phone)</b> <b>VALUES (@Name,</b> <b>@Email, @Phone)</b> <b>END</b>	<b>CREATE PROCEDURE</b> <b>InsertCustomer</b> <b>@CustomerData</b> <b>VARCHAR(MAX) AS</b> <b>BEGIN INSERT INTO</b> <b>Customers (Name,</b> <b>Email, Phone) VALUES</b> <b>(@CustomerData) END</b>	<b>CREATE PROCEDURE</b> InsertCustomer @Name VARCHAR(50), @Email VARCHAR(100), @Phone VARCHAR(20) AS BEGIN INSERT INTO Customers (Name, Email, Phone) VALUES (@Name, @Email, @Phone) END	A

S R	Questions	Option 1	Option 2	Option 3	Option 4	Corr ect Ans wer
1	Which of the following is NOT a benefit of using transactions?	Data integrity	High availability	Data consistency	Data durability	B
2	A transaction that violates the consistency property is considered to be:	Serializable	Inconsistent	Isolate	Error	B
3	Can you change the parameter values of a cursor after it has been declared and opened?	Yes, parameter values can be modified at any time.	No, parameter values are fixed once the cursor is declared and opened.	Parameter values can only be changed during cursor declaration.	Cursors cannot have parameter values.	B
4	Can you declare a cursor without specifying the SELECT statement immediately?	No, a SELECT statement must always be specified.	Yes, a SELECT statement can be added later in the code.	Cursors cannot be declared in PL/SQL.	Cursors are automatically generated in PL/SQL.	A
5	Can you declare multiple cursors with the same name but different parameters in the same PL/SQL block?	Yes, as long as the cursor names are unique.	No, cursor names must be unique regardless of the parameters.	Multiple cursors are not allowed in the same PL/SQL block.	Cursors with parameters cannot have the same name.	B
6	Can you declare multiple cursors within the same PL/SQL block? If so, how do you differentiate them?	No, only one cursor is allowed per block.	Yes, multiple cursors can be declared, and they are differentiated by their data types.	Yes, multiple cursors can be declared, and they are differentiated by their names.	Multiple cursors cannot be used in PL/SQL.	C
7	Can you fetch data from a cursor into individual variables or into a record type? Explain.	Data can only be fetched into individual variables.	Data can only be fetched into a record type.	Data can be fetched into both individual variables and a record type.	Data cannot be fetched from a cursor.	C
8	Can you nest a Cursor FOR Loop inside another Cursor FOR Loop? If so, why might you do so?	No, nesting Cursor FOR Loops is not allowed.	Yes, you can nest Cursor FOR Loops to perform complex data processing and handle related data hierarchies.	Cursor FOR Loops can only be used individually, not nested.	Nesting Cursor FOR Loops results in performance issues.	B

9	Can you use a Cursor FOR Loop to update or delete records in a database table? Explain.	No, Cursor FOR Loops are read-only.	Yes, Cursor FOR Loops can update or delete records using the UPDATE and DELETE statements.	Cursor FOR Loops can only insert records, not update or delete them.	Cursor FOR Loops can only be used for reporting purposes.	B
10	Describe the differences between an implicit cursor and an explicit cursor in PL/SQL.	Implicit cursors are used for data modeling, while explicit cursors are used for data manipulation.	Implicit cursors are automatically created for DML statements, while explicit cursors are user-defined.	Implicit cursors are used for database connections, while explicit cursors are used for loop control.	Implicit cursors are used for hardware design, while explicit cursors are used for web development.	B
11	Describe the purpose of PL/SQL collections, and provide examples of their types.	PL/SQL collections are used for defining variables.	PL/SQL collections are used for database connections.	PL/SQL collections are used for storing multiple values of the same data type.	PL/SQL collections are used for creating triggers.	C
12	Explain how cursor parameters can be used to create dynamic cursors.	Cursor parameters have no role in creating dynamic cursors.	By allowing parameterization of the WHERE clause in the cursor's SELECT statement, you can create dynamic cursors that retrieve specific data based on different criteria.	Cursor parameters can only be used with static cursors.	Cursor parameters can be used to create triggers.	B
13	Explain the concept of triggers in a database context. How are they used in PL/SQL?	Triggers are used for creating web applications.	Triggers are used for hardware design.	Triggers are used for automatically executing PL/SQL code in response to database events.	Triggers are used for data modeling.	C
14	Explain the difference between declaring a cursor and opening a cursor.	Declaring a cursor retrieves data; opening a cursor defines its structure.	Declaring a cursor defines its structure; opening a cursor retrieves data.	Declaring a cursor and opening a cursor are the same.	Declaring a cursor is not a PL/SQL concept.	B
15	Explain the importance of transactions in PL/SQL and how they are managed.	Transactions are used for web development.	Transactions are used for data modeling.	Transactions ensure data consistency and are managed using COMMIT and ROLLBACK statements.	Transactions are not supported in PL/SQL.	C



1 6	Explain the purpose of a PL/SQL package and its components.	PL/SQL packages are used for web development.	PL/SQL packages are used for encapsulating procedures and functions.	PL/SQL packages are used for data modeling.	PL/SQL packages are used for hardware design.	B
1 7	How can you pass parameters to a PL/SQL procedure or function?	Parameters are passed using the CALL statement.	Parameters are not supported in PL/SQL.	Parameters are passed as input and output variables.	Parameters are passed using the DECLARE statement.	C
1 8	How can you resolve a deadlock in a database system?	By terminating one of the transactions involved in the deadlock.	By rolling back all transactions involved in the deadlock.	By increasing the isolation level.	Deadlocks cannot be resolved.	A
1 9	How do you create and manipulate PL/SQL associative arrays (index-by tables)?	Associative arrays are created using the ARRAY keyword.	Associative arrays are created using the INDEX keyword.	Associative arrays are not supported in PL/SQL.	Associative arrays are created using the TYPE keyword.	D
2 0	How do you declare a cursor, and what are the required components?	Cursors are automatically declared in PL/SQL.	Cursors are declared using the DECLARE CURSOR statement and require a SELECT statement.	Cursors are declared using the DECLARE keyword.	Cursors are declared using the OPEN statement.	B
2 1	How do you declare a variable in PL/SQL, and what are the data types supported for variables?	Variables are declared using the DECLARE keyword, and PL/SQL supports only one data type.	Variables are declared using the VAR keyword, and PL/SQL supports multiple data types.	Variables are declared using the VARIABLE keyword, and PL/SQL supports multiple data types.	Variables are not supported in PL/SQL.	B
2 2	How do you define and use PL/SQL records and record types?	Records are used for creating tables in PL/SQL.	Records are defined using the DECLARE RECORD statement.	Records are used to hold data in a structured format.	Records are not supported in PL/SQL.	C
2 3	How do you ensure that you've fetched all available data from a cursor?	By using the CLOSE statement.	By using the OPEN statement.	By checking the cursor attribute %NOTFOUND.	Cursors automatically fetch all available data.	C
2 4	How do you handle database connections and transactions in PL/SQL?	Database connections and transactions are automatically managed by the PL/SQL engine.	Database connections and transactions are not supported in PL/SQL.	Database connections are established using the CONNECT statement, and transactions are managed using COMMIT and ROLLBACK statements.	Database connections are established using the DECLARE statement.	C

25	How do you handle exceptions in PL/SQL? Provide an example.	Exceptions are handled using the IF-ELSE statement.	Exceptions are handled using the TRY-CATCH block.	Exceptions are handled using the EXCEPTION block.	Exceptions are not supported in PL/SQL.	C
26	How do you handle exceptions that may occur when working with cursors that have parameters?	By using the FETCH statement.	By ignoring exceptions and proceeding with the cursor operations.	By using exception handling techniques such as WHEN OTHERS and specific exception handlers for cursor-related errors.	Cursors with parameters do not raise exceptions.	C
27	How do you name a cursor, and what are some best practices for naming conventions?	Cursors are named automatically.	Cursors can be named using any random string.	Cursors should have meaningful names following naming conventions such as prefixing with CUR_.	Cursors cannot have names in PL/SQL.	C
28	How do you open a cursor to make it ready for data retrieval?	Use the DECLARE CURSOR statement.	Use the OPEN CURSOR statement.	Use the FETCH statement.	Cursors are automatically opened in PL/SQL.	B
29	How do you pass values to the cursor parameters when opening the cursor?	Use the FETCH statement to provide parameter values.	Use the SET PARAMETER statement.	Use a separate ASSIGN statement to assign values to parameters before opening the cursor.	Cursor parameters do not require values when opening.	C
30	How does a Cursor FOR Loop handle exceptions compared to explicit cursor processing?	Cursor FOR Loops do not support exception handling.	Cursor FOR Loops handle exceptions more gracefully by providing built-in error handling mechanisms.	Exception handling in Cursor FOR Loops is the same as in explicit cursor processing.	Cursor FOR Loops handle exceptions less efficiently than explicit cursors.	B
31	How does a cursor with parameters differ from a cursor without parameters in terms of flexibility?	Cursors with parameters are less flexible.	Cursors with parameters are more flexible because they can retrieve data based on varying conditions.	There is no difference in flexibility between the two types of cursors.	Cursors with parameters are slower.	B
32	How is the declaration of a cursor different from a regular SQL query?	Cursors cannot be used to retrieve data.	Cursors have a SELECT statement, while regular SQL queries are standalone.	Regular SQL queries cannot be used in PL/SQL.	There is no difference; they are the same.	B

33	In a multi-user database system, what does optimistic concurrency control aim to achieve?	It aims to prevent transactions from running concurrently.	It aims to avoid blocking and allow transactions to proceed concurrently, only checking for conflicts at the end.	It aims to lock all records to avoid conflicts.	It aims to roll back all transactions.	B
34	In database recovery, what is the difference between forward recovery and backward recovery?	Forward recovery restores the database to a previous state, while backward recovery recovers the database to its current state.	Forward recovery is the same as database backup, while backward recovery restores the database to a previous state.	Forward recovery involves log analysis, while backward recovery involves restoring database backups.	There is no difference; the terms are used interchangeably.	C
35	In database recovery, what is the purpose of a database log file?	To store user data.	To record changes made to the database for recovery purposes.	To create database backups.	To store database metadata.	B
36	In order to undo the work of transaction after last commit which one should be used?	View	Commit	Rollback	Flashback	C
37	In SQL, what is the role of the ROLLBACK statement?	To save pending changes.	To begin a new transaction.	To undo all changes made during the current transaction.	To release locks on database records.	C
38	Is it necessary to declare a cursor inside a PL/SQL block, or can it be declared globally in a package?	Cursors can only be declared globally.	Cursors can only be declared inside a PL/SQL block.	Cursors can be declared both globally and inside a PL/SQL block.	Cursors are not supported in PL/SQL.	C
39	What are database triggers, and when might you use them in PL/SQL?	Database triggers are used for hardware design.	Database triggers are used for declaring variables.	Database triggers are used to automatically respond to database events and can be used for auditing or enforcing business rules.	Database triggers are used for creating tables.	C
40	What are some common use cases for using cursor parameters in PL/SQL?	Cursor parameters are rarely used in practice.	Common use cases include generating reports with different filter criteria, processing data based on user inputs, and customizing data	Cursor parameters are mainly used for database administration tasks.	Cursor parameters are only used in triggers.	B

			retrieval based on changing conditions.			
4 1	What are the advantages of using explicit cursors over implicit cursors in PL/SQL?	Explicit cursors are faster in performance.	Implicit cursors are more flexible.	Explicit cursors are easier to use and provide more control.	Implicit cursors are automatically managed by the database.	C
4 2	What are the benefits of using PL/SQL for database programming compared to using SQL alone?	PL/SQL allows for creating web applications.	PL/SQL provides procedural capabilities for better control and encapsulation of logic in the database.	PL/SQL is used for hardware design.	PL/SQL is primarily used for data modeling.	B
4 3	What does the isolation level READ COMMITTED mean?	Reads data as it was when the transaction started.	Reads uncommitted changes made by other transactions.	Prevents any reads until the transaction is committed.	Reads data from committed transactions only.	A
4 4	What does the SAVEPOINT statement do in SQL?	Marks a point in a transaction to be rolled back to later.	Commits the transaction.	Opens a new transaction.	Locks the database.	A
4 5	What happens when you fetch data from a cursor that has no more rows to retrieve?	An error occurs.	The cursor is automatically closed.	The cursor remains open and ready for the next fetch.	Cursors always have more rows to retrieve.	A
4 6	What is a cursor in PL/SQL, and why is it used?	A cursor is a database table.	A cursor is used for looping through query results.	A cursor is a data type in PL/SQL.	A cursor is used for creating triggers.	B
4 7	What is a database checkpoint?	A physical location where the database is stored.	A marker indicating the point in time up to which transactions are considered safe and can be recovered.	A log file containing SQL statements.	A password for accessing the database.	B
4 8	What is a database lock in the context of concurrency control?	A mechanism to block all database transactions.	A mechanism to prevent data corruption.	A mechanism to prevent multiple transactions from accessing the same data simultaneously.	A mechanism to unlock databases.	C

49	What is a database restore operation?	A process that erases all data from the database.	A process that removes the database log files.	A process that brings a database back to a previous state by applying database backups and log files.	A process that upgrades the database to a new version.	C
50	What is a database transaction?	A single SQL statement.	A sequence of related SQL statements that are executed as a unit.	A database schema.	A database table.	B
51	What is a deadlock in the context of concurrency control?	A situation where a transaction is rolled back.	A situation where two or more transactions are waiting for each other to release locks.	A situation where a transaction is terminated.	A situation where a transaction is committed.	B
52	What is a distributed transaction in database management?	A transaction that involves multiple databases.	A transaction that is committed automatically.	A transaction with a large number of SQL statements.	A transaction without a COMMIT.	A
53	What is a full database backup?	A backup that includes only a subset of the database.	A backup that includes all the data and structures in the database.	A backup that contains only log files.	A backup that is encrypted for security.	B
54	What is a nested transaction in SQL?	A transaction inside another transaction.	A transaction without any nested SQL statements.	A transaction that cannot be rolled back.	A transaction with a SAVEPOINT.	A
55	What is a PL/SQL function, and how does it differ from a procedure?	A function is used for controlling database transactions.	A function is used for encapsulating reusable logic and returns a value.	A procedure is used for data modeling.	A procedure is used for creating tables.	B
56	What is concurrency control in database systems?	Managing multiple database transactions simultaneously.	Controlling access to the database using passwords.	Rolling back transactions in case of errors.	Creating indexes for database tables.	A
57	What is cursor positioning, and how does it relate to fetching data?	Cursor positioning determines the cursor's name.	Cursor positioning is the process of opening a cursor.	Cursor positioning refers to the current position of the cursor relative to the result set, affecting the next fetch operation.	Cursor positioning is not relevant in PL/SQL.	C

58	What is database recovery in the context of database management systems?	Backing up the database to prevent data loss.	The process of restoring a database to a previous state after a failure.	Increasing the database size to accommodate more data.	Encrypting database files for security.	B
59	What is dynamic SQL, and why might you use it in PL/SQL?	Dynamic SQL is used for creating triggers in PL/SQL.	Dynamic SQL allows you to generate and execute SQL statements at runtime.	Dynamic SQL is used for web development.	Dynamic SQL is used for hardware design.	B
60	What is PL/SQL, and how does it differ from SQL?	PL/SQL is a markup language for web development.	PL/SQL is a procedural extension of SQL.	PL/SQL is a data modeling language.	PL/SQL is a hardware description language.	B
61	What is the ACID property in the context of database transactions?	Atomicity, Consistency, Isolation, Durability	Aggregation, Continuity, Integrity, Durability	Affinity, Consistency, Isolation, Durability	Atomicity, Cancellation, Isolation, Division	A
62	What is the advantage of using a Cursor FOR Loop over traditional cursor processing?	Cursor FOR Loops are slower than traditional cursors.	Cursor FOR Loops offer less control.	Cursor FOR Loops simplify cursor processing by handling cursor declaration, opening, fetching, and closing automatically.	Cursor FOR Loops are not recommended in PL/SQL.	C
63	What is the benefit of using cursor parameters when working with data retrieval?	Cursor parameters make cursor declaration simpler.	Cursor parameters allow for dynamic queries and customization of data retrieval based on varying conditions.	Cursor parameters improve cursor performance.	Cursor parameters are not useful in PL/SQL.	B
64	What is the default behavior of a Cursor FOR Loop if there are no rows to process?	It raises an error.	It skips the loop and continues with the next statement.	It automatically exits the loop.	It waits for rows to be available.	C
65	What is the difference between a PL/SQL procedure and a PL/SQL function?	A procedure returns a value, while a function does not.	A procedure does not return a value, while a function does.	A procedure and a function are the same.	A procedure and a function are not supported in PL/SQL.	B
66	What is the opposite of a COMMIT statement in SQL?	ROLLBACK	BEGIN TRANSACTION	SAVEPOINT	LOCK	A
67	What is the primary drawback of using pessimistic concurrency	It can lead to data inconsistency.	It can result in excessive locking and reduced concurrency.	It is slower than optimistic concurrency control.	It requires frequent COMMIT statements.	B

	control in a database system?					
68	What is the purpose of a differential backup in database recovery?	To recover the database to a specific point in time.	To restore only the data that has changed since the last full backup, reducing the recovery time.	To make a copy of the entire database.	To compress the database backup files.	B
69	What is the purpose of declaring a cursor in PL/SQL?	To insert data into a table.	To define a variable in PL/SQL.	To retrieve and manipulate query results in a controlled manner.	To create a trigger in PL/SQL.	C
70	What is the purpose of fetching data from a cursor in PL/SQL?	To insert data into a table.	To define a variable in PL/SQL.	To retrieve and manipulate query results row by row.	To create a trigger in PL/SQL.	C
71	What is the purpose of the COMMIT statement in SQL?	To roll back a transaction.	To save all pending changes permanently to the database.	To lock database records.	To create a new transaction.	B
72	What is the purpose of the FOR loop in PL/SQL, and how is it used?	The FOR loop is used for declaring variables.	The FOR loop is used for defining exceptions.	The FOR loop is used for iterative processing.	The FOR loop is used for database connections.	C
73	What is the purpose of the LOCK TABLE statement in SQL?	To unlock a table.	To create a new table.	To specify the locking mode for a table explicitly.	To commit a transaction.	C
74	What is the purpose of the WHEN OTHERS exception handler in PL/SQL?	It is used for declaring variables.	It is used for defining custom exceptions.	It is used to catch and handle unexpected exceptions.	It is used for database connections.	C
75	What is the role of a database backup in recovery?	It serves as a temporary storage location.	It records changes made to the database.	It provides a copy of the database that can be used to restore data in case of data loss or corruption.	It ensures data consistency during concurrent transactions.	C
76	What is the role of the FETCH statement in cursor processing?	It defines the cursor's name.	It specifies the number of rows to fetch.	It retrieves rows from the cursor into variables or records.	It opens the cursor for data retrieval.	C
77	What is the significance of the %ROWTYPE attribute when declaring a cursor?	It defines the cursor's name.	It specifies the number of rows the cursor can fetch.	It defines the structure of the result set the cursor will hold.	It determines the data type of cursor variables.	C

7 8	What is the significance of the recovery point objective (RPO) in database recovery planning?	It defines the maximum number of recovery points allowed.	It specifies the desired point in time to which a database should be recovered after a failure.	It defines the number of database logs to retain.	It measures the database's performance.	B
7 9	Which ACID property ensures that a transaction is completed in its entirety or not at all?	Atomicity	Consistency	Isolation	Durability	A
8 0	Which concurrency control technique allows conflicts to be detected and resolved only at the commit time?	Validation-based protocol	Timestamp ordering	Two-phase locking	Three-phase locking	A
8 1	Which database recovery model allows for point-in-time recovery to any arbitrary moment?	Simple recovery model	Full recovery model	Bulk-logged recovery model	Incremental recovery model	B
8 2	Which isolation level in SQL provides the highest level of isolation but can lead to concurrency issues?	READ COMMITTED	SERIALIZABLE	READ UNCOMMITTED	REPEATABLE READ	B
8 3	Which of the following is not a concurrency control mechanism in DBMS?	Locking	Timestamp ordering	Multiversion concurrency control	Rollback and recovery	D
8 4	Which of the following recovery techniques is based on maintaining multiple copies of the database at different points in time?	Replication	Deferred update	Redo logging	Undo logging	A
8 5	Which SQL statement is used to set a SAVEPOINT within a transaction?	BEGIN SAVEPOINT	SAVEPOINT	SET SAVEPOINT	CREATE SAVEPOINT	B
8 6	Which technique allows concurrent transactions to access different parts of a database without conflicts?	Pessimistic concurrency control	Optimistic concurrency control	Exclusive locking	Distributed transactions	B



87	[ON table_name] specifies the name of the table associated with the trigger.	Yes	No	Can be yes or no	None of the above	A
88	_____ are stored programs, which are automatically executed or fired when some events occur.	Procedure	Triggers	Collection	Transaction	B
89	A _____ consists of a sequence of query and/or update statements.	Transaction	Commit	Rollback	Flashback	A
90	A _____ is a special kind of a store procedure that executes in response to certain action on the table like insertion, deletion or updation of data.	Procedures	Triggers	Functions	None of the mentioned	B
91	A stored procedure in SQL is a _____	Block of functions	Group of Transact-SQL statements compiled into a single execution plan.	Group of distinct SQL statements.	None of the mentioned	B
92	A view is actually a?	composition of a table	decomposition of a table	associated to a table	None of the above	A
93	All objects placed in the specification are called _____ objects.	private	protected	public	None of the above	B
94	Any subprogram not in the package specification but coded in the package body is called a _____ object.	protected	private	self	public	B
95	Boyce-Codd Normal Form (BCNF) is an extension of which normal form?	A) First Normal Form (1NF)	B) Second Normal Form (2NF)	C) Third Normal Form (3NF)	D) Fourth Normal Form (4NF)	B

9 6	<p>Consider the following action:</p> <p>TRANSACTION..... Commit; ROLLBACK; What does Rollback do?</p>	Undoes the transactions before commit	Clears all transactions	Redoes the transactions before commit	No action	D
9 7	<p>Consider the following cursor declaration in SQL: DECLARE cursor1 CURSOR FOR SELECT FirstName, LastName FROM Employees WHERE Department = 'Sales' If you want to open and fetch rows from this cursor, what SQL statement should you use next?</p>	FETCH NEXT FROM cursor1;	<b>OPEN cursor1;</b>	CLOSE cursor1;	DECLARE cursor1 CURSOR FOR ...	B
9 8	<p>Consider the following database schedule with two transactions, T1 and T2 S = r2(X); r1(X); r2(Y); w1(X); r1(Y); w2(X); a1; a2 where ri(Z) denotes a read operation by transaction Ti on a variable Z, wi(Z) denotes a write operation by Ti on a variable Z and ai denotes an abort by transaction Ti .</p> <p>Which one of the following statements about the above schedule is TRUE?</p>	S is non-recoverable	S is recoverable, but has a cascading abort	S does not have a cascading abort	S is strict	C

99	<p>Consider the following stored procedure in SQL:</p> <pre> CREATE PROCEDURE CalculateTotalPrice @ProductID INT, @Quantity INT AS BEGIN     DECLARE @Price     DECIMAL(10, 2)     SELECT @Price = UnitPrice     FROM Products WHERE     ProductID = @ProductID     PRINT 'Total Price: ' +     CAST(@Price * @Quantity     AS VARCHAR) END </pre> <p>If you call this stored procedure with @ProductID = 101 and @Quantity = 5, what will be printed?</p>	Total Price: 505	<b>Total Price: 25</b>	Total Price: 101	Total Price: 5	B
100	<p>Create function dept count(dept_name varchar(20))</p> <pre> begin declare d count integer; select count(*) into d count from instructor where instructor.dept_name= dept_name return d count; end </pre> <p>Find the error in the the above statement.</p>	Return type missing	Dept_name is mismatched	Reference relation is not mentioned	All of the mentioned	A

1 0 1	<pre> CREATE OR REPLACE FUNCTION calculate_gpa(   student_id NUMBER ) RETURN NUMBER IS   total_points NUMBER := 0;   total_credits NUMBER := 0;   gpa NUMBER; BEGIN   -- Calculate GPA for a   student   -- Assume the grade scale:   A=4, B=3, C=2, D=1, F=0   -- Credits per subject: 3   credits   -- GPA = (Total Points) /   (Total Credits)   RETURN gpa; END; </pre>	Deletes students by age.	Updates student's name.	Calculates a student's GPA based on their grades.	Retrieves students in a subject above average.	C
1 0 2	<pre> CREATE OR REPLACE FUNCTION calculate_student_gpa(   student_id NUMBER ) RETURN NUMBER IS   gpa NUMBER; BEGIN   -- Calculate GPA for a   student based on their   grades and credit hours   RETURN gpa; END; </pre>	Deletes students by age.	Updates student's name.	Calculates a student's GPA based on their grades and credit hours.	Transfers students from one batch to another.	C

1 0 3	<pre> CREATE OR REPLACE FUNCTION calculate_subject_average(   subject_name VARCHAR2 ) RETURN NUMBER IS   avg_grade NUMBER; BEGIN   SELECT AVG(grade) INTO avg_grade   FROM student   WHERE subject = subject_name;   RETURN avg_grade; END; </pre>	Deletes students by age.	Updates student information.	Calculates the average grade in a specific subject.	Returns a list of students with above-average grades in a subject.	C
1 0 4	<pre> CREATE OR REPLACE FUNCTION calculate_total_students RETURN NUMBER IS   total_students NUMBER; BEGIN   SELECT COUNT(*) INTO total_students   FROM student;   RETURN total_students; END; </pre>	Deletes students by name.	Updates student information.	Calculates the total number of students.	Returns a list of students by age range.	C
1 0 5	<pre> CREATE OR REPLACE FUNCTION count_students_by_age(   age NUMBER ) RETURN NUMBER IS   student_count NUMBER; BEGIN   SELECT COUNT(*) INTO student_count   FROM student   WHERE age = age;   RETURN student_count; END; </pre>	Retrieves students by age.	Returns the total count of students by age.	Enrolls students in multiple subjects.	Deletes students by subject.	B

106	<pre> CREATE OR REPLACE FUNCTION count_students_in_subject(   subject_name VARCHAR2 ) RETURN NUMBER IS   student_count NUMBER; BEGIN   SELECT COUNT(*) INTO student_count   FROM student   WHERE subject = subject_name;   RETURN student_count; END;</pre>	Retrieves students by subject count.	Returns a list of students with the maximum number of subjects.	Deletes students by age.	Updates a student's batch.	A
107	<pre> CREATE OR REPLACE FUNCTION find_students_with_max_age RETURN SYS_REFCURSOR IS   students_cursor SYS_REFCURSOR; BEGIN   OPEN students_cursor FOR   SELECT student_id   FROM student   WHERE age = (SELECT MAX(age) FROM student);   RETURN students_cursor; END;</pre>	Retrieves students with the lowest age.	Returns a list of students with the highest age.	Deletes students by age.	Awards scholarships to deserving students.	B

108	<pre> CREATE OR REPLACE FUNCTION find_students_with_max_su bjects RETURN SYS_REFCURSOR IS   students_cursor SYS_REFCURSOR; BEGIN   OPEN students_cursor FOR     SELECT student_id     FROM (       SELECT student_id, COUNT(DISTINCT subject) AS subject_count   FROM student   GROUP BY student_id   ORDER BY subject_count DESC     )     WHERE ROWNUM = 1;   RETURN students_cursor; END; </pre>	Retrieves students with the highest grades.	Returns a list of subjects with above-average grades.	Enrolls students in multiple subjects.	Deletes students by subject.	B
-----	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------	-------------------------------------------------------	----------------------------------------	------------------------------	---

109	<pre> CREATE OR REPLACE FUNCTION find_students_with_subject _count(   subject_count NUMBER ) RETURN SYS_REFCURSOR IS   students_cursor SYS_REFCURSOR; BEGIN   OPEN students_cursor FOR     SELECT student_id     FROM (       SELECT student_id,       COUNT(DISTINCT subject)       AS subject_count       FROM student       GROUP BY student_id     )     WHERE subject_count = subject_count;   RETURN students_cursor; END; </pre>	Retrieves students by subject count.	Returns a list of students with a specific subject count.	Deletes students by age.	Updates a student's subject and grade.	A
-----	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------	-----------------------------------------------------------	--------------------------	----------------------------------------	---



110	<pre> CREATE OR REPLACE FUNCTION find_subjects_with_min_students RETURN SYS_REFCURSOR IS   subjects_cursor SYS_REFCURSOR; BEGIN   OPEN subjects_cursor FOR   SELECT subject   FROM (     SELECT subject,     COUNT(*) AS student_count   FROM student   GROUP BY subject   ORDER BY   student_count ASC   )   WHERE ROWNUM = 1;   RETURN subjects_cursor; END; </pre>	Retrieves subjects with the highest grades.	Returns a list of subjects with the lowest number of students.	Enrolls students in multiple subjects.	Deletes students by subject.	B
111	<pre> CREATE OR REPLACE FUNCTION get_highest_grade_by_subject(   subject_name VARCHAR2 ) RETURN NUMBER IS   highest_grade NUMBER; BEGIN   SELECT MAX(grade) INTO   highest_grade   FROM student   WHERE subject =   subject_name;   RETURN highest_grade; END; </pre>	Returns the highest grade of all students.	Enrolls students in a subject.	Deletes students by subject.	Returns the highest grade in a specific subject.	D

1 1 2	<pre> CREATE OR REPLACE FUNCTION get_student_count_by_subj ect(     subject_name VARCHAR2 ) RETURN NUMBER IS     student_count NUMBER; BEGIN     SELECT COUNT(*) INTO student_count     FROM student     WHERE subject = subject_name;     RETURN student_count; END; </pre>	Deletes a student by subject.	Returns the count of students in a specific subject.	Enrolls a student in a subject.	Updates the student's subject.	B
1 1 3	<pre> CREATE OR REPLACE FUNCTION get_students_above_averag e(     subject_name VARCHAR2 ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR     SELECT student_id     FROM student     WHERE subject = subject_name AND grade &gt; (SELECT AVG(grade) FROM student WHERE subject = subject_name);     RETURN students_cursor; END; </pre>	Retrieves all students.	Returns a list of students with above-average grades in a specific subject.	Enrolls students in a subject.	Deletes students by subject.	B

1 1 4	<pre> CREATE OR REPLACE FUNCTION get_students_by_age_range (     min_age NUMBER,     max_age NUMBER ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR     SELECT student_id     FROM student     WHERE age BETWEEN min_age AND max_age;     RETURN students_cursor; END; </pre>	Retrieves students by age range.	Returns a list of students with above-average grades in a specific subject.	Enrolls students in multiple subjects.	Deletes students by subject.	A
1 1 5	<pre> CREATE OR REPLACE FUNCTION get_students_by_grade_range(     min_grade NUMBER,     max_grade NUMBER ) RETURN SYS_REFCURSOR IS     students_cursor SYS_REFCURSOR; BEGIN     OPEN students_cursor FOR     SELECT student_id     FROM student     WHERE grade BETWEEN min_grade AND max_grade;     RETURN students_cursor; END; </pre>	Retrieves students by grade range.	Returns the total count of students by grade range.	Awards scholarships to deserving students.	Transfers students from one batch to another.	A

1 1 6	CREATE OR REPLACE FUNCTION get_students_by_subject_and_grade( subject_name VARCHAR2, min_grade NUMBER ) RETURN SYS_REFCURSOR IS students_cursor SYS_REFCURSOR; BEGIN OPEN students_cursor FOR SELECT student_id FROM student WHERE subject = subject_name AND grade >= min_grade; RETURN students_cursor; END;	Retrieves students by age range.	Returns a list of students with a specific subject and minimum grade.	Enrolls students in multiple subjects.	Deletes students by subject.	B
1 1 7	CREATE OR REPLACE FUNCTION get_students_in_batch RETURN SYS_REFCURSOR IS students_cursor SYS_REFCURSOR; BEGIN OPEN students_cursor FOR SELECT student_id FROM student WHERE batch_id IS NOT NULL; RETURN students_cursor; END;	Retrieves students in a specific batch.	Returns a list of students with the highest age.	Enrolls students in multiple subjects.	Deletes students by subject.	A

1 1 8	<pre> CREATE OR REPLACE FUNCTION get_students_in_subject_ab ove_average(   subject_name VARCHAR2 ) RETURN SYS_REFCURSOR IS   students_cursor SYS_REFCURSOR; BEGIN   OPEN students_cursor FOR   SELECT student_id   FROM student   WHERE subject = subject_name   AND grade &gt; (SELECT AVG(grade) FROM student WHERE subject = subject_name);   RETURN students_cursor; END; </pre>	Retrieves all students.	Returns a list of students with above-average grades in a specific subject.	Enrolls students in a subject.	Updates student information.	B
1 1 9	<pre> CREATE OR REPLACE FUNCTION get_students_with_highest_ grade RETURN SYS_REFCURSOR IS   students_cursor SYS_REFCURSOR; BEGIN   OPEN students_cursor FOR   SELECT student_id   FROM student   WHERE grade = (SELECT MAX(grade) FROM student);   RETURN students_cursor; END; </pre>	Retrieves students with the lowest grade.	Returns students with the highest grade.	Enrolls students in a batch.	Deletes students by age.	B

1 2 0	CREATE OR REPLACE FUNCTION get_students_with_lowest_ grade RETURN SYS_REFCURSOR IS students_cursor SYS_REFCURSOR; BEGIN OPEN students_cursor FOR SELECT student_id FROM student WHERE grade = (SELECT MIN(grade) FROM student); RETURN students_cursor; END;	Retrieves students with the highest grade.	Returns students with the lowest grade.	Enrolls students in a batch.	Updates student information.	B
1 2 1	CREATE OR REPLACE FUNCTION get_students_with_max_gra des_per_subject RETURN SYS_REFCURSOR IS students_cursor SYS_REFCURSOR; BEGIN OPEN students_cursor FOR SELECT student_id FROM ( SELECT student_id, MAX(grade) AS max_grade FROM student GROUP BY student_id ); RETURN students_cursor; END;	Retrieves students with the subject wise lowest grades .	Returns students with the subject wise highest grades.	Enrolls students in multiple subjects.	Deletes students by subject.	B

1 2 2	<pre> CREATE OR REPLACE FUNCTION get_subjects_by_student(   student_id NUMBER ) RETURN SYS_REFCURSOR IS   subjects_cursor SYS_REFCURSOR; BEGIN   OPEN subjects_cursor FOR   SELECT DISTINCT subject   FROM student   WHERE student_id = student_id;   RETURN subjects_cursor; END; </pre>	Deletes a student record.	Updates a student's information.	Returns a list of distinct subjects for a specific student.	Enrolls students in a subject.	C
1 2 3	<pre> CREATE OR REPLACE FUNCTION get_subjects_with_max_grades RETURN SYS_REFCURSOR IS   subjects_cursor SYS_REFCURSOR; BEGIN   OPEN subjects_cursor FOR   SELECT subject   FROM (     SELECT subject,     MAX(grade) AS max_grade     FROM student     GROUP BY subject   );   RETURN subjects_cursor; END; </pre>	Retrieves subjects with the highest grades.	Returns a list of subjects with above-average grades.	Enrolls students in multiple subjects.	Deletes students by subject.	A

1 2 4	<pre> CREATE OR REPLACE FUNCTION get_subjects_with_students _above_average RETURN SYS_REFCURSOR IS   subjects_cursor SYS_REFCURSOR; BEGIN   OPEN subjects_cursor FOR   SELECT subject   FROM (     SELECT subject,     AVG(grade) AS avg_grade     FROM student     GROUP BY subject   )   WHERE avg_grade &gt; (SELECT AVG(grade) FROM student);   RETURN subjects_cursor; END; </pre>	Retrieves subjects with students above average grades.	Returns a list of subjects with students below average grades.	Enrolls students in multiple subjects.	Deletes students by subject.	A
1 2 5	<pre> CREATE OR REPLACE PROCEDURE archive_student_records IS BEGIN   -- Write logic to archive old student records END; </pre>	Deletes students by age.	Updates student information.	Archives old student records.	Awards scholarships to deserving students.	C
1 2 6	<pre> CREATE OR REPLACE PROCEDURE assign_student_grade(   student_id NUMBER,   subject_name VARCHAR2,   grade NUMBER ) IS BEGIN   -- Write logic to assign a specific grade to a student </pre>	Deletes students by age.	Updates a student's name.	Assigns a specific grade to a student.	Promotes students to the next grade.	C



	in a subject END;					
1 2 7	CREATE OR REPLACE PROCEDURE assign_student_subjects_and grades( student_id NUMBER, subject_grades SYS.ODCINUMBERLIST ) IS BEGIN -- Write logic to assign subjects and grades to a student END;	Deletes students by age.	Updates student information.	Assigns subjects and grades to a student.	Promotes students to the next grade.	C
1 2 8	CREATE OR REPLACE PROCEDURE delete_student( student_id NUMBER ) IS BEGIN DELETE FROM student WHERE student_id = student_id; COMMIT; END;	Retrieve student details by ID.	Update student information.	Enroll a new student.	Delete a student record.	D

1 2 9	CREATE OR REPLACE PROCEDURE delete_students_by_age( max_age NUMBER ) IS BEGIN DELETE FROM student WHERE age > max_age; COMMIT; END;	Deletes students by name.	Updates student age.	Deletes students older than a specified age.	Calculates students' GPA.	C
1 3 0	CREATE OR REPLACE PROCEDURE delete_students_by_batch( batch_id NUMBER ) IS BEGIN DELETE FROM student WHERE batch_id = batch_id; COMMIT; END;	Deletes students by age.	Updates student information.	Deletes students by batch.	Assigns subjects and grades to a student.	C
1 3 1	CREATE OR REPLACE PROCEDURE delete_students_by_subject ( subject_name VARCHAR2 ) IS BEGIN DELETE FROM student WHERE subject = subject_name; COMMIT; END;	Deletes students by age.	Updates student information.	Deletes students by subject.	Enrolls students in multiple subjects.	C

1 3 2	<pre> CREATE OR REPLACE PROCEDURE enroll_student(   student_name VARCHAR2,   student_age NUMBER,   subject VARCHAR2,   student_grade NUMBER ) IS BEGIN   INSERT INTO student(student_id, name, age, subject, grade)  VALUES(student_sequence. NEXTVAL, student_name, student_age, subject, student_grade);   COMMIT; END; </pre>	Deletes a student record.	Updates a student's information.	Enrolls a new student with provided details.	Returns the count of students in a specific subject.	C
1 3 3	<pre> CREATE OR REPLACE PROCEDURE enroll_student_in_multiple_ subjects(   student_name VARCHAR2,   student_age NUMBER,   subjects VARCHAR2,   student_grades VARCHAR2 ) IS BEGIN   -- Write logic to enroll a student in multiple subjects with corresponding grades END; </pre>	Retrieve student details by ID.	Update student information.	Enroll students in multiple subjects.	Delete students by subject.	C

1 3 4	<pre> CREATE OR REPLACE PROCEDURE increase_student_grades(   grade_increase NUMBER ) IS BEGIN   UPDATE student   SET grade = grade +   grade_increase;   COMMIT; END; </pre>	Deletes students by age.	Updates student information.	Increases student grades.	Awards scholarships to deserving students.	C
1 3 5	<pre> CREATE OR REPLACE PROCEDURE print_student_details(   student_id NUMBER ) IS   student_name   VARCHAR2(100);   student_age NUMBER;   subject_name   VARCHAR2(50);   student_grade NUMBER; BEGIN   SELECT name, age, subject,   grade   INTO student_name,   student_age, subject_name,   student_grade   FROM student   WHERE student_id =   student_id;    DBMS_OUTPUT.PUT_LINE('S tudent ID: '    student_id);    DBMS_OUTPUT.PUT_LINE(' Name: '    student_name); </pre>	Deletes a student record.	Updates student information.	Prints details of a student by ID.	Returns the count of students in a specific subject.	C

	<pre>DBMS_OUTPUT.PUT_LINE(' Age: '    student_age);</pre>					
1 3 6	<pre>CREATE OR REPLACE PROCEDURE print_student_transcript(   student_id NUMBER ) IS BEGIN   -- Write logic to print the   student's transcript END;</pre>	Deletes students by age.	Updates student information.	Prints a student's transcript.	Returns students with the lowest grade.	C

1 3 7	<pre> CREATE OR REPLACE PROCEDURE print_subjects_and_grades(   student_id NUMBER ) IS BEGIN   FOR rec IN (SELECT subject, grade FROM student WHERE student_id = student_id) LOOP      DBMS_OUTPUT.PUT_LINE('S ubject: '    rec.subject    ', Grade: '    rec.grade);   END LOOP; END; </pre>	Deletes a student record.	Updates student information.	Prints subjects and grades of a student by ID.	Returns a list of subjects in which a student is enrolled.	C
1 3 8	<pre> CREATE OR REPLACE PROCEDURE transfer_student(   student_id NUMBER,   new_batch_id NUMBER ) IS BEGIN   -- Write logic to transfer a student to a new batch END; </pre>	Deletes students by age.	Updates student information.	Transfers a student to a new batch.	Calculates the average grade in a subject.	C

1 3 9	<pre> CREATE OR REPLACE PROCEDURE update_student_age(   student_id NUMBER,   new_age NUMBER ) IS BEGIN   UPDATE student   SET age = new_age   WHERE student_id = student_id;   COMMIT; END; / </pre>	Deletes a student record.	Updates a student's age.	Enrolls a new student.	Returns the count of students in a specific subject.	B
1 4 0	<pre> CREATE OR REPLACE PROCEDURE update_student_name(   student_id NUMBER,   new_name VARCHAR2 ) IS BEGIN   UPDATE student   SET name = new_name   WHERE student_id = student_id;   COMMIT; END; </pre>	Deletes students by age.	Updates a student's name.	Enrolls a new student.	Returns the count of students in a specific subject.	B
1 4 1	<pre> CREATE OR REPLACE PROCEDURE update_student_subject_and_grade(   student_id NUMBER,   subject_name VARCHAR2,   new_grade NUMBER ) IS BEGIN   -- Write logic to update a </pre>	Deletes a student's subject and grade.	Updates a student's subject and grade.	Enrolls a new student.	Returns a student's subject and grade.	B

	student's subject and grade END;					
1 4 2	<p>Create procedure dept_count proc(in dept name varchar(20), out d count integer) begin select count(*) into d count from instructor where instructor.dept name= dept count proc.dept name end Which of the following is used to call the procedure given above ?</p>	Declare d_count integer;	<p>Declare d_count integer; call dept_count proc('Physics', d_count);</p>	<p>Declare d_count integer; call dept_count proc('Physics');</p>	<p>Declare d_count; call dept_count proc('Physics', d_count);</p>	B
1 4 3	<p>Declare out of classroom seats condition</p> <p>DECLARE exit handler FOR OUT OF classroom seats BEGIN SEQUENCE OF statements END The above statements are used for</p>	Calling procedures	Handling Exception	Handling procedures	All of the mentioned	B



1 4 4	Exceptions are raised by the database server automatically whenever there is any internal database error.	Yes	No	1 or 2	All of the above	A
1 4 5	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to calculate the average grade for each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   FOR subject_rec IN     subject_cursor LOOP     -- Calculate and print the     average grade for each subject     END LOOP; END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; avg_grade NUMBER; BEGIN   FOR subject_rec IN     subject_cursor LOOP     -- Calculate and store the     average grade for each subject     in avg_grade      DBMS_OUTPUT.PUT_LINE('Subj ect: '    subject_rec.subject    ', Avg Grade: '    avg_grade);     END LOOP; END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subject_recordstudent%ROWTY PE; BEGIN   FOR subject_rec IN     subject_cursor LOOP     -- Calculate and print the     average grade for each subject     END LOOP; END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT     subject       FROM student; BEGIN   FOR subject_rec IN     subject_cursor LOOP     -- Calculate and print     the average grade for     each subject     END LOOP; END;</pre>	B

1 4 6	<p>Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to fetch all students who have a grade greater than or equal to 90 in the subject "Mathematics."</p> <p>Which of the following code snippets accomplishes this task?</p>	<pre> DECLARE   CURSOR high_math_grades IS     SELECT student_id     FROM student     WHERE subject = 'Mathematics' AND grade &gt;= 90; BEGIN   FOR rec IN high_math_grades   LOOP      DBMS_OUTPUT.PUT_LINE('Stud ent ID: '    rec.student_id);   END LOOP; END; </pre>	<pre> DECLARE   CURSOR high_math_grades IS     SELECT student_id     FROM student     WHERE subject = 'Mathematics' AND grade &gt;= 90; student_recordstudent%ROWT YPE; BEGIN   OPEN high_math_grades;   LOOP     FETCH high_math_grades     INTO student_record;     EXIT WHEN high_math_grades%NOTFOUN D;      DBMS_OUTPUT.PUT_LINE('Stud ent ID: '    student_record.student_id);   END LOOP;   CLOSE high_math_grades; END; </pre>	<pre> DECLARE   CURSOR high_math_grades IS     SELECT student_id     FROM student     WHERE subject = 'Mathematics' AND grade &gt;= 90; student_id NUMBER; BEGIN   OPEN high_math_grades;   LOOP     FETCH high_math_grades     INTO student_id;     EXIT WHEN high_math_grades%NOTFOUN D;      DBMS_OUTPUT.PUT_LINE('Stud ent ID: '    student_id);   END LOOP;   CLOSE high_math_grades; END; </pre>	<pre> DECLARE   CURSOR high_math_grades(studen t_id NUMBER) IS   SELECT student_id   FROM student   WHERE subject = 'Mathematics' AND grade &gt;= 90; BEGIN   FOR rec IN high_math_grades(90)   LOOP      DBMS_OUTPUT.PUT_LINE ('Student ID: '    rec.student_id);   END LOOP; END; </pre>	B
-------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1 4 7	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the student with the highest grade for each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   FOR subject_rec IN     subject_cursor LOOP     -- Find and print the student     with the highest grade for each     subject     END LOOP; END; </pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; highest_grade NUMBER; BEGIN   FOR subject_rec IN     subject_cursor LOOP     -- Calculate and store the     highest grade for each subject     in highest_grade      DBMS_OUTPUT.PUT_LINE('Subj     ect: '    subject_rec.subject        ', Highest Grade: '        highest_grade);     END LOOP; END; </pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; student_recordstudent%ROWT YPE; BEGIN   FOR subject_rec IN     subject_cursor LOOP     -- Find and print the student     with the highest grade for each     subject     END LOOP; END; </pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT     subject     FROM student; BEGIN   FOR subject_rec IN     subject_cursor LOOP     -- Find and print the     student with the highest     grade for each subject     END LOOP; END; </pre>	A
1 4 8	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have improved their grades in at least one subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who   have improved their grades END; </pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; improved_students VARCHAR2(4000); BEGIN   -- Calculate and store the list   of improved students in   improved_students    DBMS_OUTPUT.PUT_LINE('Impr   oved Students: '      improved_students); END; </pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who   have improved their grades END; </pre>	<pre> DECLARE   CURSOR student_cursor   IS     SELECT DISTINCT     student_id     FROM student; BEGIN   -- Find and print students   who have improved their   grades END; </pre>	B

1 4 9	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who   have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; not_improved_students VARCHAR2(4000); BEGIN   -- Calculate and store the list of students who have not improved their grades in not_improved_students  DBMS_OUTPUT.PUT_LINE('Students Who Have Not Improved Their Grades: '    not_improved_students); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT student_id   FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	B
1 5 0	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; not_improved_students VARCHAR2(4000); BEGIN   -- Calculate and store the list of students who have not improved their grades in not_improved_students  DBMS_OUTPUT.PUT_LINE('Students Who Have Not Improved Their Grades: '    not_improved_students); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT student_id   FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	B

1 5 1	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who   have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; not_improved_students VARCHAR2(4000); BEGIN   -- Calculate and store the list of students who have not improved their grades in not_improved_students  DBMS_OUTPUT.PUT_LINE('Stud ents Who Have Not Improved Their Grades: '    not_improved_students); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT student_id   FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	B
1 5 2	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; not_improved_students VARCHAR2(4000); BEGIN   -- Calculate and store the list of students who have not improved their grades in not_improved_students  DBMS_OUTPUT.PUT_LINE('Stud ents Who Have Not Improved Their Grades: '    not_improved_students); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT student_id   FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	B

1 5 3	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who   have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; not_improved_students VARCHAR2(4000); BEGIN   -- Calculate and store the list of students who have not improved their grades in not_improved_students  DBMS_OUTPUT.PUT_LINE('Students Who Have Not Improved Their Grades: '    not_improved_students); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT student_id   FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	B
1 5 4	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; not_improved_students VARCHAR2(4000); BEGIN   -- Calculate and store the list of students who have not improved their grades in not_improved_students  DBMS_OUTPUT.PUT_LINE('Students Who Have Not Improved Their Grades: '    not_improved_students); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT student_id   FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	B

1 5 5	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have not improved their grades in any subject compared to the previous year. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who   have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; not_improved_students VARCHAR2(4000); BEGIN   -- Calculate and store the list of students who have not improved their grades in not_improved_students  DBMS_OUTPUT.PUT_LINE('Stud ents Who Have Not Improved Their Grades: '    not_improved_students); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who have not improved their grades END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT student_id   FROM student; BEGIN   -- Find and print students who have not improved their grades END;</pre>	B
1 5 6	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in at least one subject and the lowest grade in at least one subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students with highest and lowest grades in subjects END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; students_with_extreme_grades VARCHAR2(4000); BEGIN   -- Calculate and store the list of students with extreme grades in students_with_extreme_grades  DBMS_OUTPUT.PUT_LINE('Stud ents with Extreme Grades: '    students_with_extreme_grades ); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students with highest and lowest grades in subjects END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT student_id   FROM student; BEGIN   -- Find and print students with highest and lowest grades in subjects END;</pre>	B

1 5 7	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; highest_grade NUMBER; BEGIN   -- Calculate and store the   highest grade for each subject   in highest_grade    DBMS_OUTPUT.PUT_LINE('Stud ents with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT     subject       FROM student; BEGIN   -- Find and print students   with the highest grade in   each subject END;</pre>	B
1 5 8	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; highest_grade NUMBER; BEGIN   -- Calculate and store the   highest grade for each subject   in highest_grade    DBMS_OUTPUT.PUT_LINE('Stud ents with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT     subject       FROM student; BEGIN   -- Find and print students   with the highest grade in   each subject END;</pre>	B



1 5 9	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; highest_grade NUMBER; BEGIN   -- Calculate and store the   highest grade for each subject   in highest_grade    DBMS_OUTPUT.PUT_LINE('Stud ents with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT     subject       FROM student; BEGIN   -- Find and print students   with the highest grade in   each subject END;</pre>	B
1 6 0	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; highest_grade NUMBER; BEGIN   -- Calculate and store the   highest grade for each subject   in highest_grade    DBMS_OUTPUT.PUT_LINE('Stud ents with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT     subject       FROM student; BEGIN   -- Find and print students   with the highest grade in   each subject END;</pre>	B

1 6 1	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the highest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; highest_grade NUMBER; BEGIN   -- Calculate and store the   highest grade for each subject   in highest_grade    DBMS_OUTPUT.PUT_LINE('Stud ents with Highest Grade in Each Subject: '    highest_grade); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students with   the highest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT     subject       FROM student; BEGIN   -- Find and print students   with the highest grade in   each subject END;</pre>	B
1 6 2	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the lowest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print students with   the lowest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; lowest_grade NUMBER; BEGIN   -- Calculate and store the   lowest grade for each subject in   lowest_grade    DBMS_OUTPUT.PUT_LINE('Stud ents with Lowest Grade in Each Subject: '    lowest_grade); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students with   the lowest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT     subject       FROM student; BEGIN   -- Find and print students   with the lowest grade in   each subject END;</pre>	B

1 6 3	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the lowest grade in each subject. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print students with   the lowest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; lowest_grade NUMBER; BEGIN   -- Calculate and store the   lowest grade for each subject in   lowest_grade    DBMS_OUTPUT.PUT_LINE('Stud ents with Lowest Grade in Each Subject: '    lowest_grade); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students with   the lowest grade in each   subject END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT   subject     FROM student; BEGIN   -- Find and print students   with the lowest grade in   each subject END;</pre>	B
1 6 4	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the same grade in all subjects. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id       FROM student; BEGIN   -- Find and print students who   have scored the same grade in   all subjects END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id       FROM student; students_with_same_grade VARCHAR2(4000); BEGIN   -- Calculate and store the list   of students with the same   grade in all subjects in   students_with_same_grade    DBMS_OUTPUT.PUT_LINE('Stud ents with Same Grade in All Subjects: '    students_with_same_grade); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id       FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who   have scored the same grade in   all subjects END;</pre>	<pre> DECLARE   CURSOR student_cursor IS   SELECT DISTINCT   student_id     FROM student; BEGIN   -- Find and print students   who have scored the   same grade in all subjects END;</pre>	B

1 6 5	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the same grade in all subjects. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; BEGIN   -- Find and print students who   have scored the same grade in   all subjects END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; students_with_same_grade VARCHAR2(4000); BEGIN   -- Calculate and store the list   of students with the same   grade in all subjects in   students_with_same_grade    DBMS_OUTPUT.PUT_LINE('Stud   ents with Same Grade in All   Subjects: '      students_with_same_grade); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id     FROM student; student_recordstudent%ROWT YPE; BEGIN   -- Find and print students who   have scored the same grade in   all subjects END;</pre>	<pre> DECLARE   CURSOR student_cursor   IS     SELECT DISTINCT   student_id   FROM student; BEGIN   -- Find and print students   who have scored the   same grade in all subjects END;</pre>	B
1 6 6	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subject with the highest overall grades (sum of grades for all students). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print the subject   with the highest overall grades END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; highest_subject VARCHAR2(100); BEGIN   -- Calculate and store the   subject with the highest overall   grades in highest_subject    DBMS_OUTPUT.PUT_LINE('Subj   ect with Highest Overall Grades:   '    highest_subject); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print the subject   with the highest overall grades END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT   subject   FROM student; BEGIN   -- Find and print the   subject with the highest   overall grades END;</pre>	B

1 6 7	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subject with the lowest average grade. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print the subject   with the lowest average grade END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; lowest_avg_grade NUMBER; BEGIN   -- Calculate and store the   lowest average grade for a   subject in lowest_avg_grade    DBMS_OUTPUT.PUT_LINE('Subj   ect with Lowest Average Grade:   '    lowest_avg_grade); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print the subject    with the lowest average grade END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT   subject     FROM student; BEGIN   -- Find and print the   subject with the lowest   average grade END;</pre>	B
1 6 8	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which all students have scored above a specified threshold (e.g., 60). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print subjects with   all scores above the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT   subject     FROM student; subjects_with_all_high_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with all high scores   in   subjects_with_all_high_scores    DBMS_OUTPUT.PUT_LINE('Subj   ect with All High Scores: '      subjects_with_all_high_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   all scores above the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT   subject     FROM student; BEGIN   -- Find and print subjects   with all scores above the   threshold END;</pre>	B

1 6 9	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which all students have scored above a specified threshold (e.g., 70). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print subjects with   all scores above the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subjects_with_all_high_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with all high scores   in   subjects_with_all_high_scores    DBMS_OUTPUT.PUT_LINE('Subj ects with All High Scores: '    subjects_with_all_high_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   all scores above the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT   subject     FROM student; BEGIN   -- Find and print subjects   with all scores above the   threshold END;</pre>	B
1 7 0	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 40). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subjects_with_low_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with at least one   low score in   subjects_with_low_scores    DBMS_OUTPUT.PUT_LINE('Subj ects with Low Scores: '    subjects_with_low_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT   subject     FROM student; BEGIN   -- Find and print subjects   with at least one score   below the threshold END;</pre>	A

1 7 1	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 50). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subjects_with_low_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with at least one   low score in   subjects_with_low_scores    DBMS_OUTPUT.PUT_LINE('Subj ects with Low Scores: '    subjects_with_low_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT   subject   FROM student; BEGIN   -- Find and print subjects   with at least one score   below the threshold END;</pre>	B
1 7 2	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 50). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subjects_with_low_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with at least one   low score in   subjects_with_low_scores    DBMS_OUTPUT.PUT_LINE('Subj ects with Low Scores: '    subjects_with_low_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS   SELECT DISTINCT   subject   FROM student; BEGIN   -- Find and print subjects   with at least one score   below the threshold END;</pre>	A

1 7 3	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 55). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subjects_with_low_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with at least one   low score in   subjects_with_low_scores    DBMS_OUTPUT.PUT_LINE('Subj   ects with Low Scores: '      subjects_with_low_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT     subject     FROM student; BEGIN   -- Find and print subjects   with at least one score   below the threshold END;</pre>	B
1 7 4	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which at least one student has scored below a specified threshold (e.g., 60). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subjects_with_low_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with at least one   low score in   subjects_with_low_scores    DBMS_OUTPUT.PUT_LINE('Subj   ects with Low Scores: '      subjects_with_low_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   at least one score below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT     subject     FROM student; BEGIN   -- Find and print subjects   with at least one score   below the threshold END;</pre>	B



1 7 5	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which every student has scored above a specified threshold (e.g., 85). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print subjects with   all scores above the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subjects_with_all_high_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with all high scores   in   subjects_with_all_high_scores    DBMS_OUTPUT.PUT_LINE('Subj   ects with All High Scores: '      subjects_with_all_high_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   all scores above the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT       subject       FROM student; BEGIN   -- Find and print subjects   with all scores above the   threshold END;</pre>	B
1 7 6	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which every student has scored above a specified threshold (e.g., 90). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print subjects with   all scores above the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subjects_with_all_high_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with all high scores   in   subjects_with_all_high_scores    DBMS_OUTPUT.PUT_LINE('Subj   ects with All High Scores: '      subjects_with_all_high_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   all scores above the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT       subject       FROM student; BEGIN   -- Find and print subjects   with all scores above the   threshold END;</pre>	B

1 7 7	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which no student has scored below a specified threshold (e.g., 60). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print subjects with   no scores below the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subjects_with_no_low_scores VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with no low scores   in   subjects_with_no_low_scores    DBMS_OUTPUT.PUT_LINE('Subj   ects with No Low Scores: '      subjects_with_no_low_scores); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   no scores below the threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT     subject     FROM student; BEGIN   -- Find and print subjects   with no scores below the   threshold END;</pre>	B
1 7 8	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which the average grade is above a specified threshold (e.g., 75). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; BEGIN   -- Find and print subjects with   an average grade above the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subjects_above_threshold VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with an average   grade above the threshold in   subjects_above_threshold    DBMS_OUTPUT.PUT_LINE('Subj   ects with Average Grade Above   Threshold: '      subjects_above_threshold); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject     FROM student; subject_recordstudent%ROWTY PE; BEGIN   -- Find and print subjects with   an average grade above the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT     subject     FROM student; BEGIN   -- Find and print subjects   with an average grade   above the threshold END;</pre>	B

1 7 9	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which the average grade is above a specified threshold (e.g., 75). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print subjects with   an average grade above the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student;   subjects_above_threshold     VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with an average   grade above the threshold in   subjects_above_threshold    DBMS_OUTPUT.PUT_LINE('Subj   ects with Average Grade Above   Threshold: '      subjects_above_threshold); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student;   subject_recordstudent%ROWTY     PE; BEGIN   -- Find and print subjects with   an average grade above the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor     IS     SELECT DISTINCT   subject     FROM student; BEGIN   -- Find and print subjects   with an average grade   above the threshold END;</pre>	B
1 8 0	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which the average grade is above a specified threshold (e.g., 80). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print subjects with   an average grade above the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student;   subjects_above_threshold     VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with an average   grade above the threshold in   subjects_above_threshold    DBMS_OUTPUT.PUT_LINE('Subj   ects with Average Grade Above   Threshold: '      subjects_above_threshold); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student;   subject_recordstudent%ROWTY     PE; BEGIN   -- Find and print subjects with   an average grade above the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor     IS     SELECT DISTINCT   subject     FROM student; BEGIN   -- Find and print subjects   with an average grade   above the threshold END;</pre>	B

1 8 1	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the subjects in which the average grade is below a specified threshold (e.g., 70). Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student; BEGIN   -- Find and print subjects with   an average grade below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student;   subjects_below_threshold     VARCHAR2(4000); BEGIN   -- Calculate and store the list   of subjects with an average   grade below the threshold in   subjects_below_threshold    DBMS_OUTPUT.PUT_LINE('Subj   ects with Average Grade Below   Threshold: '      subjects_below_threshold); END;</pre>	<pre> DECLARE   CURSOR subject_cursor IS     SELECT DISTINCT subject       FROM student;   subject_recordstudent%ROWTY   PE; BEGIN   -- Find and print subjects with   an average grade below the   threshold END;</pre>	<pre> DECLARE   CURSOR subject_cursor   IS     SELECT DISTINCT   subject     FROM student; BEGIN   -- Find and print subjects   with an average grade   below the threshold END;</pre>	B
1 8 2	Given a "student" table with columns `student_id`, `subject`, and `grade`, write a PL/SQL block that uses a cursor to find and print the students who have scored the same grade in all subjects. Which of the following code snippets accomplishes this task?	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id       FROM student; BEGIN   -- Find and print students who   have scored the same grade in   all subjects END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id       FROM student;   students_with_same_grade     VARCHAR2(4000); BEGIN   -- Calculate and store the list   of students with the same   grade in all subjects in   students_with_same_grade    DBMS_OUTPUT.PUT_LINE('Stud   ents with Same Grade in All   Subjects: '      students_with_same_grade); END;</pre>	<pre> DECLARE   CURSOR student_cursor IS     SELECT DISTINCT student_id       FROM student;   student_recordstudent%ROWT   YPE; BEGIN   -- Find and print students who   have scored the same grade in   all subjects END;</pre>	<pre> DECLARE   CURSOR student_cursor   IS     SELECT DISTINCT   student_id     FROM student; BEGIN   -- Find and print students   who have scored the   same grade in all subjects END;</pre>	B

183	How can concurrent access to shared data lead to data inconsistency in a DBMS?	A) By preventing data updates	B) By enforcing data integrity rules	C) By allowing simultaneous updates	D) By reducing query performance	C
184	How can you remove a package from the database in PL/SQL?	By using the DROP PACKAGE statement	By removing all the procedures and functions from the package body	By using the DELETE PACKAGE statement	By using the TRUNCATE PACKAGE statement	A
185	How many mandatory parts packages has?	1	2	3	4	B
186	In a DBMS, a package is typically used to:	Store and organize tables and views	Group related procedures, functions, and variables	Define user roles and permissions	Execute ad-hoc SQL queries	B
187	In a DBMS, what is a benefit of using stored procedures for business logic?	It allows for dynamic SQL execution.	<b>It centralizes and secures the business logic.</b>	It simplifies data retrieval operations.	It eliminates the need for data validation.	B
188	In a DBMS, what is a database trigger?	A procedure that runs when the database is created	A statement that rolls back database changes	A set of rules for data validation	<b>A piece of code that automatically executes in response to a specific event</b>	D
189	In a DBMS, what is the main purpose of a database trigger?	To enforce referential integrity constraints	To encapsulate business logic for data processing	To automatically generate primary keys	To record changes to data in response to events	D
190	In a DBMS, what is the primary purpose of a "SERIALIZABLE" isolation level?	To maximize concurrency	To prevent transactions from acquiring locks	To ensure that transactions execute in a specific order	<b>To provide the highest level of data consistency and isolation</b>	D
191	In a DBMS, what is the primary purpose of a database package body?	A) To define package variables	B) To provide information about the package's procedures and functions	C) To specify package triggers	D) To implement the actual code for package procedures	D
192	In a DBMS, what is the primary purpose of using "SELECT FOR UPDATE"?	To retrieve data for reporting purposes	<b>To lock rows, preventing other transactions from modifying them</b>	To retrieve data without acquiring locks	To roll back a transaction	B

193	In a DBMS, what is the primary purpose of using a "READ COMMITTED" isolation level?	To minimize data redundancy	To ensure that transactions execute in a specific order	To prevent transactions from acquiring locks	<b>To balance data consistency and concurrency</b>	D
194	In a DBMS, what is the primary purpose of using an "AFTER UPDATE" trigger?	To prevent any updates from occurring	To execute before any update operation occurs	<b>To log changes made to a table after an update</b>	To execute after a row is updated in a table	C
195	In a DBMS, what is the primary purpose of using triggers?	To simplify data retrieval operations	<b>To enforce data integrity constraints</b>	To create temporary tables	To improve query performance	B
196	In a DBMS, what is the purpose of a "SAVE TRANSACTION" statement?	To save a transaction for later execution	To create a new transaction	<b>To save the current state of a transaction as a savepoint</b>	To commit the transaction	C
197	In a DBMS, what is the purpose of a savepoint?	To commit a transaction	To roll back a transaction	<b>To create a point within a transaction to which you can later roll back</b>	To lock a table temporarily	C
198	In a DBMS, what is the purpose of the "COMMIT" statement?	To start a new transaction	<b>To save all changes made within the current transaction</b>	To roll back all changes made within the current transaction	To create a new savepoint	B
199	In a DBMS, what is the purpose of the "ROLLBACK TO SAVEPOINT" statement?	To roll back an entire transaction	To create a new savepoint	<b>To roll back to a specific savepoint within a transaction</b>	To commit a transaction	C
200	In a DBMS, what is the purpose of the "SET TRANSACTION" statement?	<b>To define transaction isolation levels</b>	To start a new transaction	To commit a transaction	To roll back a transaction	A
201	In a DBMS, what is the purpose of the procedure header?	A) To declare the procedure's input and output variables	B) To specify the operations to be performed by the procedure	C) To implement the actual code for the procedure	D) To define the package specification	A
202	In a DBMS, what is the term for a package that only contains the package specification without a package body?	A) A minimal package	B) A comprehensive package	C) A bodiless package	D) A complete package	C
203	In a DBMS, what is the typical process of developing a package?	A) Write the package body first, then the specification	B) Write the package specification first, then the body	C) Develop procedures and triggers independently of packages	D) Develop triggers before procedures	B

204	In a DBMS, which clause is used to specify the action to be taken when a trigger event occurs?	EXECUTE	FOR EACH ROW	WHEN	INSTEAD OF	C
205	In a DBMS, which type of cursor is used to fetch and process one row at a time?	Static cursor	Dynamic cursor	Forward-only cursor	Scrollable cursor	C
206	In a locking-based concurrency control system, what does a "lock" prevent other transactions from doing?	A) Accessing the locked data	B) Aborting the transaction	C) Executing queries	D) Creating database triggers	A
207	In a multi-user DBMS, what issue can occur without proper concurrency control?	A) Faster query execution	B) Data consistency problems	C) Reduced code duplication	D) Increased code modularity	B
208	In a multi-user DBMS, what problem can arise when transactions are executed concurrently without control?	A) Faster data retrieval	B) Enhanced data consistency	C) Reduced code duplication	D) Increased code modularity	B
209	In Oracle PL/SQL, can a package contain both procedures and functions?	Yes, but they must all have the same name.	No, a package can only contain either procedures or functions, not both.	<b>Yes, a package can contain a mix of procedures and functions.</b>	Yes, but they must all be stored in separate packages.	C
210	In Oracle PL/SQL, what is a benefit of using packages over standalone procedures?	Packages are easier to write and debug.	<b>Packages allow you to encapsulate related code and data.</b>	Packages execute faster than standalone procedures.	Packages are not dependent on any database schema.	B
211	In Oracle PL/SQL, which statement is used to raise an exception explicitly within a stored procedure?	RAISE EXCEPTION	SIGNAL	RAISE_APPLICATION_ERROR	THROW	C
212	In PL/SQL, the CREATE TABLESPACE is used	To create a place in the database for storage of scheme objects, rollback segments, and naming the data files to comprise the table-space	To create a database trigger	To add/rename data files, to change storage	All of the above	A

2 1 3	In PL/SQL, which of the following statements accurately describes a view?	A view is a virtual table based on the result of a SELECT query.	A view is a physical table that stores data permanently.	A view is a temporary table used for transactional purposes.	A view is a table used only for indexing purposes.	A
2 1 4	In PL/SQL, which trigger event is fired when a column value is updated to NULL?	AFTER UPDATE NULL	BEFORE UPDATE NULL	AFTER UPDATE OF column_name	BEFORE UPDATE OF column_name	C
2 1 5	In the context of a DBMS, what is the purpose of a stored procedure?	To store data in the database	To define the structure of a table	To encapsulate a series of SQL statements for reuse	To retrieve data from external sources	C
2 1 6	In the context of database transactions, what is a deadlock?	<b>A situation where two or more transactions are waiting for each other to release locks, preventing progress.</b>	A situation where a transaction reads uncommitted data.	A situation where a transaction violates integrity constraints.	A situation where a transaction is aborted due to a rollback request.	A
2 1 7	In the context of isolation levels in a DBMS, what does the "Read Uncommitted" isolation level allow?	Transactions can read uncommitted changes made by other transactions.	Transactions cannot read any data until all changes are committed.	Transactions can only read committed data.	Transactions can read their own uncommitted changes.	A
2 1 8	In the context of isolation levels in a DBMS, what does the "Read Uncommitted" isolation level allow?	<b>Transactions can read uncommitted changes made by other transactions.</b>	Transactions cannot read any data until all changes are committed.	Transactions can only read committed data.	Transactions can read their own uncommitted changes.	A
2 1 9	In the context of transactions, what does "serializability" mean?	A) The ability to serialize data	B) The ability to execute transactions concurrently	C) The ability to recover from failures	D) The ability to perform queries efficiently	A
2 2 0	OLD and NEW references are not available for table-level triggers.	TRUE	FALSE	Can be true or false	None of the above	A
2 2 1	Packages are schema objects that groups logically related PL/SQL types, variables, and subprograms.	Yes	No	Can be yes or no	none of the above	A



2 2 2	PL/SQL controls the context area through a cursor.	TRUE	FALSE	Can be true or false	All of the above	A
2 2 3	PL/SQL supports programmers to catch such conditions using _____ block in the program	Try	Throw	Catch	Exception	D
2 2 4	Repeat sequence of statements; _____ end repeat Fill in the correct option :	While Condition	Until variable	Until boolean expression	Until 0	C
2 2 5	Repeat sequence of statements; _____ end repeat Fill in the correct option :	While Condition	Until variable	Until boolean expression	Until 0	C
2 2 6	Suppose a database system crashes again while recovering from a previous crash. Assume checkpointing is not done by the database either during the transactions or during recovery.  Which of the following statements is/are correct?	The same undo and redo list will be used while recovering again.	The database will become inconsistent.	All the transactions that are already undone and redone will not be recovered again.	The system cannot recover any further.	A
2 2 7	Temporary stored procedures are stored in _____ database.	Master	Model	User specific	Tempdb	D
2 2 8	The _____ Statement is used for creating the package body.	CREATE	CREATE PACKAGE	CREATE BODY	CREATE PACKAGE BODY	D
2 2 9	The _____ Statement is used for creating the package body.	CREATE PACKAGE BODY	CREATE	CREAT BODY	CREATE BODY	A

2 3 0	The constructs of a procedure, function or a package are _____ .	Variables and Constants	Cursors	Exceptions	All of the above	D
2 3 1	The CREATE TRIGGER statement is used to create the trigger. THE _____ clause specifies the table name on which the trigger is to be attached. The _____ specifies that this is an AFTER INSERT trigger.	for insert, on	On, for insert	For, insert	None of the mentioned	B
2 3 2	The CREATE TRIGGER statement is used to create the trigger. THE _____ clause specifies the table name on which the trigger is to be attached. The _____ specifies that this is an AFTER INSERT trigger.	for insert, on	On, for insert	For, insert	None of the mentioned	A
2 3 3	The format for compound statement is	Begin ..... end	Begin atomic..... end	Begin ..... repeat	Both Begin ..... end and Begin atomic..... end	D
2 3 4	The package specification is the interface to the package.	TRUE	FALSE	Nither TRUE NOR FALSE	none of the above	A
2 3 5	The parameters can be passed as default also to the procedures and the functions.	TRUE	FALSE	Nither TRUE NOR FALSE	None of he above	A
2 3 6	The property of a schedule that states that the result of executing concurrent transactions is the same as executing them serially is known as:	Consistency	Atomicity	Serializability	Durability	C

2 3 7	The technique used to detect and resolve conflicts among concurrent transactions is called:	Two-phase locking	Timestamp ordering	Deadlock detection	Deadlock prevention	C
2 3 8	Triggers can be defined on the?	table	view	schema	All of the above	D
2 3 9	Triggers can be defined on the?	DDL	DML	Database Operation	All of the above	D
2 4 0	What does "recoverability" encompass in the context of transactions and concurrency control?	A) The ability to recover from system failures	B) The ability to execute transactions concurrently	C) The ability to lock database tables	D) The ability to perform efficient queries	A
2 4 1	What does DBA stand for in the context of databases?	A) Database Backup Administrator	B) Data Business Analyst	C) Database Architect	D) Database Administrator	D
2 4 2	What does the concept of "recoverability" in concurrency control refer to?	A) The ability to lock data	B) The ability to recover from system failures	C) The ability to perform database recovery	D) The ability to execute queries efficiently	B
2 4 3	What does the concept of "serializability" in concurrency control refer to?	A) The ability to lock data	B) The ability to execute transactions in parallel	C) The ability to perform database recovery	D) The ability to execute queries efficiently	B
2 4 4	What does the term "serializability" imply in the context of transaction execution?	A) Transactions occur in sequence	B) Transactions can execute concurrently	C) Transactions are aborted	D) Transactions are isolated	A
2 4 5	What does the term "transaction isolation" refer to in the context of concurrency control?	A) A transaction's lifespan	B) A transaction's ability to update data	C) A transaction's isolation level	D) A transaction's recovery	C
2 4 6	What is a "bodiless" package in a DBMS context?	A) A package without a body	B) A package with excessive code	C) A package with only triggers	D) A package with minimal documentation	A

2 4 7	What is a "bodiless" package in a DBMS context?	A) A package without a body	B) A package with excessive code	C) A package with only triggers	D) A package with minimal documentation	A
2 4 8	What is a "transaction" in the context of a database management system (DBMS)?	A) A data dictionary	B) A single unit of work	C) A database schema	D) A database connection	B
2 4 9	What is a common approach to resolving deadlocks in a DBMS?	Rolling back one of the transactions involved in the deadlock	Killing all transactions to release locks	Preventing transactions from acquiring locks	Using deadlock detection and resolution algorithms	D
2 5 0	What is a common drawback of "pessimistic" locking in concurrency control systems?	A) Increased code modularity	B) Reduced data consistency	C) Reduced query performance	D) Optimized query execution	C
2 5 1	What is a common use of a database trigger in a DBMS?	A) To define package specifications	B) To encapsulate related procedures and functions	C) To monitor and respond to database events	D) To create database packages	C
2 5 2	What is a database schema?	A) A collection of tables in a database	B) A diagram representing the structure of a database	C) A set of rules that define the database structure	D) A description of the database structure, including tables, fields, and relationships	D
2 5 3	What is a package body in a DBMS?	A) A part of a package that contains package variables	B) A part of a package that specifies the package's procedures and functions	C) A part of a package that defines package triggers	D) A part of a package that implements the actual code for package procedures	D
2 5 4	What is a package specification in a DBMS?	A) A part of a package that contains package variables	B) A part of a package that specifies the package's procedures and functions	C) A part of a package that defines package triggers	D) A part of a package that implements the actual code for package procedures	D
2 5 5	What is a parameterized stored procedure in a DBMS?	<b>A procedure that accepts parameters and returns a result set</b>	A procedure that uses a parameter as its name	A procedure that cannot accept any input parameters	A procedure that can only accept integer parameters	A

2 5 6	What is a transaction in a DBMS?	A database schema	A series of SQL statements	<b>A logical unit of work that is either fully completed or fully undone</b>	A data dictionary	C
2 5 7	What is correct a PL/SQL program that create Trigger to update the "salary" of an employee to 80000 if the "department" is changed to 'Management'	<pre> CREATE OR REPLACE TRIGGER trg_department_update BEFORE UPDATE OF department ON employee FOR EACH ROW BEGIN     IF :NEW.department = 'Management' THEN         :NEW.salary := 80000;     END IF; END; / </pre>	<pre> CREATE OR REPLACE TRIGGER trg_department_update BEFORE UPDATE OF department ON employee FOR EACH ROW     IF :NEW.department = 'Management' THEN         :NEW.salary := 80000;     END IF; END; / </pre>	<pre> CREATE OR REPLACE TRI trg_department_update BEFORE UPDATE OF department ON employee FOR EACH BEGIN     IF :NEW.department = 'Management' THEN         :NEW.salary := 80000;     END IF; END; / </pre>	<pre> CREATE OR REPLACE TRIGGER trg_department_update UPDATE OF department ON employee ROW BEGIN     IF :NEW.department = 'Management' THEN         :NEW.salary := 80000;     END IF; END; / </pre>	A

2 5 8	What is correct a procedure that calculates and displays the total salary of employees in a given department. The department name is an optional parameter with a default value of 'HR'.	<pre> CREATE OR REPLACE PROCEDURE total_salary_by_department(p_ department_name IN VARCHAR2 DEFAULT 'HR') AS     v_total_salary NUMBER; BEGIN     SELECT SUM(salary) INTO v_total_salary     FROM employee     WHERE department = p_department_name;      DBMS_OUTPUT.PUT_LINE('Total Salary for Department '    p_department_name    ': '    v_total_salary); EXCEPTION     WHEN OTHERS THEN      DBMS_OUTPUT.PUT_LINE('An error occurred. '); END; / </pre>	<pre> CREATE OR REPLACE PROCEDURE total_salary_by_department(p_ department_name IN VARCHAR2 ') AS     v_total_salary NUMBER; BEGIN     SELECT SUM(salary) INTO v_total_salary     FROM employee     WHERE department = p_department_name;      DBMS_OUTPUT.PUT_LINE('Total Salary for Department '    p_department_name    ': '    v_total_salary); EXCEPTION     WHEN OTHERS THEN      DBMS_OUTPUT.PUT_LINE('An error occurred. '); END; / </pre>	<pre> CREATE OR REPLACE PROCEDURE total_salary_by_department(p_ department_name IN VARCHAR2 DEFAULT 'HR') AS     v_total_salary NUMBER; SELECT SUM(salary) INTO v_total_salary     FROM employee     WHERE department = p_department_name;      DBMS_OUTPUT.PUT_LINE('Total Salary for Department '    p_department_name    ': '    v_total_salary); EXCEPTION     WHEN OTHERS THEN      DBMS_OUTPUT.PUT_LINE('An error occurred. '); END; / </pre>	<pre> CREATE OR REPLACE PROCEDURE total_salary_by_departme nt(DEFAULT 'HR') AS     v_total_salary NUMBER; BEGIN     SELCT SUM(salary) INTO v_total_salary     FROM employee     WHERE department = p_department_name;      DBMS_OUTPUT.PUT_LINE ('Total Salary for Department '    p_department_name    ': '    v_total_salary); EXCEPTION     WHEN OTHERS THEN      DBMS_OUTPUT.PUT_LINE ('An error occurred. '); END; / </pre>	A
2 5 9	What is one of the advantages of using procedures in a DBMS?	A) Increased code duplication	B) Slower query performance	C) Enhanced security vulnerabilities	D) Reduced code redundancy	D
2 6 0	What is one of the advantages of using triggers in a DBMS?	A) Increased code modularity	B) Reduced control over data changes	C) Enhanced query performance	D) Automated enforcement of data integrity rules	D

2 6 1	<p>What is Output-- Insert sample records into the "employee" table</p> <pre> INSERT INTO employee (employee_id, first_name, last_name, department, salary) VALUES (1, 'John', 'Doe', 'HR', 50000);  INSERT INTO employee (employee_id, first_name, last_name, department, salary) VALUES (2, 'Jane', 'Smith', 'Finance', 60000);  INSERT INTO employee (employee_id, first_name, last_name, department, salary) VALUES (3, 'Michael', 'Johnson', 'IT', 70000);  INSERT INTO employee (employee_id, first_name, last_name, department, salary) VALUES (4, 'Merry', 'Agarwal', 'IT', 50000);  CREATE OR REPLACE PROCEDURE delete_employee_by_id(p_ employee_id NUMBER) AS BEGIN     DELETE FROM employee     WHERE employee_id = </pre>	An error occurred.	Employee ID not found. No employee deleted.	Employee deleted successfully.	EXECUTE delete_employee_by_id(2 );	B
-------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------	---------------------------------------------	--------------------------------	------------------------------------------	---

	<pre> p_employee_id;      IF SQL%ROWCOUNT &gt; 0     THEN          DBMS_OUTPUT.PUT_LINE('E         mployee deleted         successfully.');</pre> <pre>         ELSE          DBMS_OUTPUT.PUT_LINE('E         mployee ID not found. No         employee deleted.');</pre> <pre>         END IF;     EXCEPTION         WHEN OTHERS THEN          DBMS_OUTPUT.PUT_LINE('         An error occurred.');</pre> <pre>     END; / EXECUTE delete_employee_by_id(6); </pre>					
2 6 2	What is the ACID property that ensures that transactions are performed correctly and completely?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	A
2 6 3	What is the key difference between stored procedures and functions in DBMS?	Stored procedures can return multiple values, while functions can only return a single value	Stored procedures can be executed by users, while	Stored procedures are used for data manipulation, while	Stored procedures can be called from within other	A



			functions can only be executed by the database administrator	functions are used for data retrieval	procedures, while functions cannot	
2 6 4	What is the Oracle Error Code for ACCESS_INTO_NULL?	6592	6531	1722	6530	D
2 6 5	What is the output of the following program? DECLARE A NUMBER :=2; BEGIN FOR I IN 1..3 LOOP A := A*2; END LOOP; DBMS_OUTPUT.PUT_LINE(A); END;	4	8	16	32	D
2 6 6	What is the primary advantage of using packages in a DBMS?	Improved query performance	Enhanced data security	Better code organization and reusability	Simplified database design	D
2 6 7	What is the primary difference between a row-level trigger and a statement-level trigger?	Row-level triggers are executed before statement-level triggers.	Row-level triggers are fired once for each affected row, while statement-level triggers are fired once for each SQL statement.	Statement-level triggers can be defined on tables, whereas row-level triggers cannot.	Row-level triggers can be recursive, while statement-level triggers cannot.	B
2 6 8	What is the primary drawback of "optimistic" concurrency control in a DBMS?	A) Increased code modularity	B) Slower query performance	C) Risk of transaction conflicts	D) Enhanced data integrity	C
2 6 9	What is the primary goal of concurrency control in a DBMS?	To maximize data redundancy	To minimize database access	To ensure data consistency and integrity in a multi-user environment	To eliminate the need for indexing	C
2 7 0	What is the primary goal of concurrency control in a DBMS?	To maximize data redundancy	To minimize database access	<b>To ensure data consistency and integrity in a multi-user environment</b>	To eliminate the need for indexing	C

2 7 1	What is the primary purpose of "deadlock detection" mechanisms in a DBMS that uses locking for concurrency control?	A) To prevent transaction conflicts	B) To optimize query performance	C) To eliminate transactions	D) To create database triggers	C
2 7 2	What is the primary purpose of a "BEFORE DELETE" trigger in a DBMS?	<b>To execute before any delete operation occurs</b>	To execute after a row is deleted from a table	To prevent any delete operations from taking place	To execute only if a delete operation fails	A
2 7 3	What is the primary purpose of a cursor in a stored procedure?	To store the results of a query	<b>To iterate through the records returned by a query</b>	To create a temporary table	To enforce data integrity constraints	B
2 7 4	What is the primary purpose of a database package in a DBMS?	A) To define database triggers	B) To encapsulate and group related procedures, functions, and variables	C) To establish database connections	D) To optimize query performance	B
2 7 5	What is the primary purpose of a stored procedure in DBMS?	To store and organize data in a database	To retrieve data from the database	To define the structure of the database	To encapsulate a series of database operations	D
2 7 6	What is the primary purpose of an "AFTER INSERT" trigger in a DBMS?	To execute before any insert operation occurs	<b>To execute after a new row is inserted into a table</b>	To prevent any insert operations from taking place	To execute only if an insert operation fails	B
2 7 7	What is the primary purpose of locking in concurrency control?	A) To eliminate transactions	B) To optimize query performance	C) To manage data access	D) To create database triggers	C
2 7 8	What is the primary purpose of transaction management in a database system?	A) To optimize query performance	B) To ensure data consistency	C) To define database triggers	D) To establish database connections	B
2 7 9	What is the primary purpose of using packages in a DBMS?	To store data in tables	<b>To organize related procedures, functions, and variables</b>	To manage user permissions	To enforce data integrity constraints	B
2 8 0	What is the primary syntax for creating a trigger in a DBMS?	A) CREATE PROCEDURE	B) CREATE FUNCTION	C) CREATE TRIGGER	D) CREATE TABLE	C
2 8 1	What is the purpose of "recoverability" in the context of database transactions?	A) To ensure all transactions recover	B) To prevent data recovery issues	C) To recover from system failures	D) To lock database tables	C

2 8 2	What is the purpose of declaring a cursor in SQL?	To define a new table in the database	To specify the database connection string	<b>To define a result set for query execution</b>	To create a new user account	C
2 8 3	What is the purpose of the "ROLLBACK" statement in a stored procedure?	To commit all changes made within the procedure	<b>To undo all changes made within the procedure</b>	To restart the execution of the procedure from the beginning	To create a new savepoint within the procedure	B
2 8 4	What is the purpose of the internal schema in a database system?	A) To define the logical view of the database for users	B) To specify the access controls and security settings for the database	C) To represent the physical storage structure of the database	D) To define the user views and queries for the database	C
2 8 5	What is the purpose of the JOIN operation in a relational database?	A) To add new records to a table	B) To remove records from a table	C) To combine data from multiple tables based on a related column	D) To modify existing records in a table	C
2 8 6	What is the purpose of the SAVEPOINT statement in DBMS?	To define the start of a transaction	To create a temporary table	To define a point within a transaction to which you can roll back	To release a lock on a database object	C
2 8 7	What is the role of a "transaction log" in a DBMS with respect to concurrency control?	A) To manage database locks	B) To record transaction history	C) To optimize query performance	D) To create database triggers	B
2 8 8	What is the role of a "transaction manager" in a DBMS?	A) To design the database	B) To manage database connections	C) To coordinate transaction execution	D) To create database triggers	C
2 8 9	What is the role of a package specification in a database package?	A) To define package variables	B) To provide information about the package's procedures and functions	C) To specify package triggers	D) To establish database connections	B
2 9 0	What is the role of the FETCH statement in SQL cursor operations?	It declares a new cursor.	<b>It retrieves rows from the result set and moves the cursor to the next row.</b>	It closes an open cursor.	It defines the structure of a cursor.	B
2 9 1	What is the role of the FETCH statement in SQL cursor operations?	It declares a new cursor.	<b>It retrieves rows from the result set and moves the cursor to the next row.</b>	It closes an open cursor.	It defines the structure of a cursor.	B

2 9 2	What is the syntax of User-defined exceptions?	DECLARE my-exception EXCEPTION;	DECLARE EXCEPTION;	DECLARE my-exception;	EXCEPTION;	A
2 9 3	What is the typical sequence of steps for developing a package in a DBMS?	A) Develop triggers first, then procedures	B) Develop procedures and package specification simultaneously	C) Write the package specification first, then the package body	D) Write the package body first, then the specification	C
2 9 4	What is the typical sequence of steps for developing a package in a DBMS?	A) Develop triggers first, then procedures	B) Develop procedures and package specification simultaneously	C) Write the package specification first, then the package body	D) Write the package body first, then the specification	C
2 9 5	What part of a procedure in a DBMS is responsible for declaring the input and output variables?	A) Procedure header	B) Procedure specification	C) Procedure body	D) Procedure parameters	B
2 9 6	What type of trigger is executed automatically after the triggering event?	A) After Trigger	B) Before Trigger	C) Instead of Trigger	D) Compound Trigger	A
2 9 7	What type of trigger is executed automatically before a specific event, such as an INSERT or UPDATE operation?	A) After Trigger	B) Before Trigger	C) Instead of Trigger	D) Compound Trigger	B
2 9 8	When executing a stored procedure, what keyword is commonly used to return a result set to the calling application?	RESULT	RESULTSET	<b>OUTPUT</b>	RETURN	C
2 9 9	When one transaction updates a database item, and somehow the transaction fails, and the data doesn't get ___ back, another transaction tries to access the updated database item.	Rolled	Committed	Aborted	None	A

300	Which ACID property ensures that a transaction's effects on the database are permanent?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	D
301	Which attribute is used to raise exception?	Open	Select	Raise	Try	C
302	Which attribute returns TRUE if an INSERT, UPDATE, or DELETE statement affected one or more rows?	%NOTFOUND	%ISOPEN	%ROWCOUNT	%FOUND	D
303	Which clause is used to create trigger on a view?	BEFORE	INSTEAD OF	AFTER	None of the above	B
304	Which component of a package in DBMS defines the interface and public entities?	Package body	Package Signature	Package Constructor	Package specification	D
305	Which concurrency control technique allows multiple transactions to read data simultaneously but enforces write locks to prevent data conflicts?	Two-Phase Locking (2PL)	Time-stamp Ordering	Multi-Version Concurrency Control (MVCC)	Optimistic Concurrency Control	A
306	Which concurrency control technique allows multiple transactions to read data simultaneously but enforces write locks to prevent data conflicts?	<b>Two-Phase Locking (2PL)</b>	Time-stamp Ordering	Multi-Version Concurrency Control (MVCC)	Optimistic Concurrency Control	A
307	Which cursor attribute can be used to determine the total number of rows returned by a cursor in PL/SQL?	%ROWCOUNT	%FOUND	%ISOPEN	%NOTFOUND	A

308	Which database system component is responsible for managing transactions and ensuring data integrity?	A) Database schema	B) Data dictionary	C) Concurrency control manager	D) Query optimizer	C
309	Which isolation level allows only committed data to be read?	Read Uncommitted	Read committed	Serializable	Read Update	B
310	Which isolation level allows the highest concurrency but may result in non-repeatable reads?	Read Uncommitted	Read Committed	Repeatable Read	Serializable	A
311	Which isolation level provides the highest level of data consistency but can lead to reduced concurrency?	Read Uncommitted	Read Committed	Repeatable Read	Serializable	D
312	Which isolation level provides the highest level of data consistency but can lead to reduced concurrency?	Read Uncommitted	Read Committed	Repeatable Read	<b>Serializable</b>	D
313	Which keyword is used to create a new package body in PL/SQL?	BODY	IMPLEMENTATION	DEFINE	CREATE	A
314	Which keyword is used to define an exception handler in PL/SQL?	EXCEPTION	CATCH	TRY	HANDLE	A
315	Which of the following actions can be performed by a "BEFORE INSERT" trigger in a DBMS?	Rollback the entire transaction	<b>Modify data before it is inserted</b>	Create a new table	Terminate the DBMS session	B
316	Which of the following are benefits of Triggers?	Generating some derived column values automatically	Enforcing referential integrity	Event logging and storing information on table access	All of the above	D
317	Which of the following are the advantages of PL/SQL Packages?	Modularity	Easier Application Design	Information Hiding	All of the above	D

3 1 8	Which of the following clause does not comes in the syntax while raising an exception?	DECLARE	WHEN	CLOSE	END	C
3 1 9	Which of the following database languages is used to define the structure and organization of a database?	A) Data Manipulation Language (DML)	B) Data Definition Language (DDL)	C) Data Control Language (DCL)	D) Data Query Language (DQL)	B
3 2 0	Which of the following describes the ACID properties of transactions?	Atomicity, Consistency, Isolation, Durability	Atomicity, Consistency, Isolation, Dileang	Atomicity, Consistency, Isolate, Durability	Atomic, Consistency, Isolation, Durability	A
3 2 1	Which of the following is a benefit of using procedures in a DBMS?	A) Increased code duplication	B) Reduced security	C) Improved code organization and maintenance	D) Limited code reuse	C
3 2 2	Which of the following is a benefit of using the "SAVEPOINT" statement in a DBMS?	It allows you to commit a transaction.	It allows you to roll back the entire database.	It enables you to create nested transactions.	<b>It provides a way to roll back to a specific point within a transaction.</b>	D
3 2 3	Which of the following is a benefit of using the "SAVEPOINT" statement in a DBMS?	It allows you to commit a transaction.	It allows you to roll back the entire database.	It enables you to create nested transactions.	<b>It provides a way to roll back to a specific point within a transaction.</b>	D
3 2 4	Which of the following is a characteristic of an "AUTOCOMMIT" transaction mode in a DBMS?	Each SQL statement is treated as a separate transaction.	<b>Transactions are automatically committed after each SQL statement.</b>	Transactions cannot be rolled back.	It is a read-only mode.	B
3 2 5	Which of the following is a commonly used technique for concurrency control in a DBMS?	A) Optimistic concurrency control	B) Serializability control	C) Slower query performance	D) Increased code duplication	B
3 2 6	Which of the following is a drawback of strict two-phase locking (S2PL)	Increased concurrency	Increased deadlock probability	Reduced consistency	Reduced durability	B
3 2 7	Which of the following is a property of a transaction in a database system?	A) Increased code duplication	B) Slower query performance	C) Atomicity	D) Reduced code redundancy	C

3 2 8	Which of the following is an advantage of using packages in a DBMS?	A) Limited code organization	B) Increased code redundancy	C) Enhanced code isolation	D) Improved code modularity and reusability	D
3 2 9	Which of the following is an advantage of using procedures in a DBMS?	A) Increased code redundancy	B) Slower execution of queries	C) Improved security vulnerabilities	D) Code reusability and maintainability	D
3 3 0	Which of the following is an advantage of using stored procedures in a DBMS?	Reduced security	Increased data redundancy	<b>Improved performance</b>	Decreased data integrity	C
3 3 1	Which of the following is an advantage of using triggers in a DBMS?	A) Limited code organization	B) Increased code redundancy	C) Enhanced code isolation	D) Improved code modularity and reusability	D
3 3 2	Which of the following is an example of a parameterized trigger in a DBMS?	A trigger that fires after any insert operation	A trigger that fires after a specific date and time	A trigger that fires when a specific condition is met	<b>A trigger that accepts parameters passed from the calling application</b>	D
3 3 3	Which of the following is an example of an optimistic concurrency control technique in a DBMS?	Two-Phase Locking (2PL)	<b>Multi-Version Concurrency Control (MVCC)</b>	Time-stamp Ordering	Rollback Segments	B
3 3 4	Which of the following is not a level of data abstraction in a database system?	A) Physical level	B) Logical level	C) External level	D) Semantic level	D
3 3 5	Which of the following is NOT a property of a transaction in DBMS?	Atomicity	Consistency	Durability	Isolation	B
3 3 6	Which of the following is not a type of trigger in DBMS?	Insert trigger	Update trigger	Delete trigger	Search trigger	D
3 3 7	Which of the following is not an advantage of trigger?	Various column values are automatically generated by triggers	Maintains the integrity of referential	Tables are replicated asynchronously	Validating transactions and preventing them from being invalid	C
3 3 8	Which of the following is NOT an Oracle-supported trigger?	BEFORE	DURING	AFTER	INSTEAD OF	B



3 3 9	Which of the following is the correct format for if statement?	If boolean expression then statement or compound statement elseif boolean expression then statement or compound statement else statement or compound statement end if	If boolean expression then statement or compound statement elsif boolean expression then statement or compound statement else statement or compound statement end if	If boolean expression then statement or compound statement elif boolean expression then statement or compound statement else statement or compound statement end if	If boolean expression then statement or compound statement else statement or compound statement else statement or compound statement end if	A
3 4 0	Which of the following is true about compound triggers?	They can only be defined for tables, not views	They are fired once for each row affected by the triggering event	They cannot contain any SQL statements	They are not supported in DBMS	B
3 4 1	Which of the following is true about recursive triggers?	They are triggered by other triggers	They can only be fired once per event	They can cause an infinite loop if not handled properly	They are not supported in DBMS	C
3 4 2	Which of the following is true about stored procedures?	They can only return a single scalar value.	They can contain control-of-flow statements like IF and LOOP.	They cannot accept input parameters.	They are always automatically executed when the database starts.	B
3 4 3	Which of the following is TRUE about User-defined exceptions?	Users can explicitly raise an exception by using a RAISE statement	RAISE_APPLICATION_ERROR can be used to raise a user-defined exception explicitly	both 1 and 2	None of the above	C
3 4 4	Which of the following is used to input the entry and give the result in a variable in a procedure?	Put and get	Get and put	Out and In	In and out	D
3 4 5	Which of the following makes the transaction permanent in the database?	View	Commit	Rollback	Flashback	B
3 4 6	Which of the following specifies when the trigger will be executed?	BEFORE	AFTER	INSTEAD OF	All of the above	D

3 4 7	Which of the following statements best defines database recovery in DBMS?	The process of restoring data from backup tapes	The process of ensuring that the database remains secure	The process of restoring the database to a consistent state after a failure	The process of recovering deleted data from the Recycle Bin	C
3 4 8	Which of the following statements is true about First Normal Form (1NF)?	A) It allows for multivalued dependencies.	B) It allows for partial dependencies.	C) It eliminates repeating groups and ensures atomicity of data.	D) It enforces referential integrity constraints.	C
3 4 9	Which of the following statements is true about stored procedures?	Stored procedures cannot have input parameters	Stored procedures cannot return values	Stored procedures can be reused and shared by multiple applications	Stored procedures can only be executed by the database administrator	C
3 5 0	Which of the following statements is true about the Two-tier architecture?	A) It allows for better scalability than the Three-tier architecture.	B) It is easier to maintain and modify compared to the Three-tier architecture.	C) It requires less network traffic than the Three-tier architecture.	D) It provides better security and data isolation compared to the Three-tier architecture.	C
3 5 1	Which of the following statements is true regarding stored procedures?	Stored procedures always return a single value.	Stored procedures are not allowed to contain conditional statements.	<b>Stored procedures are precompiled and stored in the database for reuse.</b>	Stored procedures can only be executed by database administrators.	C
3 5 2	Which package lets PL/SQL programs read and write operating system (OS) text files?	UTL_HTTP	UTL_FILE	UTL_SMTP	UTL_FMT	B
3 5 3	Which part of a procedure in a DBMS is responsible for specifying the operations to be performed?	A) Procedure header	B) Procedure specification	C) Procedure body	D) Procedure parameters	C
3 5 4	Which property of a transaction ensures that it does not interfere with other transactions while executing?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	C
3 5 5	Which property of a transaction ensures that it does not interfere with other transactions while executing?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	C

3 5 6	Which property of a transaction ensures that it does not violate integrity constraints?	Isolation	Atomicity	Consistency	Durability	C
3 5 7	Which property of a transaction ensures that it either completes in its entirety or has no effect at all?	A) Atomicity	B) Optimistic concurrency control	C) Slower query performance	D) Data redundancy	A
3 5 8	Which property of a transaction ensures that the database remains in a consistent state after transaction execution?	A) Atomicity	B) Consistency	C) Isolation	D) Durability	B
3 5 9	Which recovery technique uses backward recovery to undo the changes made by a failed transaction?	Undo logging	Redo logging	Deferred update	Immediate update	A
3 6 0	Which specifies the column name that will be updated?	For col_name	ON col_name	OF col_name	WHEN col_name	C
3 6 1	Which type of database constraint ensures that a foreign key value matches a primary key value in another table?	A) Unique constraint	B) Primary key constraint	C) Foreign key constraint	D) Not null constraint	C
3 6 2	Which type of database trigger in SQL is executed before the triggering event occurs?	AFTER trigger	INSTEAD OF trigger	BEFORE trigger	FOR EACH ROW trigger	C
3 6 3	Which type of error occurs when the database crashes while a transaction is being executed?	System error	Media error	Transaction error	Operator error	A
3 6 4	Which type of trigger in a DBMS can be used to prevent changes to a table?	BEFORE trigger	AFTER trigger	<b>INSTEAD OF trigger</b>	FOR EACH ROW trigger	C

3 6 5	Which type of trigger in a DBMS is fired after a triggering event and can be used for auditing purposes?	BEFORE trigger	<b>AFTER trigger</b>	INSTEAD OF trigger	FOR EACH ROW trigger	B
3 6 6	Which type of view in PL/SQL allows you to update data directly through the view?	Materialized View	Read-Only View	Updatable View	Join View	C
3 6 7	Why is "concurrency control" important in a multi-user database environment?	A) To increase query performance	B) To ensure data consistency	C) To eliminate transactions	D) To optimize database storage	B
3 6 8	Why is "concurrency" a concern in a multi-user DBMS environment?	A) To simplify data retrieval	B) To ensure data consistency	C) To reduce query performance	D) To create redundant data	B
3 6 9	Why is concurrency control needed in a database management system (DBMS)?	A) To increase data redundancy	B) To slow down query execution	C) To ensure data consistency	D) To reduce code duplication	C
3 7 0	Write a PL/SQL function named <code>get_student_average_grade</code> that takes a student ID as input and returns the average grade of the specified student across all subjects. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_average_grade(stu dent_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_average_grade(stu dent_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_average_grade(stu dent_id NUMBER, avg_grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_average_gra de(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	A

3 7 1	Write a PL/SQL function named `get_student_count_by_subject` that takes a subject as input and returns the count of students enrolled in that subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_count_by_subject( subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_count_by_subject( subject VARCHAR2) RETURN NUMBER IS student_count NUMBER; BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_count_by_subject( subject VARCHAR2, student_count OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_count_by_su bject(subject VARCHAR2) IS BEGIN -- Procedure logic here END;	a
3 7 2	Write a PL/SQL function named `get_student_grade_in_subject` that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_su bject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A
3 7 3	Write a PL/SQL function named `get_student_grade_in_subject` that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_su bject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A

3 7 4	Write a PL/SQL function named `get_student_grade_in_subject` that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_su bject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A
3 7 5	Write a PL/SQL function named `get_student_grade_in_subject` that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_su bject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A
3 7 6	Write a PL/SQL function named `get_student_grade_in_subject` that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_subject( student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade_in_su bject(student_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A

3 7 7	Write a PL/SQL function named `get_student_grade` that takes a student ID and a subject as input and returns the grade of the specified student for the given subject. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_grade(student_id NUMBER, subject VARCHAR2) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_grade(student_id NUMBER, subject VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_grade(student_id NUMBER, subject VARCHAR2, grade OUT NUMBER) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_grade(studen t_id NUMBER, subject VARCHAR2) RETURN TABLE IS BEGIN -- Function logic here END;	A
3 7 8	Write a PL/SQL function named `get_student_info` that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_info(student _id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	C
3 7 9	Write a PL/SQL function named `get_student_info` that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_info(student _id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	C

380	Write a PL/SQL function named `get_student_info` that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_info(student _id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	C
381	Write a PL/SQL function named `get_student_info` that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_info(student _id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	C
382	Write a PL/SQL function named `get_student_info` that takes a student ID as input and returns the student's name, age, and the number of subjects they are enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_info(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_info(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_info( student_id NUMBER, student_name OUT VARCHAR2, student_age OUT NUMBER, subject_count OUT NUMBER ) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_info(student _id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	C



383	Write a PL/SQL function named `get_student_subject_scores` that takes a student ID as input and returns a cursor containing the subject and grade for all subjects in which the student is enrolled. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_subject_scores(stu dent_id NUMBER) RETURN CURSOR IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_subject_scores(stu dent_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_student_subject_scores( student_id NUMBER, subject_scores OUT SYS_REFCURSOR ) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_subject_scor es(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	C
384	Write a PL/SQL function named `get_student_subjects` that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_subjects(student_i d NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_subjects(student_i d NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_i d NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(stu dent_id NUMBER) IS BEGIN -- Procedure logic here END;	A
385	Write a PL/SQL function named `get_student_subjects` that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_subjects(student_i d NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_subjects(student_i d NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_i d NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(stu dent_id NUMBER) IS BEGIN -- Procedure logic here END;	A

3 8 6	Write a PL/SQL function named `get_student_subjects` that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER) IS BEGIN -- Procedure logic here END;	A
3 8 7	Write a PL/SQL function named `get_student_subjects` that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER) IS BEGIN -- Procedure logic here END;	A
3 8 8	Write a PL/SQL function named `get_student_subjects` that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_student_subjects(student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE get_student_subjects(student_id NUMBER) IS BEGIN -- Procedure logic here END;	A
3 8 9	Write a PL/SQL function named `get_subjects_by_student` that takes a student ID as input and returns a list of subjects that the student is enrolled in. Which of the following code snippets correctly defines this function?	CREATE OR REPLACE FUNCTION get_subjects_by_student(student_id NUMBER) RETURN VARCHAR2 IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE get_subjects_by_student(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION get_subjects_by_student(student_id NUMBER, subjects OUT VARCHAR2) IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION get_subjects_by_student(student_id NUMBER) RETURN TABLE IS BEGIN -- Function logic here END;	A

390	Write a PL/SQL procedure named `add_student_subject` that takes a student ID, subject, and grade as input and adds a new subject enrollment record for the specified student in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
391	Write a PL/SQL procedure named `add_student_subject` that takes a student ID, subject, and grade as input and adds a new subject enrollment record for the specified student in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE add_student_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
392	Write a PL/SQL procedure named `calculate_avg_grade` that takes a student ID as input and calculates the average grade of that student across all subjects. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE PROCEDURE calculate_avg_grade(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE calculate_avg_grade(student_id NUMBER, avg_grade OUT NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE calculate_avg_grade(student_id NUMBER) AS avg_grade NUMBER; BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION calculate_avg_grade(stud ent_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	B

393	Write a PL/SQL procedure named `calculate_student_average` that calculates the average grade for a specific student identified by their `student_id` and stores it in a variable. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE PROCEDURE calculate_student_average(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE calculate_student_average(student_id NUMBER, avg_grade OUT NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION calculate_student_average(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE FUNCTION calculate_student_average(student_id NUMBER, avg_grade OUT NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	B
394	Write a PL/SQL procedure named `delete_student_record` that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
395	Write a PL/SQL procedure named `delete_student_record` that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B

3 9 6	Write a PL/SQL procedure named `delete_student_record` that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
3 9 7	Write a PL/SQL procedure named `delete_student_record` that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
3 9 8	Write a PL/SQL procedure named `delete_student_record` that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student_record(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B

399	Write a PL/SQL procedure named `delete_student` that takes a student ID as input and deletes the corresponding student record from the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION delete_student(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE delete_student(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE delete_student(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
400	Write a PL/SQL procedure named `enroll_student_in_subject` that takes a student ID, subject, and grade as input and inserts a new enrollment record for the specified student in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION enroll_student_in_subject(student_id NUMBER, subject VARCHAR2, grade NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE enroll_student_in_subject(student_id NUMBER, subject VARCHAR2, grade NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE enroll_student_in_subject(student_id NUMBER, subject VARCHAR2, grade NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE enroll_student_in_subject( student_id NUMBER, subject VARCHAR2, grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
401	Write a PL/SQL procedure named `enroll_student` that takes student details (name, age, subject, grade) as input and inserts a new record into the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION enroll_student(student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE enroll_student(student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE enroll_student(student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE enroll_student(student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	b

402	Write a PL/SQL procedure named `insert_student_record` that takes student details (name, age, subject, grade) as input and inserts a new record into the "student" table. Additionally, it should return the newly generated student ID. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS student_id NUMBER; BEGIN -- Function logic here END;	B
403	Write a PL/SQL procedure named `insert_student_record` that takes student details (name, age, subject, grade) as input and inserts a new record into the "student" table. Additionally, it should return the newly generated student ID. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION insert_student_record( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS student_id NUMBER; BEGIN -- Function logic here END;	B

404	Write a PL/SQL procedure named `insert_student` that takes student details (name, age, subject, grade) as input and inserts a new record into the "student" table. Additionally, it should return the newly generated student ID. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION insert_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE insert_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE insert_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER, student_id OUT NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE FUNCTION insert_student( student_name VARCHAR2, student_age NUMBER, subject VARCHAR2, student_grade NUMBER ) RETURN NUMBER IS student_id NUMBER; BEGIN -- Function logic here END;	B
405	Write a PL/SQL procedure named `update_student_grade` that takes a student ID, subject, and a new grade as input and updates the grade of the specified student for the given subject. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
406	Write a PL/SQL procedure named `update_student_record` that takes a student ID as input and updates the student's name, age, and grade in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_record(studen t_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(studen t_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(studen t_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(st udent_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B



407	Write a PL/SQL procedure named `update_student_record` that takes a student ID as input and updates the student's name, age, and grade in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_record(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
408	Write a PL/SQL procedure named `update_student_record` that takes a student ID as input and updates the student's name, age, and grade in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_record(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
409	Write a PL/SQL procedure named `update_student_record` that takes a student ID as input and updates the student's name, age, and grade in the "student" table. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_record(student_id NUMBER) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_record(student_id NUMBER) IS -- Declare variables here BEGIN -- Procedure logic here END;	B

410	Write a PL/SQL procedure named `update_student_subject_grade` that takes a student ID, subject, and a new grade as input and updates the grade of the specified student for the given subject. Which of the following code snippets correctly defines this procedure?	CREATE OR REPLACE FUNCTION update_student_subject_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) RETURN NUMBER IS BEGIN -- Function logic here END;	CREATE OR REPLACE PROCEDURE update_student_subject_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) IS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_subject_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) AS BEGIN -- Procedure logic here END;	CREATE OR REPLACE PROCEDURE update_student_subject_grade( student_id NUMBER, subject VARCHAR2, new_grade NUMBER ) IS -- Declare variables here BEGIN -- Procedure logic here END;	B
411	You have a stored procedure named UpdateEmployeeSalary that accepts an employee ID and a salary value as parameters and updates the employee's salary in the database. Which SQL statement would you use to execute this stored procedure with employee ID 101 and a new salary of 55000?	<b>EXEC UpdateEmployeeSalary 101, 55000;</b>	CALL UpdateEmployeeSalary(101, 55000);	RUN UpdateEmployeeSalary(101, 55000);	UPDATE EmployeeSalary(101, 55000);	A
412	You have a stored procedure that calculates the average salary of employees in a specific department. Which SQL statement do you use to execute this stored procedure and retrieve the result?	<b>EXEC GetAverageSalaryForDepartment 101;</b>	CALL GetAverageSalaryForDepartment(101);	EXEC GetAverageSalaryForDepartment @DepartmentID = 101;	EXEC GetAverageSalaryForDepartment @DepartmentID = 101, @Result = OUTPUT;	A

4 1 3	You have a table named "Customers" with a "LastPurchaseDate" column. You want to create a trigger that updates the "LastPurchaseDate" column to the current date whenever a new purchase is made by a customer. What type of trigger should you create?	<b>AFTER INSERT</b>	BEFORE INSERT	AFTER UPDATE	INSTEAD OF INSERT	A
4 1 4	You have a table named "Inventory" with a "Quantity" column. You want to create a trigger that automatically updates the "Quantity" column to zero when a product is marked as "Out of Stock." What type of trigger should you create?	<b>AFTER UPDATE</b>	BEFORE UPDATE	AFTER INSERT	INSTEAD OF UPDATE	A
4 1 5	You want to create a stored procedure that inserts a new customer record into the "Customers" table. The customer's name, email, and phone number will be passed as parameters. Which SQL statement creates this stored procedure?	<b>CREATE PROCEDURE</b> <b>InsertCustomer</b> <b>@Name VARCHAR(50), @Email VARCHAR(100), @Phone VARCHAR(20)</b> <b>AS</b> <b>BEGIN</b> <b>INSERT INTO Customers</b> <b>(Name, Email, Phone) VALUES</b> <b>(@Name, @Email, @Phone)</b> <b>END</b>	<b>CREATE PROCEDURE</b> InsertCustomer AS BEGIN INSERT INTO Customers (Name, Email, Phone) VALUES (@Name, @Email, @Phone) END	<b>CREATE PROCEDURE</b> InsertCustomer @CustomerData VARCHAR(MAX) AS BEGIN INSERT INTO Customers (Name, Email, Phone) VALUES (@CustomerData) END	<b>CREATE PROCEDURE</b> InsertCustomer @Name VARCHAR(50), @Email VARCHAR(100), @Phone VARCHAR(20) AS BEGIN INSERT INTO Customers (Name, Email, Phone) VALUES (@Name, @Email, @Phone) END	A

1. What is a stored program unit in a database?

- a. Table
- b. Procedure
- c. Trigger
- d. View

Answer: b. Procedure

2. Which of the following is not a part of a procedure in a database?

- a. Declaration section
- b. Exception section
- c. Body section
- d. Trigger section

Answer: d. Trigger section

3. In parameter modes for procedures, which mode allows you to pass values from the calling program to the procedure and vice versa?

- a. IN
- b. OUT
- c. IN OUT
- d. DEFAULT

Answer: c. IN OUT

4. What is one of the advantages of using procedures in a database?

- a. Simplified data modeling
- b. Enhanced security
- c. Dynamic table creation
- d. Reduced data redundancy

Answer: b. Enhanced security

5. Which of the following is not a type of trigger in a database?

- a. Before Trigger
- b. After Trigger
- c. During Trigger
- d. Instead Of Trigger

Answer: c. During Trigger

6. What is the syntax for creating a trigger in SQL?

- a. CREATE PROCEDURE
- b. CREATE FUNCTION
- c. CREATE TRIGGER
- d. CREATE VIEW

Answer: c. CREATE TRIGGER

7. Which type of trigger is fired before the execution of a DML statement in SQL?

- a. Before Trigger
- b. After Trigger
- c. Instead Of Trigger
- d. Concurrent Trigger

Answer: a. Before Trigger

8. What is the purpose of a package specification in a database?

- a. Contains the implementation details of a package
- b. Declares the public interface of a package
- c. Stores data for the package
- d. Executes procedures in the package

Answer: b. Declares the public interface of a package

9. Which part of a package in a database contains the actual code and implementation details?

- a. Package specification
- b. Package body
- c. Package header
- d. Package declaration

Answer: b. Package body

10. What is a "bodiless package" in a database?

- a. A package without a package specification
- b. A package without a package body
- c. A package without any code or implementation
- d. A package without parameters

Answer: b. A package without a package body

11. Which of the following is not an advantage of using packages in a database?

- a. Encapsulation of code
- b. Improved performance
- c. Simplified maintenance
- d. Increased data redundancy

Answer: d. Increased data redundancy

12. What does a "PRAGMA AUTONOMOUS\_TRANSACTION" do in a trigger?

- a. Executes the trigger automatically
- b. Is used to create a trigger
- c. Starts a new transaction within the trigger

d. Stops a running transaction

Answer: c. Starts a new transaction within the trigger

13. Which of the following is a valid parameter mode for a procedure in SQL?

a. IN OUT INCREMENT

b. OUT DECREMENT

c. INCREMENT OUT

d. OUT IN

Answer: a. IN OUT INCREMENT

14. What type of trigger is used to replace a DML statement with another statement?

a. Before Trigger

b. After Trigger

c. Instead Of Trigger

d. Around Trigger

Answer: c. Instead Of Trigger

15. What is the primary purpose of a trigger in a database?

a. To define data types

b. To declare variables



- c. To enforce data integrity and automate actions
- d. To create views

Answer: c. To enforce data integrity and automate actions

16. Which of the following is not a valid part of a procedure in SQL?

- a. Parameter list
- b. Declaration section
- c. Exception section
- d. Trigger section

Answer: d. Trigger section

17. What is the primary purpose of a package in SQL?

- a. To store data
- b. To define triggers
- c. To group related procedures, functions, and variables
- d. To create views

Answer: c. To group related procedures, functions, and variables

18. Which parameter mode allows a procedure to receive values from the calling program but not return values back?

- a. IN

- b. OUT
- c. IN OUT
- d. DEFAULT

Answer: a. IN

19. Which of the following is not a type of package in SQL?

- a. Standalone package
- b. Bodiless package
- c. Composite package
- d. Nested package

Answer: c. Composite package

20. What is the advantage of using triggers in a database?

- a. Improved code encapsulation
- b. Enhanced security
- c. Simplified package creation
- d. Dynamic table creation

Answer: b. Enhanced security

1. What is the primary purpose of a transaction in a DBMS?

- A. To retrieve data from the database
- B. To update the database
- C. To manage database schemas
- D. To organize data into tables

Answer: B

2. Which of the following is not a property of a transaction in DBMS?

- A. Isolation
- B. Atomicity
- C. Consistency
- D. Transparency

Answer: D

3. Which property of transactions ensures that a transaction's changes are permanent and will survive system failures?

- A. Atomicity
- B. Consistency
- C. Isolation
- D. Durability

Answer: D

4. What does the ACID acronym stand for in the context of transactions?

- A. Atomicity, Completeness, Isolation, Durability
- B. Atomicity, Consistency, Isolation, Durability
- C. Availability, Consistency, Isolation, Durability
- D. Allotment, Concurrency, Integration, Durability

Answer: B

5. Which of the following is not a common concurrency problem in DBMS?

- A. Deadlock
- B. Dirty Read
- C. Lost Update
- D. Normalization

Answer: D

6. What is the primary goal of a concurrency control mechanism in a DBMS?

- A. To ensure that transactions are executed concurrently without any restrictions
- B. To improve the performance of the database system
- C. To prevent conflicts and maintain data consistency in a multi-user environment
- D. To reduce the storage space required for the database

Answer: C

7. Which of the following is a common method for achieving isolation in DBMS?

- A. Two-Phase Locking
- B. Time-based Synchronization
- C. Data Duplication
- D. Data Sharding

Answer: A

8. What is a serializable schedule in the context of concurrency control?

- A. A schedule in which transactions are executed one after the other
- B. A schedule that preserves the original order of transactions
- C. A schedule that produces the same result as if transactions were executed serially
- D. A schedule that allows concurrent execution without any restrictions

Answer: C

9. Which of the following is a benefit of using serializability in a DBMS?

- A. Improved system performance
- B. Reduced data consistency
- C. Enhanced data integrity
- D. Faster query processing

Answer: C

10. What is the primary purpose of a lock in a DBMS?

- A. To restrict access to specific data items
- B. To encrypt data for security
- C. To optimize query execution
- D. To permanently delete data

Answer: A

11. In a DBMS, what is a shared lock used for?

- A. To allow multiple transactions to write to the same data item
- B. To prevent multiple transactions from reading the same data item simultaneously
- C. To allow multiple transactions to read the same data item simultaneously
- D. To break deadlocks

Answer: C

12. Which concurrency control technique uses a timeout mechanism to resolve conflicts?

- A. Two-Phase Locking
- B. Timestamp-Based Protocol
- C. Optimistic Concurrency Control
- D. Strict Two-Phase Locking

Answer: C

13. What is a deadlock in the context of concurrency control?

- A. A situation where two transactions are waiting for each other to release locks
- B. A situation where a transaction is permanently blocked
- C. A situation where transactions cannot be rolled back
- D. A situation where transactions cannot be committed

Answer: A

14. What is the purpose of a deadlock detection mechanism in a DBMS?

- A. To prevent deadlocks from occurring
- B. To identify and resolve deadlocks
- C. To escalate conflicts between transactions
- D. To increase the isolation level

Answer: B

15. Which of the following is not a common deadlock prevention technique?

- A. Wait-Die
- B. Wound-Wait
- C. Timeout

D. Rollback

Answer: C

16. What is the purpose of an intent lock in a DBMS?

- A. To indicate the intention of a transaction to acquire a shared lock
- B. To indicate the intention of a transaction to acquire an exclusive lock
- C. To prevent deadlocks
- D. To release all locks

Answer: B

17. In a DBMS, what is the purpose of a transaction log?

- A. To record all user queries
- B. To store database schema information
- C. To maintain a record of all committed and uncommitted transactions
- D. To store backup copies of data

Answer: C

18. What is the primary goal of a checkpoint in a DBMS?

- A. To initiate a transaction rollback
- B. To recover from system crashes



- C. To release all locks held by a transaction
- D. To optimize query execution

Answer: B

19. Which of the following is an example of a conflict-serializable schedule in a DBMS?

- A. Schedule S1:  $T1 \rightarrow T2 \rightarrow T3$
- B. Schedule S2:  $T2 \rightarrow T1 \rightarrow T3$
- C. Schedule S3:  $T1 \rightarrow T3 \rightarrow T2$
- D. Schedule S4:  $T3 \rightarrow T1 \rightarrow T2$

Answer: A

20. What does the isolation level "Serializable" in a DBMS ensure?

- A. It allows dirty reads
- B. It provides the highest level of isolation
- C. It allows transactions to write to the same data simultaneously
- D. It does not allow any concurrency

Answer: B

21. Which of the following is a benefit of using a lower isolation level, such as "Read Uncommitted," in a DBMS?

- A. Improved data integrity

- B. Higher isolation between transactions
- C. Reduced contention for locks
- D. Faster query performance

Answer: D

22. In the context of locking, what is a lock mode?

- A. The time duration for which a lock is held
- B. The type of lock (shared or exclusive) and its compatibility with other locks
- C. The order in which locks are acquired
- D. The number of transactions waiting for a lock

Answer: B

23. Which of the following is a drawback of using a high isolation level, such as "Serializable," in a DBMS?

- A. Increased likelihood of deadlocks
- B. Improved data consistency
- C. Lower transaction throughput
- D. Reduced data integrity

Answer: C

24. What is the purpose of a transaction manager in a DBMS?

- A. To optimize query execution
- B. To manage database schemas
- C. To ensure the ACID properties of transactions
- D. To store backup copies of data

Answer: C

25. What is the primary goal

of a deadlock prevention technique like "Wait-Die"?

- A. To escalate conflicts between transactions
- B. To prevent transactions from waiting indefinitely
- C. To improve query performance
- D. To increase data redundancy

Answer: B

26. In a DBMS, what is a transaction's isolation level?

- A. The number of locks acquired by the transaction
- B. The duration for which a transaction is active
- C. The level of visibility a transaction has into other transactions' changes
- D. The number of concurrent transactions

Answer: C

27. Which of the following is a disadvantage of using optimistic concurrency control?

- A. Higher contention for locks
- B. Increased likelihood of deadlocks
- C. Slower query performance
- D. Limited data consistency

Answer: D

28. What is the purpose of a timestamp in a DBMS?

- A. To record the time when a transaction started
- B. To ensure data encryption
- C. To prevent conflicts between transactions
- D. To optimize query execution

Answer: A

29. What is a conflict-serializable schedule?

- A. A schedule that contains conflicts between transactions
- B. A schedule in which transactions are executed serially
- C. A schedule that preserves the original order of transactions
- D. A schedule that is equivalent to a serial schedule with the same transactions

Answer: D

30. Which of the following is not a common concurrency control mechanism in a DBMS?

- A. Two-Phase Commit
- B. Optimistic Concurrency Control
- C. Strict Two-Phase Locking
- D. Timestamp-Based Protocol

Answer: A

31. What is the purpose of a transaction ID in a DBMS?

- A. To identify the user who initiated the transaction
- B. To indicate the transaction's priority
- C. To uniquely identify and track each transaction
- D. To store backup copies of data

Answer: C

32. Which of the following is a benefit of using a higher isolation level, such as "Serializable," in a DBMS?

- A. Improved query performance
- B. Reduced likelihood of deadlocks
- C. Higher data consistency

D. Lower transaction throughput

Answer: C

33. What is the primary goal of a timeout-based deadlock prevention technique?

- A. To prevent transactions from waiting indefinitely
- B. To prioritize transactions based on their importance
- C. To escalate conflicts between transactions
- D. To increase the isolation level

Answer: A

34. What does a "lost update" refer to in the context of concurrency control?

- A. A situation where a transaction is permanently blocked
- B. A situation where a transaction is rolled back
- C. A situation where one transaction overwrites the changes made by another transaction
- D. A situation where transactions cannot be committed

Answer: C

35. Which of the following is not a common technique for deadlock detection in a DBMS?

- A. Wait-Die
- B. Wound-Wait

C. Timeout

D. Rollback

Answer: D

36. What is the primary purpose of a deadlock prevention technique like "Wound-Wait"?

A. To escalate conflicts between transactions

B. To prevent transactions from waiting indefinitely

C. To improve query performance

D. To increase data redundancy

Answer: A

37. Which of the following statements about the "Repeatable Read" isolation level in a DBMS is true?

A. It allows dirty reads.

B. It allows lost updates.

C. It prevents phantom reads.

D. It has the lowest isolation level.

Answer: C

38. What is the primary goal of a transaction recovery manager in a DBMS?

A. To escalate conflicts between transactions

- B. To optimize query execution
- C. To ensure the durability of transactions
- D. To manage database schemas

Answer: C

39. Which of the following is not a common cause of deadlocks in a DBMS?

- A. Circular Wait
- B. Resource Preemption
- C. Hold and Wait
- D. No Concurrency

Answer: D

40. What is the purpose of a data dictionary in a DBMS?

- A. To store user data
- B. To maintain a log of transactions
- C. To manage database schemas and metadata
- D. To perform data encryption

Answer: C

**Assuming we have a "Bank" table with the following sample data:**



account_number	account_holder	balance
1001	John Doe	5000.00
1002	Jane Smith	7500.50
1003	Alice Johnson	3000.25

```

```sql
CREATE TABLE Bank (
    account_number NUMBER PRIMARY KEY,
    account_holder VARCHAR2(100),
    balance NUMBER(10, 2)
);

```

Here's a PL/SQL code snippet based on this data:

PL/SQL Code Snippet:

```

```plsql
DECLARE
    v_balance NUMBER;

```

```
BEGIN

    SELECT balance

    INTO v_balance

    FROM Bank

    WHERE account_holder = 'John Doe';

    DBMS_OUTPUT.PUT_LINE('John Doe\'s Balance: $' || v_balance);

END;

---
```

**Based on above details gives the following question's answer**

1. What is the primary purpose of the PL/SQL code snippet?

- a) Updates John Doe's account balance
- b) Deletes John Doe's account record
- c) Retrieves and displays John Doe's account balance
- d) Inserts a new account record for John Doe

Answer: c) Retrieves and displays John Doe's account balance

2. What is the data type of the "balance" column in the "Bank" table?

- a) String
- b) Date
- c) Number
- d) Boolean

Answer: c) Number

3. Which SQL operation is performed in the PL/SQL code?

- a) INSERT
- b) DELETE
- c) SELECT
- d) UPDATE

Answer: c) SELECT

4. What is the purpose of the `INTO` clause in the code?

- a) To indicate the end of the PL/SQL block
- b) To declare a new variable
- c) To specify the source of data for the SELECT statement
- d) To define a cursor

Answer: c) To specify the source of data for the SELECT statement

5. What does the `DBMS\_OUTPUT.PUT\_LINE` statement do in the code?

- a) Updates the database records
- b) Deletes database records
- c) Retrieves data from the database
- d) Displays a message in the console

Answer: d) Displays a message in the console

here are 10 multiple-choice questions (MCQs) based on a PL/SQL code snippet

PL/SQL Code Snippet:

```
```sql
DECLARE
    v_employee_count NUMBER;
BEGIN
    SELECT COUNT() INTO v_employee_count
    FROM Employees;
    DBMS_OUTPUT.PUT_LINE('Total Employees: ' || v_employee_count);
END;
```
```

**MCQs:**

1. What is the primary purpose of the PL/SQL code snippet?
  - a) Updates employee records
  - b) Deletes employee records
  - c) Retrieves and displays the total number of employees
  - d) Inserts a new employee record

Answer: c) Retrieves and displays the total number of employees

2. In the code snippet, what is the value stored in the `v\_employee\_count` variable?

- a) Employee names
- b) Employee IDs
- c) Total number of employees
- d) Employee salaries

Answer: c) Total number of employees

3. Which SQL operation is performed in the PL/SQL code?

- a) INSERT
- b) DELETE
- c) SELECT
- d) UPDATE

Answer: c) SELECT

4. What is the purpose of the `INTO` clause in the code?

- a) To indicate the end of the PL/SQL block
- b) To declare a new variable
- c) To specify the source of data for the SELECT statement
- d) To define a cursor

Answer: c) To specify the source of data for the SELECT statement

5. What does the `DBMS\_OUTPUT.PUT\_LINE` statement do in the code?

- a) Updates the database records
- b) Deletes database records
- c) Retrieves data from the database
- d) Displays a message in the console

Answer: d) Displays a message in the console

6. Which PL/SQL construct allows you to handle exceptions in a structured manner?

- a) TRY-CATCH
- b) EXCEPTION
- c) ERROR-HANDLER
- d) ON-ERROR

Answer: b) EXCEPTION

7. In PL/SQL, what is the primary purpose of a cursor?

- a) To define variables
- b) To loop through a result set
- c) To declare procedures
- d) To manage transactions

Answer: b) To loop through a result set

8. What is the expected output of the code snippet if there are 100 employees in the "Employees" table?

- a) Total Employees: 100

- b) Total Employees: 0
- c) Total Employees: 1
- d) Total Employees: 99

Answer: a) Total Employees: 100

9. What type of variable is `v\_employee\_count` in the code snippet?

- a) String
- b) Date
- c) Number
- d) Boolean

Answer: c) Number

10. In PL/SQL, how can you pass a parameter to a stored procedure?

- a) Using a RETURN statement
- b) Using a SELECT statement
- c) Using an IN parameter
- d) Using a WHERE clause

Answer: c) Using an IN parameter

**Here's a PL/SQL package with a "College" table and some basic code snippets :**

```
```sql
```

```
-- Create the College table
```

```
CREATE TABLE College (  
    student_id NUMBER PRIMARY KEY,  
    student_name VARCHAR2(50),  
    major VARCHAR2(50)  
);
```

```
+-----+-----+-----+  
| student_id | student_name | major |  
+-----+-----+-----+  
| 1 | John Smith | Computer Science |  
| 2 | Jane Doe | Biology |  
| 3 | Alice Johnson | History |  
| 4 | Bob Brown | Mathematics |  
| 5 | Eva Williams | Chemistry |  
+-----+-----+-----+
```

```
```psql
```



```
-- Create a PL/SQL package
CREATE OR REPLACE PACKAGE College_Package AS

    -- Function to retrieve student count by major
    FUNCTION getStudentCountByMajor(major IN VARCHAR2) RETURN NUMBER;

    FUNCTION mcq1 RETURN VARCHAR2;

    FUNCTION mcq2 RETURN NUMBER;

END College_Package;
/
```

```
CREATE OR REPLACE PACKAGE BODY College_Package AS

    -- Function to retrieve student count by major
    FUNCTION getStudentCountByMajor(major IN VARCHAR2) RETURN NUMBER IS
        cnt NUMBER;
    BEGIN
        SELECT COUNT() INTO cnt FROM College WHERE major = major;
        RETURN cnt;
    END;

    FUNCTION mcq1 RETURN VARCHAR2 IS
    BEGIN
        RETURN 'student_id';
    END;

    FUNCTION mcq2 RETURN NUMBER IS

        biology_count NUMBER;

    BEGIN

        biology_count := getStudentCountByMajor('Biology');

        RETURN biology_count;

    END;
```

END College\_Package;

/

...

MCQ 1: Which column is used to uniquely identify students?

A) student\_id

B) student\_name

C) major

D) None of the above

Answer: A) student\_id

MCQ 2: How many students are majoring in Computer Science?

A) 1

B) 2

C) 3

D) 0

Answer: A) 1

MCQ 3: What is the data type of the "student\_name" column in the College table?

A) NUMBER

B) VARCHAR2

C) DATE

D) BOOLEAN

Answer: B) VARCHAR2

MCQ 4: Which PL/SQL construct is used to loop through records in a result set?

A) FOR loop

B) IF statement

C) WHILE loop

D) CASE statement

Answer: A) FOR loop

MCQ 5: How many students are majoring in Chemistry?

A) 1

B) 2

C) 3

D) 0

Answer: D) 0

MCQ 6: Which PL/SQL keyword is used to declare a variable?

A) DEFINE

B) DECLARE

C) VARIABLE

D) SET

Answer: B) DECLARE

MCQ 7: What is the output of the following PL/SQL code?

```
```sql
```

```
DECLARE
```

```
    total_students NUMBER;
```

```
BEGIN
```

```
    total_students := College_Package.getStudentCountByMajor('Computer Science');
```

```
    DBMS_OUTPUT.PUT_LINE('Total students in Computer Science: ' || total_students);
```

```
END;
```

```
```
```

A) Total students in Computer Science: 1

B) Total students in Computer Science: 2

C) Total students in Computer Science: 3

D) Total students in Computer Science: 0

Answer: A) Total students in Computer Science: 1

MCQ 8: Which PL/SQL statement is used to raise an exception?

A) RAISE

B) THROW

C) EXCEPTION

D) ERROR

Answer: A) RAISE

MCQ 9: What is the purpose of the PRIMARY KEY constraint in the College table?

A) It enforces unique values in the "student\_name" column.

B) It enforces unique values in the "major" column.

C) It ensures that the "student\_id" column is not null.

D) It uniquely identifies each row in the table.

Answer: D) It uniquely identifies each row in the table.

MCQ 10: Which PL/SQL construct is used to handle exceptions in a controlled manner?

A) TRY...CATCH block

B) EXCEPTION block

C) ERROR block

D) HANDLE block

Answer: B) EXCEPTION block

**Here's a PL/SQL code snippet for a hypothetical "hospital" table, along with 10 multiple-choice questions (MCQs)**

Let's create a PL/SQL trigger for the "hospital" table. This trigger updates the "patient\_count" column in a separate "hospital\_stats" table whenever a new patient is inserted into the "hospital" table.

```
```sql
```

-- Create the hospital\_stats table to store statistics.

```
CREATE TABLE hospital_stats (  
    total_patients NUMBER  
);
```

-- Create a sequence to generate unique IDs for each patient.

```
CREATE SEQUENCE patient_id_seq START WITH 1;
```

-- Create the hospital table.

```
CREATE TABLE hospital (  
    patient_id NUMBER PRIMARY KEY,  
    patient_name VARCHAR2(50),  
    admission_date DATE,  
    discharge_date DATE  
);
```

-- Create the trigger to update patient count in hospital\_stats.

```
CREATE OR REPLACE TRIGGER update_patient_count  
AFTER INSERT ON hospital  
FOR EACH ROW  
BEGIN  
    UPDATE hospital_stats  
    SET total_patients = total_patients + 1;
```

END;

/

...

### Multiple-Choice Questions (MCQs):

1. What is the purpose of the "update\_patient\_count" trigger in the "hospital" table?

- a) To automatically update all patient records.
- b) To update the total count of patients in the "hospital\_stats" table when a new patient is inserted.
- c) To prevent new records from being inserted.
- d) To calculate the average length of stay for all patients.

Correct Answer: b

2. In which event(s) will the "update\_patient\_count" trigger execute?

- a) Before inserting a new patient record.
- b) After deleting a patient record.
- c) Before updating an existing patient record.
- d) After inserting a new patient record.

Correct Answer: d

3. What does `AFTER INSERT ON hospital` mean in the trigger definition?

- a) The trigger fires before a new patient record is inserted.
- b) The trigger fires after a patient record is deleted.
- c) The trigger fires after a new patient record is inserted.
- d) The trigger fires before an existing patient record is updated.

Correct Answer: c

4. What is the purpose of the "hospital\_stats" table in the code snippet?

- a) To store patient names.
- b) To store admission and discharge dates.
- c) To store statistics related to the hospital, such as the total number of patients.
- d) To store the patient IDs.

Correct Answer: c

5. How is the "patient\_id" assigned in the "hospital" table?

- a) Manually entered by the user.
- b) Generated automatically using a sequence.
- c) Copied from the "patient\_id" in the "hospital\_stats" table.
- d) Set to a constant value.

Correct Answer: b



6. What happens if you attempt to insert a new patient record without specifying values for "patient\_name," "admission\_date," and "discharge\_date"?

- a) The trigger inserts default values.
- b) The trigger raises an error.
- c) The trigger inserts NULL values.
- d) The trigger generates random values.

Correct Answer: b

7. Which keyword is used to specify the trigger action timing in PL/SQL?

- a) WHEN
- b) BEFORE
- c) AFTER
- d) TRIGGER

Correct Answer: c

8. What is the primary purpose of the ``UPDATE hospital_stats SET total_patients = total_patients + 1;`` statement in the trigger?

- a) To delete a patient record.
- b) To insert a new patient record.
- c) To update the "patient\_count" column in the "hospital\_stats" table.
- d) To calculate the average length of stay for all patients.

Correct Answer: c

9. Can you have multiple triggers with the same timing (e.g., AFTER INSERT) on the same table?

- a) No, only one trigger is allowed per table.
- b) Yes, but they must have different names.
- c) Yes, and they execute in a random order.
- d) No, it will result in an error.

Correct Answer: b

10. What does the `CREATE SEQUENCE patient_id_seq START WITH 1;` statement do in the code snippet?

- a) It creates a new table.
- b) It defines a new trigger.
- c) It creates a sequence for generating unique patient IDs.
- d) It initializes the patient ID to 1.

Correct Answer: c

A PL/SQL procedure that takes two numbers as input parameters, adds them together, and then displays the result using dbms\_output:

```
```sql
```

```
CREATE OR REPLACE PROCEDURE add_numbers (  
    p_num1 IN NUMBER,  
    p_num2 IN NUMBER  
) AS  
    v_result NUMBER;  
BEGIN  
    -- Perform the addition  
    v_result := p_num1 + p_num2;  
  
    -- Display the result  
    DBMS_OUTPUT.PUT_LINE('The sum of ' || p_num1 || ' and ' || p_num2 || ' is ' || v_result);  
END add_numbers;  
  
/  
```
```

Here's an example of how to call this procedure:

```
```sql
```

```
DECLARE  
    num1 NUMBER := 10;
```

```
    num2 NUMBER := 20;  
  
BEGIN  
  
    add_numbers(num1, num2);  
  
END;  
  
/  
...
```

This will call the `add\_numbers` procedure with `num1` and `num2` as arguments and display the sum.

**Based on given pl sql answer the following mcq:**

1. What is the purpose of the PL/SQL procedure mentioned in the code snippet?

- A. To subtract two numbers.
- B. To add two numbers and display the result.
- C. To multiply two numbers.
- D. To divide two numbers.

Answer: B

2. How many input parameters does the `add\_numbers` procedure have?

- A. None
- B. One
- C. Two
- D. Three

Answer: C

3. What data type are the input parameters `p\_num1` and `p\_num2` in the `add\_numbers` procedure?

- A. VARCHAR2
- B. DATE
- C. NUMBER
- D. BOOLEAN

Answer: C

4. What is the purpose of the `DBMS\_OUTPUT.PUT\_LINE` statement in the procedure?

- A. It calculates the sum of two numbers.
- B. It displays the result of the addition.
- C. It defines a new variable.
- D. It retrieves data from the database.

Answer: B

5. How is the result of the addition operation displayed in the output?

- A. Using the PRINT statement
- B. Using the RETURN statement
- C. Using the DBMS\_OUTPUT.PUT\_LINE statement
- D. Using the DISPLAY statement

Answer: C

6. What should you do to call the `add\_numbers` procedure with specific numbers as arguments?

- A. Use the CALL statement.
- B. Use the SELECT statement.
- C. Use the DECLARE block.
- D. Use the EXECUTE statement.

Answer: C

7. In the example provided for calling the procedure, what are the values of `num1` and `num2`?

- A. num1 = 20, num2 = 10
- B. num1 = 10, num2 = 30
- C. num1 = 10, num2 = 20
- D. num1 = 30, num2 = 10

Answer: C

8. What is the result of calling the `add\_numbers` procedure with `num1` and `num2` as arguments in the example?

- A. 10
- B. 20
- C. The sum of 10 and 20 is 30
- D. There will be no output.

Answer: C

9. Which SQL statement is used to create a PL/SQL procedure?

- A. CREATE PROCEDURE
- B. DECLARE PROCEDURE
- C. EXECUTE PROCEDURE
- D. CALL PROCEDURE

Answer: A

10. What is the purpose of the `DECLARE` block in the example?

- A. To define a new variable.
- B. To execute SQL statements.
- C. To declare and initialize variables before calling the procedure.
- D. To declare a function.

Answer: C

Ques1 - Consider the following PL/SQL function: (Difficulty level – Easy)

```
CREATE OR REPLACE FUNCTION calculate_total(price NUMBER, quantity NUMBER)
RETURN NUMBER IS
    total NUMBER;
BEGIN
    total := price * quantity;
    RETURN total;
END;
```

What does the PL/SQL function `calculate\_total` do?

- A. It calculates the average of `price` and `quantity`.

- B. It calculates the sum of `price` and `quantity`.
- C. It calculates the product of `price` and `quantity`.
- D. It calculates the difference between `price` and `quantity`.

Correct Option: C

---

Ques2 - Consider the following PL/SQL function: (Difficulty level – Easy)

```
CREATE OR REPLACE FUNCTION greet(name VARCHAR2)
RETURN VARCHAR2 IS
    greeting VARCHAR2(100);
BEGIN
    greeting := 'Hello, ' || name || '!';
    RETURN greeting;
END;
```

**\*\*What does the PL/SQL function `greet` do?\*\***

- A. It calculates the length of the input string `name`.
- B. It calculates the square of a numeric input.
- C. It generates a greeting message with the input `name`.
- D. It calculates the factorial of a numeric input.

**\*\*Correct Option:\*\* C**

---

Ques3 - Consider the following PL/SQL function: (Difficulty level – Easy)



```
``psql
```

```
CREATE OR REPLACE FUNCTION is_even(num NUMBER)
RETURN BOOLEAN IS
BEGIN
    IF MOD(num, 2) = 0 THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
```

```
---
```

**\*\*What does the PL/SQL function `is\_even` do?\*\***

- A. It checks if the input `num` is an even number and returns `TRUE` if it is, `FALSE` otherwise.
- B. It checks if the input `num` is a positive number and returns `TRUE` if it is, `FALSE` otherwise.
- C. It checks if the input `num` is a prime number and returns `TRUE` if it is, `FALSE` otherwise.
- D. It calculates the factorial of the input `num`.

**\*\*Correct Option:\*\* A**

```
---
```

Ques4 - Consider the following PL/SQL function: (Difficulty level – Easy)

```
``psql
```

```
CREATE OR REPLACE FUNCTION get_employee_salary(emp_id NUMBER)
RETURN NUMBER IS
    salary NUMBER;
BEGIN
    -- Retrieve the salary of the employee with the given emp_id
    SELECT salary INTO salary FROM employees WHERE employee_id = emp_id;
    RETURN salary;
```

```
END;
```

What does the PL/SQL function `get\_employee\_salary` do?

- A. It calculates the average salary of all employees.
- B. It retrieves the salary of the employee with the specified `emp\_id`.
- C. It calculates the total salary of all employees.
- D. It retrieves the highest salary among all employees.

**\*\*Correct Option:\*\* B**

Ques5 - Consider the following PL/SQL function: (Difficulty level – Easy)

```
CREATE OR REPLACE FUNCTION convert_to_uppercase(text VARCHAR2)
RETURN VARCHAR2 IS
    upper_text VARCHAR2(100);
BEGIN
    upper_text := UPPER(text);
    RETURN upper_text;
END;
```

What does the PL/SQL function `convert\_to\_uppercase` do?

- A. It calculates the length of the input `text`.
- B. It calculates the square of a numeric input.
- C. It converts the input `text` to uppercase.

D. It calculates the factorial of a numeric input.

Correct Option: C

**\*\*Ques6 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION calculate_tax(income NUMBER)
RETURN NUMBER IS
    tax NUMBER;
BEGIN
    IF income <= 50000 THEN
        tax := income * 0.1;
    ELSE
        tax := 50000 * 0.1 + (income - 50000) * 0.2;
    END IF;
    RETURN tax;
END;
```

```

**\*\*What does the PL/SQL function `calculate\_tax` do?\*\***

A. It calculates the total income after applying a tax rate.

B. It calculates the square root of the input number `income`.

C. It calculates the factorial of the input number `income`.

D. It calculates the tax amount based on the input income.

**\*\*Correct Option:\*\* D**

---

**\*\*Ques7 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

``psql

```
CREATE OR REPLACE FUNCTION reverse_and_uppercase(input_str VARCHAR2)
RETURN VARCHAR2 IS
    reversed_upper VARCHAR2(255);
BEGIN
    reversed_upper := UPPER(REVERSE(input_str));
    RETURN reversed_upper;
END;
```

``

**\*\*What does the PL/SQL function `reverse\_and\_uppercase` do?\*\***

- A. It calculates the length of the input string `input\_str`.
- B. It calculates the square of the input number `input\_str`.
- C. It reverses the characters in the input string `input\_str` and converts them to uppercase.
- D. It calculates the factorial of the input number `input\_str`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques8 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

``psql

```
CREATE OR REPLACE FUNCTION find_largest
(
    numbers VARCHAR2
)
RETURN NUMBER IS
    largest NUMBER := NULL;
    num_list VARCHAR2(255);
    num_str VARCHAR2(10);
BEGIN
    num_list := TRIM(BOTH ',' FROM numbers);
```

```

LOOP
  EXIT WHEN LENGTH(num_list) = 0;
  num_str := TRIM(SUBSTR(num_list, 1, INSTR(num_list, ',') - 1));
  num_list := SUBSTR(num_list, INSTR(num_list, ',') + 1);
  IF TO_NUMBER(num_str) > largest OR largest IS NULL THEN
    largest := TO_NUMBER(num_str);
  END IF;
END LOOP;
RETURN largest;
END;

```

---

**\*\*What does the PL/SQL function `find\_largest` do?\*\***

- A. It calculates the square root of the input string `numbers`.
- B. It calculates the sum of all numbers in the input string `numbers`.
- C. It retrieves the largest number from a comma-separated list of numbers in the input string `numbers`.
- D. It calculates the factorial of all numbers in the input string `numbers`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques9 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```plsql

```

CREATE OR REPLACE FUNCTION generate_invoice(total_amount NUMBER, customer_id NUMBER)
RETURN VARCHAR2 IS
  invoice_text VARCHAR2(500);
  customer_name VARCHAR2(255);
BEGIN
  -- Retrieve the customer's name based on the customer_id
  SELECT name INTO customer_name FROM customers WHERE customer_id = customer_id;
  invoice_text := 'Invoice for ' || customer_name || ': Total Amount - $' || total_amount;

```

```
    RETURN invoice_text;
END;
```

```

**\*\*What does the PL/SQL function `generate\_invoice` do?\*\***

- A. It generates an invoice text for a customer with the specified `customer\_id` and total amount.
- B. It calculates the average total amount for all customers.
- C. It retrieves the customer's name based on the customer\_id.
- D. It calculates the total amount for a customer with the specified `customer\_id`.

**\*\*Correct Option:\*\* A**

**\*\*Ques10 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE product_recommendations AS
    FUNCTION recommend_products(customer_id NUMBER) RETURN VARCHAR2;
    FUNCTION get_product_rating(product_id NUMBER) RETURN NUMBER;
    FUNCTION get_product_reviews(product_id NUMBER) RETURN NUMBER;
END product_recommendations;
```

/

```

**\*\*What does the PL/SQL package `product\_recommendations` contain?\*\***

- A. It contains three PL/SQL functions, `recommend\_products`, `get\_product\_rating`, and `get\_product\_reviews`, for providing product recommendations and retrieving product ratings and reviews.
- B. It contains two PL/SQL triggers, `recommend\_products`, `get\_product\_rating`, and `get\_product\_reviews`, for providing product recommendations and retrieving product ratings and reviews.
- C. It contains four PL/SQL procedures, `recommend\_products`, `get\_product\_rating`, and `get\_product\_reviews`, for providing product recommendations and retrieving product ratings and reviews.

D. It contains one PL/SQL function, `product\_recommendations`, and one PL/SQL procedure, `product\_recommendations`, for providing product recommendations and retrieving product ratings and reviews.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques11 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE inventory_management AS
    FUNCTION check_stock_availability(product_id NUMBER, warehouse_id NUMBER) RETURN BOOLEAN;
    FUNCTION transfer_product(product_id NUMBER, source_warehouse_id NUMBER, destination_warehouse_id
NUMBER, quantity NUMBER) RETURN NUMBER;
    FUNCTION get_product_location(product_id NUMBER) RETURN VARCHAR2;
END inventory_management;
```

/

**\*\*What does the PL/SQL package `inventory\_management` contain?\*\***

A. It contains four PL/SQL functions, `check\_stock\_availability`, `transfer\_product`, and `get\_product\_location`, for managing inventory and retrieving product locations.

B. It contains three PL/SQL triggers, `check\_stock\_availability`, `transfer\_product`, and `get\_product\_location`, for managing inventory and retrieving product locations.

C. It contains two PL/SQL

procedures, `check\_stock\_availability`, `transfer\_product`, and `get\_product\_location`, for managing inventory and retrieving product locations.

D. It contains one PL/SQL function, `inventory\_management`, and one PL/SQL procedure, `inventory\_management`, for managing inventory and retrieving product locations.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques12 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```
```sql
```

```
CREATE OR REPLACE PACKAGE customer_order_history AS
    FUNCTION get_order_count(customer_id NUMBER) RETURN NUMBER;
    FUNCTION get_average_order_value(customer_id NUMBER) RETURN NUMBER;
    FUNCTION get_last_order_date(customer_id NUMBER) RETURN DATE;
END customer_order_history;
```

```
/
```
```

**\*\*What does the PL/SQL package `customer\_order\_history` contain?\*\***

- A. It contains three PL/SQL functions, `get\_order\_count`, `get\_average\_order\_value`, and `get\_last\_order\_date`, for retrieving customer order history statistics.
- B. It contains three PL/SQL triggers, `get\_order\_count`, `get\_average\_order\_value`, and `get\_last\_order\_date`, for retrieving customer order history statistics.
- C. It contains three PL/SQL procedures, `get\_order\_count`, `get\_average\_order\_value`, and `get\_last\_order\_date`, for retrieving customer order history statistics.
- D. It contains one PL/SQL function, `customer\_order\_history`, and one PL/SQL procedure, `customer\_order\_history`, for retrieving customer order history statistics.

**\*\*Correct Option:\*\* A**

```
---
```

**\*\*Ques13 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```
```sql
```

```
CREATE OR REPLACE PACKAGE employee_performance AS
    FUNCTION calculate_performance_rating(employee_id NUMBER, year NUMBER) RETURN NUMBER;
    FUNCTION get_top_performing_employee(year NUMBER) RETURN VARCHAR2;
END employee_performance;
```

```
/
```
```

**\*\*What does the PL/SQL package `employee\_performance` contain?\*\***



- A. It contains two PL/SQL functions, `calculate\_performance\_rating` and `get\_top\_performing\_employee`, for calculating employee performance ratings and identifying the top-performing employee.
- B. It contains one PL/SQL triggers, `calculate\_performance\_rating` and `get\_top\_performing\_employee`, for calculating employee performance ratings and identifying the top-performing employee.
- C. It contains three PL/SQL procedures, `calculate\_performance\_rating` and `get\_top\_performing\_employee`, for calculating employee performance ratings and identifying the top-performing employee.
- D. It contains four PL/SQL function, `employee\_performance`, and one PL/SQL procedure, `employee\_performance`, for calculating employee performance ratings and identifying the top-performing employee.

**\*\*Correct Option:\*\* A**

**\*\*Ques14 - Consider the following PL/SQL package specification: (Difficulty level – Easy)\*\***

``plsql

```
CREATE OR REPLACE PACKAGE employee_info AS
    FUNCTION get_employee_name(emp_id NUMBER) RETURN VARCHAR2;
    FUNCTION get_employee_salary(emp_id NUMBER) RETURN NUMBER;
END employee_info;
```

/

**\*\*What does the PL/SQL package `employee\_info` contain?\*\***

- A. It contains four PL/SQL functions, `get\_employee\_name` and `get\_employee\_salary`.
- B. It contains PL/SQL triggers, `get\_employee\_name` and `get\_employee\_salary`.
- C. It contains two PL/SQL procedures, `get\_employee\_name` and `get\_employee\_salary`.
- D. It contains one PL/SQL function, `employee\_info`, and one PL/SQL procedure, `employee\_info`.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques15 - Consider the following PL/SQL package specification: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE math_operations AS
    FUNCTION add_numbers(num1 NUMBER, num2 NUMBER) RETURN NUMBER;
    FUNCTION subtract_numbers(num1 NUMBER, num2 NUMBER) RETURN NUMBER;
END math_operations;
/
```

```

**\*\*What does the PL/SQL package `math\_operations` contain?\*\***

- A. It contains two PL/SQL functions, `add\_numbers` and `subtract\_numbers`, for performing mathematical operations.
- B. It contains two PL/SQL triggers, `add\_numbers` and `subtract\_numbers`, for performing mathematical operations.
- C. It contains two PL/SQL procedures, `add\_numbers` and `subtract\_numbers`, for performing mathematical operations.
- D. It contains one PL/SQL function, `math\_operations`, and one PL/SQL procedure, `math\_operations`, for performing mathematical operations.

**\*\*Correct Option:\*\* A**

**\*\*Ques1 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION calculate_area(length NUMBER, width NUMBER)
RETURN NUMBER IS
    area NUMBER;
BEGIN
    area := length * width;
    RETURN area;
END;
```

---

**\*\*What does the PL/SQL function `calculate\_area` do?\*\***

- A. It calculates the perimeter of a rectangle.
- B. It calculates the area of a rectangle.
- C. It calculates the volume of a rectangle.
- D. It calculates the diagonal length of a rectangle.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques7 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION get_grade(score NUMBER)
RETURN VARCHAR2 IS
    grade VARCHAR2(2);
BEGIN
    IF score >= 90 THEN
        grade := 'A';
    ELSIF score >= 80 THEN
        grade := 'B';
    ELSIF score >= 70 THEN
        grade := 'C';
    ELSE
        grade := 'D';
    END IF;
    RETURN grade;
END;
```

```

**\*\*What does the PL/SQL function `get\_grade` do?\*\***

- A. It calculates the square root of the input `score`.
- B. It calculates the average of multiple scores.
- C. It assigns a grade ('A', 'B', 'C', or 'D') based on the input `score`.
- D. It calculates the factorial of the input `score`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques2 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION is_positive(num NUMBER)
RETURN BOOLEAN IS
BEGIN
    IF num > 0 THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
```

```

**\*\*What does the PL/SQL function `is\_positive` do?\*\***

- A. It checks if the input `num` is a positive number and returns `TRUE` if it is, `FALSE` otherwise.
- B. It checks if the input `num` is an even number and returns `TRUE` if it is, `FALSE` otherwise.
- C. It calculates the square of the input `num`.
- D. It calculates the factorial of the input `num`.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques3 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```sql

```
CREATE OR REPLACE FUNCTION reverse_string(input_str VARCHAR2)
RETURN VARCHAR2 IS
    reversed_str VARCHAR2(255);
BEGIN
    SELECT REVERSE(input_str) INTO reversed_str FROM DUAL;
    RETURN reversed_str;
END;
```

```

**\*\*What does the PL/SQL function `reverse\_string` do?\*\***

- A. It calculates the length of the input string `input\_str`.
- B. It calculates the square root of the input number `input\_str`.
- C. It reverses the characters in the input string `input\_str`.
- D. It calculates the factorial of the input number `input\_str`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques4 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```sql

```
CREATE OR REPLACE FUNCTION find_maximum(a NUMBER, b NUMBER)
RETURN NUMBER IS
    max_val NUMBER;
BEGIN
    IF a > b THEN
        max_val := a;
```

```
ELSE
    max_val := b;
END IF;
RETURN max_val;
END;
```

```

**\*\*What does the PL/SQL function `find\_maximum` do?\*\***

- A. It calculates the average of two numbers.
- B. It calculates the sum of two numbers.
- C. It calculates the maximum value between two numbers.
- D. It calculates the factorial of two numbers.

**\*\*Correct Option:\*\* C**

**\*\*Ques5 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION calculate_discount(amount NUMBER)
RETURN NUMBER IS
    discount NUMBER;
BEGIN
    IF amount >= 1000 THEN
        discount := 0.1 * amount;
    ELSE
        discount := 0;
    END IF;
    RETURN discount;
END;
```

```

**\*\*What does the PL/SQL function `calculate\_discount` do?\*\***

- A. It calculates the total cost after applying a discount of 10%.
- B. It calculates the total cost without any discount.
- C. It calculates the total cost after applying a discount of 1%.
- D. It calculates the total cost after applying a discount of 5%.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques6 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION is_vowel(character CHAR)
RETURN BOOLEAN IS
BEGIN
    IF character IN ('A', 'E', 'I', 'O', 'U', 'a', 'e', 'i', 'o', 'u') THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
```

```

**\*\*What does the PL/SQL function `is\_vowel` do?\*\***

- A. It checks if the input character is a consonant and returns `TRUE` if it is, `FALSE` otherwise.
- B. It checks if the input character is a digit and returns `TRUE` if it is, `FALSE` otherwise.
- C. It checks if the input character is a vowel and returns `TRUE` if it is, `FALSE` otherwise.
- D. It calculates the square root of the input character.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques7 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION find_length(input_str VARCHAR2)
RETURN NUMBER IS
    length NUMBER;
BEGIN
    SELECT LENGTH(input_str) INTO length FROM DUAL;
    RETURN length;
END;
```

```

**\*\*What does the PL/SQL function `find\_length` do?\*\***

- A. It calculates the factorial of the length of the input string `input\_str`.
- B. It calculates the square root of the length of the input string `input\_str`.
- C. It retrieves the length of the input string `input\_str`.
- D. It checks if the length of the input string `input\_str` is even and returns `TRUE` if it is, `FALSE` otherwise.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques8 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION calculate_average(num1 NUMBER, num2 NUMBER)
RETURN NUMBER IS
    average NUMBER;
```



```
BEGIN
  average := (num1 + num2) / 2;
  RETURN average;
END;
```

---

**\*\*What does the PL/SQL function `calculate\_average` do?\*\***

- A. It calculates the sum of two numbers.
- B. It calculates the product of two numbers.
- C. It calculates the average of two numbers.
- D. It calculates the square root of two numbers.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques9 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION is_positive_or_zero(num NUMBER)
RETURN BOOLEAN IS
BEGIN
  IF num >= 0 THEN
    RETURN TRUE;
  ELSE
    RETURN FALSE;
  END IF;
END;
```

```

**\*\*What does the PL/SQL function `is\_positive\_or\_zero` do?\*\***

- A. It checks if the input `num` is a positive number and returns `TRUE` if it is, `FALSE` otherwise.

- B. It checks if the input `num` is an even number and returns `TRUE` if it is, `FALSE` otherwise.
- C. It checks if the input `num` is a non-negative number and returns `TRUE` if it is, `FALSE` otherwise.
- D. It calculates the factorial of the input `num`.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques10 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```plsql

```
CREATE OR REPLACE FUNCTION generate_greeting(name VARCHAR2)
RETURN VARCHAR2 IS
    greeting VARCHAR2(100);
BEGIN
    greeting := 'Hi there, ' || name || '!';
    RETURN greeting;
END;
```

```

**\*\*What does the PL/SQL function `generate\_greeting` do?\*\***

- A. It calculates the length of the input string `name`.
- B. It calculates the square of a numeric input.
- C. It generates a friendly greeting message with the input `name`.
- D. It calculates the factorial of a numeric input.

**\*\*Correct Option:\*\* C**

**\*\*Ques11 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```sql

```
CREATE OR REPLACE FUNCTION calculate_factorial(n NUMBER)
RETURN NUMBER IS
    result NUMBER := 1;
BEGIN
    IF n < 0 THEN
        RETURN NULL;
    ELSIF n = 0 THEN
        RETURN 1;
    ELSE
        FOR i IN 1..n LOOP
            result := result * i;
        END LOOP;
    END IF;
    RETURN result;
END;
```

```

**\*\*What does the PL/SQL function `calculate\_factorial` do?\*\***

- A. It calculates the factorial of a non-negative integer `n`.
- B. It calculates the square root of the input number `n`.
- C. It calculates the average of multiple numbers.
- D. It calculates the sum of all integers from 1 to `n`.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques12 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```sql

```

CREATE OR REPLACE FUNCTION calculate_fibonacci(n NUMBER)
RETURN NUMBER IS
  a NUMBER := 0;
  b NUMBER := 1;
  result NUMBER := 0;
BEGIN
  IF n <= 0 THEN
    RETURN 0;
  ELSIF n = 1 THEN
    RETURN 1;
  ELSE
    FOR i IN 2..n LOOP
      result := a + b;
      a := b;
      b := result;
    END LOOP;
  END IF;
  RETURN result;
END;

```

**\*\*What does the PL/SQL function `calculate\_fibonacci` do?\*\***

- A. It calculates the sum of the first `n` Fibonacci numbers.
- B. It calculates the square root of the input number `n`.
- C. It calculates the factorial of the input number `n`.
- D. It calculates the `n`-th Fibonacci number.

**\*\*Correct Option:\*\* D**

---

**\*\*Ques13 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

``plsql

```
CREATE OR REPLACE FUNCTION calculate_power(base NUMBER, exponent NUMBER)
RETURN NUMBER IS
    result NUMBER := 1;
BEGIN
    IF exponent < 0 THEN
        RETURN NULL;
    ELSE
        FOR i IN 1..exponent LOOP
            result := result * base;
        END LOOP;
    END IF;
    RETURN result;
END;
```

---

**\*\*What does the PL/SQL function `calculate\_power` do?\*\***

- A. It calculates the product of `base` and `exponent`.
- B. It calculates the square root of `base` raised to the power of `exponent`.
- C. It calculates the factorial of `exponent`.
- D. It calculates `base` raised to the power of `exponent`.

**\*\*Correct Option:\*\* D**

---

**\*\*Ques14 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

``plsql

```
CREATE OR REPLACE FUNCTION is_palindrome(word VARCHAR2)
RETURN BOOLEAN IS
    reversed_word VARCHAR2(255);
BEGIN
```

```

reversed_word := REVERSE(word);
IF word = reversed_word THEN
    RETURN TRUE;
ELSE
    RETURN FALSE;
END IF;
END;

```

**\*\*What does the PL/SQL function `is\_palindrome` do?\*\***

- A. It checks if the input string `word` is a palindrome (reads the same forwards and backwards) and returns `TRUE` if it is, `FALSE` otherwise.
- B. It calculates the length of the input string `word`.
- C. It calculates the square root of the input number `word`.
- D. It checks if the input string `word` contains any digits and returns `TRUE` if it does, `FALSE` otherwise.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques25 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\***

```sql

```

CREATE OR REPLACE FUNCTION get_employee_salary(emp_id NUMBER)
RETURN NUMBER IS
    salary NUMBER;
BEGIN
    -- Retrieve the salary of the employee with the given emp_id
    SELECT salary INTO salary FROM employees WHERE employee_id = emp_id;
    IF SQL%FOUND THEN
        RETURN salary;
    ELSE
        RETURN NULL;
    END IF;

```

```
END;
```

\*\*\*What does the PL/SQL function `get\_employee\_salary` do?\*\*\*

- A. It calculates the average salary of all employees.
- B. It retrieves the salary of the employee with the specified `emp\_id`.
- C. It calculates the total salary of all employees.
- D. It retrieves the highest salary among all employees.

\*\*\*Correct Option:\*\*\* B

\*\*\*Ques15 - Consider the following PL/SQL function: (Difficulty level – Medium)\*\*\*

```plsql

```
CREATE OR REPLACE FUNCTION count_words(sentence VARCHAR2)
RETURN NUMBER IS
    word_count NUMBER := 0;
BEGIN
    FOR i IN 1..LENGTH(sentence) LOOP
        IF SUBSTR(sentence, i, 1) = ' ' THEN
            word_count := word_count + 1;
        END IF;
    END LOOP;
    -- Add one to count the last word
    word_count := word_count + 1;
    RETURN word_count;
END;
```

\*\*\*What does the PL/SQL function `count\_words` do?\*\*\*

- A. It calculates the number of characters in the input sentence.
- B. It calculates the number of words in the input sentence.
- C. It calculates the number of vowels in the input sentence.
- D. It calculates the number of digits in the input sentence.

**\*\*Correct Option:\*\* B**

---

2 mark questions –

**\*\*Ques1 - Consider the following PL/SQL package specification: (Difficulty level – Medium)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE product_discounts AS
    FUNCTION calculate_discount(product_id NUMBER, quantity NUMBER) RETURN NUMBER;
    FUNCTION apply_discount_to_order(order_id NUMBER) RETURN BOOLEAN;
END product_discounts;
```

/

```

**\*\*What does the PL/SQL package `product\_discounts` contain?\*\***

- A. It contains two PL/SQL functions, `calculate\_discount` and `apply\_discount\_to\_order`, for calculating and applying product discounts.
- B. It contains two PL/SQL triggers, `calculate\_discount` and `apply\_discount\_to\_order`, for calculating and applying product discounts.
- C. It contains two PL/SQL procedures, `calculate\_discount` and `apply\_discount\_to\_order`, for calculating and applying product discounts.
- D. It contains one PL/SQL function, `product\_discounts`, and one PL/SQL procedure, `product\_discounts`, for calculating and applying product discounts.



**\*\*Correct Option:\*\* A**

---

**\*\*Ques2 - Consider the following PL/SQL package specification: (Difficulty level – Medium)\*\***

```sql

```
CREATE OR REPLACE PACKAGE order_processing AS
    FUNCTION process_order(order_id NUMBER) RETURN BOOLEAN;
    FUNCTION validate_payment(order_id NUMBER) RETURN BOOLEAN;
END order_processing;
```

/

```

**\*\*What does the PL/SQL package `order\_processing` contain?\*\***

- A. It contains two PL/SQL functions, `process\_order` and `validate\_payment`, for processing orders and validating payments.
- B. It contains two PL/SQL triggers, `process\_order` and `validate\_payment`, for processing orders and validating payments.
- C. It contains two PL/SQL procedures, `process\_order` and `validate\_payment`, for processing orders and validating payments.
- D. It contains one PL/SQL function, `order\_processing`, and one PL/SQL procedure, `order\_processing`, for processing orders and validating payments.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques3 - Consider the following PL/SQL package specification: (Difficulty level – Medium)\*\***

```sql

```
CREATE OR REPLACE PACKAGE employee_management AS
    FUNCTION hire_employee(name VARCHAR2, salary NUMBER) RETURN NUMBER;
    FUNCTION terminate_employee(employee_id NUMBER) RETURN BOOLEAN;
END employee_management;
```

/

```

**\*\*What does the PL/SQL package `employee\_management` contain?\*\***

- A. It contains two PL/SQL functions, `hire\_employee` and `terminate\_employee`, for hiring and terminating employees.
- B. It contains two PL/SQL triggers, `hire\_employee` and `terminate\_employee`, for hiring and terminating employees.
- C. It contains two PL/SQL procedures, `hire\_employee` and `terminate\_employee`, for hiring and terminating employees.
- D. It contains one PL/SQL function, `employee\_management`, and one PL/SQL procedure, `employee\_management`, for hiring and terminating employees.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques4 - Consider the following PL/SQL package specification: (Difficulty level – Medium)\*\***

```pls sql

```
CREATE OR REPLACE PACKAGE order_management AS
    FUNCTION create_order(customer_id NUMBER, total_amount NUMBER) RETURN NUMBER;
    FUNCTION cancel_order(order_id NUMBER) RETURN BOOLEAN;
END order_management;
```

/

**\*\*What does the PL/SQL package `order\_management` contain?\*\***

- A. It contains two PL/SQL functions, `create\_order` and `cancel\_order`, for creating and canceling orders.
- B. It contains two PL/SQL triggers, `create\_order` and `cancel\_order`, for creating and canceling orders.
- C. It contains two PL/SQL procedures, `create\_order` and `cancel\_order`, for creating and canceling orders.
- D. It contains one PL/SQL function, `order\_management`, and one PL/SQL procedure, `order\_management`, for creating and canceling orders.

**\*\*Correct Option:\*\* A**

**\*\*Ques5 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

``psql

```
CREATE OR REPLACE PACKAGE order_tracking AS
    FUNCTION track_order(order_id NUMBER) RETURN VARCHAR2;
    FUNCTION estimate_delivery_time(order_id NUMBER) RETURN NUMBER;
    FUNCTION get_order_status(order_id NUMBER) RETURN VARCHAR2;
END order_tracking;
```

/

**\*\*What does the PL/SQL package `order\_tracking` contain?\*\***

- A. It contains three PL/SQL functions, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.
- B. It contains three PL/SQL triggers, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.
- C. It contains three PL/SQL procedures, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.
- D. It contains one PL/SQL function, `order\_tracking`, and one PL/SQL procedure, `order\_tracking`, for tracking orders and estimating delivery times.

**\*\*Correct Option:\*\* A**

---

---

**\*\*Ques6 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

``psql

```
CREATE OR REPLACE PACKAGE project_management AS
    FUNCTION allocate_resources(project_id NUMBER, resource_id NUMBER, hours NUMBER) RETURN BOOLEAN;
    FUNCTION get_project_status(project_id NUMBER) RETURN VARCHAR2;
END project_management;
```

/

...

**\*\*What does the PL/SQL package `project\_management` contain?\*\***

- A. It contains two PL/SQL functions, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- B. It contains two PL/SQL triggers, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- C. It contains two PL/SQL procedures, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- D. It contains one PL/SQL function, `project\_management`, and one PL/SQL procedure, `project\_management`, for resource allocation and project status retrieval.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques7 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE student_grading AS
    FUNCTION calculate_final_grade(student_id NUMBER, course_id NUMBER) RETURN CHAR;
    FUNCTION get_student_ranking(course_id NUMBER) RETURN NUMBER;
END student_grading;
```

/

...

**\*\*What does the PL/SQL package `student\_grading` contain?\*\***

- A. It contains two PL/SQL functions, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- B. It contains two PL/SQL triggers, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- C. It contains two PL/SQL procedures, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- D. It contains one PL/SQL function, `student\_grading`, and one PL/SQL procedure, `student\_grading`, for calculating student grades and retrieving student rankings in a course.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques8 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE medical_records AS
    FUNCTION get_patient_history(patient_id NUMBER) RETURN CLOB;
    FUNCTION analyze_patient_data(patient_id NUMBER) RETURN CLOB;
END medical_records;
/
```

```

**\*\*What does the PL/SQL package `medical\_records` contain?\*\***

- A. It contains two PL/SQL functions, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- B. It contains two PL/SQL triggers, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- C. It contains two PL/SQL procedures, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- D. It contains one PL/SQL function, `medical\_records`, and one PL/SQL procedure, `medical\_records`, for retrieving patient medical history and analyzing patient data.

**\*\*Correct Option:\*\* A**

**\*\*Ques9 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE order_tracking AS
    FUNCTION track_order(order_id NUMBER) RETURN VARCHAR2;
    FUNCTION estimate_delivery_time(order_id NUMBER) RETURN NUMBER;
    FUNCTION get_order_status(order_id NUMBER) RETURN VARCHAR2;
END order_tracking;
```

/

---

**\*\*What does the PL/SQL package `order\_tracking` contain?\*\***

- A. It contains three PL/SQL functions, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.
- B. It contains three PL/SQL triggers, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.
- C. It contains three PL/SQL procedures, `track\_order`, `estimate\_delivery\_time`, and `get\_order\_status`, for tracking orders and estimating delivery times.
- D. It contains one PL/SQL function, `order\_tracking`, and one PL/SQL procedure, `order\_tracking`, for tracking orders and estimating delivery times.

**\*\*Correct Option:\*\* A**

---

---

**\*\*Ques10 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

```plsql

```
CREATE OR REPLACE PACKAGE project_management AS
    FUNCTION allocate_resources(project_id NUMBER, resource_id NUMBER, hours NUMBER) RETURN BOOLEAN;
    FUNCTION get_project_status(project_id NUMBER) RETURN VARCHAR2;
END project_management;
```

/

---

**\*\*What does the PL/SQL package `project\_management` contain?\*\***

- A. It contains two PL/SQL functions, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- B. It contains two PL/SQL triggers, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- C. It contains two PL/SQL procedures, `allocate\_resources` and `get\_project\_status`, for resource allocation and project status retrieval.
- D. It contains one PL/SQL function, `project\_management`, and one PL/SQL procedure, `project\_management`, for resource allocation and project status retrieval.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques11 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

``psql

```
CREATE OR REPLACE PACKAGE student_grading AS
    FUNCTION calculate_final_grade(student_id NUMBER, course_id NUMBER) RETURN CHAR;
    FUNCTION get_student_ranking(course_id NUMBER) RETURN NUMBER;
END student_grading;
```

/

``

**\*\*What does the PL/SQL package `student\_grading` contain?\*\***

- A. It contains two PL/SQL functions, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- B. It contains two PL/SQL triggers, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- C. It contains two PL/SQL procedures, `calculate\_final\_grade` and `get\_student\_ranking`, for calculating student grades and retrieving student rankings in a course.
- D. It contains one PL/SQL function, `student\_grading`, and one PL/SQL procedure, `student\_grading`, for calculating student grades and retrieving student rankings in a course.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques12 - Consider the following PL/SQL package specification: (Difficulty level – Hard)\*\***

``psql

```
CREATE OR REPLACE PACKAGE medical_records AS
    FUNCTION get_patient_history(patient_id NUMBER) RETURN CLOB;
    FUNCTION analyze_patient_data(patient_id NUMBER) RETURN CLOB;
END medical_records;
```

/

\*\*\*What does the PL/SQL package `medical\_records` contain?\*\*\*

- A. It contains two PL/SQL functions, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- B. It contains two PL/SQL triggers, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- C. It contains two PL/SQL procedures, `get\_patient\_history` and `analyze\_patient\_data`, for retrieving patient medical history and analyzing patient data.
- D. It contains one PL/SQL function, `medical\_records`, and one PL/SQL procedure, `medical\_records`, for retrieving patient medical history and analyzing patient data.

\*\*\*Correct Option:\*\*\* A

\*\*\*Ques13 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\*\*

```sql

```
DECLARE
    emp_cursor CURSOR FOR
        SELECT employee_name FROM employees;
```

\*\*\*What does the SQL cursor `emp\_cursor` do?\*\*\*

- A. It retrieves all columns from the `employees` table.
- B. It retrieves the `employee\_name` column from the `employees` table.
- C. It updates the `employee\_name` column in the `employees` table.
- D. It deletes records from the `employees` table.

\*\*\*Correct Option:\*\*\* B



---

**\*\*Ques14 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

``sql

```
DECLARE
    product_cursor CURSOR FOR
        SELECT product_name, product_price FROM products;
```

``

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_name` and `product\_price` columns from the `products` table.
- C. It updates the `product\_name` and `product\_price` columns in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques15 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

``sql

```
DECLARE
    order_cursor CURSOR FOR
        SELECT order_id, order_date FROM orders;
```

``

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_id` and `order\_date` columns from the `orders` table.
- C. It updates the `order\_id` and `order\_date` columns in the `orders` table.

D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques1 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
    customer_cursor CURSOR FOR
        SELECT customer_name FROM customers;
```

```

**\*\*What does the SQL cursor `customer\_cursor` do?\*\***

A. It retrieves all columns from the `customers` table.

B. It retrieves the `customer\_name` column from the `customers` table.

C. It updates the `customer\_name` column in the `customers` table.

D. It deletes records from the `customers` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques2 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
    employee_cursor CURSOR FOR
        SELECT employee_id, employee_name FROM employees;
```

```

**\*\*What does the SQL cursor `employee\_cursor` do?\*\***

- A. It retrieves all columns from the `employees` table.
- B. It retrieves the `employee\_id` and `employee\_name` columns from the `employees` table.
- C. It updates the `employee\_id` and `employee\_name` columns in the `employees` table.
- D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques3 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

``sql

```
DECLARE
  product_cursor CURSOR FOR
    SELECT product_id FROM products WHERE product_price > 100;
```

``

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_id` column from the `products` table for products with a price greater than 100.
- C. It updates the `product\_id` column in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques4 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

``sql

```
DECLARE
order_cursor CURSOR FOR
SELECT order_date FROM orders WHERE order_status = 'Shipped';
```

---

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_date` column from the `orders` table for orders with a status of 'Shipped'.
- C. It updates the `order\_date` column in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques5 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
customer_cursor CURSOR FOR
SELECT customer_id FROM customers WHERE registration_date >= '2023-01-01';
```

---

**\*\*What does the SQL cursor `customer\_cursor` do?\*\***

- A. It retrieves all columns from the `customers` table.
- B. It retrieves the `customer\_id` column from the `customers` table for customers registered on or after January 1, 2023.
- C. It updates the `customer\_id` column in the `customers` table.
- D. It deletes records from the `customers` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques6 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
  employee_cursor CURSOR FOR
    SELECT department_id, COUNT(*) FROM employees GROUP BY department_id;
```

```

**\*\*What does the SQL cursor `employee\_cursor` do?\*\***

- A. It retrieves all columns from the `employees` table.
- B. It retrieves the `department\_id` and the count of employees in each department from the `employees` table.
- C. It updates the `department\_id` and employee counts in the `employees` table.
- D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques7 - Consider the following SQL cursor declaration: (Difficulty level – Easy)\*\***

```sql

```
DECLARE
  product_cursor CURSOR FOR
    SELECT product_name, product_category FROM products WHERE product_category = 'Electronics';
```

```

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.

B. It retrieves the `product\_name` and `product\_category` columns from the `products` table for products in the 'Electronics' category.

C. It updates the `product\_name` and `product\_category` columns in the `products` table.

D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques8 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

``sql

```
DECLARE
  employee_cursor CURSOR FOR
    SELECT employee_id, employee_name, department_id
    FROM employees
    WHERE salary > (SELECT AVG(salary) FROM employees);
```

``

**\*\*What does the SQL cursor `employee\_cursor` do?\*\***

A. It retrieves all columns from the `employees` table.

B. It retrieves the `employee\_id`, `employee\_name`, and `department\_id` columns from the `employees` table for employees with salaries above the average salary in the company.

C. It updates the `employee\_id`, `employee\_name`, and `department\_id` columns in the `employees` table.

D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques9 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

``sql

```
DECLARE
  order_cursor CURSOR FOR
    SELECT order_id, customer_id, order_date
    FROM orders
    WHERE EXISTS (SELECT 1 FROM order_items WHERE order_items.order_id = orders.order_id);
```

``

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_id`, `customer\_id`, and `order\_date` columns from the `orders` table for orders that have associated order items.
- C. It updates the `order\_id`, `customer\_id`, and `order\_date` columns in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques10 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

``sql

```
DECLARE
  customer_cursor CURSOR FOR
    SELECT customer_id, COUNT(*) AS order_count
    FROM orders
    GROUP BY customer_id
    HAVING COUNT(*) > 5;
```

``

**\*\*What does the SQL cursor `customer\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.

B. It retrieves the `customer\_id` and the count of orders placed by each customer from the `orders` table for customers who have placed more than 5 orders.

C. It updates the `customer\_id` and order counts in the `orders` table.

D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques11 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
    product_cursor CURSOR FOR
        SELECT product_id, product_name, product_price
        FROM products
        WHERE product_id IN (SELECT product_id FROM order_items GROUP BY product_id HAVING COUNT(*) >= 10);
```

```

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

A. It retrieves all columns from the `products` table.

B. It retrieves the `product\_id`, `product\_name`, and `product\_price` columns from the `products` table for products that have been ordered at least 10 times.

C. It updates the `product\_id`, `product\_name`, and `product\_price` columns in the `products` table.

D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques12 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql



```

DECLARE
order_cursor CURSOR FOR
    SELECT order_id, order_date, SUM(order_total) AS total_amount
    FROM orders
    WHERE order_status = 'Shipped'
    GROUP BY order_id, order_date
    HAVING SUM(order_total) > 1000;

```

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_id`, `order\_date`, and total order amount columns from the `orders` table for shipped orders with a total amount greater than 1000.
- C. It updates the `order\_id`, `order\_date`, and total order amount columns in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques13 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```

```sql
CREATE OR REPLACE TRIGGER update_salary_trigger
BEFORE UPDATE ON employees
FOR EACH ROW
BEGIN
    IF :NEW.salary > :OLD.salary THEN
        INSERT INTO salary_history (employee_id, old_salary, new_salary, change_date)
        VALUES (:OLD.employee_id, :OLD.salary, :NEW.salary, SYSDATE);
    END IF;
END;

```

/

---

**\*\*What does the SQL trigger `update\_salary\_trigger` do?\*\***

- A. It updates the salary of all employees in the `employees` table.
- B. It inserts a record into the `salary\_history` table whenever an employee's salary is increased.
- C. It deletes records from the `employees` table whenever an employee's salary is updated.
- D. It calculates the average salary of all employees.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques14 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE TRIGGER audit_employee_delete
AFTER DELETE ON employees
FOR EACH ROW
BEGIN
    INSERT INTO audit_log (event_type, event_date, username, details)
    VALUES ('Employee Deletion', SYSDATE, USER, 'Employee ID: ' || :OLD.employee_id);
END;
```

/

---

**\*\*What does the SQL trigger `audit\_employee\_delete` do?\*\***

- A. It updates employee records in the `employees` table.
- B. It inserts a record into the `audit\_log` table whenever an employee is deleted.
- C. It inserts a record into the `employees` table whenever an employee is deleted.
- D. It calculates the total number of employees in the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques15 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

``sql

```
CREATE OR REPLACE TRIGGER calculate_avg_salary
AFTER INSERT OR DELETE ON employees
FOR EACH ROW
BEGIN
    DECLARE
        total_salary NUMBER;
        num_employees NUMBER;
    BEGIN
        SELECT SUM(salary), COUNT(*) INTO total_salary, num_employees FROM employees;
        IF num_employees > 0 THEN
            INSERT INTO salary_stats (average_salary, total_employees, calculation_date)
            VALUES (total_salary / num_employees, num_employees, SYSDATE);
        END IF;
    END;
END;
```

/

**\*\*What does the SQL trigger `calculate\_avg\_salary` do?\*\***

- A. It updates the salary of all employees in the `employees` table.
- B. It calculates the average salary and total number of employees whenever a new employee is inserted or an employee is deleted.
- C. It inserts a record into the `salary\_stats` table whenever an employee is deleted.
- D. It calculates the total number of employees in the `employees` table.

**\*\*Correct Option:\*\* B**

**\*\*Ques1 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```sql

```
CREATE OR REPLACE TRIGGER prevent_salary_reduction
BEFORE UPDATE ON employees
FOR EACH ROW
BEGIN
    IF :NEW.salary < :OLD.salary THEN
        RAISE_APPLICATION_ERROR (-20001, 'Salary reduction is not allowed.');
```

```
    END IF;
```

```
END;
```

/

```

**\*\*What does the SQL trigger `prevent\_salary\_reduction` do?\*\***

- A. It updates the salary of all employees in the `employees` table.
- B. It prevents any attempt to reduce an employee's salary and raises a custom application error if such an update is detected.
- C. It inserts a record into the `salary\_history` table whenever an employee's salary is increased.
- D. It calculates the average salary of all employees.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques2 - Consider the following SQL trigger: (Difficulty level – Medium)\*\***

```sql

```
CREATE OR REPLACE TRIGGER audit_table_changes
AFTER INSERT OR UPDATE OR DELETE ON employees
DECLARE
    change_description VARCHAR2(500);
BEGIN
    change_description := 'Table "employees" was ';
```

```

IF INSERTING THEN
    change_description := change_description || 'inserted into.';
ELSIF UPDATING THEN
    change_description := change_description || 'updated.';
ELSIF DELETING THEN
    change_description := change_description || 'deleted from.';
END IF;
INSERT INTO audit_log (event_type, event_date, details)
VALUES ('Table Change', SYSDATE, change_description);
END;
/

```

**\*\*What does the SQL trigger `audit\_table\_changes` do?\*\***

- A. It updates the `employees` table whenever a change is made to it.
- B. It inserts a record into the `audit\_log` table whenever a change (insert, update, or delete) is made to the `employees` table, including a description of the change.
- C. It calculates the total number of employees in the `employees` table.
- D. It deletes records from the `employees` table whenever a change is made to it.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques3 - Consider the following SQL trigger: (Difficulty level – Easy)\*\***

``sql

```

CREATE OR REPLACE TRIGGER enforce_manager_approval
BEFORE INSERT ON purchase_orders
FOR EACH ROW
BEGIN
    IF :NEW.total_amount > 1000 AND :NEW.manager_approval IS NULL THEN

```

```

        RAISE_APPLICATION_ERROR (-20002, 'Manager approval is required for purchase orders over $1000.');
```

```

    END IF;
END;
/

```

**\*\*What does the SQL trigger `enforce\_manager\_approval` do?\*\***

A. It inserts records into the `purchase\_orders` table.

B. It updates records in the `purchase\_orders` table.

C. It prevents the insertion of purchase orders with a total amount over \$1000 if they don't have manager approval, raising a custom application error if such an insert is attempted.

D. It calculates the total amount of all purchase orders.

**\*\*Correct Option:\*\* C**

```

---

```

**\*\*Ques4 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

```

```sql

```

```

CREATE OR REPLACE TRIGGER calculate_total_order_amount
AFTER INSERT OR UPDATE ON order_items
FOR EACH ROW
DECLARE
    total_amount NUMBER;
BEGIN
    total_amount := 0;
    SELECT SUM(quantity * unit_price) INTO total_amount FROM order_items WHERE order_id = :NEW.order_id;
    UPDATE orders SET total_amount = total_amount WHERE order_id = :NEW.order_id;
END;
/

```

```

```

```

**\*\*What does the SQL trigger `calculate\_total\_order\_amount` do?\*\***

- A. It inserts records into the `order\_items` table.
- B. It updates records in the `order\_items` table.
- C. It calculates the total order amount for an order whenever a new order item is inserted or an existing order item is updated, and updates the `total\_amount` in the `orders` table.
- D. It calculates the average order amount.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques5 - Consider the following SQL trigger: (Difficulty level –Easy)\*\***

```sql

```
CREATE OR REPLACE TRIGGER prevent_duplicate_records
BEFORE INSERT ON employees
FOR EACH ROW
BEGIN
    IF EXISTS (SELECT 1 FROM employees WHERE employee_id = :NEW.employee_id) THEN
        RAISE_APPLICATION_ERROR (-20003, 'Employee ID must be unique.');
```

```

**\*\*What does the SQL trigger `prevent\_duplicate\_records` do?\*\***

- A. It inserts records into the `employees` table.
- B. It updates records in the `employees` table.
- C. It prevents the insertion of duplicate employee records with the same `employee\_id`, raising a custom application error if such an insert is attempted.
- D. It calculates the total number of employees in the `employees` table.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques6 - Consider the following SQL trigger: (Difficulty level – Hard)\*\***

``sql

```
CREATE OR REPLACE TRIGGER calculate_sales_bonus
AFTER INSERT OR UPDATE ON sales
FOR EACH ROW
BEGIN
    DECLARE
        bonus_amount NUMBER;
    BEGIN
        IF :NEW.sale_amount > 10000 THEN
            bonus_amount := :NEW.sale_amount * 0.05;
            UPDATE sales SET bonus = bonus_amount WHERE sale_id = :NEW.sale_id;
        END IF;
    END;
END;
/
```

```

**\*\*What does the SQL trigger `calculate\_sales\_bonus` do?\*\***

- A. It inserts records into the `sales` table.
- B. It updates records in the `sales` table.
- C. It calculates a sales bonus for sales with an amount over \$10,000 and updates the `bonus` field in the `sales` table whenever a new sale is inserted or an existing sale is updated.
- D. It calculates the average sale amount.

**\*\*Correct Option:\*\* C**



**\*\*Ques7 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

``psql

```
CREATE OR REPLACE FUNCTION get_last_name(full_name VARCHAR2)
RETURN VARCHAR2 IS
    last_name VARCHAR2(50);
BEGIN
    last_name := SUBSTR(full_name, INSTR(full_name, ' ')+1);
    RETURN last_name;
END;
```

``

**\*\*What does the PL/SQL function `get\_last\_name` do?\*\***

- A. It calculates the average length of all words in the input `full\_name`.
- B. It calculates the length of the last word in the input `full\_name`.
- C. It retrieves the last name from the input `full\_name`.
- D. It checks if the input `full\_name` contains any digits and returns `TRUE` if it does, `FALSE` otherwise.

**\*\*Correct Option:\*\* C**

---

**\*\*Ques8 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

``psql

```
CREATE OR REPLACE FUNCTION square_number(num NUMBER)
RETURN NUMBER IS
    square NUMBER;
BEGIN
    square := num * num;
    RETURN square;
```

```
END;
```

```
```
```

**\*\*What does the PL/SQL function `square\_number` do?\*\***

- A. It calculates the square root of the input number `num`.
- B. It calculates the sum of two numbers.
- C. It calculates the square of the input number `num`.
- D. It calculates the factorial of the input number `num`.

**\*\*Correct Option:\*\* C**

```
---
```

**\*\*Ques9 - Consider the following PL/SQL function: (Difficulty level – Easy)\*\***

```
```plsql
```

```
CREATE OR REPLACE FUNCTION is_prime(number NUMBER)
RETURN BOOLEAN IS
BEGIN
    IF number <= 1 THEN
        RETURN FALSE;
    END IF;
    FOR i IN 2..number-1 LOOP
        IF MOD(number, i) = 0 THEN
            RETURN FALSE;
        END IF;
    END LOOP;
    RETURN TRUE;
END;
```

```
```
```

**\*\*What does the PL/SQL function `is\_prime` do?\*\***

- A. It checks if the input `number` is a prime number and returns `TRUE` if it is, `FALSE` otherwise.
- B. It calculates the square root of the input `number`.
- C. It calculates the factorial of the input `number`.
- D. It checks if the input `number` is even and returns `TRUE` if it is, `FALSE` otherwise.

**\*\*Correct Option:\*\* A**

---

**\*\*Ques10 - Consider the following PL/**

**SQL function: (Difficulty level – Easy)\*\***

**``psql**

```
CREATE OR REPLACE FUNCTION get_day_of_week(date_value DATE)
RETURN VARCHAR2 IS
    day_of_week VARCHAR2(15);
BEGIN
    SELECT TO_CHAR(date_value, 'Day') INTO day_of_week FROM DUAL;
    RETURN day_of_week;
END;
```

**``**

**\*\*What does the PL/SQL function `get\_day\_of\_week` do?\*\***

- A. It calculates the day of the week for the input `date\_value` and returns it as a string.
- B. It calculates the square root of the input `date\_value`.
- C. It calculates the average of multiple dates.
- D. It retrieves the month of the input `date\_value`.

**\*\*Correct Option:\*\* A**

**\*\*Ques11 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

``sql

```
DECLARE
  product_cursor CURSOR FOR
    SELECT product_id, product_name
    FROM products
    WHERE product_id NOT IN (SELECT DISTINCT product_id FROM order_items);
```

``

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_id` and `product\_name` columns from the `products` table for products that have not been ordered.
- C. It updates the `product\_id` and `product\_name` columns in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques12 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

``sql

```
DECLARE
  customer_cursor CURSOR FOR
    SELECT customer_id, MAX(order_date) AS last_order_date
    FROM orders
    GROUP BY customer_id
    HAVING MAX(order_date) < TO_DATE('2023-01-01', 'YYYY-MM-DD');
```

``

**\*\*What does the SQL cursor `customer\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `customer\_id` and the last order date columns from the `orders` table for customers whose last order date is before January 1, 2023.
- C. It updates the `customer\_id` and last order date columns in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques13 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
    employee_cursor CURSOR FOR
        SELECT employee_id, employee_name, department_id
        FROM employees
        WHERE department_id = (SELECT department_id FROM departments WHERE department_name =
'Engineering');
```

```

**\*\*What does the SQL cursor `employee\_cursor` do?\*\***

- A. It retrieves all columns from the `employees` table.
- B. It retrieves the `employee\_id`, `employee\_name`, and `department\_id` columns from the `employees` table for employees in the 'Engineering' department.
- C. It updates the `employee\_id`, `employee\_name`, and `department\_id` columns in the `employees` table.
- D. It deletes records from the `employees` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques14 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
  product_cursor CURSOR FOR
    SELECT product_id, product_name
    FROM products
    WHERE product_price = (SELECT MAX(product_price) FROM products);
```

```

**\*\*What does the SQL cursor `product\_cursor` do?\*\***

- A. It retrieves all columns from the `products` table.
- B. It retrieves the `product\_id` and `product\_name` columns from the `products` table for products with the highest product price.
- C. It updates the `product\_id` and `product\_name` columns in the `products` table.
- D. It deletes records from the `products` table.

**\*\*Correct Option:\*\* B**

---

**\*\*Ques15 - Consider the following SQL cursor declaration: (Difficulty level – Hard)\*\***

```sql

```
DECLARE
  order_cursor CURSOR FOR
    SELECT order_id, order_date
    FROM orders
    WHERE order_id = (SELECT MAX(order_id) FROM orders);
```

```

**\*\*What does the SQL cursor `order\_cursor` do?\*\***

- A. It retrieves all columns from the `orders` table.
- B. It retrieves the `order\_id` and `order\_date` columns from the `orders` table for the order with the highest order ID.
- C. It updates the `order\_id` and `order\_date` columns in the `orders` table.
- D. It deletes records from the `orders` table.

**\*\*Correct Option:\*\* B**

1. The recovery scheme must also provide

- a) High availability
- b) Low availability
- c) High reliability
- d) High durability

Answer: a

2. Which one of the following is a failure to a system

- a) Boot crash
- b) Read failure
- c) Transaction failure
- d) All of the mentioned

Answer: c

3. Which of the following belongs to transaction failure

- a) Read error
- b) Boot error
- c) Logical error
- d) All of the mentioned

Answer: c

4. The system has entered an undesirable state (for example, deadlock), as a result of which a transaction cannot continue with its normal execution. This is

- a) Read error
- b) Boot error
- c) Logical error
- d) System error

Answer: c.

5. The transaction can no longer continue with its normal execution because of some internal condition, such as bad input, data not found, overflow, or resource limit exceeded. This is

- a) Read error
- b) Boot error
- c) Logical error



d) System error

Answer: c

6. The assumption that hardware errors and bugs in the software bring the system to a halt, but do not corrupt the nonvolatile storage contents, is known as the

- a) Stop assumption
- b) Fail assumption
- c) Halt assumption
- d) Fail-stop assumption

Answer: d

7. Which kind of failure loses its data in head crash or failure during a transfer operation.

- a) Transaction failure
- b) System crash
- c) Disk failure
- d) All of the mentioned

Answer: c

8. The failure occurred sufficiently early during the transfer that the destination block remains intact.

- a) Partial Failure
- b) Total failure
- c) Successful completion
- d) Data transfer failure

Answer: a

9. The database is partitioned into fixed-length storage units called

- a) Parts
- b) Blocks
- c) Reads
- d) Build

Answer: b

10. Which of the following causes system to crash

- a) Bug in software
- b) Loss of volatile data
- c) Hardware malfunction
- d) All of the mentioned

Answer: d

1. The log is a sequence of \_\_\_\_\_ recording all the update activities in the database.

- a) Log records
- b) Records
- c) Entries
- d) Redo

Answer: a

2. In the \_\_\_\_\_ scheme, a transaction that wants to update the database first creates a complete copy of the database.

- a) Shadow copy
- b) Shadow Paging
- c) Update log records
- d) All of the mentioned

Answer: a

3. The \_\_\_\_\_ scheme uses a page table containing pointers to all pages; the page table itself and all updated pages are copied to a new location.

- a) Shadow copy
- b) Shadow Paging
- c) Update log records
- d) All of the mentioned

Answer: b

4. The current copy of the database is identified by a pointer, called \_\_\_\_\_ which is stored on disk.

- a) Db-pointer
- b) Update log
- c) Update log records

d) All of the mentioned

Answer: a

5. If a transaction does not modify the database until it has committed, it is said to use the \_\_\_\_\_ technique.

- a) Deferred-modification
- b) Late-modification
- c) Immediate-modification
- d) Undo

Answer: a

6. If database modifications occur while the transaction is still active, the transaction is said to use the \_\_\_\_\_ technique.

- a) Deferred-modification
- b) Late-modification
- c) Immediate-modification
- d) Undo

Answer: c

7. \_\_\_\_\_ using a log record sets the data item specified in the log record to the old value.

- a) Deferred-modification
- b) Late-modification
- c) Immediate-modification
- d) Undo

Answer: d

8. In the \_\_\_\_\_ phase, the system replays updates of all transactions by scanning the log forward from the last checkpoint.

- a) Repeating
- b) Redo
- c) Replay
- d) Undo

Answer: b

9. The actions which are played in the order while recording it is called \_\_\_\_\_ history.

- a) Repeating
- b) Redo
- c) Replay
- d) Undo

Answer: a

10. A special redo-only log record  $\langle T_i, X_j, V_1 \rangle$  is written to the log, where  $V_1$  is the value being restored to data item  $X_j$  during the rollback. These log records are sometimes called

- a) Log records
- b) Records
- c) Compensation log records
- d) Compensation redo records

Answer: c

1. In order to reduce the overhead in retrieving the records from the storage space we use

- a) Logs
- b) Log buffer
- c) Medieval space
- d) Lower records

Answer: b

2. The order of log records in the stable storage \_\_\_\_\_ as the order in which they were written to the log buffer.

- a) Must be exactly the same
- b) Can be different
- c) Is opposite
- d) Can be partially same

Answer: a

3. Before a block of data in main memory can be output to the database, all log records pertaining to data in that block must have been output to stable storage. This is

- a) Read-write logging
- b) Read-ahead logging

- c) Write-ahead logging
- d) None of the mentioned

Answer: c

4. Writing the buffered log to \_\_\_\_\_ is sometimes referred to as a log force.

- a) Memory
- b) Backup
- c) Redo memory
- d) Disk

Answer: d

5. The \_\_\_\_\_ policy, allows a transaction to commit even if it has modified some blocks that have not yet been written back to disk.

- a) Force
- b) No-force
- c) Steal
- d) No-steal

Answer: b

6. \_\_\_\_\_ policy allows multiple updates to accumulate on a block before it is output to stable storage, which can reduce the number of output operations greatly for frequently updated blocks.

- a) Force
- b) No-force
- c) Steal
- d) No-steal

Answer: b

7. The \_\_\_\_\_ policy, allows the system to write modified blocks to disk even if the transactions that made those modifications have not all committed.

- a) Force
- b) No-force
- c) Steal

d) No-steal

Answer: c

8. Locks on buffer blocks are unrelated to locks used for concurrency-control of transactions, and releasing them in a non-two-phase manner does not have any implications on transaction serializability. This is

- a) Latches
- b) Swap Space
- c) Dirty Block
- d) None of the mentioned

Answer: a

9. The \_\_\_\_\_ contains a list of blocks that have been updated in the database buffer.

- a) Latches
- b) Swap Space
- c) Dirty Block
- d) None of the mentioned

Answer: c

10. The operating system reserves space on disk for storing virtual-memory pages that are not currently in main memory; this space is called

- a) Latches
- b) Swap Space
- c) Dirty Block
- d) None of the mentioned

Answer: b

1. The silicon chips used for data processing are called

- a) RAM chips
- b) ROM chips
- c) Micro processors
- d) PROM chips

Answer: d

2. Which of the following is used for manufacturing chips?

- a) Control bus
- b) Control unit
- c) Parity unit
- d) Semiconductor

Answer: d

3. What was the name of the first commercially available microprocessor chip?

- a) Intel 308
- b) Intel 33
- c) Intel 4004
- d) Motorola 639

Answer: c

4. The magnetic storage chip used to provide non-volatile direct access storage of data and that have no moving parts are known as

- a) Magnetic core memory
- b) Magnetic tape memory
- c) Magnetic disk memory
- d) Magnetic bubble memory

Answer: d

5. The ALU of a computer normally contains a number of high speed storage element called

- a) Semiconductor memory
- b) Registers
- c) Hard disks
- d) Magnetic disk

Answer: b

6. Which of the following is used only for data entry and storage, and never for processing?

- a) Mouse
- b) Dumb terminal
- c) Micro computer
- d) Dedicated data entry system

Answer: b

7. Non-volatile storage needs to have a \_\_\_\_\_ where the losses in future can be recovered.

- a) Dump
- b) Recover place
- c) Disk
- d) Redo plan

Answer: a

8. A dump of the database contents is also referred to as an \_\_\_\_\_ dump.

- a) Archival
- b) Fuzzy
- c) SQL
- d) All of the mentioned

Answer: a

9. \_\_\_\_\_ dump, writes out SQL DDL statements and SQL insert statements to a file, which can then be reexecuted to re-create the database.

- a) Archival
- b) Fuzzy
- c) SQL
- d) All of the mentioned

Answer: c

10. \_\_\_\_\_ dump schemes have been developed that allow transactions to be active while the dump is in progress.

- a) Archival
- b) Fuzzy
- c) SQL



d) All of the mentioned

Answer: b

1. Which lock should be obtained to prevent a concurrent transaction from executing a conflicting read, insert or delete operation on the same key value.

- a) Higher-level lock
- b) Lower-level lock
- c) Read only lock
- d) Read write

Answer: a

2. Once the lower-level lock is released, the operation cannot be undone by using the old values of updated data items, and must instead be undone by executing a compensating operation; such an operation is called

- a) Logical operation
- b) Redo operation
- c) Logical undo operation
- d) Undo operation

Answer: a

3. Which of the following is used for undo operations alone?

- a) Logical logging
- b) Physical logging
- c) Physical log records
- d) Physical logging and Physical log records

Answer: a

4. Redo operations are performed exclusively using

- a) Logical logging
- b) Physical logging
- c) Physical log records
- d) Both Physical logging and Physical log records

Answer: d

5. To perform logical redo or undo, the database state on disk must be operation \_\_\_\_\_ that is, it should not have partial effects of any operation.

- a) Persistent
- b) Resistant
- c) Consistent
- d) None of the mentioned

Answer: c

6. An operation is said to be \_\_\_\_\_ if executing it several times in a row gives the same result as executing it once.

- a) Idempotent
- b) Changed
- c) Repetitive
- d) All of the above

Answer: a

7. Immediate database modification technique uses

- a) Both undo and redo
- b) Undo but no redo
- c) Redo but no undo
- d) Neither undo nor redo

Answer: a

8. Shadow paging has

- a) no redo
- b) no undo
- c) redo but no undo
- d) neither redo nor undo

Answer: a

9. For correct behaviour during recovery, undo and redo operation must be

- a) Commutative
- b) Associative
- c) Idempotent

d) Distributive

Answer: c

10. If \_\_\_\_\_ are not obtained in undo operation it will cause problem in undo-phase

- a) Higher-level lock
- b) Lower-level lock
- c) Read only lock
- d) Read write

Answer: b

1. The remote backup site is sometimes also called the

- a) Primary Site
- b) Secondary Site
- c) Tertiary Site
- d) None of the mentioned

Answer: b

2. Remote backup system must be \_\_\_\_\_ with the primary site.

- a) Synchronised
- b) Separated
- c) Connected
- d) Detached but related

Answer: a

3. The backup is taken by

- a) Erasing all previous records
- b) Entering the new records
- c) Sending all log records from primary site to the remote backup site
- d) Sending selected records from primary site to the remote backup site

Answer: c

4. When the \_\_\_\_\_ the backup site takes over processing and becomes the primary.

- a) Secondary fails
- b) Backup recovers

- c) Primary fails
- d) None of the mentioned

Answer: c

5. The simplest way of transferring control is for the old primary to receive \_\_\_\_\_ from the old backup site.

- a) Undo logs
- b) Redo Logs
- c) Primary Logs
- d) All of the mentioned

Answer: c

6. The time to process the remote backup can be reduced by

- a) Flags
- b) Breakpoints
- c) Redo points
- d) Checkpoints

Answer: d

7. A \_\_\_\_\_ configuration can make takeover by the backup site almost instantaneous.

- a) Hot-spare
- b) Remote
- c) Direct
- d) Spare

Answer: d

8. A transaction commits as soon as its commit log record is written to stable storage at the primary site. This is

- a) One Safe
- b) Two Safe
- c) Two-very Safe
- d) Very Safe

Answer: a

9. A transaction commits as soon as its commit log record is written to stable storage at the primary and the backup site. This is

- a) One Safe
- b) Two Safe
- c) Two-very Safe
- d) Very Safe

Answer: c

10. If only the primary is active, the transaction is allowed to commit as soon as its commit log record is written to stable storage at the primary site. This is

- a) One Safe
- b) Two Safe
- c) Two-very Safe
- d) Very Safe

Answer: b

1. Which of the following best describes the purpose of a database backup?

- A) To improve query performance
- B) To protect data from accidental loss
- C) To optimize data storage
- D) To enforce data integrity

**\*\*Answer: B) To protect data from accidental loss\*\***

2. In a database recovery system, what is a "point-in-time recovery"?

- A) Recovering the entire database to a specific time
- B) Recovering individual records
- C) Rolling back the database to its initial state
- D) Creating a new database snapshot

**\*\*Answer: A) Recovering the entire database to a specific time\*\***

3. What is the primary purpose of a transaction log in a DBMS?

- A) To store user login information
- B) To maintain a history of executed queries
- C) To record changes made to the database
- D) To manage concurrent user connections

**\*\*Answer: C) To record changes made to the database\*\***

4. Which backup type captures all data changes since the last full backup and is typically faster to perform than a full backup?

- A) Full backup
- B) Incremental backup
- C) Differential backup
- D) Snapshot backup

**\*\*Answer: B) Incremental backup\*\***

5. What is the primary goal of the ACID properties in database transactions?

- A) To optimize query performance
- B) To ensure data redundancy
- C) To maintain data consistency
- D) To secure data access

**\*\*Answer: C) To maintain data consistency\*\***

6. Which recovery model in SQL Server allows for the capture of all changes, even if a full backup hasn't been taken?

- A) Simple Recovery Model
- B) Full Recovery Model
- C) Bulk-Logged Recovery Model
- D) Partial Recovery Model

**\*\*Answer: B) Full Recovery Model\*\***

7. What does the term "RPO" stand for in the context of database recovery?

- A) Recovery Point Objective
- B) Rollback Procedure Optimization
- C) Recovered Process Outcome
- D) Redundant Point of Origin

**\*\*Answer: A) Recovery Point Objective\*\***

8. In a database system, which of the following is NOT typically part of a backup strategy?

- A) Off-site storage of backups
- B) Frequent full backups
- C) Regular integrity checks
- D) Public key encryption

**\*\*Answer: D) Public key encryption\*\***

9. Which SQL statement is used to restore a database from a backup file in SQL Server?

- A) RESTORE DATABASE
- B) BACKUP DATABASE
- C) RECOVER DATABASE
- D) IMPORT DATABASE

**\*\*Answer: A) RESTORE DATABASE\*\***

10. Which type of database backup includes all the data that has changed since the last full backup and all previous incremental backups?

- A) Full backup
- B) Differential backup
- C) Incremental backup
- D) Log backup

**\*\*Answer: B) Differential backup\*\***

11. What is the purpose of a "cold backup" in a database system?

- A) To back up the database while it's actively running
- B) To create a backup when the database is offline
- C) To back up only the system tables
- D) To capture real-time data changes

**\*\*Answer: B) To create a backup when the database is offline\*\***

12. Which of the following is a common backup storage medium for databases?

- A) CD-ROM



- B) Floppy disk
- C) Magnetic tape
- D) Hard disk drive

**\*\*Answer: C) Magnetic tape\*\***

13. In a database recovery system, what is a "rollforward recovery"?

- A) Recovering the entire database to a specific time
- B) Reapplying committed transactions from the transaction log
- C) Restoring the database to a previous state
- D) Reverting changes made to the database

**\*\*Answer: B) Reapplying committed transactions from the transaction log\*\***

14. What does "RTO" stand for in the context of database recovery?

- A) Recovery Task Objective
- B) Rollback Time Optimization
- C) Recovery Time Objective
- D) Redundancy Test Outcome

**\*\*Answer: C) Recovery Time Objective\*\***

15. Which of the following is a primary goal of a disaster recovery plan for a database system?

- A) Improving query performance
- B) Ensuring data consistency
- C) Enhancing data storage efficiency

D) Securing network connections

**\*\*Answer: B) Ensuring data consistency\*\***

16. In the context of database recovery, what is the purpose of a "backup set"?

A) A group of users with backup privileges

B) A collection of backup files created at the same time

C) A set of log files for recovery

D) A temporary storage location for backups

**\*\*Answer: B) A collection of backup files created at the same time\*\***

17. What is a "point-of-failure" backup?

A) A backup taken at a specific time

B) A backup taken at a designated point in a transaction

C) A backup created when the system fails

D) A backup of system configuration settings

**\*\*Answer: B) A backup taken at a designated point in a transaction\*\***

18. Which of the following is NOT a key component of database recovery?

A) Backup

B) Rollback

C) Redo

D) Undo

**\*\*Answer: B) Rollback\*\***

19. What is a "full recovery" in a database context?

- A) Recovering the entire database from the last full backup
- B) Restoring the database to its initial state
- C) Recovering all data since the last incremental backup
- D) Reverting all committed transactions

**\*\*Answer: A) Recovering the entire database from the last full backup\*\***

20. Which backup method typically provides the fastest recovery time but requires more storage space than other methods?

- A) Full backup
- B) Incremental backup
- C) Differential backup
- D) Log backup

**\*\*Answer: A) Full backup\*\***

21. What does the term "log shipping" refer to in the context of database recovery?

- A) Shipping physical backup copies to a remote location
- B) Transmitting transaction logs to a standby server for recovery
- C) Distributing database backups to multiple servers
- D) Exporting database schemas to a different location

**\*\*Answer: B) Transmitting transaction logs to a standby server for recovery\*\***

22. In a database recovery scenario, what is the primary purpose of the "redo log"?

- A) To record changes made to the database
- B) To store a copy of the database for disaster recovery
- C) To maintain a history of executed queries
- D) To keep track of database administrators' actions

**\*\*Answer: A) To record changes made to the database\*\***

23. Which of the following is not typically part of a comprehensive backup strategy for a database?

- A) Regular database integrity checks
- B) Frequent full backups
- C) Encryption of all data at rest
- D) Version control of database schemas

**\*\*Answer: C) Encryption of all data at rest\*\***

24. What is the primary purpose of a "database snapshot" in a DBMS?

- A) To take a point-in-time backup of the entire database
- B) To create a read-only copy of the database
- C) To capture real-time changes in the database
- D) To optimize query performance

**\*\*Answer: B) To create a read-only copy of the database\*\***

25. Which of the following statements is true regarding "point-of-failure" backups?

- A) They are always taken at regular intervals.
- B) They capture all committed changes up to the point of failure.
- C) They are taken after every full backup.
- D) They are stored off-site by default.

**\*\*Answer: B) They capture all committed changes up to the point of failure.\*\***

26. In a database recovery context, what is the primary function of "database mirroring"?

- A) Creating a copy of the entire database
- B) Maintaining a redundant database server for failover
- C) Optimizing query performance
- D) Storing backup copies in a remote location

**\*\*Answer: B) Maintaining a redundant database server for failover\*\***

27. Which of the following backup methods is known for being the slowest to perform?

- A) Full backup
- B) Incremental backup
- C) Differential backup
- D) Log backup

**\*\*Answer: D) Log backup\*\***

28. What is the primary purpose of a "transaction log backup" in a DBMS?

- A) To take a backup of all log files

- B) To capture all transactions that occurred since the last backup
- C) To create a new transaction log file
- D) To take a backup of the entire database

**\*\*Answer: B) To capture all transactions that occurred since the last backup\*\***

29. In a database recovery context, what does "DBCC" stand for in Microsoft SQL Server?

- A) Database Configuration Control Center
- B) Database Consistency Checker
- C) Database Communication Channel
- D) Database Configuration Console

**\*\*Answer: B) Database Consistency Checker\*\***

30. What is a "hot backup" in the context of database recovery?

- A) A backup taken when the database is offline
- B) A backup taken while the database is actively running
- C) A backup of only the system tables
- D) A backup taken when the system is overheating

**\*\*Answer: B) A backup taken while the database is actively running\*\***

31. In a database recovery context, what is a "dirty read"?

- A) Reading data that is inconsistent due to a pending transaction
- B) Reading data that has been corrupted during a backup
- C) Reading data without proper authorization

D) Reading data from the transaction log

**\*\*Answer: A) Reading data that is inconsistent due to a pending transaction\*\***

32. Which of the following is NOT a common type of backup storage location for databases?

A) Cloud storage

B) External hard drive

C) Printer

D) Network-attached storage (NAS)

**\*\*Answer: C) Printer\*\***

33. What is a "hot standby" in the context of database recovery?

A) A secondary server that is kept up-to-date and can be used for failover

B) A server that overheats during database backups

C) A database backup taken when the system is active

D) A backup of the transaction log

**\*\*Answer: A) A secondary server that is kept up-to-date and can be used for failover\*\***

34. In the context of database recovery, what does "DBMS" stand for?

A) Database Management System

B) Data Backup and Management Service

C) Database Monitoring and Security

D) Distributed Backup Management System

**\*\*Answer: A) Database Management System\*\***

35. Which type of backup is the quickest to perform but provides the least granularity for recovery?

- A) Full backup
- B) Incremental backup
- C) Differential backup
- D) Log backup

**\*\*Answer: A) Full backup\*\***

36. What is a "data warehouse" in the context of database management?

- A) A type of database that stores historical data for analysis
- B) A highly available database for transaction processing
- C) A database for unstructured data storage
- D) A database used exclusively for backups

**\*\*Answer: A) A type of database that stores historical data for analysis\*\***

37. What is the purpose of a "backup retention policy" in a database recovery strategy?

- A) To decide when to take backups
- B) To define the maximum storage capacity for backups
- C) To specify how long backup copies are retained
- D) To set the encryption level for backups

**\*\*Answer: C) To specify how long backup copies are retained\*\***



38. Which of the following is not a common method of protecting database backups?

- A) Encryption
- B) Off-site storage
- C) Frequent deletion of backups
- D) Access controls

**\*\*Answer: C) Frequent deletion of backups\*\***

39. What does "disaster recovery" refer to in the context of database management?

- A) The process of recovering lost files
- B) The restoration of data from a backup
- C) The procedures and plans for recovering from major system failures
- D) The routine maintenance of a database system

**\*\*Answer: C) The procedures and plans for recovering from major system failures\*\***

40. In database recovery, what is the primary purpose of a "backup schedule"?

- A) To decide the format of backup files
- B) To define the frequency and timing of backups
- C) To specify the location of transaction logs
- D) To manage user permissions

**\*\*Answer: B) To define the frequency and timing of backups\*\***

41. What is a "full recovery" model in the context of database management?

- A) It involves recovering all data changes since the last full backup.
- B) It is a model for disaster recovery.
- C) It focuses on optimizing database performance.
- D) It requires no backups.

**\*\*Answer: A) It involves recovering all data changes since the last full backup.\*\***

42. Which of the following is a common technique used to ensure data consistency in a database recovery strategy?

- A) RAID (Redundant Array of Independent Disks)
- B) Snapshot backups
- C) Database normalization
- D) Replication

**\*\*Answer: C) Database normalization\*\***

43. In the context of database recovery, what is a "standby database"?

- A) A database that is always in a recovery state
- B) A redundant copy of the database used for failover
- C) A database that cannot be recovered
- D) A database used exclusively for reporting

**\*\*Answer: B) A redundant copy of the database used for failover\*\***

44. What is a "point-of-consistency" in a database recovery strategy?

- A) A specific point in time to recover to
- B) A point where all transactions are rolled back
- C) A point where data is in a consistent state
- D) A point at which new transactions are blocked

**\*\*Answer: C) A point where data is in a consistent state\*\***

45. Which of the following is NOT typically part of a disaster recovery plan for a database system?

- A) Regular backups
- B) Redundant hardware
- C) Load balancing
- D) Off-site data storage

**\*\*Answer: C) Load balancing\*\***

46. In a database recovery context, what does "PITR" stand for?

- A) Point in Time Recovery
- B) Primary Index Table Restoration
- C) Public Information Technology Resource
- D) Permanent Incremental Transaction Restore

**\*\*Answer: A) Point in Time Recovery\*\***

47. What is the primary purpose of "data archiving" in a database recovery strategy?

- A) To recover lost data

- B) To permanently delete data
- C) To store historical data for compliance and reference
- D) To create redundant copies of data

**\*\*Answer: C) To store historical data for compliance and reference\*\***

48. Which of the following backup methods captures all changes since the last full backup, without needing to perform a full backup each time?

- A) Full backup
- B) Differential backup
- C) Incremental backup
- D) Log backup

**\*\*Answer: C) Incremental backup\*\***

49. What is the primary purpose of a "recovery catalog" in a database recovery system?

- A) To manage user permissions
- B) To store backup metadata and information
- C) To execute backup commands
- D) To optimize query performance

**\*\*Answer: B) To store backup metadata and information\*\***

50. What is the role of a "backup administrator" in a database recovery strategy?

- A) Monitoring database queries
- B) Managing database transactions
- C) Overseeing the backup and recovery processes
- D) Designing database schemas

**\*\*Answer: C) Overseeing the backup and recovery processes\*\***