**Q. 1**  
Which of the following queries will **create a copy of the structure** of a table Customers into a new table Customers\_Copy, without copying the data?

CREATE TABLE Customers\_Copy AS SELECT \* FROM Customers WHERE 1 = 0;

**a)** Creates both the structure and data of Customers.  
**b)** Creates only the structure of Customers and no data.   
**c)** Creates only the data of Customers without structure.  
**d)** Copies data from Customers to Customers\_Copy.

**Q. 2**  
Which of the following **Relational Algebra operations** is used to return rows that **do not appear in another set**?

**a)** Union  
**b)** Intersect  
**c)** Minus   
**d)** Cartesian Product

**Q. 3**  
Which of the following **TCL commands** is used to **permanently save changes** made to a transaction?

**a)** ROLLBACK  
**b)** COMMIT   
**c)** SAVEPOINT  
**d)** GRANT

**Q. 4**  
Given the following **SQL statement**:

SELECT \* FROM Orders WHERE EXISTS (SELECT 1 FROM Customers WHERE Customers.CustomerID = Orders.CustomerID);

Which type of subquery is being used?

**a)** Simple Subquery  
**b)** Correlated Subquery   
**c)** Inline View  
**d)** EXISTS Subquery

**Q. 5**  
Which of the following **SQL statements** can be used to **revoke** the privilege of selecting from the Employees table?

**a)** REVOKE SELECT ON Employees FROM User1;   
**b)** GRANT SELECT ON Employees TO User1;  
**c)** DENY SELECT ON Employees FROM User1;  
**d)** REMOVE SELECT ON Employees FROM User1;

**Q. 6**  
Which SQL command is used to **define a view** named EmployeeView that selects EmployeeID and EmployeeName from the Employees table?

CREATE VIEW EmployeeView AS

SELECT EmployeeID, EmployeeName FROM Employees;

What will this query do?

**a)** It will create a **temporary** view.  
**b)** It will create a **permanent** view.   
**c)** It will create a new table.  
**d)** It will create a new stored procedure.

**Q. 7**  
Given the following SQL query, what is the purpose of the **EXISTS** keyword?

SELECT \* FROM Orders

WHERE EXISTS (SELECT \* FROM Customers WHERE Orders.CustomerID = Customers.CustomerID);

**a)** Checks for records where Orders.CustomerID matches Customers.CustomerID.  
**b)** Returns all rows from Orders where the Customers table contains any matching records.   
**c)** Ensures that Customers table has no matching records.  
**d)** Filters rows that exist only in the Orders table.

**Q. 8**  
What will this **SQL query** do?

SELECT COUNT(\*) FROM Orders

WHERE OrderDate BETWEEN '2022-01-01' AND '2022-12-31';

**a)** Count of all orders placed in the year 2022.   
**b)** Count of all orders where the OrderDate is null.  
**c)** Count of all orders where the OrderDate is after 2022-12-31.  
**d)** Count of all orders placed before 2022.

**Q. 9**  
What does this **SQL query** return?

SELECT DISTINCT DepartmentID FROM Employees;

**a)** Returns all unique department IDs from the Employees table.   
**b)** Returns all department IDs, including duplicates.  
**c)** Returns only the first department ID found.  
**d)** Returns all columns from the Employees table, with duplicates.

**Q. 10**  
Which of the following **TCL commands** is used to **undo** changes made in the current transaction?

**a)** COMMIT  
**b)** ROLLBACK   
**c)** SAVEPOINT  
**d)** GRANT

**Code Snippets**

**Q. 11**  
Given the following SQL query, what does the **AUTO\_INCREMENT** feature do?

CREATE TABLE Employees (

EmployeeID INT AUTO\_INCREMENT,

EmployeeName VARCHAR(100),

PRIMARY KEY (EmployeeID)

);

**a)** Automatically increments the EmployeeID each time a new row is inserted.   
**b)** Automatically sets the EmployeeID to a constant value.  
**c)** Automatically sets the EmployeeID to the maximum value of EmployeeID.  
**d)** Automatically inserts a value into the EmployeeName column.

**Q. 12**  
What is the main **benefit of using indexes** in a database?

**a)** Improves the performance of INSERT and UPDATE operations  
**b)** Speeds up data retrieval operations such as SELECT queries   
**c)** Reduces the need for normalization  
**d)** Ensures data consistency across tables

**Q. 13**  
Which of the following **index types** is used in MySQL to speed up searches and ensures that values in the indexed column are unique?

**a)** Full-text Index  
**b)** Hash Index  
**c)** Unique Index   
**d)** Spatial Index

**Q. 14**  
In which **MySQL Storage Engine** are **transactions, foreign key constraints**, and **ACID properties** supported?

**a)** MyISAM  
**b)** InnoDB   
**c)** MEMORY  
**d)** MERGE

**Q. 15**  
Which of the following **ACID properties** ensures that a transaction will either be fully completed or not executed at all in case of a failure?

**a)** Atomicity   
**b)** Consistency  
**c)** Isolation  
**d)** Durability

**Q. 16**  
Which of the following **statements** about **Temporary Tables** is true in MySQL?

**a)** Temporary tables are automatically dropped when the session ends   
**b)** Temporary tables are visible to all sessions  
**c)** Temporary tables must be dropped manually to release resources  
**d)** Temporary tables can only be used with the InnoDB storage engine

**Q. 17**  
What is the difference between **a database instance** and a **schema** in MySQL?

**a)** A database instance refers to the actual data while schema defines the structure of the database.   
**b)** A database instance is a set of schemas, and a schema is a collection of users.  
**c)** A database instance and schema are the same thing in MySQL.  
**d)** A database instance is only used in cloud databases, while a schema is used in relational databases.

**Q. 18**  
Which **MySQL command** is used to **create a temporary table**?

CREATE TEMPORARY TABLE temp\_table (

id INT PRIMARY KEY,

name VARCHAR(100)

);

What does this query do?

**a)** Creates a temporary table that will be available for the duration of the session   
**b)** Creates a permanent table that will persist across sessions  
**c)** Creates a table that can only be used for SELECT queries  
**d)** Creates a table that will be dropped after one day

**Q. 19**  
Which of the following **statements** is **correct** about **stored procedures**?

**a)** Stored procedures can have **IN**, **OUT**, and **INOUT** parameters to handle input and output values   
**b)** Stored procedures cannot accept parameters  
**c)** Stored procedures are only used to insert data into a table  
**d)** Stored procedures are only available in MySQL's InnoDB storage engine

**Q. 20**  
What is the **primary advantage** of using **stored procedures** in MySQL?

**a)** Stored procedures increase the complexity of queries  
**b)** Stored procedures are not reusable once created  
**c)** Stored procedures improve performance by reducing network traffic and providing encapsulation   
**d)** Stored procedures are faster than direct SQL queries in every case

**Q. 21**  
Given the following code, which **parameter mode** does the procedure CalculateSalary use for the salary parameter?

DELIMITER $$

CREATE PROCEDURE CalculateSalary(IN employeeID INT, OUT employeeSalary DECIMAL(10,2))

BEGIN

SELECT salary INTO employeeSalary FROM Employees WHERE EmployeeID = employeeID;

END$$

DELIMITER ;

**a)** IN  
**b)** OUT   
**c)** INOUT  
**d)** IN/OUT

**Q. 22**  
Given the following code snippet, what will happen when the procedure UpdateEmployeeDetails is called?

CREATE PROCEDURE UpdateEmployeeDetails(IN empID INT, IN empName VARCHAR(100), IN empDept VARCHAR(100))

BEGIN

UPDATE Employees SET Name = empName, Department = empDept WHERE EmployeeID = empID;

END;

**a)** The procedure will update the Name and Department columns for the specified EmployeeID.   
**b)** The procedure will create a new employee with the provided details.  
**c)** The procedure will delete the employee with the provided EmployeeID.  
**d)** The procedure will insert a new record for the provided empID.

**Q. 23**  
Given the following **SQL** code, how does **MySQL handle indexes** in relation to **queries**?

SELECT Name, Department FROM Employees WHERE EmployeeID = 101;

**a)** MySQL will use an **index** on EmployeeID if one exists, to quickly locate the record.   
**b)** MySQL will always perform a full table scan, ignoring any indexes.  
**c)** MySQL will create a new index on the fly for the EmployeeID column.  
**d)** MySQL will ignore the index if EmployeeID is a primary key.

**Q. 24**  
Which **SQL command** can be used to **remove** a stored procedure in MySQL?

**a)** DELETE PROCEDURE procedure\_name;  
**b)** DROP PROCEDURE procedure\_name;   
**c)** REMOVE PROCEDURE procedure\_name;  
**d)** ALTER PROCEDURE procedure\_name;

**Q. 25**  
What would be the result of the following SQL statement?

CREATE PROCEDURE AddEmployee(IN employeeName VARCHAR(100), IN employeeDept VARCHAR(100))

BEGIN

INSERT INTO Employees (Name, Department) VALUES (employeeName, employeeDept);

END;

**a)** It creates a stored procedure to add an employee to the Employees table   
**b)** It adds an employee to the Emp table immediately upon execution  
**c)** It creates a view that displays employee details  
**d)** It updates an existing employee record

**Q. 26**  
Consider the following stored procedure. What will happen when GetEmployeeInfo is called?

CREATE PROCEDURE GetEmployeeInfo(IN empID INT)

BEGIN

SELECT Name, Department FROM Employees WHERE EmployeeID = empID;

END;

**a)** It will return the Name and Department of the employee with the given empID   
**b)** It will return all employees' details in the database  
**c)** It will insert a new employee record  
**d)** It will delete the employee with the given empID

**Q. 27**  
Which **flow control** statement will **repeat** a block of code as long as the given condition is **true**?

**a)** LOOP  
**b)** WHILE   
**c)** REPEAT  
**d)** ITERATE

**Q. 28**  
Which **MySQL flow control** statement is used to **exit a loop** prematurely in a stored procedure?

**a)** LEAVE   
**b)** ITERATE  
**c)** BREAK  
**d)** EXIT

**Q. 29**  
Which of the following **conditional statements** in MySQL is **not supported**?

**a)** IF-ELSE  
**b)** IF-ELSE-THEN  
**c)** SWITCH CASE   
**d)** CASE WHEN

**Q. 30**  
Consider the following **stored procedure**. What will happen when executed?

CREATE PROCEDURE CheckSalary(IN empID INT)

BEGIN

DECLARE empSalary DECIMAL(10,2);

SELECT Salary INTO empSalary FROM Employees WHERE EmployeeID = empID;

IF empSalary > 50000 THEN

SELECT 'High Salary';

ELSE

SELECT 'Low Salary';

END IF;

END;

**a)** It will check if the salary of an employee is greater than 50,000 and return either 'High Salary' or 'Low Salary'   
**b)** It will return an error because salary values are not allowed to be stored in variables  
**c)** It will update the employee's salary based on the condition  
**d)** It will throw an error because there is no error handling

**Q. 31**  
Which **flow control statement** is used to **continue** to the next iteration of a loop without executing the remaining statements for the current iteration?

**a)** LEAVE  
**b)** ITERATE   
**c)** CONTINUE  
**d)** EXIT

**Q. 32**  
Given the following code, which of the following **flow control constructs** are used inside the stored procedure?

CREATE PROCEDURE CalculateBonus()

BEGIN

DECLARE bonus DECIMAL(10,2);

DECLARE done INT DEFAULT FALSE;

DECLARE cur CURSOR FOR SELECT Salary FROM Employees;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO bonus;

IF done THEN

LEAVE read\_loop;

END IF;

UPDATE Employees SET Salary = Salary + bonus WHERE Salary = bonus;

END LOOP;

CLOSE cur;

END;

**a)** LOOP and LEAVE   
**b)** REPEAT and ITERATE  
**c)** IF and WHILE  
**d)** SWITCH CASE and ITERATE

**Q. 33**  
Which of the following **MySQL Built-in functions** is used to **round a number** to a specified number of decimal places?

**a)** ROUND(**num**, **decimal places**)   
**b)** TRUNCATE(**num**, **decimal places**)  
**c)** FLOOR(**num**)  
**d)** CEIL(**num**)

**Q. 34**  
In the following **MySQL function**, what will it return?

CREATE FUNCTION GetTotalSalary() RETURNS DECIMAL(10,2)

BEGIN

DECLARE totalSalary DECIMAL(10,2);

SELECT SUM(Salary) INTO totalSalary FROM Employees;

RETURN totalSalary;

END;

**a)** It will return the total salary of all employees   
**b)** It will return the highest salary of an employee  
**c)** It will return the number of employees  
**d)** It will return an error because functions cannot return values

**Q. 35**  
Which of the following **conditional statements** will execute the code block if the condition is **false**?

**a)** IF  
**b)** IF-ELSE   
**c)** CASE  
**d)** SWITCH CASE

**Q. 36**  
Given the following SQL code for a **function**:

CREATE FUNCTION CalculateTax(salary DECIMAL(10,2)) RETURNS DECIMAL(10,2)

BEGIN

IF salary <= 50000 THEN

RETURN salary \* 0.05;

ELSE

RETURN salary \* 0.10;

END IF;

END;

What does this **function** do?

**a)** It calculates the tax based on a salary threshold of 50,000, with 5% tax for salaries up to 50,000 and 10% for higher salaries   
**b)** It returns a fixed tax amount of 5% on all salaries  
**c)** It calculates the bonus for employees based on their salaries  
**d)** It applies a tax rate of 15% regardless of salary

**Q. 37**  
Which **MySQL loop construct** is used to repeatedly execute a block of code **until a condition is false**?

**a)** LOOP  
**b)** REPEAT   
**c)** WHILE  
**d)** ITERATE

**Q. 38**  
Given the following **stored procedure**, what is the purpose of the LEAVE statement?

CREATE PROCEDURE ProcessEmployeeData()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE cur CURSOR FOR SELECT Name FROM Employees;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO @name;

IF done THEN

LEAVE read\_loop;

END IF;

-- Process data here

END LOOP;

CLOSE cur;

END;

**a)** It exits the LOOP and ends the execution of the procedure  
**b)** It continues the loop for the next iteration  
**c)** It pauses the loop and waits for user input  
**d)** It initializes the cursor to begin fetching rows

**Q. 39**  
What will the following **MySQL code** do?

CREATE PROCEDURE GetEmployeeBonus(IN empID INT)

BEGIN

DECLARE empBonus DECIMAL(10,2);

DECLARE exit\_loop BOOLEAN DEFAULT FALSE;

DECLARE employee\_cursor CURSOR FOR

SELECT Bonus FROM Employees WHERE EmployeeID = empID;

OPEN employee\_cursor;

FETCH employee\_cursor INTO empBonus;

IF empBonus > 5000 THEN

SELECT 'Bonus is High';

ELSE

SELECT 'Bonus is Low';

END IF;

CLOSE employee\_cursor;

END;

**a)** It checks the bonus of the employee and returns whether it is 'High' or 'Low' based on the value   
**b)** It returns an error because the FETCH statement cannot be used with a cursor in this way  
**c)** It updates the bonus of the employee  
**d)** It deletes the employee record from the table

**Q. 40**  
Which **MySQL function** would be used to **extract the year** from a given date?

SELECT YEAR('2023-08-15');

**a)** EXTRACT()  
**b)** DATE\_PART()  
**c)** YEAR()   
**d)** DATE\_FORMAT()