Q1. Examine the following statement.

mysql> SELECT if (10=10, 1, 2);

What will be the output?

1. **1**
2. 2
3. Null
4. The statement is invalid.

Q2. Examine the following statement.

mysql> SELECT if (10 <=> 10, 1, 2);

What will be the output?

1. **1**
2. 2
3. Null
4. The statement is invalid.

Q3. Examine the following statement.

mysql> SELECT if(10="10",'A',"a");

What will be the output?

1. **A**
2. a
3. Error: because lowercase "a" is given double quotes.
4. None of the above.

Q4. Examine the following statement.

mysql> SELECT if(strcmp('A', 'a'),1,'Two');

What will be the output?

1. 1
2. **Two**
3. Error: because the third parameter is a string
4. None of the above.

Q5. Examine the following statement.

mysql> SELECT if(10="10",'A',"a");

What will be the output?

1. **A**
2. a
3. Error: because integer value "10" is passed as string.
4. None of the above.

Q6. Examine the following statement.

mysql> SELECT if(strcmp('A', 'A'),1,'Two');

What will be the output?

1. **1**
2. Two
3. Error: because the third parameter is a string
4. None of the above.

Q7. Examine the following statement.

mysql> SELECT if (ASCII('AB') = 65 , 'The char is A', 'The char is B');

What will be the output?

1. **The char is A**
2. The char is B
3. Error: because only single char is given in ASCII string function.
4. None of the above.

Q8. Examine the following statement.

mysql> SELECT if (null , 'The char is A', 'The char is B');

What will be the output?

1. The char is A
2. **The char is B**
3. NULL
4. Error: because invalid condition in if condition.

Q9. Examine the following statement.

mysql> SELECT if (null = null , 'String One', 'String Two');

What will be the output?

1. String One
2. **String Two**
3. NULL
4. Error: because invalid comparisons of null value.

Q10. Examine the following statement.

mysql> SELECT if (null is null , 'String One', 'String Two');

What will be the output?

1. **String One**
2. String Two
3. NULL
4. Error: because invalid comparisons of null value.

Q11. Examine the following statement.

mysql> SELECT if (65 = ASCII('A') , 'String One', 'String Two');

What will be the output?

1. **String One**
2. String Two
3. NULL
4. None of the above.

Q12. Examine the following statement.

mysql> SELECT ifnull(1,0) from dual;

What will be the output?

1. 0
2. **1**
3. NULL
4. Error: because invalid table name.

Q13. Examine the following statement.

mysql> SELECT ifnull(null,0) from dual;

What will be the output?

1. **0**
2. 1
3. NULL
4. Error: because invalid table name.

Q14. Examine the following statement.

mysql> SELECT ifnull('null',0) from dual;

What will be the output?

1. 0
2. 1
3. **null**
4. Error: because invalid table name.

Q15. Examine the following statement.

mysql> SELECT ifnull('0',0) from dual;

What will be the return datatype of the output?

1. **string**
2. int
3. Error: because string data cannot be compared with int data.
4. None of the above

Q16. Examine the following statement.

mysql> SELECT ifnull('Yes', ifnull('No', "Can't say")) from dual;

What will be output?

1. **Yes**
2. No
3. Can’t say
4. Null

Q17. Examine the following statement.

mysql> SELECT ifnull('Yes', ifnull(NULL, "Can't say")) from dual;

What will be the output?

1. **Yes**
2. No
3. Can’t say
4. Null

Q18. Examine the following statement.

mysql> SELECT ifnull(null, ifnull('Yes', "Can't say")) from dual;

What will be the output?

1. **Yes**
2. No
3. Can’t say
4. Null

Q19. Examine the following statement.

mysql> SELECT ifnull (null, ifnull (null, "Can't say")) from dual;

What will be the output?

1. 1
2. 0
3. **Can’t say**
4. Null

Q20. Examine the following statement.

mysql> SELECT ifnull (null, ifnull ("Can't say", null)) from dual;

What will be the output?

1. 0
2. 1
3. **Can’t say**
4. Null

Q21. Examine the following statement.

mysql> CREATE TABLE TEMP SELECT IFNULL ('Infoway', 'Pune') AS Address;

What will be the result of the statement?

1. **TEMP table is created.**
2. Infoway table is created
3. Address table is created
4. Error: select cannot be used with ddl command (create).

Q22. Examine the following statement.

mysql> CREATE TABLE TEMP SELECT IFNULL ('Infoway', 'Pune') AS Address;

mysql> DESC TEMP;

How many number of columns will be there in TEMP table?

1. 0
2. **1**
3. 2
4. 3

Q23. Examine the following statement.

mysql> CREATE TABLE TEMP SELECT IFNULL ('Infoway', 'Pune') AS Address;

mysql> DESC TEMP;

What will be the name of column in TEMP table?

1. Infoway
2. Pune
3. **Address**
4. None of the above.

Q24. Examine the following statement.

mysql> CREATE TABLE TEMP SELECT IFNULL (null, 'Pune') AS Address;

mysql> DESC TEMP;

What will be the name of column in TEMP table?

1. Infoway
2. Pune
3. **Address**
4. None of the above.

Q25. Examine the following statement.

mysql> CREATE TABLE TEMP SELECT IFNULL (null, null) AS Address;

mysql> DESC TEMP;

What will be the name of column in TEMP table?

1. null
2. nullAddress
3. **Address**
4. None of the above.

Q26. Examine the following statement.

mysql> CREATE TABLE T SELECT IFNULL ('Infoway', 'Pune') AS Address;

What value the statement will return after giving the following select command?

mysql> select \* from TEMP;

1. **Infoway**
2. Pune
3. Infoway and Pune
4. Null

Q27. Examine the following statement.

mysql> CREATE TABLE T SELECT IFNULL (null, 'Pune') AS Address;

What value the statement will return after giving the following select command?

mysql> select \* from TEMP;

1. Infoway
2. **Pune**
3. Infoway and Pune
4. Null

Q28. Examine the following statement.

mysql> CREATE TABLE T SELECT IFNULL ('Infoway', null) AS Address;

What value the statement will return after giving the following select command?

mysql> select \* from TEMP;

1. **Infoway**
2. Pune
3. Null
4. None of the above.

Q29. Examine the following statement.

mysql> CREATE TABLE T SELECT IFNULL (null, null) AS Address;

What value the statement will return after giving the following select command?

mysql> select \* from TEMP;

1. Infoway
2. Pune
3. **Null**
4. None of the above.

Q30. Which line of the following code has an error?

SELECT \*

FROM EMP

Where comm = NULL

Order by ename;

1. Select \*
2. From emp
3. Where comm = NULL
4. **There is no error in this statement.**

Q31. Which statements are true about NULL values?

1. You cannot search for a NULL value in column using the where clause.
2. Only = and != operator can be used to search for NULL values in a column.
3. **In an ascending order sort, NULL values appear at the top of the result set.**
4. None of the above.

Q32. Which statements are true about NULL values?

1. You cannot search for a NULL value in column using the where clause.
2. Only = and != operator can be used to search for NULL values in a column.
3. **In descending order sort, NULL values appear at the bottom of the result set.**
4. None of the above.

Q33. The following statements will raise an exception on which line?

Select dept\_name, avg (all salary), count (\*) “number of employees”

From EMP, dept

Where deptno=dept\_no

And count (\*) >5

Group by dept\_name Order by 2 desc;

1. Select dept\_name, avg (all salary), count (\*) “number of employees”
2. Where deptno=dept\_no
3. **And count (\*) >5**
4. Group by dept\_name Order by 2 desc;

Q34. Which keywords are used to specify a Cartesian join using this syntax?

1. NATURAL JOIN
2. CARTESIAN JOIN
3. INNER JOIN
4. **CROSS JOIN**

Q35. Outer joins in MySQL can be specified using the syntax <table name> LEFT OUTER JOIN <table name>. Which keyword is optional?

1. JOIN
2. **OUTER**
3. JOIN AND OUTER
4. None

Q36. A user is setting up a join operation between EMP and DEPT tables. The query should return all the employees, who are assigned dept, as well as the employees, which are yet not assigned any DEPT.

1. **Select \* from EMP right outer join DEPT on emp.deptno = dept.deptno;**
2. Select \* from EMP right outer join DEPT where emp.deptno = dept.deptno;
3. Select e.empid, d.head from EMP e, DEPT d where e.deptno = d.deptno (+);
4. Select e.empid, d.head from EMP e, DEPT d where e.deptno (+) = d.deptno;

Q37. Output from a table called PLAYS with two columns, PLAY\_NAME and AUTHOR, is shown next. Which of the following SQL statements produced it?

PLAY\_TABLE

"Midsummer Night's Dream", SHAKESPEARE

"Waiting For Godot", BECKETT

"The Glass Menagerie", WILLIAMS

1. Select PLAY\_NAME|| AUTHOR from PLAYS;
2. Select PLAY\_NAME, AUTHOR from PLAYS;
3. Select PLAY\_NAME||', ' || AUTHOR from PLAYS;
4. **Select PLAY\_NAME||', ' || AUTHOR play\_table from PLAYS;**

Q38. Which of the following statements contains an error?

1. Select \* from EMP where EMPID = 493945;
2. Select EMPID from EMP where EMPID = 493945;
3. Select EMPID from EMP;
4. **Select EMPID where EMPID = 56949 and LASTNAME = 'SMITH';**

Q39. Which of the following statements are NOT TRUE about ORDER BY clauses?

1. Ascending or descending order can be defined with the asc or desc keywords.
2. **Only one column can be used to define the sort order in an order by clause.**
3. Multiple columns can be used to define sort order in an order by clause.
4. Columns can be represented by numbers indicating their listed order in the select clause within order by.

Q40. A user is setting up a join operation between tables EMP and DEPT. There are some employees in the EMP table that the user wants returned by the query, but the employees are not assigned to department heads yet. Which SELECT statement is most appropriate for this user?

1. Select e.empid, d.head from EMP e, dept d;
2. Select e.empid, d.head from EMP e, dept d where e.dept# = d.dept#;
3. Select e.empid, d.head from EMP e, dept d where e.dept# = d.dept# (+);
4. **Select \* from emp right outer join dept on emp.deptno=dept.deptno;**

Q41. Consider a table OLD with few rows. The statement

CREATE TABLE new AS SELECT \* FROM OLD WHERE 1=2;

1. Copies all the rows from OLD to NEW
2. Creates a dummy NEW table
3. **Creates an empty NEW table with column definitions same as OLD**
4. Creates a table NEW that acts as a synonym for OLD

Q42. You perform the following two queries:

1. SELECT empno enumber, ename FROM EMP order by 1;

2. SELECT empno, ename FROM EMP order by empno ASC;

Which of the following is true?

1. **Statement 1 & 2 will produce the same result.**
2. Statement 1 will execute; statement 2 will return error.
3. Statement 2 will execute; statement 1 will return error.
4. Statement 1 & 2 both will execute but produce different results.

Q43. You issue the following SELECT statement on the EMP table.

SELECT (200 + ((salary \* 0.1) / 2)) FORM EMP;

What will happen to the result if all of the parentheses are removed?

1. No difference, because the answer will always be NULL.
2. **No difference, because the result will be same.**
3. The result will be higher.
4. The result will be lower.

Q44. What will happen if you query the EMP table?

SELECT empno, DISTINCT ename, Sal from EMP;

1. EMPNO, unique values of ENAME and then SAL are displayed.
2. EMPNO, unique values of the two columns, ENAME and SAL are displayed.
3. Distinct is not a valid keyword in SQL.
4. **No values will be displayed because the statement will return an error.**

Q45. How many rows will the following query returns?

SELECT \* FROM EMP WHERE ename BETWEEN 'A' and 'C'

1. All the rows who’s first character in their name are <= 'C'
2. **All the rows who’s first character in their name are < 'C'**
3. A character column cannot be used in the BETWEEN operator
4. None of the above.

Q46. When you issue the following query, which line has an error?

SELECT empno, "Enumber", ename "EmpName"

FROM EMP

WHERE deptno = 10

AND "Enumber" = 7788

ORDER By "Enumber";

1. 1
2. 4
3. 5
4. **No error, the statement will finish successfully.**

Q47. When you issue the following query, which line has an error?

SELECT empno, "Enumber", ename "EmpName"

FROM EMP

WHERE deptno = 10

AND "Enumber" = "Enumber"

ORDER By "Enumber";

1. 1
2. 4
3. 5
4. **Displays all record who’s deptno=10.**

Q48. When you issue the following query;

SELECT empno, ename From EMP where empno=7788 or empno = 7876;

Which other operator can replace the OR condition in the WHERE clause?

1. **IN**
2. BETWEEN ... AND …
3. LIKE
4. <=

Q49. Which statement searches for PRODUCT\_ID values that begin with DI\_ from the ORDERS table?

1. SELECT \* FROM ORDERS WHERE PRODUCT\_ID = 'DI%';
2. **SELECT \* FROM ORDERS WHERE PRODUCT\_ID LIKE 'DI/\_';**
3. SELECT \* FROM ORDERS WHERE PRODUCT\_ID LIKE 'DI\_' ESCAPE '\';
4. SELECT \* FROM ORDERS WHERE PRODUCT\_ID LIKE 'DI\\_%' ESCAPE '\';

Q50. Which statement searches for PRODUCT\_ID values that begin with DI\_ from the ORDERS table?

1. SELECT \* FROM ORDERS WHERE PRODUCT\_ID = 'DI%';
2. SELECT \* FROM ORDERS WHERE PRODUCT\_ID LIKE '\DI\_';
3. **SELECT \* FROM ORDERS WHERE PRODUCT\_ID LIKE 'DI/\_%' ESCAPE '/';**
4. SELECT \* FROM ORDERS WHERE PRODUCT\_ID LIKE 'DI\\_%' ESCAPE '\';

Q51. COUNTRY\_NAME and REGION\_ID are valid column names in the COUNTRIES table. Which one of the following statements will execute without an error?

1. SELECT country\_name, region\_id,

CASE region\_id = 1 THEN 'Europe',

Region\_id = 2 THEN 'America',

Region\_id = 3 THEN 'Asia',

ELSE 'Other' END Continent

FROM Countries;

1. SELECT country\_name, region\_id,

CASE (region\_id WHEN 1 then 'Europe',

WHEN 2 then ' America',

WHEN 3 then 'Asia',

ELSE 'Other') Continent

FROM countries;

1. **SELECT country\_name, region\_id,**

**CASE region\_id WHEN 1 then 'Europe',**

**WHEN 2 then 'America',**

**WHEN 3 then 'Asia',**

**ELSE 'Other' END Continent**

**FROM countries;**

1. SELECT country\_name, region\_id,

CASE region\_id WHEN 1 then 'Europe',

WHEN 2 then 'America',

WHEN 3 then 'Asia',

ELSE 'Other' Continent

FROM countries;

Q52. Which SQL statement will query the EMPLOYEES table for FIRST\_NAME, LAST\_NAME AND SALARY of all employees in DEPARTMENT\_ID 40 in the alphabetical order of last name?

1. Select frist\_name, last\_name salary from employees ORDER By last\_name where department\_id = 40;
2. Select first\_name, last\_name, salary from employees ORDER BY last\_name ASC where department\_id = 40;
3. Select first\_name last\_name salary from employees where department\_id = 40 ORDER BY last\_name ASC;
4. **Select first\_name, last\_name, salary from employees where department\_id = 40 ORDER BY last\_name;**

Q53. Write a query to display the names (first\_name, last\_name) using alias name “First Name", "Last Name".

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT first\_name "First Name", last\_name "Last Name" FROM employees;**
2. SELECT "First Name" first\_name, last\_name "Last Name" FROM employees;
3. SELECT first\_name "First Name", "Last Name" last\_name FROM employees;
4. SELECT first\_name First Name, last\_name Last Name FROM employees;

Q54. Write a query to get unique department ID from employee table

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. SELECT DISTINCT department\_id FROM employees;
2. SELECT UNIQUE department\_id FROM employees;
3. SELECT DISTINCTROW department\_id FROM employees;
4. **Both A and C**

Q55. Write a query to get all employee details from the employee table order by first name, descending

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. SELECT \* FROM employees ORDER BY first\_name DESC;
2. SELECT \* FROM employees ORDER BY 2 DESC;
3. SELECT \* FROM employees ORDERED BY first\_name DESC;
4. **Both A and B**

Q56. Write a query to get the names (first\_name, last\_name), salary, PF of all the employees (PF is calculated as 12% of salary).

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. SELECT first\_name, last\_name, salary, salary\*.15 PF FROM employees;
2. SELECT first\_name, last\_name, salary, .15\*salary PF FROM employees;
3. SELECT first\_name, last\_name, salary, salary\*.15 as PF FROM employees;
4. **All of the above.**

Q57. Write a query to get the employee ID, names (first\_name, last\_name), salary in ascending order of salary

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. SELECT employee\_id, first\_name, last\_name, salary New\_Salary FROM EMPLOYEES ORDER BY New\_Salary;
2. SELECT employee\_id, first\_name, last\_name, salary \_NewSalary FROM EMPLOYEES ORDER BY \_NewSalary;
3. SELECT employee\_id, first\_name, last\_name, salary FROM EMPLOYEES ORDER BY salary;
4. **All of the above.**

Q58. Write a query to get the total salaries payable to employees

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. SELECT SUM salary FROM employees;
2. **SELECT SUM (salary) FROM employees;**
3. SELECT SUM of (salary) FROM employees;
4. None of the above.

Q59. Write a query to get the maximum and minimum salary from employees table

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. SELECT MAX (salary) "MAX\_Salary", MIN (salary) "MIN\_SAL" FROM employees;
2. SELECT MAX (salary) MAX\_Salary, MIN (salary) MIN\_SAL FROM employees;
3. SELECT MAX (salary) as MAX\_Salary, MIN (salary) as MIN\_SAL FROM employees;
4. **All of the above.**

Q60. Write a query to get the average salary and number of employees in the employees table

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT AVG (salary), COUNT (\*) FROM employees;**
2. SELECT AVERAGE (salary), COUNT (\*) FROM employees;
3. SELECT AVG (salary), CNT (\*) FROM employees;
4. All of the above.

Q61. Write a query to get the number of employees working with the company

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT COUNT(\*) FROM employees;**
2. SELECT TOTALCOUNT(\*) FROM employees;
3. SELECT COUNTOF(\*) FROM employees;
4. None of the above

Q62. Write a query to get the number of jobs available in the employees table

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT DISTINCT (COUNT job\_id) FROM employees;**
2. SELECT COUNT(DISTINCT job\_id) FROM employees;
3. Both A and B
4. None of the above.

Q63. Write a query get all first name from employees table in upper case.

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. SELECT UCASE(first\_name) FROM employees;
2. SELECT UPPER(first\_name) FROM employees;
3. SELECT UPPERCASE(first\_name) FROM employees;
4. **Both A and B**

Q64. Write a query to get the first 3 characters of first name from employees table.

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT SUBSTR (first\_name, 1, 3) FROM employees;**
2. SELECT SUB\_STR (first\_name, 1, 3) FROM employees;
3. SELECT first\_name (SUBSTR, 1, 3) FROM employees;
4. All of the above.

Q65. Write a query to get the first 3 characters of first name from employees table.

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT SUBSTRING (first\_name, 1, 3) FROM employees;**
2. SELECT STRINGSUB (first\_name, 1, 3) FROM employees;
3. SELECT SUB-STRING (first\_name, 1, 3) FROM employees;
4. All of the above.

Q66. Write a query to calculate 171\*214+625

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. SELECT 171\*214+625 Result;
2. SELECT (171\*214+625) Result;
3. SELECT 171\*214+625 Result from dual;
4. **All of the above**

Q67. Write a query to get first name from employees table after removing white spaces from both side.

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT TRIM(first\_name) FROM employees;**
2. SELECT first\_name.TRIM FROM employees;
3. SELECT TRIM().first\_name FROM employees;
4. All of the above

Q68. Write a query to get the length of the employee names (first\_name, last\_name) from employees table.

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT first\_name, last\_name, LENGTH (first\_name) + LENGTH (last\_name) 'Length of Names ' FROM employees;**
2. SELECT first\_name, last\_name, LENGTH (first\_name + last\_name) 'Length of Names' FROM employees;
3. SELECT first\_name, last\_name, LEN (first\_name) + LEN (last\_name) 'Length of Names' FROM employees;
4. None of the above.

Q69. Write a query to check if the first\_name fields of the employees table contains numbers.

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT \* FROM employees WHERE first\_name REGEXP '[0-9]';**
2. SELECT \* FROM employees WHERE is\_number (first\_name);
3. SELECT \* FROM employees WHERE NUMERICAL (first\_name);
4. SELECT \* FROM employees WHERE IS\_NUMERICAL (first\_name);

Q70. Write a query to select first 10 records from a table.

**Sample table: employees**

(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID)

1. **SELECT employee\_id, first\_name FROM employees LIMIT 10;**
2. SELECT employee\_id, first\_name FROM employees LIMITS 10;
3. SELECT employee\_id, first\_name FROM employees LIMITED 10;
4. All of the above

Q71. Write a query to get Product name and quantity/unit.

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. SELECT ProductName, QuantityPerUnit FROM Products;
2. SELECT ProductName and QuantityPerUnit FROM Products;
3. SELECT ProductName ProductName, QuantityPerUnit QuantityPerUnit FROM Products;
4. **Both A and C**

Q72. Write a query to get current Product list (Product ID and name).

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. SELECT ProductID, ProductName FROM Products WHERE Discontinued = 0 ORDER BY ProductName;
2. SELECT ProductID, ProductName FROM Products WHERE Discontinued = False ORDER BY ProductName;
3. **Both A and B**
4. None of the above.

Q73. Write a query to get discontinued Product list (Product ID and name)

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. SELECT ProductID, ProductName FROM Products WHERE Discontinued = 1 ORDER BY ProductName;
2. SELECT ProductID, ProductName FROM Products WHERE Discontinued = TRUE ORDER BY ProductName;
3. SELECT ProductID, ProductName FROM Products WHERE Discontinued = FALSE + 1 ORDER BY ProductName;
4. **All of the above**

Q74. Write a query to get most expense and least expensive Product list (name and unit price).

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. SELECT ProductName, UnitPrice from Products where UnitPrice in (SELECT max (UnitPrice), min (UnitPrice) FROM Products);
2. SELECT ProductName, UnitPrice from Products where UnitPrice in any (SELECT max (UnitPrice), min (UnitPrice) FROM Products);
3. **SELECT ProductName, UnitPrice from Products where UnitPrice = (SELECT max (UnitPrice) FROM Products) or UnitPrice = (SELECT min (UnitPrice) FROM Products);**
4. None of the above.

Q75. Write a query to get Product list (id, name, unit price) where current products cost less than $20

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. SELECT ProductID, ProductName, UnitPrice FROM Products WHERE UnitPrice < 20 AND Discontinued = 0 ORDER BY UnitPrice DESC;
2. SELECT ProductID, ProductName, UnitPrice FROM Products WHERE UnitPrice < 20 AND Discontinued = False ORDER BY UnitPrice DESC;
3. SELECT ProductID, ProductName, UnitPrice FROM Products WHERE (((UnitPrice) < 20) AND ((Discontinued) = False)) ORDER BY UnitPrice DESC;
4. **All of the above**

Q76. Write a query to get Product list (name, unit price) where products cost between $15 and $25.

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. SELECT ProductName, UnitPrice FROM Products WHERE UnitPrice between 15 and 25 AND Products.Discontinued = False;
2. SELECT ProductName, UnitPrice FROM Products WHERE UnitPrice >= 15 and UnitPrice<=25 AND Products.Discontinued = False;
3. SELECT ProductName, UnitPrice FROM Products WHERE (((UnitPrice) >= 15 and (UnitPrice) <= 25) AND ((Products.Discontinued) = False));
4. **All of the above.**

Q77. Write a query to get Product list (id, name, unit price) of above average price.

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. **SELECT ProductName, UnitPrice FROM Products WHERE UnitPrice > (SELECT avg (UnitPrice) FROM Products) ORDER BY UnitPrice;**
2. SELECT ProductName, UnitPrice FROM Products WHERE UnitPrice > (SELECT average (UnitPrice) FROM Products) ORDER BY UnitPrice;
3. SELECT ProductName, UnitPrice FROM Products WHERE avg(UnitPrice) > (SELECT UnitPrice FROM Products) ORDER BY UnitPrice;
4. None of the above

Q78. Write a query to count current and discontinued products.

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. SELECT COUNT (DISTINCT ProductName) FROM Products;
2. **SELECT COUNT (ProductName) FROM Products GROUP BY Discontinued;**
3. SELECT COUNT (ProductName) FROM Products ORDER BY Discontinued;
4. None of the above.

Q79. Write a query to get Product list (name, units on order, and units in stock) of stock is less than the quantity on order.

**Sample table: products**

(ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)

**Note: Discontinued: 0 / 1 (0 is Continued and 1 is Discontinued)**

1. SELECT ProductName, UnitsOnOrder, UnitsInStock FROM Products WHERE Discontinued = False AND UnitsInStock < UnitsOnOrder;
2. SELECT ProductName, UnitsOnOrder, UnitsInStock FROM Products WHERE (((Discontinued) = False) AND ((UnitsInStock)<UnitsOnOrder));
3. **Both A and B**
4. None of the above.

Q80. Write a SQL statement to create a simple table countries including columns country\_id, country\_name and region\_id.

1. CREATE TABLE COUNTRIES (COUNTRY\_ID varchar , COUNTRY\_NAME varchar(40), REGION\_ID decimal(10,0));
2. **CREATE TABLE COUNTRIES (COUNTRY\_ID varchar(2), COUNTRY\_NAME varchar(40), REGION\_ID decimal(10,0));**
3. CREATE TABLE COUNTRIES TABLE (COUNTRY\_ID varchar(2), COUNTRY\_NAME varchar(40), REGION\_ID decimal(10,0));
4. Both A and B

Q81. Write a SQL statement to create a simple table countries including columns country\_id, country\_name and region\_id which is already exists.

1. **CREATE TABLE IF NOT EXISTS countries (COUNTRY\_ID varchar (2), COUNTRY\_NAME varchar (40), REGION\_ID decimal (10, 0));**
2. CREATE TABLE NOT IF EXISTS countries (COUNTRY\_ID varchar (2), COUNTRY\_NAME varchar (40), REGION\_ID decimal (10, 0));
3. CREATE TABLE EXISTS IF NOT countries (COUNTRY\_ID varchar (2), COUNTRY\_NAME varchar (40), REGION\_ID decimal (10, 0));
4. All of the above

Q82. Write a SQL statement to create the structure of a table duplicate\_countries similar to countries.

1. CREATE TABLE IF NOT EXISTS COUNTRIES in DUPLICATE\_COUNTRIES;
2. **CREATE TABLE IF NOT EXISTS DUPLICATE\_COUNTRIES LIKE COUNTRIES;**
3. CREATE TABLE NOT IF EXISTS DUPLICATE\_COUNTRIES LIKE COUNTRIES;
4. CREATE TABLE IF NOT EXISTS COUNTRIES into DUPLICATE\_COUNTRIES;

Q83. Write a SQL statement to create a duplicate copy of COUNTRIES table including structure and data by name DUPLICATE\_COUNTRIES.

1. CREATE TABLE IF NOT EXISTS DUPLICATE\_COUNTRIES IS SELECT \* FROM COUNTRIES;
2. **CREATE TABLE IF NOT EXISTS DUPLICATE\_COUNTRIES AS SELECT \* FROM COUNTRIES;**
3. CREATE TABLE IF NOT EXISTS DUPLICATE\_COUNTRIES LIKE SELECT \* FROM COUNTRIES;
4. CREATE TABLE IF NOT EXISTS DUPLICATE\_COUNTRIES JUST LIKE SELECT \* FROM COUNTRIES;

Q84. Write a SQL statement to create a table countries set a constraint NULL.

1. **CREATE TABLE IF NOT EXISTS countries (COUNTRY\_ID varchar (2) NOT NULL, COUNTRY\_NAME varchar (40) NOT NULL, REGION\_ID decimal (10, 0) NOT NULL);**
2. CREATE TABLE IF NOT EXISTS countries (COUNTRY\_ID varchar (2) as NOT NULL, COUNTRY\_NAME varchar (40) as NOT NULL, REGION\_ID decimal (10, 0) as NOT NULL);
3. CREATE TABLE IF NOT EXISTS countries (COUNTRY\_ID varchar (2) with NOT NULL, COUNTRY\_NAME varchar (40) with NOT NULL, REGION\_ID decimal (10, 0) with NOT NULL);
4. Either A or B

Q85. Write a SQL statement to create a table named JOBS including columns job\_id, job\_title, min\_salary, max\_salary and check whether the max\_salary amount exceeding the upper limit 25000.

1. **CREATE TABLE IF NOT EXISTS JOBS (JOB\_ID varchar (10) NOT NULL, JOB\_TITLE varchar (35) NOT NULL, MIN\_SALARY decimal (6, 0), MAX\_SALARY decimal (6, 0) CHECK (MAX\_SALARY <= 25000));**
2. CREATE TABLE IF NOT EXISTS JOBS (JOB\_ID varchar (10) NOT NULL, JOB\_TITLE varchar (35) NOT NULL, MIN\_SALARY decimal (6, 0), MAX\_SALARY decimal (6, 0) MAX (MAX\_SALARY <= 25000));
3. CREATE TABLE IF NOT EXISTS JOBS (JOB\_ID varchar (10) NOT NULL, JOB\_TITLE varchar (35) NOT NULL, MIN\_SALARY decimal (6, 0), MAX\_SALARY decimal (6, 0) UPPER (MAX\_SALARY <= 25000));
4. CREATE TABLE IF NOT EXISTS JOBS (JOB\_ID varchar (10) NOT NULL, JOB\_TITLE varchar (35) NOT NULL, MIN\_SALARY decimal (6, 0), MAX\_SALARY decimal (6, 0) MAXIMUM (MAX\_SALARY <= 25000));

Q86. Write a query to display sum of two numbers 10 and 15.

1. SELECT 10 + 15;
2. SELECT SUM (10 + 15)
3. SELECT 10 + 15 FROM DUAL;
4. **All of the above.**

Q87. Write a SQL statement to create a table named Persons where ID column must be given a primary key.

1. CREATE TABLE Persons (ID int NOT NULL, LastName varchar (255) NOT NULL, FirstName varchar (255), Age int, PRIMARY KEY (ID));
2. CREATE TABLE Persons (ID int NOT NULL, LastName varchar (255) NOT NULL, FirstName varchar (255), Age int, CONSTRAINT PK\_Person PRIMARY KEY (ID));
3. CREATE TABLE Persons (ID int NOT NULL PRIMARY KEY, LastName varchar (255) NOT NULL, FirstName varchar (255), Age int);
4. **All of the above.**

Q88. Write a query to INSERT a record in Person table, structure as follows.

CREATE table Person (ID number unique, FIRSTNAME varchar (10), mobile int, primary key (ID, FIRSTNAME));

Insert statement are given in following sequence, in which line number error will raise.

1. INSERT into Person values (1001,'Rahul', 9850884221);

2. INSERT into Person values (1001,'Rahul', 9850884222);

3. INSERT into Person values (1002,'Rahul', 9850884223);

1. Line 1
2. **Line 2**
3. Line 3
4. All insert statement will work properly.

Q89. To remove duplicate rows from the result set of a SELECT use the following keyword:

1. NO DUPLICATE
2. UNIQUE
3. **DISTINCT**
4. None of the above.

Q90. Which of the following can add a row to a table?

1. Add
2. **Insert**
3. Update
4. Alter

Q91. In a LIKE clause, you can could ask for any value ending in "qpt" by writing

1. **LIKE %qpt**
2. LIKE ^qtp
3. LIKE $qtp
4. LIKE ^.\*ton$

Q92. The \_\_\_\_\_\_\_\_\_\_ statement tells MySQL to use the db\_name database as the default (current) database for subsequent statements.

1. **USE db\_name**
2. SET db\_name
3. GET db\_name
4. USER db\_name

Q93. How will the result of the following two statements differ?

SQL> SELECT MAX (longitude), MAX (latitude) from zip\_state\_city;

SQL> SELECT MAX (longitude), MAX (latitude) from zip\_state\_city GROUP BY state;

1. Statement 1 will fail because it is missing a GROUP BY clause.
2. Statement 2 will return one row, and statement 1 may return more than one row.
3. Statement 2 will display a longitude and latitude for each ZIP\_STATE\_CITY.
4. **Statement 1 will display two values and Statement 2 will display two values for each state.**

Q94. Which SQL statement will query the EMPLOYEES table for FIRST\_NAME, LAST\_NAME AND SALARY of all employees in DEPARTMENT\_ID 40 in the alphabetical order of last name?

1. Select frist\_name, last\_name salary from employees ORDER By last\_name where department\_id = 40;
2. Select first\_name, last\_name, salary from employees ORDER BY last\_name ASC where department\_id = 40;
3. Select first\_name last\_name salary from employees where department\_id = 40 ORDER BY last\_name ASC;
4. **Select first\_name, last\_name, salary from employees where department\_id = 40 ORDER BY last\_name;**

Q95. You want to create a report to show different jobs in each department. You do not want to display any duplicate rows in the report.

Which SELECT statement do you use to create the report?

1. SELECT deptno, job FROM EMP;
2. SELECT NODUPLICATE deptno, job FROM EMP;
3. **SELECT DISTINCT deptno, job FROM EMP;**
4. CREATE REPORT DISPLAY deptno, job FROM EMP;

Q96. The PRODUCT table contains these columns:

ID INT (9) PK

COST FLOAT (7, 2)

SALE\_PRICE FLOAT (7,2)

Management has asked you to calculate the net revenue per unit for each product if the cost of each product is increased by 10% and the sale price of each product is increased by 25%.

You issue this SQL statement:

**SELECT id, sale\_price \* 1.25 - cost \* 1.10 FROM product;**

Which conclusion can you draw from the results?

1. **Only the required results are displayed.**
2. The results provide more information than management requested.
3. A function needs to be included in the SELECT statement to achieve the desired results.
4. The order of the operations in the calculation needs to be changed to achieve the required results.

Q97. Evaluate this SQL statement:

SELECT e.id, (.15 \* e.salary) + (.25 \* e.bonus) + (s.sale\_amount \* (.15 \* e.commission\_pct)) FROM employee e, sale s WHERE e.id = s.emp\_id;

What would happen if you removed all the parentheses from the calculation?

1. The results will be lower.
2. The results will be higher.
3. The statement will not execute.
4. **The statement will achieve the same results.**

Q98. For which three tasks would you use the WHERE clause?

1. Compare two values
2. Display only unique data
3. Restrict the rows displayed
4. **Both A and C**

Q99. The EMPLOYEE table contains these columns:

LAST\_NAME VARCHAR (25)

FIRST\_NAME VARCHAR (25)

DEPT\_ID INT (9)

You need to display the names of employees that are not assigned to a department.

Evaluate this SQL statement:

SELECT last\_name, first\_name FROM employee WHERE dept\_id = NULL;

Which change should you make to achieve the desired result?

1. Create an outer join.
2. Change the column in the WHERE condition.
3. **Change the operator in the WHERE condition.**
4. Add a second condition to the WHERE condition.

Q100. You need to retrieve the employee names and salaries from your EMP table sorted by salary in descending order. If two names match for a salary, the names must be displayed in alphabetical order.

Which statement produces the required results?

1. SELECT ename, sal FROM EMP ORDER BY ename, sal;
2. SELECT ename, sal FROM EMP ORDER BY sal, ename;
3. SELECT ename, sal FROM EMP SORT BY sal DESC, ename;
4. **SELECT ename, sal FROM EMP ORDER BY sal DESC, ename;**

Q101. Find the SQL statement below that is equal to the following: SELECT NAME FROM CUSTOMER WHERE STATE = 'VA';

1. SELECT NAME IN CUSTOMER WHERE STATE IN ('VA');
2. SELECT NAME IN CUSTOMER WHERE STATE = 'VA';
3. SELECT NAME IN CUSTOMER WHERE STATE = 'V';
4. **SELECT NAME FROM CUSTOMER WHERE STATE IN ('VA');**

Q102. Which rows will this SELECT statement return?

SELECT \* FROM products WHERE ordernumber > = 1000

1. **All rows in the products table who’s order number is greater than or equal to 1000**
2. All rows in the products table who’s order number is less than or equal to 1000
3. The first 1000 rows of the product table
4. All rows in the products table who’s order number is greater than 1000

Q103. What will be the output of the following MySQL statement?

SELECT true = true;

1. 0
2. **1**
3. true
4. None of the above.

Q104. What will be the output of the following MySQL statement?

SELECT true = false;

1. **0**
2. 1
3. true
4. None of the above.

Q105. What will be the output of the following MySQL statement?

SELECT false = false;

1. 0
2. **1**
3. true
4. None of the above.

Q106. Write a query to display leading zeros before maximum and minimum salary.

1. **SELECT job\_id, LPAD(max\_salary, 7, '0') 'Max Salary', LPAD(min\_salary, 7, '0') 'Min Salary' FROM jobs;**
2. SELECT job\_id, LPAD(max\_salary, 7, '0') 'Max Salary', PAD(min\_salary, 7, '0') 'Min Salary' FROM jobs;
3. SELECT job\_id, PAD(max\_salary, 7, '0') 'Max Salary', PAD(min\_salary, 7, '0') 'Min Salary' FROM jobs;
4. SELECT job\_id, LEFTPAD(max\_salary, 7, '0') 'Max Salary', LEFTPAD(min\_salary, 7, '0') 'Min Salary' FROM jobs;

Q107. Write a query to print 'true' of the salary of the employee is 3000.

1. **SELECT sal, IF(sal==3000,'true','false') from emp;**
2. SELECT sal, IF(sal -eq 3000,'true','false') from emp;
3. SELECT sal, IF(sal=3000,'true','false') from emp;
4. SELECT sal, IF(sal equals 3000,'true','false') from emp;

Q108. Write a query to find second highest salary from EMP table.

1. **SELECT max(sal) from EMP where sal < (SELECT max(sal) from EMP);**
2. SELECT max(sal) from EMP where sal > (SELECT max(sal) from EMP);
3. SELECT max(sal) from EMP where sal in (SELECT max(sal) from EMP);
4. SELECT max(sal) from EMP where sal <> (SELECT max(sal) from EMP);

Q109. Write a query to print the 5th record from the EMP table.

1. **SELECT \* from EMP limit 1 offset 4;**
2. SELECT \* from EMP limit 5 offset 4;
3. SELECT \* from EMP limit 5 offset 1;
4. SELECT \* from EMP limit 5 offset 5;

Q110. Write a query to print the record from the EMP table between 5 to 9.

1. **SELECT \* from EMP limit 5 offset 4;**
2. SELECT \* from EMP limit 5 offset 9;
3. SELECT \* from EMP limit between 5 and 9;
4. None of the above.

Q111. Write a query to display all record from the EMP with serial number starting with 1.

1. SELECT @x:= @x+1, emp.\* from EMP;
2. SELECT @x:= @x+1, emp.\* from EMP, (SELECT @x:=0);
3. **SELECT @x:= @x+1, emp.\* from EMP, (SELECT @x:=0) e;**
4. SELECT x:= x + 1, emp.\* from EMP;

Q112. Write a query to display all record from the EMP with serial number starting with 10.

1. SELECT \* from (SELECT @x:= @x+1, emp.\* from EMP, (SELECT @x:=10) e) EMP;
2. **SELECT \* from (SELECT @x:= @x+1, emp.\* from EMP, (SELECT @x:=9) e) EMP;**
3. SELECT \* from (SELECT @x:= @x+10, emp.\* from EMP, (SELECT @x:=0) e) EMP;
4. Any of the above.

Q113. Write a query to randomly display all record from the EMP

1. SELECT \* from EMP order by rand;
2. **SELECT \* from EMP order by rand();**
3. SELECT \* from EMP order by random();
4. SELECT \* from EMP order by rand(1,100);

Q114. Write a statement returns the book name and in which language the book have been published. The status of language is English Book for pub\_lang English otherwise it returns 'Other Language'.

1. **SELECT book\_name, IF(pub\_lang='English',"English Book","Other Language") AS Language FROM book\_mast;**
2. SELECT book\_name, IF(pub\_lang=='English',"English Book","Other Language") AS Language FROM book\_mast;
3. SELECT book\_name, IF(pub\_lang IS 'English',"English Book","Other Language") AS Language FROM book\_mast;
4. SELECT book\_name, IF(pub\_lang IN 'English',"English Book","Other Language") AS Language FROM book\_mast;

Q115. Write a query to display N/A instead of NULL for comm column of EMP table.

1. **SELECT comm, IF(comm is null, 'N/A', comm) from EMP;**
2. SELECT comm, nvl(comm is null, 'N/A', comm) from EMP;
3. SELECT comm, ISNULL(comm is null, 'N/A', comm) from EMP;
4. SELECT comm, IFNVL(comm is null, 'N/A', comm) from EMP;

Q116. Write a query to count how many employees are not getting commission. The employee table has 17 records of which only 6 employees are getting commission.

Count Output: 11

1. SELECT count(comm) from EMP where comm is null;
2. SELECT count(comm) from EMP where comm = null;
3. SELECT count(comm) from EMP group by comm having comm is null;
4. **None of the above**

Q117. Write a query to display all records from EMP table who have been hired before 'SMITH'.

1. **SELECT \* from EMP where hiredate < (SELECT hiredate from EMP where ename='smith');**
2. SELECT \* from EMP where hiredate < hiredate and ename='smith';
3. SELECT \* from EMP where hiredate is (SELECT hiredate from EMP where ename='smith')
4. Any of the above.

Q118. Write a query to count how many employees are getting commission.

1. select count(if (comm <> NULL, 0, null)) from emp;
2. **select count(if (comm is not null, 0, null)) from emp;**
3. select count( if(comm is not null, 0, null) END if) from emp;
4. select count(if (comm is not null, 0, null) ENFIF) from emp;

Q119. Write a query to count how many employees are not getting commission.

1. **select count(if(comm is null, 1, null)) from emp;**
2. select count(if (comm = null, 0, null)) from emp;
3. select count( if(comm is not null, 0, null) END if) from emp;
4. select count(if (comm is not null, 0, null) ENFIF) from emp;

Q120. Write a query to display the employees who have been hired 180 day after smith.

1. **select \* from emp where hiredate >(select DATE\_ADD(hiredate, INTERVAL 180 DAY) from emp where ename='smith');**
2. select \* from emp where hiredate >(select DATE\_ADDS(hiredate, INTERVAL 180 DAY) from emp where ename='smith');
3. select \* from emp where hiredate >(select ADD\_DAYS(hiredate, INTERVAL 180 DAY) from emp where ename='smith');
4. Any of the above.

Q121. Write a query to list the emps who’s names having a character set 'll' together.

1. **select \* from emp where ename like '%LL%';**
2. select \* from emp where ename like '%L%L%';
3. select \* from emp where ename like 'LL%';
4. select \* from emp where ename like '%L\_L%';

Q122. Write a query to list the emps those are having five characters in their names.

1. select \* from emp where ename like '\_\_\_\_\_' ;
2. select \* from emp where length(ename) = 5;
3. select \* from emp where ename like '5%' ;
4. **Both A and B**

Q123. Write a query to list the emps who does not belong to Deptno 20.

1. select \* from emp where deptno not in (20);
2. select \* from emp where deptno != 20;
3. select \* from emp where deptno <>20;
4. **Any of the above**

Q124. Write a query to list all emps who are hired in the year 1980.

1. **select \* from emp where date\_format(hiredate,'%Y') = 1980;**
2. select \* from emp where date\_format(hiredate,'YEAR') = 1980;
3. select \* from emp where date\_format(hiredate,'year') = 1980;
4. None of the above

Q125. Write a query to list all the emps who joined before or after 1980.

1. select \* from emp where date\_format(hiredate,'%Y') <> 1980;
2. select \* from emp where date\_format(hiredate,'%Y') not in 1980;
3. select \* from emp where date\_format(hiredate,'%Y') not in (1980);
4. **Both A and C**

Q126. Write a query to list all emps who are hired in the year 1980.

1. select \* from emp where extract(year from hiredate) =1980;
2. select \* from emp where date\_format(hiredate,'%Y') = 1980;
3. select \* from emp where date\_format(hiredate,'%y') = 1980;
4. **Either A or B**

Q127. Write a query to list all emps who are hired in the 4th quarter of a year.

1. **select \* from emp where extract(QUARTER from hiredate) = 4;**
2. select \* from emp where extract(4th QUARTER from hiredate) = 4;
3. select \* from emp where extract(QUR from hiredate) = 4;
4. select \* from emp where extract(hiredate from 4th quarter) = 4;

Q128. List the emps who joined in any year but belongs to the month of DECEMBER.

1. **select \* from emp where monthname(hiredate)='december';**
2. select \* from emp where extract(month from hiredate) = 'december';
3. select \* from emp where extract(month of hiredate) = 'december';
4. select \* from emp where extract(month of 'DECAMBER') = 'december';

Q129. Write a query to display all employee details who are getting highest salary job wise.

1. select \* from emp where sal = (select max(Sal) from emp group by job);
2. **select \* from emp where sal in (select max(Sal) from emp group by job);**
3. select \* from emp where sal as (select max(Sal) from emp group by job);
4. None of the above.

Q130. Write a query to create a table having virtual column.

1. **CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always as (c1+c2));**
2. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated as (c1+c2));
3. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always like (c1+c2));
4. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always is (c1+c2));

Q131. Write a query to create a table having virtual column.

1. **CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always as (c1+c2) VIRTUAL);**
2. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated as (c1+c2));
3. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always like (c1+c2));
4. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always is (c1+c2));

Q132. Write a query to create a table having virtual column.

1. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always as (c1+c2) VIRTUAL);
2. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always as (c1+c2));
3. CREATE TABLE TEMP (c1 int, c2 int, c3 int generated always like (c1+c2));
4. **Both A and B**

Q133. Write a query that uses the EXISTS operator to extract all salespeople who have customers with a rating of 200.

Salespeople (Snum PK, Sname, City, Comm)

Customers (Cnum PK, Cname, City, Rating, Snum FK)

ORDERS (Onum PK, Amt, Odate, Cnum FK, Snum FK)

1. **SELECT S.\* from SALESPEOPLE S where EXISTS (SELECT \* from CUSTOMERS C where C.snum = S.snum and rating > 200);**
2. SELECT S.\* from SALESPEOPLE S where S.SNUM EXISTS (SELECT \* from CUSTOMERS C where C.snum = S.snum and rating >200);
3. SELECT S.\* from SALESPEOPLE S where EXISTS SELECT \* from CUSTOMERS C where C.snum = S.snum and rating > 200;
4. SELECT S.\* from SALESPEOPLE S where EXISTS IN (SELECT \* from CUSTOMERS C where C.snum = S.snum and rating > 200);

Q134. Find all customers with CNUM, 1000 above the SNUM of Serres

Salespeople (Snum PK, Sname, City, Comm)

Customers (Cnum PK, Cname, City, Rating, Snum FK)

ORDERS (Onum PK, Amt, Odate, Cnum FK, Snum FK)

1. SELECT \* from CUSTOMERS C, SALESPEOPLE S where C.snum = S.snum and sname='Serres' AND C.cnum > S.snum+1000;
2. SELECT \* from CUSTOMERS C, SALESPEOPLE S where C.snum = S.snum and sname='Serres' AND C.cnum > (S.snum+1000);
3. SELECT \* from CUSTOMERS C, SALESPEOPLE S where C.snum = S.snum and sname='Serres' OR C.cnum > S.snum+1000;
4. **Both A and B.**

Q135. Find the largest order taken by each salesperson on each date, eliminating those MAX orders, which are less than $3000.00 in value.

Salespeople (Snum PK, Sname, City, Comm)

Customers (Cnum PK, Cname, City, Rating, Snum FK)

ORDERS (Onum PK, Amt, Odate, Cnum FK, Snum FK)

1. SELECT odate, snum, max(amt) from ORDERS group by odate, snum having amt > 3000;
2. **SELECT odate, snum, max(amt) from ORDERS group by odate, snum having max(amt) > 3000;**
3. SELECT odate, snum, max(amt) from where max(amt) > 3000 ORDERS group by odate, snum;
4. SELECT odate, snum, amt from ORDERS group by odate, snum having max(amt) > 3000;

Q136. Find all customers located in cities where Serres (SNUM 1002) has customers.

Salespeople (Snum PK, Sname, City, Comm)

Customers (Cnum PK, Cname, City, Rating, Snum FK)

ORDERS (Onum PK, Amt, Odate, Cnum FK, Snum FK)

1. **SELECT C.\* from CUSTOMERS C, SALESPEOPLE S where C.city = S.city and sname='Serres';**
2. SELECT \* from CUSTOMERS C, SALESPEOPLE S where C.city == S.city and sname='Serres';
3. SELECT \* from CUSTOMERS C, SALESPEOPLE S where C.city = S.city and sname='Serres';
4. None of the above.

Q137. Select all customers with a rating above 200.00.

Salespeople (Snum PK, Sname, City, Comm)

Customers (Cnum PK, Cname, City, Rating, Snum FK)

ORDERS (Onum PK, Amt, Odate, Cnum FK, Snum FK)

1. **SELECT C.\* from CUSTOMERS C where rating >200;**
2. SELECT C.\* from CUSTOMERS C where rating greater than 200;
3. SELECT C.\* from CUSTOMERS C where rating more than 200;
4. SELECT C.\* from CUSTOMERS C where rating 200;

Q138. Count the number of salespeople currently listing orders in the Orders table.

Salespeople (Snum PK, Sname, City, Comm)

Customers (Cnum PK, Cname, City, Rating, Snum FK)

ORDERS (Onum PK, Amt, Odate, Cnum FK, Snum FK)

1. SELECT S.snum, count(\*) from ORDERS O, SALESPEOPLE S where O.snum = S.snum group by S.snum order by 2;
2. SELECT S.snum, count(\*) from ORDERS O, SALESPEOPLE S where O.snum = S.snum group by S.snum order by count(\*);
3. SELECT S.snum, count(\*) R1 from ORDERS O, SALESPEOPLE S where O.snum = S.snum group by S.snum order by R1;
4. **Any one of the above.**

Q139. Write a query to display all employee details from EMP table.

1. SELECT \* FROM EMP;
2. SELECT ALL \* FROM EMP;
3. SELECT \* ALL FROM EMP;
4. **Both A and B**

Q140. Write a query that produces all customers serviced by salespeople with a commission above 12%. Output the customer’s name and the salesperson‘s rate of commission.

Salespeople (Snum PK, Sname, City, Comm)

Customers (Cnum PK, Cname, City, Rating, Snum FK)

ORDERS (Onum PK, Amt, Odate, Cnum FK, Snum FK)

1. SELECT cname, comm from CUSTOMERS C, SALESPEOPLE S where C.snum = S.snum and comm\*100 > 12;
2. SELECT cname, comm from CUSTOMERS C, SALESPEOPLE S where C.snum = S.snum and comm\*100 > 12%;
3. SELECT cname, comm from CUSTOMERS C, SALESPEOPLE S where C.snum = S.snum and comm > .12;
4. **Both A and C**

Q141. Find salespeople who have multiple customers.

Salespeople (Snum PK, Sname, City, Comm)

Customers (Cnum PK, Cname, City, Rating, Snum FK)

ORDERS (Onum PK, Amt, Odate, Cnum FK, Snum FK)

1. SELECT sname, count (\*) from SALESPEOPLE S, CUSTOMERS C where S.snum = C.snum group by sname having count (\*) >1;
2. SELECT sname, count(\*) r1 from SALESPEOPLE S, CUSTOMERS C where S.snum = C.snum group by sname having r1 > 1;
3. **Both A and B**
4. None of the above.

Q142. Write a query to display ename, and dname of all employees

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. **select ename, dname from emp, dept where emp.deptno=dept.deptno and locationid=1;**
2. select ename, dname from emp, dept where emp.deptno=dept.deptno and locationid in 1;
3. select ename, dname from emp, dept where emp.deptno=dept.deptno and locationid is 1;
4. select ename, dname from emp, dept where emp.deptno=dept.deptno and locationid like IN 1;

Q143. Write a query to display how many employees are working in 'OPERATIONS RESEARCH' department?

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. SELECT count(\*) from EMP E, DEPT D where e.deptno=d.deptno and dname='OPERATIONS RESEARCH';
2. SELECT count(e.\*) FROM EMP E, DEPT D where e.deptno=d.deptno and dname='OPERATIONS RESEARCH';
3. SELECT count(ALL \*) from EMP E, DEPT D where e.deptno=d.deptno and dname='OPERATIONS RESEARCH';
4. **Both A and C**

Q144. Write a query to display the maximum employees (COUNT) working in any department.

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. **SELECT max(r1) from (SELECT deptno, count(\*) r1 from EMP group by deptno) E;**
2. SELECT max(r1) from (SELECT deptno, count(\*) r1 from EMP group by deptno);
3. SELECT deptno, max(r1) from (SELECT deptno, count(\*) r1 from EMP group by deptno) E;
4. None of the above.

Q145. Write a query to display the maximum employees (COUNT), and his deptno working in any department.

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. SELECT deptno, count(\*) from EMP group by deptno having count(\*) = (SELECT max(r1) from (SELECT deptno, count(\*) r1 from EMP group by deptno) e);
2. SELECT deptno, count(\*) from EMP group by deptno having count(\*) = (SELECT max(r1) from (SELECT deptno, count(\*) from EMP group by deptno) e);
3. SELECT deptno, count(\*) from EMP group by deptno having count(\*) IN (SELECT max(r1) from (SELECT deptno, count(\*) r1 from EMP group by deptno) e);
4. **Both A and C**

Q146. Write a query to display all employee name and corresponding department name.

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. **SELECT ename, dname from EMP e inner join DEPT d using(deptno);**
2. SELECT ename, dname from EMP e inner join DEPT d using(e.deptno);
3. SELECT ename, dname from EMP e inner join DEPT d using deptno=deptno;
4. SELECT ename, dname from EMP e inner join DEPT d using(d.deptno);

Q147. Write a query to display the details of latest hired employee in the organization.

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. SELECT \* from EMP where hiredate = (SELECT max(hiredate) from EMP);
2. SELECT \* from EMP where hiredate <=> (SELECT max(hiredate) from EMP);
3. SELECT \* from EMP where hiredate IN (SELECT max(hiredate) from EMP);
4. **Any one of the above.**

Q148. Write a query to display the details of earliest hired employee in the organization.

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. SELECT \* from EMP where hiredate = (SELECT min(hiredate) from EMP);
2. SELECT \* from EMP where hiredate <=> (SELECT min(hiredate) from EMP);
3. SELECT \* from EMP where hiredate IN (SELECT min(hiredate) from EMP);
4. **Any one of the above.**

Q149. Write a query to display the details of all employees where have been hired between the month of '1st JANUARY' and '31st MARCH'.

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. **SELECT \* from EMP where date\_format(hiredate, '%m') between 1 and 3;**
2. SELECT \* from EMP where date\_format(hiredate,'%m') in (1, 3);
3. Both A and B
4. None of the above.

Q150. Write a query to display how many employees joined the organization between the month of '1st JANUARY' and '31st MARCH'.

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. SELECT count(\*) from EMP where date\_format(hiredate,'%m') between 1 and 3;
2. SELECT count(ALL \*) from EMP where date\_format(hiredate,'%m') between 1 and 3;
3. SELECT count(\*) from EMP where date\_format(hiredate,'%m') between (1 and 3);
4. **Both A and B**

Q151. Write a query to display employee name with his manager’s name.

EMP (EMPNO PK, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO FK, HOBBYID, BONUSID)

DEPT (DEPTNO PK, DNAME, LOCATIONID FK)

LOCATION (LOCATIONID PK, LOCATIONNAME)

1. SELECT e.ename 'Employee Name', m.ename as 'Managers Name' from EMP e, EMP m where m.empno = e.mgr;
2. SELECT e.ename 'Employee Name', m.ename 'Managers Name' from EMP e inner join EMP m on m.empno = e.mgr;
3. SELECT e.ename as 'Employee Name', m.ename as 'Managers Name' from EMP e join EMP m on m.empno = e.mgr;
4. **Any one of the above**

Q152. Consider the following schema − STUDENTS (student\_code, first\_name, last\_name, email, phone\_no, date\_of\_birth, honours\_subject, percentage\_of\_marks)

Which of the following query would display names and percentage of marks of all students sorted by honours subject, and then order by percentage of marks?

1. **select first\_name, last name, honours\_subject, percentage\_of\_marks from students order by honours\_subject, percentage\_of\_marks;**
2. select first\_name, last name, honours\_subject, percentage\_of\_marks order by percentage\_of\_marks desc from students;
3. select first\_name, last name, percentage\_of\_marks from students order by percentage\_of\_marks desc;
4. select first\_name, last name, percentage\_of\_marks from students order by percentage\_of\_marks, honours\_subject;

Q153. Consider the following schema −

STUDENTS (student\_code, first\_name, last\_name, email, phone\_no, date\_of\_birth, honours\_subject, percentage\_of\_marks);

Which of the following query would display all the students who’s first name starts with the character 'A'?

1. **SELECT first\_name from STUDENTS where first\_name like 'A%';**
2. SELECT first\_name from STUDENTS where first\_name like '%A';
3. SELECT first\_name from STUDENTS where first\_name like '%A%';
4. SELECT first\_name from STUDENTS where first\_name like 'A';

Q154. Write a query to sort data only on specific data (that is. only on 'MANAGER' or 'SALESMAN')

1. **SELECT \* from EMP order by field (job, 'manager', 'salesman')**
2. SELECT \* from EMP order by field ('manager', 'salesman')
3. SELECT \* from EMP order by job = 'manager' or job = 'salesman'
4. None of the above.

Q155. Write the SQL statement to display all employees who’s comm is null.

1. SELECT \* from EMP where comm = NULL;
2. **SELECT \* from EMP where comm <=> NULL;**
3. SELECT \* from EMP where comm == NULL;
4. All of the above.

Q156. Write the query to display all employee details along his department details.

Note: All the common columns should display only once.

1. **SELECT \* from EMP natural join DEPT;**
2. SELECT \* from EMP inner join DEPT;
3. SELECT \* from EMP join DEPT;
4. Both B and C

Q157. Note: Nulls by default occur at the top

Write a query to display all employees in assending of commission but all the null commission must be arranged at the end.

1. SELECT \* from EMP order by comm ;
2. SELECT \* from EMP order by Null(comm),comm ;
3. **SELECT \* from EMP order by ISNULL(comm),comm ;**
4. All of the above.

Q158. You want to retrieve all employees, whether or not they have matching departments in the departments table. Which query would you use?

EMPLOYEES (LAST\_NAME, DEPARTMENT\_ID, SALARY)

DEPARTMENT (DEPARTMENT\_ID, DEPARTMENT\_NAME)

1. **SELECT last\_name, department\_name FROM employees e LEFT OUTER JOIN departments d ON (e.department\_id = d.department\_id);**
2. SELECT last\_name, department\_name FROM employees e RIGHT OUTER JOIN departments d ON (e.department\_id = d.department\_id);
3. SELECT last\_name, department\_name FROM employees e FULL OUTER JOIN departments d ON (e.department\_id = d.department\_id);
4. None of the above.

Q159. The following view statements contain the name, job title and the annual salary of employees working in the department 20:

1. CREATE view DEPT20 is SELECT ENAME, JOB, SAL\*12 "ANNUAL SALARY" from EMP where DEPTNO = 20;
2. CREATE view DEPT20 as SELECT ENAME, JOB, SAL\*12 ANNUAL SALARY from EMP where DEPTNO = 20;
3. **CREATE view DEPT20 as SELECT ENAME, JOB, SAL\*12 "ANNUAL SALARY" from EMP where DEPTNO = 20;**
4. CREATE view DEPT20 as SELECT ENAME, JOB, SAL\*12 "ANNUAL SALARY" from EMP where DEPTNO <> 20;

Q160. Which SQL statement will display the 3rd highest salary?

1. **SELECT sal from EMP group by sal desc limit 2,1;**
2. SELECT sal from EMP group by sal desc limit 3,1;
3. SELECT sal from EMP group by sal desc limit 3,-1;
4. Both A and C

Q161. List the total information of EMP table along with DNAME and Loc of all the emps Working Under 'ACCOUNTING' & 'RESEARCH' in the asc Deptno.

1. SELECT \* from EMP e, DEPT d where dname = 'ACCOUNTING' or dname ='RESEARCH' and e.deptno = d.deptno order by e.deptno asc;
2. SELECT \* from EMP e, DEPT d where (dname = 'ACCOUNTING' or dname ='RESEARCH') and e.deptno = d.deptno order by e.deptno asc;
3. SELECT \* from EMP e, DEPT d where d.dname in ('ACCOUNTING','RESEARCH') and e.deptno = d.deptno order by e.deptno asc;
4. **Both B and C**

Q162. List all the Grade2 and Grade 3 emps.

EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO, BONUSID, USER NAME, PWD)

SALGRADE (GRADE, LOSAL, HISAL)

1. SELECT \* from EMP e where e.empno in (SELECT e.empno from EMP e, SALGRADE s where e.sal between s.losal and s.hisal and s.grade in(2, 3));
2. SELECT \* from EMP e, SALGRADE s where e.sal between s.losal and s.hisal and s.grade in (2, 3);
3. **Both A and B**
4. None of the above.

Q163. List the Emps who are senior to their own MGRS.

1. SELECT w.hiredate, m.hiredate from EMP w, EMP m where w.mgr = m.empno;
2. SELECT w.hiredate, m.hiredate from EMP w, EMP m where w.mgr = m.empno and w.hiredate < m.hiredate;
3. SELECT w.hiredate, m.hiredate from EMP w, EMP m where w.empno= m.mgr and w.hiredate > m.hiredate;
4. **Both B and C**

Q164. List the Emps who’s Sal is more than any of the salaries of the SALESMAN.

1. **SELECT \* from EMP where sal >any (select sal+ifnull(comm,0) from emp where job='salesman');**
2. SELECT \* from EMP where sal > (SELECT sal+ifnull(comm,0) from EMP where job='salesman');
3. select \* from emp where sal in (SELECT sal+ifnull(comm,0) from EMP where job='salesman');
4. SELECT \* from EMP where sal like (SELECT sal+ifnull(comm,0) from EMP where job='salesman');

Q165. List the emps who are senior to BLAKE working at CHICAGO & BOSTON.

1. select \* from emp inner join dept using(deptno) where hiredate > (select hiredate from emp where ename='blake') and loc in('CHICAGO','BOSTON');
2. select \* from emp e ,dept d where d.loc in ('CHICAGO','BOSTON') and e.deptno = d.deptno and e.hiredate <(select e.hiredate from emp e where e.ename = 'BLAKE') ;
3. select \* from emp inner join dept where emp.deptno=dept.deptno and hiredate > (select hiredate from emp where ename='blake') and loc in ('CHICAGO', 'BOSTON');
4. **All of the above.**

Q166. Find out all employees who are earning the highest and the lowest.

1. SELECT \* from EMP where sal = ((SELECT max(sal) from EMP),(SELECT min(Sal) from EMP));
2. **SELECT \* from EMP where sal IN ((SELECT max(sal) from EMP), (SELECT min(Sal) from EMP));**
3. SELECT \* from EMP where sal IN (SELECT max(sal) from EMP, SELECT min(Sal) from EMP);
4. None of the above.

Q167. List the employee name who are the MANAGER of some employee.

1. SELECT distinct m.ename from EMP e, EMP m where e.mgr = m.empno;
2. SELECT distinctrow m.ename from EMP e, EMP m where e.mgr = m.empno;
3. SELECT \* from EMP m where exists (SELECT \* from EMP e where m.empno = e.mgr);
4. **All of the above.**

Q168. List the employees who are senior to most recently hired employee working under king.

1. SELECT \* from EMP where hiredate = (SELECT max(e.hiredate) from EMP e, EMP m where e.mgr=m.empno and m.ename='king');
2. SELECT \* from EMP where hiredate in (SELECT max(e.hiredate) from EMP e, EMP m where e.mgr=m.empno and m.ename='king');
3. SELECT \* from EMP where hiredate <=> (SELECT max(e.hiredate) from EMP e, EMP m where e.mgr=m.empno and m.ename='king');
4. **All of the above.**

Q169. Find out how many employees are working in every department.

1. SELECT dname, count(\*) from EMP JOIN DEPT using (deptno) group by dname;
2. SELECT dname, count(\*) from EMP INNER JOIN DEPT using (deptno) group by dname;
3. SELECT dname, count(\*) from EMP e, DEPT d where e.deptno = d.deptno group by dname;
4. **All of the above.**

Q170. Consider the following sequence of SQL structure issued.

CREATE TABLE TEMP (c1 char(10));

INSERT INTO TEMP VALUES ('1'), ('2'), ('3');

SELECT SUM(C1) FROM TEMP;

What will be the output?

1. 0
2. **6**
3. Error
4. Null

Q171. List the manage no and the number of employees working for those mgrs in the ascending MGR No.

1. SELECT w.mgr, count(\*) from EMP w, EMP m where w.mgr = m.empno group by w.mgr order by w.mgr;
2. SELECT w.mgr, count(\*) from EMP w, EMP m where w.mgr = m.empno group by w.mgr order by w.mgr asc;
3. SELECT w.mgr, count(\*) from EMP w, EMP m where w.mgr = m.empno group by w.mgr order by w.mgr asci;
4. **Both A and B**

Q172. List the department details where at least two emps are working

1. SELECT deptno, count(\*) from EMP group by deptno having count(\*) >= 2;
2. SELECT deptno, count(\*) from EMP e group by deptno having count(e.deptno) >= 2;
3. SELECT deptno, count(\*) from EMP group by deptno having count(emp.\*) >= 2;
4. **Both A and B**

Q173. Display the Grade, Number of emps, and max sal of each grade.

EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO, BONUSID, USER NAME, PWD)

SALGRADE (GRADE, LOSAL, HISAL)

1. SELECT s.grade, count(\*), max(sal) from EMP e, SALGRADE s where e.sal > s.losal and e.sal < s.hisal;
2. SELECT s.grade, count(\*), max(sal) from EMP e, SALGRADE s where e.sal between s.losal and s.hisal group by s.grade;
3. SELECT s.grade, count(\*), max(sal) from EMP e, SALGRADE s where e.sal > s.losal and e.sal < s.hisal group by s.grade;
4. **Both B and C**

Q174. What is the alternative of the following SQL statement?

SELECT MAX(SAL) FROM EMP;

1. SELECT sal from EMP group by sal order by sal desc limit 1;
2. SELECT sal from EMP group by sal order by sal desc limit 0,1;
3. SELECT sal from EMP where sal = (SELECT max(Sal) from EMP);
4. **All of the above.**

Q175. Write a query to get maximum salary and store into a temporary variable.

1. SELECT @max\_sal:=MAX(sal) FROM EMP;
2. SELECT max(sal) into @max\_sal from EMP;
3. **Both A and B**
4. None of the above.

Q176. What will be the output of the following SQL statements?

SELECT DEPTNO INTO D FROM EMP WHERE ENAME='KING';

SELECT \* FROM EMP WHERE DEPTNO=@D;

1. Invalid set of statements.
2. **Will display all employees who are working in the same department of KING.**
3. Will display all employees who are not working in the same department of KING.
4. Will display all employees who are not working in the same department '@d'.

Q177. Write a query to display all employees whose job type is either 'MANAGER' or 'SALESMAN'.

1. SELECT \* from EMP where job='MANAGER' OR job='SALESMAN';
2. SELECT \* from EMP where job='MANAGER' AND job='SALESMAN';
3. SELECT \* from EMP where job='MANAGER' union SELECT \* from EMP where job='SALESMAN';
4. **Both A and C**

Q178. The following sequence of SQL statement were issued.

CREATE TABLE TEMP (C1 INT PRIMARY KEY AUTO\_INCREMENT, C2 INT);

INSERT INTO TEMP VALUE (0, 1);

INSERT INTO TEMP VALUE (NULL, 2);

INSERT INTO TEMP VALUE (0, 3);

What will be the output of the following SQL statements issued?

SELECT LAST\_INSERT\_ID();

1. 1
2. 2
3. **3**
4. NULL

Q179. What query you will issue to fine the department names and count of employees working in each department.

1. SELECT dname, count(\*) from EMP inner join DEPT using(deptno) where EXISTS (SELECT \* from DEPT where emp.deptno = dept.deptno);
2. **SELECT dname, count(\*) from EMP inner join DEPT using(deptno) where EXISTS (SELECT \* from DEPT where emp.deptno = dept.deptno) group by dname;**
3. SELECT dname, count(\*) from EMP inner join DEPT using(deptno) where EXISTS IN (SELECT \* from DEPT where emp.deptno = dept.deptno) group by dname;
4. Both B and C

Q180. What query you will issue to fine the department names where more than 5 employees are working.

1. **SELECT dname, count(\*) from EMP inner join DEPT using(deptno) where EXISTS (SELECT \* from DEPT where emp.deptno = dept.deptno) group by dname having count(\*) > 5;**
2. SELECT dname, count(\*) from EMP inner join DEPT using(deptno) where EXISTS (SELECT \* from DEPT where emp.deptno = dept.deptno) group by dname where count(\*) > 5;
3. SELECT dname, count(\*) from EMP inner join DEPT using(deptno) where EXISTS (SELECT \* from DEPT where emp.deptno = dept.deptno) and where count(\*) > 5 group by dname;
4. All of the above

Q181. What happens when we issue the following SQL statements?

CREATE TABLE TEMP(Col1 INT, Col2 ENUM('A', 'B', 'C') NOT NULL);

INSERT into TEMP(Col1) values(1);

What will the record stores in the TEMP table?

1. 1, NULL
2. 1, FALSE
3. **1, 'A'**
4. 1, 'A B C'

Q182. What happens when we issue the following SQL statements?

CREATE TABLE TEMP(Col1 INT, Col2 ENUM('A', 'B', 'C') );

INSERT into TEMP(Col1) values(1);

What will the record stores in the TEMP table?

1. **1, NULL**
2. 1, FALSE
3. 1, 'A'
4. 1, 'A B C'

Q183. What command you will issue to create table.

1. CREATE TABLE sizes (size ENUM('small', 'med','ium', 'large'));
2. CREATE TABLE sizes (size ENUM('small', "med","ium", 'large'));
3. CREATE TABLE sizes (size ENUM('small', ('med','ium'), 'large'));
4. **Both A and B**

Q184. What command you will issue to create table.

1. CREATE TABLE sizes (size ENUM('small', CONCAT('med','ium'), 'large'));
2. CREATE TABLE sizes (size ENUM('small', ('med','ium'), 'large'));
3. CREATE TABLE sizes (size ENUM('small', ['med','ium'], 'large'));
4. **None of the above**

Q185. List the highest paid salary working under KING.

1. SELECT max(e.sal) from EMP e, EMP m where m.empno = e.mgr and m.ename='king';
2. SELECT max(e.sal) from EMP e inner join EMP m on m.empno = e.mgr and m.ename='king';
3. SELECT sal from EMP where sal in (SELECT max(sal) from EMP where mgr in (SELECT empno from EMP where ename = 'KING'));
4. **All of the above.**

Q186. List the details of most recently hired employee of dept 30.

1. SELECT \* from EMP where hiredate = (SELECT max(hiredate) from EMP where deptno = 30);
2. SELECT \* from EMP where hiredate in (SELECT max(hiredate) from EMP where deptno = 30);
3. SELECT \* from EMP where hiredate like (SELECT max(hiredate) from EMP where deptno = 30);
4. **All of the above**

Q187. List the emps whose sal is same as 'FORD' or 'BLAKE' (excluding 'FORD' or 'BLAKE').

1. SELECT \* from EMP where sal in (SELECT sal from EMP e where e.ename in ('FORD','BLAKE'));
2. SELECT \* from EMP where sal in (SELECT sal from EMP e where e.ename in ('FORD','BLAKE') and emp.empno <> e.empno);
3. SELECT \* from EMP where sal in (SELECT sal from EMP e where e.ename in ('FORD','BLAKE')) and ename not in ('FORD','BLAKE');
4. **Both A and C**

Q188. Which is the not valid statement?

1. CREATE TABLE V$TEST (Col1 int, Col2 int);
2. CREATE TABLE $V\_TEST (Col1 int, Col2 int);
3. **CREATE TABLE V@TEST (Col1 int, Col2 int);**
4. All of the above.

Q189. Which of the following statement is proper command to create a view by changing the column name in the view.

1. CREATE view view1 (col1, col2) as SELECT empno, ename from EMP;
2. CREATE view view1 as SELECT empno col1, ename col2 from EMP;
3. CREATE view view1 as SELECT empno as col1, ename as col2 from EMP;
4. **All of the above.**

Q190. You issued the following set of commands.

CREATE view VIEW1 as SELECT \* from EMP;

RENAME table EMP to EMPLOYEE;

What happens after the SELECT command is given.

SELECT \* from VIEW1;

1. Will display all records from EMPLOYEE table.
2. **Error, the VIEW becomes invalid.**
3. TABLE cannot be renamed;
4. Invalid VIEW name.

Q191. The user wants to drop multiple columns and add a new column. Which of the following statement he will issue?

1. ALTER table TEMP drop column (COL1, COL2), add COL3 int;
2. ALTER table TEMP drop columns (COL1, COL2), add COL3 int;
3. ALTER table TEMP drop column (COL1, COL2), add column COL3 int;
4. **ALTER table TEMP drop column COL1, drop column COL2, add COL3 int;**

Q192. Display all the employees whose salary is more than the average salary of job 'MANAGER'.

**SELECT \* from EMP where sal > (SELECT avg(sal) from EMP where job='manager');**

SELECT \* from EMP where avg(sal) > (SELECT sal from EMP where job='manager');

SELECT \* from EMP where avg(sal) > (SELECT sal from EMP where job='manager') group by \*;

SELECT \* from EMP having avg(sal) > (SELECT sal from EMP where job='manager');

Q193. Write a query to find employee details whose salary is more than the minimum average salary of any deptno.

1. **SELECT \* from EMP where sal > (SELECT min(R1) from (SELECT avg(sal) R1 from EMP group by deptno) e);**
2. SELECT \* from EMP where sal > (SELECT min(avg (sal)) from (SELECT avg(sal) R1 from EMP group by deptno) e);
3. SELECT \* from EMP having sal > (SELECT min(avg(sal)) from (SELECT avg(sal) R1 from EMP group by deptno) e);
4. None of the above.

Q194. Write a query to find the highest and lowest salary of all employees.

1. SELECT max(sal), min(sal) from EMP;
2. **SELECT max(sal), min(sal) from EMP group by job;**
3. SELECT ((SELECT max(sal) from EMP group by job), (SELECT min(sal) from EMP group by job));
4. Both B and C

Q195. Write a query to ADD UNIQUE constraint.

1. ALTER table TEMP add unique (c1, c2);
2. ALTER table TEMP add constraint unk unique (c1, c2);
3. **Both A and B**
4. None of the above.

Q196. How will you rename the TABLE?

1. RENAME old\_tbl\_name to new\_tbl\_name
2. **RENAME TABLE old\_tbl\_name to new\_tbl\_name**
3. RENAME TABLE old\_tbl\_name as new\_tbl\_name
4. RENAME TABLE old\_tbl\_name like new\_tbl\_name

Q197. How will you rename the TABLE?

1. RENAME old\_tbl\_name to new\_tbl\_name
2. RENAME TABLE old\_tbl\_name to new\_tbl\_name
3. ALTER TABLE tbl\_name rename to new\_tbl\_name
4. **Both B and C**

Q198. Display all columns from EMP table whose column name starts with 'E'

1. show columns from emp where field = 'E%';
2. **show columns from emp where field like 'E%';**
3. show columns from emp where column like 'E%';
4. show columns from emp where columns like 'E%';

Q199. Display all table starts with 'E'

1. **show tables where tables\_in\_user01 like 'e%';**
2. show tables where table\_name like 'e%';
3. show tables where tables like 'e%';
4. None of the above

Q200. What is the syntax of calling stores procedure?

1. CALL routine\_name;
2. routine\_name [()];
3. CALL routine\_name([proc\_param, [proc\_param]])
4. **All of the above.**

Q201. Which 0f the following statement is used to connect to MySQL server?

1. **mysql --host=127.0.0.1 --port=3306 --user=root --password=ROOT db1**
2. mysql -host=127.0.0.1 -port=3306 -user-root -password-ROOT db1
3. mysql -h=127.0.0.1 -P=3306 -u=root -p=ROOT db1
4. All of the above

Q202. Which of the following statement is as valid DDL statement?

1. create table x(from char);
2. create table x('from' char(1));
3. create table x("from" char);
4. **None of the above.**

Q203. Relation **r{1, 2, -1, 3, -2, 1, 2, 1}**, What will be the output of the following statement?

SELECT SUM(c1) FROM r;

1. **7**
2. 10
3. -7
4. Null

Q204. How Would You Select All the Users, Whose Phone Number Is Null?

1. SELECT Users\_name, Phone\_Number FROM emp WHERE isnull(Phone\_Number);
2. SELECT Users\_name, Phone\_Number FROM emp WHERE Phone\_Number <=> null;
3. **Both A and B**
4. All of the above.

Q205. Examine the following statements.

Insert into dept(deptno, dname, loc) values(41, 'HRD', 'PUNE ');

Create table temp (id int, name varchar(10));

Rollback;

1. Only Create table statemen will be rolled back.
2. Only insert into statement will be rolled back;
3. Both table and record will be rolled back;
4. **Nothing will be rolled back.**

Q206. Examine the following statements.

Create TABLE temp (id int primary key, name varchar(10));

CREATE VIEW v1 as SELECT name FROM temp;

INSERT INTO v1 VALUES('saleel');

1. Record will be stored in a VIEW
2. Record will be stored in a TABLE
3. **Will raise an error**
4. None of the above.

Q207. What will be the output of the following SQL statement?

SELECT table\_name FROM information\_schema.tables WHERE table\_schema=db1;

1. It will display the list of databases.
2. It will display all table names from all schemas.
3. It will display all table names from db1 schema.
4. **Error in this SQL statement.**

Q208. What will be the output of the following SQL statement?

SELECT table\_name FROM information\_schema.tables WHERE table\_schema='db1';

1. It will display the list of databases.
2. It will display all table names from all schemas.
3. **It will display all table names from db1 schema.**
4. Invalid SQL statement.

Q209. What will be output of the following SQL statement?

SELECT CASE WHEN NULL = NULL THEN 'equals' ELSE 'not equals' END AS ANSWER;

1. equals
2. **not equals**
3. Error in the statement. WHEN must be written in lowercase.
4. Error in the statement. ELSE must be written in lowercase.

Q210. What will be output of the following SQL statement?

SELECT CASE WHEN NULL <=> NULL THEN 'equals' ELSE 'not equals' END AS ANSWER;

1. **equals**
2. not equals
3. Error in the statement. WHEN must be written in lowercase.
4. Error in the statement. ELSE must be written in lowercase.

Q211. What will be output of the following SQL statement?

SELECT CASE WHEN NULL IS NULL THEN 'equals' ELSE 'not equals' END AS ANSWER;

1. **equals**
2. not equals
3. Error in the statement. WHEN must be written in lowercase.
4. Error in the statement. ELSE must be written in lowercase.

Q212. What will be output of the following SQL statement?

SELECT CASE WHEN NULL = 0 THEN 'equals' ELSE 'not equals' END AS ANSWER;

1. equals
2. **not equals**
3. Error in the statement. WHEN must be written in lowercase.
4. Error in the statement. ELSE must be written in lowercase.

Q213. What will be output of the following SQL statement?

SELECT CASE WHEN NULL = true THEN 'equals' ELSE 'not equals' END AS ANSWER;

1. equals
2. **not equals**
3. Error in the statement. WHEN must be written in lowercase.
4. Error in the statement. ELSE must be written in lowercase.

Q214. What will be output of the following SQL statement?

SELECT CASE WHEN NULL IS UNKNOWN THEN 'equals' ELSE 'not equals' END AS ANSWER;

1. **equals**
2. not equals
3. Error in the statement. WHEN must be written in lowercase.
4. Error in the statement. ELSE must be written in lowercase.

Q215. Examine the following Sets

A = {"Jack", "Jill"},

B = {"New York", "Berlin", "Moscow"}

What will be the output of A U B?

1. **Jack, Jill, New York, Berlin, Moscow**
2. New York, Berlin, Moscow, Jack, Jill
3. Berlin, Moscow, Jack, Jill, New York
4. Moscow, Jack, Jill, New York, Berlin

Q216. Examine the following Sets

A = {"Jack", "Jill"},

B = {"New York", "Berlin", "Moscow"}

What will be the output of SELECT \* FROM A WHERE NOT EXISTS (SELECT \* FROM B WHERE A.c1 = B.c1);

1. **Jack, Jill**
2. New York, Berlin
3. Berlin, Jack
4. New York, Jill

Q217.

Q218.

Q219.

Q220.

Q221.

Q222.

Q223.

Q224.

Q225.

Q226.

Q227.

Q228.

Q229.

Q300.