INFORMATION PRACTICES

Class XII — CBSE / State Board

Chapter 1

Querying and SQL Functions

Question Bank — Complete Edition

30 Multiple Choice Questions | 20 Fill in the Blanks 20 Frequently Asked Questions | Short & Long Answer Questions Matching Questions | Teacher's Notes Throughout

Prepared with genuine teacher's insight to help students master SQL concepts with clarity, confidence, and exam readiness. Every question here has been crafted keeping the CBSE examination pattern and practical understanding in mind.

SECTION A: Multiple Choice Questions (MCQs)

Each question carries 1 mark. Choose the most appropriate answer.

■ *Teacher's Note: MCQs test quick recall and conceptual clarity. Read every option carefully — SQL function questions often have tricky distractors!*

Q1. Which of the following is a Single Row Function in SQL?

(a) COUNT()
(b) SUM()
(c) ROUND()
(d) AVG()

■ ***Answer: (c) ROUND()***

■ *Explanation: Single row functions operate on one row at a time. ROUND(), POWER(), MOD() are numeric single row functions.*

Q2. What will SELECT POWER(3,4) return?

(a) 12
(b) 64
(c) 81
(d) 34

■ ***Answer: (c) 81***

■ *Explanation: POWER(x,y) = x raised to y. So POWER(3,4) = 3×3×3×3 = 81.*

Q3. SELECT ROUND(456.7896, 2) will return:

(a) 456.78
(b) 456.79
(c) 456.80
(d) 456.00

■ ***Answer: (b) 456.79***

■ *Explanation: ROUND(number, decimal_places) rounds to specified decimal places. 456.7896 rounded to 2 decimal places = 456.79.*

Q4. SELECT MOD(17, 5) returns:

(a) 3
(b) 2
(c) 5
(d) 1

■ ***Answer: (b) 2***

■ *Explanation: MOD(m,n) returns the remainder when m is divided by n. 17 ÷ 5 = 3 remainder 2.*

Q5. Which function converts a string to UPPERCASE?

(a) LOWER()
(b) UCASE()
(c) MID()
(d) LTRIM()

■ ***Answer: (b) UCASE()***

■ *Explanation: UCASE() or UPPER() converts a string to uppercase letters.*

Q6. SELECT LENGTH('Informatics') returns:

(a) 10
(b) 11
(c) 12

(d) 9

■ ***Answer: (b) 11***

■ *Explanation: LENGTH() counts the total number of characters including spaces. 'Informatics' has 11 characters.*

Q7. SELECT LEFT('Computer Science', 4) gives:

(a) 'Comp'
(b) 'Scie'
(c) 'Compu'
(d) 'nce'

■ ***Answer: (a) 'Comp'***

■ *Explanation: LEFT(string, n) extracts the leftmost n characters. First 4 characters of 'Computer Science' = 'Comp'.*

Q8. SELECT RIGHT('SCIENCE', 3) returns:

(a) 'SCI'
(b) 'NCE'
(c) 'CIE'
(d) 'ENE'

■ ***Answer: (b) 'NCE'***

■ *Explanation: RIGHT(string, n) extracts the rightmost n characters. Last 3 of 'SCIENCE' = 'NCE'.*

Q9. SELECT MID('Informatics', 3, 4) returns:

(a) 'Info'
(b) 'form'
(c) 'orma'
(d) 'nfor'

■ ***Answer: (b) 'form'***

■ *Explanation: MID(string, pos, n) extracts n characters from position pos. From pos 3 in 'Informatics': f-o-r-m = 'form'.*

Q10. Which function returns the current system date and time?

(a) DATE()
(b) NOW()
(c) MONTH()
(d) YEAR()

■ ***Answer: (b) NOW()***

■ *Explanation: NOW() returns the current system date and time. DATE() extracts just the date part from a datetime expression.*

Q11. SELECT MONTH('2023-08-15') returns:

(a) 2023
(b) 15
(c) 8
(d) 08

■ ***Answer: (c) 8***

■ *Explanation: MONTH(date) returns the month number from a date. August = 8.*

Q12. SELECT MONTHNAME('2023-11-26') returns:

(a) 'November'
(b) '11'
(c) 'Nov'
(d) '26'

■ ***Answer: (a) 'November'***

■ *Explanation: MONTHNAME() returns the full name of the month, not an abbreviation or number.*

Q13. Which aggregate function ignores NULL values?

(a) COUNT(*)
(b) SUM(column)
(c) Both (a) and (b)
(d) Neither

■ ***Answer: (b) SUM(column)***

■ *Explanation: COUNT(*) counts all rows including NULLs. SUM(column), AVG(column), MIN(), MAX() all ignore NULL values.*

Q14. SELECT COUNT(*) FROM EMPLOYEE counts:

(a) Only non-NULL rows
(b) Rows with NULL designation
(c) All rows including NULLs
(d) Only distinct rows

■ ***Answer: (c) All rows including NULLs***

■ *Explanation: COUNT(*) counts every row in the table regardless of NULL values. COUNT(column_name) ignores NULLs.*

Q15. Which clause is used with GROUP BY to filter grouped records?

(a) WHERE
(b) ORDER BY
(c) HAVING
(d) DISTINCT

■ ***Answer: (c) HAVING***

■ *Explanation: HAVING clause filters groups formed by GROUP BY. WHERE filters individual rows before grouping.*

Q16. UNION operation between two tables requires:

(a) Same number of columns only
(b) Same data types in corresponding columns only
(c) Both same columns and compatible data types
(d) No conditions at all

■ ***Answer: (c) Both same columns and compatible data types***

■ *Explanation: For UNION, INTERSECT, MINUS — both tables must have same number of attributes and corresponding attributes must have same domain.*

Q17. INTERSECT operation returns:

(a) All rows from both tables
(b) Rows present in first table only
(c) Common rows present in both tables
(d) Rows present in second table only

■ ***Answer: (c) Common rows present in both tables***

■ *Explanation: INTERSECT returns only those tuples/rows which appear in BOTH the tables.*

Q18. MINUS operation (Table A MINUS Table B) returns:

(a) All rows from A and B
(b) Rows in A that are not in B
(c) Rows in B that are not in A
(d) Common rows of A and B

■ ***Answer: (b) Rows in A that are not in B***

■ *Explanation: MINUS returns rows from the first table which are NOT present in the second table. It is also called set difference.*

Q19. Cartesian product of two tables with 4 and 5 rows respectively gives:

(a) 9 rows
(b) 20 rows
(c) 1 row
(d) 45 rows

■ ***Answer: (b) 20 rows***

■ *Explanation: Cartesian product = m × n rows. 4 × 5 = 20 rows. The degree (columns) = sum of degrees of both tables.*

Q20. Which JOIN type eliminates the redundant common column?

(a) Simple JOIN
(b) Cartesian JOIN
(c) NATURAL JOIN
(d) CROSS JOIN

■ ***Answer: (c) NATURAL JOIN***

■ *Explanation: NATURAL JOIN automatically matches on common columns and does NOT repeat the common column in the result.*

Q21. SELECT INSTR('Informatics', 'ma') returns:

(a) 7
(b) 8
(c) 6
(d) 9

■ ***Answer: (b) 8***

■ *Explanation: INSTR(string, substring) returns the position of first occurrence. In 'Informatics', 'ma' starts at position 8.*

Q22. SELECT LTRIM(' Hello') returns:

(a) ' Hello'
(b) 'Hello'
(c) 'Hello '
(d) 'Hello'

■ ***Answer: (b) 'Hello'***

■ *Explanation: LTRIM removes leading (left-side) whitespace from a string.*

Q23. What is the output of SELECT ROUND(156.862, -1)?

(a) 156.9
(b) 160
(c) 150
(d) 157

■ ***Answer: (b) 160***

■ *Explanation: When decimal_places is negative, ROUND works on the left side of decimal point. -1 rounds to nearest 10. 156.862 → 160.*

Q24. Which of these is NOT a Date function in MySQL?

(a) DAY()
(b) DAYNAME()
(c) DATEDIFF()
(d) DATESTR()

■ ***Answer: (d) DATESTR()***

■ *Explanation: DATESTR() does not exist in MySQL. DAY(), DAYNAME(), DATEDIFF() are valid MySQL date functions.*

Q25. SELECT AVG(Price) FROM INVENTORY will:

(a) Return the largest price
(b) Return the total of all prices
(c) Return the average of all prices
(d) Return the number of prices

■ ***Answer: (c) Return the average of all prices***

■ *Explanation: AVG(column) is an aggregate function that returns the arithmetic mean of all non-NULL values in the column.*

Q26. When using GROUP BY, which of the following is correct?

(a) We cannot use ORDER BY with GROUP BY
(b) HAVING is used before GROUP BY
(c) GROUP BY groups rows with same values in specified column(s)
(d) GROUP BY can only be used with COUNT

■ ***Answer: (c) GROUP BY groups rows with same values in specified column(s)***

■ *Explanation: GROUP BY groups rows sharing the same values. ORDER BY can be used after GROUP BY, and HAVING comes after GROUP BY.*

Q27. The degree of Cartesian product of two relations R(3 attributes) and S(4 attributes) is:

(a) 12
(b) 3
(c) 4
(d) 7

■ ***Answer: (d) 7***

■ *Explanation: Degree of Cartesian Product = Sum of degrees = 3 + 4 = 7. Cardinality = product of cardinalities.*

Q28. SELECT TRIM(' SQL ') returns:

(a) ' SQL '
(b) 'SQL '
(c) ' SQL'
(d) 'SQL'

■ ***Answer: (d) 'SQL'***

■ *Explanation: TRIM() removes both leading and trailing whitespace from a string.*

Q29. Which SQL clause is used to display only unique/distinct grouped values?

(a) WHERE
(b) UNIQUE
(c) HAVING
(d) DISTINCT

■ ***Answer: (d) DISTINCT***

■ *Explanation: DISTINCT eliminates duplicate values from a result set. Example: SELECT DISTINCT Model FROM INVENTORY.*

Q30. In JOIN operation, a table alias is:

(a) A permanent rename of the table
(b) A short name valid only for that query
(c) Required mandatorily in every query
(d) Used only in subqueries

■ ***Answer: (b) A short name valid only for that query***

■ *Explanation: Table aliases (like D for DANCE, M for MUSIC) are temporary and valid only within the current query. They help resolve ambiguity.*

SECTION B: Fill in the Blanks

Each blank carries 1 mark. Write the most appropriate term/function.

■ *Teacher's Note: In fill-in-the-blanks, spelling of SQL functions matters. Write exactly as the function name appears — e.g., MONTHNAME, not Monthname.*

1. Functions that work on a single row and return one result per row are called _____ functions.

■ ***Answer: Single Row (Scalar) Functions***

2. Functions that work on a group of rows and return a single result are called _____ functions.

■ ***Answer: Aggregate (Multiple Row) Functions***

3. The function _____ returns the remainder when one number is divided by another.

■ ***Answer: MOD()***

4. SELECT _____ (Price, 2) rounds the Price column to 2 decimal places.

■ ***Answer: ROUND***

5. The string function _____ returns the number of characters in a string.

■ ***Answer: LENGTH()***

6. To extract the substring 'Inf' from 'Informatics' starting at position 1 with length 3, we use _____(string, 1, 3).

■ ***Answer: MID or SUBSTR***

7. The function _____ removes trailing whitespace from a string.

■ ***Answer: RTRIM()***

8. SELECT _____ ('hello world') will convert the string to 'HELLO WORLD'.

■ ***Answer: UCASE() or UPPER()***

9. The date function that returns the name of the day for a given date is _____.

■ ***Answer: DAYNAME()***

10. SELECT YEAR('2017-08-12') will return the value _____.

■ ***Answer: 2017***

11. The aggregate function that returns the largest value in a column is _____.

■ ***Answer: MAX()***

12. The aggregate function that calculates the total sum of a numeric column is _____.

■ ***Answer: SUM()***

13. The _____ clause in SQL is used to filter groups created by GROUP BY.

■ ***Answer: HAVING***

14. The operation that combines rows from two tables and shows common rows only once is called _____.

■ ***Answer: UNION***

15. The _____ operation returns only those rows which are present in both the tables.

■ ***Answer: INTERSECT***

16. The _____ operation returns rows from the first table that are NOT present in the second table.

■ ***Answer: MINUS (Set Difference)***

17. The Cartesian product of two relations with cardinalities 5 and 6 will have _____ rows.

■ ***Answer: 30***

18. In a JOIN operation, _____ JOIN automatically eliminates the repeated common attribute from the result.

■ ***Answer: NATURAL***

19. To display details of customers with a Yahoo email ID, we use: SELECT * FROM CUSTOMER WHERE Email LIKE _____.

■ ***Answer: '%yahoo%'***

20. The function _____ returns the position of a substring within a string.

■ ***Answer: INSTR()***

SECTION C: Frequently Asked Questions (FAQs)

These questions are commonly asked in board examinations. Answers are given in student-friendly language.

■ *Teacher's Note: FAQ answers should be written in your own words. Memorising definitions word-for-word is less helpful than understanding the concept and explaining it naturally.*

Q1. What is the difference between Single Row functions and Aggregate functions?

Single Row functions (also called Scalar functions) operate on one row at a time and return one result per row. For example, ROUND(Price, 2) gives a rounded value for each row individually.

Aggregate functions (also called Multiple Row functions) work on a set of rows and return a single result for the entire group. For example, SUM(Price) adds up all prices and gives one total.

The key difference: single row → many results; aggregate → one result.

Q2. Name and explain any three Numeric functions in SQL.

Three commonly used Numeric functions are:

(i) POWER(X, Y) — Returns X raised to the power Y. Example: POWER(2,5) = 32.

(ii) ROUND(N, D) — Rounds N to D decimal places. If D is 0, it rounds to the nearest whole number. Example: ROUND(4.567, 2) = 4.57.

(iii) MOD(M, N) — Returns the remainder when M is divided by N. Example: MOD(10, 3) = 1.

Q3. What is the purpose of GROUP BY clause? Give an example.

The GROUP BY clause is used to arrange identical data into groups. It groups rows that have the same values in the specified column(s) so that aggregate functions can be applied on each group separately.

Example: SELECT CustID, COUNT(*) AS 'Number of Cars' FROM SALE GROUP BY CustID;

This query groups all sales records by customer ID and counts how many cars each customer has purchased. Without GROUP BY, COUNT(*) would just count all rows together.

Q4. Differentiate between WHERE and HAVING clauses.

WHERE clause filters individual rows BEFORE grouping takes place. It cannot be used with aggregate functions.

HAVING clause filters groups AFTER the GROUP BY operation. It is specifically used with aggregate functions.

Example: SELECT EmpID, COUNT(*) FROM SALE GROUP BY EmpID HAVING COUNT(*) > 1;

Here, HAVING keeps only those employees who made more than one sale. WHERE could not achieve this.

Q5. What is UNION operation? What are its conditions?

UNION operation combines the rows of two tables and displays all unique rows from both, eliminating duplicates. It is represented by the symbol ∪.

Conditions for UNION (and other set operations): 1. Both tables must have the same number of attributes. 2. Corresponding attributes in both tables must have the same domain (data type).

Syntax: SELECT columns FROM Table1 UNION SELECT columns FROM Table2;

Q6. Explain INTERSECT operation with an example.

INTERSECT operation returns only those rows which are present in BOTH the tables. It shows the common tuples.

Example: If DANCE table has students {Aastha, Mohit, Mahira, Sanjay} and MUSIC table has {Mohak, Mahira, Lavanya, Sanjay, Abhay}, then: DANCE INTERSECT MUSIC = {Mahira, Sanjay}

These are students enrolled in BOTH events. INTERSECT is represented by the symbol ∩.

Q7. What is Cartesian Product? How is its degree and cardinality calculated?

Cartesian Product combines every tuple of the first relation with every tuple of the second relation. If Table A has 'm' rows and Table B has 'n' rows, the Cartesian product has m × n rows.

Degree of result = Sum of degrees of both tables (i.e., total number of columns combined). Cardinality of result = Product of cardinalities = m × n.

Example: DANCE (4 rows, 3 columns) × MUSIC (5 rows, 3 columns) = 20 rows, 6 columns.

Q8. What is NATURAL JOIN? How is it different from a simple JOIN?

NATURAL JOIN combines two tables based on their common attribute(s) and automatically eliminates the duplicate common column from the result.

Simple JOIN (with WHERE or ON clause) may keep both copies of the common column, requiring the use of table aliases to distinguish them.

Syntax: SELECT * FROM UNIFORM NATURAL JOIN COST;

The result of NATURAL JOIN is the same as JOIN with condition, but the common column appears only once, making the output cleaner.

Q9. Explain any four String functions with examples.

Four useful String functions:

(i) UCASE('hello') → 'HELLO' — Converts to uppercase.

(ii) MID('Informatics', 3, 4) → 'form' — Extracts 4 characters from position 3.

(iii) LENGTH('Computer') → 8 — Returns count of characters.

(iv) INSTR('Informatics', 'ma') → 8 — Returns position of 'ma' in the string.

Q10. What are Date and Time functions in SQL? Give three examples.

Date and Time functions are built-in SQL functions that perform operations on date or time data stored in a table.

(i) NOW() — Returns the current system date and time. Example: 2025-07-11 10:41:17

(ii) MONTHNAME(date) — Returns the full name of the month. MONTHNAME('2023-11-26') → 'November'

(iii) DAYNAME(date) — Returns the name of the day. DAYNAME('2023-03-24') → 'Friday'

These are especially useful for generating reports based on dates.

Q11. Write a query to display employee names whose salary is more than ■30,000 using the EMPLOYEE table.

Query: SELECT EmpName, Salary FROM EMPLOYEE WHERE Salary > 30000;

Explanation: The WHERE clause filters rows — only those employees whose Salary column value is greater than 30000 will appear in the result.

Q12. How is a table alias used in a JOIN query? Give an example.
A table alias gives a table a short, temporary name that is valid only within that particular query. It helps avoid ambiguity when two tables have columns with the same name.

Example: SELECT * FROM DANCE D, MUSIC M WHERE D.Name = M.Name;

Here, 'D' is the alias for DANCE and 'M' is the alias for MUSIC. D.Name refers to the Name column of DANCE table. The alias is only valid in this query — the table's original name is not changed.

Q13. What is the significance of COUNT(*) vs COUNT(column_name)?
COUNT(*) counts ALL rows in the table, including rows that have NULL values in any column.

COUNT(column_name) counts only those rows where the specified column is NOT NULL. It ignores NULL values.

Example: If MANAGER table has 4 rows but one row has NULL in MNAME column: SELECT COUNT(*) → returns 4 SELECT COUNT(MNAME) → returns 3

Q14. Explain MINUS (set difference) operation with an example.
MINUS operation returns all rows from the first table that are NOT present in the second table. It shows what is unique to the first table.

Example: DANCE MINUS MUSIC gives students who are in DANCE but NOT in MUSIC. DANCE = {Aastha, Mahit, Mahira, Sanjay}; MUSIC = {Mohak, Mahira, Lavanya, Sanjay, Abhay} DANCE MINUS MUSIC = {Aastha, Mohit, Lavanya (not in DANCE)...} = {Aastha, Mohit}

Note: MINUS is not supported in MySQL but is available in Oracle SQL. In MySQL, we use NOT IN or LEFT JOIN.

Q15. What do you understand by Aggregate functions? Name all five with their purpose.
Aggregate functions work on a set of rows and return a single value as the result. They are also called Multiple Row functions.

Five Aggregate functions: 1. MAX(col) — Returns the maximum (largest) value in the column. 2. MIN(col) — Returns the minimum (smallest) value. 3. AVG(col) — Returns the average (mean) value. 4. SUM(col) — Returns the sum of all values. 5. COUNT(col) — Returns the count of non-NULL values.

All these (except COUNT(*)) ignore NULL values.

Q16. How can you display the payment mode that has been used more than once?
We need to group by PaymentMode and then filter groups having count > 1:

Query: SELECT PaymentMode, COUNT(PaymentMode) FROM SALE GROUP BY PaymentMode HAVING COUNT(PaymentMode) > 1;

This groups all sales by payment mode, counts occurrences in each group, and then HAVING filters to show only those modes used more than once.

Q17. What is the output of the following: SELECT LCASE('INFORMATION PRACTICES')?
Output: 'information practices'

LCASE() (or LOWER()) converts all uppercase letters in a string to lowercase. It does not affect spaces, numbers, or special characters.

Q18. Explain the use of LIKE operator with % and _ wildcards in SQL.

The LIKE operator is used in the WHERE clause to search for a specified pattern in a column.

% (percent) wildcard matches zero or more characters of any kind. Example: WHERE Email LIKE '%yahoo%' — finds emails containing 'yahoo' anywhere.

_ (underscore) wildcard matches exactly ONE character. Example: WHERE Model LIKE 'L_I' — finds models like 'LXI', 'LZI', etc.

These are very useful for searching partial data in string columns.

Q19. Write a query to find the total number of different models available in the INVENTORY table.
Query: SELECT COUNT(DISTINCT Model) FROM INVENTORY;

Explanation: DISTINCT ensures each unique model is counted only once, even if it appears multiple times. COUNT() then counts those distinct values.

Q20. What is the role of ORDER BY in a GROUP BY query?
ORDER BY sorts the final result in ascending (ASC, default) or descending (DESC) order.

When used with GROUP BY, ORDER BY is placed AFTER the HAVING clause and sorts the grouped results.

Example: SELECT PaymentMode, COUNT(PaymentMode) FROM SALE GROUP BY PaymentMode ORDER BY PaymentMode;

This shows payment modes and their counts, sorted alphabetically by payment mode name.

SECTION D: Short Answer Type Questions (2–3 Marks)

■ *Teacher's Note: Short answer questions expect precision. Write to the point — no lengthy introductions needed. Use examples wherever possible.*

Q1. Write the SQL query to add a new column 'Discount' (numeric, 5 digits, 2 decimal places) to the INVENTORY table.

ALTER TABLE INVENTORY ADD Discount NUMERIC(5,2);

This uses the ALTER TABLE command with ADD clause to add a new column to an existing table.

Q2. Write a query to display the car name and GST (12% of Price) from the INVENTORY table, rounded to 1 decimal place.

SELECT CarName, ROUND((12/100)*Price, 1) 'GST' FROM INVENTORY;

ROUND() is applied to the calculated GST value to display it up to 1 decimal place.

Q3. Write a query to display customer names in lowercase and their email in uppercase from the CUSTOMER table.

SELECT LCASE(CustName), UCASE(Email) FROM CUSTOMER;

LCASE() converts to lowercase; UCASE() converts to uppercase. These are string functions.

Q4. What will be the output of the following queries? a) SELECT POWER(2,10) b) SELECT MOD(100,7)

a) POWER(2,10) = 2¹■ = 1024 b) MOD(100,7) — 100 ÷ 7 = 14 remainder 2 → Output: 2

Q5. Write a query to display the day, month number, and year of joining for all employees from the EMPLOYEE table.

SELECT DAY(DOJ), MONTH(DOJ), YEAR(DOJ) FROM EMPLOYEE;

DAY(), MONTH(), YEAR() are date functions that extract the respective part from a date column.

Q6. Explain with example how HAVING is different from WHERE when used with GROUP BY.

WHERE filters BEFORE grouping: SELECT CustID FROM SALE WHERE SalePrice > 500000 GROUP BY CustID;

HAVING filters AFTER grouping: SELECT CustID, COUNT(*) FROM SALE GROUP BY CustID HAVING COUNT(*) > 1;

HAVING can use aggregate functions; WHERE cannot.

Q7. Write a query to display all details from the SALE table where the mode of payment is 'Bank Finance'.

SELECT * FROM SALE WHERE PaymentMode = 'Bank Finance';

The WHERE clause with an equality condition on PaymentMode filters only those rows where payment was made via Bank Finance.

Q8. Write a query to display InvoiceNo and Commission (12% of SalePrice, rounded to 0 decimal places) from SALE where Commission > 73000.

SELECT InvoiceNo, ROUND(Commission, 0) FROM SALE WHERE Commission > 73000;

(Assumes Commission column has been added with 12% of SalePrice already computed.)

Q9. What is the output of SELECT SUBSTR('Informatics', 3, 7)?

SUBSTR('Informatics', 3, 7) extracts 7 characters starting from position 3. Positions: I(1) n(2) f(3) o(4) r(5) m(6) a(7) t(8) i(9) c(10) s(11) Starting at 3 (f), taking 7 characters: f-o-r-m-a-t-i Output: 'formati'

Q10. Write a query to count the total number of employees in each designation from the EMPLOYEE table.

SELECT Designation, COUNT(*) 'Total Employees' FROM EMPLOYEE GROUP BY Designation;

GROUP BY Designation groups rows with the same designation together, and COUNT(*) counts employees in each group.

SECTION E: Long Answer Type Questions (4–5 Marks)

■ *Teacher's Note: Long answers should follow a clear structure — definition, explanation, syntax, example, and diagram/table if applicable. Aim for about 8–10 lines.*

Q1. Explain the three categories of Single Row Functions in SQL with two examples each.

Single Row functions are applied on each row individually and return one result per row. They are categorised as:

(A) Numeric Functions — Work on numeric data and return numeric results. Examples: i. POWER(2,8) → 256 — raises 2 to the power 8. ii. ROUND(345.678, 1) → 345.7 — rounds to 1 decimal place.

(B) String Functions — Work on character data and return character or numeric results. Examples: i. UCASE('hello') → 'HELLO' — converts to uppercase. ii. LENGTH('Computer') → 8 — counts characters.

(C) Date and Time Functions — Accept date/time input and return date, time, or numeric results. Examples: i. MONTHNAME('2023-04-15') → 'April' — returns month name. ii. DAY('2023-08-26') → 26 — extracts the day number.

Summary: All three categories make SQL very flexible in processing and presenting data in real-world applications.

Q2. Describe the four set operations on relations: UNION, INTERSECT, MINUS, and Cartesian Product. Use diagrams and examples.

(i) UNION (∪) — Combines all unique rows from both tables. Duplicate rows appear only once. Condition: Same number of columns, compatible data types. Example: DANCE ∪ MUSIC = all students in either event (no repeats).

(ii) INTERSECT (∩) — Returns only the rows that are COMMON to both tables. Example: DANCE ∩ MUSIC = students enrolled in BOTH dance and music.

(iii) MINUS (–) — Returns rows present in the first table but NOT in the second table. Example: DANCE – MUSIC = students in DANCE but not in MUSIC.

(iv) Cartesian Product (×) — Combines every row of the first table with every row of the second. Degree of result = degree(R) + degree(S) Cardinality of result = cardinality(R) × cardinality(S) Example: DANCE (4 rows) × MUSIC (5 rows) = 20 rows in result.

Key Note: UNION, INTERSECT, MINUS are binary operations requiring same structure. Cartesian Product works on any two tables.

Q3. Explain JOIN operation in SQL. Describe the three ways to write a JOIN query with examples from the UNIFORM and COST tables.

JOIN combines tuples from two tables based on a common attribute or condition. It is an extension of Cartesian product with a condition applied.

UNIFORM Table: UCode, UName, UColor COST Table: UCode, Size, Price Common attribute: UCode

(A) Using WHERE clause (Implicit JOIN): SELECT * FROM UNIFORM, COST WHERE UNIFORM.UCode = COST.UCode; Here we manually specify the join condition in the WHERE clause.

(B) Using JOIN...ON clause (Explicit JOIN): SELECT * FROM UNIFORM JOIN COST ON UNIFORM.UCode = COST.UCode; The ON clause specifies the condition. No condition needs to be given in WHERE clause.

(C) Using NATURAL JOIN: SELECT * FROM UNIFORM NATURAL JOIN COST; SQL automatically identifies the common column (UCode) and joins on it. The common column appears only ONCE in the

result — no duplication.

Important points: • For N tables, N−1 JOIN conditions are needed. • Table aliases (e.g., U for UNIFORM) resolve ambiguity when column names are common to both tables. • NATURAL JOIN is the cleanest syntax as it avoids redundancy.

Q4. Explain GROUP BY clause with HAVING. Write SQL queries for the CARSHOWROOM database to demonstrate at least three uses.

GROUP BY clause groups rows that have the same values in specified columns. Aggregate functions are then applied to each group. HAVING clause filters the groups based on aggregate conditions.

Syntax: SELECT column, aggregate_function FROM table GROUP BY column HAVING condition ORDER BY column;

Example 1 — Count of cars bought by each customer: SELECT CustID, COUNT(*) 'Number of Cars' FROM SALE GROUP BY CustID;

Example 2 — Customers who bought more than 1 car: SELECT CustID, COUNT(*) FROM SALE GROUP BY CustID HAVING COUNT(*) > 1;

Example 3 — Total sales and count grouped by payment mode: SELECT PaymentMode, COUNT(PaymentMode), SUM(SalePrice) FROM SALE GROUP BY PaymentMode ORDER BY PaymentMode;

Key difference: WHERE cannot be used with aggregate functions; HAVING is designed specifically for this. GROUP BY must come before HAVING, and HAVING must come before ORDER BY in the query.

Q5. Create the CARSHOWROOM database schema. Write SQL queries to: (a) Display the FinalPrice after adding 12% GST; (b) Calculate EMI if the car's FinalPrice is paid in 10 instalments; (c) Display the remaining amount if each EMI is rounded to nearest 1000.

Database: CARSHOWROOM Table: INVENTORY (CarID, CarName, Price, Model, YearManufacture, FuelType)

Step 1 — Add FinalPrice column: ALTER TABLE INVENTORY ADD FinalPrice NUMERIC(10,1); UPDATE INVENTORY SET FinalPrice = ROUND(Price + (12/100)*Price, 1);

(a) Display FinalPrice with GST: SELECT CarID, CarName, Price, ROUND((12/100)*Price, 1) 'GST', FinalPrice FROM INVENTORY;

(b) Calculate monthly EMI (FinalPrice ÷ 10 instalments): SELECT CarID, FinalPrice, ROUND(FinalPrice/10, 0) 'EMI' FROM INVENTORY;

(c) Remaining amount after rounding EMI to nearest 1000: SELECT CarID, FinalPrice, ROUND(FinalPrice/10, -3) 'EMI', MOD(FinalPrice, ROUND(FinalPrice/10, -3) * 10) 'Remaining Amount' FROM INVENTORY;

This elegant query uses ROUND() with negative precision to round to nearest 1000, and MOD() to find the leftover amount not covered by rounded EMIs.

SECTION F: Matching Questions

Match each item in Column A with the correct item in Column B.

■ *Teacher's Note: Matching questions test exact knowledge of function names and their purposes. Go through the list systematically rather than guessing.*

Matching Table 1: SQL Functions and Their Purpose

Column A (Function)	Column B (Purpose/Output)
1. POWER(2, 5)	A. Returns 'November'
2. ROUND(3.567, 2)	B. Returns 'HELLO'
3. MOD(25, 7)	C. Returns 32
4. UCASE('hello')	D. Returns 3.57
5. MONTHNAME('2023-11-10')	E. Returns 4
6. LENGTH('SQL')	F. Returns 3
7. LEFT('Science', 3)	G. Returns current date and time
8. NOW()	H. Returns 'Sci'

■ ***Answers: 1-C, 2-D, 3-E, 4-B, 5-A, 6-F, 7-H, 8-G***

Matching Table 2: SQL Operations and Descriptions

Column A (Operation/Clause)	Column B (Description)
1. UNION	A. Filters groups after GROUP BY
2. INTERSECT	B. Returns rows from first table not in second
3. MINUS	C. Combines all unique rows from both tables
4. HAVING	D. Removes both leading and trailing spaces
5. NATURAL JOIN	E. Returns only rows common to both tables
6. GROUP BY	F. Eliminates duplicate common column in result
7. TRIM()	G. Groups rows with same column values together
8. DISTINCT	H. Eliminates duplicate rows from result

■ ***Answers: 1-C, 2-E, 3-B, 4-A, 5-F, 6-G, 7-D, 8-H***

Matching Table 3: Function Category Match

Column A (Function Name)	Column B (Category)
1. NOW()	A. Aggregate Function

2. AVG()	B. Date Function
3. INSTR()	C. Numeric Function
4. ROUND()	D. String Function
5. DAYNAME()	E. Date Function
6. SUM()	F. Aggregate Function
7. LTRIM()	G. String Function
8. MOD()	H. Numeric Function

■ ***Answers: 1-B, 2-A, 3-D, 4-C, 5-E, 6-F, 7-G, 8-H***

Quick Reference: All SQL Functions at a Glance

■ *Teacher's Note: Keep this page handy during revision. Going through this table 2–3 times before the exam will boost your confidence significantly!*

Function	Category	Syntax	Example & Output
POWER	Numeric	POWER(x,y)	POWER(2,8) → 256
ROUND	Numeric	ROUND(n,d)	ROUND(4.567,2) → 4.57
MOD	Numeric	MOD(m,n)	MOD(17,5) → 2
UCASE/UPPER	String	UCASE(str)	UCASE('hi') → 'HI'
LCASE/LOWER	String	LCASE(str)	LCASE('HI') → 'hi'
MID/SUBSTR	String	MID(str,pos,n)	MID('Info',2,3) → 'nfo'
LENGTH	String	LENGTH(str)	LENGTH('SQL') → 3
LEFT	String	LEFT(str,n)	LEFT('Comp',4) → 'Comp'
RIGHT	String	RIGHT(str,n)	RIGHT('Sci',3) → 'Sci'
INSTR	String	INSTR(str,sub)	INSTR('Info','fo') → 3
LTRIM	String	LTRIM(str)	Removes left spaces
RTRIM	String	RTRIM(str)	Removes right spaces
TRIM	String	TRIM(str)	Removes both side spaces
NOW	Date/Time	NOW()	Current date & time
DATE	Date/Time	DATE(expr)	Extracts date part
MONTH	Date/Time	MONTH(date)	MONTH('2023-08-15') → 8
MONTHNAME	Date/Time	MONTHNAME(date)	→ 'August'
YEAR	Date/Time	YEAR(date)	YEAR('2023-08-15') → 2023
DAY	Date/Time	DAY(date)	DAY('2023-08-15') → 15
DAYNAME	Date/Time	DAYNAME(date)	→ 'Tuesday'
MAX	Aggregate	MAX(col)	Largest value
MIN	Aggregate	MIN(col)	Smallest value
AVG	Aggregate	AVG(col)	Average value
SUM	Aggregate	SUM(col)	Total sum
COUNT	Aggregate	COUNT(*)/COUNT(col)	Row count

Best of luck for your examinations! Remember — understanding SQL concepts and practising queries regularly on a computer is the best preparation. Every function you practise today becomes second nature in the exam hall tomorrow.