**SQL ASSIGNMENT**

* **Vrushasen Deshmukh**

**Part A – Conceptual Questions**

**SQL Basics :**

**Q.1] What is SQL and how is it different from MySQL/SQL Server/Oracle?**

* SQL stands for Structure Query Language, and it is used to query and manage relational databases.
* Database Management Systems (DBMS) also use SQL, but also have their own added features like tools, performance optimizations, and security.

**Q.2] Explain DDL, DML, DCL, TCL with examples.**

**DDL** - Used to define or change the structure of the database objects (tables, views, indexes). Commands: CREATE, ALTER, DROP, TRUNCATE. Example: CREATE TABLE Student (

id INT PRIMARY KEY,

name VARCHAR(50),

age INT

);

**DML** - Used to insert, update, delete, and retrieve data from tables.

Commands: INSERT, UPDATE, DELETE, SELECT.

Example: INSERT INTO Student VALUES (1,'Rahul',20);

UPDATE Student SET age=21 WHERE id=1;

DELETE FROM Student WHERE id=1;

**DCL** - Used to control user access and permissions in the database.

Commands: GRANT, REVOKE.

Example: GRANT SELECT ON Student TO user1;

REVOKE SELECT ON Student FROM user1;

**TCL** - Used to manage transactions and maintain database integrity.

Commands: COMMIT, ROLLBACK, SAVEPOINT.

Example: BEGIN;

UPDATE Student SET age=22 WHERE id=1;

SAVEPOINT before\_change;

ROLLBACK TO before\_change; -- undo back to savepoint

COMMIT; -- finalize all changes

**Q.3] What are constraints in SQL? Name all types.**

* Constraints are rules applied on table columns to maintain accuracy, consistency, and integrity of the data in a database.
* They also prevent invalid data from being entered.
* Types of constraints:
  + - NOT NULL
    - UNIQUE
    - PRIMARY KEY
    - FOREIGN KEY
    - CHECK
    - DEFAULT

**Q.4] What is the difference between PRIMARY KEY and UNIQUE KEY?**

* Primary Key : Uniquely identifies each row, only one per table, cannot be NULL.
* Unique Key : Ensures uniqueness but can allow NULLs, multiple unique keys possible.

**Q.5] What is the difference between DELETE, TRUNCATE, and DROP?**

* DELETE : Removes specific rows from a table using a condition (WHERE clause).
* TRUNCATE : Removes all rows from a table (empties the table).
* DROP : Deletes the entire table along with its data and structure.

Q.6] **Explain the difference between CHAR and VARCHAR.**

* CHAR :- A fixed-length string that always uses *n* characters, padding with spaces if the data is shorter.
* VARCHAR(n) :- A variable-length string that uses only the required space for actual data (plus 1–2 bytes).

**Q.7] What is the difference between WHERE and HAVING?**

* **WHERE: Used to filter rows before grouping or aggregation.**
* **HAVING: Used to filter groups after aggregation (with GROUP BY).**

**Q.8] What are aggregate functions in SQL? Give examples.**

* Functions in SQL that perform calculations on multiple rows and return a single summarized value.
* Examples:
  + COUNT()
  + SUM()
  + AVG()
  + MIN()
  + MAX()

**Q.9] What is the difference between COUNT(\*) and COUNT(column\_name)?**

* COUNT(\*): Counts all rows in a table, including rows with NULL values.
* COUNT(column\_name): Counts only rows where the specified column is NOT NULL.

**Q.10] Explain the difference between BETWEEN and IN.**

* BETWEEN: Used to check if a value lies within a continuous range (inclusive).
* IN: Used to check if a value matches any value from a given list.

**Joins :**

**Q.11] What is the difference between INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN?**

* Inner Join → Returns only the rows where A and B both match.
* Left Join → Returns all rows from A, plus matching rows from B.
* Right Join → Returns all rows from B, plus matching rows from A.
* Full Join → Returns all rows from both A and B, with NULLs where no match exists.

**Q.12] What is a SELF JOIN?**

* A self join is a type of join in SQL where a table is joined with itself, treating the same table as if it were two different tables by using table aliases.

Q.13] **What are CROSS JOIN and CARTESIAN PRODUCT?**

* Cross Join : It returns the Cartesian Product of two tables, meaning every row from the first table is paired with every row from the second table.
* Cartesian Product : It is the mathematical result of combining all rows of one set with all rows of another set. In SQL, it’s the output of a cross join.

Q.14] **What is the difference between ANTI JOIN and SEMI JOIN?**

* Semi Join : Returns rows from the left table where a match exists in the right table.
* Anti Join : Returns rows from the left table where no match exists in the right table.

Q.15 ] **Can we join a table to itself? How?**

* Yes, a table can be joined to itself using a SELF JOIN. This is done by treating the same table as two separate instances using table aliases, enabling comparison of rows within the same table.

**Indexes & Performance :**

Q.16] **What is an index in SQL? Why is it used?**

* An index in SQL is a data structure that improves the speed of data retrieval operations on a database table.
* Indexes are used to enhance query performance by reducing the amount of data the database must scan, especially for large tables. They are most useful in searching, filtering, and sorting operations.

Q.17] **Difference between Clustered and Non-Clustered Index.**

* Clustered Index:
  + Defines the **physical order** of data in the table.
  + A table can have **only one** clustered index.
  + Data rows are stored in the order of the clustered index key.
* Non-Clustered Index:
  + Stores a **separate structure** that points to the data rows.
  + A table can have **multiple** non-clustered indexes.
  + Does **not** affect the physical order of data in the table.

Q.18] **What is a composite index?**

* A composite index is an index in SQL created on **two or more columns** of a table, used to speed up queries that filter, sort, or search based on those combined columns.

Q.19] **Does an index improve INSERT performance? Why/Why not?**

* No, an index does not improve INSERT performance because it slows down INSERT operations because whenever a new row is added, the database must also update all the related indexes to maintain them.

**Q.20]** **What is an execution plan in SQL?**

* An execution plan in SQL is basically the database’s roadmap for running a query. It shows the steps the system takes—like which indexes it uses, how it joins tables, and the order of operations—to fetch the result as efficiently as possible.

**Advanced Concepts :**

**Q.21]** **What is a subquery? Difference between correlated and non-correlated subqueries.**

* Subquery : It is a query inside another query, used to return data that the outer query can use for filtering or calculations.
* Correlated Subquery: Depends on the outer query for its values and is executed repeatedly, once for each row processed by the outer query.
* Non-Correlated Subquery: Runs independently of the outer query and is executed once, with its result passed to the main query.

**Q.22] What is the difference between View and Materialized View?**

* A view is like a saved SQL query—it doesn’t store data itself but always shows the latest results from the underlying tables whenever you run it.
* A materialized view stores the results of the query, so it’s much faster to read from, but the data can get outdated unless it’s refreshed.

**Q.23] Can we perform DML operations on a view?**

* Yes, you can perform DML operations on a view, but only if the view is updatable.

**Q.24] What is a stored procedure?**

* A stored procedure is like a saved block of SQL code that you can run whenever you need.
* Instead of writing the same queries again and again, you put them into a procedure, and then just call it by name.

**Q.25] Difference between stored procedure and function.**

* A stored procedure is like a routine that can run multiple SQL statements, perform actions and may or may not return a value.
* A function, on the other hand, is designed to always return a single value and is mostly used inside queries, just like built-in functions.

**Q.26] What are window functions in SQL? Name a few.**

* Window functions in SQL are special functions that perform calculations across a set of rows related to the current row, without collapsing them into a single result like GROUP BY does. They’re often used with the OVER() clause.
* Some common ones are:
  + ROW\_NUMBER()
  + RANK()
  + DENSE\_RANK()
  + NTILE()
  + SUM(), AVG(), COUNT()

Q.27] **Difference between RANK(), DENSE\_RANK(), and ROW\_NUMBER().**

* RANK() → Gives ranks with gaps. If two rows tie at rank 1, the next row gets rank 3.
* DENSE\_RANK() → Similar to RANK, but no gaps. If two rows tie at rank 1, the next row gets rank 2.
* ROW\_NUMBER() → Just gives a unique number to each row, no matter if values tie.

**Q.28] What is the difference between LEAD and LAG functions?**

* LEAD gives you access to the next row’s value in the result set.
* LAG gives you access to the previous row’s value.

**Q.29] What are CTEs in SQL? How are they different from subqueries?**

* CTE stands for Common Table Expression. It is like a temporary, named result set that you define using ‘WITH’ and then use in your main query.
* The difference between CTE and subqueries is that:
  + CTE is more readable, can be reused multiple times in the same query, and supports recursion.
  + While a subquery is just an inline query used once inside another query.

**Q.30] What is the difference between UNION and UNION ALL?**

* UNION combines results from two queries and removes duplicates.
* UNION ALL also combines results but keeps all duplicates.

**Transactions:**

**Q.31] What are ACID properties?**

* ACID properties are the rules that make sure database transactions are safe and reliable:
  + A : Atomicity → Everything in a transaction happens completely or not at all.
  + C : Consistency → The data always follows the rules and stays valid.
  + I : Isolation → One transaction doesn’t mess with another while they’re running.
  + D : Durability → Once saved, the changes stick—even if the system crashes.

**Q.32] Difference between COMMIT and ROLLBACK.**

* COMMIT is used to save all the changes you’ve made in a transaction permanently to the database.
* ROLLBACK is used to undo all the changes made in the current transaction, bringing the database back to its previous state.

**Q.33] What is a SAVEPOINT in SQL?**

* A SAVEPOINT is basically a bookmark inside a transaction.
* It lets you go back just to that spot if something goes wrong, instead of cancelling the whole transaction.

**Q.34] Difference between Implicit and Explicit transactions.**

* Implicit transactions happen automatically—SQL starts and handles them for you whenever you run certain statements like INSERT or UPDATE.
* Explicit transactions are ones you control yourself, using BEGIN, COMMIT, and ROLLBACK to decide exactly when to start, save, or undo changes.

**Data Engineering Focus :**

**Q.35] How do you handle NULL values in SQL?**

* In SQL, you handle ‘NULL’ values by either checking for them or replacing them.
* You can use ‘IS NULL’ or ‘IS NOT NULL’ to find rows with or without ‘NULLs’, and functions like ‘COALESCE()’ or ‘IFNULL()’ let you give a default value instead of ‘NULL’.
* Basically, you decide whether to filter them out or fill them in.

**Q.36] Difference between IS NULL and = NULL.**

* In SQL, you can’t use ‘= NULL’ to check for ‘NULL’ —it won’t work because ‘NULL’ means “unknown.”
* You must use ‘IS NULL’ to test for ‘NULL’ values.

Q.37] **How can you find duplicate records in a table?**

* You can find duplicate records in a table by grouping the rows and using ‘COUNT()’ to see which values appear more than once. For example, ‘HAVING COUNT(\*) > 1’.

**Q.38] How to remove duplicates while keeping only one record?**

* To remove duplicates but keep one record, you can use a ‘DELETE’ with a ‘CTE’ or a subquery that identifies duplicates using ‘ROW\_NUMBER()’.
* Then delete all rows where the row number is greater than 1.

**Q.39] What is normalization? Name different normal forms.**

* Normalization is the process of organizing a database to reduce redundancy and ensure data is stored efficiently. It helps avoid duplicate data and makes updates easier.
* Forms of Normalization :
  + 1NF (First Normal Form)
  + 2NF (Second Normal Form)
  + 3NF (Third Normal Form)
  + BCNF (Boyce-Codd Normal Form)
  + 4NF (Fourth Normal Form)
  + 5NF (Fifth Normal Form)

**Q.40] Difference between OLTP and OLAP databases.**

* OLTP (Online Transaction Processing) is used for daily operations, like inserting, updating, or deleting data in applications.
* OLAP (Online Analytical Processing) is used for analysis and reporting.