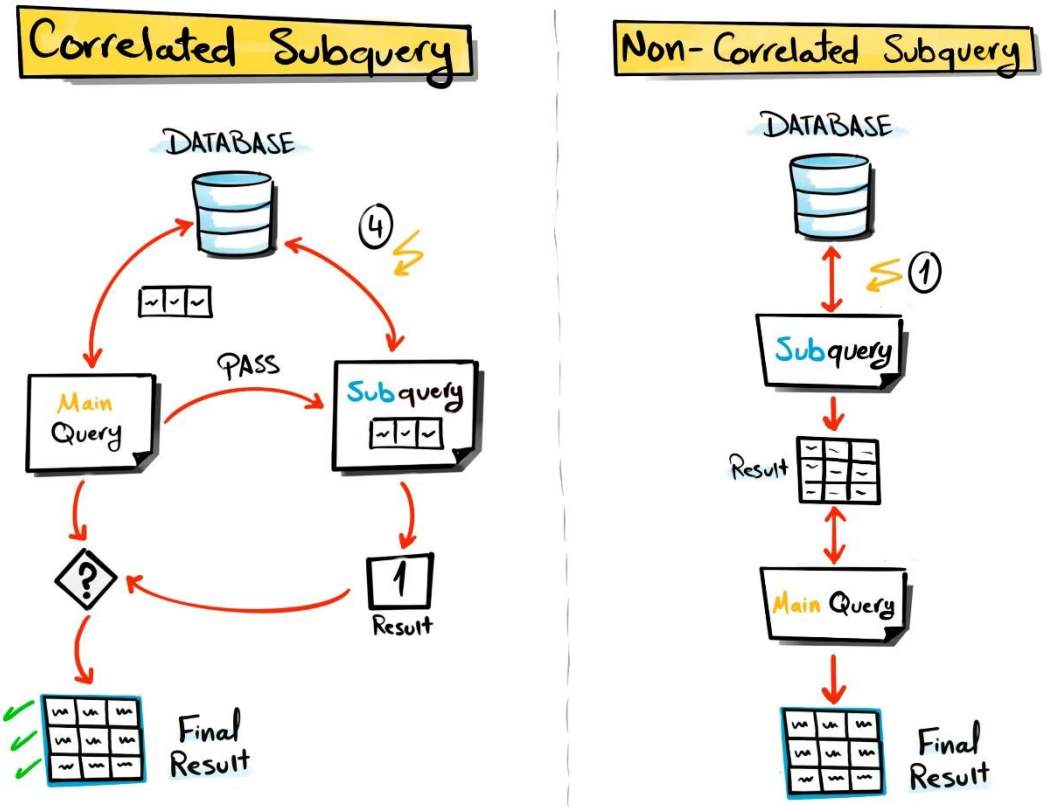
Nitya CloudTech Pvt Ltd.

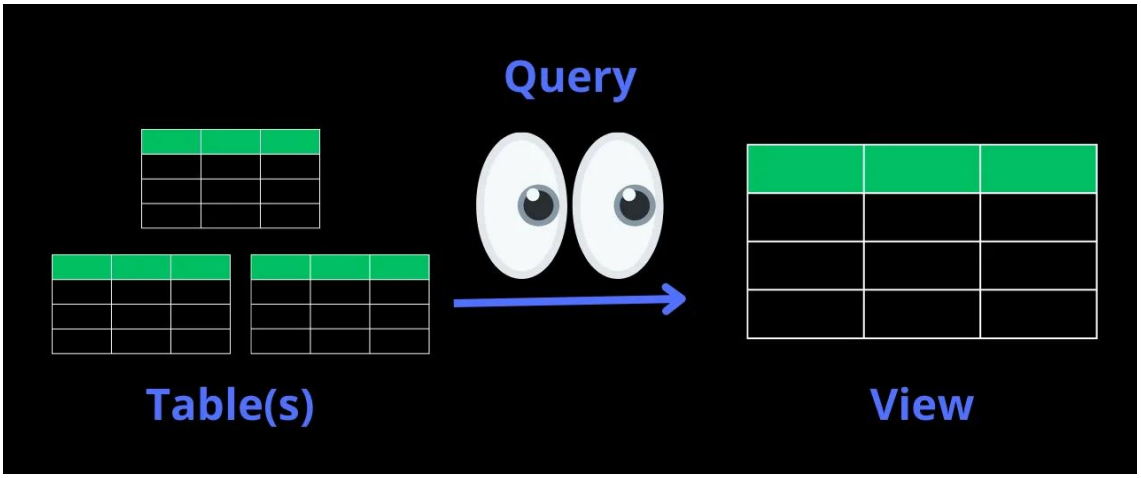
SQL Scenario-Based Interview Questions & Answers

Intermediate to Advanced SQL Questions

1. **What is the difference between correlated and non-correlated subqueries?**
   - **Answer**:
     - **Non-correlated subquery**: A subquery that can be run independently and returns a result which is used by the outer query.
     - **Correlated subquery**: A subquery that refers to columns in the outer query, requiring it to be executed for each row processed by the outer query.



2. **What is a view, and why would you use it?**
   - **Answer**: A view is a virtual table created by a `SELECT` query. It allows for simplified, reusable queries, data security (by restricting access to specific columns), and abstraction over complex joins or aggregations.



3. **How can you find duplicate records in a table?**

➢ **Answer**:

```
SELECT column_name, COUNT(*)
FROM table_name
GROUP BY column_name
HAVING COUNT(*) > 1;
```

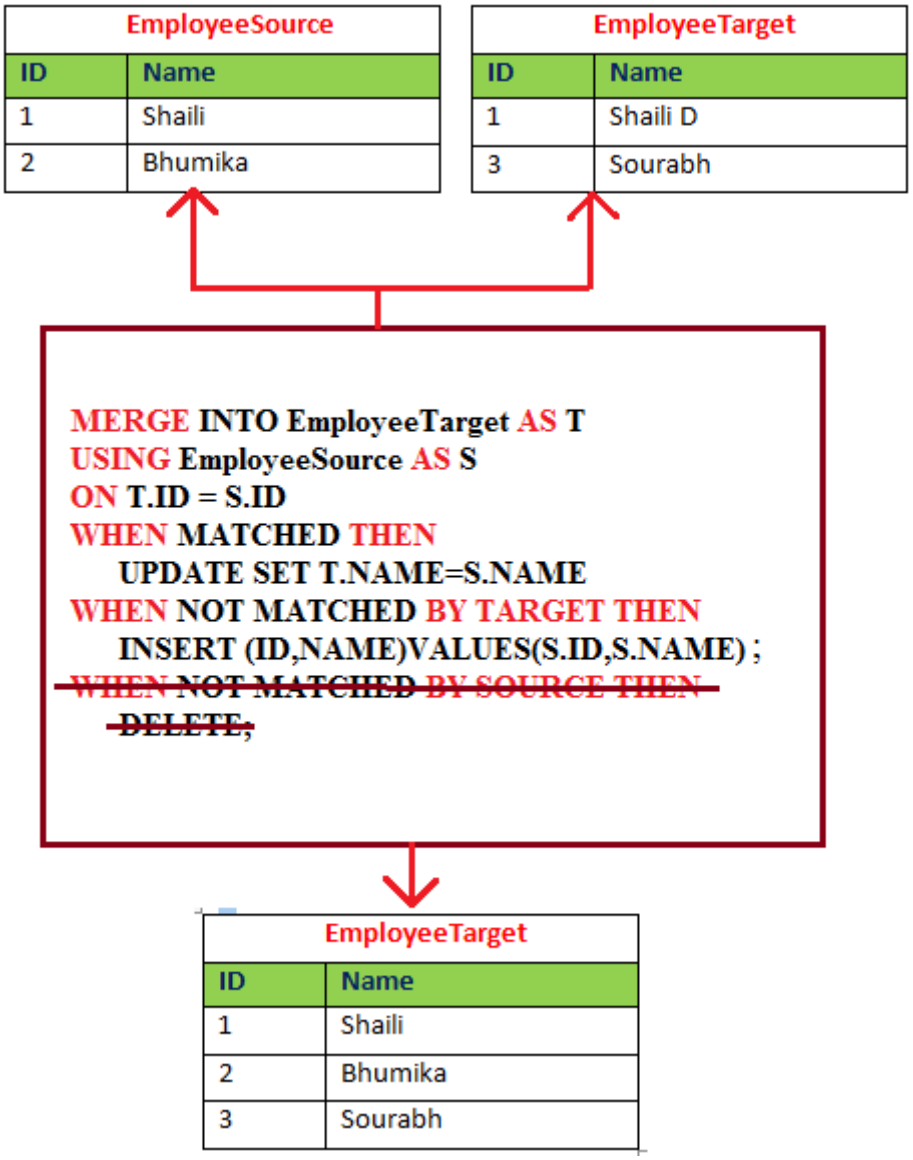
This query groups by the specified column and shows values that appear more than once.

id	first_name	last_name	email
1	Carine	Schmitt	carine.schmitt@verizon.net
4	Janine	Labrune	janine.labrune@aol.com
6	Janine	Labrune	janine.labrune@aol.com
2	Jean	King	jean.king@me.com
12	Jean	King	jean.king@me.com
5	Jonas	Bergulfsen	jonas.bergulfsen@mac.com
10	Julie	Murphy	julie.murphy@yahoo.com
11	Kwai	Lee	kwai.lee@google.com
3	Peter	Ferguson	peter.ferguson@google.com
9	Roland	Keitel	roland.keitel@yahoo.com
14	Roland	Keitel	roland.keitel@yahoo.com
7	Susan	Nelson	susan.nelson@comcast.net
13	Susan	Nelson	susan.nelson@comcast.net
8	Zbyszek	Piestrzeniewicz	zbyszek.piestrzeniewicz@att.net

4. **What is the purpose of the `MERGE` statement?**

➢ **Answer**: `MERGE` allows you to perform `INSERT`, `UPDATE`, or `DELETE` operations in a single statement based on conditions.

It’s commonly used for handling data changes in data warehousing.

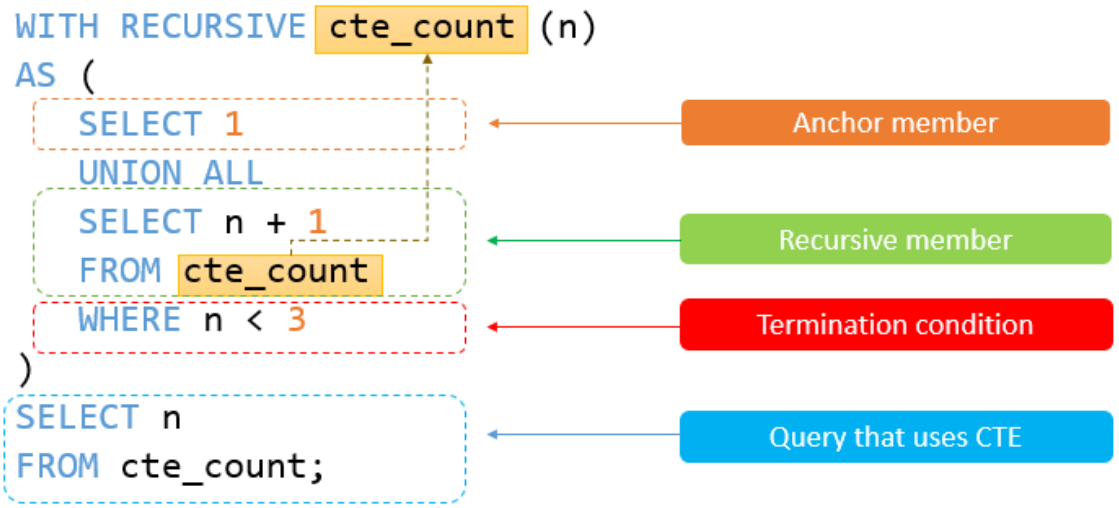


5. What is a recursive CTE?

- **Answer**: A recursive CTE is a CTE that references itself. It's useful for hierarchical data, such as organizational structures or folder directories.

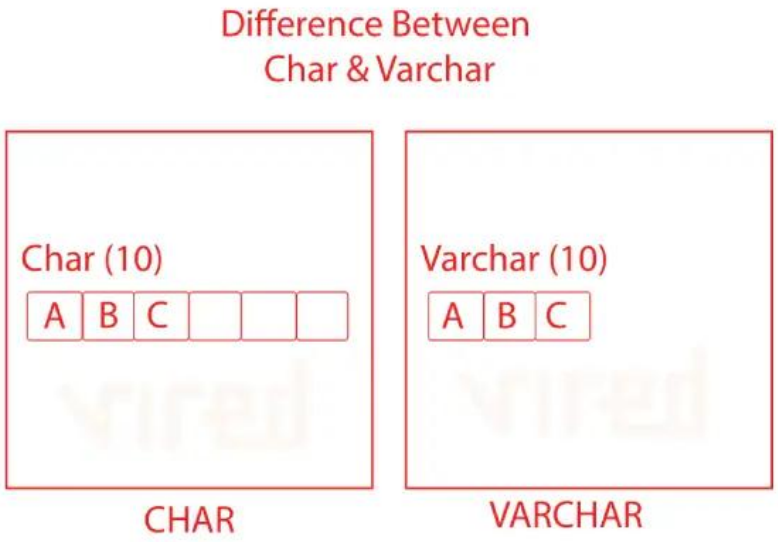
```
WITH RECURSIVE hierarchy AS (
    SELECT employee_id, manager_id
    FROM employees
    WHERE manager_id IS NULL
    UNION ALL
    SELECT e.employee_id, e.manager_id
    FROM employees e
```

```
    INNER JOIN hierarchy h ON e.manager_id
= h.employee_id
)
SELECT * FROM hierarchy;
```



6. What is the difference between CHAR and VARCHAR?

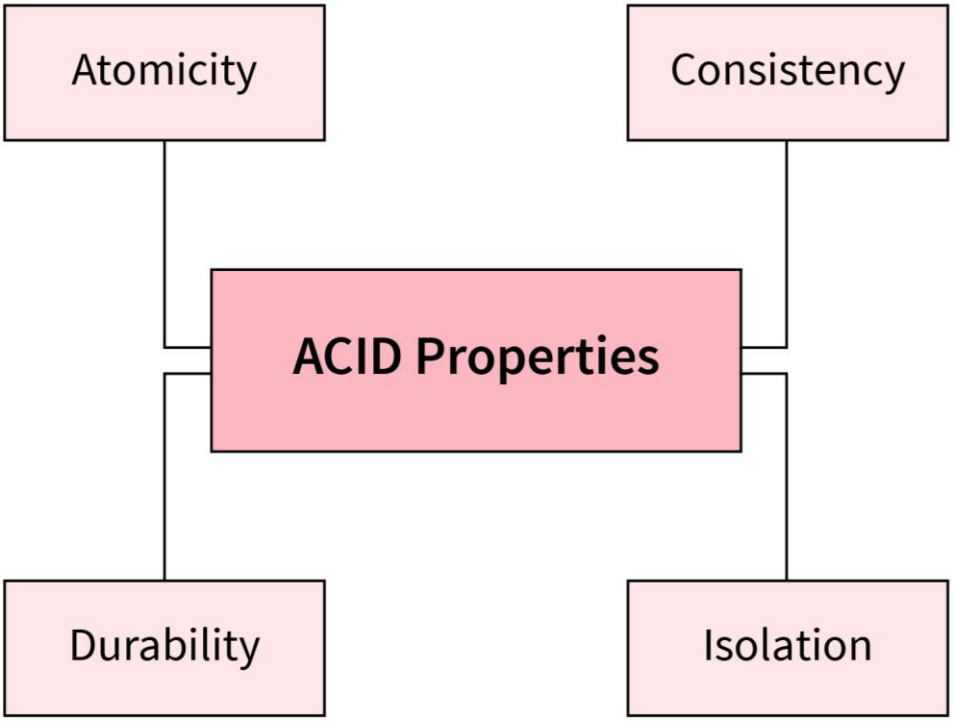
- **Answer**: CHAR is a fixed-length string, padding with spaces if necessary, whereas VARCHAR is a variable-length string, storing only the actual characters. VARCHAR is more space-efficient for variable-length data.



7. Explain `ACID` properties in SQL.

➢ **Answer**:

- **Atomicity**: Ensures that all operations within a transaction are completed; if one fails, the transaction is aborted.
- **Consistency**: Guarantees data integrity by ensuring the database remains valid after a transaction.
- **Isolation**: Ensures transactions are executed independently.
- **Durability**: Ensures completed transactions are saved even if the system crashes.



8. What is a materialized view, and how does it differ from a regular view?

➢ **Answer**: A materialized view stores the query results physically on disk, making data retrieval faster. Unlike regular views, it doesn't require re-running the query each time.

eventname	total_sales
Dolly Parton	10.00
Gotterburg	48.00

events

eventid	catid	eventname
1	8	Gotterburg
18	8	Gotterburg
5311	9	Dolly Parton

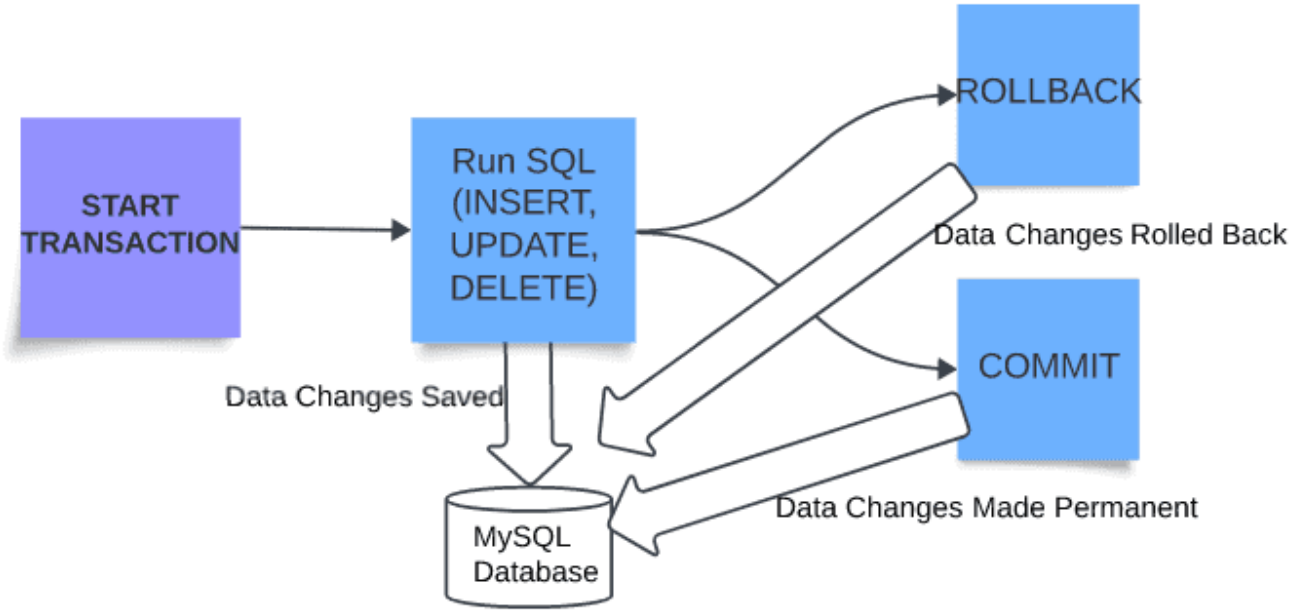
sales

salesid	eventid	cust	price
1	1	c1	12.00
2	18	c1	36.00
3	5311	c2	10.00

9. What are `ROLLBACK`, `COMMIT`, and `SAVEPOINT`?

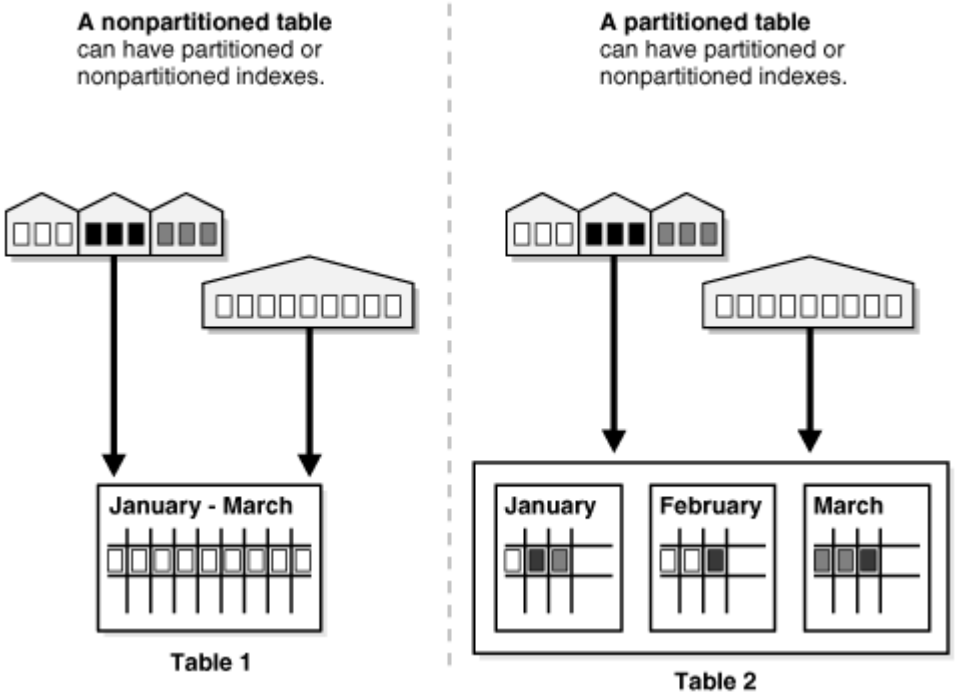
➢ **Answer**:

- **COMMIT**: Saves changes made by the transaction.
- **ROLLBACK**: Reverts changes made by the transaction.
- **SAVEPOINT**: Creates a checkpoint within a transaction to roll back to if needed without rolling back the entire transaction.



10. **What is a partitioned table, and why use it?**

- **Answer**: Partitioning divides a large table into smaller, manageable pieces, improving query performance by reducing the data scanned. Common partitioning strategies are range, list, and hash partitioning.



11. **Explain the `OVER` clause and give an example of its use.**

- **Answer**: The `OVER` clause defines a window for applying window functions. For example:

```
SELECT employee_id, salary,
       AVG(salary) OVER (PARTITION BY
department_id) AS avg_dept_salary
FROM employees;
```

This calculates the average salary for each department.

```
SELECT StudId, SubjectId, Marks,
ROW_NUMBER() OVER
(PARTITION BY StudId ORDER BY Marks DESC) AS [Rank_By_Marks]
FROM dbo.StudentScoreCard
```

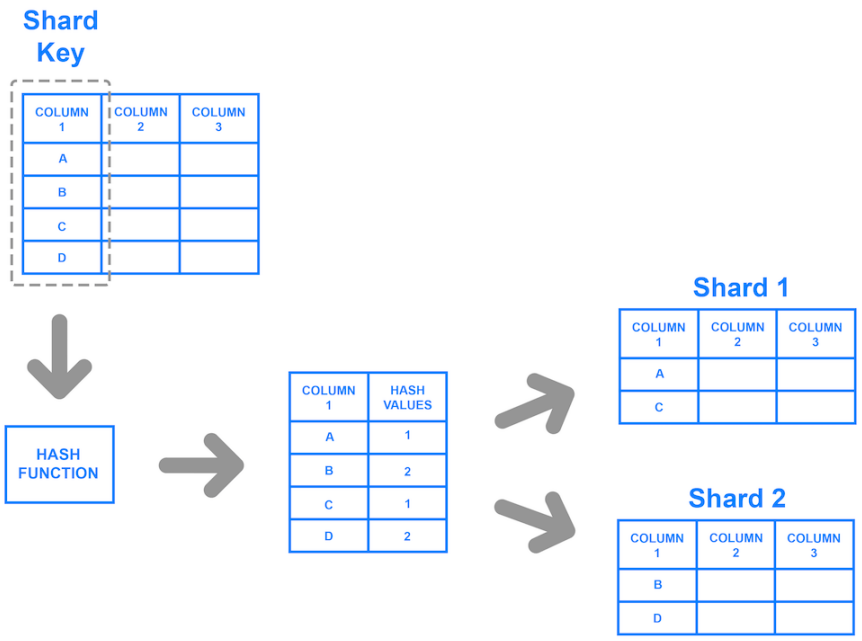
Results | Messages

StudId	SubjectId	Marks	Rank_By_Marks
1	101	85	1
1	102	81	2
1	103	75	3
1	104	70	4
2	101	78	1
2	102	70	2
2	103	65	3
2	104	55	4
3	104	83	1
3	103	70	2
3	101	70	3
3	102	69	4
4	103	90	1
4	104	85	2
4	102	80	3
4	101	77	4

Rank is assigned by partitioning student, rank order is sort based on marks. Highest marks will be on top rank

12. What is database sharding?

➢ **Answer**: Sharding is a database architecture pattern that partitions large datasets across multiple servers. It enhances performance and allows horizontal scaling.



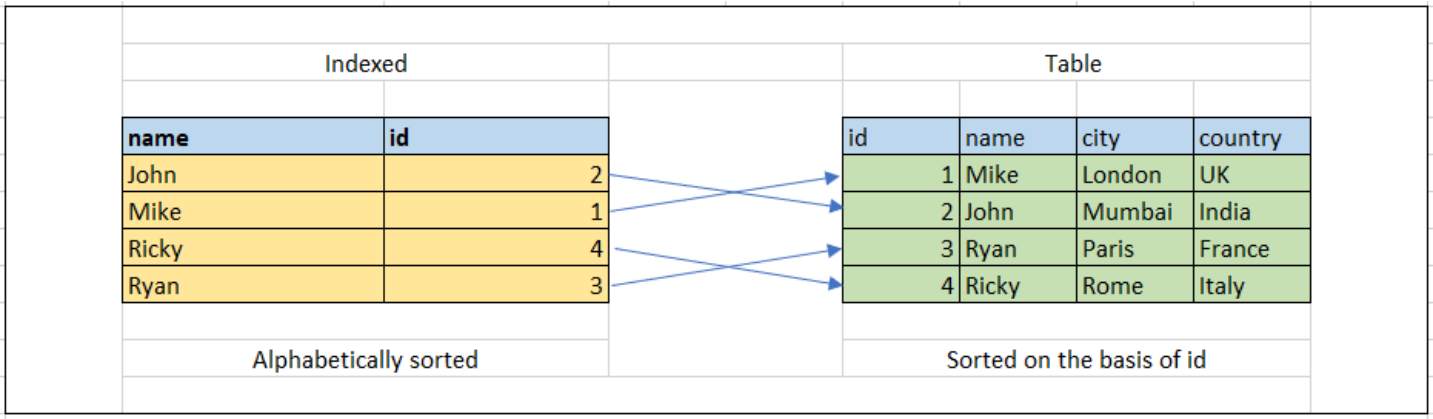
13. **How can you retrieve the current date and time in SQL?**
    - **Answer**: Use the `CURRENT_TIMESTAMP` function or its variations like `GETDATE()` (SQL Server) or `NOW()` (MySQL).

14. **How do you calculate the difference between two dates in SQL?**
    - **Answer**:

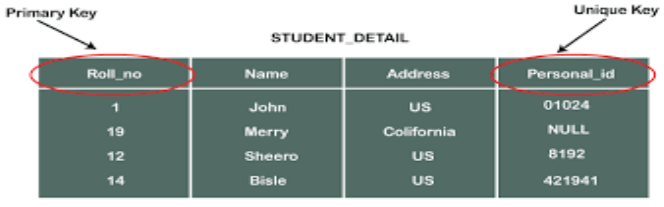
```
SELECT DATEDIFF(day, start_date, end_date)
AS date_difference;
```

The function and syntax vary across database systems (`DATEDIFF`, `TIMESTAMPDIFF`, etc.).

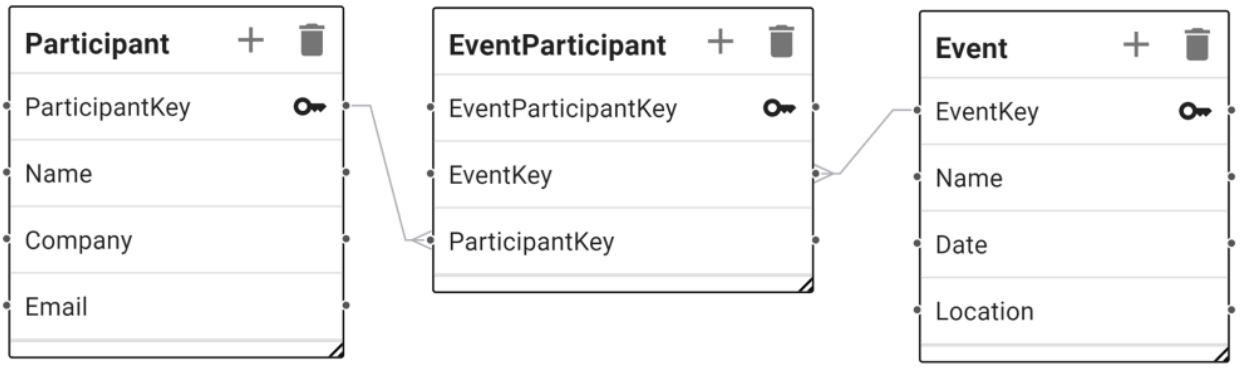
15. **What is the purpose of indexing in SQL, and when should you avoid it?**
    - **Answer**: Indexing speeds up data retrieval. Avoid over-indexing or indexing frequently updated columns, as it can slow down `INSERT`, `UPDATE`, and `DELETE` operations.



16. **Explain `UNIQUE` constraint vs. `PRIMARY KEY`.**
    - **Answer**: Both enforce uniqueness, but a table can have multiple `UNIQUE` constraints, while it can only have one `PRIMARY KEY`, which also disallows nulls.

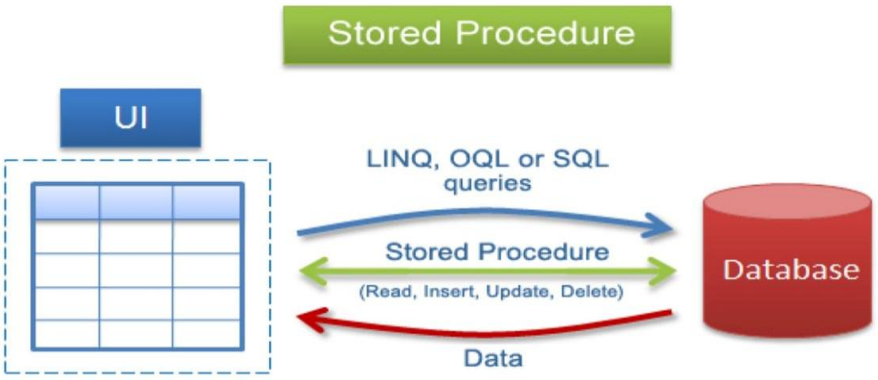


17. **How do you implement many-to-many relationships in SQL?**
    - **Answer**: Use a junction (or associative) table with foreign keys linking the two tables in a many-to-many relationship.



18. **What is the purpose of `COALESCE` and `NULLIF` functions?**
    - **Answer**:
      - **COALESCE**: Returns the first non-null value in a list of arguments.
      - **NULLIF**: Returns null if two expressions are equal; otherwise, it returns the first expression.

19. **How do you handle stored procedure error handling in SQL?**
    - **Answer**: Use `TRY...CATCH` blocks (in SQL Server) or `DECLARE EXIT HANDLER` (in MySQL) to catch and handle errors in stored procedures.



20. What is data denormalization, and when might it be used?

- **Answer**: Denormalization adds redundancy to speed up read operations by combining tables. It's useful in read-heavy applications, particularly in OLAP (Online Analytical Processing).

Student_ID	Name
1	Alex
2	Marie

Student_ID	Assignment_ID	Mark
1	1	20
1	2	35.50
2	1	45
2	2	45

⟹

Student_ID	Name	Total_Marks
1	Alex	55.5
2	Marie	90

These questions expand on essential SQL topics, including database design, data manipulation, indexing, transaction handling, and optimization. Knowing these concepts in depth will help you tackle a broad range of SQL interview questions confidently.

If you like the content, please consider supporting us by sharing it with others who may benefit. Your support helps us continue creating valuable resources!

Scan any QR using PhonePe App

Aditya chandak