Manish Jaiswal

SQL Interview Question

MASTERING

Clustered Index in SQL

swipe

Manish Jaiswal

SQL Interview Question

01

What is a Clustered Index?

- A clustered index organizes the actual data in the table based on the index key.
- Think of it like the table of contents in a book—guiding you straight to the content.
- Only one clustered index per table because it’s tied to the table’s structure.

swipe

Manish Jaiswal

SQL Interview Question

02

How Clustered Indexes Work

- Physically **rearranges** data in the table to align with the indexed column(s).
- Great for **range** queries and fast data **retrieval** because data is stored in sorted order.
- Commonly created on **primary key** columns by default.

swipe

03

Example: E-Commerce Database

- Imagine a table storing Order History with millions of rows.
- Creating a clustered index on **OrderID** or **OrderDate** can speed up searches on recent orders.
- Especially useful for report generation where date-based retrievals are common.

swipe

Manish Jaiswal

SQL Interview Question

04

When and Where to Use

- Use clustered indexes on columns frequently used in:
  - WHERE clauses for data retrieval.
  - Range-based searches (e.g., date ranges).
  - Sorting operations to optimize data retrieval.
- **Avoid** on columns with **frequent updates**—can cause data shuffling and impact performance.

swipe

Manish Jaiswal

SQL Interview Question

05

Why Use Clustered Indexes?

- **Improves performance** by minimizing the number of data reads.
- Reduced **I/O operations** due to sorted data structure.
- Enhances **speed** for sequential data access (e.g., pulling a range of dates or ID ranges).

swipe

Manish Jaiswal

SQL Interview Question

06

Performance Considerations

- Be mindful of:
  - Data shuffling with frequent updates.
  - Impact on INSERT and UPDATE operations.
- Tip: Use clustered indexes on columns with low volatility and high retrieval rates.

swipe

Manish Jaiswal

SQL Interview Question

07

Best Practices for Clustered Indexes

- **Primary Key** column often a good candidate.
- Ideal for **tables with large datasets** needing optimized read performance.
- Avoid adding clustered indexes to frequently modified columns to maintain efficiency.

swipe