Perfect 👍 Let’s learn **Indexes in SQL** step by step — from **scratch to expert level (concept → syntax → types → performance → interview)**.

**🧩 1. What is an Index in SQL?**

**Definition:**  
An **index** is a *database object* that improves the **speed of data retrieval** on a table.  
It works like an **index in a book** — instead of scanning the entire book, you directly jump to the page number.

**👉 Without Index:**  
The database performs a **Full Table Scan** (checks every row).

**👉 With Index:**  
The database uses the **Index Tree** (B-Tree, Hash, etc.) to quickly locate data.

**⚙️ 2. Why Use Indexes?**

| **Scenario** | **Effect** |
| --- | --- |
| Searching (WHERE, JOIN, ORDER BY, GROUP BY) | ✅ Much faster |
| Inserting, Updating, Deleting | ❌ Slightly slower (index must be updated) |
| Reading often, Writing rarely | ✅ Best for indexing |

**🔹 Remember:**

Index = Faster SELECTs but Slower INSERT/UPDATE/DELETE

**🧱 3. How an Index Works (Internally)**

Most databases (MySQL, SQL Server, Oracle, PostgreSQL) use a **B-Tree (Balanced Tree)** structure.

**Example:**

If you create an index on Employee(Name), it internally stores:

A → rowid=1

B → rowid=7

C → rowid=3

...

Searching for “C” becomes logarithmic time O(log n) instead of linear O(n).

**🧾 4. Syntax of Creating and Managing Indexes**

**➤ Create Index**

CREATE INDEX index\_name

ON table\_name(column\_name);

**➤ Create Unique Index**

CREATE UNIQUE INDEX idx\_employee\_email

ON Employee(Email);

* Prevents duplicate values.
* Same effect as UNIQUE constraint.

**➤ Create Composite (Multi-Column) Index**

CREATE INDEX idx\_emp\_name\_dept

ON Employee(Name, DepartmentID);

* Used when queries use **multiple columns** in WHERE.

**➤ Drop Index**

DROP INDEX idx\_employee\_email;

**➤ Show Indexes**

SHOW INDEX FROM Employee;

**🧩 5. Types of Indexes**

| **Type** | **Description** | **Example** |
| --- | --- | --- |
| **1. Single-Column Index** | Index on one column | CREATE INDEX idx\_name ON Emp(Name); |
| **2. Composite Index** | Index on multiple columns | CREATE INDEX idx\_name\_dept ON Emp(Name, DeptID); |
| **3. Unique Index** | Prevents duplicates | CREATE UNIQUE INDEX idx\_email ON Emp(Email); |
| **4. Primary Key Index** | Automatically created when you define PRIMARY KEY | PRIMARY KEY(ID) |
| **5. Clustered Index** | Physically rearranges table data in sorted order | Each table can have **only one** |
| **6. Non-Clustered Index** | Creates a separate index structure pointing to table rows | You can have **many** |
| **7. Full-Text Index** | For searching long text (e.g., articles, documents) | CREATE FULLTEXT INDEX idx\_text ON Posts(Content); |
| **8. Bitmap Index** (Oracle) | For low-cardinality columns (like Gender, Yes/No) | Efficient for analytics |
| **9. Spatial Index** | For geographic (GIS) data | Used in mapping applications |

**🧭 6. Clustered vs Non-Clustered Index**

| **Feature** | **Clustered** | **Non-Clustered** |
| --- | --- | --- |
| Storage | Sorts & stores rows physically | Separate structure points to table |
| Number allowed | Only one per table | Many allowed |
| Speed | Faster for range queries | Slower compared to clustered |
| Example | Primary Key | Secondary column indexes |

**Example:**

-- Clustered Index (by default for PK)

CREATE TABLE Employee(

EmpID INT PRIMARY KEY,

Name VARCHAR(50),

DeptID INT

);

-- Non-clustered Index

CREATE INDEX idx\_emp\_dept

ON Employee(DeptID);

**⚡ 7. When to Create Indexes (Best Practices)**

✅ **Good candidates:**

* Columns used frequently in WHERE
* Columns used in JOIN
* Columns used in ORDER BY or GROUP BY
* Columns with high **cardinality** (many unique values)

❌ **Avoid on:**

* Small tables
* Columns with few distinct values (Gender, Status)
* Columns frequently updated

**🧮 8. Query Performance Example**

-- Without Index

SELECT \* FROM Employee WHERE DepartmentID = 10;

-- Scans all rows.

-- With Index

CREATE INDEX idx\_dept ON Employee(DepartmentID);

-- Only scans matching index entries.

⏱️ Result:  
Query execution time reduces dramatically for large tables.

**🔍 9. Index and Query Execution Plan**

You can **check if an index is used**:

EXPLAIN SELECT \* FROM Employee WHERE DeptID = 10;

Output shows:

* **Using index** → Good
* **Using where** → Full table scan

**🧰 10. Maintenance and Management**

* **Rebuild Index (after heavy updates):**
* ALTER INDEX idx\_name REBUILD;
* **Disable Index:**
* ALTER INDEX idx\_name DISABLE;
* **Drop Unused Indexes:** They waste disk space and slow writes.

**🧠 11. Advanced Indexing Concepts**

**1. Covering Index**

All columns needed by the query are in the index itself — no table access required.

CREATE INDEX idx\_emp\_cover ON Employee(DeptID, Salary);

-- Query: SELECT DeptID, Salary FROM Employee WHERE DeptID=10;

**2. Filtered / Partial Index**

Index only specific rows (e.g., active records).

CREATE INDEX idx\_active\_emp ON Employee(Status)

WHERE Status = 'Active';

**3. Descending Index**

For queries using ORDER BY column DESC

CREATE INDEX idx\_salary\_desc ON Employee(Salary DESC);

**4. Function-based Index**

Stores the result of an expression.

CREATE INDEX idx\_lower\_name ON Employee(LOWER(Name));

**💬 12. Interview Questions (with Answers)**

| **#** | **Question** | **Answer** |
| --- | --- | --- |
| 1 | What is an index? | A data structure that speeds up data retrieval. |
| 2 | Difference between clustered and non-clustered index? | Clustered rearranges physical data, non-clustered creates a separate index structure. |
| 3 | How many clustered indexes per table? | Only one. |
| 4 | Does creating index improve all queries? | No — mostly read queries. It slows down writes. |
| 5 | Can index be created on multiple columns? | Yes, called composite index. |
| 6 | What happens if a column has many duplicate values? | Index is not effective. |
| 7 | Does primary key automatically create an index? | Yes, a clustered index (by default). |
| 8 | What’s a covering index? | Index that contains all columns needed by a query. |
| 9 | Can we index NULL values? | Yes, most DBMSs allow it (though MySQL ignores NULL in some cases). |
| 10 | How to see all indexes of a table? | SHOW INDEX FROM table\_name; |

**🎯 Summary Mind Map**

Indexes

│

├── Purpose → Speed up queries

├── Types

│ ├─ Clustered

│ ├─ Non-clustered

│ ├─ Unique

│ ├─ Composite

│ ├─ Full-text

│ ├─ Bitmap

│ └─ Spatial

├── Syntax → CREATE / DROP INDEX

├── Use cases → WHERE, JOIN, ORDER BY

├── Drawbacks → Slower write ops, more space

└── Tools → EXPLAIN, ANALYZE for checking