# ****Types of Indexes in SQL – With Examples****

Indexes help databases retrieve data efficiently by avoiding full table scans. Below are common types of indexes with use cases and SQL examples.

## ****1. Single-Column Index****

**Definition:** An index created on a single column.

sql

CopyEdit

CREATE INDEX idx\_employee\_name ON Employees(Name);

**Use:**  
Speeds up:

sql

CopyEdit

SELECT \* FROM Employees WHERE Name = 'Alice';

## ****2. Composite Index (Multi-Column Index)****

**Definition:** An index on two or more columns.

sql

CopyEdit

CREATE INDEX idx\_emp\_dept ON Employees(DepartmentID, Name);

**Use:**  
Optimized for:

sql

CopyEdit

SELECT \* FROM Employees WHERE DepartmentID = 2 AND Name = 'Bob';

⚠️ Order matters: It will not work efficiently for just WHERE Name = 'Bob'.

## ****3. Unique Index****

**Definition:** Ensures values in a column (or combination) are unique.

sql

CopyEdit

CREATE UNIQUE INDEX idx\_email\_unique ON Employees(Email);

**Use:** Prevents duplicate emails.

## ****4. Primary Key Index****

**Definition:** Automatically created on a PRIMARY KEY.

sql

CopyEdit

CREATE TABLE Departments (

DeptID INT PRIMARY KEY,

Name VARCHAR(100)

);

**Use:** Enforces uniqueness and improves row lookups.

## ****5. Clustered Index****

**Definition:** Sorts and stores the table rows based on the key.  
Only **one** per table.

sql

CopyEdit

CREATE CLUSTERED INDEX idx\_salary ON Employees(Salary);

**Use:** Efficient for range queries like:

sql

CopyEdit

SELECT \* FROM Employees WHERE Salary BETWEEN 50000 AND 100000;

## ****6. Non-Clustered Index****

**Definition:** Stores a pointer to the actual data row.  
You can have **many** non-clustered indexes per table.

sql

CopyEdit

CREATE NONCLUSTERED INDEX idx\_jobtitle ON Employees(JobTitle);

**Use:** Optimized lookups for specific columns.

## ****7. Full-Text Index****

**Definition:** Enables searching within text fields.

sql

CopyEdit

CREATE FULLTEXT INDEX idx\_description ON Products(Description);

**Use:**

sql

CopyEdit

SELECT \* FROM Products WHERE MATCH(Description) AGAINST ('laptop');

## ****8. Filtered Index**** (SQL Server only)

**Definition:** Indexes only rows that meet a condition.

sql

CopyEdit

CREATE INDEX idx\_active\_employees ON Employees(Status)

WHERE Status = 'Active';

**Use:** Faster access to filtered datasets.

## ****9. Bitmap Index**** (Oracle)

**Definition:** Efficient for columns with few distinct values (e.g., Gender).

sql

CopyEdit

-- Oracle syntax

CREATE BITMAP INDEX idx\_gender ON Employees(Gender);

**Use:** Low-cardinality values (e.g., Gender, Yes/No flags).

## ****📌 Summary Table****

| Index Type | Use Case | Example Column(s) |
| --- | --- | --- |
| Single-Column | Basic search | Name |
| Composite | Combined filters | DepartmentID, Name |
| Unique | Enforce uniqueness | Email |
| Primary Key | Row identification | ID |
| Clustered | Range queries / sorting | Salary |
| Non-Clustered | General purpose lookups | JobTitle |
| Full-Text | Word-based search in large text fields | Description |
| Filtered | Conditional filtering | Status = 'Active' |
| Bitmap | Low-cardinality columns (Oracle) | Gender, MaritalStatus |