1. **RDBMS**

RDBMS stands for **R**elational **D**atabase **M**anagement **S**ystem. RDBMS is the basis for SQL, and for all modern database systems like MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.

A Relational database management system (RDBMS) is a database management system (DBMS) that is based on the relational model as introduced by E. F. Codd.

|  |  |  |
| --- | --- | --- |
| Database Name | Company Name | License or Open Source |
| Oracle | Oracle | License |
| IBM | DB2 | License |
| Microsoft | MS SQL Server, Microsoft Access | License |
| My SQL | My SQL | Open Source |
| Postgres | Postgres | Open Source |

**Note: - Now we are using Postgres Database for all below SQL execution**

1. **TABLE CREATE, INSERT, UPDATE AND DELETE SQL**

-----------------------------------------------------------------

--COURSE TABLE SQL

-----------------------------------------------------------------

--DROP TABLE COURSE;

SELECT \* FROM COURSE ORDER BY COURSE\_ID ASC;

CREATE TABLE COURSE

(

COURSE\_ID INTEGER PRIMARY KEY,

COURSE\_NAME TEXT UNIQUE NOT NULL

);

INSERT INTO COURSE VALUES(1,'B-TECH');

INSERT INTO COURSE VALUES(2, 'MCA');

INSERT INTO COURSE VALUES(3, 'MBA');

INSERT INTO COURSE VALUES(4, 'BCA');

INSERT INTO COURSE VALUES(5, 'BBA');

UPDATE COURSE SET COURSE\_NAME ='BTECH' WHERE COURSE\_ID=1

DELETE FROM COURSE WHERE COURSE\_ID=4;

-------------------------------------------------------------------------

-- BRANCH TABLE SQL

-------------------------------------------------------------------------

DROP TABLE BRANCH

CREATE TABLE BRANCH

(

BRANCH\_ID INTEGER PRIMARY KEY,

BRANCH\_NAME TEXT UNIQUE NOT NULL,

COURSE\_ID INTEGER NOT NULL

);

SELECT \* FROM BRANCH

INSERT INTO BRANCH VALUES(1, 'CS',1);

INSERT INTO BRANCH VALUES(2, 'IT',1);

INSERT INTO BRANCH VALUES(3, 'EC',1);

INSERT INTO BRANCH VALUES(4, 'ME',1);

INSERT INTO BRANCH VALUES(5, 'FINANCE',3);

INSERT INTO BRANCH VALUES(6, 'HR',3);

--INSERT INTO BRANCH VALUES(7, 'IT',3);

INSERT INTO BRANCH VALUES(7, 'MARKETING',3);

DELETE FROM BRANCH

----------------------------------------------------------------------------

--STUDENT TABLE SQL

---------------------------------------------------------------------------

DROP TABLE STUDENT

CREATE TABLE STUDENT

(

STU\_ID INTEGER PRIMARY KEY,

STU\_NAME TEXT NOT NULL,

STU\_COURSE INTEGER NOT NULL REFERENCES COURSE(COURSE\_ID),

STU\_BRANCH INTEGER REFERENCES BRANCH(BRANCH\_ID),

STU\_MOBILE BIGINT NOT NULL UNIQUE,

STU\_EMAIL TEXT NOT NULL UNIQUE

);

DELETE FROM STUDENT

INSERT INTO STUDENT VALUES(1,'SURYANSH TYAGI',1,1,8278443333,'abc@gmail.com');

INSERT INTO STUDENT VALUES(2,'PRANAV TYAGI',1,2,9999222200,'game@gmail.com');

INSERT INTO STUDENT VALUES(3,'ABHISHEK TYAGI',1,1,8278443334,'XXX@gmail.com');

INSERT INTO STUDENT VALUES(4,'ARNAV GUPTA',3,6,8899887766,'XXY@gmail.com');

INSERT INTO STUDENT VALUES(5,'VISHAL TYAGI',3,NULL,9910455253,'XZY@gmail.com');

SELECT \* FROM STUDENT

SELECT \* FROM STUDENT WHERE STU\_NAME LIKE '%SUR%';

SELECT \* FROM STUDENT WHERE STU\_NAME = 'PRANAV TYAGI';

1. **SQL Constraints**

* NOT NULL
* UNIQUE
* PRIMARY KEY
* FOREIGN KEY
* CHECK
* DEFAULT

1. **Key Constraint in RDBMS**

* Super Key
* Primary Key
* Candidate Key
* Alternate Key
* Foreign Key
* Compound Key
* Composite Key

## Primary Key- A column or group of columns in a table which helps us to uniquely identifies every row in that table is called a primary key. This DBMS can't be a duplicate. The same value can't appear more than once in the table.

### Rules for defining Primary key:

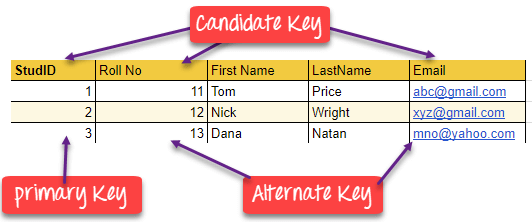
* Only one primary key can have on one table.
* primary key is combination of UNIQUE and NOT NULL constraint.
* The primary key field should have non duplicate, not null and unique value.
* The value in a primary key column can never be modified or updated if any foreign key refers to that primary key.

## Alternate key- All the keys which not primary key are called an alternate key. It is a candidate key which is currently not the primary key.

## Candidate Key-A super key with no repeated attribute is called candidate key. The Primary key should be selected from the candidate keys. Every table must have at least a single candidate key.

**Properties of Candidate key:**

* It must contain unique values
* Candidate key may have multiple attributes
* Must not contain null values
* It should contain minimum fields to ensure uniqueness
* Uniquely identify each record in a table

[](https://www.guru99.com/images/1/100518_0517_DBMSKeysPri1.png)

## What is the Foreign key?

A foreign key is a column which is added to create a relationship with another table. Foreign keys help us to maintain data integrity and also allows navigation between two different instances of an entity. Every relationship in the model needs to be supported by a foreign key.

This concept is also known as Referential Integrity.

## Compound key- Compound key has many fields which allow you to uniquely recognize a specific record. It is possible that each column may be not unique by itself within the database. However, when combined with the other column or columns the combination of composite keys become unique.

## What is the Composite key?

A key which has multiple attributes to uniquely identify rows in a table is called a composite key. The difference between compound and the composite key is that any part of the compound key can be a foreign key, but the composite key may or maybe not a part of the foreign key.

## Difference Between Primary key & Foreign key

|  |  |
| --- | --- |
| **Primary Key** | **Foreign Key** |
| Helps you to uniquely identify a record in the table. | It is a field in the table that is the primary key of another table. |
| Primary Key never accept null values. | A foreign key may accept multiple null values. |
| Primary key is a clustered index and data in the DBMS table are physically organized in the sequence of the clustered index. | A foreign key cannot automatically create an index, clustered or non-clustered. However, you can manually create an index on the foreign key. |
| You can have the single Primary key in a table. | You can have multiple foreign keys in a table. |

1. **Joins:-**

* INNER JOIN OR JOIN
* LEFT JOIN OR LEFT OUTER JOIN
* RIGHT JOIN OR RIGHT OUTER JOIN
* FULL JOIN
* CROSS JOIN
* SELF JOIN

**INNER JOIN:** The INNER JOIN keyword selects all rows from both the tables as long as the condition satisfies. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be same.  
**Syntax**:

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

INNER JOIN table2

ON table1.matching\_column = table2.matching\_column;

**LEFT JOIN**: This join returns all the rows of the table on the left side of the join and matching rows for the table on the right side of join. The rows for which there is no matching row on right side, the result-set will contain *null*. LEFT JOIN is also known as LEFT OUTER JOIN.**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

LEFT JOIN table2

ON table1.matching\_column = table2.matching\_column;

**RIGHT JOIN**: RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of join. The rows for which there is no matching row on left side, the result-set will contain *null*. RIGHT JOIN is also known as RIGHT OUTER JOIN.**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

RIGHT JOIN table2

ON table1.matching\_column = table2.matching\_column;

**FULL JOIN:** FULL JOIN creates the result-set by combining result of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both the tables. The rows for which there is no matching, the result-set will contain *NULL* values.**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

FULL JOIN table2

ON table1.matching\_column = table2.matching\_column;

## **Cross JOIN or Cartesian Product**

This type of JOIN returns the cartesian product of rows from the tables in Join. It will return a table which consists of records which combines each row from the first table with each row of the second table.

**SELF JOIN**

create table emp

(

id integer,

name text,

m\_id integer

);

select \* from emp

insert into emp values(1,'A',2);

insert into emp values(2,'B',2);

insert into emp values(3,'C',null);

insert into emp values(4,'D',5);

insert into emp values(5,'E',null);

select e1.id, e2.id, e1.name,e2.name from emp e1 join emp e2 on e1.m\_id = e2.id

order by e1.id

## What is Indexing?

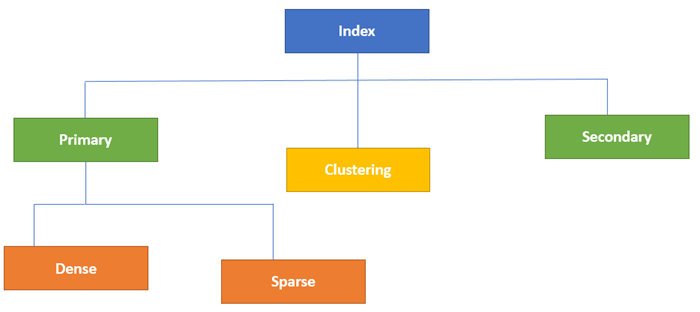
Indexing is defined as a data structure technique which allows you to quickly retrieve records from a database file. It is based on the same attributes on which the Indices has been done.

An index

* Takes a search key as input
* Efficiently returns a collection of matching records.

An Index is a small table having only two columns. The first column comprises a copy of the primary or candidate key of a table. Its second column contains a set of pointers for holding the address of the disk block where that specific key value stored.

## Types of Indexing

[](https://www.guru99.com/images/1/070119_0833_IndexinginD1.png)Type of Indexes

Database Indexing is defined based on its indexing attributes. Two main types of indexing methods are:

* Primary Indexing
* Secondary Indexing

## Primary Indexing

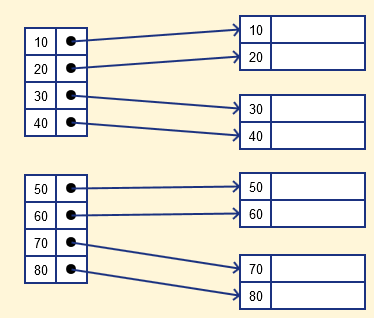
Primary Index is an ordered file which is fixed length size with two fields. The first field is the same a primary key and second, filed is pointed to that specific data block. In the primary Index, there is always one to one relationship between the entries in the index table.

The primary Indexing is also further divided into two types.

* Dense Index
* Sparse Index

### Dense Index

In a dense index, a record is created for every search key valued in the database. This helps you to search faster but needs more space to store index records. In this Indexing, method records contain search key value and points to the real record on the disk.

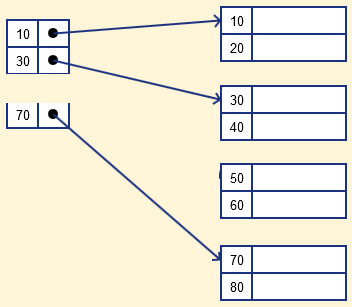
[](https://www.guru99.com/images/1/070119_0833_IndexinginD2.png)

### Sparse Index

It is an index record that appears for only some of the values in the file. Sparse Index helps you to resolve the issues of dense Indexing. In this method of indexing technique, a range of index columns stores the same data block address, and when data needs to be retrieved, the block address will be fetched.

However, sparse Index stores index records for only some search-key values. It needs less space, less maintenance overhead for insertion, and deletions but It is slower compared to the dense Index for locating records.

Example of Sparse Index

[](https://www.guru99.com/images/1/070119_0833_IndexinginD3.png)

## Secondary Index or Non-Clustering Index

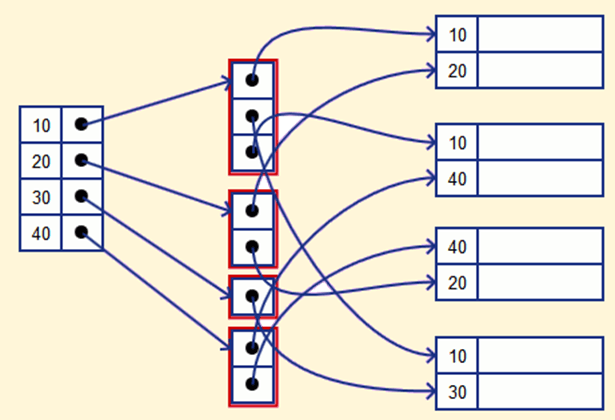
The secondary Index can be generated by a field which has a unique value for each record, and it should be a candidate key. It is also known as a non-clustering index.

This two-level database indexing technique is used to reduce the mapping size of the first level. For the first level, a large range of numbers is selected because of this; the mapping size always remains small.

### Example of secondary Indexing

In a bank account database, data is stored sequentially by acc\_no; you may want to find all accounts in of a specific branch of ABC bank.

Here, you can have a secondary index for every search-key. Index record is a record point to a bucket that contains pointers to all the records with their specific search-key value.

[](https://www.guru99.com/images/1/070119_0833_IndexinginD4.png)

## Clustering Index

In a clustered index, records themselves are stored in the Index and not pointers. Sometimes the Index is created on non-primary key columns which might not be unique for each record. In such a situation, you can group two or more columns to get the unique values and create an index which is called clustered Index. This also helps you to identify the record faster.

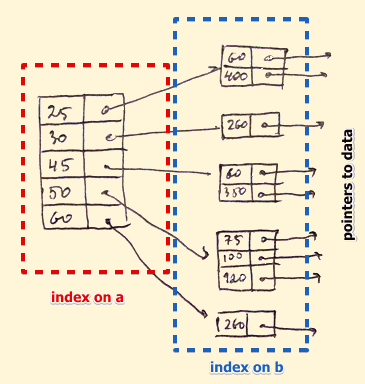
**Example:**

Let's assume that a company recruited many employees in various departments. In this case, clustering indexing should be created for all employees who belong to the same dept.

It is considered in a single cluster, and index points point to the cluster as a whole. Here, Department \_no is a non-unique key.

## What is Multilevel Index?

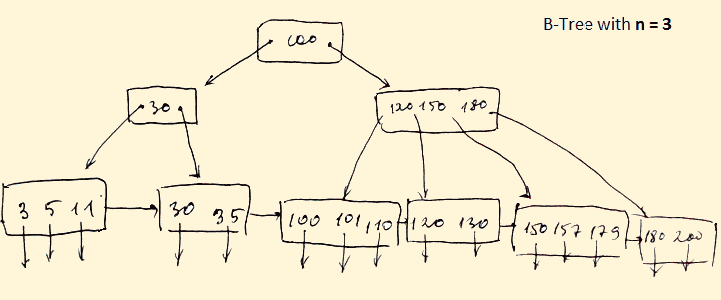
Multilevel Indexing is created when a primary index does not fit in memory. In this type of indexing method, you can reduce the number of disk accesses to short any record and kept on a disk as a sequential file and create a sparse base on that file.

[](https://www.guru99.com/images/1/070119_0833_IndexinginD5.png)

## B-Tree Index

B-tree index is the widely used data structures for Indexing. It is a multilevel index format technique which is balanced binary search trees. All leaf nodes of the B tree signify actual data pointers.

Moreover, all leaf nodes are interlinked with a link list, which allows a B tree to support both random and sequential access.

[](https://www.guru99.com/images/1/070119_0833_IndexinginD6.png)

* Lead nodes must have between 2 and 4 values.
* Every path from the root to leaf are mostly on an equal length.
* Non-leaf nodes apart from the root node have between 3 and 5 children nodes.
* Every node which is not a root or a leaf has between n/2] and n children.

## Advantages of Indexing

Important pros/ advantage of Indexing are:

* It helps you to reduce the total number of I/O operations needed to retrieve that data, so you don't need to access a row in the database from an index structure.
* Offers Faster search and retrieval of data to users.
* Indexing also helps you to reduce tablespace as you don't need to link to a row in a table, as there is no need to store the ROWID in the Index. Thus you will able to reduce the tablespace.
* You can't sort data in the lead nodes as the value of the primary key classifies it.

## Disadvantages of Indexing

Important drawbacks/cons of Indexing are:

* To perform the indexing database management system, you need a primary key on the table with a unique value.
* You can't perform any other indexes on the Indexed data.
* You are not allowed to partition an index-organized table.
* SQL Indexing Decrease performance in INSERT, DELETE, and UPDATE query.

### Summary:

* Indexing is a small table which is consist of two columns.
* Two main types of indexing methods are 1)Primary Indexing 2) Secondary Indexing.
* Primary Index is an ordered file which is fixed length size with two fields.
* The primary Indexing is also further divided into two types 1)Dense Index 2)Sparse Index.
* In a dense index, a record is created for every search key valued in the database.
* A sparse indexing method helps you to resolve the issues of dense Indexing.
* The secondary Index is an indexing method whose search key specifies an order different from the sequential order of the file.
* Clustering index is defined as an order data file.
* Multilevel Indexing is created when a primary index does not fit in memory.
* The biggest benefit of Indexing is that it helps you to reduce the total number of I/O operations needed to retrieve that data.
* The biggest drawback to performing the indexing database management system, you need a primary key on the table with a unique value.

