**Database**

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a DBMS. Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

A database is a systematic collection of data. They support electronic storage and manipulation of data. Databases make data management easy.

Let us discuss a database example: An online telephone directory uses a database to store data of people, phone numbers, and other contact details. Your electricity service provider uses a database to manage billing, client-related issues, handle fault data, etc.

Let us also consider Facebook. It needs to store, manipulate, and present data related to members, their friends, member activities, messages, advertisements, and a lot more. We can provide a countless number of examples for the usage of databases.

Types of Databases:

* Centralized database
* Distributed database
* Personal database
* End-user database
* Commercial database
* NoSQL database
* Operational database
* Relational database
* Cloud database
* Object-oriented database
* Graph database

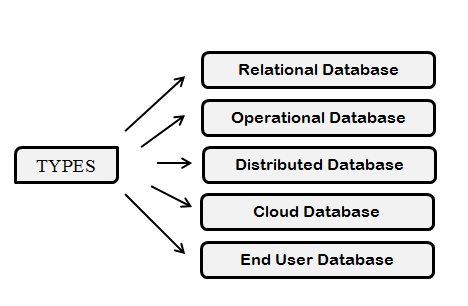
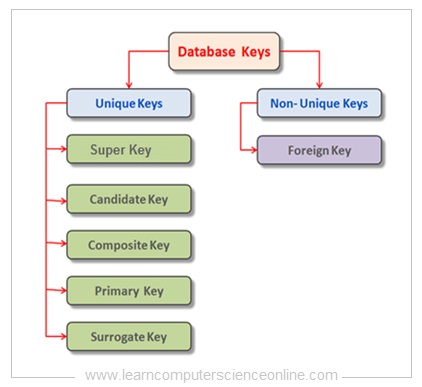
SQL is a programming language used by nearly all relational database to query, manipulate, and define data, and to provide access control. SQL was first developed at IBM in the 1970s with Oracle as a major contributor, which led to implementation of the SQL ANSI standard, SQL has spurred many extensions from companies such as IBM, Oracle, and Microsoft. Although SQL is still widely used today, new programming languages are beginning to appear.

A database typically requires a comprehensive database software program known as a database management system (DBMS). A DBMS serves as an interface between the database and its end users or programs, allowing users to retrieve, update, and manage how the information is organized and optimized. A DBMS also facilitates oversight and control of databases, enabling a variety of administrative operations such as performance monitoring, tuning, and backup and recovery.

Some examples of popular database software or DBMSs include MySQL, Microsoft Access, Microsoft SQL Server, FileMaker Pro, Oracle Database, and dBase.

**KEYS in DBMS** is an attribute or set of attributes which helps you to identify a row(tuple) in a relation(table). They allow you to find the relation between two tables. Keys help you uniquely identify a row in a table by a combination of one or more columns in that table. Key is also helpful for finding unique record or row from the table. Database key is also helpful for finding unique record or row from the table.

some reasons for using SQL key in the DBMS system.

* Keys help you to identify any row of data in a table. In a real-world application, a table could contain thousands of records. Moreover, the records could be duplicated. Keys in RDBMS ensure that you can uniquely identify a table record despite these challenges.
* Allows you to establish a relationship between and identify the relation between tables
* Help you to enforce identity and integrity in the relationship.
* **Super Key –**A super key is a group of single or multiple keys which identifies rows in a table.
* **Primary Key –**is a column or group of columns in a table that uniquely identify every row in that table.
* **Candidate Key –**is a set of attributes that uniquely identify tuples in a table. Candidate Key is a super key with no repeated attributes.

### 

### Database Table Relationships

One of the huge advantages of a relational database is that, once you have your data held in clearly defined, compact tables, you can connect or relate the data held in different tables. There are three types of relationships between the data you are likely to encounter at this stage in the design: one-to-one, one-to-many, and many-to-many. To be able to identify these relationships, you need to examine the data and have an understanding of what business rules apply to the data and tables. If you're not sure, it can be helpful to meet with someone who does have a thorough knowledge of the data.-

**One-to-one**

A one-to-one (1:1) relationship means that each record in Table A relates to one, and only one, record in Table B, and each record in Table B relates to one, and only one, record in Table A.

Both tables can have only one record on each side of the relationship.

Each primary key value relates to none or only one record in the related table.

Most one-to-one relationships are forced by business rules and do not flow naturally from the data. Without such a rule, you can typically combine both tables without breaking any normalization rules.

**one-to-many**

The primary key table contains only one record that relates to none, one, or many records in the related table.

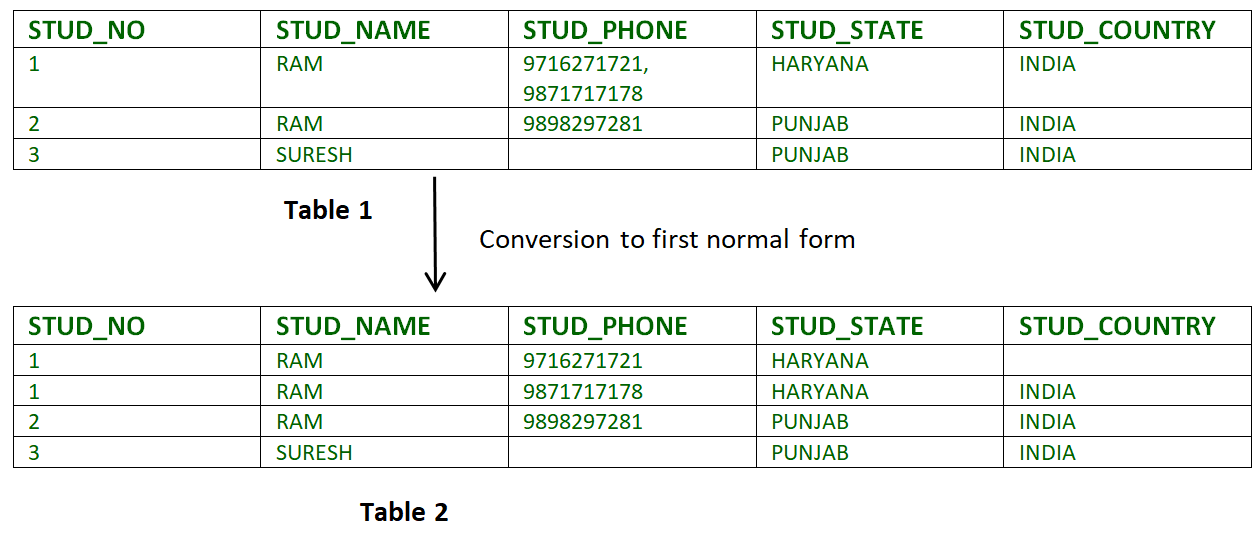
In a one-to-many relationship, the table on the one side of the relationship is the primary table and the table on the many side is the related table.

**many-to-many**

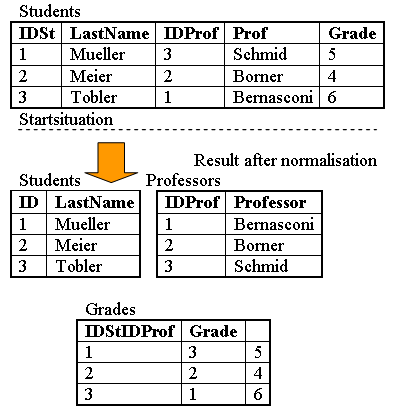
Each record in both tables can relate to none or any number of records in the other table. These relationships require a third table, called an associate or linking table, because relational systems cannot directly accommodate the relationship.

**First Normal Form (1NF)**

to handle and update the database, without facing data loss. It will also eat up extra memory space and Insertion, Update and Deletion Anomalies are very frequent if database is not normalized.

normalized. is the process of minimizing redundancy from a relation or set of relations Redundancy in relation may cause insertion, deletion and update anomalies, So, it helps to minimize the redundancy in relations, Normal forms are used to eliminate or reduce redundancy in database tables.

**Second Normal Form (2NF):**

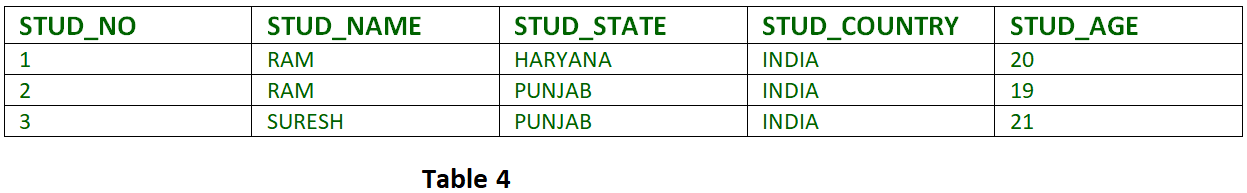
Second Normal Form (2NF) is based on the concept of full functional dependency. Second Normal Form applies to relations with composite keys, that is, relations with a primary key composed of two or more attributes. A relation with a single-attribute primary key is automatically in at least 2NF. A relation that is not in 2NF may suffer from the update anomalies.

To be in second normal form, a relation must be in first normal form and relation must not contain any partial dependency. A relation is in 2NF if it has No Partial Dependency, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

**Third Normal Form (3NF):**  
A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.

A relation is in 3NF if at least one of the following condition holds in every non-trivial function dependency X –> Y:

1. X is a super key.
2. Y is a prime attribute (each element of Y is part of some candidate key).



FD set:  
{STUD\_NO -> STUD\_NAME, STUD\_NO -> STUD\_STATE, STUD\_STATE -> STUD\_COUNTRY, STUD\_NO -> STUD\_AGE}

Candidate Key:  
{STUD\_NO}

For this relation in table 4, STUD\_NO -> STUD\_STATE and STUD\_STATE -> STUD\_COUNTRY are true. so STUD\_COUNTRY is transitively dependent on STUD\_NO. It violates the third normal form. To convert it in third normal form, we will decompose the relation STUDENT (STUD\_NO, STUD\_NAME, STUD\_PHONE, STUD\_STATE, STUD\_COUNTRY\_STUD\_AGE) .

**Querying a Database**

Queries are one of the things that make databases so powerful. A "query" refers to the action of retrieving data from your database. Usually, you will be selective with how much data you want returned. If you have a lot of data in your database, you probably don't want to see everything. More likely, you'll only want to see data that fits a certain Criteria.

For example, you might only want to see how many individuals in your database live in a given city. Or you might only want to see which individuals have registered with your database within a given time period.