## *UNIT-1*

## What is Database

The database is a collection of inter-related data which is used to retrieve, insert and delete the data efficiently. It is also used to organize the data in the form of a table, schema, views, and reports, etc.

**For example:** The college Database organizes the data about the admin, staff, students and faculty etc.

Using the database, you can easily retrieve, insert, and delete the information.

## Database Management System

* Database management system is a software which is used to manage the database. For example: [MySQL](https://www.javatpoint.com/mysql-tutorial), [Oracle](https://www.javatpoint.com/oracle-tutorial), etc are a very popular commercial database which is used in different applications.
* DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.
* It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.

**DBMS allows users the following tasks:**

* **Data Definition:** It is used for creation, modification, and removal of definition that defines the organization of data in the database.
* **Data Updation:** It is used for the insertion, modification, and deletion of the actual data in the database.
* **Data Retrieval:** It is used to retrieve the data from the database which can be used by applications for various purposes.
* **User Administration:** It is used for registering and monitoring users, maintain data integrity, enforcing data security, dealing with concurrency control, monitoring performance and recovering information corrupted by unexpected failure.

## Characteristics of DBMS

* It uses a digital repository established on a server to store and manage the information.
* It can provide a clear and logical view of the process that manipulates data.
* DBMS contains automatic backup and recovery procedures.
* It contains ACID properties which maintain data in a healthy state in case of failure.
* It can reduce the complex relationship between data.
* It is used to support manipulation and processing of data.
* It is used to provide security of data.
* It can view the database from different viewpoints according to the requirements of the user.

## Advantages of DBMS

* **Controls database redundancy:** It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database.
* **Data sharing:** In DBMS, the authorized users of an organization can share the data among multiple users.
* **Easily Maintenance:** It can be easily maintainable due to the centralized nature of the database system.
* **Reduce time:** It reduces development time and maintenance need.
* **Backup:** It provides backup and recovery subsystems which create automatic backup of data from [hardware](https://www.javatpoint.com/hardware) and [software](https://www.javatpoint.com/software) failures and restores the data if required.
* **multiple user interface:** It provides different types of user interfaces like graphical user interfaces, application program interfaces

## Disadvantages of DBMS

* **Cost of Hardware and Software:** It requires a high speed of data processor and large memory size to run DBMS software.
* **Size:** It occupies a large space of disks and large memory to run them efficiently.
* **Complexity:** Database system creates additional complexity and requirements.
* **Higher impact of failure:** Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.

**Purpose of Database of management system:**

The Database Management System (DBMS) is defined as a software system that allows the user to define, create and maintain the database and provide control access to the data.

It is a collection of programs used for managing data and simultaneously it supports different types of users to create, manage, retrieve, update and store information.

**Purpose**

The purpose of DBMS is to transform the following −

* Data into information.
* Information into knowledge.
* Knowledge to the action.

The diagram given below explains the process as to how the transformation of data to information to knowledge to action happens respectively in the DBMS −



Previously, the database applications were built directly on top of the file system.

**Drawbacks in File System**

There are so many drawbacks in using the file system. These are mentioned below −

* Data redundancy and inconsistency: Different file formats, duplication of information in different files.
* Difficulty in accessing data: To carry out new task we need to write a new program.
* Data Isolation − Different files and formats.
* Integrity problems.
* Atomicity of updates − Failures leave the database in an inconsistent state. For example, the fund transfer from one account to another may be incomplete.
* Concurrent access by multiple users.
* Security problems.

Database system offer so many solutions to all these problems

**Uses of DBMS**

The main uses of DBMS are as follows −

* Data independence and efficient access of data.
* Application Development time reduces.
* Security and data integrity.
* Uniform data administration.
* Concurrent access and recovery from crashes.

**Applications of DBMS**

The different applications of DBMS are as follows −

* **Railway Reservation System** − It is used to keep record of booking of tickets, departure of the train and the status of arrival and give updates to the passengers with the help of a database.
* **Library Management System** − There will be so many numbers of books in the library and it is very hard to keep a record of all the books in a register or a copy. So, DBMS is necessary to keep track of all the book records, issue dates, name of the books, author and maintain the records.
* **Banking** − We are doing a lot of transactions daily without directly going to the banks. The only reason is the usage of databases and it manages all the data of the customers over the database.
* **Educational Institutions** − All the examinations and the data related to the students maintained over the internet with the help of a database management system. It contains registration details of the student, results, grades and courses available. All these works can be done online without visiting an institution.
* **Social Media Websites** − By filling the required details we are able to access social media platforms. Many users daily sign up for social websites such as Facebook, Pinterest and Instagram. All the information related to the users are stored and maintained with the help of DBMS.

**Data redundancy:**

**Data redundancy occurs when the same piece of data exists in multiple places, whereas data inconsistency is when the same data exists in different formats in multiple tables**. Unfortunately, data redundancy can cause data inconsistency, which can provide a company with unreliable and/or meaningless information.

What is data inconsistency?

Data inconsistency is **a situation where there are multiple tables within a database that deal with the same data but may receive it from different inputs**. Inconsistency is generally compounded by data redundancy.

**View of Data:**

Database systems are made-up of complex data structures. To ease the user interaction with database, the developers hide internal irrelevant details from users. This process of hiding irrelevant details from user is called **data abstraction**. The term “irrelevant” used here with respect to the user, it doesn’t mean that the hidden data is not relevant with regard to the whole database. It just means that the **user is not concerned about that data**.

**For example:** When you are booking a train ticket, you are not concerned how data is processing at the back end when you click “book ticket”, what processes are happening when you are doing online payments. **You are just concerned about the message that pops up when your ticket is successfully booked**. This doesn’t mean that the process happening at the back end is not relevant, it just means that you as a user are not concerned what is happening in the database.

## Three levels of abstraction


**Physical level**: This is the lowest level of data abstraction. It describes how data is actually stored in database. You can get the complex data structure details at this level.

**Logical level**: This is the middle level of 3-level data abstraction architecture. It describes what data is stored in database.

**View level**: Highest level of data abstraction. This level describes the user interaction with database system.

**Example**: Let’s say we are storing customer information in a customer table. At **physical level** these records can be described as blocks of storage (bytes, gigabytes, terabytes etc.) in memory. These details are often hidden from the programmers.

At the **logical level** these records can be described as fields and attributes along with their data types, their relationship among each other can be logically implemented. The programmers generally work at this level because they are aware of such things about database systems.

At **view level**, user just interact with system with the help of GUI and enter the details at the screen, they are not aware of how the data is stored and what data is stored; such details are hidden from them.

 **Types of Data Model**

For the creation of any database, the data model is considered a logical structure for creating a database. The data model includes entities, attributes, constraints, relationships, etc. The data models are used to represent the data and how it is stored in the database, how data is accessible and updated in the database management system. There are four types of data models: Hierarchical model, Network model, [Entity-relationship model](https://www.educba.com/entity-relationship-model/), Relational model. These models have further categories which are used according to a different use case.

**Different Types of Data Models**

There are 4 different types of data models:

**1. Hierarchical Model**

In this type of data model, the data is organized into a tree-like structure that has a single root and the data is linked to the root. In this model, the [main hierarchy begins](https://www.educba.com/hierarchy-in-tableau/) from the root and it expands like a tree that has child nodes and further expands in the same manner. In this model the child node has on;e single parent node but one parent can have multiple child nodes. As the data is stored like tree structure in this data model when data is retrieved the whole tree is traversed from the root node. The hierarchical data model contains a one-to-many relationship between various types of data. The data is stored in the form of a record and is connected through links.

**For Example-**there is an organization that has a requirement to store the information of its employees. The table contains the following attributes: employee name, employee code, department name, and last name. And the organization provides a computer for each employee. So there is a requirement for storing information on a computer which is stored in a separate table. The computer table store employee code, serial number, and type. According to the hierarchical data model, the employee table can be considered as a parent table and a computer table can be considered as a child node.

**2. Network model**

The network model is a type of database model which is designed based on a flexible approach for representing objects and the relationship exist among objects. The schema is very important in the network data model which can be represented in the form of a graph where a relationship is represented using edges and the nodes are used to represent objects. The basic difference between the hierarchical model and network model is that data is represented in the form of hierarchy in a hierarchical data model whereas in network model the data is represented in the form of a graph. One of the advantages of a network model is that the basic connections are also represented in this data model. There are different types of relationship can exist in this data model like one to one, many to many, etc. The data access becomes simple to compare [to other data model](https://www.educba.com/data-model-in-cassandra/) like the hierarchical model. The parent node and child node are always connected as there is always a relationship exist among parent-child node. And the data is not dependent on the other node. One of the key drawbacks of this model is that this system is not adaptive toward changes. It means when there is a requirement of some modification of system it requires to change the whole system which takes a lot of effort. And to maintain data is difficult to part in this model as every record is connected via some pointers which makes it difficult to maintain and make the system complex.

**3. E-R model**

The ER model is used to describe the database structure using the entity-relationship diagram. The E-R model is just like the blueprint of a database which is used to implement the database. In the entity set, the relationship exists which can be shown using the ER diagram. The entity set consists of a similar type of entities that consist of attributes.

The components of the ER model are relationship set and entity set and attributes. The entity is the component of data which is represented as a rectangle in the ER diagram. For example, there are two entities college and student and there exist one too many relationships as there can be more than one student who can go to college.

The entity which cannot be identified by attributes and which require a relationship is called a weak entity. For representing a weak entity the entity is represented in a double rectangle. For example, there is a bank account but it cannot relate until the bank name is not known to the bank account is termed as a weak entity.

The attributes are used to represent the property of the entity. In the ER diagram, the attribute is represented as an oval. There are different types of attributes like key attribute, composite attribute, multivalued attribute and derived attribute. For example, a student is an entity and the related attributes for student entity are student name, student age, student roll number, student address, etc.

The relationship is represented in a diamond shape in the ER diagram. The relationships exist among entities. There are multiple types of relationships like one to one, one to many, many to one, and many to many.

#### 4. Relational model

In this data model, the data tables are used to collect a group of elements into the relations. In this model, the relationships and data are represented using interrelated tables. And in the table, there are multiple rows and multiple columns in which column represents the attribute of the entity and the rows are used to represent records. In this data model there exist different primary key which issued to distinguish each record in the table. And for retrieving the data elements the SQL (Structured Query Language) is used. For using the relational data model the primary key issued as the fundamental tool. And for each entry in the data set, it needs to be unique. The data table should not contain any type of inconsistency as it can create a problem at the time of data retrieval. The other problem with the relational data model is data duplicacy, incomplete data and inappropriate links used to connect data.

# DBMS Architecture

* The DBMS design depends upon its architecture. The basic client/server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.
* The client/server architecture consists of many PCs and a workstation which are connected via the network.
* DBMS architecture depends upon how users are connected to the database to get their request done.

## Types of DBMS Architecture



Database architecture can be seen as a single tier or multi-tier. But logically, database architecture is of two types like: **2-tier architecture** and **3-tier architecture**.

### 1-Tier Architecture

* In this architecture, the database is directly available to the user. It means the user can directly sit on the DBMS and uses it.
* Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
* The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.

### 2-Tier Architecture

* The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC**, **JDBC** are used.
* The user interfaces and application programs are run on the client-side.
* The server side is responsible to provide the functionalities like: query processing and transaction management.
* To communicate with the DBMS, client-side application establishes a connection with the server side.



**Fig: 2-tier Architecture**

### 3-Tier Architecture

* The 3-Tier architecture contains another layer between the client and server. In this architecture, client can't directly communicate with the server.
* The application on the client-end interacts with an application server which further communicates with the database system.
* End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.
* The 3-Tier architecture is used in case of large web application.



**Fig: 3-tier Architecture**

# What is RDBMS (Relational Database Management System)

**RDBMS** stands for Relational Database Management System.

All modern database management systems like SQL, MS SQL Server, IBM DB2, ORACLE, My-SQL, and Microsoft Access are based on RDBMS.

It is called Relational Database Management System (RDBMS) because it is based on the relational model introduced by E.F. Codd.

## How it works

Data is represented in terms of tuples (rows) in RDBMS.

A relational database is the most commonly used database. It contains several tables, and each table has its primary key.

Due to a collection of an organized set of tables, data can be accessed easily in RDBMS.

## Brief History of RDBMS

From 1970 to 1972, E.F. Codd published a paper to propose using a relational database model.

RDBMS is originally based on E.F. Codd's relational model invention.

**Following are the various terminologies of RDBMS:**



## What is table/Relation?

Everything in a relational database is stored in the form of relations. The RDBMS database uses tables to store data. A table is a collection of related data entries and contains rows and columns to store data. Each table represents some real-world objects such as person, place, or event about which information is collected. The organized collection of data into a relational table is known as the logical view of the database.

**Properties of a Relation:**

* Each relation has a unique name by which it is identified in the database.
* Relation does not contain duplicate tuples.
* The tuples of a relation have no specific order.
* All attributes in a relation are atomic, i.e., each cell of a relation contains exactly one value.

A table is the simplest example of data stored in RDBMS.

**Let's see the example of the student table.**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **AGE** | **COURSE** |
| 1 | Ajeet | 24 | B.Tech |
| 2 | Aryan | 20 | C.A |
| 3 | Mahesh | 21 | BCA |
| 4 | Ratan | 22 | MCA |
| 5 | Vimal | 26 | BSC |

## What is a row or record?

A row of a table is also called a record or tuple. It contains the specific information of each entry in the table. It is a horizontal entity in the table. For example, The above table contains 5 records.

**Properties of a row:**

* No two tuples are identical to each other in all their entries.
* All tuples of the relation have the same format and the same number of entries.
* The order of the tuple is irrelevant. They are identified by their content, not by their position.

**Let's see one record/row in the table.**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **AGE** | **COURSE** |
| 1 | Ajeet | 24 | B.Tech |

## What is a column/attribute?

A column is a vertical entity in the table which contains all information associated with a specific field in a table. For example, "name" is a column in the above table which contains all information about a student's name.

**Properties of an Attribute:**

* Every attribute of a relation must have a name.
* Null values are permitted for the attributes.
* Default values can be specified for an attribute automatically inserted if no other value is specified for an attribute.
* Attributes that uniquely identify each tuple of a relation are the primary key.

|  |
| --- |
| **Name** |
| Ajeet |
| Aryan |
| Mahesh |
| Ratan |
| Vimal |

## What is data item/Cells?

The smallest unit of data in the table is the individual data item. It is stored at the intersection of tuples and attributes.

**Properties of data items:**

* Data items are atomic.
* The data items for an attribute should be drawn from the same domain.

In the below example, the data item in the student table consists of Ajeet, 24 and Btech, etc.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **AGE** | **COURSE** |
| 1 | Ajeet | 24 | B.Tech |

### Degree:

The total number of attributes that comprise a relation is known as the degree of the table.

**For example, the student table has 4 attributes, and its degree is 4.**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **AGE** | **COURSE** |
| 1 | Ajeet | 24 | B.Tech |
| 2 | Aryan | 20 | C.A |
| 3 | Mahesh | 21 | BCA |
| 4 | Ratan | 22 | MCA |
| 5 | Vimal | 26 | BSC |

### Cardinality:

The total number of tuples at any one time in a relation is known as the table's cardinality. The relation whose cardinality is 0 is called an empty table.

**For example, the student table has 5 rows, and its cardinality is 5.**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **AGE** | **COURSE** |
| 1 | Ajeet | 24 | B.Tech |
| 2 | Aryan | 20 | C.A |
| 3 | Mahesh | 21 | BCA |
| 4 | Ratan | 22 | MCA |
| 5 | Vimal | 26 | BSC |

### Domain:

The domain refers to the possible values each attribute can contain. It can be specified using standard data types such as integers, floating numbers, etc. **For example**, An attribute entitled Marital\_Status may be limited to married or unmarried values.

### NULL Values

The NULL value of the table specifies that the field has been left blank during record creation. It is different from the value filled with zero or a field that contains space.

### Data Integrity

There are the following categories of data integrity exist with each RDBMS:

**Entity integrity**: It specifies that there should be no duplicate rows in a table.

**Domain integrity**: It enforces valid entries for a given column by restricting the type, the format, or the range of values.

**Referential integrity** specifies that rows cannot be deleted, which are used by other records.

**User-defined integrity**: It enforces some specific business rules defined by users. These rules are different from the entity, domain, or referential integrity.

# Relational Model concept

Relational model can represent as a table with columns and rows. Each row is known as a tuple. Each table of the column has a name or attribute.

**Domain:** It contains a set of atomic values that an attribute can take.

**Attribute:** It contains the name of a column in a particular table. Each attribute Ai must have a domain, dom(Ai)

**Relational instance:** In the relational database system, the relational instance is represented by a finite set of tuples. Relation instances do not have duplicate tuples.

**Relational schema:** A relational schema contains the name of the relation and name of all columns or attributes.

**Relational key:** In the relational key, each row has one or more attributes. It can identify the row in the relation uniquely.

**Example: STUDENT Relation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NAME** | **ROLL\_NO** | **PHONE\_NO** | **ADDRESS** | **AGE** |
| Ram | 14795 | 7305758992 | Noida | 24 |
| Shyam | 12839 | 9026288936 | Delhi | 35 |
| Laxman | 33289 | 8583287182 | Gurugram | 20 |
| Mahesh | 27857 | 7086819134 | Ghaziabad | 27 |
| Ganesh | 17282 | 9028 9i3988 | Delhi | 40 |

* In the given table, NAME, ROLL\_NO, PHONE\_NO, ADDRESS, and AGE are the attributes.
* The instance of schema STUDENT has 5 tuples.
* t3 = <Laxman, 33289, 8583287182, Gurugram, 20>

## Properties of Relations

* Name of the relation is distinct from all other relations.
* Each relation cell contains exactly one atomic (single) value
* Each attribute contains a distinct name
* Attribute domain has no significance
* tuple has no duplicate value
* Order of tuple can have a different sequence

# Keys

* Keys play an important role in the relational database.
* It is used to uniquely identify any record or row of data from the table. It is also used to establish and identify relationships between tables.

**For example,** ID is used as a key in the Student table because it is unique for each student. In the PERSON table, passport\_number, license\_number, SSN are keys since they are unique for each person.



## Types of keys:



### 1. Primary key

* It is the first key used to identify one and only one instance of an entity uniquely. An entity can contain multiple keys, as we saw in the PERSON table. The key which is most suitable from those lists becomes a primary key.
* In the EMPLOYEE table, ID can be the primary key since it is unique for each employee. In the EMPLOYEE table, we can even select License\_Number and Passport\_Number as primary keys since they are also unique.
* For each entity, the primary key selection is based on requirements and developers.



### 2. Candidate key

* A candidate key is an attribute or set of attributes that can uniquely identify a tuple.
* Except for the primary key, the remaining attributes are considered a candidate key. The candidate keys are as strong as the primary key.

**For example:** In the EMPLOYEE table, id is best suited for the primary key. The rest of the attributes, like SSN, Passport\_Number, License\_Number, etc., are considered a candidate key.



### 3. Super Key

Super key is an attribute set that can uniquely identify a tuple. A super key is a superset of a candidate key.



**For example:** In the above EMPLOYEE table, for(EMPLOEE\_ID, EMPLOYEE\_NAME), the name of two employees can be the same, but their EMPLYEE\_ID can't be the same. Hence, this combination can also be a key.

The super key would be EMPLOYEE-ID (EMPLOYEE\_ID, EMPLOYEE-NAME), etc.

### 4. Foreign key

* Foreign keys are the column of the table used to point to the primary key of another table.
* Every employee works in a specific department in a company, and employee and department are two different entities. So we can't store the department's information in the employee table. That's why we link these two tables through the primary key of one table.
* We add the primary key of the DEPARTMENT table, Department\_Id, as a new attribute in the EMPLOYEE table.
* In the EMPLOYEE table, Department\_Id is the foreign key, and both the tables are related.



### 5. Alternate key

There may be one or more attributes or a combination of attributes that uniquely identify each tuple in a relation. These attributes or combinations of the attributes are called the candidate keys. One key is chosen as the primary key from these candidate keys, and the remaining candidate key, if it exists, is termed the alternate key. **In other words,** the total number of the alternate keys is the total number of candidate keys minus the primary key. The alternate key may or may not exist. If there is only one candidate key in a relation, it does not have an alternate key.

**For example,** employee relation has two attributes, Employee\_Id and PAN\_No, that act as candidate keys. In this relation, Employee\_Id is chosen as the primary key, so the other candidate key, PAN\_No, acts as the Alternate key.



### 6. Composite key

Whenever a primary key consists of more than one attribute, it is known as a composite key. This key is also known as Concatenated Key.



**For example,** in employee relations, we assume that an employee may be assigned multiple roles, and an employee may work on multiple projects simultaneously. So the primary key will be composed of all three attributes, namely Emp\_ID, Emp\_role, and Proj\_ID in combination. So these attributes act as a composite key since the primary key comprises more than one attribute.



### 7. Artificial key

The key created using arbitrarily assigned data are known as artificial keys. These keys are created when a primary key is large and complex and has no relationship with many other relations. The data values of the artificial keys are usually numbered in a serial order.

**For example,** the primary key, which is composed of Emp\_ID, Emp\_role, and Proj\_ID, is large in employee relations. So it would be better to add a new virtual attribute to identify each tuple in the relation uniquely.

# Relational Calculus

There is an alternate way of formulating queries known as Relational Calculus. Relational calculus is a non-procedural query language. In the non-procedural query language, the user is concerned with the details of how to obtain the end results. The relational calculus tells what to do but never explains how to do. Most commercial relational languages are based on aspects of relational calculus including SQL-QBE and QUEL.

### Why it is called Relational Calculus?

It is based on Predicate calculus, a name derived from branch of symbolic language. A predicate is a truth-valued function with arguments. On substituting values for the arguments, the function result in an expression called a proposition. It can be either true or false. It is a tailored version of a subset of the Predicate Calculus to communicate with the relational database.

**Many of the calculus expressions involves the use of Quantifiers. There are two types of quantifiers:**

* **Universal Quantifiers:** The universal quantifier denoted by ∀ is read as for all which means that in a given set of tuples exactly all tuples satisfy a given condition.
* **Existential Quantifiers:** The existential quantifier denoted by ∃ is read as for all which means that in a given set of tuples there is at least one occurrences whose value satisfy a given condition.

Before using the concept of quantifiers in formulas, we need to know the concept of Free and Bound Variables.

A tuple variable t is bound if it is quantified which means that if it appears in any occurrences a variable that is not bound is said to be free.

Free and bound variables may be compared with global and local variable of programming languages.

## Types of Relational calculus:



### 1. Tuple Relational Calculus (TRC)

It is a non-procedural query language which is based on finding a number of tuple variables also known as range variable for which predicate holds true. It describes the desired information without giving a specific procedure for obtaining that information. The tuple relational calculus is specified to select the tuples in a relation. In TRC, filtering variable uses the tuples of a relation. The result of the relation can have one or more tuples.

**Notation:**

A Query in the tuple relational calculus is expressed as following notation

1. {T | P (T)}   or {T | Condition (T)}

Where

**T** is the resulting tuples

**P(T)** is the condition used to fetch T.

**For example:**

1. { T.name | Author(T) AND T.article = 'database' }

**Output:** This query selects the tuples from the AUTHOR relation. It returns a tuple with 'name' from Author who has written an article on 'database'.

TRC (tuple relation calculus) can be quantified. In TRC, we can use Existential (∃) and Universal Quantifiers (∀).

**For example:**

1. { R| ∃T ∈ Authors(T.article='database' AND R.name=T.name)}

**Output:** This query will yield the same result as the previous one.

### 2. Domain Relational Calculus (DRC)

The second form of relation is known as Domain relational calculus. In domain relational calculus, filtering variable uses the domain of attributes. Domain relational calculus uses the same operators as tuple calculus. It uses logical connectives ∧ (and), ∨ (or) and ┓ (not). It uses Existential (∃) and Universal Quantifiers (∀) to bind the variable. The QBE or Query by example is a query language related to domain relational calculus.

**Notation:**

1. { a1, a2, a3, ..., an | P (a1, a2, a3, ... ,an)}

Where

**a1, a2** are attributes
**P** stands for formula built by inner attributes

**For example:**

1. {< article, page, subject > |  ∈ javatpoint ∧ subject = 'database'}

**Output:** This query will yield the article, page, and subject from the relational javatpoint, where the subject is a database.

# SQL

* SQL stands for Structured Query Language. It is used for storing and managing data in relational database management system (RDMS).
* It is a standard language for Relational Database System. It enables a user to create, read, update and delete relational databases and tables.
* All the RDBMS like MySQL, Informix, Oracle, MS Access and SQL Server use SQL as their standard database language.
* SQL allows users to query the database in a number of ways, using English-like statements.

### Rules:

SQL follows the following rules:

* Structure query language is not case sensitive. Generally, keywords of SQL are written in uppercase.
* Statements of SQL are dependent on text lines. We can use a single SQL statement on one or multiple text line.
* Using the SQL statements, you can perform most of the actions in a database.
* SQL depends on tuple relational calculus and relational algebra.

### SQL process:

* When an SQL command is executing for any RDBMS, then the system figure out the best way to carry out the request and the SQL engine determines that how to interpret the task.
* In the process, various components are included. These components can be optimization Engine, Query engine, Query dispatcher, classic, etc.
* All the non-SQL queries are handled by the classic query engine, but SQL query engine won't handle logical files.



# Characteristics of SQL

* SQL is easy to learn.
* SQL is used to access data from relational database management systems.
* SQL can execute queries against the database.
* SQL is used to describe the data.
* SQL is used to define the data in the database and manipulate it when needed.
* SQL is used to create and drop the database and table.
* SQL is used to create a view, stored procedure, function in a database.
* SQL allows users to set permissions on tables, procedures, and views.

# Advantages of SQL

There are the following advantages of SQL:

### High speed

Using the SQL queries, the user can quickly and efficiently retrieve a large amount of records from a database.

### No coding needed

In the standard SQL, it is very easy to manage the database system. It doesn't require a substantial amount of code to manage the database system.

### Well defined standards

Long established are used by the SQL databases that are being used by ISO and ANSI.

### Portability

SQL can be used in laptop, PCs, server and even some mobile phones.

### Interactive language

SQL is a domain language used to communicate with the database. It is also used to receive answers to the complex questions in seconds.

### Multiple data view

Using the SQL language, the users can make different views of the database structure.