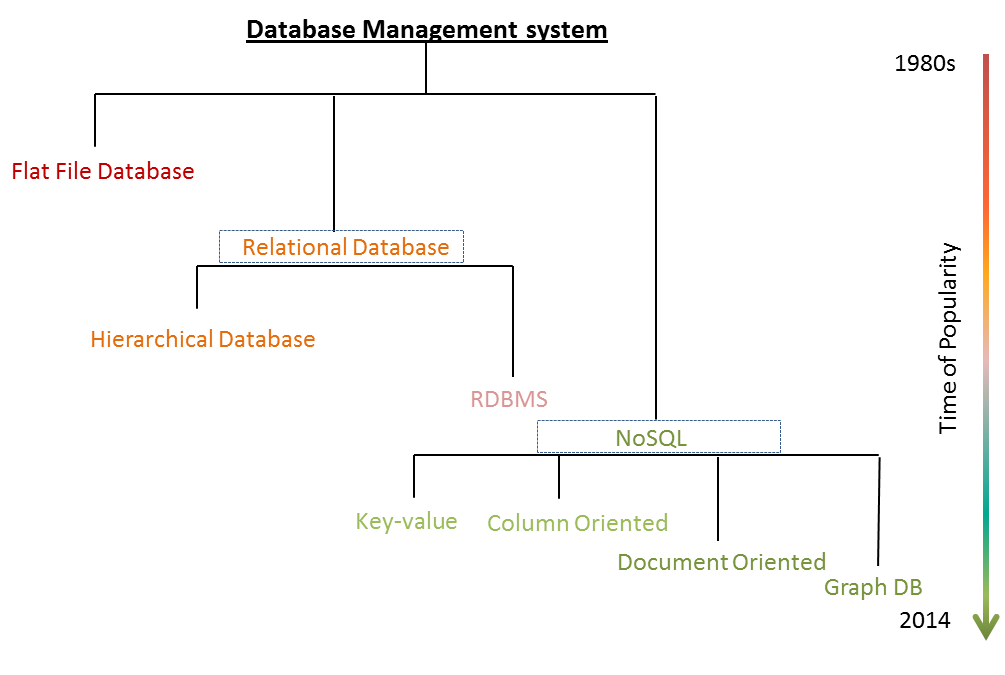
**Dbms :** **database** is a systematic collection of data. They support electronic storage and manipulation of data

Databases are classified based on organizing the data and relationship between data



**flat-file database**

A **flat-file database** is a [database](https://en.wikipedia.org/wiki/Database) stored in a file called a **flat file**. Records follow a uniform format, and there are no structures for indexing or recognizing relationships between records. The file is simple. A flat file can be a [plain text](https://en.wikipedia.org/wiki/Plain_text) file, or a [binary file](https://en.wikipedia.org/wiki/Binary_file).

Flat file databases were **developed by IBM** in the early **1970s.**

**Where we can use Flat file database?**

Flat-file databases could be used for a number of things, eg:

* usernames and passwords
* contact details
* product details
* game or music collections
* entities and attributes

A **flat file database** is basically a giant collection of data in which the tables and records have no relation between any other tables. In fact, one could have a single table (e.g., My Small Business Data) with everything stored in it, from customers to sales to orders to invoices.

Sound too messy? Often it is. But there are uses. One doesn't necessarily have to **normalize** a database. Sometimes simple is best.

**When we can use Flat file database?**

* Whenever we don’t want to create multiple tables and relation in between.

**RELATIONAL DATABASE**

A relational database is a type of database. It uses a structure that allows us to identify and access data in relation to another piece of data in the database. Often, data in a relational database is organized into tables.

Based on relationship and organizing data,the relational database is classified into

* Hierarchical database
* Relational database management system

**HIERARCHICAL DATABASE**

A **hierarchical database model** is a [data model](https://en.wikipedia.org/wiki/Data_model) in which the data are organized into a [tree](https://en.wikipedia.org/wiki/Tree_data_structure)-like structure. The data are stored as **records** which are connected to one another through **links**. A record is a collection of fields, with each field containing only one value. The **type** of a record defines which fields the record contains.

The hierarchical database model mandates that each child record has only one parent, whereas each parent record can have one or more child records. In order to retrieve data from a hierarchical database, the whole tree needs to be traversed starting from the root node. This model is recognized as the first database model created by IBM in the 1960s

**Where we can use Hierarchical database?**

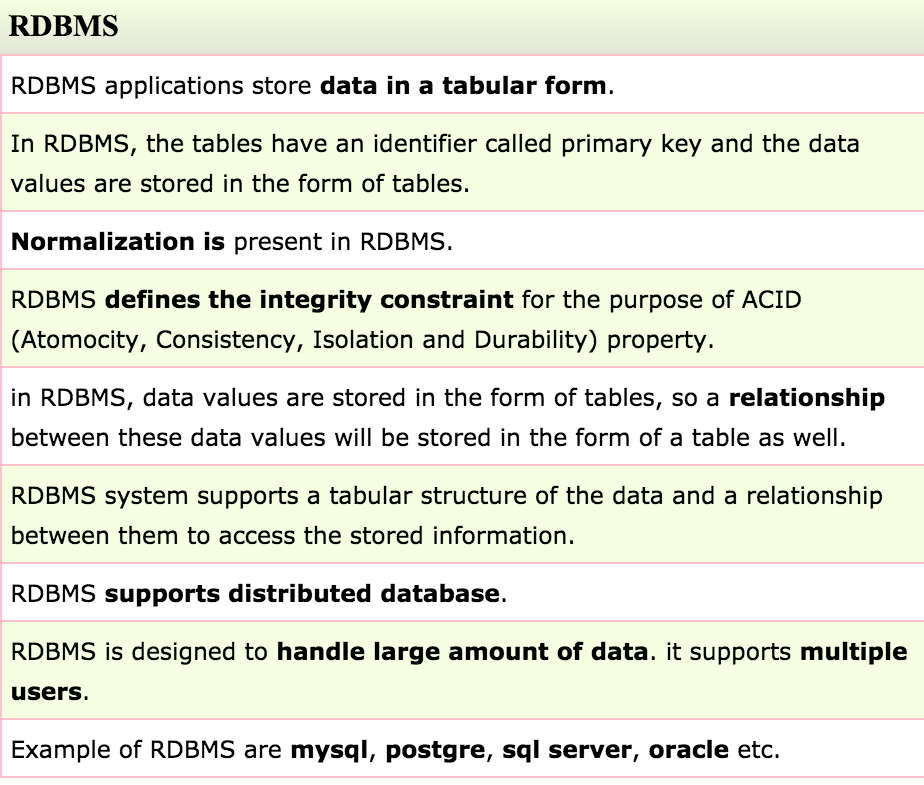
**The hierarchical** structure is **used** primarily today for storing geographic information and file systems. Currently, **hierarchical databases** are still widely **used** especially in applications that require very high performance and availability such as banking and telecommunications.

**RELATIONAL DATABASE MANAGEMENT SYSTEM**

A relational database management system (RDBMS or just RDB) is a common type of database that stores data in tables, so it can be used in relation to other stored datasets.

The relationships between data in the tables in the relational database can be linked in several ways:

* Characteristics of one table record may be linked to a record in another table
* A table record could be linked to many records in another table
* Many table records may be related to many records in another table.



In Relational database , data is stored in the form of tables.

**TABLES**

A **table** is a collection of related data held in a [table](https://en.wikipedia.org/wiki/Table_(information)) format within a [database](https://en.wikipedia.org/wiki/Database). It consists of [columns](https://en.wikipedia.org/wiki/Column_(database)) and [rows](https://en.wikipedia.org/wiki/Row_(database)).

**The characteristics of a relational table are summarized as follows:**

1. A table is perceived as a two-dimensional structure composed of rows and columns.

2. Each table row (tuple) represents a single entity occurrence within the entity set.

3. Each table column represents an attribute, and each column has a distinct name.

4. Each row/column intersection represents a single data value.

5. All values in a column must conform to the same data format.

6. Each column has a specific range of values known as the attribute domain.

7. The order of the rows and columns is immaterial to the DBMS.

8. Each table must have an attribute or a combination of attributes that uniquely identifies each row.

Advantages

* Simple Model Data Accuracy
* Easy Access to Data.
* Data Integrity.
* Flexibility.
* Normalization.
* High Security.
* Feasible for Future Modifications.

Disadvantages

* If you have many  tables in the same database directory, open, close, and create operations are slow.
* If you execute [SELECT](https://dev.mysql.com/doc/refman/8.0/en/select.html) statements on many different tables, there is a little overhead when the table cache is full, because for every table that has to be opened, another must be closed. You can reduce this overhead by increasing the number of entries permitted in the table cache.

**Views**

a **view** is the [result set](https://en.wikipedia.org/wiki/Result_set) of a *stored* [query](https://en.wikipedia.org/wiki/Query_language) on the [data](https://en.wikipedia.org/wiki/Data), which the [database](https://en.wikipedia.org/wiki/Database) users can query just as they would in a persistent database collection object.

## Types of Views

Views are of two types-

* System defined view
* User-defined view

System defined view -- Already defined in the system

User-defined view -- Defined by the user

### ****System Defined View****

Three types-

* Information schema view
* Catalog view
* Dynamic management view

**User Defined Views**

* **Simple View:** A view based on only a single table, which doesn't contain GROUP BY clause and any functions.
* **Complex View:** A view based on multiple tables, which contain GROUP BY clause and functions.
* **Inline View:** A view based on a subquery in FROM Clause, that subquery creates a temporary table and simplifies the complex query.
* **Materialized View:** A view that stores the definition as well as data. It creates replicas of data by storing it physically.

**Advantages:**

1. Views don't store data in a physical location.

2. The view can be used to hide some of the columns from the table.

3. Views can provide Access Restriction, since data insertion, update and deletion is not possible with the view.

**Disadvantages:**

1. When a table is dropped, associated view become irrelevant.

2. Since the view is created when a query requesting data from view is triggered, its a bit slow.

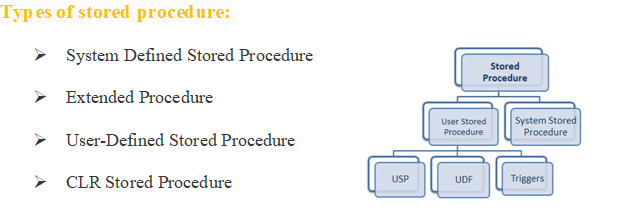
3. When views are created for large tables, it occupies more memory.

**Difference between table and view**

|  |  |  |
| --- | --- | --- |
| **SN** | **Table** | **View** |
| 1. | A table is used to organize data in the form of rows and columns and displayed them in a structured format. It makes the stored information more understandable to the human. | Views are treated as a virtual/logical table used to view the table. It is a database object that contains rows and columns the same as real tables. |
| 2. | Table is a physical entity that means data is actually stored in the table. | The view is a virtual entity, which means data is not actually stored in the table. |
| 3. | It is used to store the data. | It is used to extract data from the table. |
| 4. | It generates a fast result. | The view generates a slow result because it renders the information from the table every time we query it. |
| 5. | It is an independent data object. | It depends on the table. Therefore we cannot create a view without using tables. |
| 6. | Table allows us to perform DML operations. | The view will enable us to perform DML operations. |
| 7. | It is not an easy task to replace the table directly because of its physical storage. | It is an easy task to replace the view and recreate it whenever needs. |
| 8. | It occupies space on the systems. | It does not occupy space on the systems. |

**STORED PROCEDURE AND STORED FUNCTION**

Stored Procedures are pre-compiled code which are compiled for the first time and its compiled format is saved, which executes (compiled code) whenever it is called.



# 

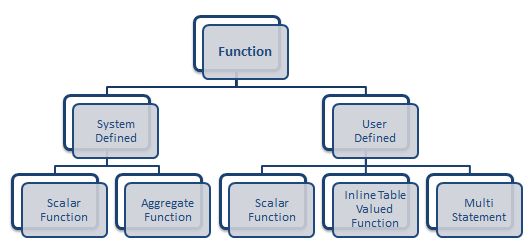
Advantages and disadvantages of stored procedure

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| It is faster. | It is difficult to debug. |
| It is pre-compiled. | Need expert developer, since difficult to write code. |
| It reduces network traffic. | It is database dependent. |
| It is reusable. | It is non-portable. |
| It’s security is high . | It is expensive. |

**Function:**

A function is compiled and executed every time whenever it is called. A **stored function** is a set of **SQL** statements that perform some operation and return a single value.

**Types of function:**



## System Defined Function

## 1.Scalar Function - Scalar functions operate on a single value and return a single value.

|  |  |
| --- | --- |
| **Function** | **Description** |
| LCASE() | Used to convert string column values to lowercase |
| UCASE() | This function is used to convert a string column values to Uppercase. |
| LEN() | Returns the length of the text values in the column. |
| MID() | Extracts substrings in SQL from column values having String data type. |
| ROUND() | Rounds off a numeric value to the nearest integer. |
| NOW() | This function is used to return the current system date and time. |
| FORMAT() | Used to format how a field must be displayed. |

## 2. Aggregate Function - Aggregate functions operate on a collection of values and return a single value.

|  |  |
| --- | --- |
| Function | Description |
| SUM() | Used to return the sum of a group of values. |
| COUNT() | Returns the number of rows either based on a condition, or without a condition. |
| AVG() | Used to calculate the average value of a numeric column. |
| MIN() | This function returns the minimum value of a column. |
| MAX() | Returns a maximum value of a column. |
| FIRST() | Used to return the first value of the column. |
| LAST() | This function returns the last value of the column. |

Go through: https://www.edureka.co/blog/sql-functions

## User Defined Function

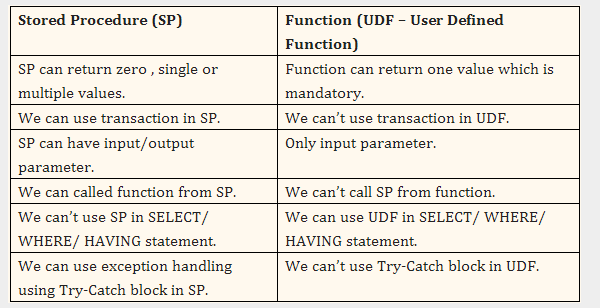
## 1. Scalar Function - The user-defined scalar function also returns a single value as a result of actions performed by the function.

## 2. Inline Table-Valued Function - The user-defined inline table-valued function returns a table variable as a result of actions performed by the function. The value of the table variable should be derived from a single SELECT statement.

## 3. Multi-Statement Table-Valued Function - A user-defined multi-statement table-valued function returns a table variable as a result of actions performed by the function. In this, a table variable must be explicitly declared and defined whose value can be derived from multiple SQL statements.

Go through: https://www.c-sharpcorner.com/UploadFile/3194c4/user-defined-functions-in-sql-server/

Differences between stored procedure and stored function



**Trigger:**

A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.

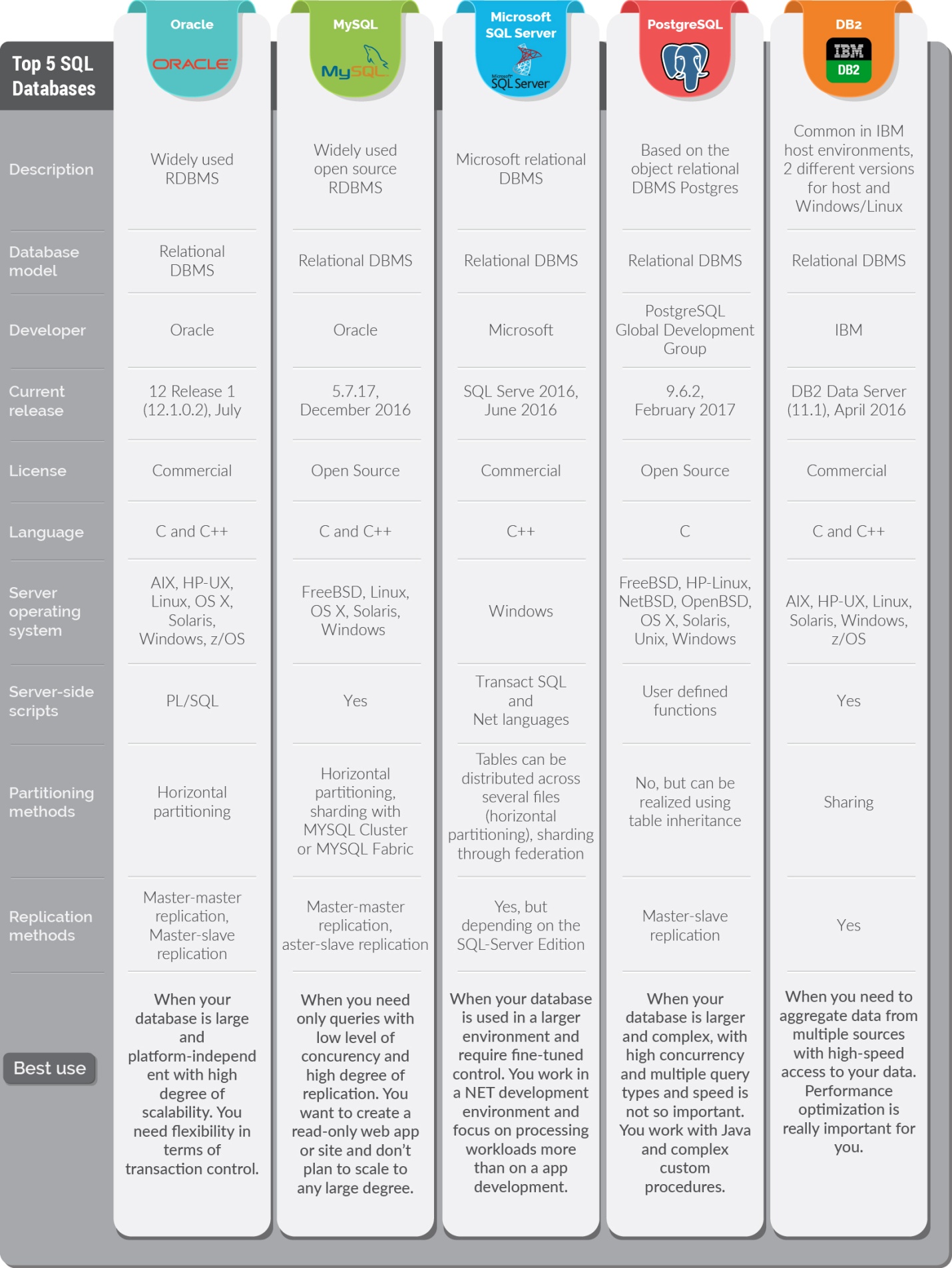
**Types of triggers**

There are three types of triggers in SQL Server.

1. DDL Trigger ( DDL events are CREATE, ALTER and DROP statements)
2. DML Trigger (**DML** events include INSERT, UPDATE, or DELETE statements.)
3. Logon Trigger

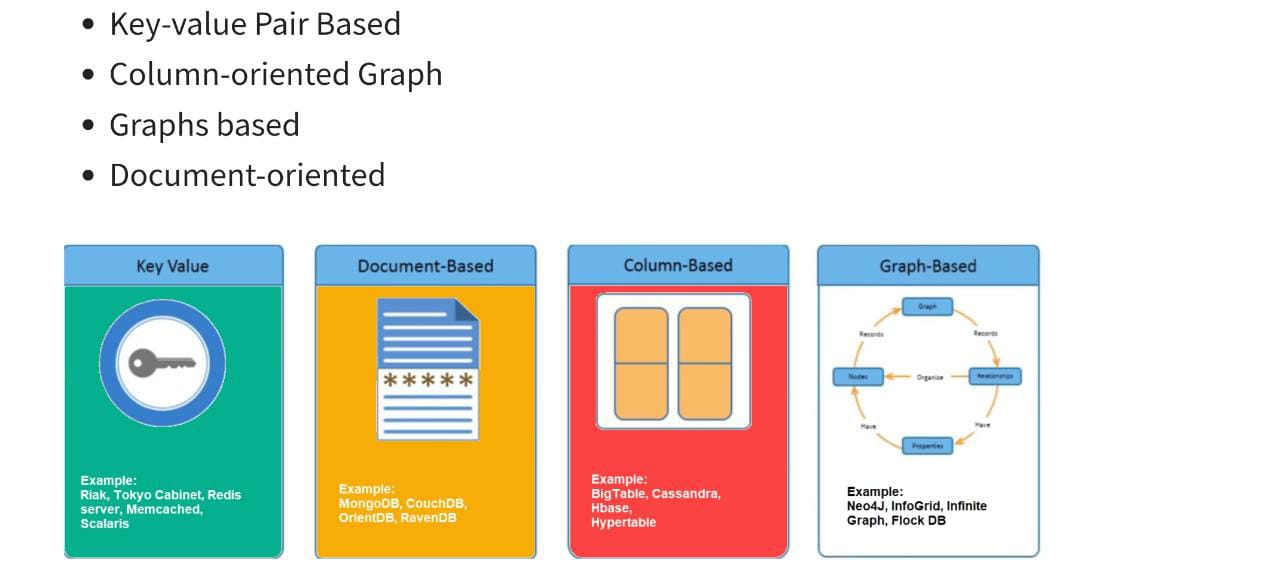
**Difference between Triggers and Stored Procedures**

| **Sr. No.** | **Key** | **Triggers** | **Stored procedures** |
| --- | --- | --- | --- |
| 1 | Basic | trigger is a stored procedure that runs automatically when various events happen (eg update, insert, delete) | Stored procedures are a pieces of the code in written in PL/SQL to do some specific task |
| 2 | Running Methodology | It can execute automatically based on the events | It can be invoked explicitly by the user |
| 3 | Parameter | It can not take input as parameter | It can take input as a parameter |
| 4 | Transaction statements | we can't use transaction statements inside a trigger | We can use transaction statements like begin transaction, commit transaction, and rollback inside a stored procedure |
| 5 | Return | Triggers can not return values | Stored procedures can return values |



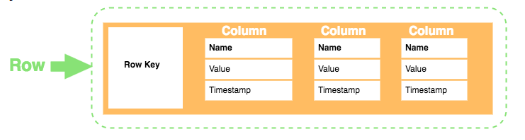
**NO SQL DATABASE**

NoSQL databases (aka "not only SQL") are non tabular, and store data differently than relational tables. NoSQL databases come in a variety of types based on their data model. The main types are document, key-value, wide-column, and graph. They provide flexible schemas and scale easily with large amounts of data and high user loads.



**KEY – VALUE DATABASE**

A **key–value database**, or **key–value store**, is a data storage paradigm designed for storing, retrieving, and managing [associative arrays](https://en.wikipedia.org/wiki/Associative_array), and a [data structure](https://en.wikipedia.org/wiki/Data_structure) more commonly known today as a *dictionary* or [*hash table*](https://en.wikipedia.org/wiki/Hash_table). Dictionaries contain a [collection](https://en.wikipedia.org/wiki/Collection_(abstract_data_type)) of [*objects*](https://en.wikipedia.org/wiki/Object_(computer_science)), or [*records*](https://en.wikipedia.org/wiki/Record_(computer_science)), which in turn have many different [*fields*](https://en.wikipedia.org/wiki/Field_(computer_science)) within them, each containing data. These records are stored and retrieved using a *key* that uniquely identifies the record, and is used to find the data within the [database](https://en.wikipedia.org/wiki/Database).



**COLUMN ORIENTED DATABASE**

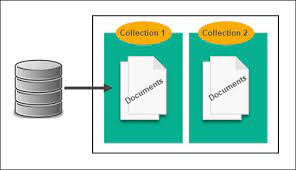
A **column-oriented DBMS** or **columnar DBMS** is a [database management system](https://en.wikipedia.org/wiki/Database_management_system) (DBMS) that stores data tables by [column](https://en.wikipedia.org/wiki/Column_(data_store)) rather than by row.

The goal of column oriented database is to efficiently read and write data to and from hard disk storage in order to speed up the time it takes to return a query

**DOCUMENT ORIENTED DATABASE**

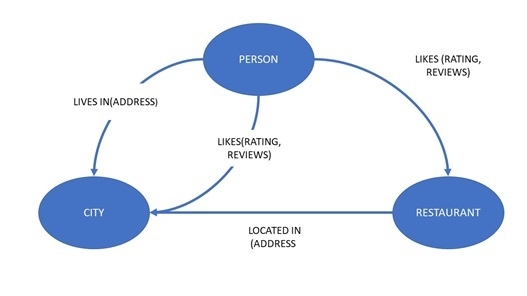
A document database is a type of non-relational database that is designed to store and query data as JSON-like documents. Document databases make it easier for developers to store and query data in a database by using the same document-model format they use in their application code.

WHERE WE CAN USE : catalogs, user profiles,etc,.



**GRAPH DATABASE**

The graph is a collection of nodes and edges where each node is used to represent an entity and each edge describes the relationship between entities. A graph-oriented database, or graph database, is a type of NoSQL database that uses graph theory to store, map and query relationships.



EXAMPLES: TWITTER, FACEBOOK ,ETC.

