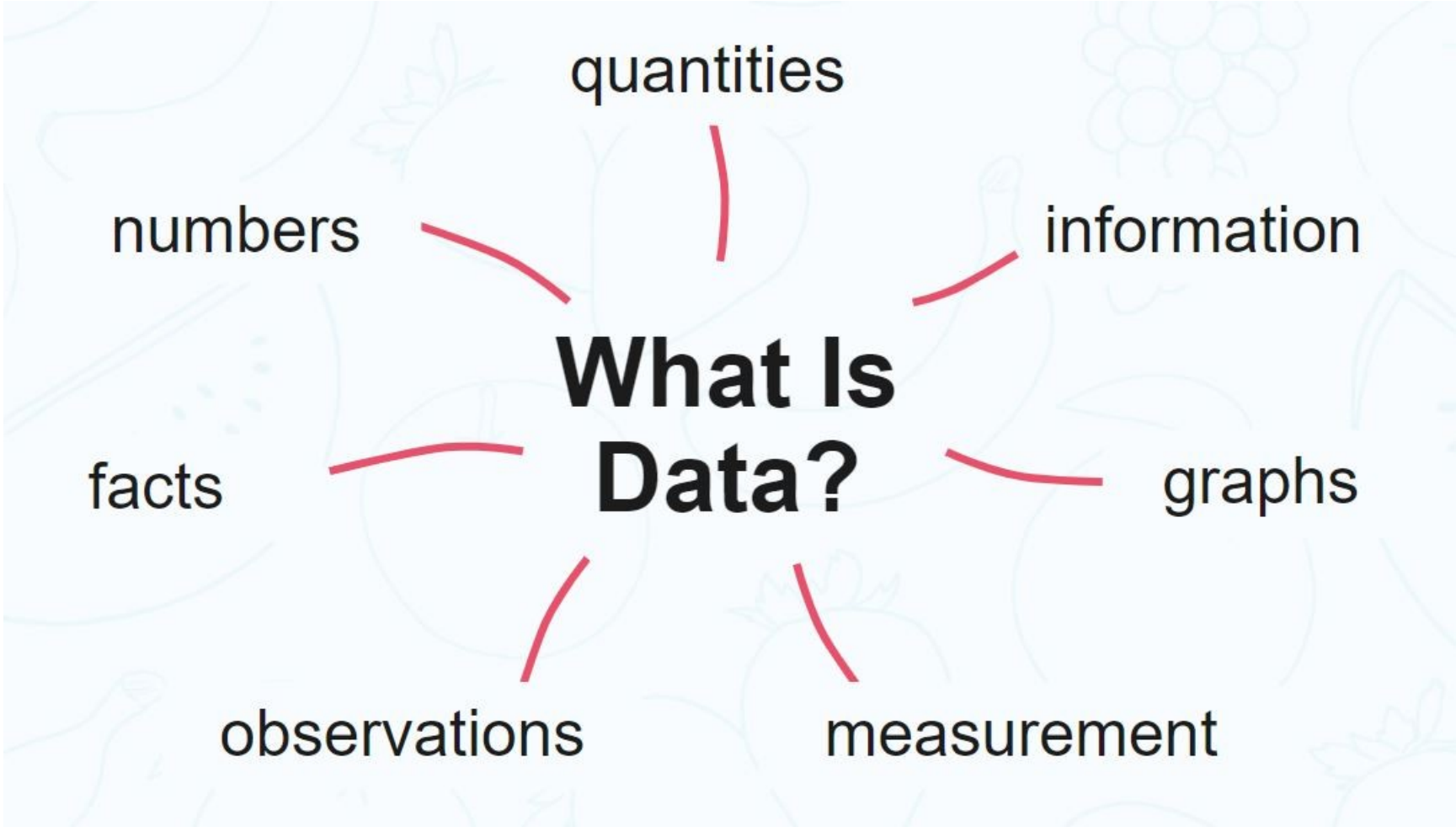
Database Management System

Tushar B. Kute,
http://tusharkute.com

What is Data?


quantities
numbers
information
What Is Data?
facts
graphs
observations
measurement

What is Data?

- Data is a collection of a distinct small unit of information. It can be used in a variety of forms like text, numbers, media, bytes, etc. it can be stored in pieces of paper or electronic memory, etc.
- Word 'Data' is originated from the word 'datum' that means 'single piece of information.' It is plural of the word datum.
- In computing, Data is information that can be translated into a form for efficient movement and processing. Data is interchangeable.

What is Database?

- A database is an organized collection of data, so that it can be easily accessed and managed.
- You can organize data into tables, rows, columns, and index it to make it easier to find relevant information.
- Database handlers create a database in such a way that only one set of software program provides access of data to all the users.
- The main purpose of the database is to operate a large amount of information by storing, retrieving, and managing data.

What is Database Management System?

- A database management system (DBMS) is system software for creating and managing databases.
- A DBMS makes it possible for end users to create, protect, read, update and delete data in a database.
- The most prevalent type of data management platform, the DBMS essentially serves as an interface between databases and users or application programs, ensuring that data is consistently organized and remains easily accessible.

What DBMS do?

- The DBMS manages the data; the database engine allows data to be accessed, locked and modified; and the database schema defines the database's logical structure.
- These three foundational elements help provide concurrency, security, data integrity and uniform data administration procedures.
- The DBMS supports many typical database administration tasks, including change management, performance monitoring and tuning, security, and backup and recovery.
- Most database management systems are also responsible for automated rollbacks and restarts as well as logging and auditing of activity in databases and the applications that access them.

What DBMS do?

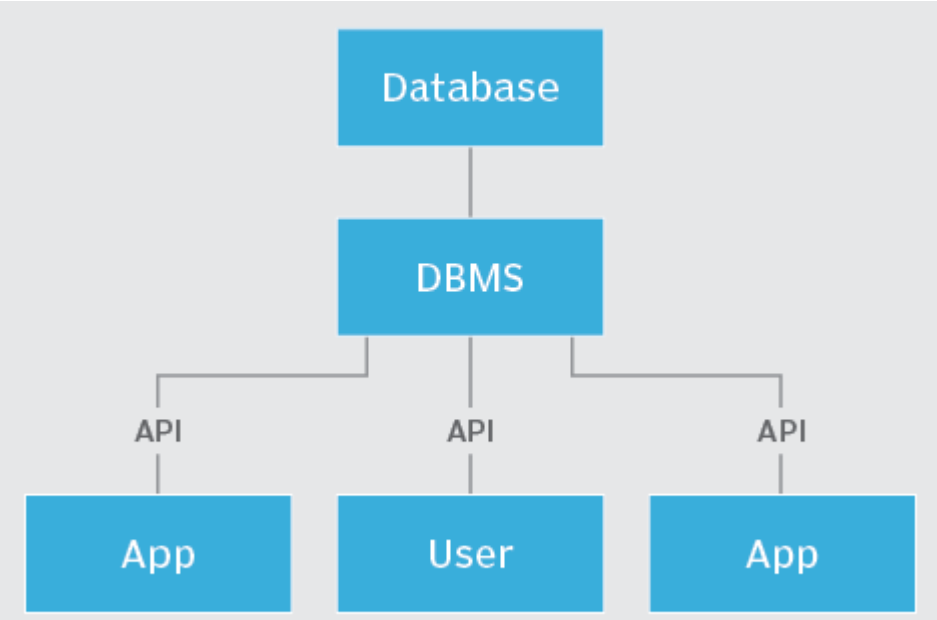
- The DBMS provides a centralized view of data that can be accessed by multiple users from multiple locations in a controlled manner.
- A DBMS can limit what data end users see and how they view the data, providing many views of a single database schema.
- End users and software programs are free from having to understand where the data is physically located or on what type of storage medium it resides because the DBMS handles all requests.

What DBMS do?

- The DBMS can offer both logical and physical data independence to protect users and applications from having to know where data is stored or from being concerned about changes to the physical structure of data.
- So long as programs use the application programming interface (API) for the database that the DBMS provides, developers won't have to modify programs just because changes have been made to the database.

DBMS Components

- A DBMS is a sophisticated piece of system software consisting of multiple integrated components that deliver a consistent, managed environment for creating, accessing and modifying data in databases



Storage Engine

- This basic element of a DBMS is used to store data. The DBMS must interface with a file system at the operating system (OS) level to store data.
- It can use additional components to store data or interface with the actual data at the file system level.

Metadata Catlog

- Sometimes called a system catalog or database dictionary, a metadata catalog functions as a repository for all the database objects that have been created.
- When databases and other objects are created, the DBMS automatically registers information about them in the metadata catalog.
- The DBMS uses this catalog to verify user requests for data, and users can query the catalog for information about the database structures that exist in the DBMS.
- The metadata catalog can include information about database objects, schemas, programs, security, performance, communication and other environmental details about the databases it manages.

Database access language

- The DBMS also must provide an API to access the data, typically in the form of a database access language to access and modify data but may also be used to create database objects and secure and authorize access to the data.
- SQL is an example of a database access language and encompasses several sets of commands, including Data Control Language for authorizing data access, Data Definition Language for defining database structures and Data Manipulation Language for reading and modifying data.

DBMS Components

- Optimization engine.
  - A DBMS may also provide an optimization engine, which is used to parse database access language requests and turn them into actionable commands for accessing and modifying data.
- Query processor.
  - After a query is optimized, the DBMS must provide a means for running the query and returning the results.

DBMS Components

- Lock manager.
  - This crucial component of the DBMS manages concurrent access to the same data. Locks are required to ensure multiple users aren't trying to modify the same data simultaneously.
- Log manager.
  - The DBMS records all changes made to data managed by the DBMS. The record of changes is known as the log, and the log manager component of the DBMS is used to ensure that log records are made efficiently and accurately.
  - The DBMS uses the log manager during shutdown and startup to ensure data integrity, and it interfaces with database utilities to create backups and run recoveries.

DBMS Components

- Data utilities.
  - A DBMS also provides a set of utilities for managing and controlling database activities.
  - Examples of database utilities include reorganization, runstats, backup and copy, recover, integrity check, load data, unload data and repair database.

DBMS Features

- 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.

DBMS Advantanges

- 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 and 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

DBMS Disadvantanges

- 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.

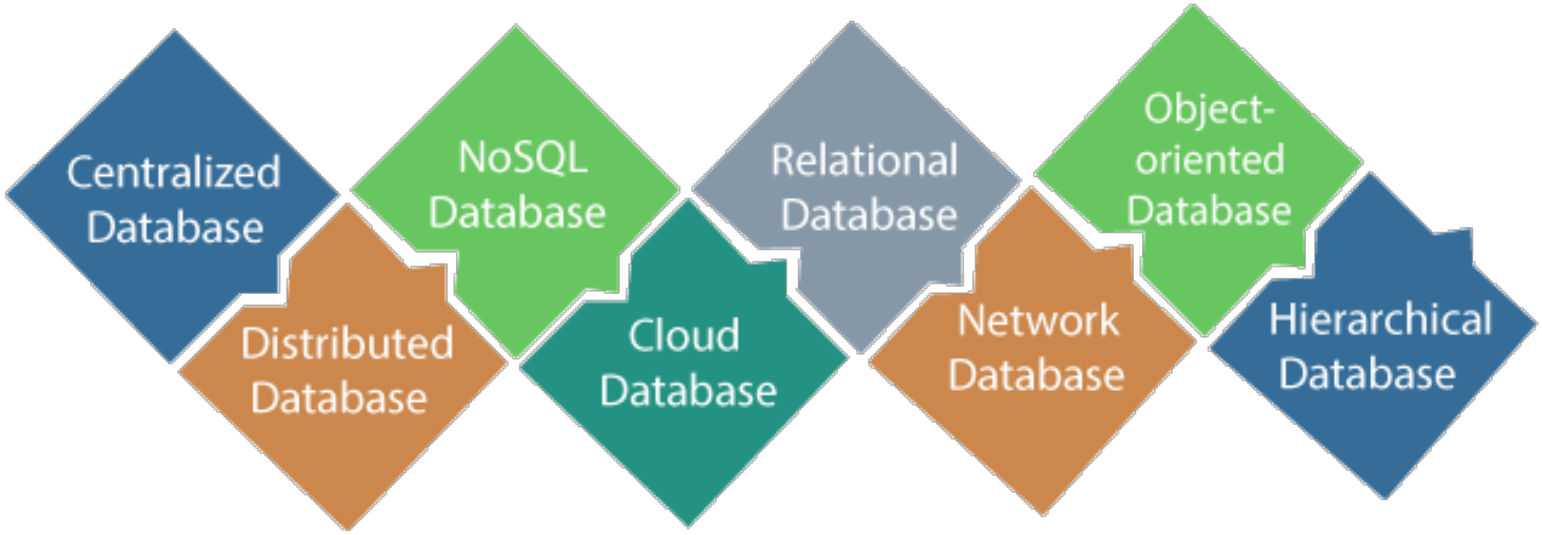
DBMS Terminologies

- Database – A database is a collection of tables, with related data.
- Table – A table is a matrix with data. A table in a database looks like a simple spreadsheet.
- Column – One column (data element) contains data of one and the same kind, for example the column postcode.
- Row – A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.
- Redundancy – Storing data twice, redundantly to make the system faster.

DBMS Terminologies

- Primary Key – A primary key is unique. A key value can not occur twice in one table. With a key, you can only find one row.
- Foreign Key – A foreign key is the linking pin between two tables.
- Compound Key – A compound key (composite key) is a key that consists of multiple columns, because one column is not sufficiently unique.
- Index – An index in a database resembles an index at the back of a book.
- Referential Integrity – Referential Integrity makes sure that a foreign key value always points to an existing row.

Types of Databases



Centralized Database

- It is the type of database that stores data at a centralized database system.
- It comforts the users to access the stored data from different locations through several applications.
- These applications contain the authentication process to let users access data securely.
- An example of a Centralized database can be Central Library that carries a central database of each library in a college/university.

Distributed Database

- Unlike a centralized database system, in distributed systems, data is distributed among different database systems of an organization.
- These database systems are connected via communication links. Such links help the end-users to access the data easily.
- Examples of the Distributed database are Apache Cassandra, HBase, Ignite, etc.

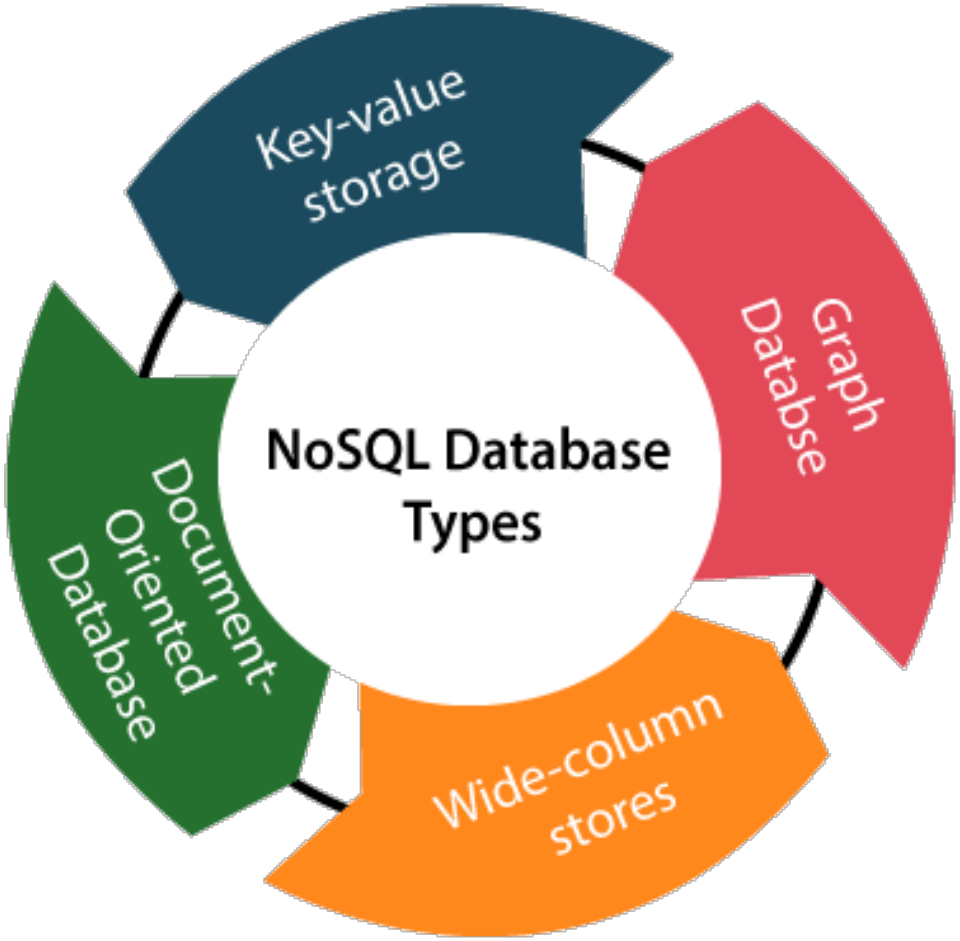
Relational Database

- This database is based on the relational data model, which stores data in the form of rows(tuple) and columns(attributes), and together forms a table(relation).
- A relational database uses SQL for storing, manipulating, as well as maintaining the data. E.F. Codd invented the database in 1970.
- Each table in the database carries a key that makes the data unique from others.
- Examples of Relational databases are MySQL, Microsoft SQL Server, Oracle, etc.

NoSQL Database

- Non-SQL/Not Only SQL is a type of database that is used for storing a wide range of data sets.
- It is not a relational database as it stores data not only in tabular form but in several different ways.
- It came into existence when the demand for building modern applications increased. Thus, NoSQL presented a wide variety of database technologies in response to the demands.
- We can further divide a NoSQL database into the following four types

NoSQL Database



Cloud Database

- A type of database where data is stored in a virtual environment and executes over the cloud computing platform.
- It provides users with various cloud computing services (SaaS, PaaS, IaaS, etc.) for accessing the database. There are numerous cloud platforms, but the best options are:
  - Amazon Web Services(AWS)
  - Microsoft Azure
  - Kamatera
  - PhonixNAP
  - ScienceSoft
  - Google Cloud SQL, etc

Object Oriented Database

- The type of database that uses the object-based data model approach for storing data in the database system.
- The data is represented and stored as objects which are similar to the objects used in the object-oriented programming language.

Hierarchical Database

- It is the type of database that stores data in the form of parent-children relationship nodes. Here, it organizes data in a tree-like structure.
- Data get stored in the form of records that are connected via links.
- Each child record in the tree will contain only one parent. On the other hand, each parent record can have multiple child records.

What is SQL?

- SQL is the standard language for dealing with Relational Databases.
- SQL can be used to insert, search, update, and delete database records. SQL can do lots of other operations, including optimizing and maintenance of databases.
- SQL Full Form
  - SQL stands for Structured Query language, pronounced as “S-Q-L” or sometimes as “See-Quel”… Relational databases like MySQL Database, Oracle, MS SQL Server, Sybase, etc. use ANSI SQL.

What is SQL used for?

- Here are important reasons for using SQL
  - It helps users to access data in the RDBMS system.
  - It helps you to describe the data.
  - It allows you to define the data in a database and manipulate that specific data.
  - With the help of SQL, you can create and drop databases and tables.
  - SQL offers you to use the function in a database, create a view, and stored procedure.
  - You can set permissions on tables, procedures, and views.

MySQL

- MySQL is released under an open-source license. So you have nothing to pay to use it.
- MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
- MySQL uses a standard form of the well-known SQL data language.
- MySQL works on many operating systems and with many languages including Python, PHP, PERL, C, C++, JAVA, etc.
- MySQL works very quickly and works well even with large data sets.

MySQL

- MySQL works very quickly and works well even with large data sets.
- MySQL is very friendly to PHP, the most appreciated language for web development.
- MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
- MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

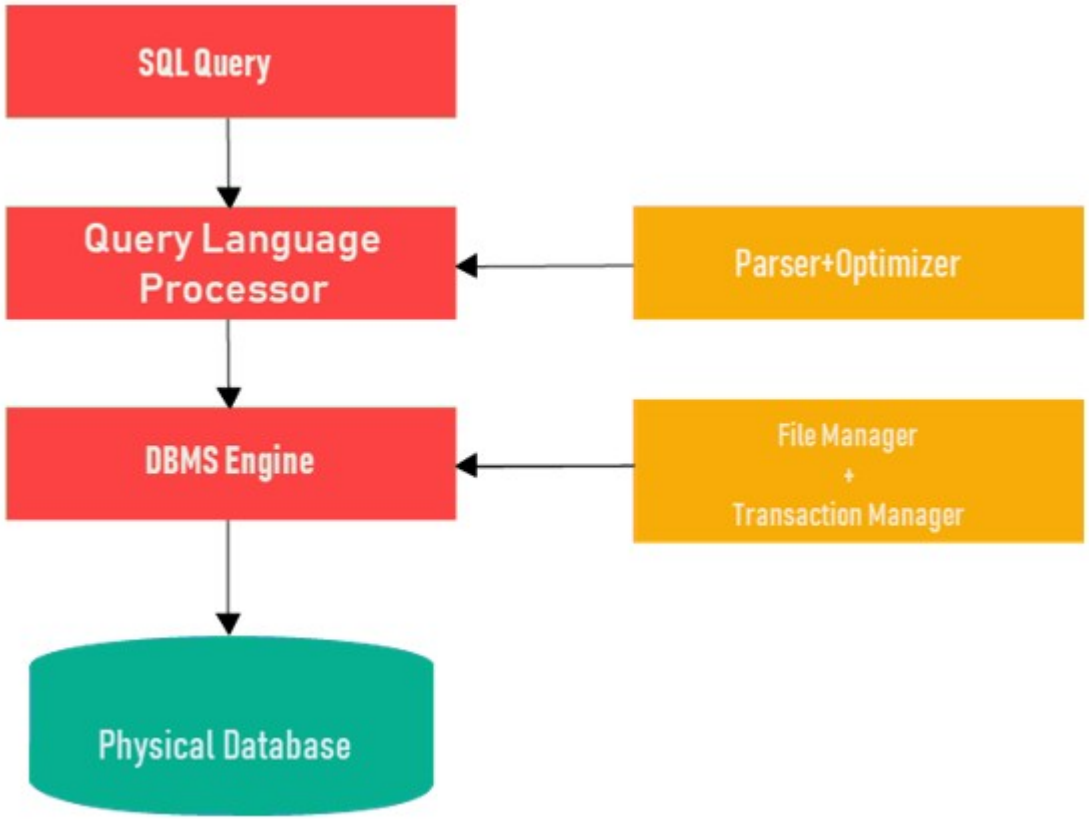
Types of SQL Statements

- Here are five types of widely used SQL queries.
  - Data Definition Language (DDL)
  - Data Manipulation Language (DML)
  - Data Control Language (DCL)
  - Transaction Control Language (TCL)
  - Data Query Language (DQL)

Common SQL Commands

- CREATE – defines the database structure schema
- INSERT – inserts data into the row of a table
- UPDATE – updates data in a database
- DELETE – removes one or more rows from a table
- SELECT – selects the attribute based on the condition described by the WHERE clause
- DROP – removes tables and databases

SQL Process


SQL Query
Query Language Processor
Parser+Optimizer
DBMS Engine
File Manager
+
Transaction Manager
Physical Database

NoSQL

- A NoSQL (originally referring to "non-SQL" or "non-relational") database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.
- Such databases have existed since the late 1960s, but the name "NoSQL" was only coined in the early 21st century, triggered by the needs of Web 2.0 companies.
- NoSQL databases are increasingly used in big data and real-time web applications.
- NoSQL systems are also sometimes called "Not only SQL" to emphasize that they may support SQL-like query languages or sit alongside SQL databases in polyglot-persistent architectures

NoSQL

- Motivations for this approach include: simplicity of design, simpler "horizontal" scaling to clusters of machines (which is a problem for relational databases), finer control over availability and limiting the object-relational impedance mismatch.
- The data structures used by NoSQL databases (e.g. key–value pair, wide column, graph, or document) are different from those used by default in relational databases, making some operations faster in NoSQL.
- The particular suitability of a given NoSQL database depends on the problem it must solve. Sometimes the data structures used by NoSQL databases are also viewed as "more flexible" than relational database tables.

NoSQL – Types

- Wide column: Azure Cosmos DB, Accumulo, Cassandra, Scylla, HBase.
- Document: Azure Cosmos DB, Apache CouchDB, ArangoDB, BaseX, Clusterpoint, Couchbase, eXist-db, IBM Domino, MarkLogic, MongoDB, OrientDB, Qizx, RethinkDB
- Key–value: Azure Cosmos DB, Aerospike, Apache Ignite, ArangoDB, Berkeley DB, Couchbase, Dynamo, FoundationDB, InfinityDB, MemcacheDB, MUMPS, Oracle NoSQL Database, OrientDB, Redis, Riak, SciDB, SDBM/Flat File dbm, ZooKeeper
- Graph: Azure Cosmos DB, AllegroGraph, ArangoDB, InfiniteGraph, Apache Giraph, MarkLogic, Neo4J, OrientDB, Virtuoso

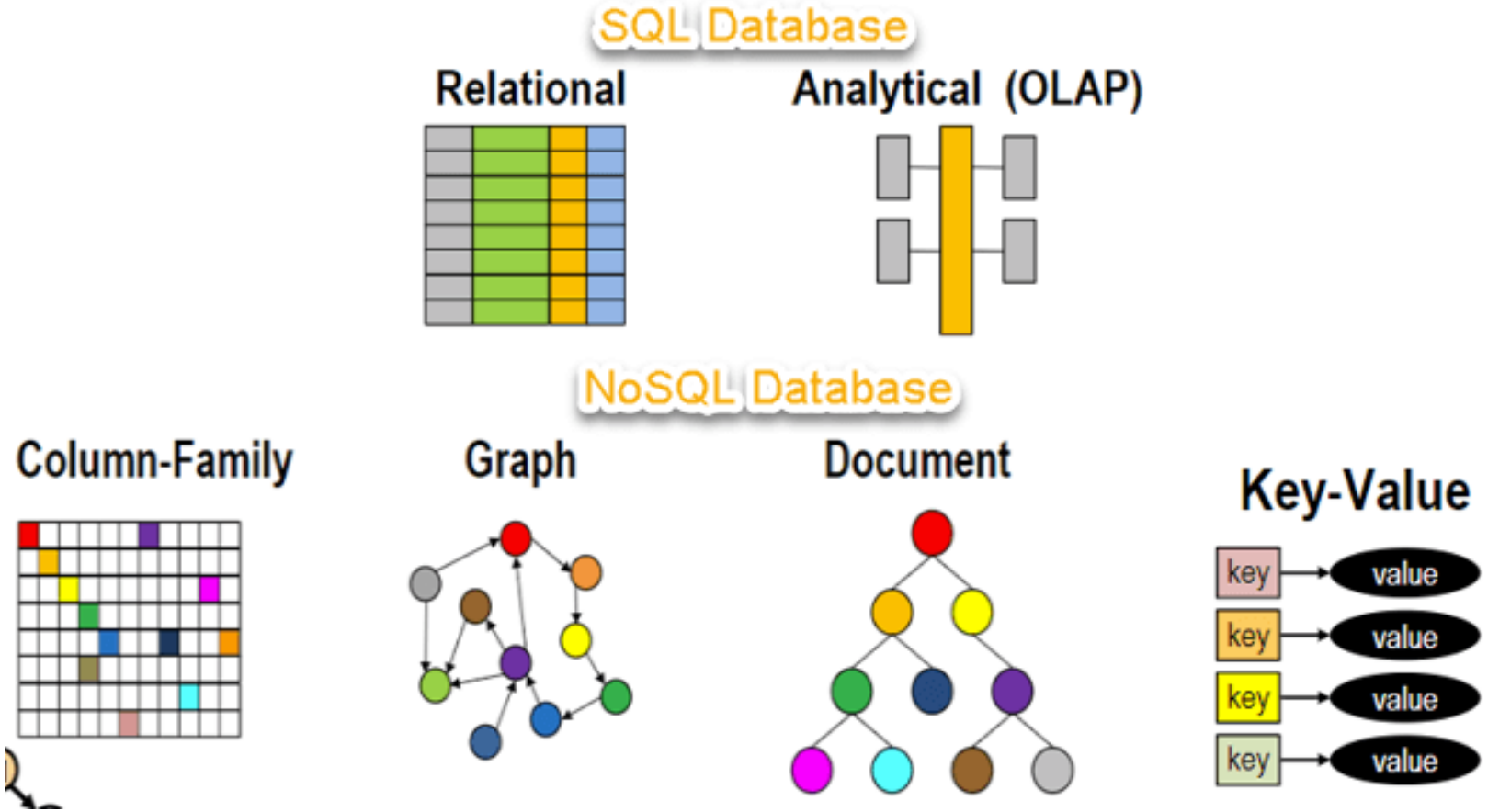
Database as a Service – SQL

- Amazon Aurora, MySQL based service
- Amazon Relational Database Service
- Clustrix Database as a Service
- Crunchy Bridge, PostgreSQL as a Service.
- EnterpriseDB Postgres Plus Cloud Database
- Google Cloud SQL
- Heroku PostgreSQL as a Service (shared and dedicated database options)
- MariaDB SkySQL
- Microsoft Azure SQL Database (MS SQL)
- Oracle Database Cloud Service
- Snowflake Cloud Data Warehouse
- Xeround Cloud Database* – MySQL front-end (*service no longer available)

Database as a Service – NoSQL

- Amazon DynamoDB
- Amazon SimpleDB
- Azure Cosmos DB
- Cloudant Data Layer (CouchDB)
- EnterpriseDB Postgres Plus Cloud Database
- Google Cloud Bigtable
- Google Cloud Datastore
- MongoDB Database as a Service (several options)
- RavenDB Cloud Database as a Service
- Oracle NoSQL Database Cloud Service
- Amazon DocumentDB

Database Types



Comparison

SQL	NOSQL
Relational Database management system	Distributed Database management system
Vertically Scalable	Horizontally Scalable
Fixed or predifined Schema	Dynamic Schema
Not suitable for hierarchical data storage	Best suitable for hierarchical data storage
Can be used for complex queries	Not good for complex queries

Why NoSQL?

- NoSQL databases are used in nearly every industry.
- Use cases range from the highly critical (e.g., storing financial data and healthcare records) to the more fun and frivolous (e.g., storing IoT readings from a smart kitty litter box).

Why NoSQL?

- Support large numbers of concurrent users (tens of thousands, perhaps millions)
- Deliver highly responsive experiences to a globally distributed base of users
- Be always available – no downtime
- Handle semi- and unstructured data
- Rapidly adapt to changing requirements with frequent updates and new features

When NoSQL?

- When deciding which database to use, decision-makers typically find one or more of the following factors lead them to selecting a NoSQL database:
  - Fast-paced Agile development
  - Storage of structured and semi-structured data
  - Huge volumes of data
  - Requirements for scale-out architecture
  - Modern application paradigms like microservices and real-time streaming

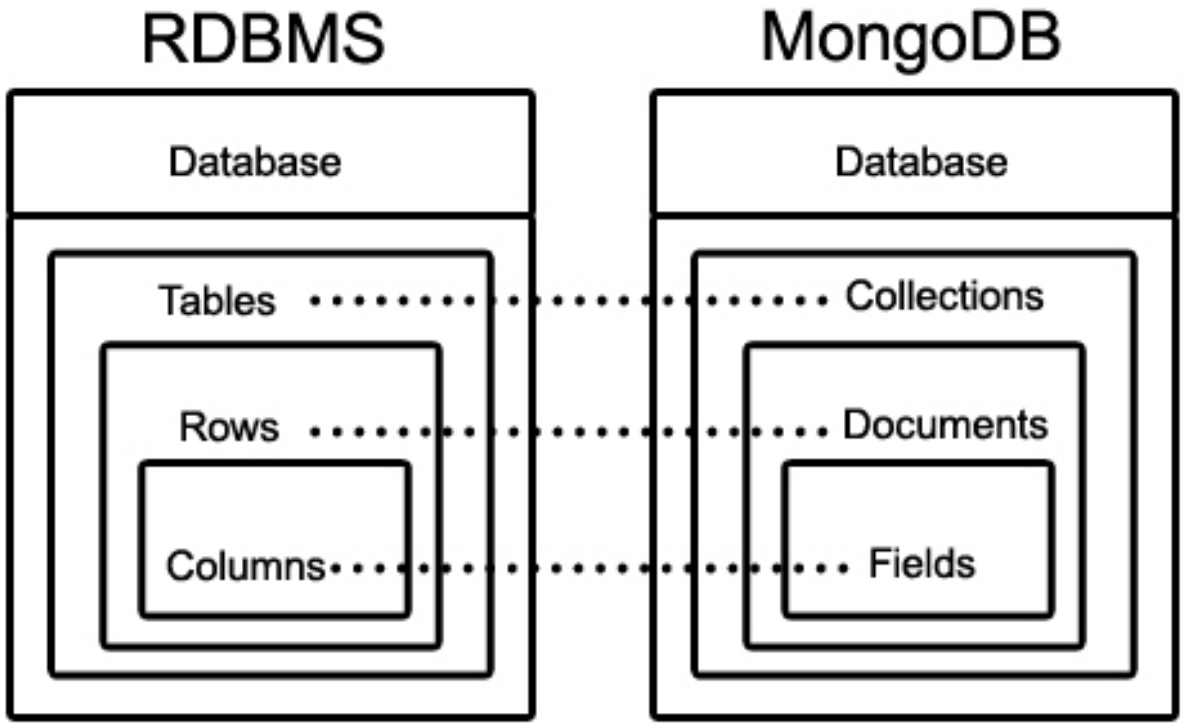
Terminologies

- Database
  - Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.
- Collection
  - Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database.
  - Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.
- Document
  - A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

RDBMS vs. MongoDB

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by MongoDB itself)
Database Server and Client	
mysqld/Oracle	mongod
mysql/sqlplus	mongo

RDBMS vs. MongoDB



Example Document

```
{
   _id: ObjectId(7df78ad8902c)
   title: 'MongoDB Overview',
   description: 'MongoDB is no sql database',
   by: 'MiTU Skillologies',
   url: 'https://www.mitu.co.in',
   tags: ['mongodb', 'database', 'NoSQL'],
   comments: [
      {
         user:'user1',
         message: 'My first comment',
         like: 0
      },
      {
         user:'user2',
         message: 'My second comments',
         dateCreated: new Date(2011,1,25,7,45),
         like: 5
      }
   ]
}
```

Advantages

- Schema less – MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
- Structure of a single object is clear.
- No complex joins.
- Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.
- Ease of scale-out – MongoDB is easy to scale.
- Conversion/mapping of application objects to database objects not needed.
- Uses internal memory for storing the (windowed) working set, enabling faster access of data.

Why to use NoSQL?

- Document Oriented Storage – Data is stored in the form of JSON style documents.
- Index on any attribute
- Replication and high availability
- Auto-Sharding
- Rich queries
- Fast in-place updates
- Professional support by MongoDB

Where to use ?

- Big Data
- Content Management and Delivery
- Mobile and Social Infrastructure
- User Data Management
- Data Hub

File System

- A file system -- sometimes written filesystem -- is the way in which files are named and where they are placed logically for storage and retrieval.
- Without a file system, stored information wouldn't be isolated into individual files and would be difficult to identify and retrieve.
- As data capacities increase, the organization and accessibility of individual files are becoming even more important in data storage.

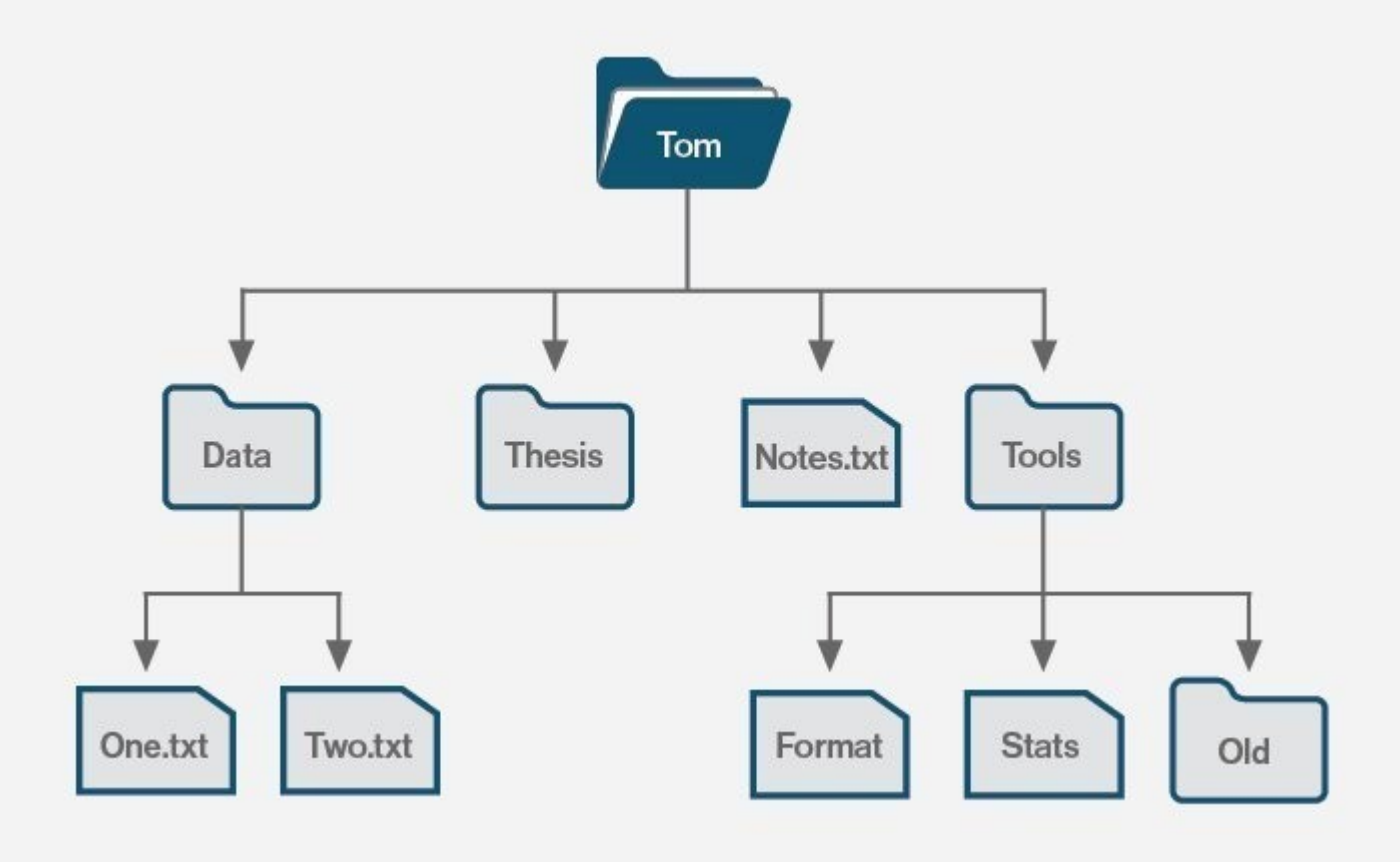
File System

- Digital file systems and files are named for and modeled after paper-based filing systems using the same logic-based method of storing and retrieving documents.
- File systems can differ between operating systems (OS), such as Microsoft Windows, macOS and Linux-based systems.
- Some file systems are designed for specific applications. Major types of file systems include distributed file systems, disk-based file systems and special purpose file systems.

File System: How it works?

- A file system stores and organizes data and can be thought of as a type of index for all the data contained in a storage device.
- These devices can include hard drives, optical drives and flash drives.
- File systems specify conventions for naming files, including the maximum number of characters in a name, which characters can be used and, in some systems, how long the file name suffix can be.
- In many file systems, file names are not case sensitive.

File System: How it works?



File System: Metadata

- File systems use metadata to store and retrieve files. Examples of metadata tags include:
  - Date created
  - Date modified
  - Last date of access
  - Last backup
  - User ID of the file creator
  - Access permissions
  - File size

File System: Types

- File allocation table (FAT) is supported by the Microsoft Windows OS.
- FAT is considered simple and reliable, and it is modeled after legacy file systems.
- FAT was designed in 1977 for floppy disks, but was later adapted for hard disks.
- While efficient and compatible with most current OSes, FAT cannot match the performance and scalability of more modern file systems.

File System: Types

- Global file system (GFS) is a file system for the Linux OS, and it is a shared disk file system.
- GFS offers direct access to shared block storage and can be used as a local file system.
- GFS2 is an updated version with features not included in the original GFS, such as an updated metadata system.
- Under the terms of the GNU General Public License, both the GFS and GFS2 file systems are available as free software.

File System: Types

- Hierarchical file system (HFS) was developed for use with Mac operating systems.
- HFS can also be referred to as Mac OS Standard, and it was succeeded by Mac OS Extended.
- Originally introduced in 1985 for floppy and hard disks, HFS replaced the original Macintosh file system. It can also be used on CD-ROMs.

File System: Types

- The NT file system -- also known as the New Technology File System (NTFS) -- is the default file system for Windows products from Windows NT 3.1 OS onward.
- Improvements from the previous FAT file system include better metadata support, performance and use of disk space.
- NTFS is also supported in the Linux OS through a free, open-source NTFS driver. Mac OSes have read-only support for NTFS.

File System: Types

- Universal Disk Format (UDF) is a vendor-neutral file system used on optical media and DVDs.
- UDF replaces the ISO 9660 file system and is the official file system for DVD video and audio as chosen by the DVD Forum.

File System vs. DBMS

- Like a file system, a database management system (DBMS) efficiently stores data that can be updated and retrieved. The two are not interchangeable, however.
- While a file system stores unstructured, often unrelated files, a DBMS is used to store and manage structured, related data.
- A DBMS creates and defines the restraints for a database. A file system allows access to single files at a time and addresses each file individually.
- Because of this, functions such as redundancy are performed on an individual level, not by the file system itself. This makes a file system a much less consistent form of data storage than a DBMS, which maintains one repository of data that is defined once.

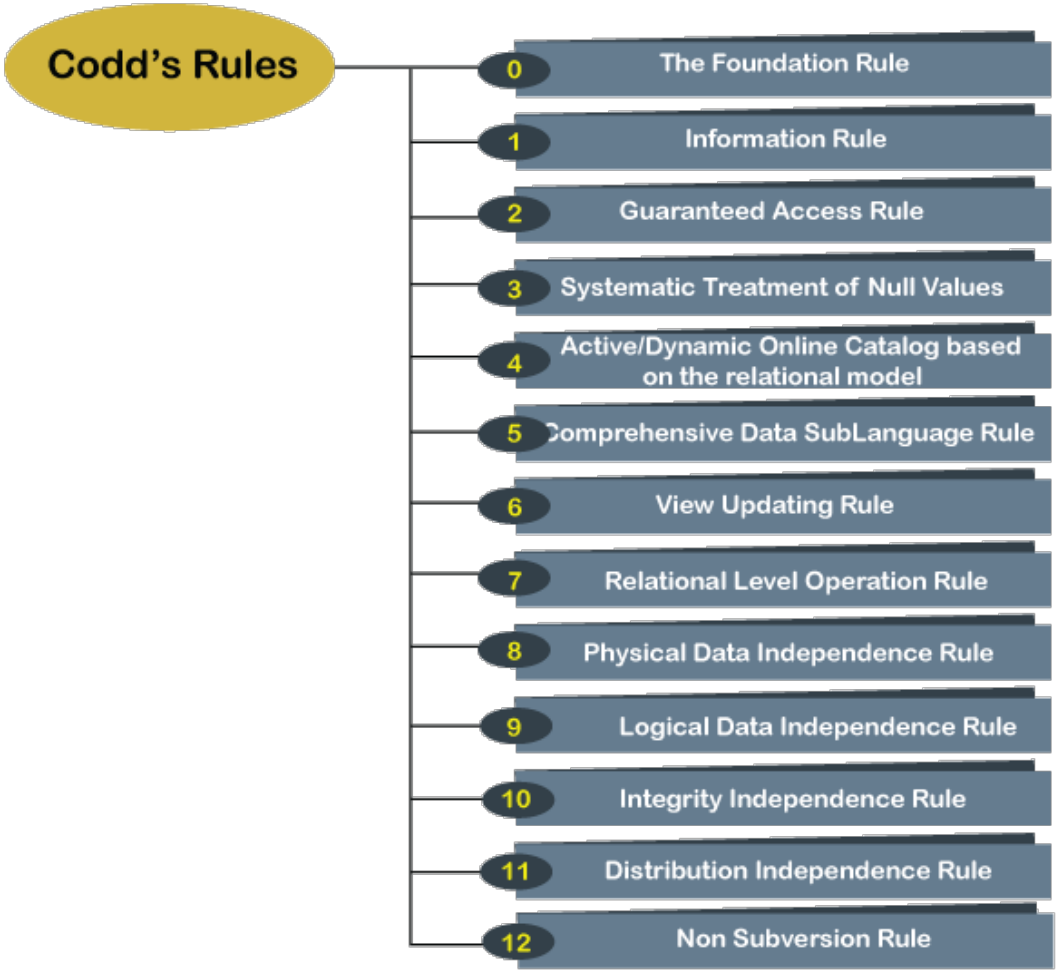
File System vs. DBMS

- The centralized structure of a DBMS allows for easier file sharing than a file system and prevents anomalies that can occur when separate changes are made to files in a file system.
- There are methods to protect files in a file system, but for heavy-duty security, a DBMS is the way to go. Security in a file system is determined by the OS, and it can be difficult to maintain over time as files are accessed and authorization is granted to users.
- A DBMS keeps security constraints high, relying on password protection, encryption and limited authorization. More security does result in more obstacles when retrieving data, so in terms of general, simple-to-use file storage and retrieval, a file system may be preferred.

Codd's 12 Rules for RDBMS

- Every database has tables, and constraints cannot be referred to as a rational database system. And if any database has only relational data model, it cannot be a Relational Database System (RDBMS).
- So, some rules define a database to be the correct RDBMS. These rules were developed by Dr. Edgar F. Codd (E.F. Codd) in 1985, who has vast research knowledge on the Relational Model of database Systems.
- Codd presents his 13 rules for a database to test the concept of DBMS against his relational model, and if a database follows the rule, it is called a true relational database (RDBMS).

Codd's 12 Rules for RDBMS



Codd's 12 Rules for RDBMS

- Rule 0: The Foundation Rule
  - The database must be in relational form. So that the system can handle the database through its relational capabilities.
- Rule 1: Information Rule
  - A database contains various information, and this information must be stored in each cell of a table in the form of rows and columns.
- Rule 2: Guaranteed Access Rule
  - Every single or precise data (atomic value) may be accessed logically from a relational database using the combination of primary key value, table name, and column name.

Codd's 12 Rules for RDBMS

- Rule 3: Systematic Treatment of Null Values
  - This rule defines the systematic treatment of Null values in database records. The null value has various meanings in the database, like missing the data, no value in a cell, inappropriate information, unknown data and the primary key should not be null.
- Rule 4: Active/Dynamic Online Catalog based on the relational model
  - It represents the entire logical structure of the descriptive database that must be stored online and is known as a database dictionary. It authorizes users to access the database and implement a similar query language to access the database.

Codd's 12 Rules for RDBMS

- Rule 5: Comprehensive Data SubLanguage Rule
  - The relational database supports various languages, and if we want to access the database, the language must be the explicit, linear or well-defined syntax, character strings and supports the comprehensive: data definition, view definition, data manipulation, integrity constraints, and limit transaction management operations.
  - If the database allows access to the data without any language, it is considered a violation of the database.

Codd's 12 Rules for RDBMS

- Rule 6: View Updating Rule
  - All views table can be theoretically updated and must be practically updated by the database systems.
- Rule 7: Relational Level Operation (High-Level Insert, Update and delete) Rule
  - A database system should follow high-level relational operations such as insert, update, and delete in each level or a single row. It also supports union, intersection and minus operation in the database system.

Codd's 12 Rules for RDBMS

- Rule 8: Physical Data Independence Rule
  - All stored data in a database or an application must be physically independent to access the database. Each data should not depend on other data or an application.
  - If data is updated or the physical structure of the database is changed, it will not show any effect on external applications that are accessing the data from the database.

Codd’s 12 Rules for RDBMS

- Rule 9: Logical Data Independence Rule
  - It is similar to physical data independence. It means, if any changes occurred to the logical level (table structures), it should not affect the user's view (application).
  - For example, suppose a table either split into two tables, or two table joins to create a single table, these changes should not be impacted on the user view application.

Codd’s 12 Rules for RDBMS

- Rule 10: Integrity Independence Rule
  - A database must maintain integrity independence when inserting data into table's cells using the SQL query language.
  - All entered values should not be changed or rely on any external factor or application to maintain integrity.
  - It is also helpful in making the database-independent for each front-end application.

Codd's 12 Rules for RDBMS

- Rule 11: Distribution Independence Rule
  - The distribution independence rule represents a database that must work properly, even if it is stored in different locations and used by different end-users.
  - Suppose a user accesses the database through an application; in that case, they should not be aware that another user uses particular data, and the data they always get is only located on one site.
  - The end users can access the database, and these access data should be independent for every user to perform the SQL queries.

Codd's 12 Rules for RDBMS

- Rule 12: Non Subversion Rule
  - The non-submersion rule defines RDBMS as a SQL language to store and manipulate the data in the database.
  - If a system has a low-level or separate language other than SQL to access the database system, it should not subvert or bypass integrity to transform data.

Thank you

@mitu_skillologies

@mITuSkillologies

@mitu_group

@mitu-skillologies

@MITUSkillologies

kaggle

@mituskillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

@mituskillologies

contact@mitu.co.in

tushar@tusharkute.com