Introduction To DBMS

Alpesh Vasant, G Academy

System Software

- Also known as Operating System
- Operating System is a vital link (like bridge) between lower level language (machine language) and application programs written in higher level languages(using Java, Python, C etc.)
- To boot and run successfully, computer system need MINIMUM ONE operating system like…
- DOS
- Windows
- Linux
- Sun Solaris
- Unix

File System

- It is the system using which the files and folders are arranged by operating system
- Every Operating System has its own way of storing files and folders
- It defines how the files are named, stored and retrieved from storage devices
- Without file system, everything stored in computer is just big heap of data
- Every Operating System essentially partition the storage in a very specific way E.g. FAT32, NTFS, Linux etc.
- You may have more than one File System (Operating System) on same computer. However at a given point of time, you use only one

Data

- Facts in the form of numbers or reports
- Values of qualitative or quantitative variable
- Often collected for references or analysis. E.g. Temperature of city collected over the years
- Used as a basis for discussion, reasoning or calculation
- In computing world, it's a information processed or stored by a computer.
- This information may be in the form of text documents, images, audio clips, software programs, or other types of *data*

Data Base

- Collection & Storage of Data
- Collection of facts and figures
- May be related or unrelated directly

DBMS & It's Need

- DataBae Management System
- System to manage the data in most ordered and friendly way
- Helps to retrieve , utilize , update, delete or add data
- Now a days, Software Applications are best utilized today for DBMS

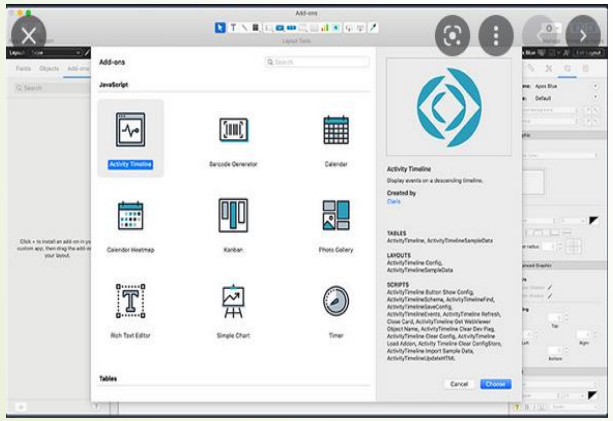
Why DBMS?

- Better way to manage the data
- Storage, Retrieval, updating and deleting the data becomes easy
- Digital DBMS offer very high computational power
- Correlations becomes easy
- Helps in making quick business decisions
- Savings on Time , Energy & Money

DBMS software applications

dBase

Clipper

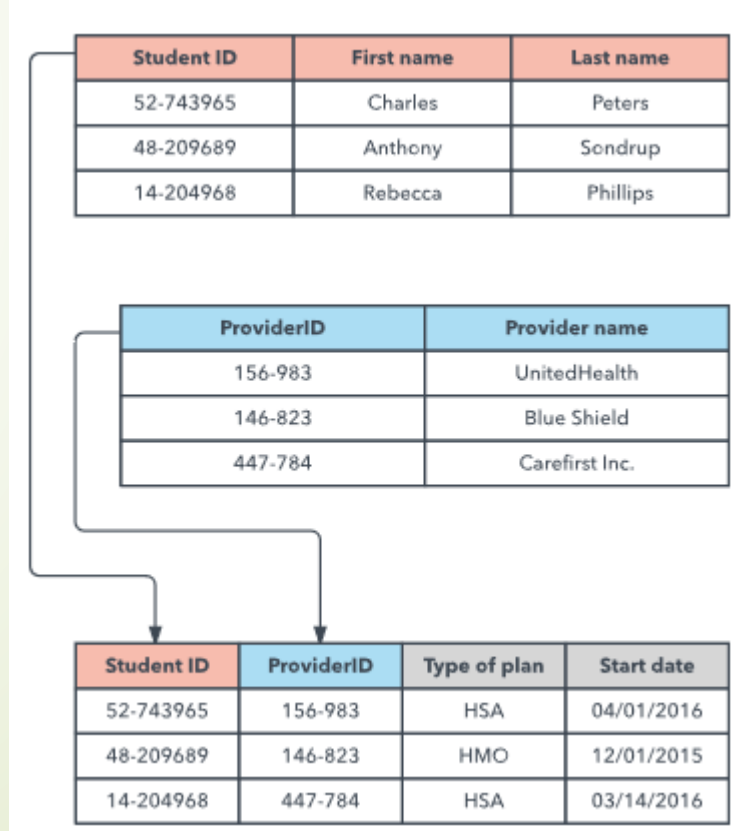


FileMaker

DBMS software applications

RDBMS

- Relational databases
  - Most widely and commonly used model of database
  - Data stored in the TABLES also known as RELATIONS
  - Each table consists of rows & columns
  - Each row is a record also known as tuple
  - Introduced by E.F. Codd in 1970
- Examples
  - Oracle
  - MS SQL Server
  - MySQL
  - PostgreSQL
  - MS Access



Student ID	First name	Last name
52-743965	Charles	Peters
48-209689	Anthony	Sondrup
14-204968	Rebecca	Phillips

ProviderID	Provider name
156-983	UnitedHealth
146-823	Blue Shield
447-784	Carefirst Inc.

Student ID	ProviderID	Type of plan	Start date
52-743965	156-983	HSA	04/01/2016
48-209689	146-823	HMO	12/01/2015
14-204968	447-784	HSA	03/14/2016

Codd's 12 rules for a Relational Database

- **Rule 1:** The *information rule*
  - All Data must be stored in the form of **table Cell**

Codd's 12 rules for a Relational Database

- **Rule 2:** The *guaranteed access rule*
  - Each unique piece of data(atomic value) should be accessible ONLY by : **Table Name + Primary Key(Row) + Attribute(column)**
  - Ability to directly access via **POINTER** is **big NO**

Codd's 12 rules for a Relational Database

- **Rule 3:** Systematic treatment of null values
  - It should be handled consistently.
  - It should have only one of the following three meanings.
    - missing data, not applicable or no value
  - **Primary key must not be null, ever**

Codd's 12 rules for a Relational Database (cont..)

- **Rule 4:** Active Online Catalog
  - Database dictionary(catalog) is the structure description of the complete **Database** and it must be stored online.
  - The Catalog must be governed by same rules as rest of the database.
  - The same query language should be used on catalog as used to query database

Codd's 12 rules for a Relational Database (cont..)

- **Rule 5:** The *comprehensive data sublanguage rule*
  - A database can only be accessed using a language having linear syntax that supports data definition, data manipulation, and transaction management operations. E.g. **SQL**
  - If the database **allows** access to the data **without the use of this language**, then that is a **violation**

Codd's 12 rules for a Relational Database (cont..)

- **Rule 6: View *Updation* Rule**
  - All the view that are theoretically updatable should be updatable by the system as well.

Codd's 12 rules for a Relational Database (cont..)

- **Rule 7: Relational Level Operation**
  - There must be Insert, Delete, Update operations at each level of relations.
  - Set operation like Union, Intersection and minus should also be supported.

Codd's 12 rules for a Relational Database (cont..)

- **Rule 8: Physical Data Independence**
  - The physical storage of data should not matter to the system.
  - If say, some file supporting table is renamed or moved from one disk to another, it should not effect the application

Codd's 12 rules for a Relational Database (cont..)

- **Rule 9: Logical Data Independence**
  - If there is change in the logical structure(table structures) of the database the user view of data should not change.
  - If a table is split into two tables, a new view should give result as the join of the two tables

Codd's 12 rules for a Relational Database (cont..)

- **Rule 10: Integrity Independence**
  - The database should be able to enforce its own integrity rather than using other programs.
  - Key and Check constraints, trigger etc, should be stored in Data Dictionary. This also make **RDBMS** independent of front-end

Codd's 12 rules for a Relational Database (cont..)

- **Rule 11: Distribution Independence**
  - A database should work properly regardless of its distribution across a network.
  - Even if a database is geographically distributed, with data stored in pieces, the end user should get an impression that it is stored at the same place.
  - This lays the foundation of **distributed database**.

Codd's 12 rules for a Relational Database (cont..)

- **Rule 12: Non subversion Rule**
  - If low level access is allowed to a system it should not be able to subvert or bypass integrity rules to change the data.
  - This can be achieved by some sort of locking or encryption.