Database is where data is stored.

Database system the program that manages the database.

Suppose we have a database where we store artist, album, and year in a csv file.

If we query data from that database, it will be like

Graphical user interface, text, application

Description automatically generated

The above process is bad(inefficient) because querying is linear O(n).

The other problems encountered with the above process are:

* How do we ensure that the artist is the same for each album entry? (Caps, small)
* What if somebody overwrites the album year with an invalid string?
* What if there are multiple artists on an album?
* What happens if we delete an artist that has albums?
* How do you find a particular record?
* What if we now want to create a new application that uses the same database using another programming language?
* What if two threads try to write to the same file at the same time?
* What if the machine crashes while our program is updating a record?
* What if we want to replicate the database on multiple machines for high availability?

A database management system (DBMS) is software that allows applications to store and analyze information in a database.

A general-purpose DBMS supports the definition, creation, querying, update, and administration of databases in accordance with some data model.

A data model is a collection of concepts for describing the data in a database.

A schema is a description of a particular collection of data, using a given data model.

**DATA MODELS**

Relational -Most DBMS use this

Key/ Value -NoSQL

Graph -NoSQL

NoSQL - NoSQL

Document / Object - NoSQL

Wide-Column / Column-family - NoSQL

Array / Matrix / Vectors -Machine Learning

Hierarchical -Obsolete

Network - Obsolete

Multi-Value - Obsolete

**Early DBMS**

Early database applications were difficult to build and maintain on available DBMSs in the 1960s.

Examples: IDS, IMS, CODASYL

Computers were expensive, humans were cheap.

Tight coupling between logical and physical layers.

Programmers had to (roughly) know what queries the application would execute before they could deploy the database.

If we create the database with tree data structure, we cannot write a code based on hashing.

**RELATIONAL MODEL**

The relational model defines a database abstraction based on relations to avoid maintenance overhead.

Key tenets:

Store database in simple data structures (relations).(Tables)

Physical storage left up to the DBMS implementation. (Let the database make decision how to extract data for the query)

Access data through high-level language, DBMS figure out best execution strategy.

**Relational Model**

Structure: The definition of the database's relations and their contents.

Integrity: Ensure the database's contents satisfy constraints.

Manipulation: Programming interface for accessing and modifying a database's contents.

A relation is an unordered set that contains the relationship of attributes that represent entities.

A tuple is a set of attribute values (also known as its domain) in the relation.

Values are (normally) atomic/scalar.

The special value NULL is a member of every domain (if allowed).



Tuple=row=record

**Primary Key**

A relation's primary key uniquely identifies a single tuple.

Some DBMSs automatically create an internal primary key if a table does not define one.

Auto-generation of unique integer primary keys:

SEQUENCE (SQL:2003)

AUTO\_INCREMENT (MySQL)

**Foreign Key**

A foreign key specifies that an attribute from one relation has to map to a tuple in another relation.

**DATA MANIPULATION LANGUAGES (DML)**

Methods to store and retrieve information from a database.

**Procedural:**

The query specifies the (high-level) strategy to find the desired result based on sets / bags.

**Non-Procedural (Declarative):**

The query specifies only what data is wanted and not how to find it.

Relational Algebra

Fundamental operations to retrieve and manipulate tuples in a relation it is based on set algebra.

Each operator takes one or more relations as its inputs and outputs a new relation.

We can "chain" operators together to create more complex operations.

(SELECT, PROJECTION, UNION, INTERSECTION, DIFFERENCE)

**SELECT**

Graphical user interface, text, application

Description automatically generated

**SQL Equivalent**

SELECT \* FROM R WHERE column value=” required value”

**PROJECTION**

Graphical user interface, text

Description automatically generated

Projection means choosing which columns (or expressions) the query shall return. Selection means which rows are to be returned.

**UNION**

Graphical user interface

Description automatically generated with medium confidence

**SQL Equivalent**

(SELECT \* FROM R) UNION ALL (SELECT \* FROM S)

**INTERSECTION**

Graphical user interface

Description automatically generated with medium confidence

**SQL Equivalent**

(SELECT \* FROM R) INTERSECT (SELECT \* FROM S)

**DIFFERENCE**

Text

Description automatically generated with medium confidence

**SQL Equivalent**

(SELECT \* FROM R) EXCEPT (SELECT \* FROM S)

**PRODUCT**

Graphical user interface

Description automatically generated with medium confidence

**SQL Equivalent**

(SELECT \* FROM R) CROSS JOIN S

**JOIN**

A picture containing table

Description automatically generated

**SQL Equivalent**

SELECT \* FROM R JOIN S USING (ATTRIBUTE 1)

Text

Description automatically generated

**RELATIONAL MODEL: QUERIES**

The relational model is independent of any query language implementation.

SQL is the defacto standard (many dialects).