Chapter-1

Introduction:

A database is a set of data organized for easy access. The database is the actual data. It is the database that you will be accessing when you need to retrieve data.

The DBMS is the software & collection of tools that managed the database. Or we can say DBMS is a collection of interrelated data and a set of programs to access those data.

Database systems are designed to manage a large body of information. Management of data involves both defining structures for storage of information and providing mechanisms for the manipulation of information. In addition the database system must ensure the safety of the information stored despite system crashes or attempt to unauthorized access. Because information is so important in most organizations, computer scientists have developed a large body of concepts and techniques for managing data that is what DBMS. Today in most representative applications for example banking, airlines, universities, credit card transaction, telecommunication, finance, sales, manufacturing, human resources etc.

A RDBMS (relational database management system) in a DBMS that is relational in nature. This means the internal working access data in relational manner. Oracle is an RDBMS. The features for data storage and retrieval are very advanced in RDBMS. This system is based on set theory. The RDBMS ensured that the data stored in the database is accurate and relevant. Excellent security features are offered by this system.

DBMS V/s. RDBMS:

DBMS	RDBMS
In DBMS relationship between two table or files are maintained programmatically.	In RDBMS relationship between two tables or files can be specified at the time of table creation.
DBMS cannot support Client/Server architecture.	Most of the RDBMS support Client/Server architecture.
DBMS does not support Distributed Database.	Most of the RDBMS support Distributed database.
In DBMS there is no security of data.	In RDBMS there are multiple level Security. 1] Logging in at O/S level. 2] Command level 3] Object Level
Each table in DBMS is given extension.	Many tables are grouped in one table in RDBMS.
Naming conventions used in DBMS	Naming conventions used in RDBMS
Field	Column, Attributes
Record	Row, Tuple, Entity
File	Table, Relation, Entity class

Characteristics of RDBMS:

The relational data management model eliminated all parent child relationship and instead represented all data in the database as simple row/column tables of data values.

A relation is similar to a table with rows/columns of data values. The rows of a table are referred to as tuple and the columns are referred to an attribute. Several tulples of equal length placed one below the other create a table.

- Write intensive operations:
  The RDBMS is frequently written to and its often used in transaction oriented application.

- Data in flux or historical data:
  The RDBMS is designed to handle frequently changing data. Alternatively a RDBMS can also store vast amounts of historical data, which can later be analyzed.

- Application specific schema:
  The RDBMS is configured on a per application basis and a unique schema exists to support each application.

- Complex data models:
  The relational nature of the RDBMS makes it suitable for handling sophisticated, complex data models that require many tables, foreign key values, complex join operations and so on.

- Data integrity:
  The RDBMS features many components designed to ensure data integrity. This includes rollback operations, referential integrity and transaction-oriented operations.

- ACID (**A**tomic, **C**onsistent, **I**solation, **D**urable) transactions:
  RDBMS supports ACID property.
  1. Atomic: atomic transactions consist of grouping of changes to tables or rows such that all or none of the changes take place. A rollback operation can reverse all the actions of the atomic transaction.
  2. Consistent: Transactions operate on a consistent view of the data. When the transaction is completed, the data is left in consistent state.
  3. Isolation: Transactions run isolated from other transactions. So if transactions are running concurrently, the effects of transaction are invisible to other transaction and vice-versa until the transaction is complete.
  4. Durable: Upon commitment of the transaction, its changes are guaranteed. Until the transaction commits, none of its actions are durable or persistent. If the system crashes prior to a commit the effects of the transaction will be rolled back.

Rules of RDBMS: (CODD’s Rules)/ Dr. E. F. Codd`s Rules

Information Representation:

All information in a relational database (including table and column names) is represented explicitly as values in tables.

Guaranteed access rule:

All data should be accessible without ambiguity (duplicate record). This can be accomplished through a combination of the table name, primary key & column name.

Systematic treatment of NULL values:

A field should be allowed to remain empty. This involves support of NULL value, which is distinct from an empty string or a number with a value of zero. Of course this can’t apply to primary keys, in addition most database implementations supports the concept of a not null field constraint that prevents null values in a specific table column.

Catalog facilities:

The description of the database and its contents is represented at the logical level as tables and can therefore be queried using the database language.

Data sub-language:

The database must support at least one clearly define language that includes functionality for data definition, data manipulation, and data integrity and database transaction control. All commercial relational databases use forms of standard SQL as their supported comprehensive language.

View updating:

Data can be presented in different logical combinations called views. Each view should support the same full range of data manipulation that has direct access to a table available. In practice providing update and delete access to logical views is difficult and us not fully supported by any a current database.

Insert, Update, Delete:

Data can be retrieved from a relational database in sets constructed of data from multiple rows/or multiple tables. This rule states that insert, update and delete operation should be supported for any retrievable set rather than just for a single row in a single table.

Physical data independence:

Application programs and ad hoc programs are logically unaffected when physical access methods or storage structures are altered.

Logical data independence:

How data is viewed should not be changed when the logical structure of the database changes. This rule is particularly difficult to satisfy. Most databases rely on strong ties between the data view and the actual structure of the underlying tables.

Integrity independence:

The database language (SQL) should support constraints on user input that maintain database integrity. This rule is not fully implemented by most major vendors. At a minimum all databases do preserve tow constraints through SQL. No component of a primary key can have a null value. If a foreign key is defined in one table, any value in it must exist as a primary key in another table.

Distribution independence:

A user should be totally unaware of whether or not the database is distributed. A variety of reasons make this rule difficult to implement. Application programs and any query must logically unaffected when data is distributed.

Non sub-version rules:

There should be no way to modify the database structure other than through the multiple rows DB language. Most DB today support administrative tools allow some direct manipulation of the data structure.

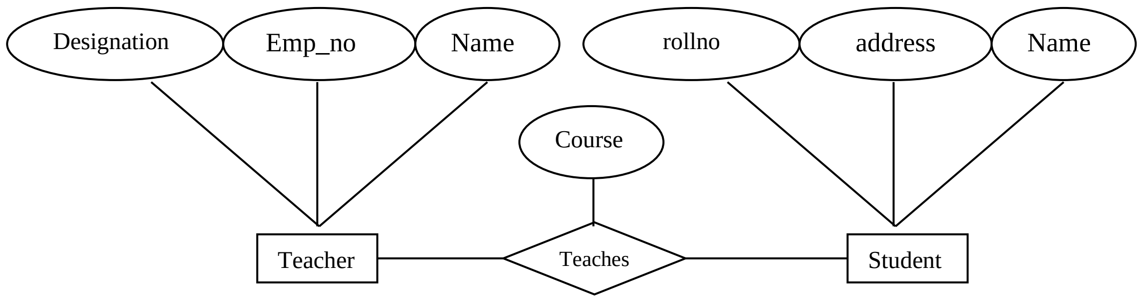
E-R Diagram (Entity Relation Diagram):

Definition:

An E.R. Diagram can express the overall logical structure of a database graphically. E-R diagram provides clear and simple view of database structure.

The diagram consist of the following major component:

1] **Rectangle**: which represents the entity set.
2] **Ellipse**: which represents the attributes.
3] **Diamond**: which represents relationship set.
4**] Line**: which links attributes to the entity set & the entity set to relationship set.



The transformation of the above E.R. diagram to the relational model consists of three relations give below:

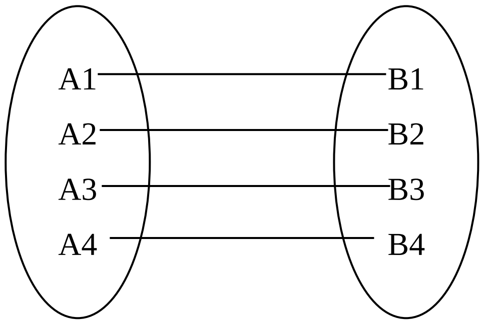
Teacher: Emp_no, Name, Designation   Student: Roll_no, Name, address

Teaches: Emp_no, Roll_no, Course

Relationship:

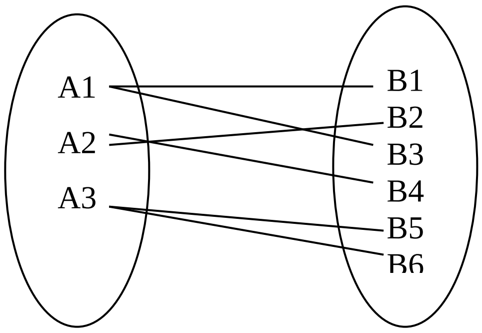
One-to-One Relationship:

In a one to one relationship each record in table ’A’ can have only one matching record in table ‘B’ and each record in table ‘B’ can have only matching record in table ‘A’. This type of relationship is not common, because most information related in this way would be in one table. You might use a one to one relationship to divide a table with many fields, to isolated part of a table for security reasons. Or to store information that applies only to subset of the main table.



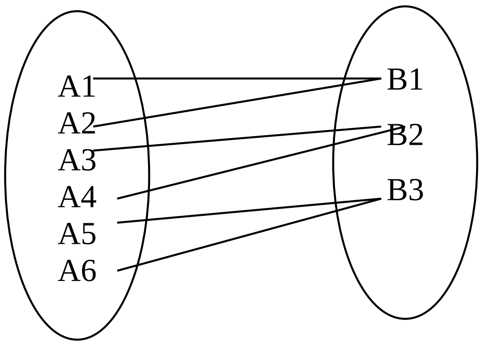
One-to-Many Relationship:

A one to many relationships is that most common type of relationship. The one to many relationship describe a situation in which every record in a table many be related to many records in another table.



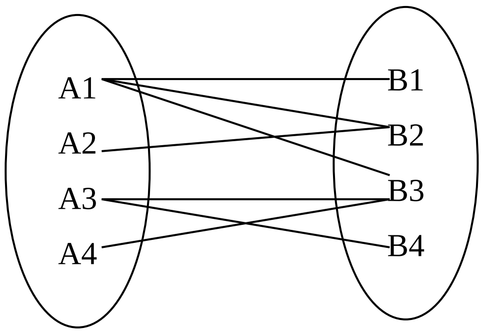
Many-to-One Relationship:

An entity ‘A’ is associated with at most one entity of ‘B’ but each entity of ‘B’ may associated with more then one entities of ‘B’.



Many-to-Many Relationship:

A relationship between tables where, records in each table have multiple marching records in the related table. In another word you can say in a many to many relationship a record in table ‘A’ can have many matching records in table ‘B’ & a record in table ‘B’ can have many marching records in table ’A’. This type of relationship is only possible by defining a third table, which is called the junction table that consists of two field as the foreign key from both tables ‘A’ and ‘B’.



Justify oracle as RDBMS:

According to fundamental definitions of RDBMS a DBMS may considered as relational if it satisfy following three rules.

- All information must be held in tables
- Retrieval of the data must be possible using the following types of operations such as select, join etc.
- All relationships between data must be represented explicitly in that data itself.

But these are the minimum requirements of DBMS to be relational. To define the requirements more accurately, following 12 rules must be considered.

Information Representation:

All information in a relational database (including table and column names) is represented explicitly as values in tables.

Guaranteed access rule:

All data should be accessible without ambiguity (duplicate record). This can be accomplished through a combination of the table name, primary key & column name.

Systematic treatment of NULL values:

A field should be allowed to remain empty. This involves support of NULL value, which is distinct from an empty string or a number with a value of zero. Of course this can’t apply to primary keys, in addition most database implementations supports the concept of a not null field constraint that prevents null values in a specific table column.

Catalog facilities:

The description of the database and its contents is represented at the logical level as tables and can therefore be queried using the database language.

Data sub-language:

The database must support at least one clearly define language that includes functionality for data definition, data manipulation, and data integrity and database transaction control. All commercial relational databases use forms of standard SQL as their supported comprehensive language.

View updating:

Data can be presented in different logical combinations called views. Each view should support the same full range of data manipulation that has direct access to a table available. In practice providing update and delete access to logical views is difficult and us not fully supported by any a current database.

Insert, Update, Delete:

Data can be retrieved from a relational database in sets constructed of data from multiple rows/or multiple tables. This rule states that insert, update and delete operation should be supported for any retrievable set rather than just for a single row in a single table.

Physical data independence:

Application programs and ad hoc programs are logically unaffected when physical access methods or storage structures are altered.

Logical data independence:

How data is viewed should not be changed when the logical structure of the database changes. This rule is particularly difficult to satisfy. Most databases rely on strong ties between the data view and the actual structure of the underlying tables.

Integrity independence:

The database language (SQL) should support constraints on user input that maintain database integrity. This rule is not fully implemented by most major vendors. At a minimum all databases do preserve tow constraints through SQL. No component of a primary key can have a null value. If a foreign key is defined in one table, any value in it must exist as a primary key in another table.

Distribution independence:

A user should be totally unaware of whether or not the database is distributed. A variety of reasons make this rule difficult to implement. Application programs and any query must logically unaffected when data is distributed.

Non sub-version rules:

There should be no way to modify the database structure other than through the multiple rows DB language. Most DB today support administrative tools allow some direct manipulation of the data structure.

Oracle history

Here is the list of important events that took place in the last 30 years of Oracle history. Oracle is celebrating its 30 wonderful years in the industry (from 1977 to 2007).

This article was updated on **6th August, 2007.**

- **In 1977**, **Larry Ellison, Bob Miner** and **Ed Oates** founded **Software Development Laboratories** to undertake development work
- After reading a paper by **Codd** in IBM Journal of Research and Development, they created **Oracle**. The first version was never released. It was written in Assembly language of (*Programmed Data Processor)* PDP which ran in 128kb of RAM.
-
- Version **2.0** of Oracle was released in **1979** and it became first commercial relational database and first SQL database. The company changed its name to **Relational Software Inc**. (RSI).
- **In 1981**, RSI started developing tools for Oracle.
- **In 1982**, RSI was renamed to **Oracle Corporation**. It held its first user conference in San Francisco.
- **In 1983**, Oracle released version **3.0**, which was **rewritten in C** language and ran on multiple platforms.
- **In 1984**, Oracle version **4.0** was released. It contained features like **concurrency** control - multi-version read consistency etc.
- **By 1985**, Oracle released version **5.0** and became the first relational database that worked in **client/server** environment.

- **In 1986**, Oracle goes public on the **NASDAQ** exchange.

- **In 1987**, Oracle wanted to create **enterprise applications** that take advantage of their database.

- **In 1988**, Oracle version **6.0** was released. It provided **row-level locking, hot backup** and **PL/SQL** as main features.

- **By 1989,** Oracle moved to new headquarters in **Redwood Shores**, **California**.

- **In 1990**, they released **Oracle Applications Release 8**, which included account software for client/server

- **In 1992,** they released **Oracle 7.0**. It provided better performance, administrative utilities, **application development tools**, security features, **stored procedures**, **triggers** and **declarative integrity**.

- **In 1995**, Oracle became the first major company to announce a **comprehensive internet strategy**.

- **In 1997,** Oracle released **Oracle 8.0** and **Oracle Applications 10.7**. It started embracing **Java**. **Partitioning**, support for different types of data like images, large text, external data etc.(**lobs**) are provided. It also started providing support for Object in database becoming an **Object-Relational DBMS**

- **By 1999**, Oracle realized that "**Internet Changes Everything**". Oracle released **Oracle 8i** and **Oracle Applications 11i**. They supported open standards like XML. Oracle8i provided Java Virtual Machine (JVM) to run Java program in Oracle Database and also scalability, which was required for internet databases.
  To read more about what is new in Oracle8i, read What's new in Oracle8i and PL/SQL Enhancements articles.

- **In 2000**, **Oracle9iAS** was released. Oracle corporation is no longer a company providing only database management system and instead started providing all that it takes to develop and deploy a complete application. AS(Application Server) runs on middle tier in 3-tier Client/Server architecture boosting the performance.

- **In 2001, Oracle9i** was released. It allows Oracle to run on **RAC** (Real Application Cluster), which is a collection of low-cost servers. It also allowed XML documents to be stored and queried in Oracle Database.
  You can get information about new features that were introduced in Oracle9i using New features of Oracle9i Database
- **In 2003, Oracle10g** was released, where g stands for Grid computing, which servers computing power across the enterprise as a utility, automatically shifting

processing load based on demand. Oracle10g also made a lot of administrative tasks automatic.

- **In 2007,** Oracle has released **Oracle11g.** The new version focused on better partitioning, easy migration etc.

****