Database Model

- A Database model defines the logical design of data.
- The model describes the relationships between different parts of the data.
- Historically, in database design, *3 models are commonly used*. They are,

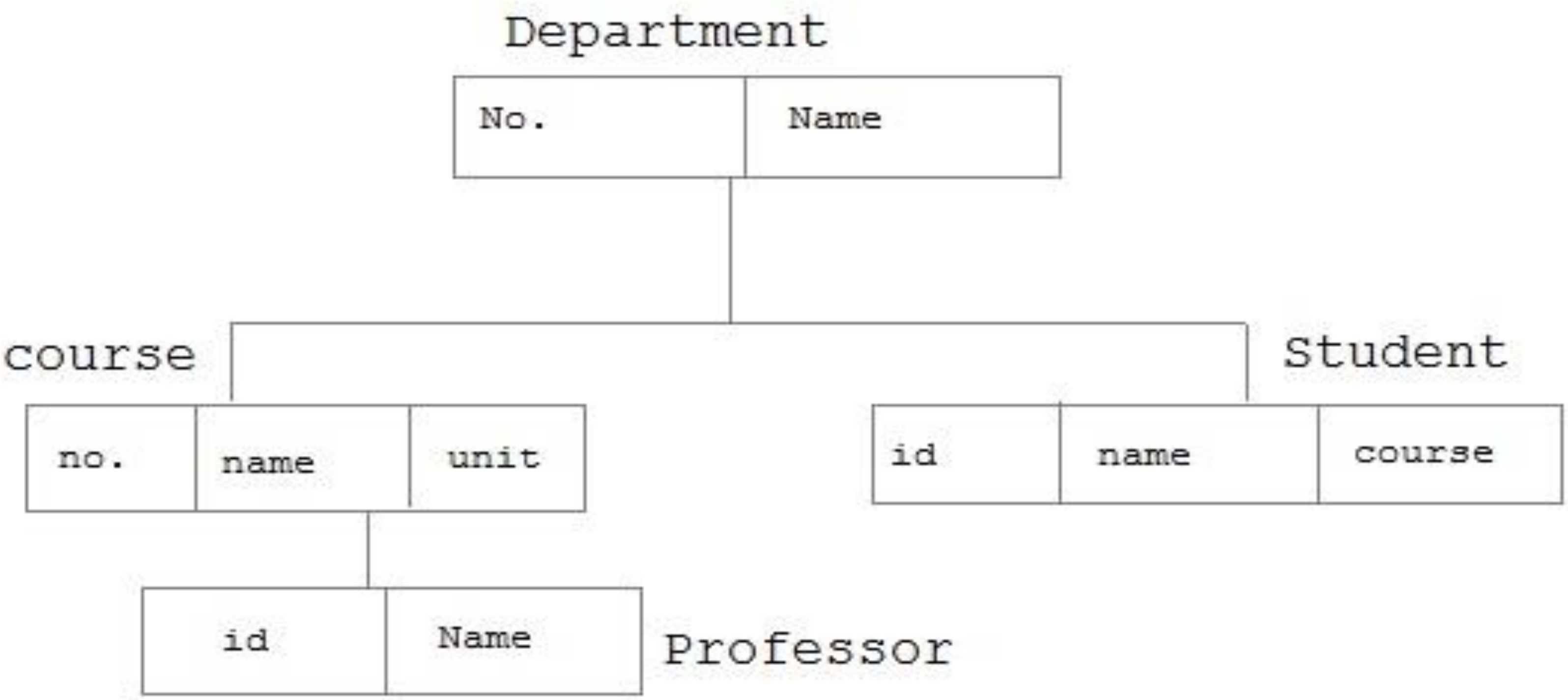
I. Hierarchical Model

II. Network Model

III. Relational Model

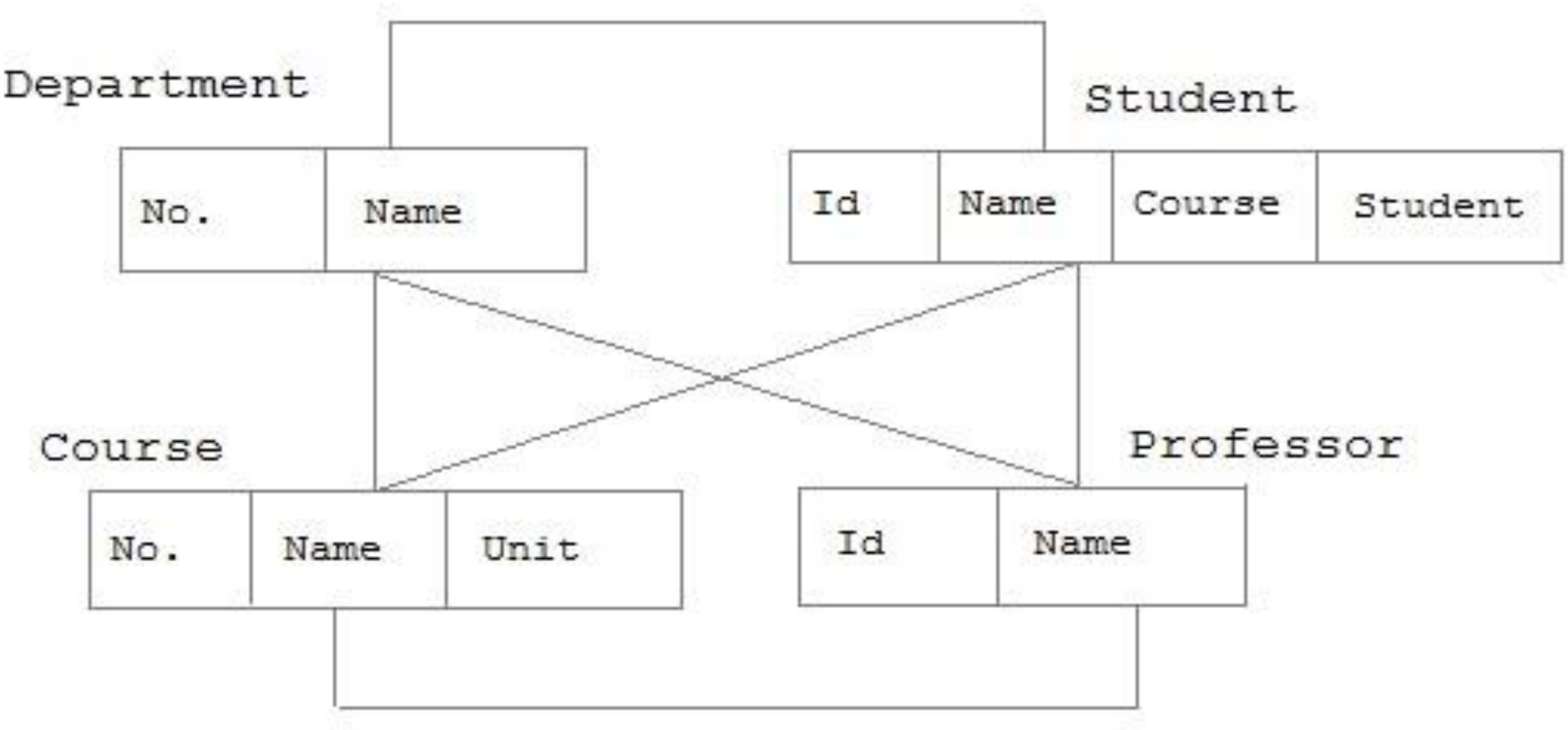
Hierarchical Model

- In this model, *each entity has only one parent* but can have *several children* . At the top of hierarchy there is only one entity which is called **Root**.



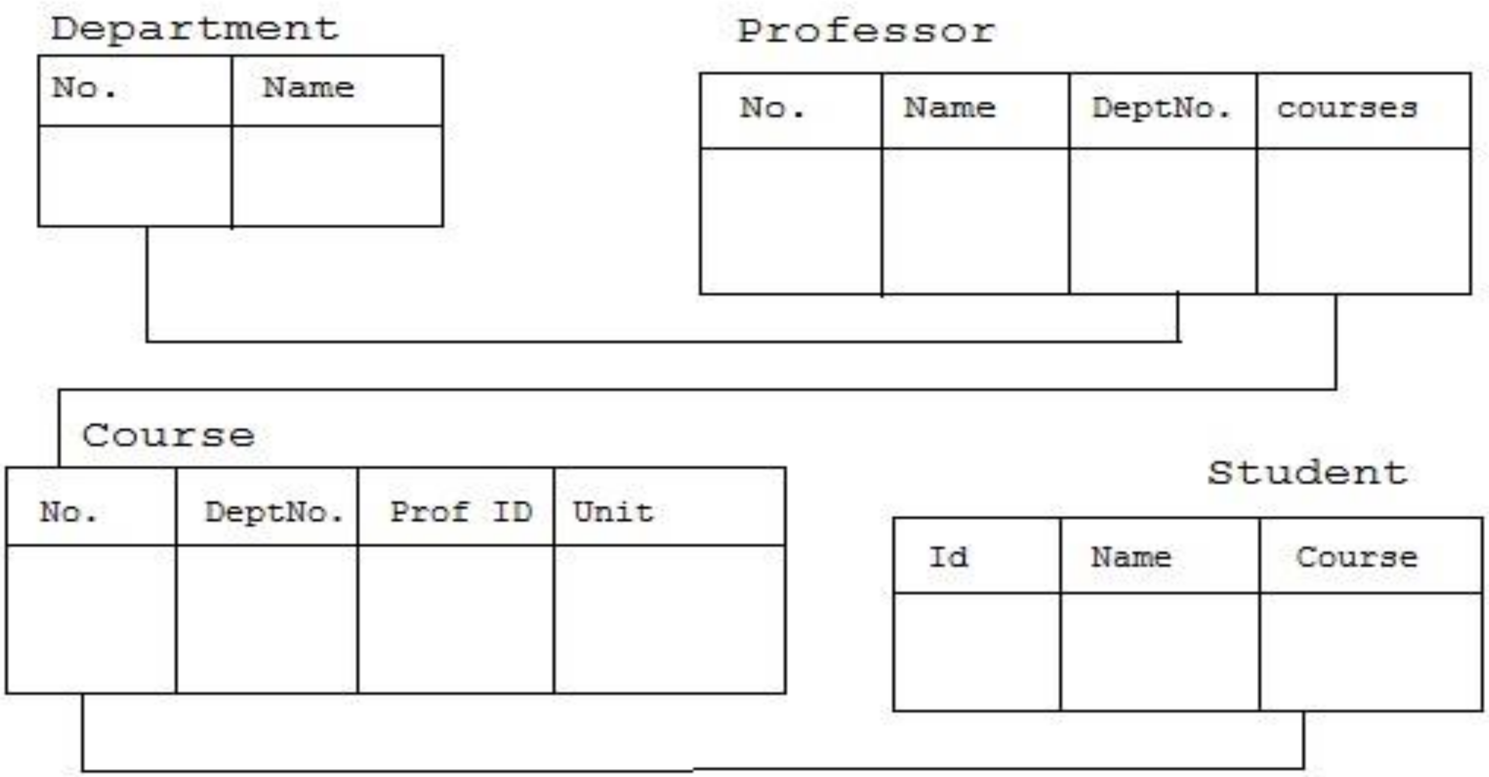
Network Model

- In this model, entities are ***organized in a graph*** , in which some entities can be accessed through several path.



Relational Model

- In this model, data is organized in two-dimensional tables called ***relations***. The tables or relation are related to each other.



Department

No.	Name

Professor

No.	Name	DeptNo.	courses

Course

No.	DeptNo.	Prof ID	Unit

Student

Id	Name	Course

RDBMS

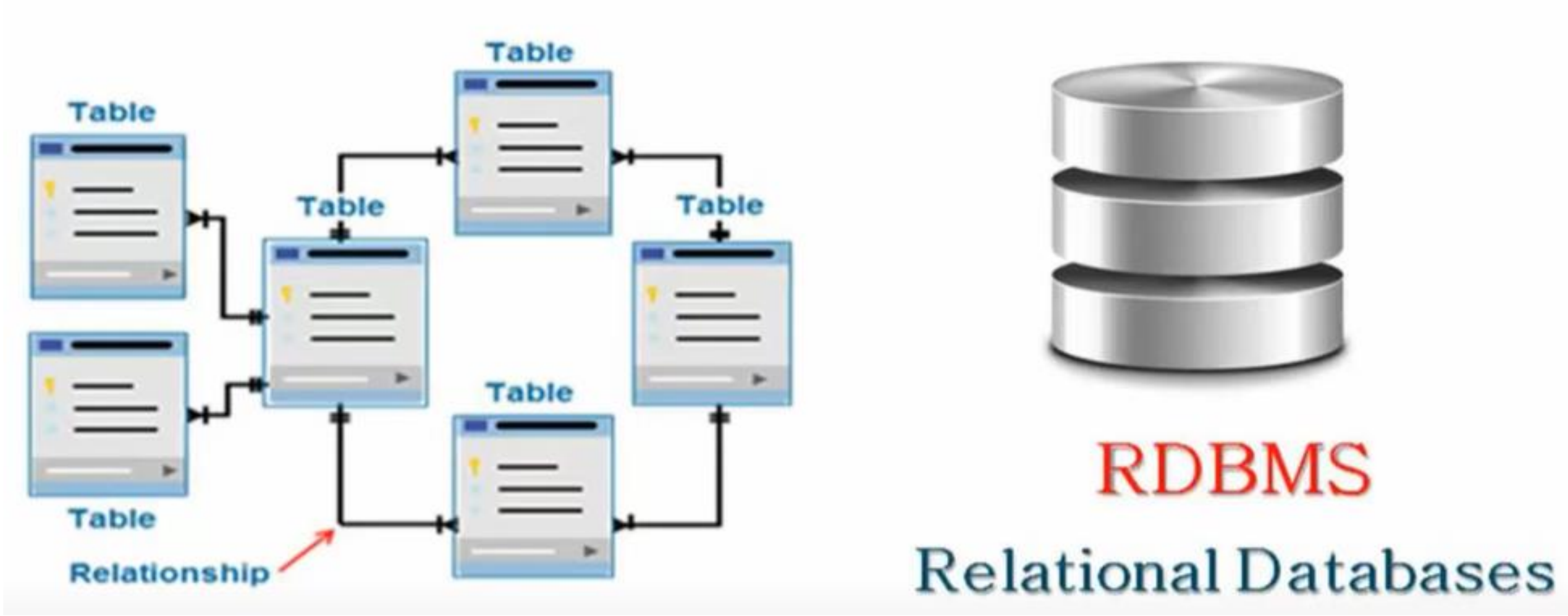
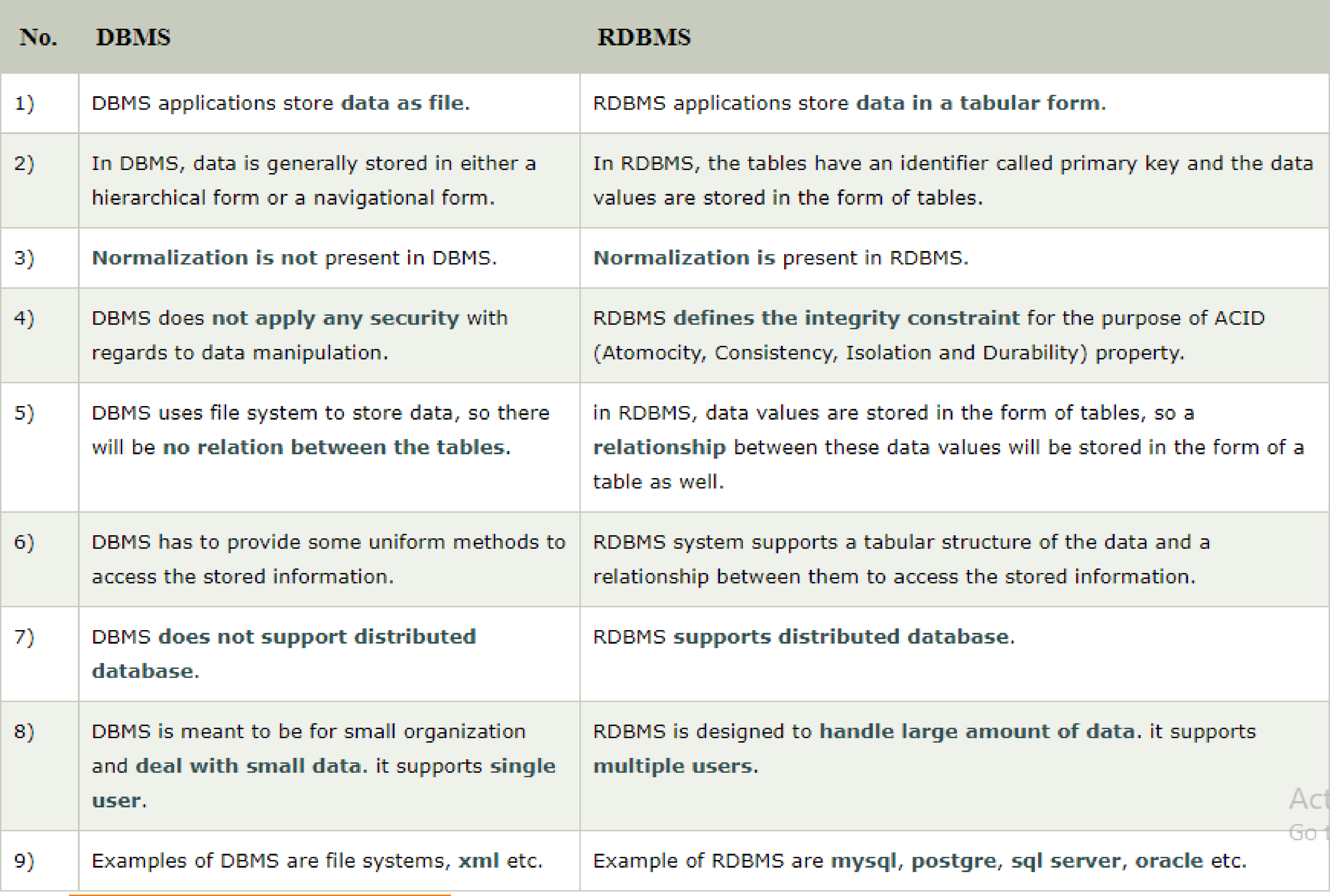


Table
Table
Table
Table
Table
Table
Table
Relationship
RDBMS
Relational Databases

Introduction

- The Relational Database Management Systems(RDBMS) are based on relational Model.
- It is prescription for how to represent and manipulate data.
- All modern database management systems like SQL, MS SQL Server, IBM DB2, ORACLE, My-SQL and Microsoft Access are based on RDBMS.
- It is called Relational Data Base Management System (RDBMS) because it is *based on relational model introduced by* ***E.F. Codd***.
- In Relational DBMS, the values of each table are related to others. It has the capability to handle larger magnitudes of data and simulate queries easily.



No.	DBMS	RDBMS
1)	DBMS applications store **data as file.**	RDBMS applications store **data in a tabular form.**
2)	In DBMS, data is generally stored in either a hierarchical form or a navigational form.	In RDBMS, the tables have an identifier called primary key and the data values are stored in the form of tables.
3)	**Normalization is not** present in DBMS.	**Normalization is** present in RDBMS.
4)	DBMS does **not apply any security** with regards to data manipulation.	RDBMS **defines the integrity constraint** for the purpose of ACID (Atomocity, Consistency, Isolation and Durability) property.
5)	DBMS uses file system to store data, so there will be **no relation between the tables.**	in RDBMS, data values are stored in the form of tables, so a **relationship** between these data values will be stored in the form of a table as well.
6)	DBMS has to provide some uniform methods to access the stored information.	RDBMS system supports a tabular structure of the data and a relationship between them to access the stored information.
7)	DBMS **does not support distributed database.**	RDBMS **supports distributed database.**
8)	DBMS is meant to be for small organization and **deal with small data.** it supports **single user.**	RDBMS is designed to **handle large amount of data.** it supports **multiple users.**
9)	Examples of DBMS are file systems, **xml** etc.	Example of RDBMS are **mysql, postgre, sql server, oracle** etc.

RDBMS Components

- More precisely the relational model is concerned with three components:

1. Data Structure
2. Data Integrity
3. Data Manipulation

I- RELATIONAL DATA STRUCTURE

Relational Data Structure

1. **Entity:** An entity can be a real-world object, either animate or inanimate, that can be easily identifiable. For example, in a school database, students, teachers, classes, and courses offered can be considered as entities.

2. **Relation:** A Relation is a table with rows and columns . All data and relationships are represented in 2D table called relation. RDBMS requires only the user perceive data as table .

3. **Attribute**: An attribute is a named column of a relation. Attributes can appear in any order and the relation will still be the same relation , and therefore convey same meaning.

4. Domain: A domain is the set of allowed values for one or more attributes.

A domain is the set of all possible values that an attribute may validly contain. Each attribute in the model should be assigned domain information that includes:

i. *Data Type*: Basic data type as integer, decimal or character. Most databases support variants of these.

ii. *Length:* this is the number of digits or characters in the values.

iii. *Date format:* the format of date value as dd/mm/yy or mm/dd/yy

iv. Range: the range specifies the lower and upper boundaries of the values of attributes may legally have.

v. *Constraints:* are special restrictions on allowed values

vi. *Null Support:* Indicates whether the attribute can have null or unknown value.

vii. *Default Value:* the value an attribute will have if a value is not entered.

5. **Tuple:** A tuple is a row of a relation.

6. **Extension of a Relation:** It is the set of tuples appearing in that relation at any given instant of time . The extension thus varies with time.

7. **Intension of a Relation :** It is the permanent part of the relation and independent of time , it correspond to what is specified in the relational schema.

8. **Degree :** the degree of relation is the number of attributes it contains.

9. **Cardinality :** The cardinality of a relation is the number of tuple(rows) it contains.

II-DATA INTEGRITY

- *"Data integrity"* refers to the accuracy and consistency of data stored in a database, data warehouse, data mart or other construct.
- The term Data Integrity can be used to describe a state, a process or a function – and is often *used as a proxy for "DATA QUALITY".*
- In order to discuss data integrity ,we have to understand keys (which have been covered in data modeling chapter)
- There should not be any duplicate tuple within a relation, therefore we should identify one or more attributes(called relational keys) that uniquely identify each tuple in a relation.

III-DATA MANIPULATION

Types of DBMS languages

Data Definition Language (DDL)

- It is used to define database structure or pattern.
- It is used to create schema, tables, indexes, constraints, etc. in the database.
- Using the DDL statements, you can create the skeleton of the database.
- Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

- To create the database instance – **CREATE**
- To alter the structure of database – **ALTER**
- To drop database instances – **DROP**
- To delete tables in a database instance – **TRUNCATE**
- To rename database instances – **RENAME**

All these commands specify or update the database schema that’s why they come under Data Definition language.

Data Manipulation Language (DML):

- It is used for accessing and manipulating data in a database.
- It handles user requests.

- Here are some tasks that come under DML:

a. To read records from table(s) – **SELECT**

b. To insert record(s) into the table(s) – **INSERT**

c. Update the data in table(s) – **UPDATE**

d. Delete all the records from the table – **DELETE**

Data Control language (DCL):

- DCL is used for granting and revoking user access on a database.

a. To grant access to user – **GRANT**

b. To revoke access from user – **REVOKE**

- There are the following operations which have the authorization of Revoke:

  CONNECT, INSERT, USAGE, EXECUTE, DELETE, UPDATE and SELECT.

- In practical data definition language, data manipulation language and data control languages are not separate language; rather they are the parts of a single database language such as SQL.

Transaction Control Language:

- It manage the changes made by DML .
- Here are some tasks that come under TCL:

a. **COMMIT** -make transaction changes permanent

b. **ROLLBACK** -undo changes made by transaction either since it started or since save point

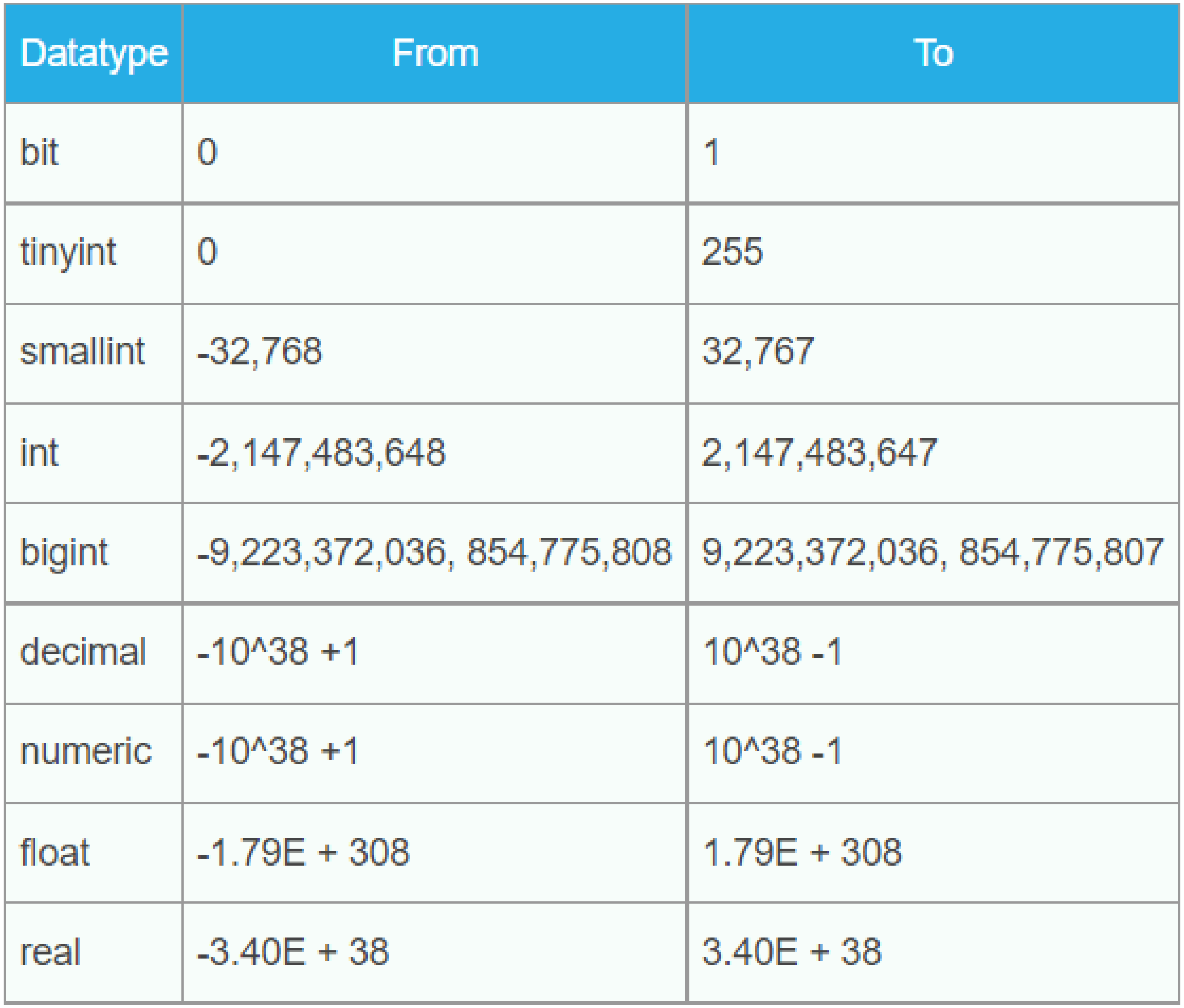
c. **SAVEPOINT** -set point to which transaction can be rolled back

d. **SET TRANSACTION** -establish properties for transaction

Data types and Operators

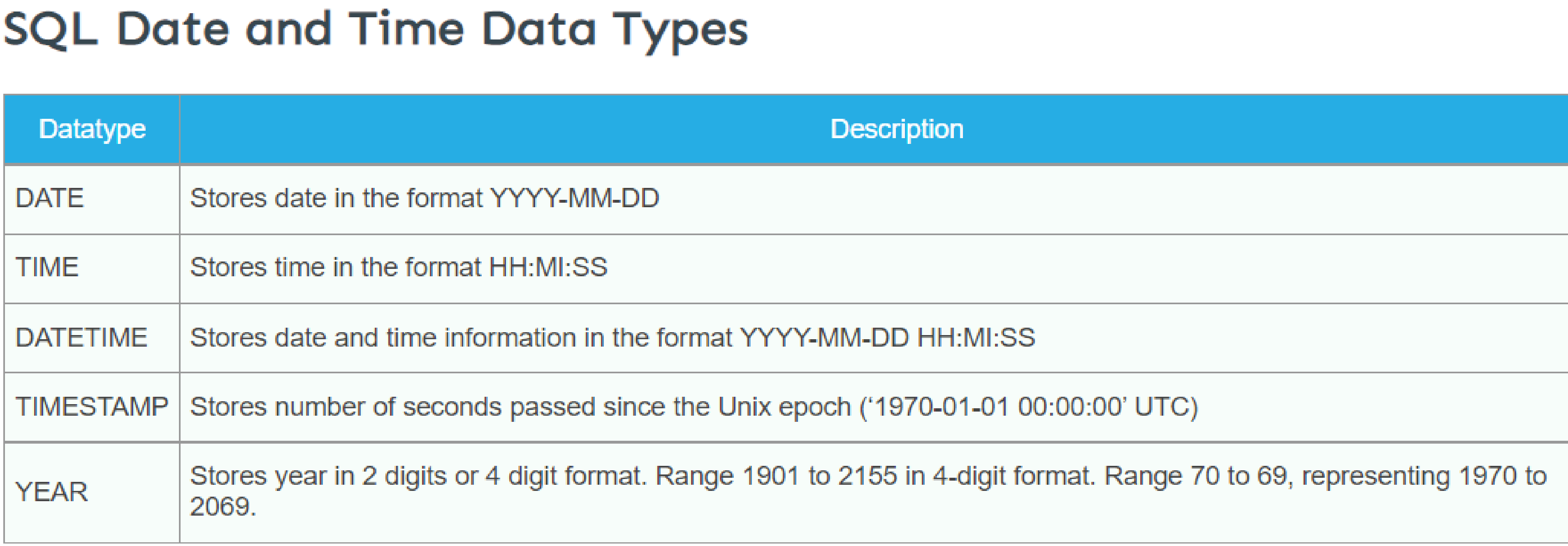
Data-type	Syntax	Explanation
Integer	INTEGER	The integer data type is used to specify an integer value.
Smallint	SMALLINT	The smallint data type is used to specify small integer value.
Number	NUMBER(P)	It specifies a numeric value. Here 'p' is precision value and 's' is scale value.
Real	REAL	The real integer is used to specify a single precision floating point number.
Decimal	DECIMAL(P,S)	It specifies a decimal value. Here 'p' is precision value and 's' is scale value.

SQL Numeric Data Types



Datatype	From	To
bit	0	1
tinyint	0	255
smallint	-32,768	32,767
int	-2,147,483,648	2,147,483,647
bigint	-9,223,372,036, 854,775,808	9,223,372,036, 854,775,807
decimal	-10^38 +1	10^38 -1
numeric	-10^38 +1	10^38 -1
float	-1.79E + 308	1.79E + 308
real	-3.40E + 38	3.40E + 38

Double precision	DOUBLE PRECISION	It specifies double precision floating point number.
Float	FLOAT(P)	It specifies floating-point value e.g. 12.3, 4.5 etc. Here, 'p' is precision value.
Character	CHAR(X)	Here, 'x' is the character's number to store.
Character varying	VARCHAR2(X)	Here, 'x' is the character's number to store
Bit	BIT(X)	Here, 'x' is the number of bits to store
Bit varying	BIT VARYING(X)	Here, 'x' is the number of bits to store (length can vary up to x).
Date	DATE	It stores year, month and days values.
Time	TIME	It stores hour, minute and second values
Timestamp	TIMESTAMP	The timestamp data type is used to store year, month, day, hour, minute and second values.
Time with time zone	TIME WITH TIME ZONE	It is exactly same as time but also store an offset from UTC of the time specified.
Timestamp with time zone	TIMESTAMP with TIME ZONE	It is same as timestamp but also stores an offset from UTC of the time specified.



SQL Date and Time Data Types

Datatype	Description
DATE	Stores date in the format YYYY-MM-DD
TIME	Stores time in the format HH:MI:SS
DATETIME	Stores date and time information in the format YYYY-MM-DD HH:MI:SS
TIMESTAMP	Stores number of seconds passed since the Unix epoch (‘1970-01-01 00:00:00’ UTC)
YEAR	Stores year in 2 digits or 4 digit format. Range 1901 to 2155 in 4-digit format. Range 70 to 69, representing 1970 to 2069.

Operator	Description	Example
-	It subtracts right hand operand from left hand operand	a-b will give -50
*	It multiply both operand?s values	a*b will give 5000
/	It divides left hand operand by right hand operand	b/a will give 2
+	It is used to add containing values of both operands	a+b will give 150
%	It divides left hand operand by right hand operand and returns reminder	b%a will give 0

Operator	Description	Example
=	Examine both operands value that are equal or not,if yes condition become true.	(a=b) is not true
!=	This is used to check the value of both operands equal or not,if not condition become true.	(a!=b) is true
< >	Examines the operand?s value equal or not, if values are not equal condition is true	(a<>b) is true
>	Examine the left operand value is greater than right Operand, if yes condition becomes true	(a>b) is not true
<	Examines the left operand value is less than right Operand, if yes condition becomes true	(a<="" td="">
>=	Examines that the value of left operand is greater than or equal to the value of right operand or not,if yes condition become true	(a>=b) is not true
<=	Examines that the value of left operand is less than or equal to the value of right operand or not, if yes condition becomes true	(a<=b) is true
!<	Examines that the left operand value is not less than the right operand value	(a!<="" td="">
!>	Examines that the value of left operand is not greater than the value of right operand	(a!>b) is true

Operator	Description
ALL	this is used to compare a value to all values in another value set.
AND	this operator allows the existence of multiple conditions in an SQL statement.
ANY	this operator is used to compare the value in list according to the condition.
BETWEEN	this operator is used to search for values, that are within a set of values
IN	this operator is used to compare a value to that specified list value
NOT	the NOT operator reverse the meaning of any logical operator
OR	this operator is used to combine multiple conditions in SQL statements
EXISTS	the EXISTS operator is used to search for the presence of a row in a specified table
LIKE	this operator is used to compare a value to similar values using wildcard operator