



RDBMS Concept



Topics

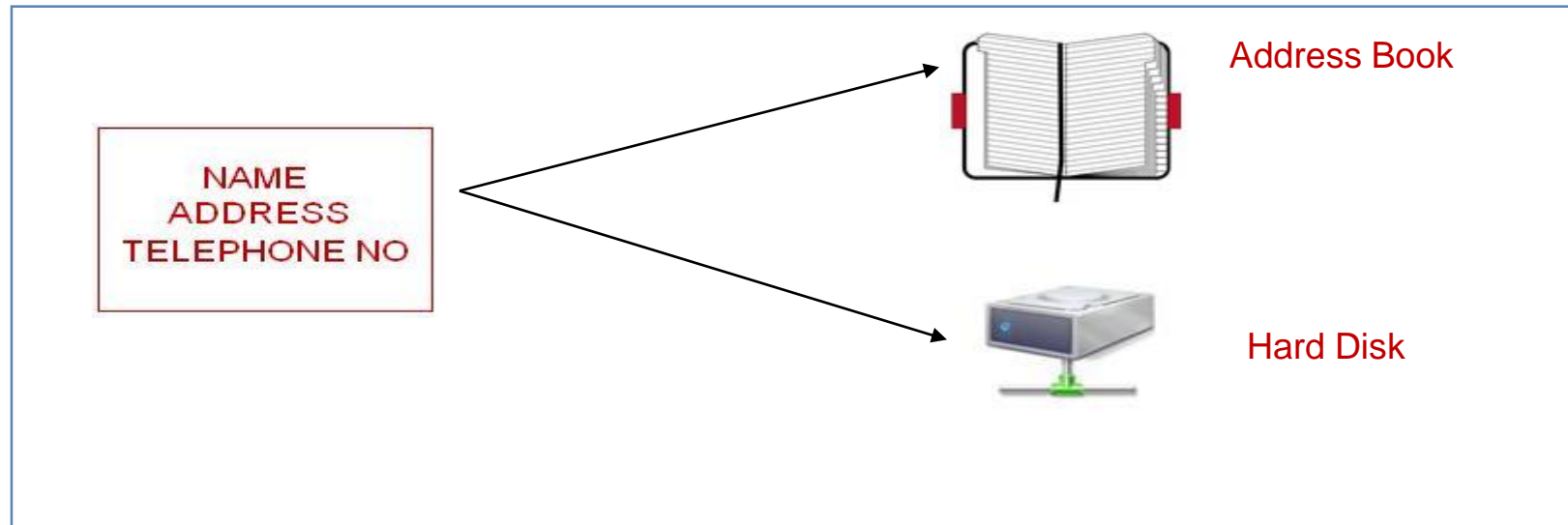
- What is Database?
- What is DBMS?
- RDBMS Concepts
- Data Model
- Keys
- Normalization





What is a Database?

- A Database is a collection of related data organized in a way that data can be easily accessed, managed and updated. Any piece of information can be a data, for example name of your school. Database is actually a place where related piece of information is stored and various operations can be performed on it.





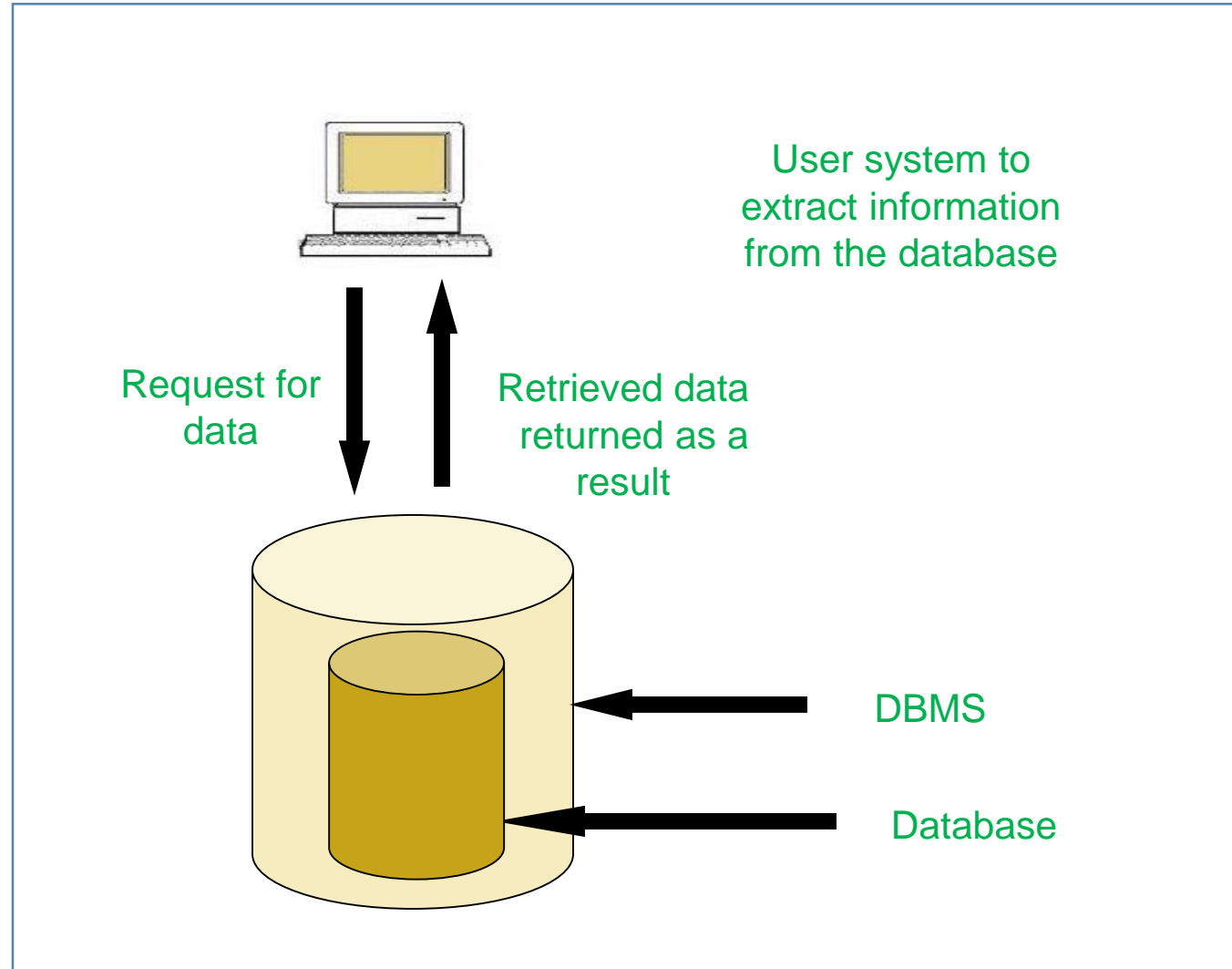
What is a DBMS?

- A DBMS is a software that allows creation, definition and manipulation of database. DBMS is actually a tool used to perform any kind of operation on data in database. DBMS also provides protection and security to database. It maintains data consistency in case of multiple users. Here are some examples of popular dbms, MySql, Oracle, Sybase, Microsoft Access and IBM DB2 etc.





Database Management System





RDBMS Concepts

- A Relational Database management System(RDBMS) is a database management system based on relational model introduced by E.F Codd. In relational model, data is represented in terms of tuples(rows).
- RDBMS is used to manage Relational database. Relational database is a collection of organized set of tables from which data can be accessed easily. Relational Database is most commonly used database. It consists of number of tables and each table has its own primary key.





RDBMS Concepts

- Different Objects of RDBMS Structure :
 1. Database
 2. Table
 3. Field
 4. Record





RDBMS Concepts

Database Object	Explanation
Database	A database is an organized collection of data for one or more purposes, usually in digital form. (e.g College Database)
Table	Table is a set of data elements (values) that is organized using a model of vertical columns (which are identified by their name) and horizontal rows.
Field	Individual Column of the database table is called as "Field"
Record	Individual Row of the database table is called as "Record"





What is Data Model?

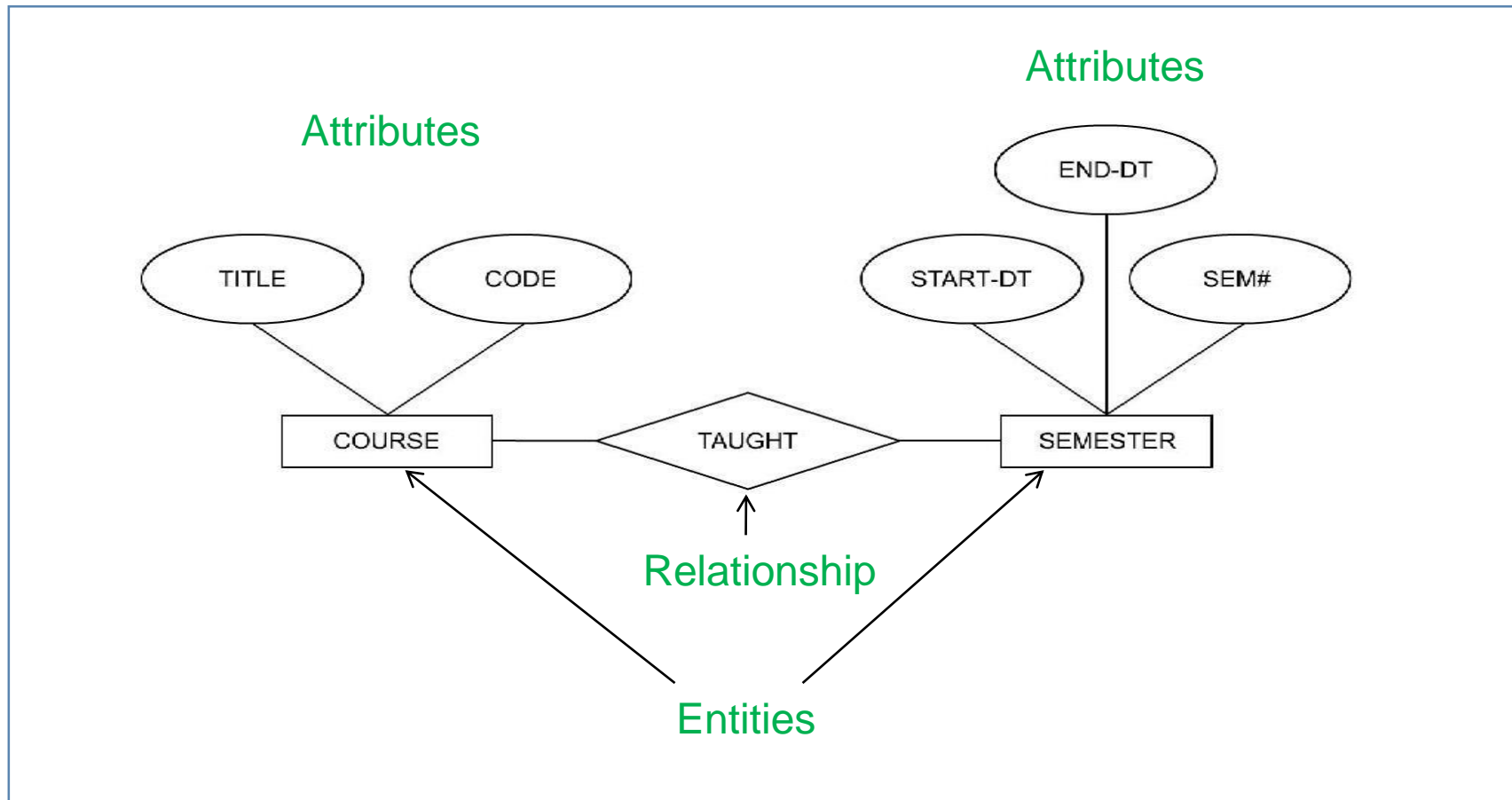
- A data model is a description of the data in a database.
- Data models can be broadly classified into the following categories:
 1. Object-based logical model: Focuses on describing the data, the relationship among the data, and any constraints defined.
 2. Record-based logical model: Focuses on specifying the logical structuring of the database.





Object-based logical model

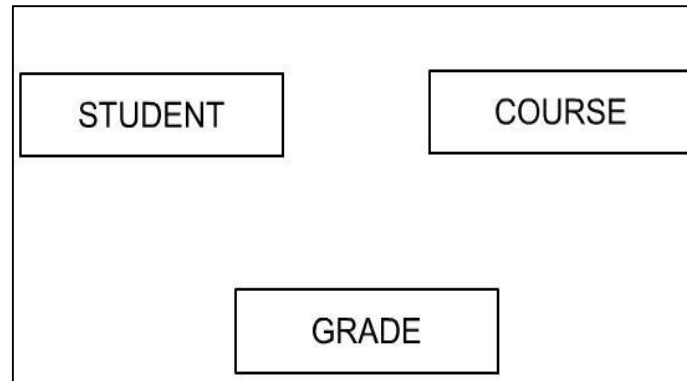
- The ER model: Views the real world as a collection of objects or entities and the relationship among them. Has a corresponding diagramming technique.





What is Data Model

- An entity:
 1. Is any object, place, person, or activity about which the data is recorded.
 2. Can be categorized as entity type and entity instance.
 3. In the ER model diagramming technique, entities are named and represented inside a box.
 4. For example:

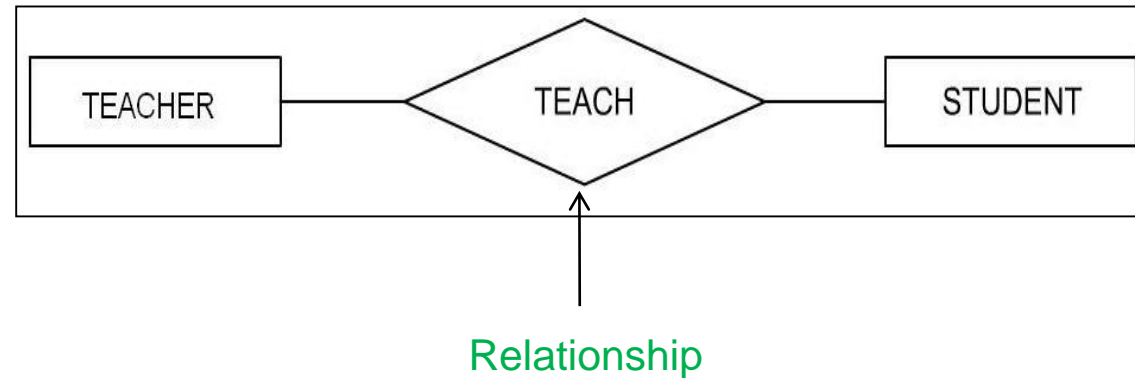




What is Data Model?

- A Relationship

- Chen defined a relationship as “an association among entities”.
- A relationship is depicted as a diamond with the name of the relationship type.
- For example:



- 3 types :

1. One-to-one

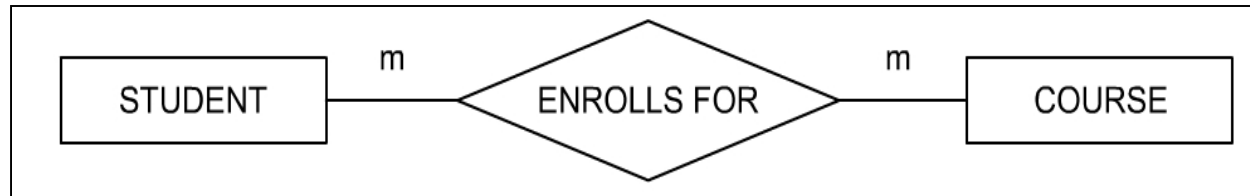
2. One-to-many

3. Many-to-many





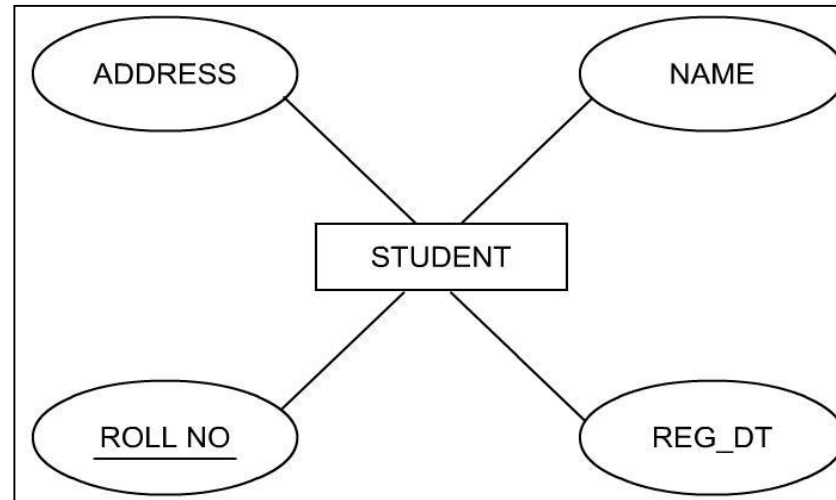
What is Data Model





What is Data Model

- An attribute: Is a property of a given entity.
- Is depicted as ellipses, labeled with the name of the property.
- The following diagram shows the various attributes of the entity, STUDENT.





Record-based logical model

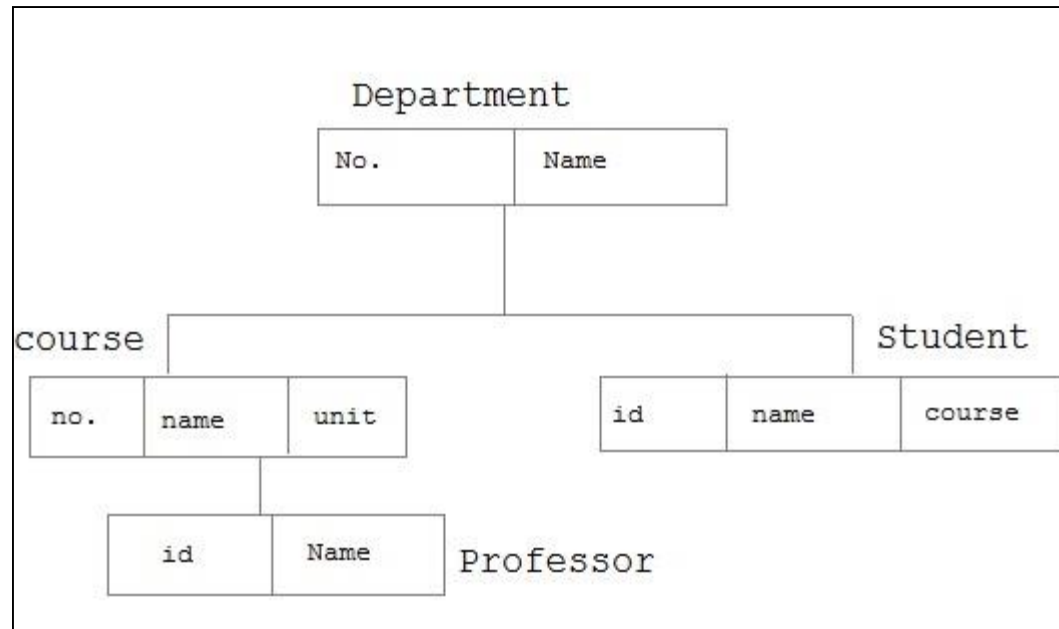
- A Database model defines the logical design of data. The model describes the relationships between different parts of the data. Historically, in database design, three models are commonly used. They are,
 1. Hierarchical Model
 2. Network Model
 3. Relational Model





Hierarchical Model

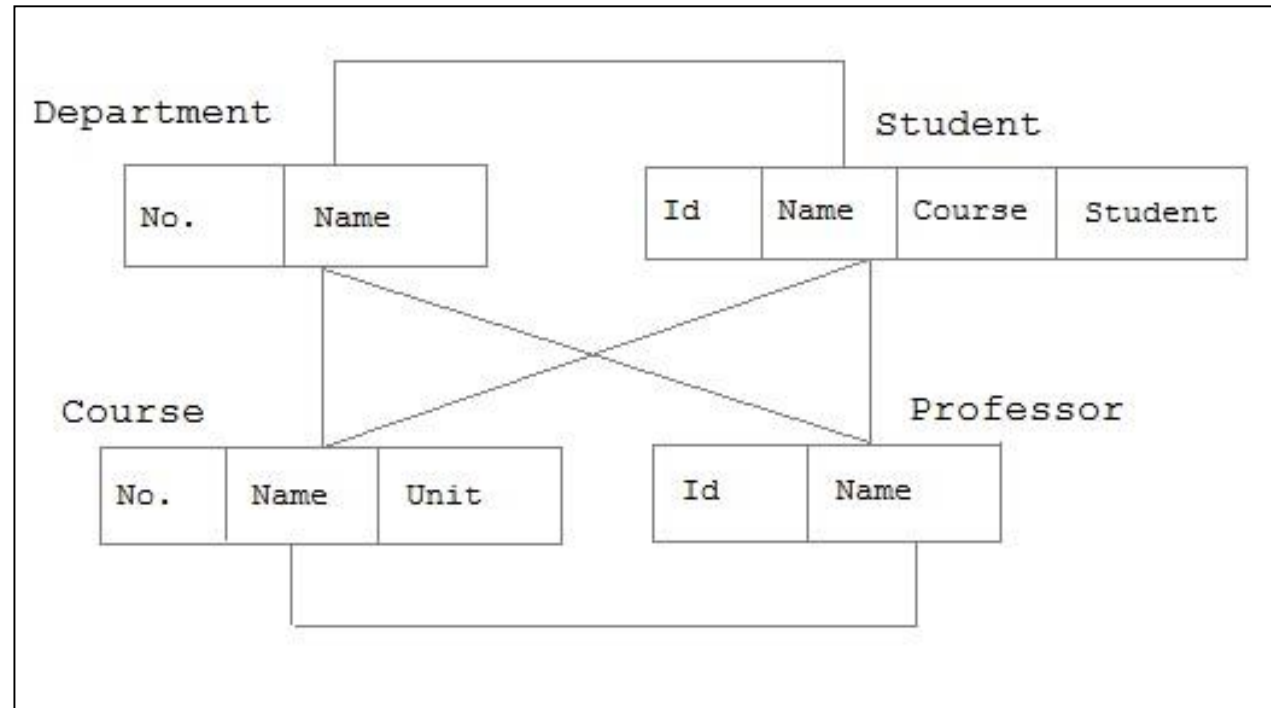
- In this model each entity has only one parent but can have several children .
At the top of hierarchy there is only one entity which is called Root.





Network Model

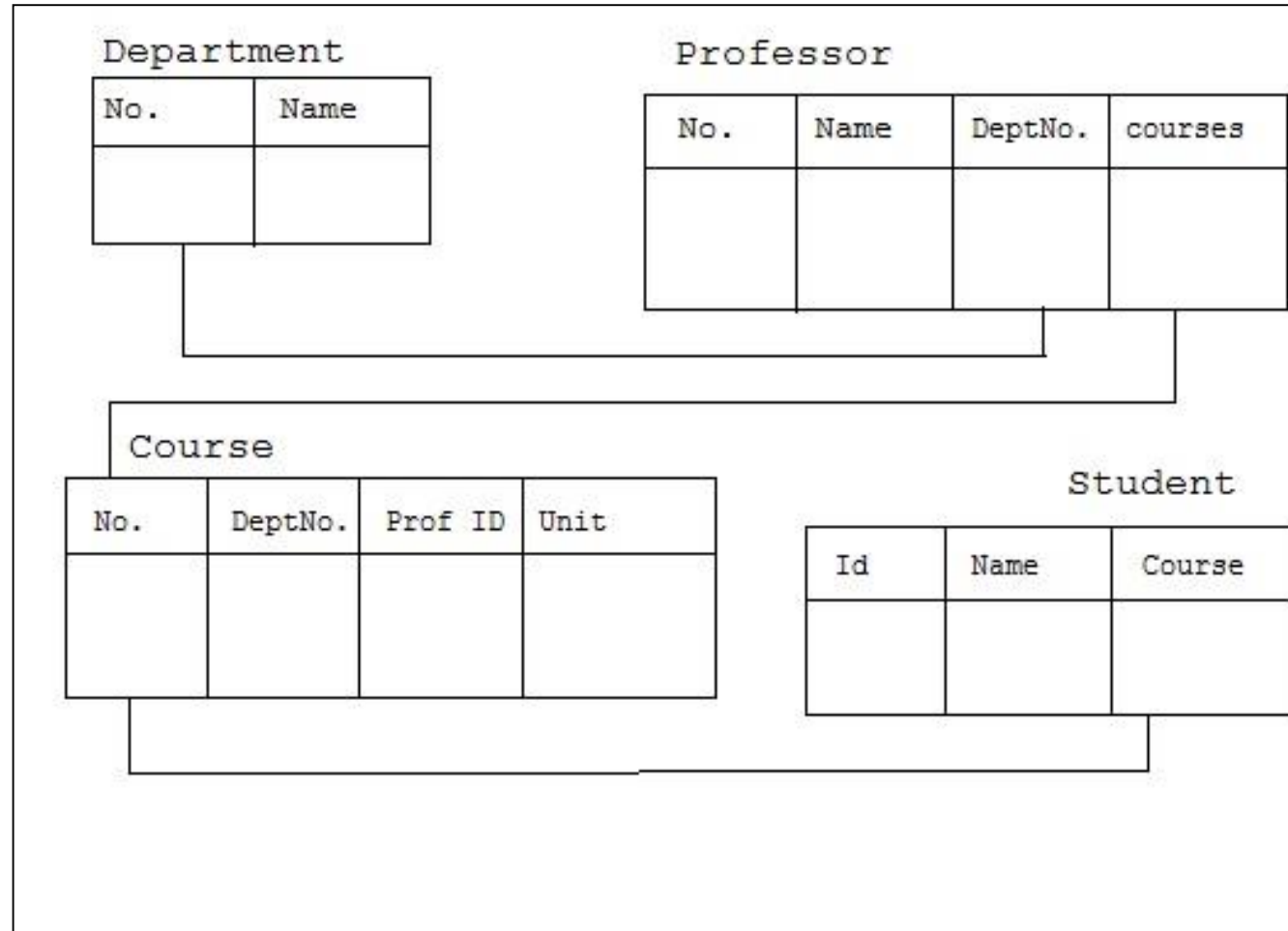
- In the network model, entities are organized in a graph, in which some entities can be accessed through several path





Relational Model

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





Keys

- An RDBMS identifies and locates rows by value.
- The physical address is transparent to the user.
- Relational systems require keys that can uniquely identify the rows of a table.
- The various types of keys used in an RDBMS are:
 1. Primary
 2. Foreign
 3. Candidate
 4. Alternate
 5. Composite





Keys

- Any attribute (or set of attributes) that uniquely identifies a row in a table is a candidate for the primary key. Such an attribute is called a candidate key.
- Any attribute that is a candidate for the primary key but is not the primary key is called the alternate key.
- When the key that uniquely identifies the rows of the table is made up of more than one attribute, it is called a composite key.





Definition of Normalization

- Normalization is a method of breaking down complex table structures into simple table structures by using certain rules. Using this method, you can reduce redundancy in a table, and eliminate the problems of inconsistency and disk space usage. You can also ensure that there is no loss of information.
- Normalization has the following benefits:
 1. It helps in maintaining data integrity.
 2. It helps in simplifying the structure of tables, therefore, making a database more compact.
 3. It helps in reducing the null values, which reduces the complexity of data operations.





Definition of Normalization

- Some rules that should be followed to achieve a good database design are:
 1. Each table should have a unique identifier.
 2. Each table should store data for a single type of entity.
 3. Columns that accept NULL values should be avoided.
 4. The repetition of values or columns should be avoided.





Types of Normalization

- The most important and widely used normal forms are:
 1. First Normal Form (1NF)
 2. Second Normal Form (2NF)
 3. Third Normal Form (3NF)
 4. Boyce-Codd Normal Form (BCNF)





First Normal Form

- A table is said to be in 1NF when each cell of the table contains precisely one value.
- Each table should be organized into rows, and each row should have a primary key that distinguishes it as unique.
- The Primary key is usually a single column, but sometimes more than one column can be combined to create a single primary key. For example consider a table which is not in First normal form

Student Table :

Student	Age	Subject
Adam	15	Biology, Maths
Alex	14	Maths
Stuart	17	Maths





First Normal Form

- In First Normal Form, any row must not have a column in which more than one value is saved, like separated with commas. Rather than that, we must separate such data into multiple rows.
- Student Table following 1NF will be :

Student	Age	Subject
Adam	15	Biology
Adam	15	Maths
Alex	14	Maths
Stuart	17	Maths

- Using the First Normal Form, data redundancy increases, as there will be many columns with same data in multiple rows but each row as a whole will be unique.





Functional Dependency

- The normalization theory is based on the fundamental notion of functional dependency. Given a relation (you may recall that a table is also called a relation) R , attribute A is functionally dependent on attribute B if each value of A in R is associated with precisely one value of B .
- In other words, attribute A is functionally dependent on B if and only if, for each value of B , there is exactly one value of A .
- Attribute B is called the determinant.





Functional Dependency

- In this example, if we know the value of Employee number, we can obtain Employee Name, city, salary, etc. By this, we can say that the city, Employee Name, and salary are functionally depended on Employee number.

EMPLOYEE NUMBER	EMPLOYEE NAME	SALARY	CITY
1	Dana	50000	San Francisco
2	Francis	38000	London
3	Andrew	25000	Tokyo





Functional Dependency

- Consider the REPORT table, as shown in the following diagram.

Primary key

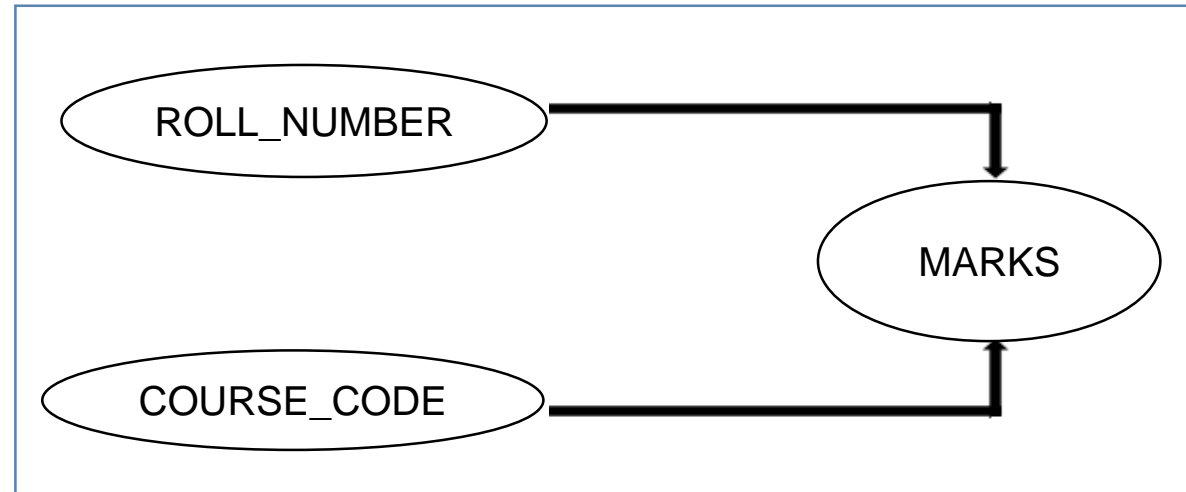
ROLL NUMBER	COURSE _CODE	COURSE NAME	T_NAME	ROOM_NUMBER	MARKS	GRADE
R001	C100	Java	James	301	88	A
R002	C101	C#.NET	Peter	302	75	B
R003	C101	C#.NET	Peter	302	60	C
R004	C100	Java	James	301	72	B





Functional Dependency

- In the REPORT table:
 - For a particular value of ROLL_NUMBER+COURSE_CODE, there is precisely one corresponding value for MARKS.
 - Hence, MARKS is functionally dependent on ROLL_NUMBER+COURSE_CODE.
 - This can be symbolically represented as: $(\text{ROLL_NUMBER}, \text{COURSE_CODE}) \rightarrow \text{MARKS}$.
 - The following diagram shows the functional dependency between MARKS and ROLL_NUMBER+COURSE_CODE.





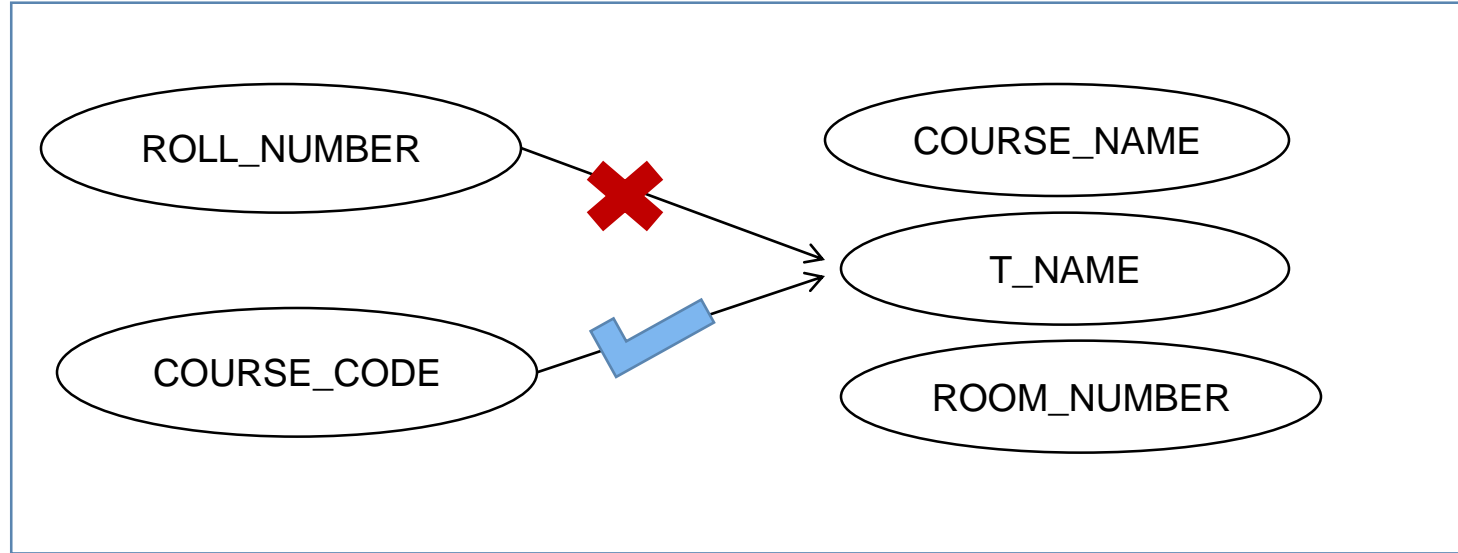
Functional Dependency

- The other functional dependencies in the REPORT table are:
 - COURSE_CODE->COURSE_NAME
 - COURSE_CODE->T_NAME (Assuming one course is taught by only one teacher.)
 - T_NAME->ROOM_NUMBER (Assuming each teacher has his/her own, unshared room.)
 - MARKS->GRADE
 - COURSE_NAME, T_NAME, and ROOM_NUMBER attributes are partially dependent on the whole key.
 - This dependency is called partial dependency, as shown in the following diagram.





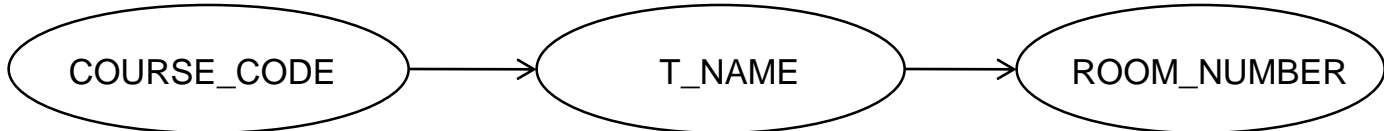
Functional Dependency





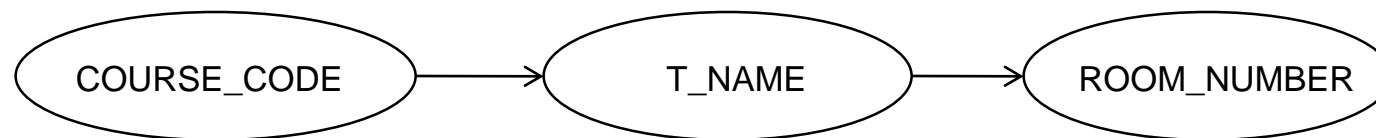
Functional Dependency

- ROOM_NUMBER is dependent on T_NAME, and T_NAME is dependent on COURSE_CODE, as shown in the following diagram.



ROLL NUMBER	COURSE _CODE	COURSE NAME	T_NAME	ROOM_NUMBER	MARKS	GRADE
R001	C100	Java	James	301	88	A
R002	C101	C#.NET	Peter	302	75	B
R003	C101	C#.NET	Peter	302	60	C
R004	C100	Java	James	301	72	B

This type of dependency is called as transitive dependency.





Second Normal Form

- A table is said to be in 2NF when:
 - It is in the 1NF, and
 - No partial dependency exists between non-key attributes and key attributes.
 - The guidelines for converting a table into 2NF are:
 - Find and remove attributes that are functionally dependent on only a part of the key and not on the whole key. Place them in a different table.
 - Group the remaining attributes.





Second Normal Form

- Consider the PROJECT table, as shown in the following diagram.

<i>ECODE</i>	<i>PROJCODE</i>	<i>DEPT</i>	<i>DEPTHEAD</i>	<i>HOURS</i>
<i>E101</i>	<i>P27</i>	<i>Systems</i>	<i>E901</i>	<i>90</i>
<i>E305</i>	<i>P27</i>	<i>Finance</i>	<i>E909</i>	<i>10</i>
<i>E508</i>	<i>P51</i>	<i>Admin</i>	<i>E908</i>	<i>NULL</i>
<i>E101</i>	<i>P51</i>	<i>Systems</i>	<i>E901</i>	<i>101</i>
<i>E101</i>	<i>P20</i>	<i>Systems</i>	<i>E901</i>	<i>60</i>
<i>E508</i>	<i>P27</i>	<i>Admin</i>	<i>E908</i>	<i>72</i>





Second Normal Form

DEPTHEAD is functionally dependent on ECODE; however, it is not dependent on the attribute, PROJCODE.

Dept is functionally dependent on part of the key, which is ECODE.

Hours is functionally dependent on the whole key, ECODE+PROJCODE.

Composite key

ECODE	PROJCODE	DEPT	DEPTHEAD	HOURS
E101	P27	Systems	E901	90
E305	P27	Finance	E909	10
E508	P51	Admin	E908	NULL
E101	P51	Systems	E901	101
E101	P20	Systems	E901	60
E508	P27	Admin	E908	72





Second Normal Form

- The EMPLOYEEDEPT and PROJECT tables are in 2NF, as shown in the following diagram.

EMPLOYEEDEPT

<i>ECODE</i>	<i>DEPT</i>	<i>DEPTHEAD</i>
<i>E101</i>	<i>Systems</i>	<i>E901</i>
<i>E305</i>	<i>Finance</i>	<i>E909</i>
<i>E508</i>	<i>Admin</i>	<i>E908</i>

PROJECT

<i>ECODE</i>	<i>PROJCODE</i>	<i>HOURS</i>
<i>E101</i>	<i>P27</i>	<i>90</i>
<i>E101</i>	<i>P51</i>	<i>101</i>
<i>E101</i>	<i>P20</i>	<i>60</i>
<i>E305</i>	<i>P27</i>	<i>10</i>
<i>E508</i>	<i>P51</i>	<i>NULL</i>
<i>E508</i>	<i>P27</i>	<i>72</i>





Third Normal Form

- A relation is said to be in the 3NF if and only if:
 - It is in 2NF, and
 - No transitive (indirect) dependency exists between non-key attributes and key attributes.
 - The guidelines for converting a table into 3NF are:
 - Find and remove non-key attributes that are functionally dependent on attributes that are not the primary key. Place them in a different table.
 - Group the remaining attributes.





Third Normal Form

- Consider the EMPLOYEE table, as shown in the following diagram.

Primary key

DEPTHEAD is functionally dependent on DEPT, which is not a primary key.

<i>ECODE</i>	<i>DEPT</i>	<i>DEPTHEAD</i>
<i>E101</i>	<i>Systems</i>	<i>E901</i>
<i>E305</i>	<i>Finance</i>	<i>E909</i>
<i>E402</i>	<i>Sales</i>	<i>E906</i>
<i>E508</i>	<i>Admin</i>	<i>E908</i>
<i>E607</i>	<i>Finance</i>	<i>E909</i>
<i>E608</i>	<i>Finance</i>	<i>E909</i>





Third Normal Form

- To convert the EMPLOYEE table into 3NF, you must remove the DEPTHEAD column and place it in another table, as shown in the following diagram.

EMPLOYEE

<i>ECODE</i>	<i>DEPT</i>
<i>E101</i>	<i>Systems</i>
<i>E305</i>	<i>Finance</i>
<i>E402</i>	<i>Sales</i>
<i>E508</i>	<i>Admin</i>
<i>E607</i>	<i>Finance</i>
<i>E608</i>	<i>Finance</i>

DEPARTMENT

<i>DEPT</i>	<i>DEPTHEAD</i>
<i>Systems</i>	<i>E901</i>
<i>Sales</i>	<i>E906</i>
<i>Admin</i>	<i>E908</i>
<i>Finance</i>	<i>E909</i>





Boyce-Codd Normal Form (BCNF)

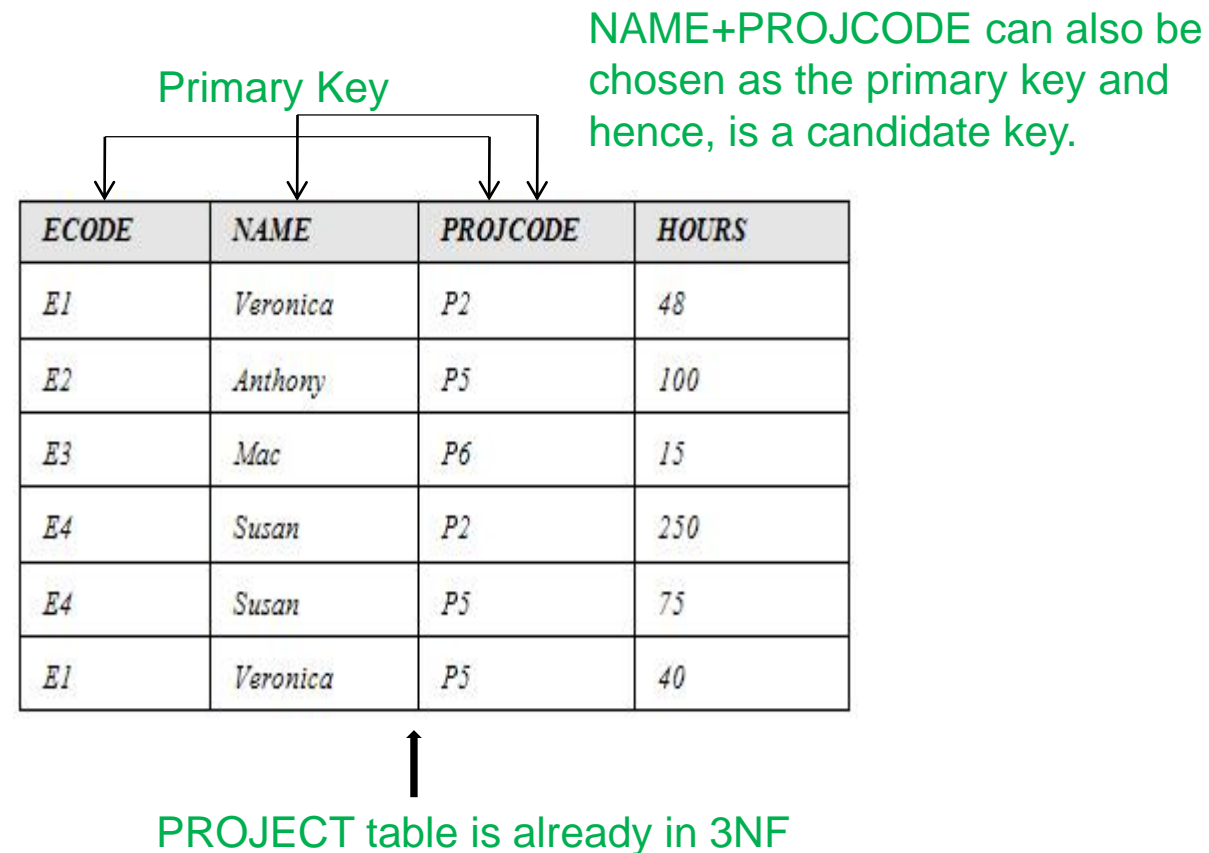
- The original definition of 3NF was not sufficient in some situations. It was not satisfactory for the tables:
 - That had multiple candidate keys.
 - Where the multiple candidate keys were composite.
 - Where the multiple candidate keys overlapped (had at least one attribute in common).
- The guidelines for converting a table into BCNF are:
 - Find and remove the overlapping candidate keys. Place the part of the candidate key and the attribute it is functionally dependent on, in a different table.
 - Group the remaining items into a table.





Boyce-Codd Normal Form (BCNF)

- Consider the PROJECT table, as shown in the following diagram.





Boyce-Codd Normal Form (BCNF)

- The following points describe the functional dependencies in the PROJECT table:
 - HOURS is functionally dependent on ECODE+PROJCODE.
 - HOURS is also functionally dependent on NAME+PROJCODE.
 - NAME is functionally dependent on ECODE.
 - ECODE is functionally dependent on NAME.





Boyce-Codd Normal Form (BCNF)

- You will notice that the PROJECT table has:
 - Multiple candidate keys that are ECODE+PROJCODE and NAME+PROJCODE.
 - Composite candidate keys.
 - Candidate keys that overlap since the PROJCODE attribute is common between the two candidate keys.
 - The only non-key item is HOURS, which is dependent on the whole key, ECODE+PROJCODE or NAME+PROJCODE.
 - ECODE and NAME are determinants since they are functionally dependent on each other.
 - As per BCNF, the determinants have to be candidate keys.
 - You can remove NAME and ECODE and place them in a different table.





Boyce-Codd Normal Form (BCNF)

- You can remove NAME and ECODE and place them in a different table, as shown in the following diagram

<i>ECODE</i>	<i>NAME</i>
<i>E1</i>	<i>Veronica</i>
<i>E2</i>	<i>Anthony</i>
<i>E3</i>	<i>Mac</i>
<i>E4</i>	<i>Susan</i>

<i>ECODE</i>	<i>PROJCODE</i>	<i>HOURS</i>
<i>E1</i>	<i>P2</i>	<i>48</i>
<i>E2</i>	<i>P5</i>	<i>100</i>
<i>E3</i>	<i>P6</i>	<i>15</i>
<i>E4</i>	<i>P2</i>	<i>250</i>
<i>E4</i>	<i>P5</i>	<i>75</i>
<i>E1</i>	<i>P5</i>	<i>40</i>





Summary

- Database is a collection of related data organized in order to provide ease in data access and management.
- DBMS allows overall management of database possible.
- RDBMS is used to manage relational database.
- Data model provides description of data.
- Primary key is used to uniquely identify records.
- Normalization is a method of breaking down complex tables into smaller ones by following certain rules. It also avoids data redundancy and provides consistency.



