Module III

Relational Model and relational Algebra

Content

- Introduction to the Relational Model
  Relational Model Constraints & Relational Database schema
  Concept of keys, Primary Key, Secondary key, Foreign Key
  Mapping the ER and EER Model to the Relational Model
  Set Theory operations
  Binary Relational operation
  Relational Algebra operators
  Relational Algebra Queries

Introduction to the Relational Model

- **Relational data model** stores data in the **form of tables**. Relational data model is the primary data model, which is used widely around the world for data storage and processing.
- This model is simple and it has all the properties and capabilities required to process data with storage efficiency.
- Relational Model (RM) represents the database as a collection of relations.
- A relation is nothing but a table of values.
- Every row in the table represents a collection of **related data values**.

INFORMAL DEFINITIONS

- **RELATION: A table of values**
  - A relation may be thought of as a **set of rows**.
  - A relation may alternately be though of as a **set of columns**.
  - Each row represents a fact that corresponds to a real-world **entity** or **relationship**.
  - Each row has a value of an item or set of items that uniquely identifies that row in the table.
  - Each column typically is called by its column name or column header or attribute name.

- A **Relation** may be defined in multiple ways.
- The **Schema** of a Relation: *R* (A1, A2, ...An)
  Relation schema *R* is defined over **attributes** A1, A2, ...An
  For Example -

  CUSTOMER (Cust-id, Cust-name, Address, Phone#)

  Here, CUSTOMER is a relation defined over the four attributes Cust-id, Cust-name, Address, Phone#, each of which has a **domain** or a set of valid values. For example, the domain of Cust-id is 6 digit numbers.

- A **tuple** is an ordered set of values
- Each value is derived from an appropriate domain.
- Each row in the CUSTOMER table may be referred to as a tuple in the table and would consist of four values.

<101, “Anil", “Mumbai", “9585485858">

is a tuple belonging to the CUSTOMER relation.

- A relation may be regarded as a ***set of tuples*** (rows).
- Columns in a table are also called attributes of the relation.

- A **domain** has a logical definition: e.g., “mobile_numbers” are the set of 10 digit mobile numbers.

- A domain may have a data-type or a format defined for it. E.g., Dates have various formats such as monthname, date, year or yyyy-mm-dd, or dd mm,yyyy etc.

DEFINITION SUMMARY

Relational Model is made up of tables

- A row of table = a relational instance/tuple
- A column of table = an attribute
- A table = a schema/relation
- Cardinality = number of rows
- Degree = number of columns
- Domain = Values in a column
- Table Definition =Schema of a Relation

Concepts

Tables − In relational data model, **relations** are saved in the format of **Tables.**

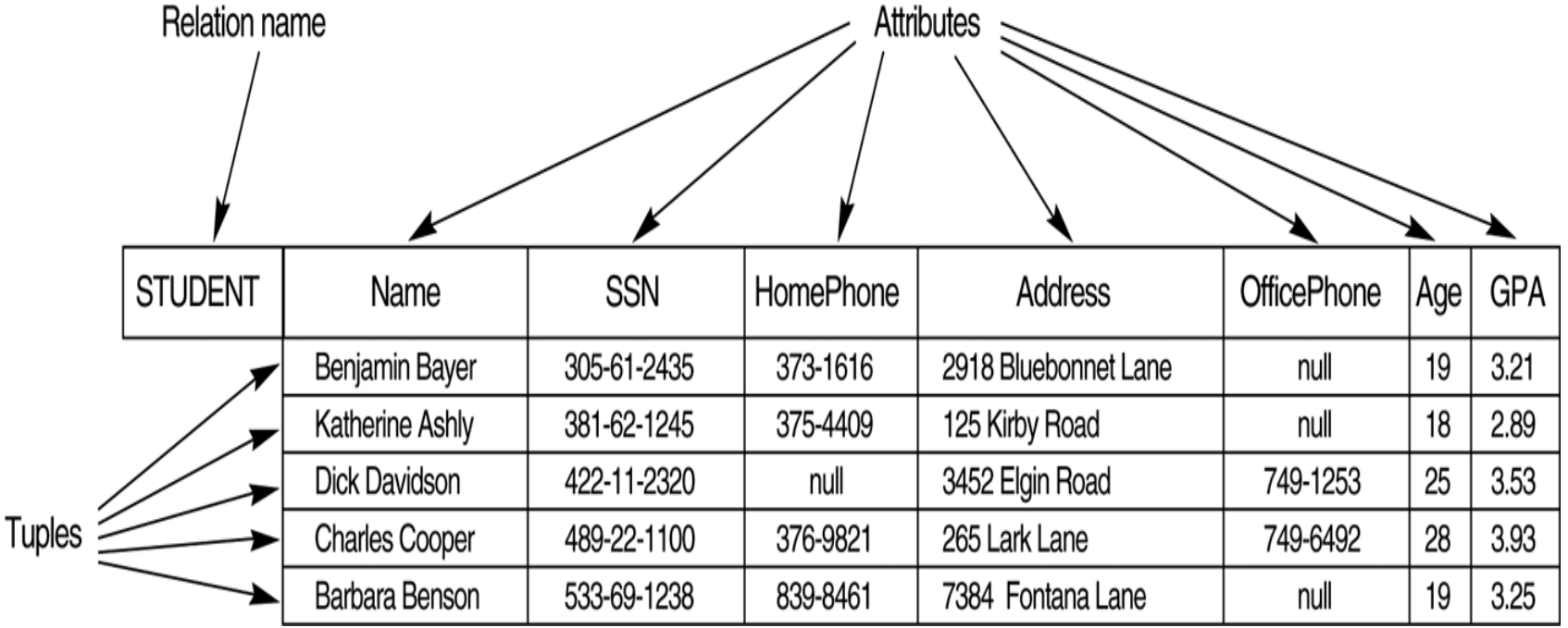
A table has **rows and columns**, where rows represents records and columns represent the **attributes.**

Tuple − A single **row of a table**, which contains a single record for that relation is called a tuple.

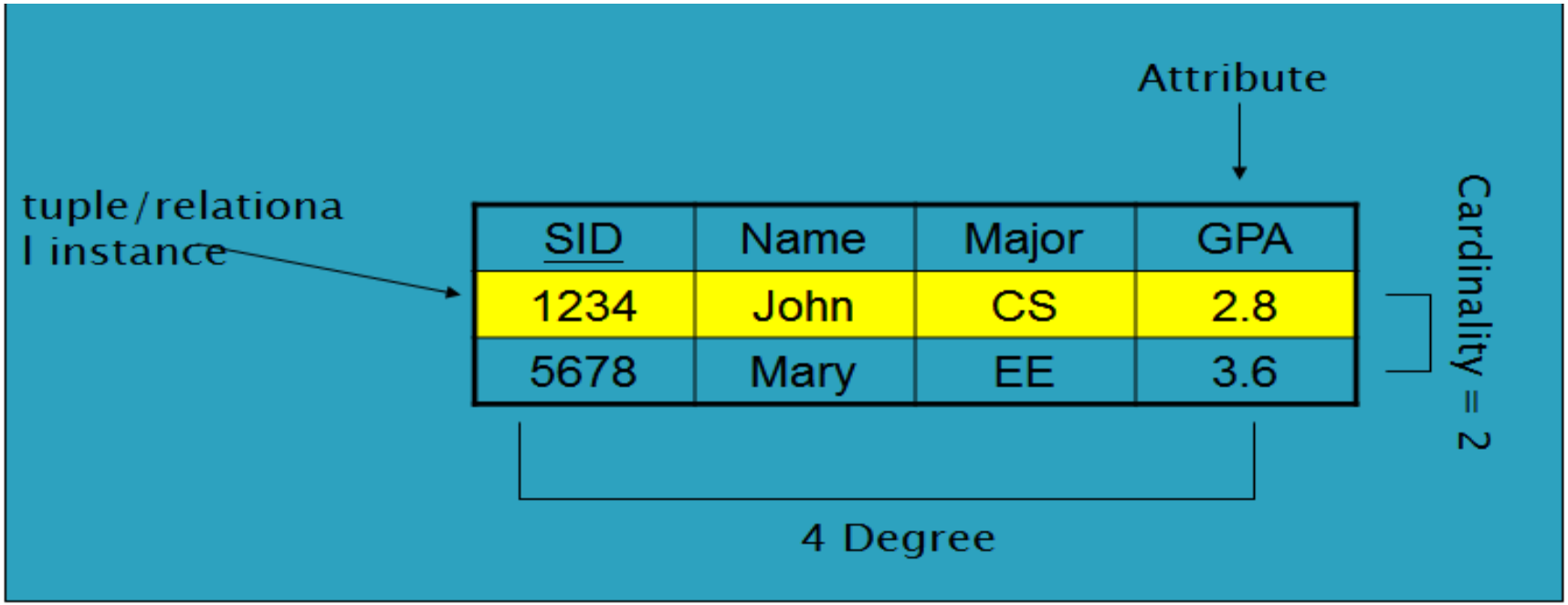
Relation instance − **A finite set of tuples** in the relational database system represents relation instance. Relation instances do not have duplicate tuples.

Relation schema − A relation schema describes the relation name (table name), attributes, and their names.

Relation key − Each row has one or more attributes, known as **relation key,** which can identify the row in the relation (table) uniquely.



STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25



A Schema / Relation

- **Example: STUDENT Relation**

NAME	ROLL_NO	PHONE_NO	ADDRESS	AGE
Ram	14795	7305758992	Noida	24
Shyam	12839	9026288936	Delhi	35
Laxman	33289	8583287182	Gurugram	20
Mahesh	27857	7086819134	Ghaziabad	27
Ganesh	17282	9028 9i3988	Delhi	40

In the given table, NAME, ROLL_NO, PHONE_NO, ADDRESS, and AGE are the attributes.
Relational instances: The instance of schema STUDENT has 5 tuples.
Tuple: t3=<Laxman, 33289, 8583287182, Gurugram, 20>
Relation schema: Student (Name,roll_no,phone_no,address,age)

Properties of the relational database model

- Data is presented as a **collection of relations**.
- Columns are attributes that belong to the entity modeled by the table

(ex. In a student table, you could have name, address, student ID, major, etc.).

- Each row ("tuple") represents a single entity or an instance of that particular entity

(ex. In a student table, John Smith, 14 Oak St, 9002342, Accounting, would represent one student entity).

- Every table has a set of attributes that taken together as a "key" (technically, a "superkey") uniquely identifies each entity

(Ex. In the student table, “student ID” would uniquely identify each student – no two students would have the same student ID).

- **Rules**

1. The **order** of tuples and attributes is not important. (Ex. Attribute order not important…if you have name before address, is the same as address before name).
2. Every tuple is **unique**. This means that for every record in a table there is something that uniquely identifies it from any other tuple.
3. Cells contain **single values**. This means that each cell in a table can contain only one value.
4. All values within an attribute are from the **same domain**. This means that however the attribute is defined, the values for each tuple fall into that definition. For example, if the attribute is labeled as Date, you would not enter a dollar amount, shirt size, or model number in that column, only dates.
5. **Table names in the database must be unique and attribute names in tables must be unique**. No two tables can have the same name in a database. Attributes (columns) cannot have the same name in a table. You can have two different tables that have similar attribute names.

Operations in Relational Model

Four basic update operations performed on relational database model are Insert, update, delete and select.

- Insert is used to insert data into the relation
- Delete is used to delete tuples from the table.
- Modify allows you to change the values of some attributes in existing tuples.
- Select allows you to choose a specific range of data.

Update Operation

You can see that in the below-given relation table CustomerName= ‘Apple’ is updated from Inactive to Active.

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Active
4	Alibaba	Active

Delete Operation

To specify deletion, a condition on the attributes of the relation selects the tuple to be deleted.

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Active
4	Alibaba	Active

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
4	Alibaba	Active

Select Operation

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
4	Alibaba	Active

CustomerID	CustomerName	Status
2	Amazon	Active

Relational Model Constraints

Relational constraints are the **restriction**s imposed on the **database contents and operations**. They ensure the **correctness** of data in the database.
The purpose of constraints is **to maintain the data integrity** during changes like **update/delete/insert**

Mainly Constraints on the relational database are of 4 types:

1. Domain constraints
2. Key Constraints
3. Entity Integrity constraints
4. Referential integrity constraints

*Enterprise Constraints

1. Domain constraints :

1.Every domain must contain **atomic values**(smallest indivisible units) it means **composite** and **multi-valued attributes** are not allowed.
2.We perform **datatype check** here, which means when we assign a data type to a column we limit the values that it can contain. Eg. If we assign the datatype of attribute age as int, we cant give it values other then int datatype.

Example:

EID	Name	Phone
01	Bikash Dutta	123456789 234456678

Explanation:
In the above relation, Name is a composite attribute and Phone is a multi-values attribute, so it is violating domain constraint.

2. Key Constraints or Uniqueness Constraints

1.These are called **uniqueness constraints** since it ensures that every tuple in the relation should be unique.
2.A relation can have multiple keys or candidate keys(minimal superkey), out of which we choose one of the keys as primary key, we don't have any restriction on choosing the primary key out of candidate keys, but it is suggested to go with the candidate key with less number of attributes.
3.Null values are not allowed in the primary key, **hence Not Null constraint** is also a part of key constraint.

Example:

EID	Name	Phone
01	Bikash	6000000009
02	Paul	9000090009
01	Tuhin	9234567892

Explanation:
In the above table, EID is the **primary key**, and first and the last tuple has the same value in EID ie 01, so it is violating the key constraint.

3. Entity Integrity Constraints :

1.Entity Integrity constraints says that **no primary key can take NULL value**, since using primary key we identify each tuple uniquely in a relation.

EID	Name	Phone
01	Bikash	9000900099
02	Paul	600000009
NULL	Sony	9234567892

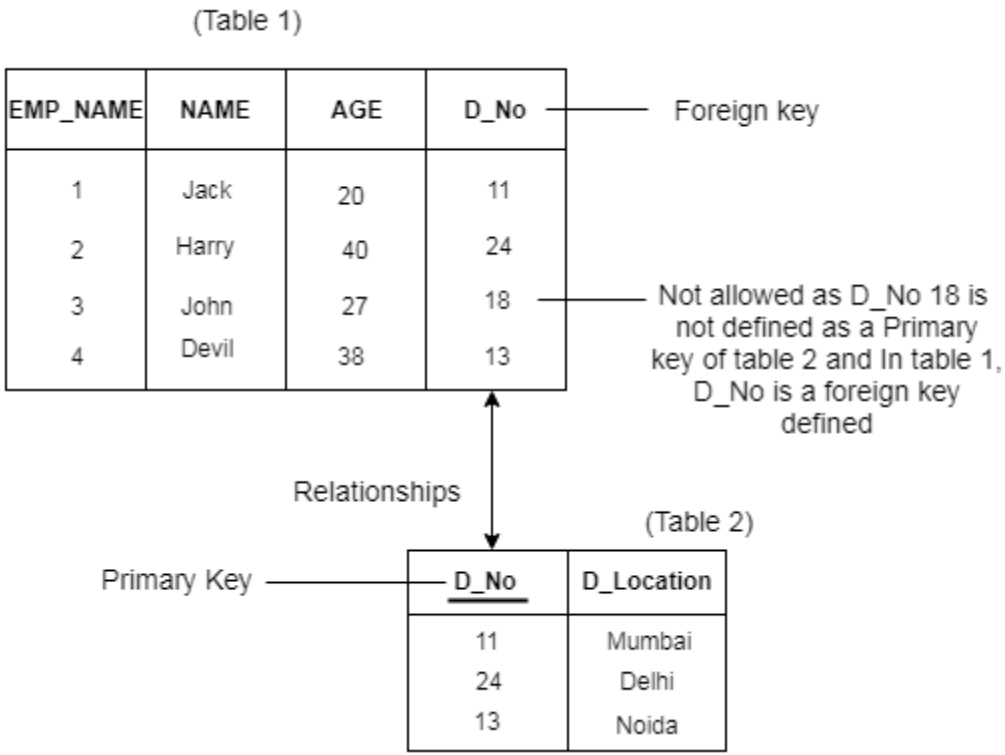
Explanation:

In the above relation, EID is made primary key, and the primary key cant take NULL values but in the third tuple, the primary key is null, so it is a violating Entity Integrity constraints.

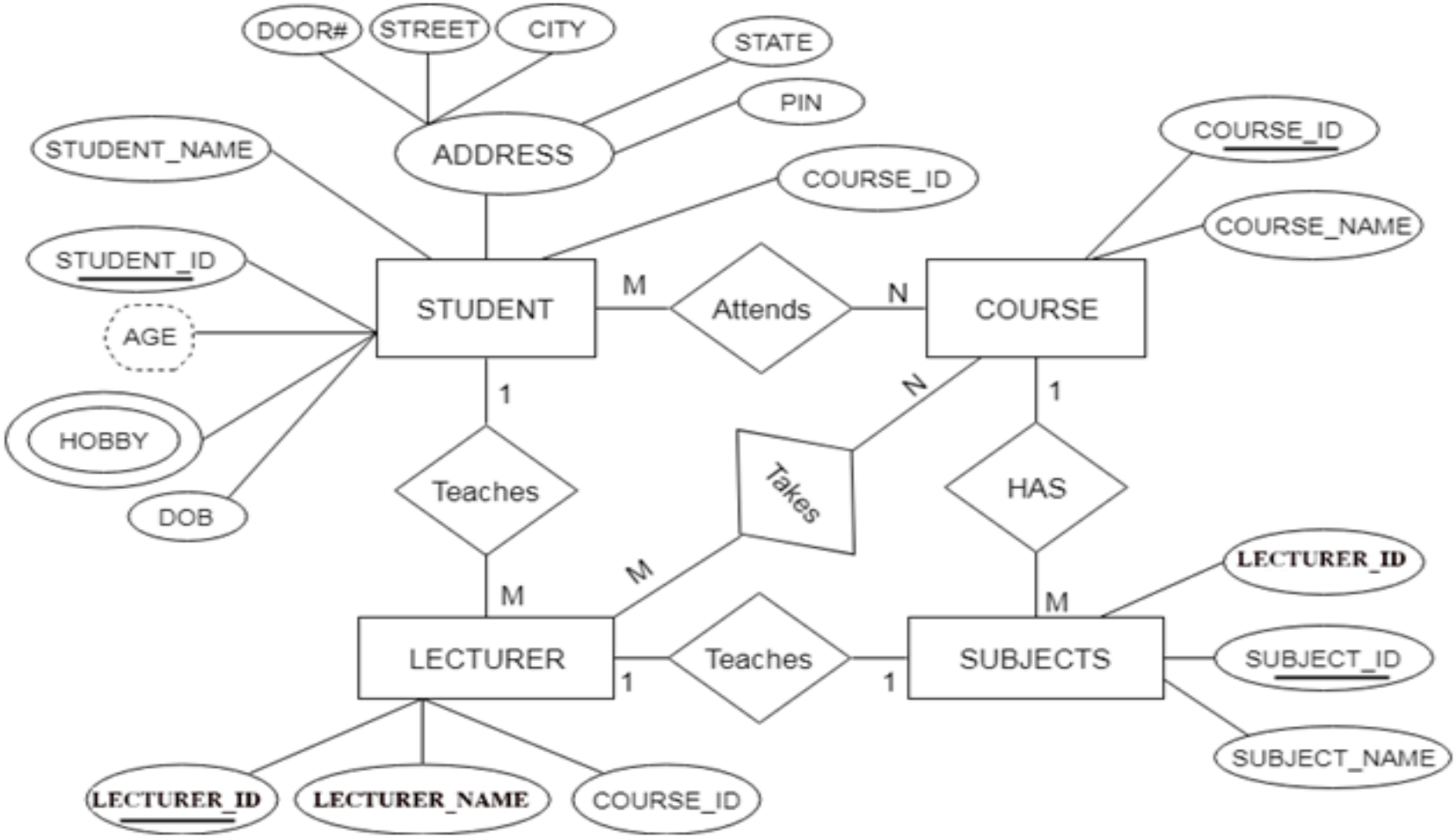
4. Referential Integrity Constraints

1.The Referential integrity constraints is specified between two relations or tables and used to maintain the consistency among the tuples in two relations.
2.This constraint is enforced through foreign key, when an attribute in the foreign key of relation R1 have the same domain(s) as the primary key of relation R2, then the foreign key of R1 is said to reference or refer to the primary key of relation R2.
3.The values of the foreign key in a tuple of relation R1 can either take the values of the primary key for some tuple in relation R2, or can take NULL values, but can’t be empty.

Example


(Table 1)
EMP_NAME
NAME
AGE
D_No
Foreign key
1
Jack
20
11
2
Harry
40
24
3
John
27
18
4
Devil
38
13
Not allowed as D_No 18 is not defined as a Primary key of table 2 and In table 1, D_No is a foreign key defined
Relationships
(Table 2)
Primary Key
D_No
D_Location
11
Mumbai
24
Delhi
13
Noida

Steps for Mapping ER/ERR with relational model


DOOR#
STREET
CITY
STATE
PIN
ADDRESS
STUDENT_NAME
COURSE_ID
COURSE_ID
COURSE_NAME
STUDENT_ID
AGE
HOBBY
DOB
STUDENT
M
Attends
N
COURSE
1
N
1
Teaches
Takes
HAS
M
M
M
LECTURER
1
Teaches
1
SUBJECTS
LECTURER_ID
SUBJECT_ID
SUBJECT_NAME
LECTURER_ID
LECTURER_NAME
COURSE_ID

- **Entity type becomes a table.**

In the given ER diagram, LECTURE, STUDENT, SUBJECT and COURSE forms individual tables.

- **All single-valued attribute becomes a column for the table.**

In the STUDENT entity, STUDENT_NAME and STUDENT_ID form the column of STUDENT table. Similarly, COURSE_NAME and COURSE_ID form the column of COURSE table and so on.

- **A key attribute of the entity type represented by the primary key.**

In the given ER diagram, COURSE_ID, STUDENT_ID, SUBJECT_ID, and LECTURE_ID are the key attribute of the entity.

- **The multivalued attribute is represented by a separate table.**

In the student table, a hobby is a multivalued attribute. So it is not possible to represent multiple values in a single column of STUDENT table. Hence we create a table STUD_HOBBY with column name STUDENT_ID and HOBBY. Using both the column, we create a composite key.

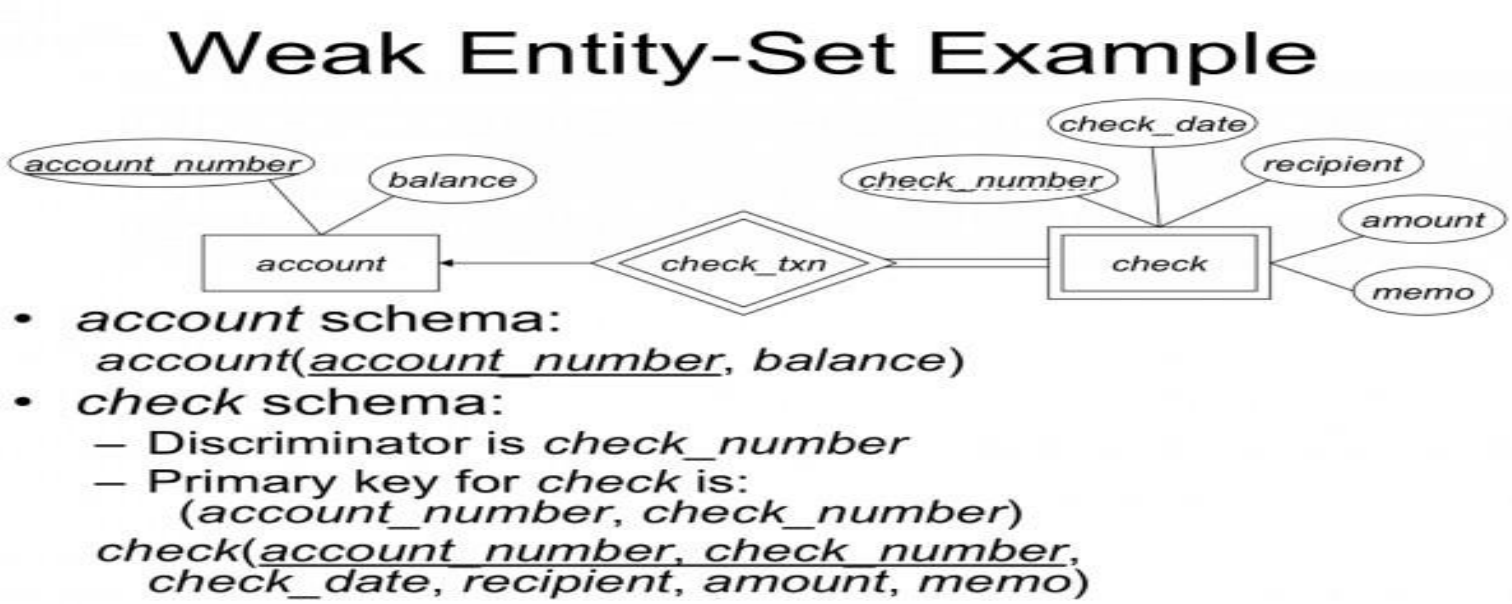
- **Composite attribute represented by components.**

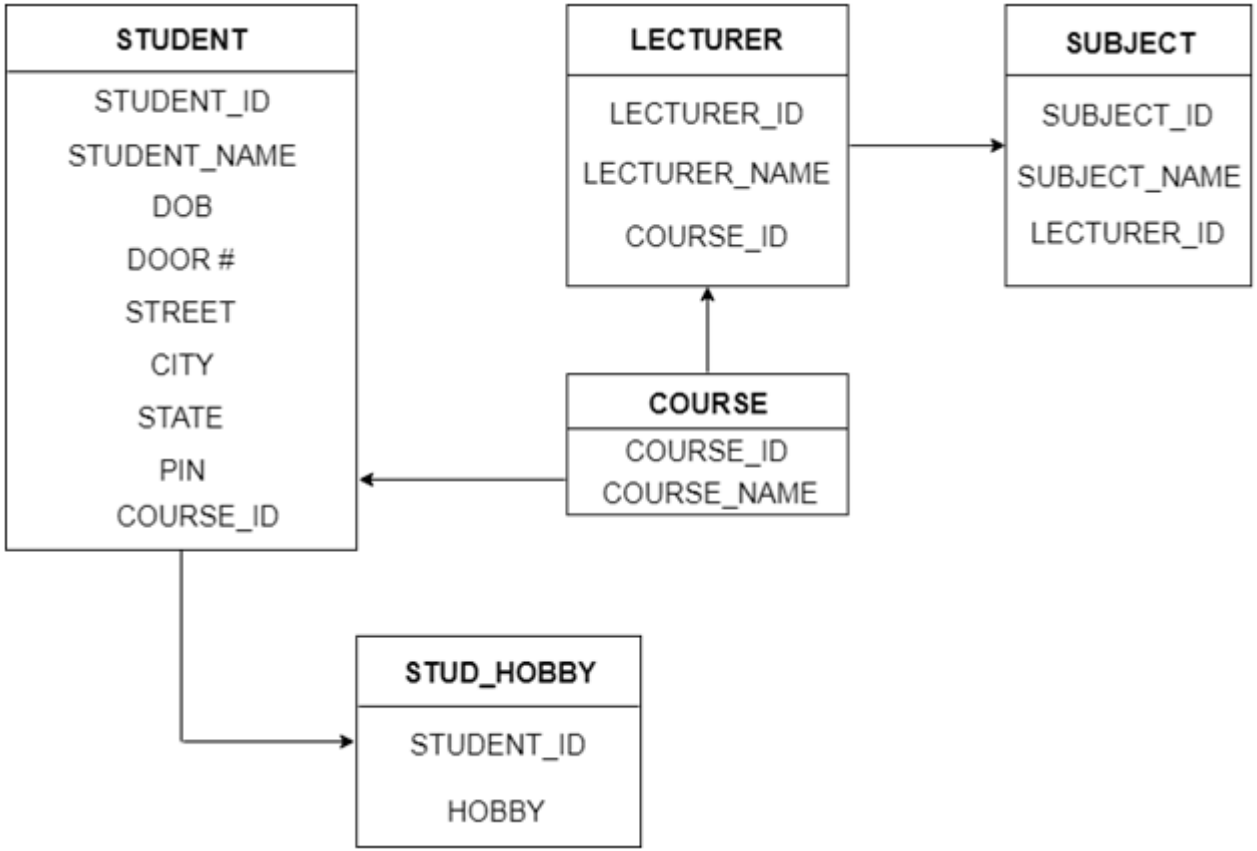
In the given ER diagram, student address is a composite attribute. It contains CITY, PIN, DOOR, STREET, and STATE. In the STUDENT table, these attributes can consider as an individual column.

- **Derived attributes are not considered in the table.**

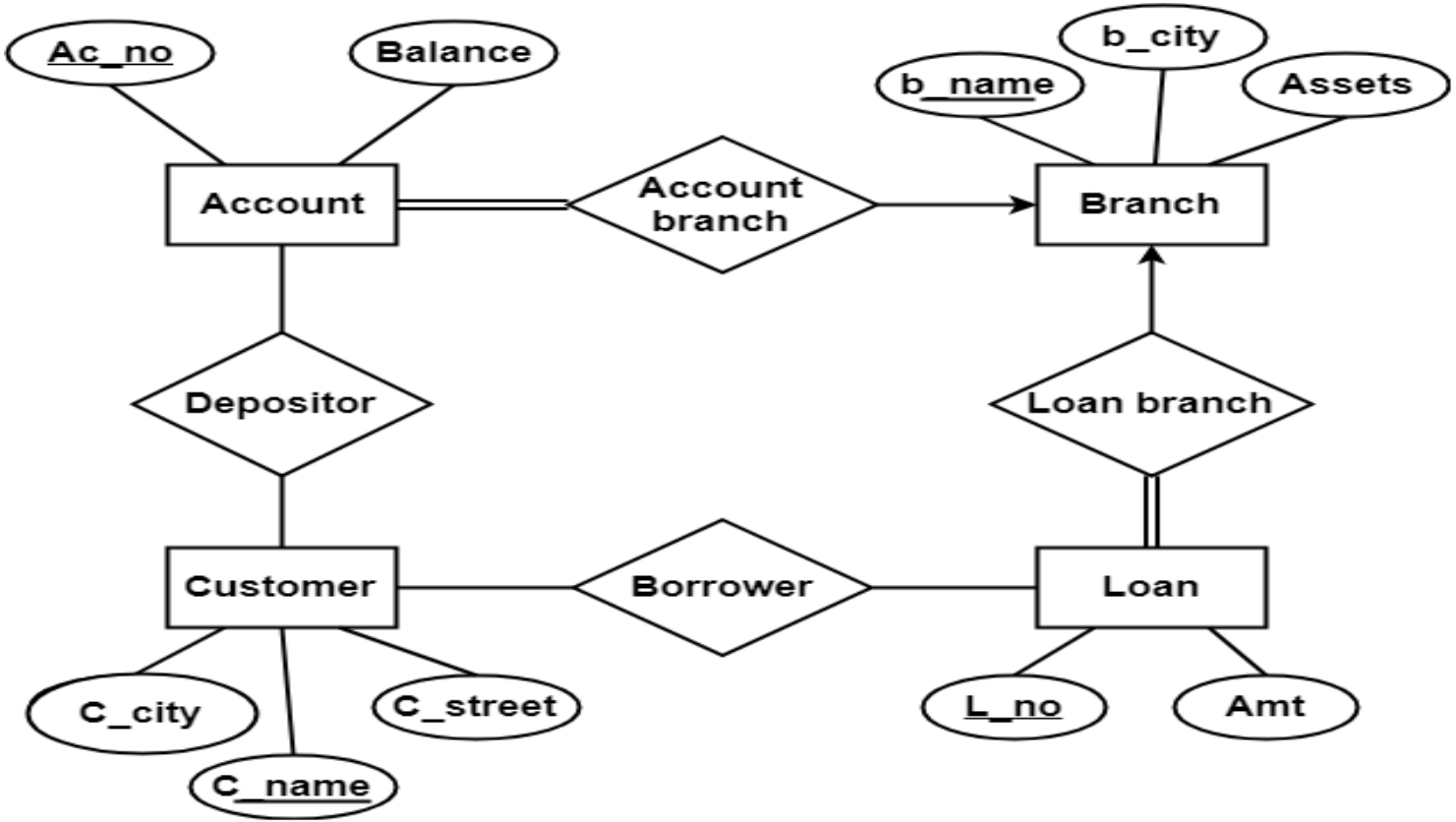
In the STUDENT table, Age is the derived attribute. It can be calculated at any point of time by calculating the difference between current date and Date of Birth.

- **In case of weak entity add all attributes including discriminator(partial key in table) as well primary key of all entities which are related to that entity.**

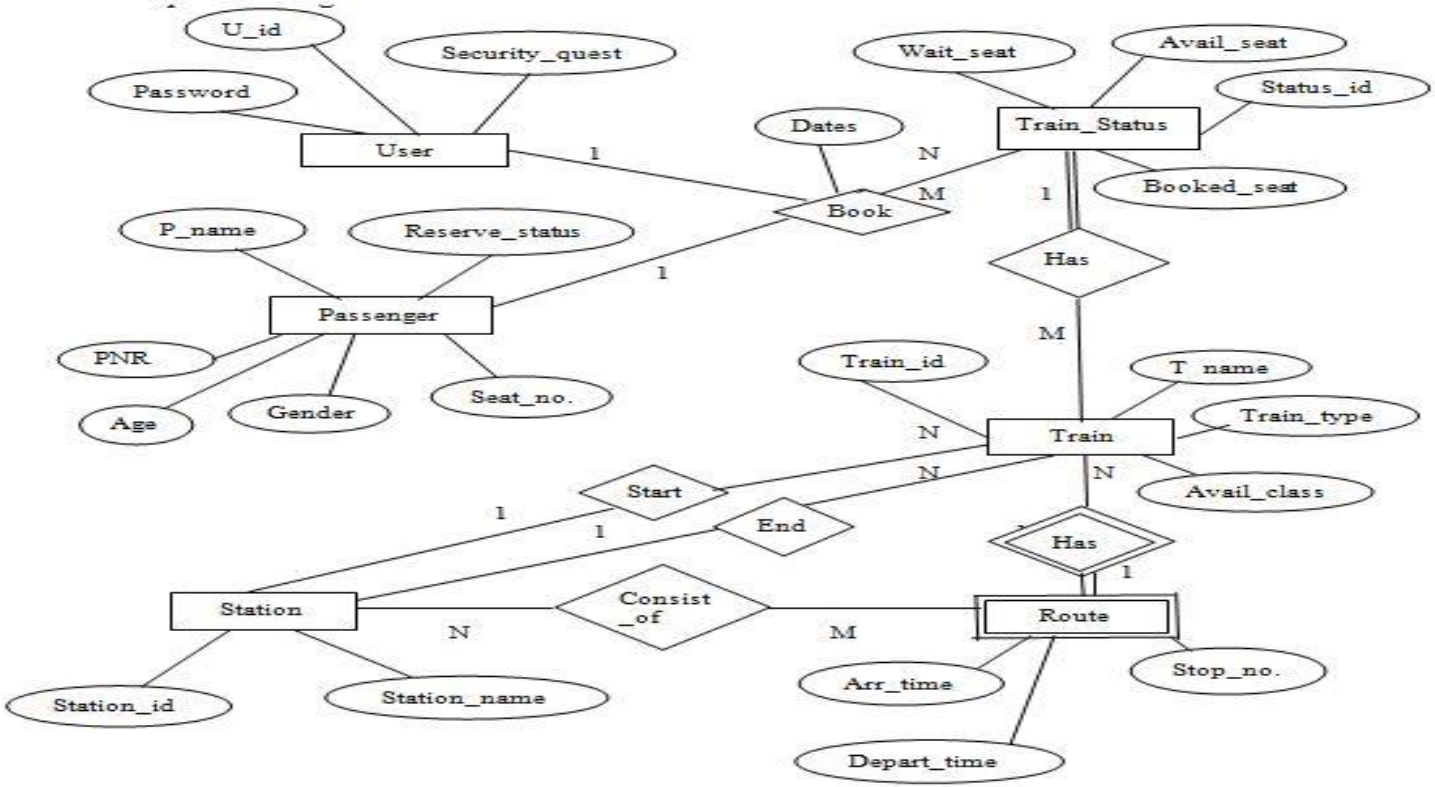



STUDENT
STUDENT_ID
STUDENT_NAME
DOB
DOOR #
STREET
CITY
STATE
PIN
COURSE_ID
LECTURER
LECTURER_ID
LECTURER_NAME
COURSE_ID
SUBJECT
SUBJECT_ID
SUBJECT_NAME
LECTURER_ID
COURSE
COURSE_ID
COURSE_NAME
STUD_HOBBY
STUDENT_ID
HOBBY

ER Diagram for Banking System

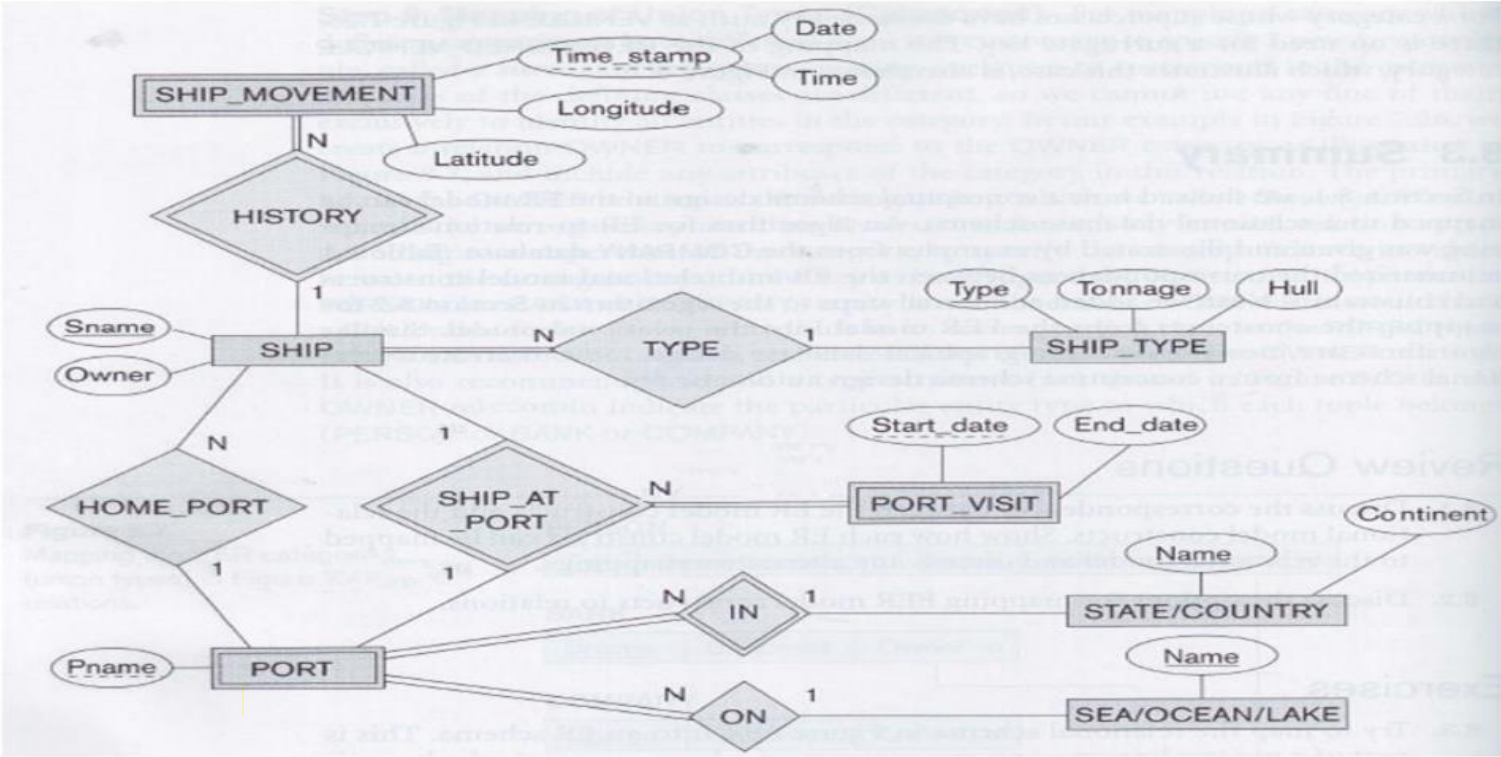


ER Diagram for Railway Reservation System



ER Diagram for Ship Tracking system

Map the following ER diagram of a Ship Tracking database to a relational schema:



Relational Algebra-operators

- **RELATIONAL ALGEBRA** is a widely used procedural query language.(it tells what data to be retrieved and how to be retrieved.)
- The purpose of a query language is to retrieve data from database or perform various operations such as insert, update, delete on the data.

The fundamental operations of relational algebra are as follows −

- Select
- Project
- Union
- Set difference
- Cartesian product
- Rename

- **Unary Relational Operations**

1. SELECT (symbol: σ)
2. PROJECT (symbol: π)
3. RENAME (symbol: ρ)

- **Relational Algebra Operations From Set Theory**

1. UNION (υ)
2. INTERSECTION (⋂),
3. DIFFERENCE (-)
4. CARTESIAN PRODUCT (x)

- **Binary Relational Operations**

1. JOIN
2. Division

SELECT (σ)

- The SELECT operation is used for selecting a subset of the tuples according to a given selection condition.
- Sigma(σ)Symbol denotes it.
- It is used as an expression to choose tuples which meet the selection condition.
- Select operator selects tuples that satisfy a given predicate.

Notation

$\sigma_p(r)$

σ is the predicate
r stands for relation which is the name of the table
p is prepositional logic formula which may use connectors like: AND OR and NOT. These relational can use as relational operators like =, ≠, ≥, <, >, ≤.

- **Example 1**

$$\sigma_{\text{topic = "Database"}} (\text{Tutorials})$$

Output –

Selects tuples from Tutorials where topic = ‘Database’.

- **Example 2**

$\sigma_{\text{topic = "Database" and author = “navathe"}}(\text{Tutorials})$

- **Output** –

Selects tuples from Tutorials where the topic is ‘Database’ and ‘author’ is navathe.

- **Example 3**

$$\sigma_{sales > 50000}(Customers)$$

- **Output** –

Selects tuples from Customers where sales is greater than 50000

Q.1 Select tuples from a relation “Books” where subject is “database”

Q.2 Select tuples from a relation “Books” where subject is “database” and price is “450”

Q.3 Select tuples from a relation “Books” where subject is “database” and price is “450” or have a publication year after 2010

A.1 $\sigma_{\text{subject = “database”}}$ (Books)

- A.2 $\sigma_{\text{subject} = \text{“database”} \wedge \text{price} = \text{“450”}}$ (Books)

- $\sigma_{\text{subject} = \text{“database”} \wedge \text{price} = \text{“450”} \vee \text{year} > \text{”2010″}}$ (Books)