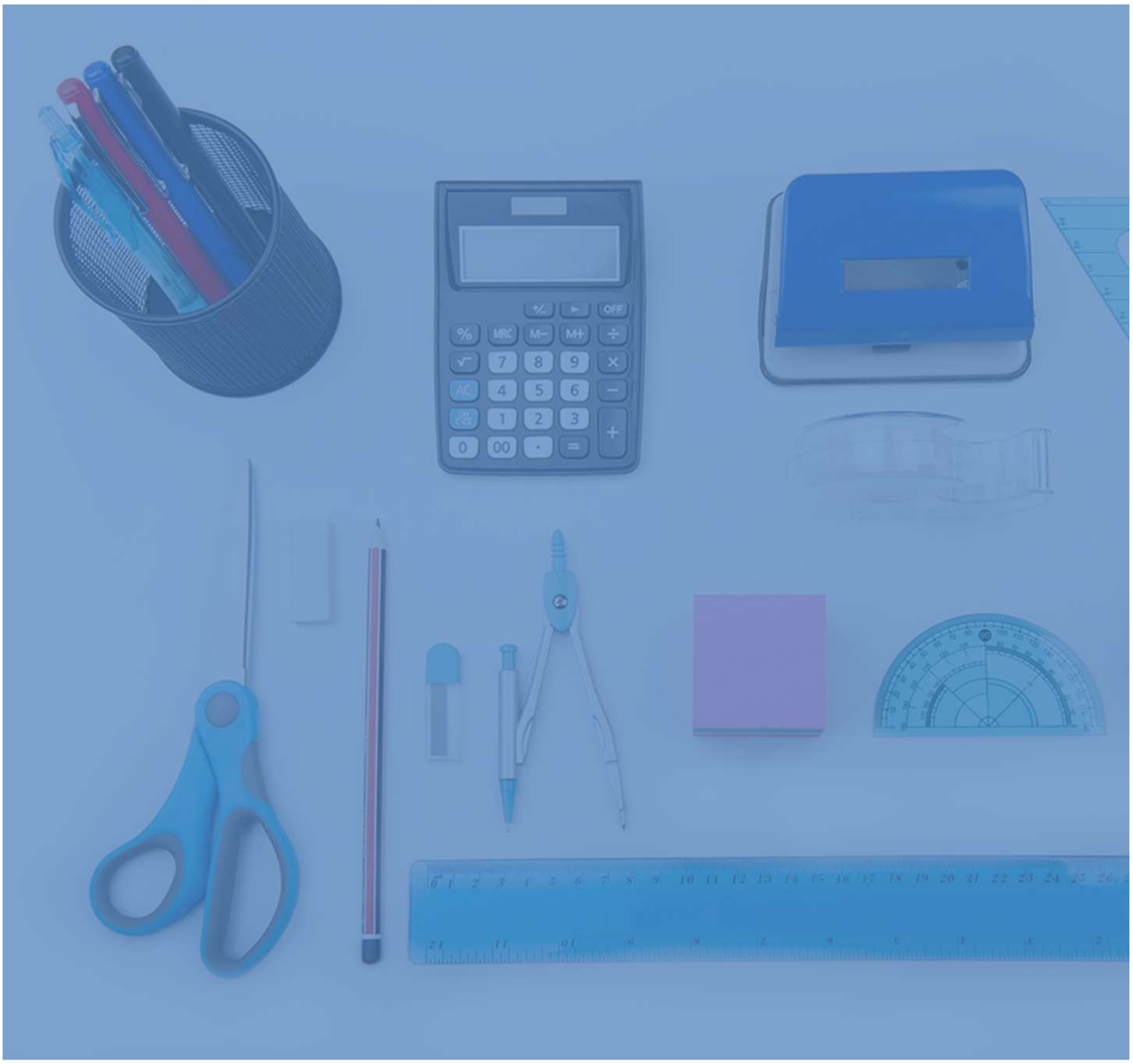
Week-4

Introduction to Database Systems

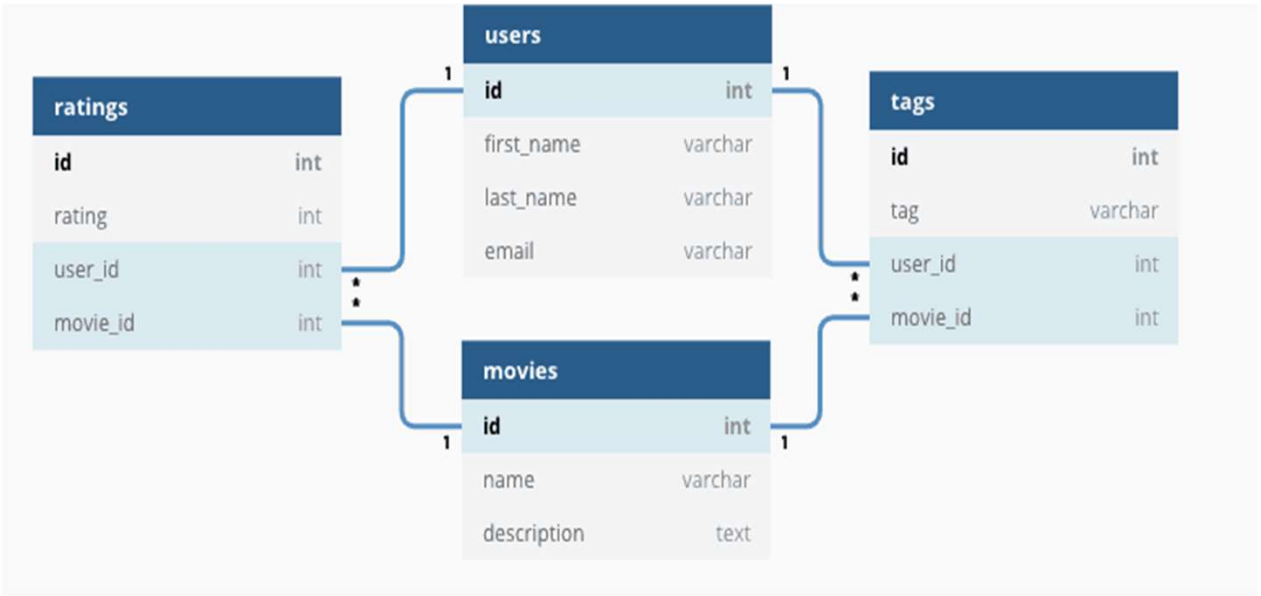
Shah Nawaz



ER into Relational Schema

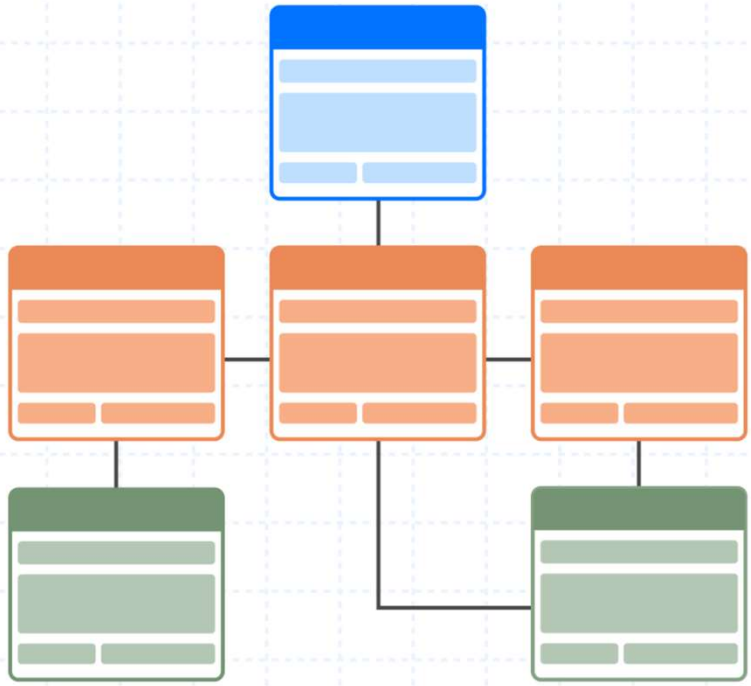
Relational Database

- A relational database (RDB) is a way of structuring information in tables, rows, and columns.
- RDB has the ability to establish links or relationships between information by joining tables, which makes it easy to understand and gain insights about the relationship between various data points.

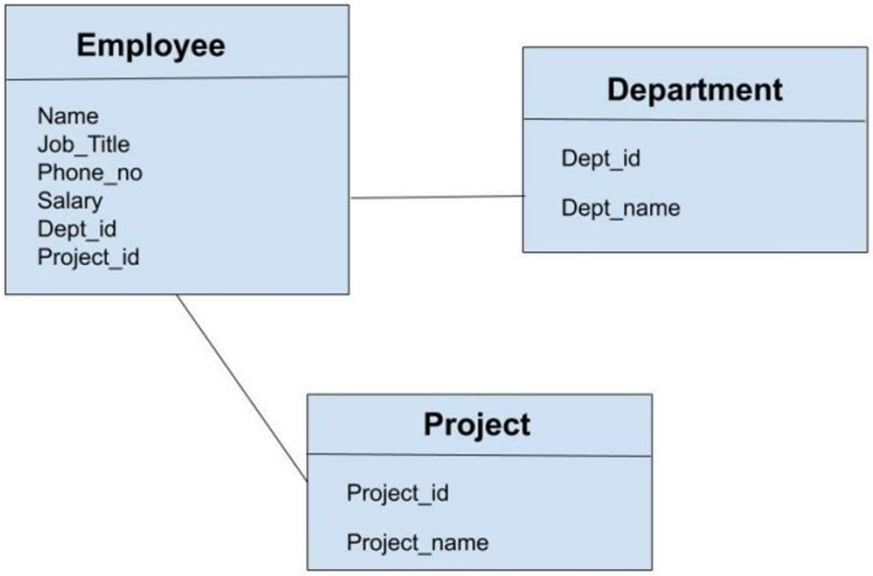


Database Schema

- A database schema is considered the “blueprint” of a database which describes how the data may relate to other tables or other data models.
- However, the schema does not actually contain data.



Database Schema



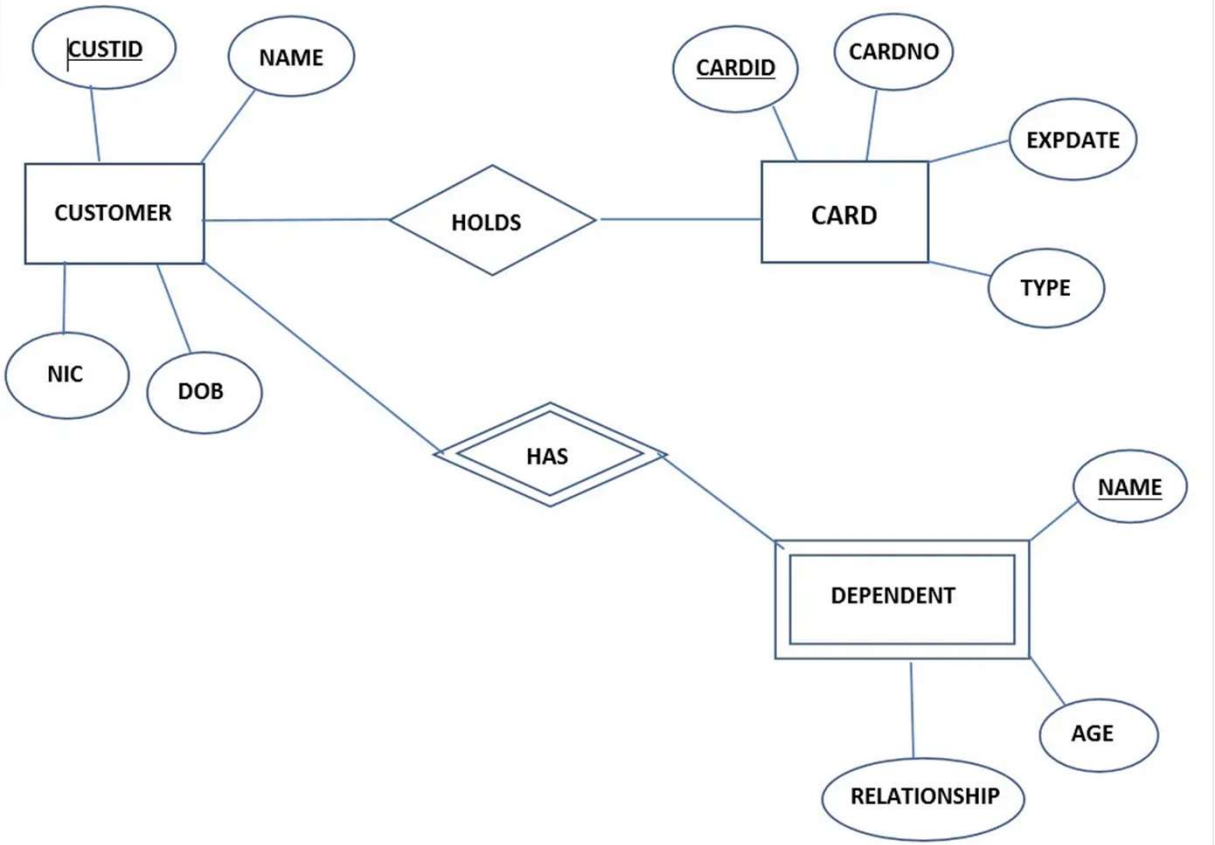
Schema

Conversion

ER into Relational Schema

- **We need to follow some set of rules to achieve this.**
  1. **Mapping strong entities.**
  2. **Mapping weak entities.**
  3. **Map binary one-to-one relations.**
  4. **Map binary one-to-many relations**
  5. **Map binary many-to-many relations.**
  6. **Map multivalued attributes.**
  7. **Map N-ary relations**

1. Mapping strong entities.
2. Mapping weak entities.



1. Mapping strong entities.
2. Mapping weak entities.

- When you going to make a relational schema first you have to identify all entities with their attributes.
- You have to write attributes in brackets as shown here. Definitely you have to underline the primary keys.

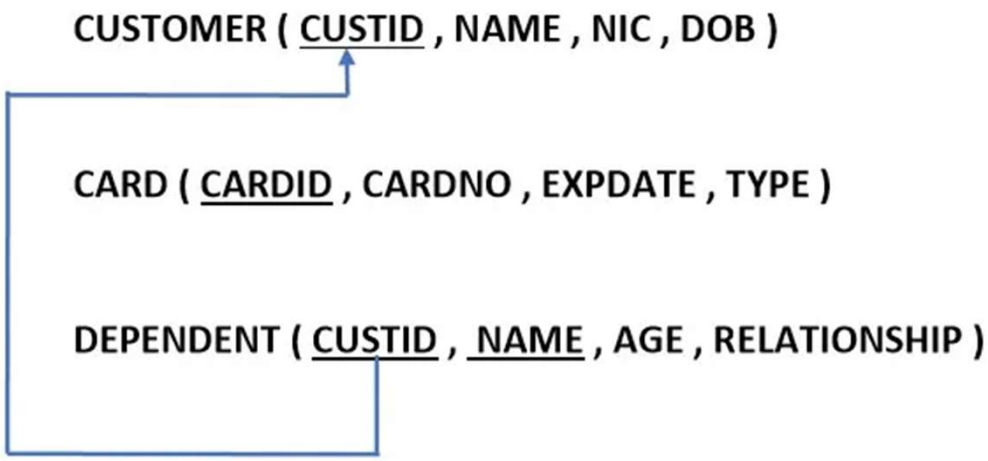
CUSTOMER (CUSTID , NAME , NIC , DOB)

CARD (CARDID , CARDNO , EXPDATE , TYPE)

DEPENDENT (CUSTID , NAME , AGE , RELATIONSHIP)

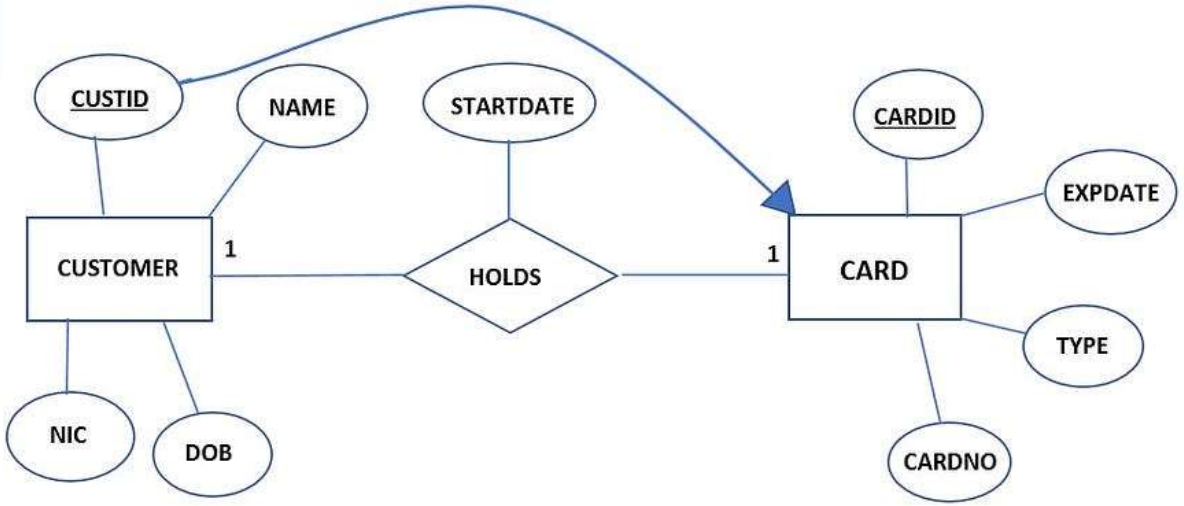
1. Mapping strong entities.
2. Mapping weak entities.

- Here **DEPENDENT** is a weak entity. To make it strong go through the weak relationship and identify the entity which connects with this. Then you have written that entity's primary key inside the weak entity bracket.
- Then you have to map the primary key to where you took from as shown below.

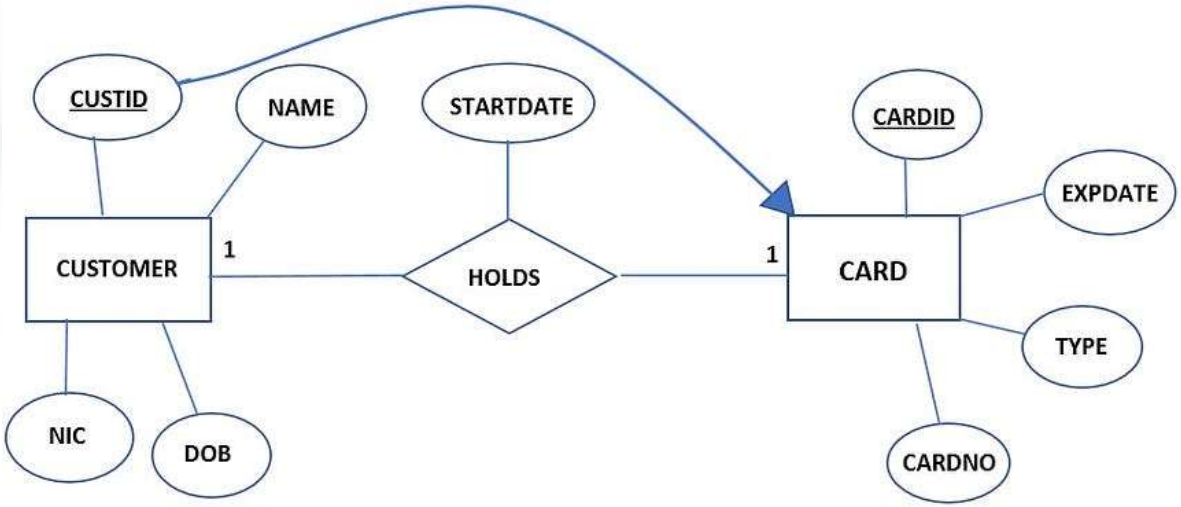


3. Map binary one to one relations.

- Let’s assume that the relationship between CUSTOMER and CARD is one to one.



3. Map binary one to one relations.

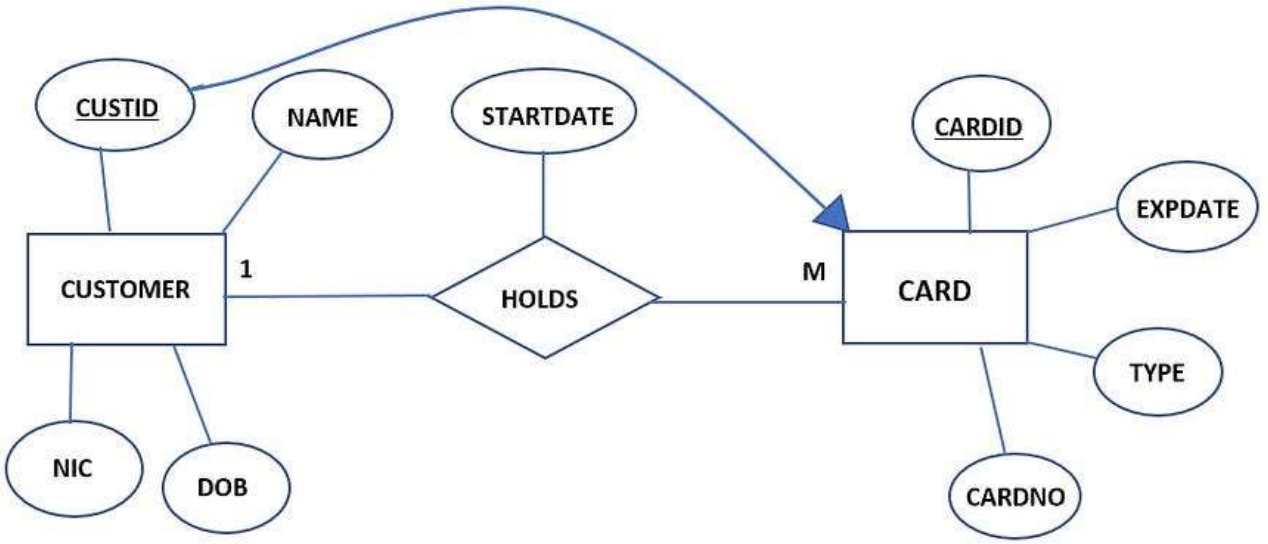


CUSTOMER (CUSTID , NAME , NIC , DOB)

CARD (CARDID , CARDNO , EXPDATE , TYPE , CUSTID ,STARTDATE)

CUST_HOLD(CUSTID , NAME , NIC , DOB , CARDID , CARDNO , EXPDATE , TYPE ,STARTDATE)

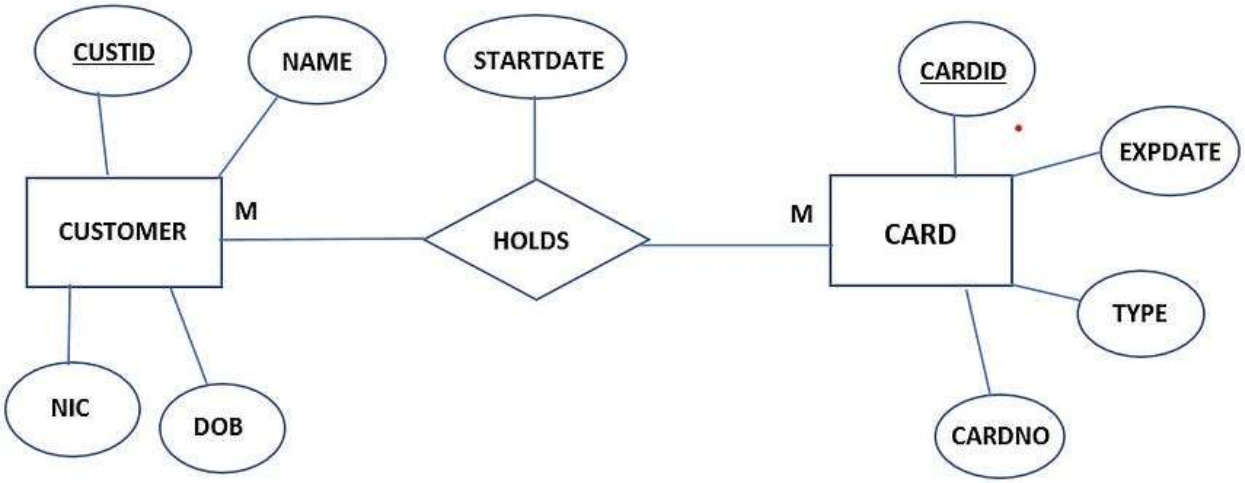
4. Convert 1:N relationship



CUSTOMER (CUSTID , NAME , NIC , DOB)

CARD (CARDID , CARDNO , EXPDATE , TYPE , CUSTID ,STARTDATE)

5. Convert M:N relationship

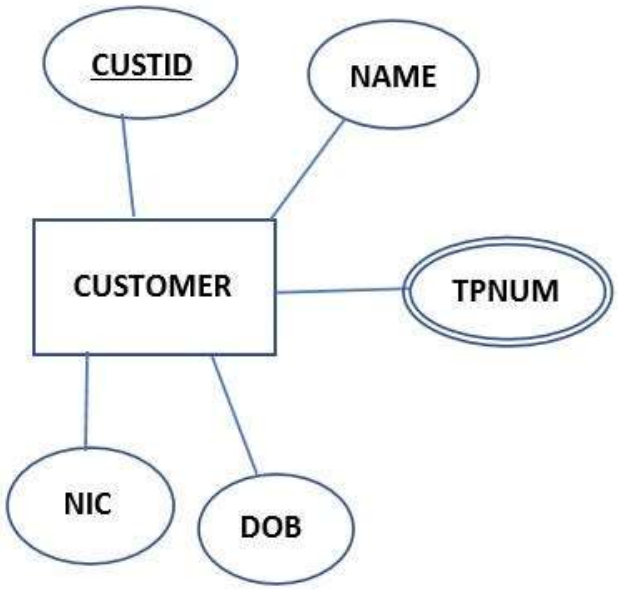


CUSTOMER (CUSTID , NAME , NIC , DOB)

CARD (CARDID , CARDNO , EXPDATE)

HOLDS (CUSTID , CARDID , STARTDATE)

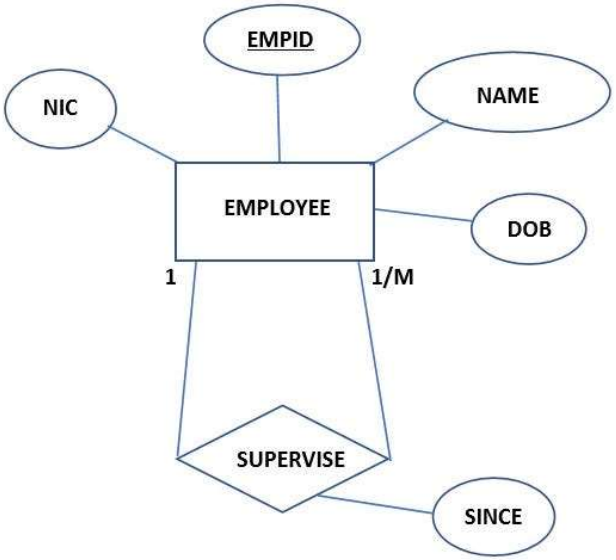
6. Map Multivalued attributes

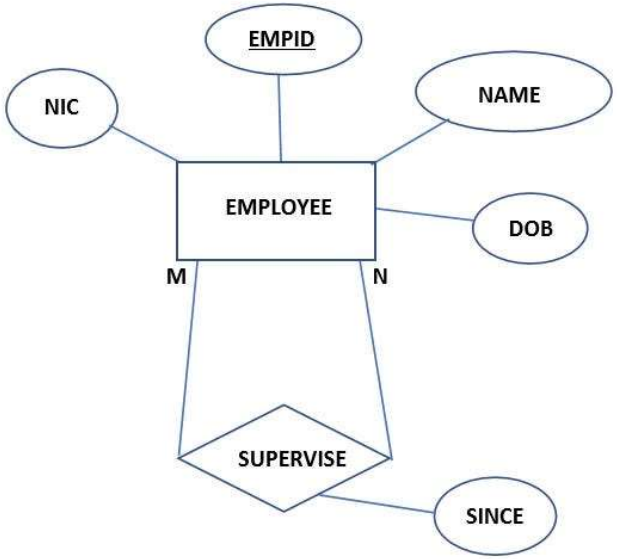


CUSTOMER (CUSTID , NAME , NIC , DOB)

CUST_TP (CUSTID , TPNUM)

7. Map N-ary relations

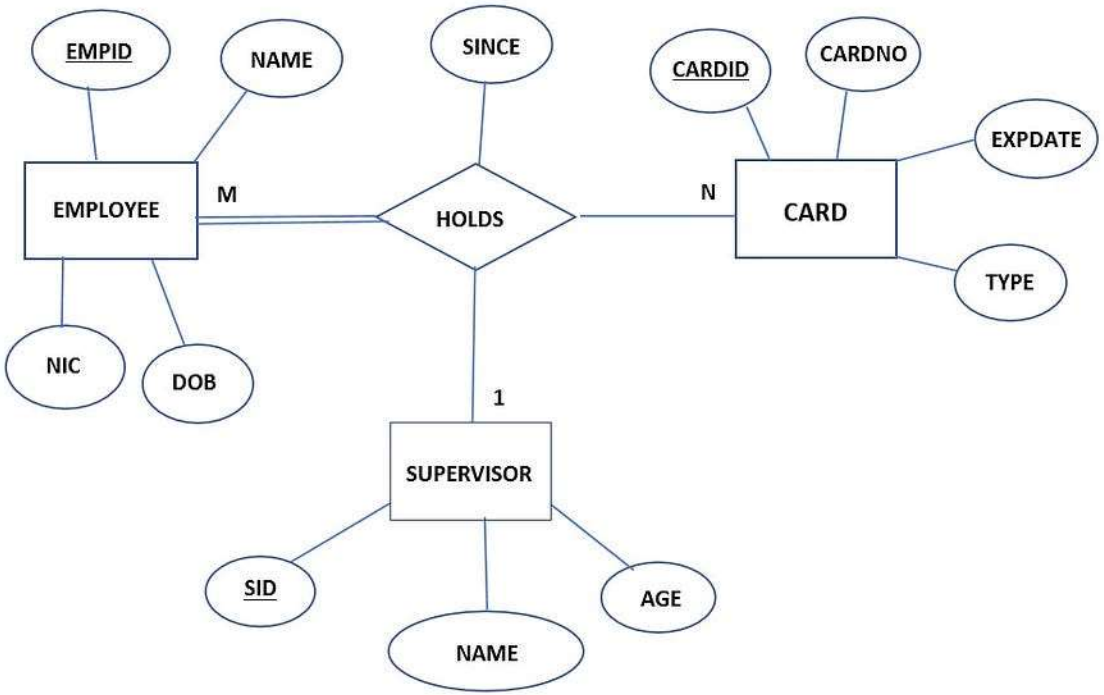




EMPLOYEE (EMPID , NAME , NIC , DOB)

SUPERVISOR (EMPID , SID , SINCE)

Resultant Logical Schema



EMPLOYEE (EMPID , NAME , NIC , DOB)

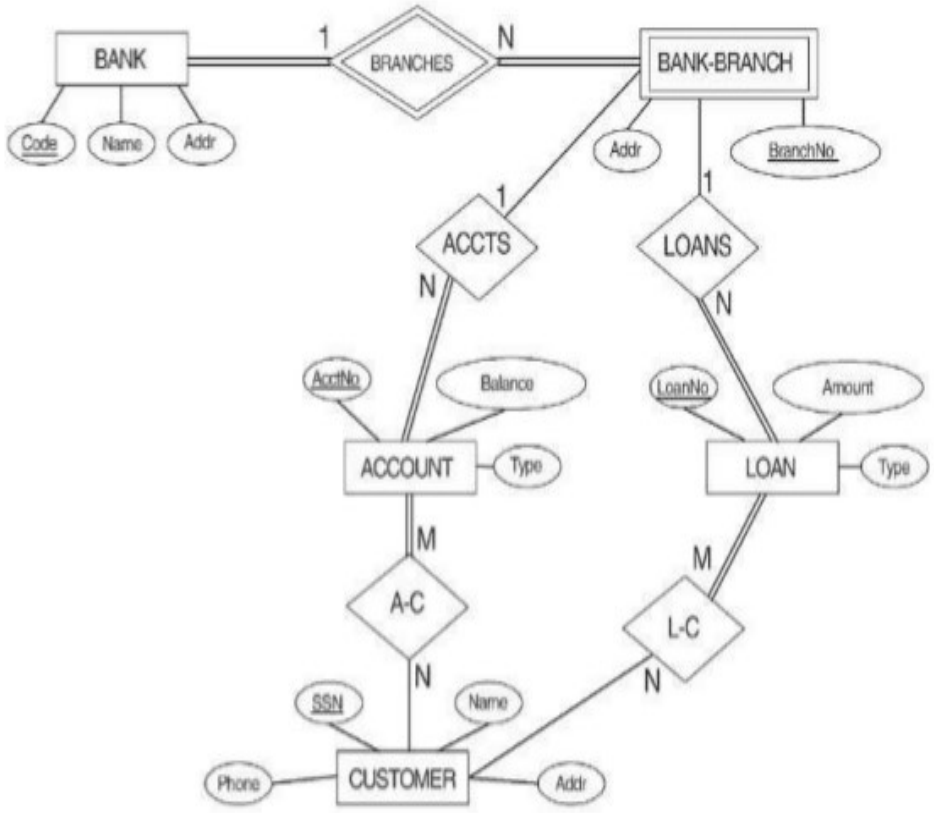
CARD (CARDID , CARDNO , EXPDATE , TYPE)

SUPERVISOR (SID , NAME , AGE)

HOLDS (EMPID , CARDID , SID , SINCE)

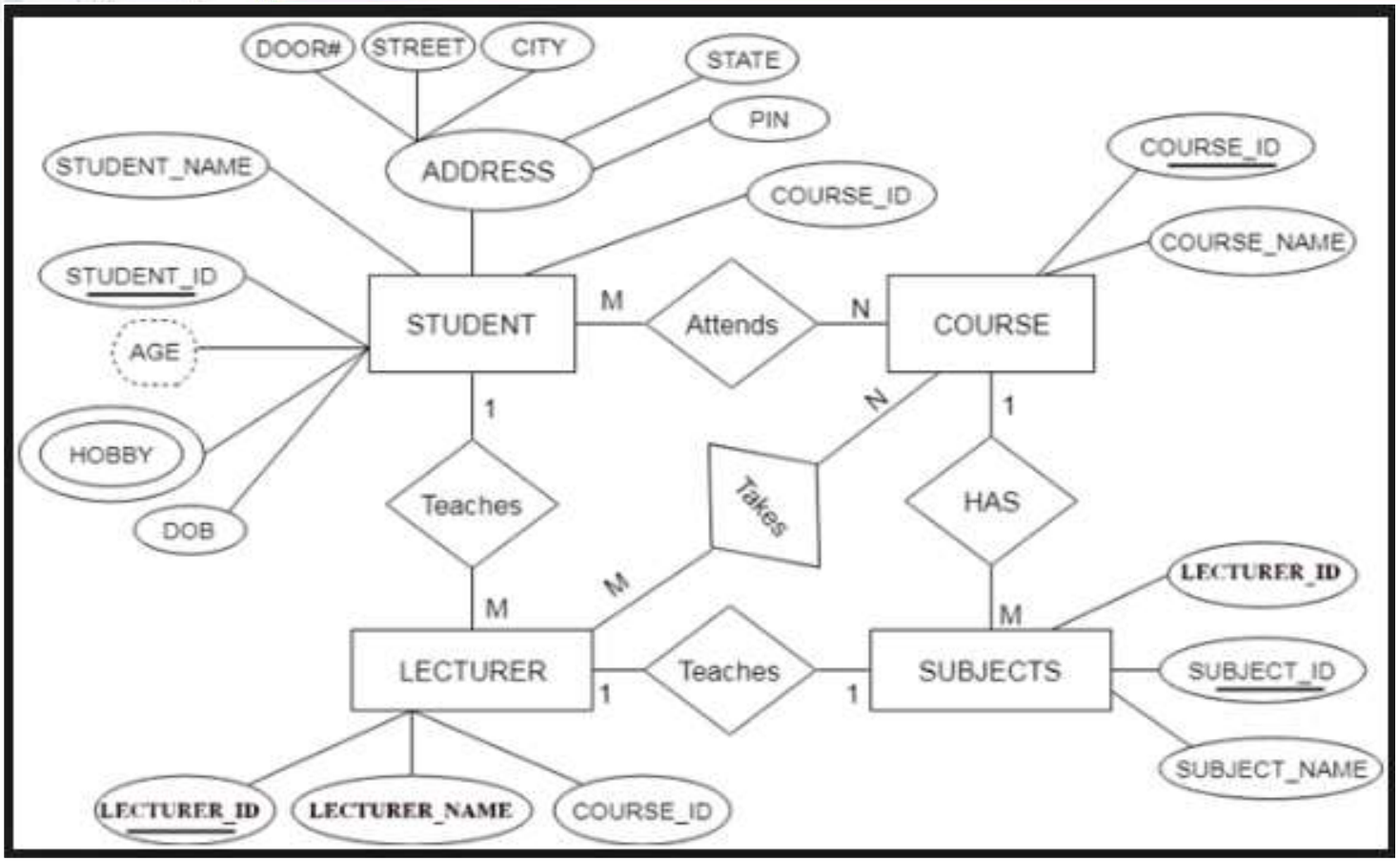
Class activity 1

Convert following ER to Relational Schema



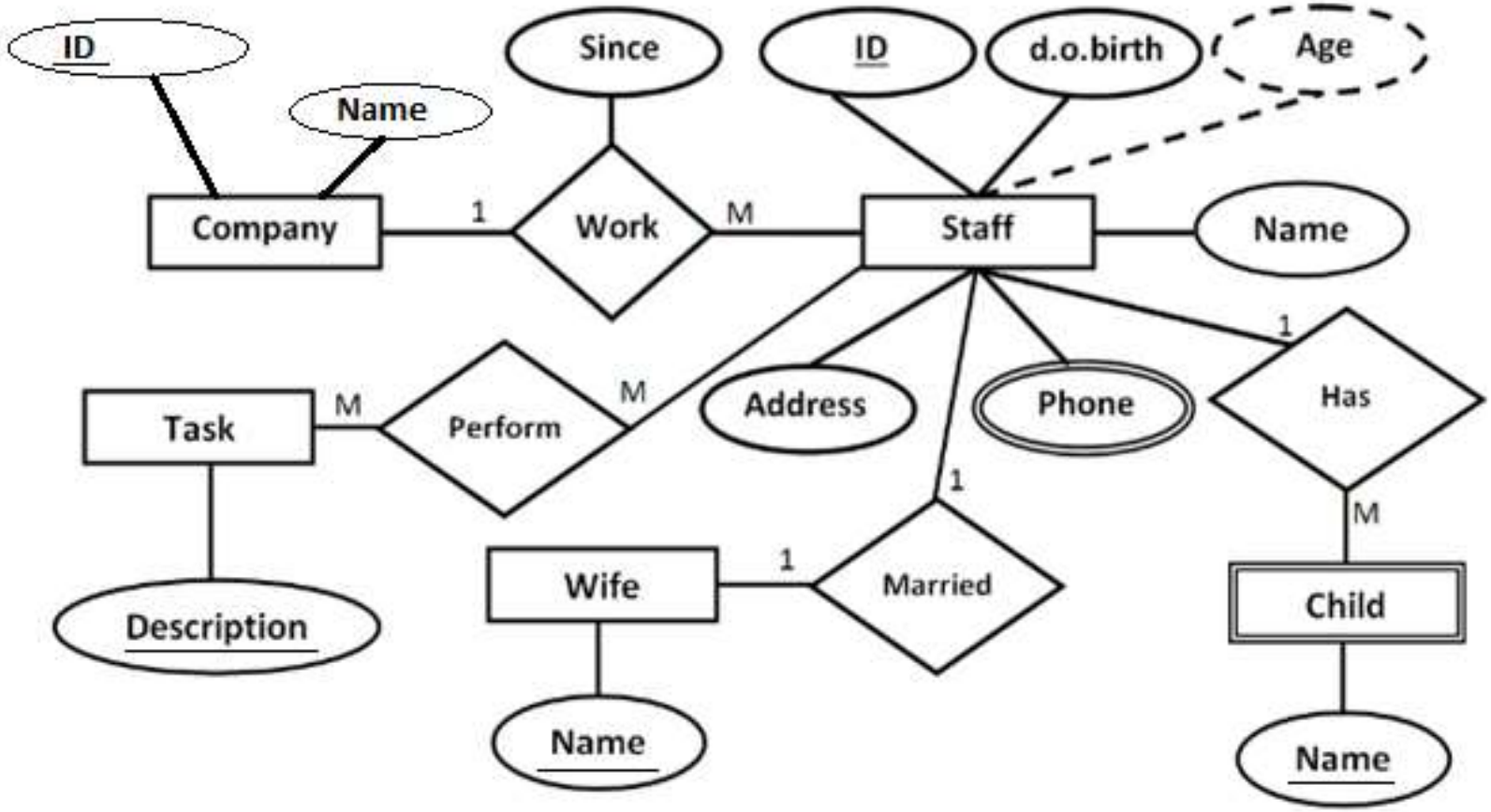
Class Activity 2

Convert following ER to Relational Schema



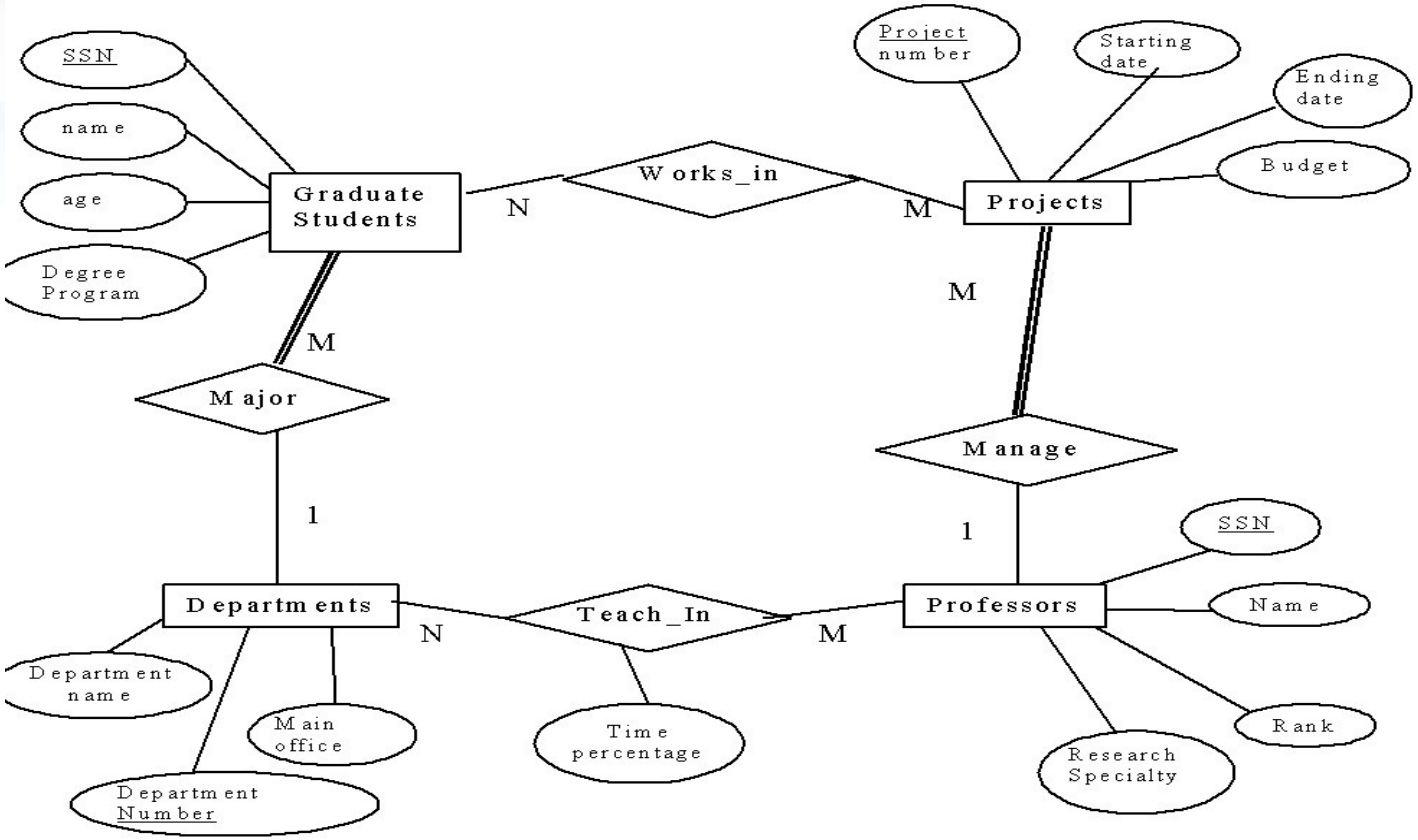
Class Activity 3

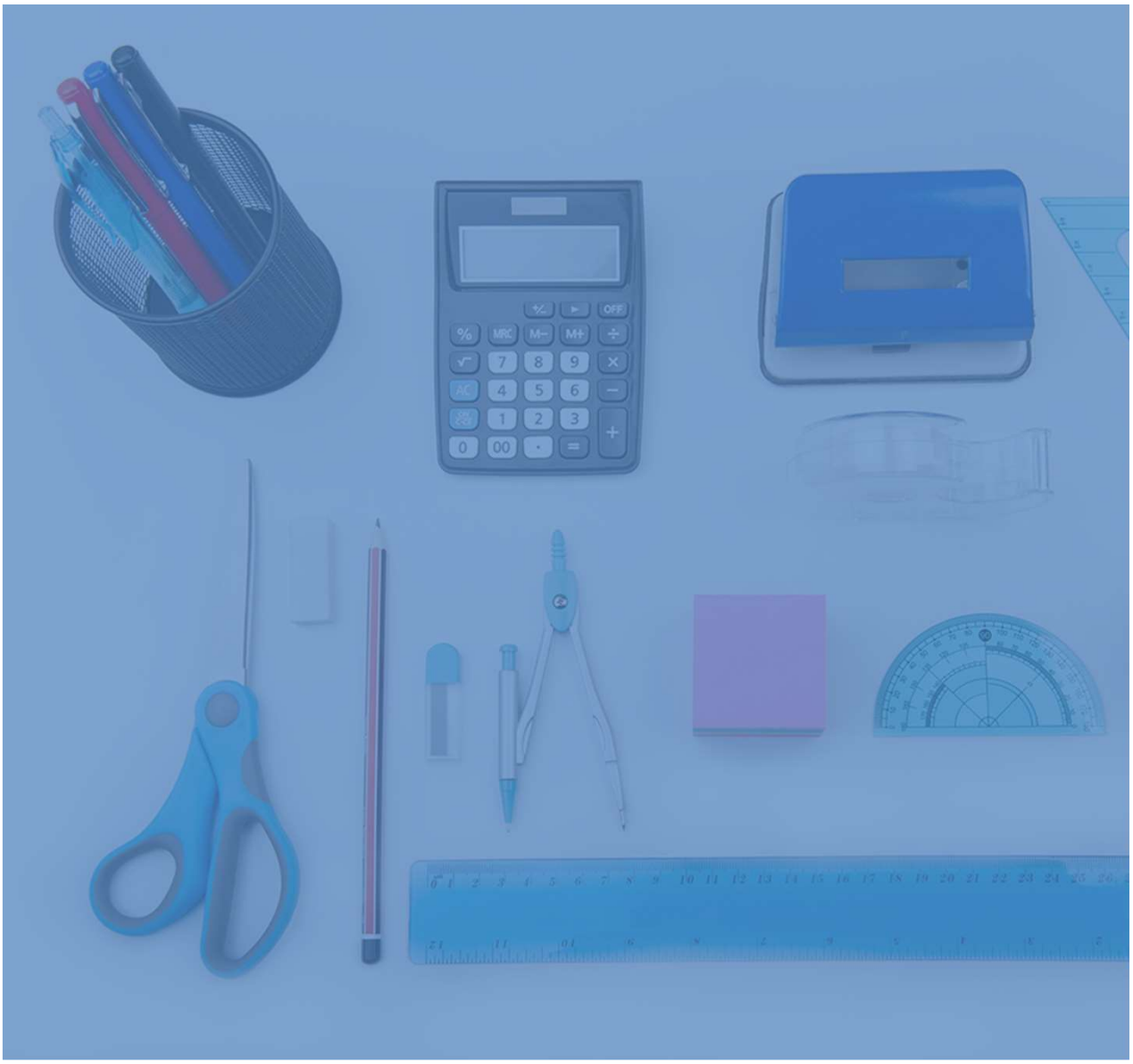
Convert following ER to Relational Schema



Class Activity 4

Convert following ER to Relational Schema





Relational Model

Relational Model

- E.F. Codd proposed the relational Model to model data in the form of relations or tables.
- The relational model represents how data is stored in Relational Databases. A relational database consists of a collection of tables, each of which is assigned a unique name.
- After designing the conceptual model of the Database using ER diagram, we need to convert the conceptual model into a relational model which can be implemented using any RDBMS language like Oracle SQL, MySQL, etc.

Relational Model

- Consider a relation STUDENT with attributes ROLL_NO, NAME, ADDRESS, PHONE, and AGE shown in the table.

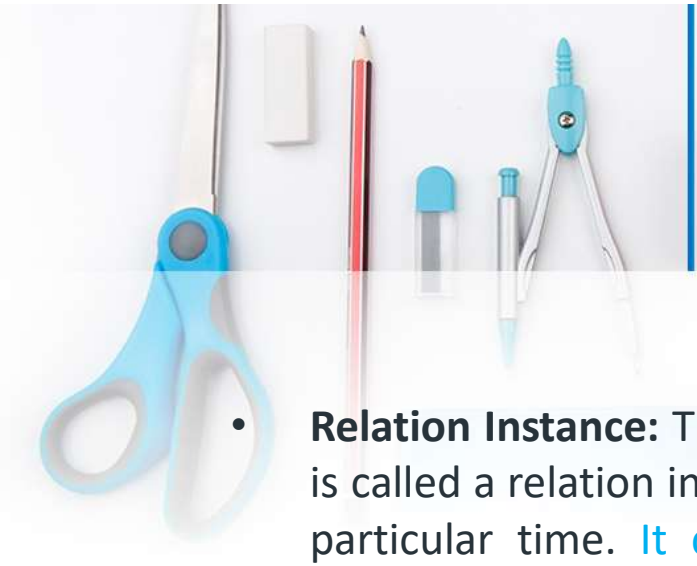
Table Student

ROLL_NO	NAME	ADDRESS	PHONE	AGE
1	Ali	Multan	9455123451	19
2	Kiran	Okara	9652431543	17
3	Khalid	Kasur	9156253131	20
4	Noor	Lahore		18

Relational Model- Important Terminologies

- **Attribute:** Attributes are the properties that define an entity. e.g.; **ROLL_NO**, **NAME, ADDRESS**
- **Relation Schema:** A relation schema defines the structure of the relation and represents the name of the relation with its attributes. e.g.;
- **STUDENT (ROLL_NO, NAME, ADDRESS, PHONE, and AGE)** is the relation schema for STUDENT. If a schema has more than 1 relation, it is called Relational Schema.
- **Tuple:** Each row in the relation is known as a tuple. The STUDENT relation contains 4 tuples, one of which is shown as:

1	Ali	Multan	9455123451	19



Relational Model- Important Terminologies

- **Relation Instance:** The set of tuples of a relation at a particular instance of time is called a relation instance. Table 1 shows the relation instance of STUDENT at a particular time. It can change whenever there is an insertion, deletion, or update in the database.
- **Degree:** The number of attributes in the relation is known as the degree of the relation. The **STUDENT** relation defined above has degree 5.
- **Cardinality:** The number of tuples in a relation is known as cardinality. The **STUDENT** relation defined above has cardinality 4.
- **Column:** The column represents the set of values for a particular attribute. The column **NAME** is extracted from the relation STUDENT.

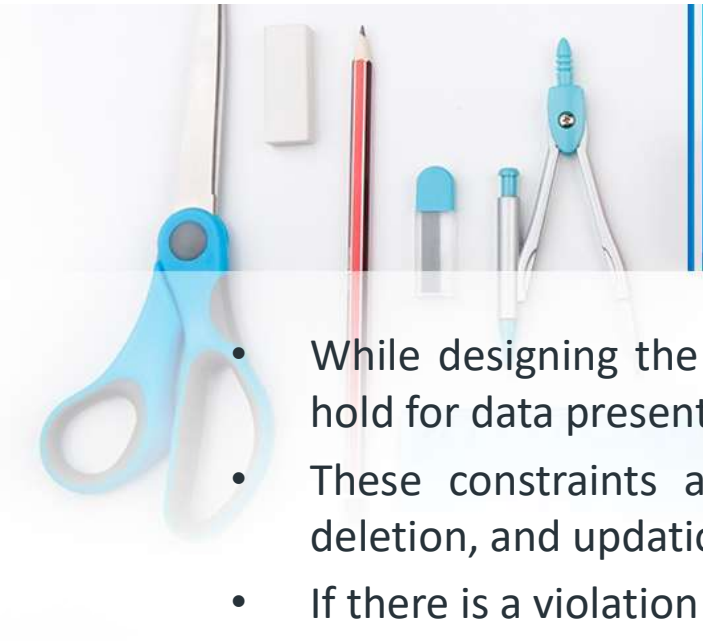
NAME
Ali
Kiran
Khalid
Noor

Relational Model- Important Terminologies

- **NULL Values:** The value which is not known or unavailable is called a NULL value. It is represented by blank space. e.g.; PHONE of STUDENT having ROLL_NO 4 is NULL.

4	Noor	Lahore		18

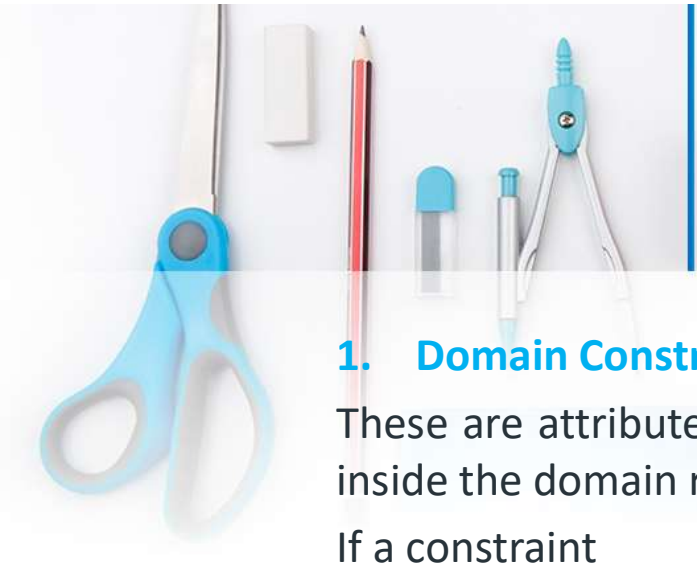
- **Relation Key:** These are basically the keys that are used to identify the rows uniquely or also help in identifying tables. These are of the following types.
  - Primary Key
  - Candidate Key
  - Super Key
  - Foreign Key
  - Alternate Key
  - Composite Key



Constraints in Relational Model

- While designing the Relational Model, we define some conditions which must hold for data present in the database are called Constraints.
- These constraints are checked before performing any operation (insertion, deletion, and updation) in the database.
- If there is a violation of any of the constraints, the operation will fail.

- Constraints have following three types:
  1. Domain Constraints
  2. Key Integrity Constraints
  3. Referential Integrity Constraints



Constraints in Relational Model

1. Domain Constraints

These are attribute-level constraints. An attribute can only take values that lie inside the domain range. e.g.;

If a constraint

AGE>0

is applied to STUDENT relation, inserting a negative value of AGE will result in failure.

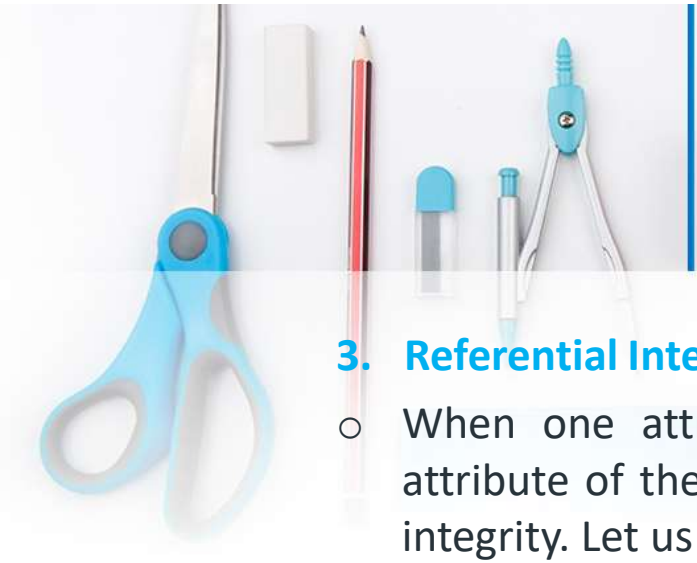
Constraints in Relational Model

2. Key Integrity Constraints

- Every relation in the database should have at least one set of attributes that defines a tuple uniquely. Those set of attributes is called keys. e.g.; ROLL_NO in STUDENT Relation is key. No two students can have the same roll number. So a key has two properties:
- It should be unique for all tuples.
- It can't have NULL values.

STUDENT

ROLL_NO	NAME	ADDRESS	PHONE	AGE
1	Ali	Multan	9455123451	19
2	Kiran	Okara	9652431543	17
3	Khalid	Kasur	9156253131	20
4	Noor	Lahore		18



Constraints in Relational Model

3. Referential Integrity Constraints

- When one attribute of a relation can only take values from another attribute of the same relation or any other relation, it is called referential integrity. Let us suppose we have 2 relations

Student

ROLL_NO	NAME	ADDRESS	PHONE	AGE	Dept_Code
1	Ali	Multan	9455123451	19	CS
2	Kiran	Okara	9652431543	17	MT
3	Khalid	Kasur	9156253131	20	Ph
4	Noor	Lahore		18	Ch

Department

Dept_Code	Dept_ Name
CS	Computer Science
MT	Mathematics
Ph	Physics
Ch	Chemistry

Constraints in Relational Model

3. Referential Integrity

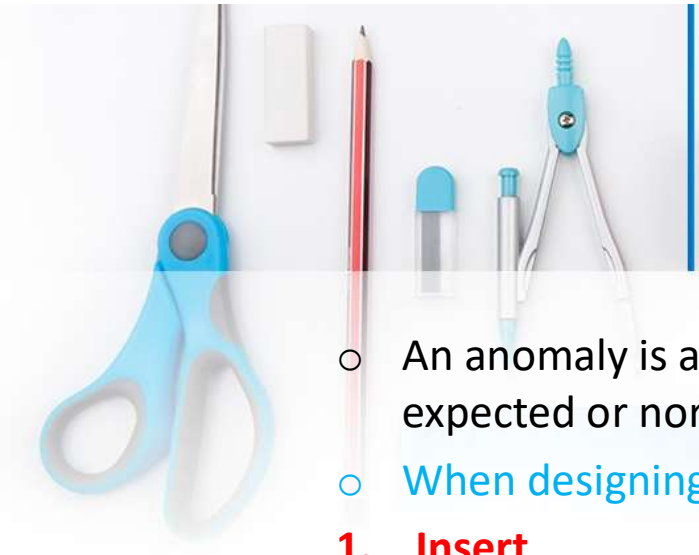
Table Student

ROLL_NO	NAME	ADDRESS	PHONE	AGE	Dept_Code
1	Ali	Multan	9455123451	19	CS
2	Kiran	Okara	9652431543	17	MT
3	Khalid	Kasur	9156253131	20	Ph
4	Noor	Lahore		18	Ch

Table Department

Dept_Code	Dept_ Name
CS	Computer Science
MT	Mathematics
Ph	Physics
Ch	Chemistry

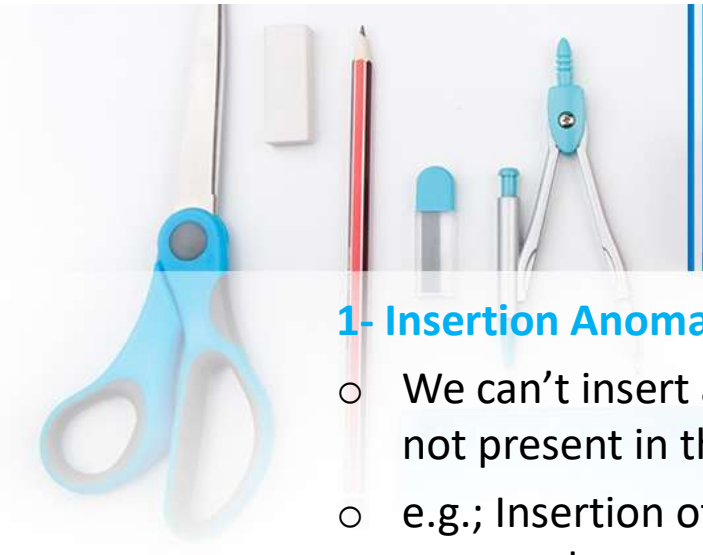
- DEPT_CODE of STUDENT can only take the values which are present in DEPT_CODE of DEPARTMENT which is called referential integrity constraint.
- The relation which is referencing another relation is called **REFERENCING RELATION** (STUDENT in this case) and the relation to which other relations refer is called **REFERENCED RELATION** (DEPARTMENT in this case).



Anomalies in Relational Model

- An anomaly is an irregularity or something which deviates from the expected or normal state.
- When designing databases, we identify three types of anomalies:

1. **Insert,**
2. **Update, and**
3. **Delete.**



Anomalies in Relational Model

1- Insertion Anomaly in Referencing Relation

- We can't insert a row in REFERENCING RELATION if referencing attribute's value is not present in the referenced attribute value.
- e.g.; Insertion of a student with DEPT_CODE 'IT' in STUDENT relation will result in an error because 'IT' is not present in DEPT_CODE of DEPARTMENT.

ROLL_NO	NAME	ADDRESS	PHONE	AGE	Dept_Code
1	Ali	Multan	9455123451	19	CS
2	Kiran	Okara	9652431543	17	MT
3	Khalid	Kasur	9156253131	20	Ph
4	Noor	Lahore		18	Ch
5	Nida	Lahore	919244334	20	IT

Dept_Code	Dept_ Name
CS	Computer Science
MT	Mathematics
Ph	Physics
Ch	Chemistry

Anomalies in Relational Model

2 - Deletion/ Updation Anomaly in Referenced Relation:

- We can't delete or update a row from REFERENCED RELATION if the value of REFERENCED ATTRIBUTE is used in the value of REFERENCING ATTRIBUTE.
- e.g; if we try to delete a tuple from DEPARTMENT having DEPT_CODE 'CS', it will result in an error because 'CS' is referenced by DEPT_CODE of STUDENT, but if we try to delete the row from DEPARTMENT with BRANCH_CODE IT, it will be deleted as the value is not been used by referencing relation.
- It can be handled by the following method:
  - On Delete Cascade
  - On Update Cascade

Anomalies in Relational Model

2 - Deletion/ Updation Anomaly in Referenced Relation:

- On Delete Cascade
  - It will delete the tuples from REFERENCING RELATION if the value used by REFERENCING ATTRIBUTE is deleted from REFERENCED RELATION.
  - e.g.; For, if we delete a row from DEPARTMENT with DEPT_CODE 'CS', the rows in STUDENT relation with DEPT_CODE CS (ROLL_NO 1 and 2 in this case) will be deleted.

ROLL_NO	NAME	ADDRESS	PHONE	AGE	Dept_Code
1	Ali	Multan	9455123451	19	CS
2	Kiran	Okara	9652431543	17	MT
3	Khalid	Kasur	9156253131	20	Ph

Dept_Code	Dept_ Name
CS	Computer Science
MT	Mathematics
Ph	Physics
Ch	Chemistry

Anomalies in Relational Model

2 - Deletion/ Updation Anomaly in Referenced Relation:

○ On Update Cascade

- It will update the REFERENCING ATTRIBUTE in REFERENCING RELATION if the attribute value used by REFERENCING ATTRIBUTE is updated in REFERENCED RELATION.
- e.g;, if we update a row from DEPARTMENT with DEPT_CODE ‘CS’ to ‘CSE’, the rows in STUDENT relation with DEPT_CODE CS (ROLL_NO 1 and 2 in this case) will be updated with DEPT_CODE ‘CSE’.

ROLL_NO	NAME	ADDRESS	PHONE	AGE	Dept_Code
1	Ali	Multan	9455123451	19	CS
2	Kiran	Okara	9652431543	17	MT
3	Khalid	Kasur	9156253131	20	Ph

Dept_Code	Dept_ Name
CS	Computer Science
MT	Mathematics
Ph	Physics
Ch	Chemistry

E.F. Codd rules of Relational Model

1. **Foundation Rule -** The database must be able to manage data in relational form.
2. **Information Rule -** All data stored in the database must exist as a value of some table cell.
3. **Guaranteed Access Rule -** Every unique data element should be accessible by only a combination of the table name, primary key value, and the column name.
4. **Systematic Treatment of NULL values -** Database must support NULL values.
5. **Active Online Catalog -** The organization of the database must exist in an online catalog that can be queried by authorized users.
6. **Comprehensive Data Sub-Language Rule -** Database must support at least one language that supports: data definition, view definition, data manipulation, integrity constraints, authorization, and transaction boundaries.
7. **View Updating Rule -** All views should be theoretically and practically updatable by the system.

E.F. Codd rules of Relational Model

8. **Relational Level Operation Rule -** The database must support high-level insertion, updation, and deletion operations.
9. **Physical Data Independence Rule -** Data stored in the database must be independent of the applications that can access it i.e., the data stored in the database must not depend on any other data or an application.
10. **Logical Data Independence Rule -** Any change in the logical representation of the data (structure of the tables) must not affect the user's view.
11. **Integrity independence -** Changing the integrity constraints at the database level should not reflect any change at the application level.
12. **Distribution independence -** The database must work properly even if the data is stored in multiple locations or is being used by multiple end-users.
13. **Non-subversion Rule -** Accessing the data by low-level relational language should not be able to bypass the integrity rules and constraints expressed in the high-level relational language.

Thank You all!