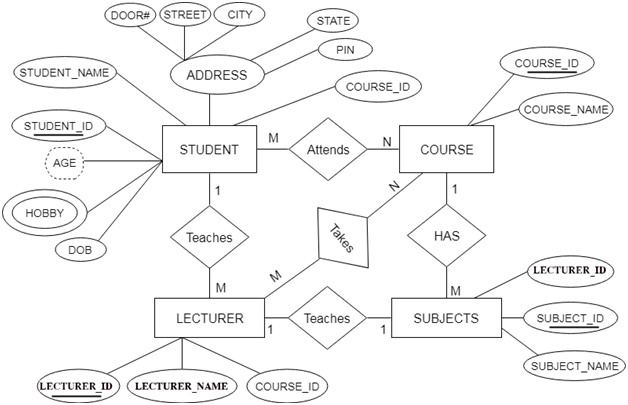
**TASK – 1**

**Convert ER Diagram into Relational Model**

**AIM:** To convert the given Entity-Relationship (ER) diagram into a relational model following standard database design mapping rules**.**

**DIAGRAM:**



**Steps for converting the ER diagram to the table**

1. **Entity type becomes a table**

each entity type in the ER diagram becomes a table in the relational model.

From the ER diagram, the main entity types are:

* STUDENT
* LECTURER
* COURSE
* SUBJECTS

Each of these will be represented as a separate table in your relational database design.

For example:

* The STUDENT entity becomes the STUDENT table.
* The LECTURER entity becomes the LECTURER table.
* The COURSE entity becomes the COURSE table.
* The SUBJECTS entity becomes the SUBJECT table.

Each table will then contain columns that represent that entity's attributes (fields such as STUDENT\_ID, STUDENT\_NAME for the STUDENT table, and so on).

**STUDENT**

| **Column Name** | **Description** |
| --- | --- |
| STUDENT\_ID | Primary Key |
| STUDENT\_NAME | Student name |
| DOB | Date of birth |
| DOOR# | Address component |
| STREET | Address component |
| CITY | Address component |
| STATE | Address component |
| PIN | Address component |
| COURSE\_ID | Foreign Key referencing COURSE |

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

**Rule:**All single-valued attributes in the ER diagram become columns for the corresponding table in the relational model**.**

**Application to Provided ER Diagram:**

For each entity type, identify the single-valued attributes (those represented by ovals and directly connected to the entity, not multivalued or composite):

* **STUDENT:**
  + STUDENT\_ID
  + STUDENT\_NAME
  + DOB
  + COURSE\_ID
* **LECTURER:**
  + LECTURER\_ID
  + LECTURER\_NAME
  + COURSE\_ID
* **COURSE:**
  + COURSE\_ID
  + COURSE\_NAME
* **SUBJECTS:**
  + SUBJECT\_ID
  + SUBJECT\_NAME
  + LECTURER\_ID
* **ADDRESS (Composite, but components are single-valued):**
  + DOOR#
  + STREET
  + CITY
  + STATE
  + PIN
  + **COURSE**

| * + **Column Name** | * + **Description** |
| --- | --- |
| * + COURSE\_ID | * + Primary Key |
| * + COURSE\_NAME | * + Name of the course |
|  |  |

**. LECTURER**

| **Column Name** | **Description** |
| --- | --- |
| LECTURER\_ID | Primary Key |
| LECTURER\_NAME | Lecturer's name |
| COURSE\_ID | Foreign Key referencing COURSE |

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

**Rule:**A key attribute of each entity type in the ER diagram is represented as the primary key in the corresponding table in the relational model.

**Application to Provided ER Diagram**:

For every entity:

* Identify the attribute that uniquely distinguishes each instance of that entity (usually underlined in the ER diagram).
* This attribute becomes the primary key for the table created for that entity.

**Primary Keys for the Entities**:

* **STUDENT:**
  + STUDENT\_ID *(Primary Key)*
* **LECTURER:**
  + LECTURER\_ID *(Primary Key)*
* **COURSE:**
  + COURSE\_ID *(Primary Key)*
* **SUBJECTS:**
  + SUBJECT\_ID *(Primary Key)*

**SUBJECT**

| **Column Name** | **Description** |
| --- | --- |
| **SUBJECT\_ID** | **Primary Key** |
| **SUBJECT\_NAME** | **Subject name** |
| **LECTURER\_ID** | **Foreign Key referencing LECTURER** |

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

**Rule:**A multivalued attribute is represented by a separate table in the relational model.

**Application to Provided ER Diagram:**

In your ER diagram, the entity STUDENT has a multivalued attribute: HOBBY.

Implementation:

* You create a new table (e.g., STUD\_HOBBY) to represent the combination of the student and their hobbies.
* Each row in this new table links a student's primary key to one of their hobbies.

**Table Structure Example:**

| **STUDENT\_ID (FK)** | **HOBBY** |
| --- | --- |
| 1001 | Drawing |
| 1001 | Singing |
| 1002 | Sports |

* **STUDENT\_ID:** Foreign Key referencing the STUDENT table**.**
* **HOBBY:** One hobby per row**.**

1. **Composite attribute represented by components**

**Rule:**A composite attribute is represented by its component attributes as individual columns in the table**.**

**Application to Provided ER Diagram:**

In your ER diagram, the ADDRESS attribute for the entity STUDENT is a composite attribute. It consists of the following simple attributes:

* DOOR#
* STREET
* CITY
* STATE
* PIN

**Implementation:**

* Instead of creating a single column for ADDRESS, you create separate columns for each of its components in the STUDENT table.

Resulting Columns in the STUDENT Table:

* DOOR#
* STREET
* CITY
* STATE
* PIN

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

**Rule:**Derived attributes are not considered in the table when converting an ER diagram to a relational model.

**Application to Provided ER Diagram**:

In your ER diagram, the attribute AGE for the entity STUDENT is shown as a derived attribute (usually denoted by a dashed oval). Derived attributes can be calculated from other data stored in the database (e.g., AGE can be derived from DOB).

**Implementation:**

* Do not create a column for derived attributes (like AGE) in the STUDENT table.
* Instead, derive such values using queries or calculations when needed, based on stored attributes like DOB.

**RESULT:**

**Mapping/Relationships:**

* STUDENT.COURSE\_ID → COURSE.COURSE\_ID (many students enroll in a course)
* LECTURER.COURSE\_ID → COURSE.COURSE\_ID (lecturers teach courses)
* SUBJECT.LECTURER\_ID → LECTURER.LECTURER\_ID (lecturers teach subject)
* STUD\_HOBBY.STUDENT\_ID → STUDENT.STUDENT\_ID (students have multiple hobbies)