Vidyavardhini's College of Engineering and Technology, Vasai
Department of Artificial Intelligence & Data Science

AY: 2024-25

Class:	**SE**	**Semester:**	**IV**
Course Code:	**CSL402**	**Course Name:**	**DBMS Lab**

Name of Student:	**Shravani Sandeep Raut**
Roll No. :	**48**
Experiment No.:	**2**
Title of the Experiment:	**Mapping ER/EER to Relational schema model.**
Date of Performance:	**29/01/2025**
Date of Submission:	**05/02/2025**

Evaluation

Performance Indicator	Max. Marks	Marks Obtained
Performance	5	
Understanding	5	
Journal work and timely submission	10	
Total	20	

Performance Indicator	Exceed Expectations (EE)	Meet Expectations (ME)	Below Expectations (BE)
Performance	4-5	2-3	1
Understanding	4-5	2-3	1
Journal work and timely submission	8-10	5-8	1-4

Checked by

Name of Faculty : Ms. Neha Raut

Signature :

Date:

Aim :- Prepare the schema for Relational Model with the ER/ERR diagram, drawn for the identified case study in experiment no.1.

Objective :- To map the Entity Relationship (ER) / Extended Entity-Relationship (EER) Diagram to Relational Model schema and learn to incorporate various schema-based constraints.

Theory:

Mapping an Entity-Relationship (ER) model to a relational database schema involves translating the conceptual model represented in the ER diagram into tables and relationships in a relational database management system (DBMS). Here are the general rules for mapping ER to a schema in a DBMS:

1. Entities to Tables:
    a. Each entity in the ER diagram corresponds to a table in the relational schema.
    b. The attributes of the entity become the columns of the table.
    c. The primary key of the entity becomes the primary key of the table.

2. Relationships to Tables:
    a. Many-to-Many Relationships:
        i. Convert each many-to-many relationship into a new table.
        ii. Include foreign key columns in this table to reference the participating entities.
        iii. The primary key of this table may consist of a combination of the foreign keys from the participating entities.
    b. One-to-Many and One-to-One Relationships:
        i. Represented by foreign key columns in one of the participating tables.
        ii. The table on the "many" side of the relationship includes the foreign key column referencing the table on the "one" side.
        iii. The foreign key column typically references the primary key of the related table.

3. Attributes to Columns:
    a. Each attribute of an entity becomes a column in the corresponding table.
    b. Choose appropriate data types for each attribute based on its domain and constraints.
    c. Ensure that attributes participating in relationships are represented as foreign keys when needed.

4. Primary and Foreign Keys:
    a. Identify the primary key(s) of each table based on the primary key(s) of the corresponding entity.
    b. Ensure referential integrity by defining foreign keys in tables to establish relationships between them.

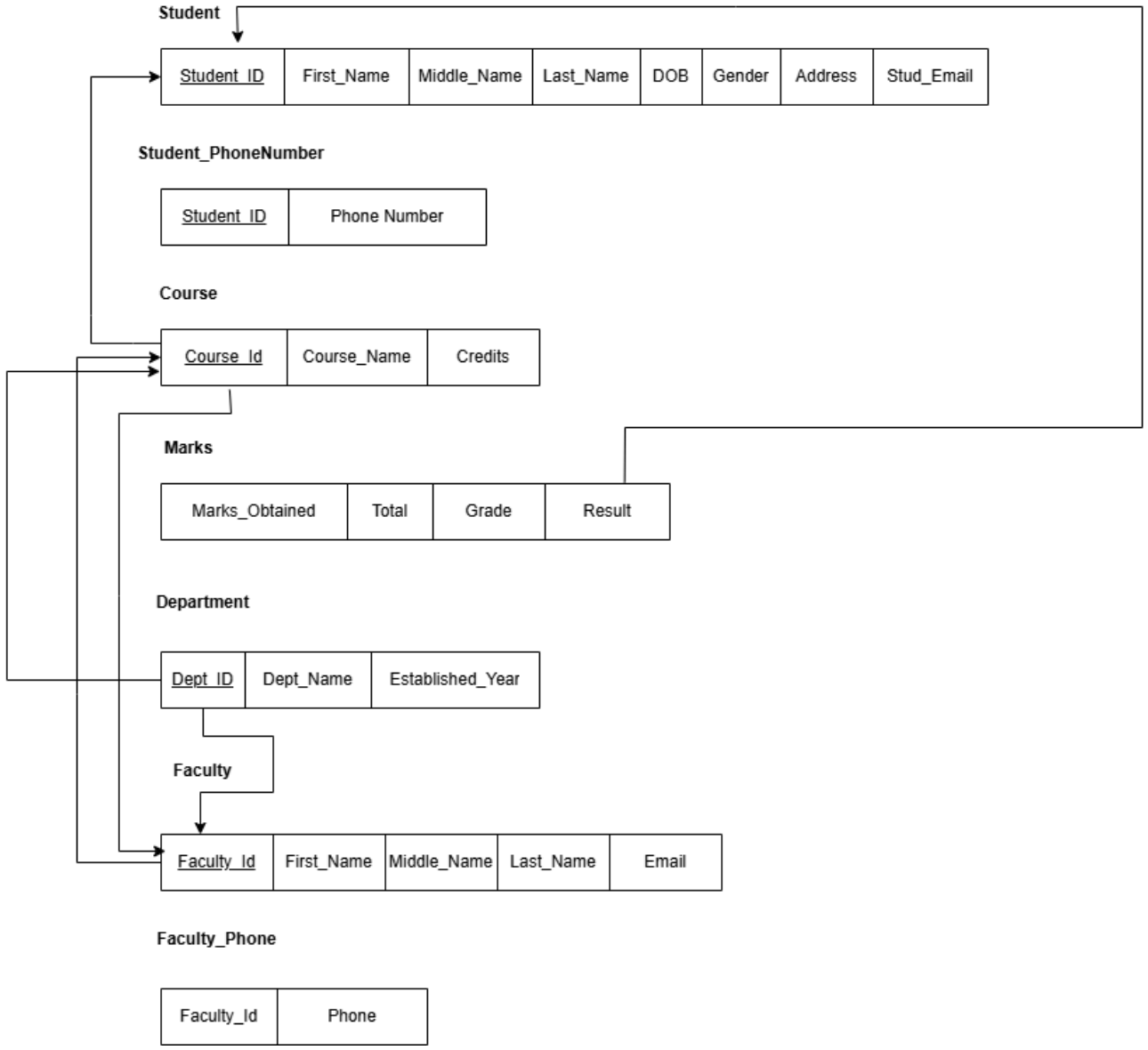
   c. Foreign keys should reference the primary key(s) of related tables.
   d. Ensure that foreign keys have appropriate constraints, such as ON DELETE CASCADE or ON UPDATE CASCADE, to maintain data integrity.

5. Cardinality Constraints:
   a. Use the cardinality constraints from the ER diagram to determine the multiplicity of relationships in the relational schema.
   b. Ensure that the constraints are enforced through the appropriate use of primary and foreign keys.

6. Normalization:
   a. Normalize the schema to minimize redundancy and dependency.
   b. Follow normalization rules such as First Normal Form (1NF), Second Normal Form (2NF), Third Normal Form (3NF), etc., to ensure data integrity and minimize anomalies.

7. Indexing and Optimization:
   a. Consider indexing frequently queried columns to improve query performance.
   b. Evaluate the schema design for optimization opportunities based on query patterns and performance requirements.

Implementation



Conclusion

a. Write definition of relational schema and notations

A **relational schema** is a blueprint or structure that defines how data is organized in a relational database. It outlines the tables, the attributes of each table, and the relationships between them.

The relational schema is expressed in terms of **relations** (tables), where each relation consists of a set of attributes (columns) and a set of tuples (rows).

Components of a Relational Schema:

1. **Relation (Table)**: A collection of related data in the form of rows and columns.
2. **Attributes (Columns)**: The properties or characteristics of the entities represented by the relation.
3. **Domain**: The set of possible values for each attribute.
4. **Primary Key**: An attribute (or a set of attributes) that uniquely identifies each tuple (row) in the relation.
5. **Foreign Key**: An attribute (or a set of attributes) that links two relations, enforcing referential integrity.

Relational Schema Notations

Relation (Table) Notation : A relation is represented as:
RelationName (Attribute1, Attribute2, ..., AttributeN)

Primary Key (PK) :Denoted by underlining the attribute or labeling it as **PK**.
Example: Employee (EmployeeID(PK), Name, Age, DepartmentID)

Foreign Key (FK) : Denoted by labeling the attribute as **FK** or referencing another relation's primary key.
Example: Employee (EmployeeID(PK), Name, Age, DepartmentID(FK))

b. Write various schema-based constraints

Schema-based constraints are rules that define the structure and integrity of data within a relational database.

1. Domain Constraints

Definition: Specifies the permissible values for an attribute in a relation (table). Each attribute in a relation has a domain, which defines the set of allowed values for that attribute.

2. Key Constraints

Definition: Specifies that an attribute or a combination of attributes (called a **primary key**) must uniquely identify a tuple (row) in a relation.

3. Entity Integrity Constraint

Definition: Ensures that every relation (table) has a primary key and that no primary key attribute can have a **NULL** value.

4. Referential Integrity Constraint

- **Definition**: Ensures that a foreign key value in one relation must either be **NULL** or match a primary key value in another relation. This ensures that relationships between tables are consistent.

5. Unique Constraints

Definition: Ensures that all values in a column or a set of columns are unique across all rows in the table, except for **NULL** values.

6. Not Null Constraint

Definition: Specifies that a particular column cannot have a **NULL** value. This ensures that the attribute must always have a valid value.