AY: 2024-25

Class:	**SE**	**Semester:**	**IV**
Course Code:	**CSL402**	**Course Name:**	**Database Management System Lab**

Name of Student:	**Shruti Gauchandra**
Roll No. :	**16**
Experiment No.:	**2**
Title of the Experiment:	**Mapping ER/EER to Relational schema model.**
Date of Performance:	**23/01/25**
Date of Submission:	**30/01/25**

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:

Experiment No 2

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.

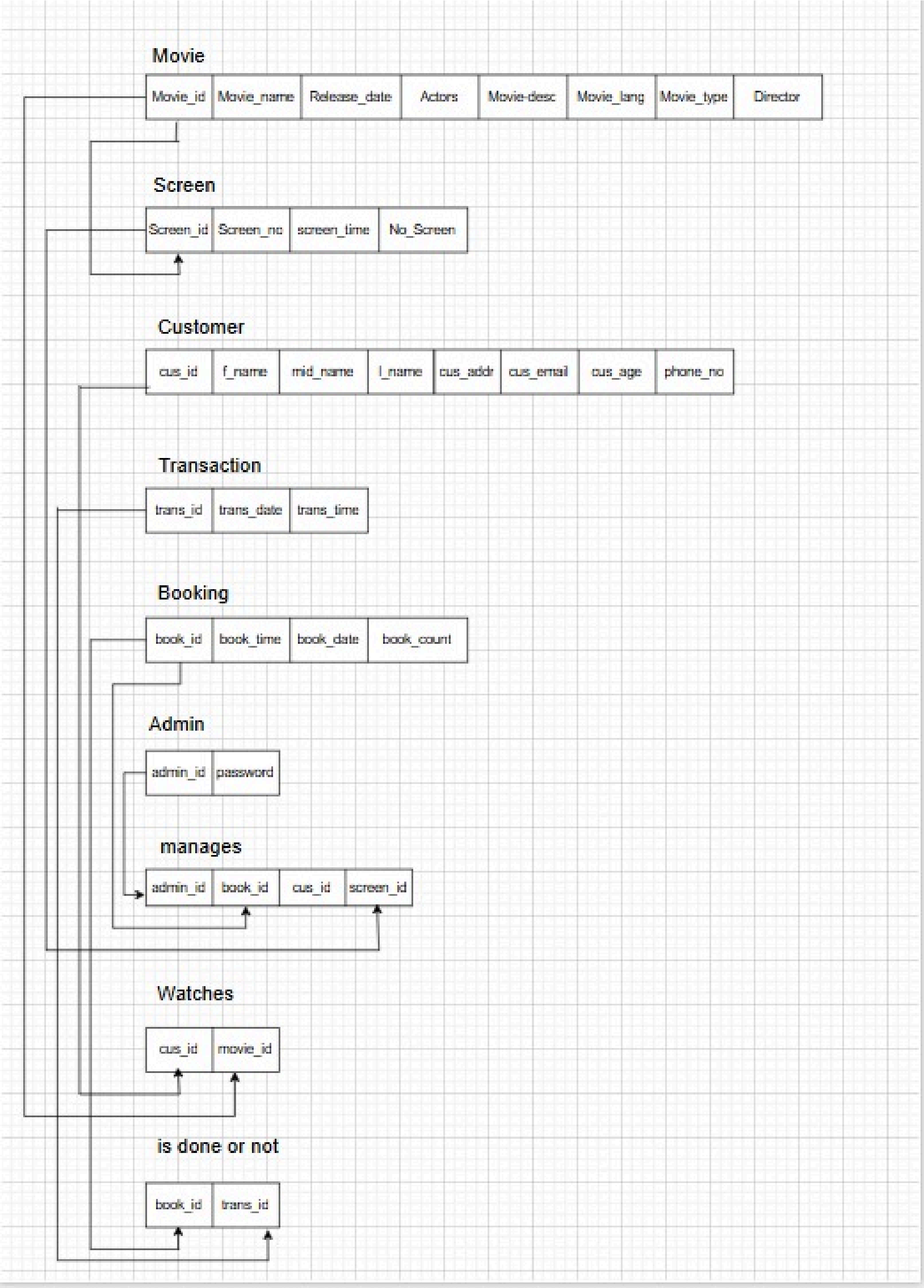
c. Foreign keys should reference the primary key(s) of related tables.
d. Ensure that foreign keys have appropriate constraints, such as ON DELETE CASCADE or ON UPDATE CASCADE, to maintain data integrity.

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:



Movie

Movie_id	Movie_name	Release_date	Actors	Movie-desc	Movie_lang	Movie_type	Director

Screen

Screen_id	Screen_no	screen_time	No_Screen

Customer

cus_id	f_name	mid_name	l_name	cus_addr	cus_email	cus_age	phone_no

Transaction

trans_id	trans_date	trans_time

Booking

book_id	book_time	book_date	book_count

Admin

admin_id	password

manages

admin_id	book_id	cus_id	screen_id

Watches

cus_id	movie_id

is done or not

book_id	trans_id

Conclusion

Mapping ER/EER models to relational schemas transforms conceptual designs into implementable database structures. Entities become tables, attributes become columns, and relationships are represented using foreign keys. This process ensures data integrity, reduces redundancy, and supports efficient database operations.

a. write definition of relational schema and notations.

Ans. **A relational schema defines the structure of a relation (table) in a relational database. It describes the relation's name and its set of attributes, each with an associated data type.**

Notation:

A relational schema is represented as:
R(A_1, A_2, ..., A□)

- **R → Name of the relation (table)**
- **A_1, A_2, ..., A□ → Attributes (columns) of the relation**

Example:
Student(RollNo, Name, Age, Dept)
This means Student is a table with attributes RollNo, Name, Age, and Dept.

b. write various schema-based constraints

Ans. **Schema-based constraints are rules defined at the schema level to ensure data integrity in a relational database. The main types include:**

1. **Domain Constraint**
   - **Specifies the allowed data type and values for an attribute.**
   - **Example: Age must be an integer between 0 and 100.**
2. **Key Constraint**

- **Ensures that a set of attributes (like a primary key) uniquely identifies each tuple (row).**
- **Example: RollNo in a Student table must be unique.**

3. **Entity Integrity Constraint**
   - **States that the primary key of a table cannot be null.**
   - **Ensures every record can be uniquely identified.**
4. **Referential Integrity Constraint**
   - **Ensures that a foreign key value in one table matches a primary key value in another table.**
   - **Example: DeptID in Student must exist in the Department table.**
5. **Tuple Uniqueness Constraint**
   - **Ensures that no two rows in a table are identical in all attribute values.**

These constraints help maintain accurate, consistent, and reliable data in the database.