Computer Engineering Department

King Mongkut's University of Technology Thonburi

CPE 241 Database

Problem Session #03 ER Diagram Exercise and Modify Existing Database

Date: Check PS03 assessment activities on LEB2

Introduction

In this third problem session, you will practice creating Entity-Relationship (ER) diagrams for the twelve relations you worked on in PS01 and PS02. This exercise aims to enhance your understanding of data structure and relationships while familiarizing you with various ways to represent them. After creating the diagrams, you will revisit the database you previously created to evaluate how well it adheres to the new constraints. Finally, you will modify the tables as needed to ensure they comply with these updated constraints.

Task List

1. **ER Diagram Exercise**
2. **Modify Modified Existing Database**

Task 1: ER Diagram Exercise

In this task, you will visually represent your previously designed schemas by creating Entity-Relationship (ER) diagrams. These diagrams will help you better understand the relationships between entities and prepare you for database modeling and design. You will create three types of diagrams: **Chen ERD, Crow's Foot ERD, and UML Class Diagram**. Each diagram will include **primary keys and foreign key constraints** to accurately reflect the structure of your database.

Objectives of Task 1

1. Visualize the structure and relationships of your database schema at conceptual, logical, and physical levels.
2. Represent primary keys (PK) and foreign keys (FK) in your database design.
3. Practice using Chen, Crow's Foot, and UML Class Diagram notations.

Instructions

STEP 1: Review Your Schema

- Begin by reviewing the database schema you designed in previous sessions (PS01 and PS02).
- Ensure you have identified the following for each table:
    - Primary Keys (PK): Attributes that uniquely identify each record in a table.
    - Foreign Keys (FK): Attributes that create relationships between tables by referencing primary keys in another table.

STEP 2: Create the ER Diagrams

You will create three diagrams based on your schema, ensuring primary and foreign key constraints are included:

1. **Chen ER Diagram (Conceptual/Logical Level)**
    - Use Chen notation to create a conceptual or logical-level diagram.
    - Represent: entities as rectangles, relationships as diamonds, and attributes as ovals.

- Highlight:
    - **Primary Keys (PK):** Underline attributes that serve as primary keys.
    - **Foreign Keys (FK):** Use notation or labels to indicate attributes that are foreign keys.
- Add relationship cardinalities (1:1, 1:N, N:M) based on your schema.

2. **Crow's Foot ER Diagram (Physical Level)**
    - Create a detailed physical-level diagram using Crow's Foot notation.
    - Represent:
        - Entities as rectangles with columns for attributes.
        - Relationships as connecting lines with cardinality symbols (| for "one," O for "optional," ∞ for "many").
    - Highlight:
        - **Primary Keys (PK):** Place them at the top of each entity and mark them clearly (e.g., bold or underline).
        - **Foreign Keys (FK):** Indicate attributes serving as foreign keys and show their relationship to the primary key in the referenced table.
3. **UML Class Diagram (Physical Level)**
    - Use UML Class Diagram notation to represent the physical structure of your database.
    - Represent: classes as boxes with three sections: class name, attributes, and methods (methods can be skipped for this task).
    - Highlight:
        - **Primary Keys (PK):** Clearly identify the primary key attribute(s) in each class.
        - **Foreign Keys (FK):** Indicate foreign key relationships using associations with multiplicity (e.g., 1..*, 0..1).

STEP 3: Create the DDL and tables in database

You are going to **pick three cases** from twelve cases that you draw ERDs for. Then write the SQL create statements with primary key and foreign key constraints to include in the report and use those statements to create tables in your database.

Deliverables

You are required to submit the following:

1. **Chen ER Diagram**
    - A conceptual or logical-level diagram showing all entities, attributes, relationships, and cardinalities.
2. **Crow's Foot ER Diagram**
    - A physical-level diagram illustrating detailed table structures, including primary and foreign key constraints.
3. **UML Class Diagram**
    - A physical-level diagram showing the class structure, attributes with data types, and relationships, including PK and FK constraints.

Submission Format

- Compile all diagrams into a single PDF report.
- Organize the report with clear sections:
    - Chen ER Diagram
    - Crow's Foot ER Diagram
    - UML Class Diagram
- Include captions for each diagram explaining PK and FK constraints.

Task 2: Modify Modified Existing Database

In this task, you will identify and resolve data anomalies in your existing tables (`students`, `courses`, and `grades`) by introducing primary key (PK) and foreign key (FK) constraints. You will also expand your database design by adding a new entity, department, and linking it to students and courses. Finally, you will conclude by reflecting on the lab.

Objectives of Task 2

1. Understand Data Anomalies - identify and resolve common data anomalies to maintain consistency.
2. Reinforce the Importance of Constraints - learn how primary and foreign key constraints ensure data integrity and prevent invalid operations.
3. Expand Database Design - introduce the `departments` entity and establish relationships with `students` and `courses` for a more comprehensive schema.

Instructions

STEP 1: Insert Duplicate and Conflicting Data

1. Insert the following **anomalous data** into your tables:

   A duplicate `studentID` but with a different name.
   ```
   INSERT INTO students (studentID, firstname, lastname) VALUES (101, 'Mark', 'Taylor');
   ```

   A duplicate `courseID` but with a different course name.
   ```
   INSERT INTO courses (courseID, courseName) VALUES (201, 'Advanced Databases');
   ```

   The same combination of `studentID` and `courseID` but with a different grade.
   ```
   INSERT INTO grades (studentID, courseID, grade) VALUES (101, 201, 'B');
   ```

2. Capture the Anomalies:

   Query your tables to display all the anomalies.
   ```
   SELECT * FROM students;
   SELECT * FROM courses;
   SELECT * FROM grades;
   ```
3. Include the anomalies in your report by showing the table data before fixing them.

STEP 2: Resolve the Anomalies

1. Delete the duplicate and conflicting rows:

```
DELETE FROM students WHERE studentID = 101 AND firstname =
'Mark';

DELETE FROM courses WHERE courseID = 201 AND courseName =
'Advanced Databases';

DELETE FROM grades WHERE studentID = 101 AND courseID = 201
AND grade = 'B';
```

2. Verify the tables to ensure the anomalies are resolved:

```
SELECT * FROM students;
SELECT * FROM courses;
SELECT * FROM grades;
```

STEP 3: Add Primary Key and Foreign Key Constraints

1. Add primary key (PK) constraints to each table:

```
ALTER TABLE students ADD CONSTRAINT pk_students PRIMARY KEY
(studentID);
ALTER TABLE courses ADD CONSTRAINT pk_courses PRIMARY KEY
(courseID);
ALTER TABLE grades ADD CONSTRAINT pk_grades PRIMARY KEY
(studentID, courseID);
```

2. Add foreign key (FK) constraints:

```
ALTER TABLE grades ADD CONSTRAINT fk_grades_students FOREIGN
KEY (studentID) REFERENCES students(studentID);
ALTER TABLE grades ADD CONSTRAINT fk_grades_courses FOREIGN
KEY (courseID) REFERENCES courses(courseID);
```

3. Replay the Insertion Attempts:
    - Attempt to reinsert the previous duplicate and conflicting data.
    - Capture the error messages displayed by DBMS in your report.

Step 4: Add the `departments` Entity and more tables

1. Create a new table, `departments`:
```
CREATE TABLE departments (
     departmentID INT(8) PRIMARY KEY,
     departmentName VARCHAR(50)
);
```

2. Modify the students and courses tables to include a departmentID column:
```
ALTER TABLE students ADD departmentID INT(8);
ALTER TABLE courses ADD departmentID INT(8);
```

3. Add foreign key (FK) constraints linking departments to students and courses:
```
ALTER TABLE students ADD CONSTRAINT fk_students_department
FOREIGN KEY (departmentID) REFERENCES
departments(departmentID);
ALTER TABLE courses ADD CONSTRAINT fk_courses_department
FOREIGN KEY (departmentID) REFERENCES
departments(departmentID);
```

4. Insert sample data into the departments table and link it to students and courses:
```
INSERT INTO departments (departmentID, departmentName) VALUES
(1, 'Computer Science');
INSERT INTO departments (departmentID, departmentName) VALUES
(2, 'Information Systems');

UPDATE students SET departmentID = 1 WHERE studentID IN (101,
102, 103);
UPDATE courses SET departmentID = 2 WHERE courseID IN (201,
202, 203);
```

5. Verify the updated tables:
```
SELECT * FROM department;
SELECT * FROM students;
SELECT * FROM courses;
```

6. Create selected tables from SQL statements from Task 1

Step 5: Conclude the Lab

Write a summary reflecting on your experience in this lab. Your conclusion should include:

- A brief explanation of the anomalies you encountered and how they were resolved.
- Observations about the importance of primary key and foreign key constraints in maintaining data integrity.
- Insights gained from expanding the database design to include the department entity.
- Any challenges you faced during the process.

Final Submission

- Combine results from Task 1 and Task 2 in a PDF file and ensure it is named appropriately (e.g., `PS03_YourGroupName.pdf`).
- Upload the PDF report to the Assessment Activities section for PS03 on LEB2.