OCTOBER 31, 2024

HOMEWORK 4 A

CS 457 B

UMAIR DADA
2O281 - SFBU

(4.19)

In order to address the GTL database case study, we should analyze and pinpoint the main ideas and connections. This is the way we can tackle each aspect of the design:

Important ideas in the Library Database

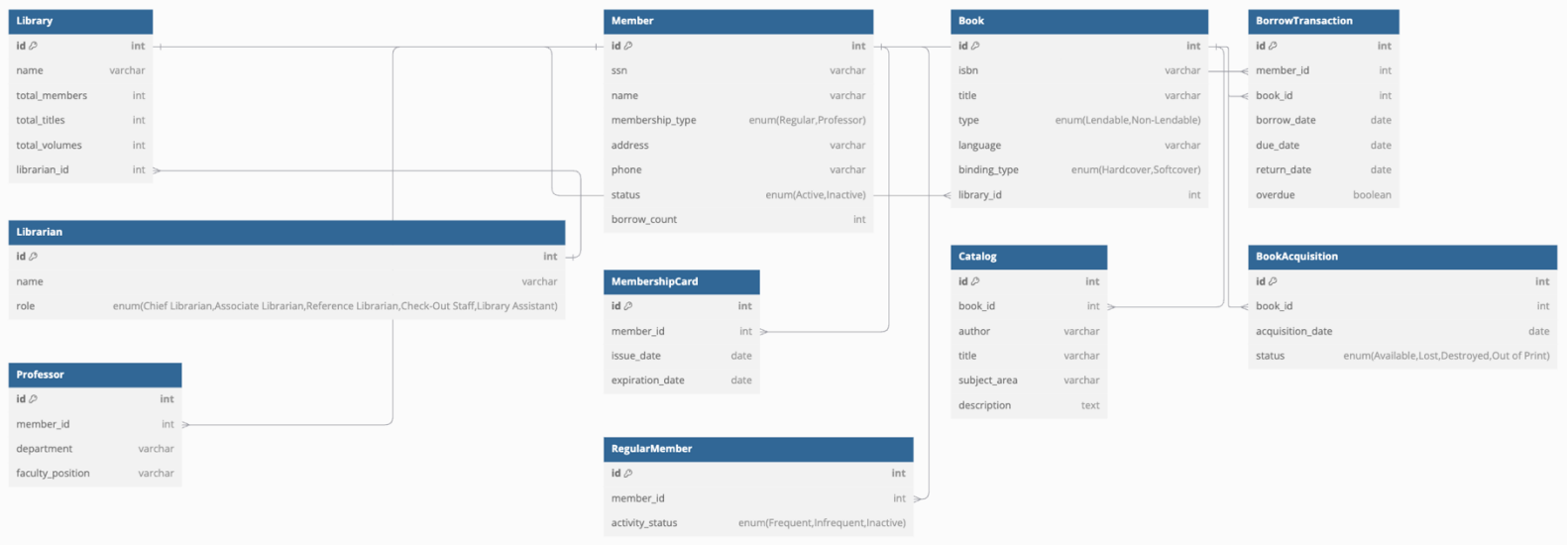
- Types of entities
    - Member: Serves as the representation of individuals who are part of the library.
    - Title of the book: Titles that are one-of-a-kind in the library, distinguished by ISBN.
    - Book Copy (Volume): Singular physical copies of every title.
    - Staff: Library employees, categorized by responsibilities.
    - Loan: Information regarding borrowing books.

- Types of Relationships
    - Membership: The connection between each member and the library.
    - Lending: Connection between Members and Book Copies, with limits on the number of books per member.
    - Cataloging involves connecting book titles with catalog entries.
    - Employment refers to the interaction between library staff members who have various job responsibilities.

- Combining data sources into one dataset.
    - The library contains books with various titles and multiple copies of each book.
    - Loan Activity: Combining loans by each member to monitor members with high levels of borrowing.

- Recognition
    - Book Copy: Distinguished by a copy ID and ISBN.
    - Member: Distinguished through SSN, also possesses a library card number.
    - Title of book: Distinguished through ISBN, not through title.

- Focus on a specific area or broaden your knowledge.
    - Member Specialization: Professors are a specific type of Member with additional privileges.
    - Specialization of book types: Divided into categories of books that can be borrowed and books that cannot be borrowed.

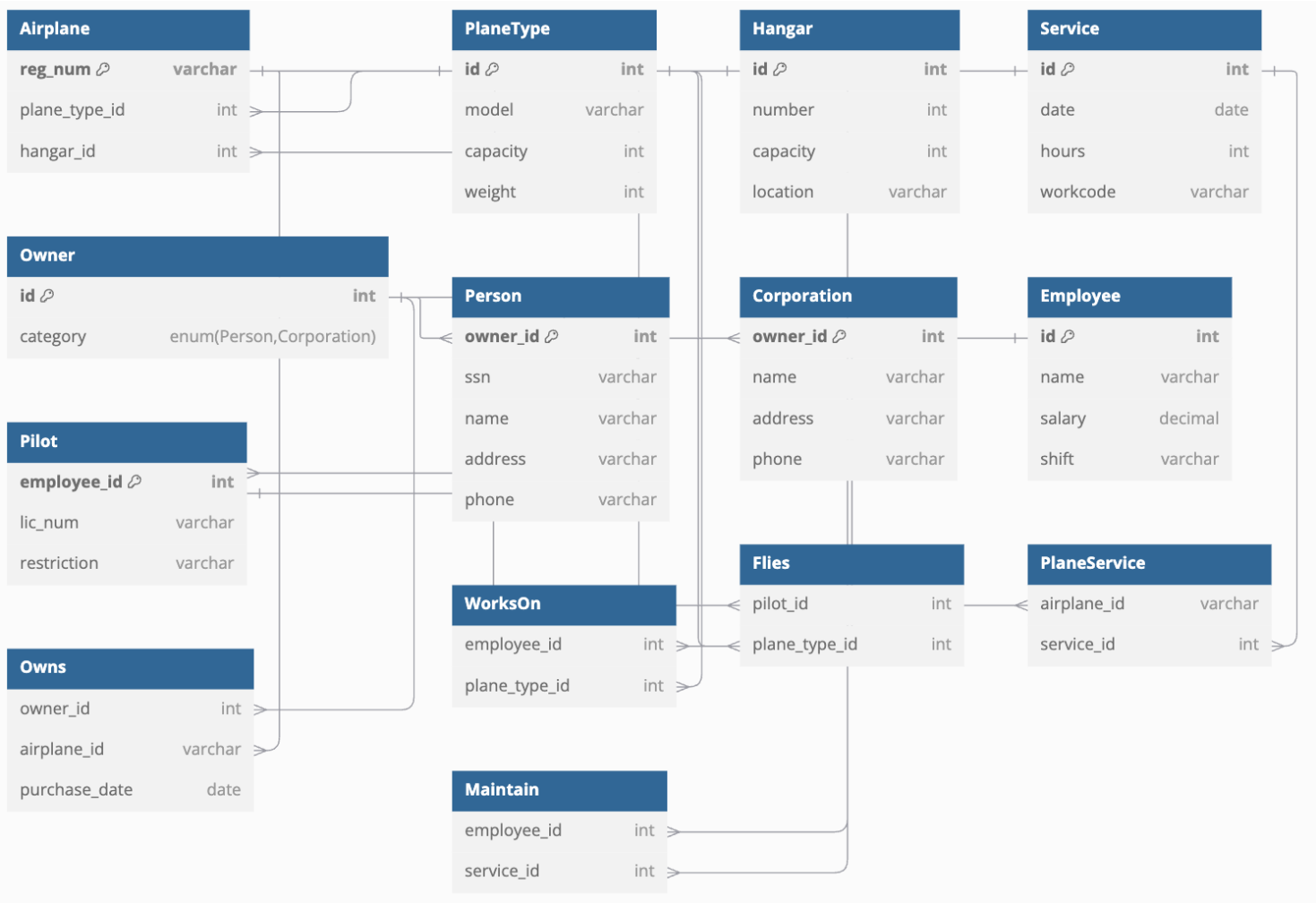


(4.21)

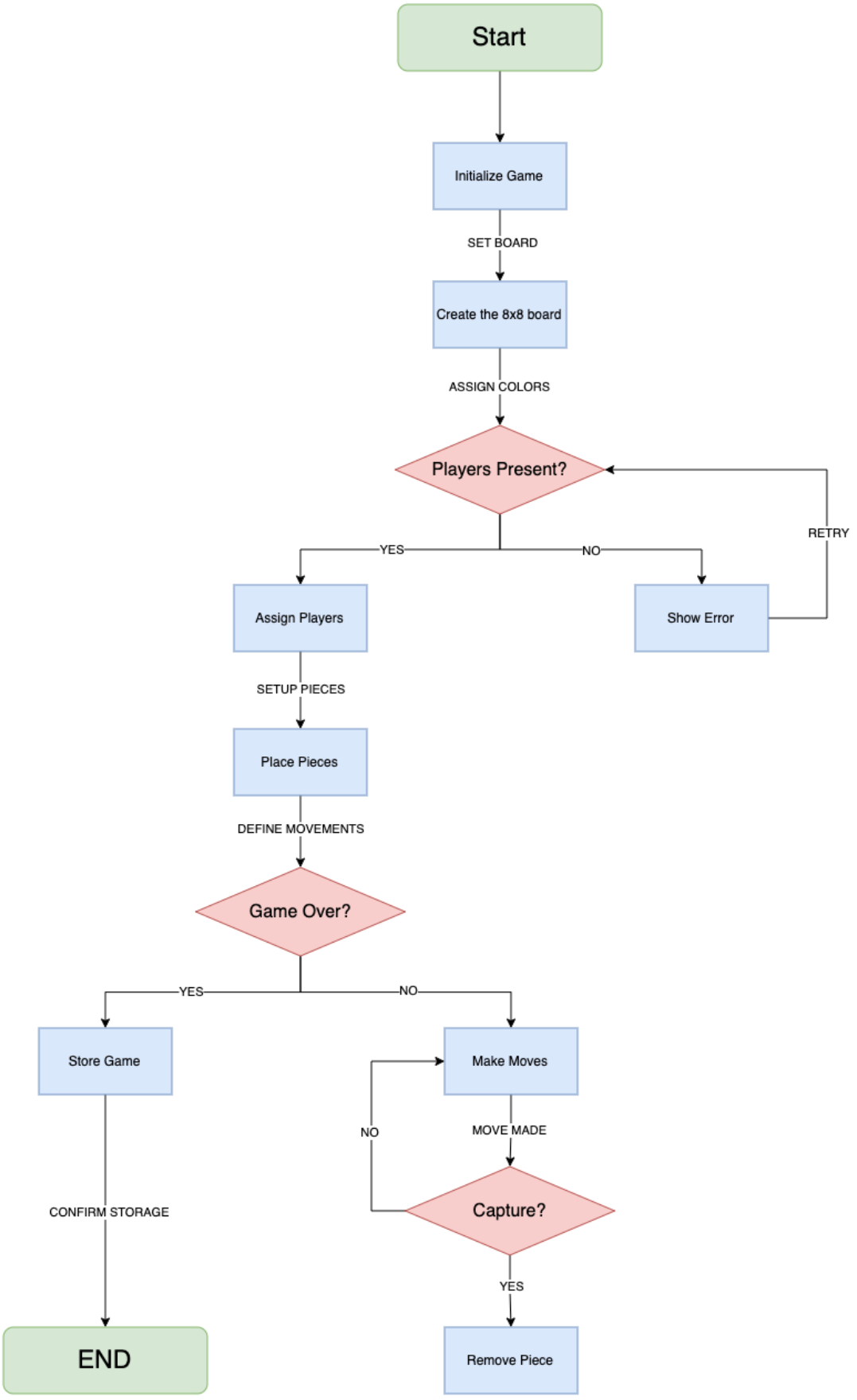
Key Elements

- **Entities**:
    - **Airplane**: Contains registration number (reg_num), associated with PlaneType and Hangar.
    - **PlaneType**: Attributes like model, capacity, and weight.
    - **Hangar**: Has unique number, capacity, and location.
    - **Service**: Contains details about maintenance service including date, hours, and work code.
    - **Owner**: Uses a category (union type) with subtypes **Person** and **Corporation**.
    - **Employee**: Has generic employee attributes.
    - **Pilot**: Specialized from **Employee**, with additional attributes for license number and restrictions.

- **Relationships**:
    - **Owns**: Connects Owner and Airplane with purchase date as an attribute.
    - **PlaneService**: Represents a many-to-many relationship between Airplane and Service.
    - **WorksOn**: Connects Employee to PlaneType, representing which types of planes they can maintain.
    - **Flies**: Connects Pilot to PlaneType, indicating which planes they are authorized to fly.
    - **Maintain**: Associates Employee with Service, showing maintenance work performed.



(4.24)


Start
Initialize Game
SET BOARD
Create the 8x8 board
ASSIGN COLORS
Players Present?
YES
NO
RETRY
Assign Players
Show Error
SETUP PIECES
Place Pieces
DEFINE MOVEMENTS
Game Over?
YES
NO
Store Game
Make Moves
NO
MOVE MADE
CONFIRM STORAGE
Capture?
YES
END
Remove Piece

To model the storage of a chess game in a database, we define the core entities and their relationships based on key assumptions about chess gameplay and interactions. Here is a structured UML outline:

Assumptions:

1. **Game Players**: Each game has two players, assigned either white or black.
2. **Piece Control**: Each player controls pieces that follow standard chess movement rules.
3. **Board Structure**: Pieces start on designated positions within an 8x8 grid.
4. **Move Dynamics**: Moves can involve capturing an opponent's piece or promoting pawns under specific conditions.

UML Structure:

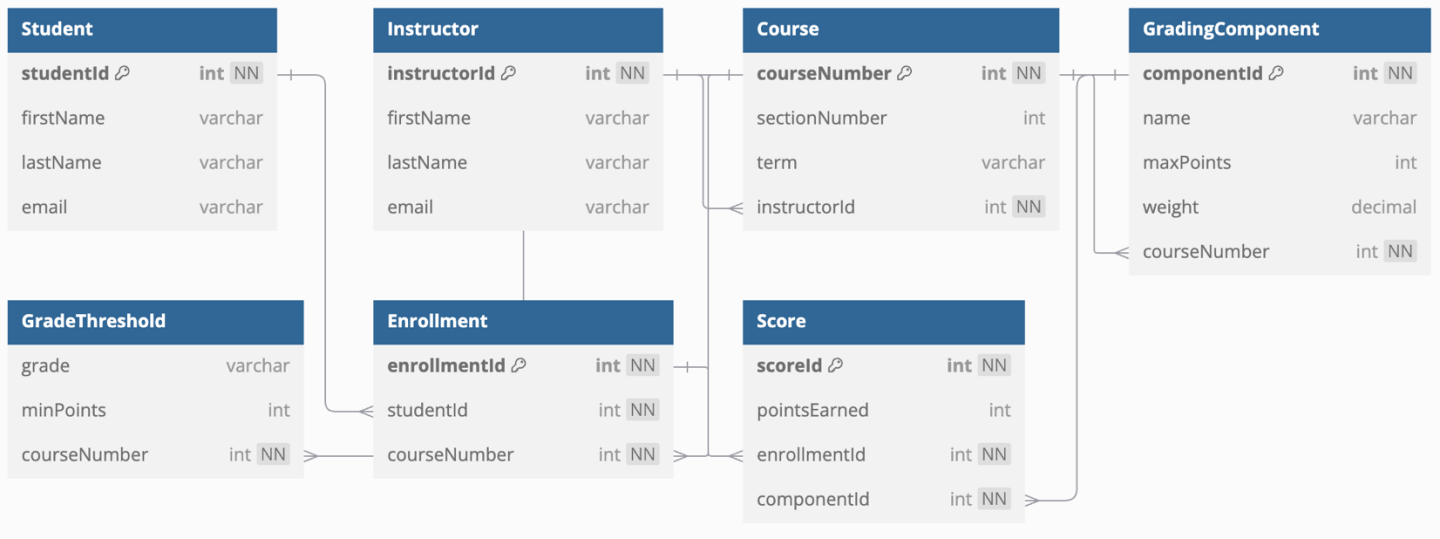
The design includes five main classes: **Game**, **Player**, **Piece**, **Board**, and **Move**, each representing an aspect of the chess game.

Classes and Relationships

1. **Game**
    - **Attributes**:
        - **gameId** (PK): Unique identifier for each game.
        - **datePlayed**: Date the game was played.
        - **result**: Outcome of the game (e.g., win, loss, draw).
    - **Relationships**:
        - *1.. association with Move**: A game consists of multiple moves.
        - **1..2 association with Player**: Each game involves exactly two players.
2. **Player**
    - **Attributes**:
        - **playerId** (PK): Unique identifier for each player.
        - **name**: Player's name.
        - **color**: The color assigned to the player (either black or white).
    - **Relationships**:
        - *1.. association with Piece**: Each player has multiple pieces.
        - *0.. association with Game**: A player may participate in multiple games over time.

3. **Piece**
    - **Attributes**:
        - **pieceId** (PK): Unique identifier for each piece.
        - **type**: Type of chess piece (e.g., King, Queen, Rook, Bishop, Knight, Pawn).
        - **color**: Color of the piece, aligned with the player's color.
        - **isCaptured** (boolean): Indicates whether the piece is currently captured.
    - **Relationships**:
        - *1.. association with Move**: Each piece may have multiple moves.
        - **1 association with Player**: Each piece belongs to one player.
4. **Board**
    - **Attributes**:
        - **boardId** (PK): Unique identifier for each board layout.
        - **layout**: Representation of the board's current state (e.g., an 8x8 matrix or array).
    - **Relationships**:
        - **1..1 composition with Game**: Each game has a unique board configuration.
5. **Move**
    - **Attributes**:
        - **moveId** (PK): Unique identifier for each move.
        - **startPosition**: Starting position of the piece on the board (e.g., "E2").
        - **endPosition**: Ending position of the piece on the board (e.g., "E4").
        - **timestamp**: Time when the move was made.
        - **isCapture** (boolean): Indicates if the move involved capturing an opponent's piece.
        - **isPromotion** (boolean): Indicates if the move resulted in a pawn promotion.
    - **Relationships**:
        - **1 association with Piece**: Each move involves one specific piece.
        - **1 association with Game**: Each move belongs to a particular game

(4.27)



To model a university course grading system, we define key entities and relationships that capture student enrollment, course structure, grading components, and score tracking.

Key Entities and Attributes:

Student

- Attributes:
  - studentId (Primary Key): Unique identifier for each student.
  - firstName: First name of the student.
  - lastName: Last name of the student.
  - email: Email address of the student.

Instructor

- Attributes:
  - instructorId (Primary Key): Unique identifier for each instructor.
  - firstName: First name of the instructor.
  - lastName: Last name of the instructor.
  - email: Email address of the instructor.

Course

- Attributes:
    - courseNumber: Unique identifier for each course.
    - sectionNumber: Section number for the course.
    - term: Term in which the course is offered (e.g., Fall 2024).
- Relationships:
    - Teaches: An instructor teaches one or more courses each term.
    - Has Grading Components: Each course has multiple grading components.

GradingComponent

- Attributes:
    - componentId (Primary Key): Unique identifier for each grading component.
    - name: Name of the grading component (e.g., Midterm, Final Exam).
    - maxPoints: Maximum points possible for this component.
    - weight: Weight of the grading component in percentage (e.g., 20%).
- Relationships:
    - Belongs to Course: Each grading component is associated with a specific course.

GradeThreshold

- Attributes:
    - grade: Grade letter (A, B, C, D, F).
    - minPoints: Minimum points required to earn this grade.
- Relationships:
    - Set by Course: Each course has multiple GradeThreshold entries that define the point thresholds for letter grades.

Enrollment

- Attributes:
    - enrollmentId (Primary Key): Unique identifier for each enrollment record.
- Relationships:
    - Student Enrolls in Course: Links a student to a course for enrollment.
    - Tracks Scores: Links the points earned by each student in each grading component of a course.

Score

- Attributes:
  - scoreId (Primary Key): Unique identifier for each score entry.
  - pointsEarned: Points a student earned in a particular grading component.
- Relationships:
  - Part of Enrollment: Each score is associated with an enrollment record.
  - Linked to GradingComponent: Each score is linked to a specific grading component.

EER Diagram Structure

- Entities: Student, Instructor, Course, GradingComponent, GradeThreshold, Enrollment, Score.
- Relationships:
  - Teaches: Between Instructor and Course.
  - Enrolls: Between Student and Course, forming Enrollment.
  - Has Grading Components: Between Course and GradingComponent.
  - Set by Course: Between Course and GradeThreshold.
  - Scores: Between Enrollment and GradingComponent for each score entry.