INFO6001 2024 T1

Assignment 3

Project: Database design of SCS Resource Management

*Physical Database Design*

*Final Report*

Table of Contents

[Preface 4](#_Toc164195515)

[Part 1.1: Reflection on Assignment 2 5](#_Toc164195516)

[Requirements 5](#_Toc164195517)

[Data Requirements 5](#_Toc164195518)

[Transaction Requirements 6](#_Toc164195519)

[INSERT QUERY: 6](#_Toc164195520)

[UPDATE QUERY: 6](#_Toc164195521)

[DELETE QUERY: 6](#_Toc164195522)

[SOME OTHER QUERIES: 6](#_Toc164195523)

[Query 1: 6](#_Toc164195524)

[Query 2: 7](#_Toc164195525)

[Query 3: 7](#_Toc164195526)

[Query 4: 7](#_Toc164195527)

[Query 5: 7](#_Toc164195528)

[Business Rules 7](#_Toc164195529)

[Part 1.2: EER Model 9](#_Toc164195530)

[EER Model 9](#_Toc164195531)

[Data Dictionary 10](#_Toc164195532)

[Entity 10](#_Toc164195533)

[Relationships 11](#_Toc164195534)

[Attributes: 13](#_Toc164195535)

[Mapping the EER to Relational Model 17](#_Toc164195536)

[Part 1.3: Normalizing the Scheme up to BCNF 22](#_Toc164195537)

[Step 1: Identify Dependencies 22](#_Toc164195538)

[Acquisition Table: 22](#_Toc164195539)

[Member Table: 22](#_Toc164195540)

[Staff Table: 22](#_Toc164195541)

[Student Table: 22](#_Toc164195542)

[Loan Table: 22](#_Toc164195543)

[Reservation Table: 22](#_Toc164195544)

[CourseOffering Table: 22](#_Toc164195545)

[Privilege Table: 23](#_Toc164195546)

[Category Table: 23](#_Toc164195547)

[Resource Table: 23](#_Toc164195548)

[Movable Table: 23](#_Toc164195549)

[Immovable Table: 23](#_Toc164195550)

[Part 2.1: Creating Physical Design of the table as the SQL Code 23](#_Toc164195551)

[Part 2.2: Input values in every table SQL Code. 29](#_Toc164195552)

[Part 2.3: Implementing the given queries SQL Code. 34](#_Toc164195553)

[Q1: Print the name of student(s) who has/have enrolled in the course with 34](#_Toc164195554)

[course id xxx. 34](#_Toc164195555)

[Q2: Print the maximal number of speakers that the student with name xxx can borrow. The student is enrolled in the course with course id yyy. Note: speaker is a category. 34](#_Toc164195556)

[Q3: For a staff member with id number xxx, print his/her name and phone number, the total number of reservations that the staff had made in 2022. 35](#_Toc164195557)

[Q4: Print the name(s) of the student member(s) who has/have borrowed the category with the name of camera, of which the model is xxx, in this year. Note: camera is a category, model attribute must be in movable table, and “this year” must be decided by the system. 35](#_Toc164195558)

[Q5: Find the moveable resource that is the mostly loaned in current year. Print the resource id and resource name. Note: “current year” must be decided by the system. 36](#_Toc164195559)

[Q6: For each of the three days, including May 1, 2024, June 5, 2024 and September 19, 2024, print the date, the name of the room, and the total number of reservations made for the room xxx on each day. 36](#_Toc164195560)

# Preface

This is the requirements analysis, conceptual design and logical database design for this assignment, we embark on the next phase of our journey towards constructing a comprehensive database solution for SCS Resource Management. Building upon the groundwork laid in previous assignments we delve deeper into refining the conceptual and logical designs, thus ensuring their coherence with the organizations operational requirements and business rules. Meticulously revisiting the EER model and mapping it to a normalized relational schema and we aim to optimize database structure while minimizing redundancy and enhancing data integrity.

Furthermore, assignment serves as a bridge between theoretical concepts and practical implementation as we transition from design to physical database creation. Through the meticulous crafting of SQL scripts, we instantiate the normalized database schema and incorporate essential elements such as primary keys, foreign keys, and constraints. Additionally, we fulfill transactional needs by executing SQL statements to insert data and perform queries, and ensure databases functionality aligns seamlessly with organizational requirements.

This stage of the database development process represents a significant milestone in our endeavor to construct a robust and efficient database system tailored to the unique needs of SCS Resource Management. Integrating theoretical principles with practical implementation strategies. Thus, we aim to deliver a database solution poised to streamline resource management, optimize operational efficiency and support the organizations overarching objectives.

Task 1

# Part 1.1: Reflection on Assignment 2

In Assignment 2, we embarked on a journey to refine and expand upon the conceptual and logical foundations established in Assignment 1. Our focus shifted towards the development of an Extended Entity Relationship (EER) model, which served as a comprehensive blueprint for the SCS Resource Management database. Through meticulous analysis of data requirements, transactional needs, and business rules, we crafted an EER model that encapsulated the complexities of resource management within the organization. This involved delineating entities, attributes, and relationships, while also considering specialized concepts such as superclass and subclass relationships. Additionally, we tackled the challenge of mapping the intricate EER model to a normalized relational schema, ensuring data integrity and minimizing redundancy. By scrutinizing each facet of the database design process, we aimed to create a robust foundation upon which the physical database implementation could flourish.

# Requirements

## Data Requirements

* **Loan Service**

Loan

Loan describes loans that members have made. It is created when a loan is made by a member.

Each member can loan movable resources, the number of resources the member can loan is dependent on the privileges. As staff does not have privilege, the amount of resource they can loan are not restricted. Information stored include (the resource loaned, the member lending it), date and time loaned, due date and time and date and time returned. All loans have a unique loan id. The due date will depend on the duration allowed by the category.

Reservation

Reservations are the reservations that members make. In the reservation, members reserve it for themselves and the resources. Reservations will be categorized by category. Reservations has a unique reservation id. That will be used to keep the data unique in all the rows. However, the time and the date of the reservation also store inside the database.

Course Offering

Course will be offered to the students. The student can enroll in many offered courses. Course has some privilege. The course has course id. That keeps the data unique in the database. The course contains some name. Course offered also take information about the semester in which student is enrolling and the year. It also keep the track of the start date of the semester and the end date of the semester or course.

## Transaction Requirements

Data Manipulation

### INSERT QUERY:

INSERT INTO Acquisition (resourceName, description, make, model, year, urgency, status, fundCode, vendorCode, price, notes, member\_id)

VALUES ('Printer', 'High-speed laser printer', 'HP', 'LaserJet Pro', 2022, 'High', 'Available', 'FC123', 'VC456', 599.99, 'Office use', 1);

### UPDATE QUERY:

UPDATE Acquisition

SET status = 'In Progress'

WHERE acqID = 1;

### DELETE QUERY:

DELETE FROM Member

WHERE member\_id = 2;

## SOME OTHER QUERIES:

### Query 1:

Select \* from loan;

### Query 2:

SELECT \*

FROM Member

INNER JOIN Acquisition ON Member.member\_id = Acquisition.member\_id;

### Query 3:

SELECT Loan.\*, Member.name AS member\_name, Resource.description AS resource\_description

FROM Loan

INNER JOIN Member ON Loan.member\_id = Member.member\_id

INNER JOIN Movable ON Loan.loan\_id = Movable.loan\_id

INNER JOIN Resource ON Movable.resource\_id = Resource.resource\_id;

### Query 4:

SELECT \*

FROM Acquisition

WHERE status = 'In Progress'

ORDER BY year DESC;

### Query 5:

SELECT Loan.\*, Member.name AS member\_name, Resource.description AS resource\_description

FROM Loan

INNER JOIN Member ON Loan.member\_id = Member.member\_id

INNER JOIN Movable ON Loan.loan\_id = Movable.loan\_id

INNER JOIN Resource ON Movable.resource\_id = Resource.resource\_id

WHERE Loan.datetime\_borrowed BETWEEN '2024-01-01' AND '2024-03-31'

ORDER BY Loan.datetime\_borrowed DESC;

## Business Rules

* Student member set to ‘Disabled’ if the current date is later than end date of his course offering.
* Each acquisition should have a unique combination of resource name and member.
* The status of an acquisition should be one of a predefined set of values.
* The price of an acquisition should be non-negative.
* The year of acquisition should be within a valid range

# Part 1.2: EER Model

## EER Model



Data Dictionary

## Entity

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Loan | describing loans that members have made | Resource borrowed | When a loan is made by a member |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Reservations | describing reservations that members have made | Resource borrowed | When a reservation is made by a member |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Member | Members are important for acquisition | The person made reservations and loans | Members are the entities that will relate with each other to make it meaningful. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Course Offering | describing enroll courses that members have made | Enroll by students | Occur when students enroll into the courses. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Privilege | describing privilege of the items | Resource borrowed | When a student enroll into the course. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Resources | describing resources that have been reserved by members | Resource borrowed | When a resource is reserved by a member |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Category | Category is the thing that save about the privilege and the for the time resource is reserved | Resource borrowed | Resource category reserve |

## Relationships

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Loan | 0..\* | is of | 1..1 | Member |
| 0..\* | to | 1..1 | Movable |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Member | 0..\* | is of | 1..1 | loan |
| 0..\* | to | 1..1 | reservations |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Reservations | 0..\* | is of | 1..1 | Member |
| 0..\* | to | 1..1 | Resource |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Resource | 0..\* | is of | 1..1 | Reservations |
| 0..\* | to | 1..1 | category |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Category | 0..\* | is of | 1..1 | Resource |
| 0..\* | Is of | 1..1 | Privilige |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Privilege | 0..\* | is of | 1..1 | Course Offering |
| 0..\* | to | 1..1 | Category |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Course Offering | 1..\* | is of | 0..\* | Student |
| 0..\* | to | 0..\* | privilege |

## Attributes:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Loan | LoanID |  |  |  |  |  |  |
|  | dateOfLoan | The date the resource is loaned out | date | FALSE | FALSE | FALSE | current date |
|  | dateOfDue | The date the resource is due for return | date | FALSE | FALSE | FALSE |  |
|  | dateOfReturn | The date the resource is returned | date | TRUE | FALSE | TRUE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Resource | Resource ID |  |  |  |  |  |  |
|  | description | Description of the source | Varchar(100) | FALSE | FALSE | FALSE | current date |
|  | status | Status of the resource | Varchar(100) | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Immovable | Resource ID |  | int |  |  |  |  |
|  | Capacity | Capacity of the room | Int | FALSE | FALSE | FALSE |  |
|  | Room | Room no | VARCHAR(50) | FALSE | FALSE | FALSE |  |
|  | Building | Building | Varchar | TRUE | FALSE | TRUE |  |
| … | Campus | And the campus in which these resources are | Varchar |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| MOvable | Resource ID |  | int |  |  |  |  |
|  | manufacturer | The date the resource is loaned out | date | FALSE | FALSE | FALSE |  |
|  | model | Model of the resources | Varchar(100) | FALSE | FALSE | FALSE |  |
|  | year | Year it is built | date | TRUE | FALSE | TRUE |  |
|  | AssetValue |  | Varchar(50) |  |  |  |  |
|  | Building ID |  | Varchar(50) |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Category | Code |  | int |  |  |  |  |
|  | name | Name of the category | Varchar(100) | FALSE | FALSE | FALSE |  |
|  | description | Description of the category | Varchar(100) | FALSE | FALSE | FALSE |  |
|  | duration days | For days they have reserved | int | TRUE | FALSE | TRUE |  |
|  | duration Hours | For hours they have reserved | time |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Privilege | Privi ID | Primary key | INt |  |  |  |  |
|  | name | Name of the privilege | Varchar(100) | FALSE | FALSE | FALSE | current date |
|  | description | Description of the privilege | Varchar(100) | FALSE | FALSE | FALSE |  |
|  | Max Items | Max item of the privilege | int | TRUE | FALSE | TRUE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Course Offering | Offer ID | Primary key | int |  |  |  |  |
|  | cid | Course id | Int | FALSE | FALSE | FALSE | current date |
|  | course | Course name | Varchar (100) | FALSE | FALSE | FALSE |  |
|  | semester | Semester in which it is offered | Int | TRUE | FALSE | TRUE |  |
| … | year | Year in which it is offered | date |  |  |  |  |
|  | dateBegin | The day it is begin | date |  |  |  |  |
|  | dateEnd | The day it is end | date |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Reservations | Reservation ID |  | int |  |  |  |  |
|  | Date time reserved | The time made reservations | date | FALSE | FALSE | FALSE | current date |
|  | Date time due | Reservations till | date | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Member | Member id | Primary key | int |  |  |  |  |
|  | name | The date the resource is loaned out | Varchar(100) | FALSE | FALSE | FALSE | current date |
|  | address | The date the resource is due for return | Varchar(100) | FALSE | FALSE | FALSE |  |
|  | phone | The date the resource is returned | date | TRUE | FALSE | TRUE |  |
|  | email | Email of the memeber | Varchar(100) |  |  |  |  |
|  | status | Status of that person | Varchar(50) |  |  |  |  |
|  | comment |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Acquisition | ACq ID | Primary key | int |  |  |  |  |
|  | Resource name | Name of the resource | Varchar(100) | FALSE | FALSE | FALSE | current date |
|  | description | Description of the acquisition | Varchar(100) | FALSE | FALSE | FALSE |  |
|  | make | Make of the aquisition | Varchar(50) | TRUE | FALSE | TRUE |  |
| … | model | Acquisition model | Varchar(50) |  |  |  |  |
|  | year | Year it is made | int |  |  |  |  |
|  | Urgency |  | Varchar(100) |  |  |  |  |
|  | Status | Status of the acuisition | Varchar(100) |  |  |  |  |
|  | fundCode | The code at which it is fund | Int |  |  |  |  |
|  | VendorCode |  | Int |  |  |  |  |
|  | price | The price of the item | Int |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| STAFF | Member ID |  |  |  |  |  |  |
|  | Staff ID the staff id | The staff contain ID | int | FALSE | FALSE | FALSE | current date |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| Student | Student ID |  |  |  |  |  |  |
|  | Point earned | The point student get | intnt | FALSE | FALSE | FALSE | current date |

# Mapping the EER to Relational Model

Using the mapping rules, got the following relations for all entities in EER.

-- Create database

CREATE DATABASE IF NOT EXISTS ResourceManagement;

-- Use the database

USE ResourceManagement;

-- Create tables

CREATE TABLE Acquisition (

acqID INT PRIMARY KEY AUTO\_INCREMENT,

resourceName VARCHAR(255),

description TEXT,

make VARCHAR(255),

model VARCHAR(255),

year INT,

urgency VARCHAR(50),

status VARCHAR(50),

fundCode VARCHAR(50),

vendorCode VARCHAR(50),

price DECIMAL(10,2),

notes TEXT,

member\_id INT,

FOREIGN KEY (member\_id) REFERENCES Member(member\_id)

);

CREATE TABLE Member (

member\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(255),

address TEXT,

phone\_no VARCHAR(20),

email VARCHAR(100),

status VARCHAR(50),

comments TEXT

);

CREATE TABLE Staff (

member\_id INT,

staff\_id INT PRIMARY KEY AUTO\_INCREMENT,

FOREIGN KEY (member\_id) REFERENCES Member(member\_id)

);

CREATE TABLE Student (

student\_id INT PRIMARY KEY AUTO\_INCREMENT,

points\_earned INT

);

CREATE TABLE Loan (

loan\_id INT PRIMARY KEY AUTO\_INCREMENT,

datetime\_borrowed DATETIME,

datetime\_returned DATETIME,

datetime\_due DATETIME,

member\_id INT,

FOREIGN KEY (member\_id) REFERENCES Member(member\_id)

);

CREATE TABLE Reservation (

reservation\_id INT PRIMARY KEY AUTO\_INCREMENT,

datetime\_reserved DATETIME,

datetime\_due DATETIME

);

CREATE TABLE CourseOffering (

offer\_id INT PRIMARY KEY AUTO\_INCREMENT,

cid INT,

course VARCHAR(255),

semester VARCHAR(50),

year INT,

begin\_date DATE,

end\_date DATE

);

CREATE TABLE Privilege (

privil\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(255),

description TEXT,

maxItems INT

);

CREATE TABLE Category (

code VARCHAR(50) PRIMARY KEY,

name VARCHAR(255),

description TEXT,

duration\_days INT,

duration\_hours INT,

privi\_id INT,

FOREIGN KEY (privi\_id) REFERENCES Privilege(privil\_id)

);

CREATE TABLE Resource (

resource\_id INT PRIMARY KEY AUTO\_INCREMENT,

description TEXT,

status VARCHAR(50)

);

CREATE TABLE Movable (

name VARCHAR(255),

manufacturer VARCHAR(255),

model VARCHAR(255),

year INT,

asset\_value DECIMAL(10,2),

building\_ID INT,

loan\_id INT,

resource\_id INT,

FOREIGN KEY (loan\_id) REFERENCES Loan(loan\_id),

FOREIGN KEY (resource\_id) REFERENCES Resource(resource\_id)

);

CREATE TABLE Immovable (

capacity INT,

room VARCHAR(50),

building VARCHAR(255),

campus VARCHAR(255),

member\_id INT,

FOREIGN KEY (member\_id) REFERENCES Member(member\_id)

);

# Part 1.3: Normalizing the Scheme up to BCNF

According to the definitions of 1NF, 2NF, 3NF and BCNF, it is identified that relations x1, x2, ..xn are all in BCNF, since all the attributes are atomic, and there exists only one function dependency in each table, and the left side of the FD is a PK.

### Step 1: Identify Dependencies

#### **Acquisition Table:**

* resourceName, description, make, model, year, urgency, status, fundCode, vendorCode, price, notes depend on acqID.
* member\_id depends on acqID.

### Acquisition Table:

* acqID determines resourceName, description, make, model, year, urgency, status, fundCode, vendorCode, price, notes, member\_id.
* member\_id depends on the member\_id in the Member table.

### Member Table:

* member\_id determines name, address, phone\_no, email, status, comments.

### Staff Table:

* member\_id depends on the member\_id in the Member table.

### Student Table:

* No dependencies other than the primary key.

### Loan Table:

* loan\_id determines datetime\_borrowed, datetime\_returned, datetime\_due, member\_id.
* member\_id depends on the member\_id in the Member table.

### Reservation Table:

* No dependencies other than the primary key.

### CourseOffering Table:

* No dependencies other than the primary key.

### Privilege Table:

* No dependencies other than the primary key.

### Category Table:

* No dependencies other than the primary key.

### Resource Table:

* No dependencies other than the primary key.

### Movable Table:

* loan\_id depends on the loan\_id in the Loan table.
* resource\_id depends on the resource\_id in the Resource table.

### Immovable Table:

* member\_id depends on the member\_id in the Member table.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Task 2

Part 2.1: Creating Physical Design of the table as the SQL Code**.**

*SQL Code:*

CREATE DATABASE IF NOT EXISTS ResourceManagement;

USE ResourceManagement;

CREATE TABLE Member (

memberID INT PRIMARY KEY,

name VARCHAR(255),

address VARCHAR(255),

phone VARCHAR(20),

email VARCHAR(255),

status VARCHAR(50),

comments TEXT

);

-- Create the Acquisition table

CREATE TABLE Acquisition (

acqID INT PRIMARY KEY,

resourceName VARCHAR(255),

description TEXT,

make VARCHAR(100),

model VARCHAR(100),

year INT,

urgency VARCHAR(50),

status VARCHAR(50),

fundCode VARCHAR(50),

VendorCode VARCHAR(50),

price DECIMAL(10, 2),

notes TEXT,

memberID INT,

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON DELETE CASCADE

);

CREATE TABLE Staff (

staffID INT PRIMARY KEY,

memberID INT,

-- Add other attributes for Staff here

FOREIGN KEY (memberID ) REFERENCES Member(memberID) ON DELETE CASCADE

);

-- Create the Student table

CREATE TABLE Student (

studentID INT PRIMARY KEY,

memberID INT,

pointsEarned INT,

-- Add other attributes for Student here

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON DELETE CASCADE

);

-- Create the Resource table

CREATE TABLE Resource (

resourceID INT PRIMARY KEY,

description TEXT,

status VARCHAR(50),

-- Add other attributes for Resource here

);

-- Create the Movable table

CREATE TABLE Movable (

resourceID INT PRIMARY KEY,

name VARCHAR(255),

manufacturer VARCHAR(255),

model VARCHAR(255),

year INT,

assetValue DECIMAL(10, 2),

BuildingID INT,

-- Add other attributes for Movable here

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON DELETE CASCADE

);

-- Create the Immovable table

CREATE TABLE Immovable (

resourceID INT PRIMARY KEY,

Capacity INT,

Room VARCHAR(255),

Building VARCHAR(255),

Campus VARCHAR(255),

-- Add other attributes for Immovable here

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON DELETE CASCADE

);

-- Create the Loan table

CREATE TABLE Loan (

loanID INT PRIMARY KEY,

memberID INT,

resourceID INT,

dateTimeBorrowed DATETIME,

dateTimeReturned DATETIME,

dateTimeDue DATETIME,

-- Add other attributes for Loan here

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON DELETE CASCADE,

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON DELETE CASCADE

);

-- Create the Reservation table

CREATE TABLE Reservation (

reservationID INT PRIMARY KEY,

memberID INT,

resourceID INT,

dateTimeReserved DATETIME,

dateTimeDue DATETIME,

-- Add other attributes for Reservation here

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON DELETE CASCADE,

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON DELETE CASCADE

);

-- Create the CourseOffering table

CREATE TABLE CourseOffering (

offerID INT PRIMARY KEY,

studentID INT,

cid INT,

course VARCHAR(255),

semester VARCHAR(50),

year INT,

dateBegin DATE,

dateEnd DATE,

-- Add other attributes for CourseOffering here

FOREIGN KEY (studentID) REFERENCES Student(studentID) ON DELETE CASCADE

);

-- Create the Privilege table

CREATE TABLE Privilege (

privilegeID INT PRIMARY KEY,

name VARCHAR(255),

description TEXT,

maxItems INT,

);

-- Create the Category table

CREATE TABLE Category (

categoryID INT PRIMARY KEY,

name VARCHAR(255),

description TEXT,

durationDays INT,

durationHours INT,

resourceID INT,

privilegeID INT,

-- Add other attributes for Category here

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON DELETE CASCADE,

FOREIGN KEY (privilegeID) REFERENCES Privilege(privilegeID) ON DELETE CASCADE

);

-- Create the CourseOffering\_Privilege junction table

CREATE TABLE CourseOffering\_Privilege (

courseOfferingID INT,

privilegeID INT,

PRIMARY KEY (courseOfferingID, privilegeID),

FOREIGN KEY (courseOfferingID) REFERENCES CourseOffering(offerID) ON DELETE CASCADE,

FOREIGN KEY (privilegeID) REFERENCES Privilege(privilegeID) ON DELETE CASCADE

);

# Part 2.2: Input values in every table SQL Code.

INSERT INTO Member (memberID, name, address, phone, email, status, comments)

VALUES

(1, 'John Doe', '123 Main St', '555-1234', 'john@example.com', 'active', 'Regular member'),

(2, 'Jane Smith', '456 Elm St', '555-5678', 'jane@example.com', 'active', 'Premium member'),

(3, 'Bob Johnson', '789 Oak St', '555-9012', 'bob@example.com', 'inactive', 'Former member'),

(4, 'Alice Brown', '101 Maple St', '555-3456', 'alice@example.com', 'active', NULL),

(5, 'Charlie Green', '202 Pine St', '555-7890', 'charlie@example.com', 'active', 'VIP member');

-- Inserting data into the Staff table

INSERT INTO Staff (staffID, memberID)

VALUES

(101, 1),

(102, 3),

(103, 4),

(104, 5),

(105, 2);

-- Inserting data into the Student table

INSERT INTO Student (studentID, memberID, pointsEarned)

VALUES

(201, 4, 100),

(202, 5, 150),

(203, 1, 200),

(204, 2, 175),

(205, 3, 80);

-- Inserting data into the Acquisition table

INSERT INTO Acquisition (acqID, memberID, resourceName, description, make, model, year, urgency, status, fundCode, VendorCode, price, notes)

VALUES

(301, 1, 'Laptop', 'High-performance laptop', 'Dell', 'XPS 15', 2022, 'High', 'Pending', 'FUND001', 'VENDOR001', 1500.00, NULL),

(302, 2, 'Projector', '1080p projector', 'Epson', 'PowerLite', 2021, 'Medium', 'Approved', 'FUND002', 'VENDOR002', 800.00, NULL),

(303, 3, 'Tablet', 'Tablet for classroom use', 'Samsung', 'Galaxy Tab S7', 2023, 'Low', 'Pending', 'FUND003', 'VENDOR003', 600.00, 'Urgent requirement'),

(304, 4, 'Printer', 'Color laser printer', 'HP', 'Color LaserJet Pro', 2020, 'Low', 'Approved', 'FUND004', 'VENDOR004', 400.00, NULL),

(305, 5, 'Smartwatch', 'Fitness tracker', 'Fitbit', 'Versa 3', 2022, 'Medium', 'Pending', 'FUND005', 'VENDOR005', 250.00, NULL);

-- Inserting data into the Loan table

INSERT INTO Loan (loanID, memberID, resourceID, dateTimeBorrowed, dateTimeReturned, dateTimeDue)

VALUES

(401, 1, 301, '2024-04-01 09:00:00', '2024-04-10 09:00:00', '2024-04-15 09:00:00'),

(402, 2, 302, '2024-04-02 10:00:00', '2024-04-11 10:00:00', '2024-04-16 10:00:00'),

(403, 3, 303, '2024-04-03 11:00:00', '2024-04-12 11:00:00', '2024-04-17 11:00:00'),

(404, 4, 304, '2024-04-04 12:00:00', '2024-04-13 12:00:00', '2024-04-18 12:00:00'),

(405, 5, 305, '2024-04-05 13:00:00', NULL, '2024-04-20 13:00:00');

-- Inserting data into the Reservation table

INSERT INTO Reservation (reservationID, memberID, resourceID, dateTimeReserved, dateTimeDue)

VALUES

(501, 1, 301, '2024-04-01 08:00:00', '2024-04-02 08:00:00'),

(502, 2, 302, '2024-04-02 09:00:00', '2024-04-03 09:00:00'),

(503, 3, 303, '2024-04-03 10:00:00', '2024-04-04 10:00:00'),

(504, 4, 304, '2024-04-04 11:00:00', '2024-04-05 11:00:00'),

(505, 5, 305, '2024-04-05 12:00:00', '2024-04-06 12:00:00');

-- Inserting data into the Resource table

INSERT INTO Resource (resourceID, categoryID, privilegeID)

VALUES

(301, 1, 1),

(302, 2, 2),

(303, 3, 3),

(304, 4, 4),

(305, 5, 5);

-- Inserting data into the Movable table

INSERT INTO Movable (resourceID, name, manufacturer, model, year, assetValue, BuildingID)

VALUES

(301, 'Laptop', 'Dell', 'XPS 15', 2022, 1500.00, NULL),

(302, 'Projector', 'Epson', 'PowerLite', 2021, 800.00, NULL),

(303, 'Tablet', 'Samsung', 'Galaxy Tab S7', 2023, 600.00, NULL),

(304, 'Printer', 'HP', 'Color LaserJet Pro', 2020, 400.00, NULL),

(305, 'Smartwatch', 'Fitbit', 'Versa 3', 2022, 250.00, NULL);

-- Inserting data into the Immovable table

INSERT INTO Immovable (resourceID, Capacity, Room, Building, Campus)

VALUES

(301, NULL, NULL, NULL, NULL),

(302, NULL, NULL, NULL, NULL),

(303, NULL, NULL, NULL, NULL),

(304, NULL, NULL, NULL, NULL),

(305, NULL, NULL, NULL, NULL);

-- Inserting data into the Category table

INSERT INTO Category (categoryID, name, description, durationDays, durationHours)

VALUES

(1, 'Electronics', 'Electronic devices', NULL, NULL),

(2, 'Projectors', 'Projection equipment', NULL, NULL),

(3, 'Tablets', 'Portable touch-screen computers', NULL, NULL),

(4, 'Printers', 'Output devices for documents', NULL, NULL),

(5, 'Speaker', 'Audio speakers for various applications', NULL, NULL);

-- Inserting data into the Privilege table

INSERT INTO Privilege (privilegeID, name, description, maxItems)

VALUES

(1, 'Standard', 'Standard user privileges', 3),

(2, 'Premium', 'Enhanced user privileges', 5),

(3, 'Admin', 'Administrator privileges', NULL),

(4, 'Guest', 'Limited guest privileges', 1),

(5, 'VIP', 'VIP user privileges', 10);

-- Inserting data into the CourseOffering table

INSERT INTO CourseOffering (offerID, studentID, cid, course, semester, year, dateBegin, dateEnd)

VALUES

(601, 201, 101, 'Introduction to Computer Science', 'Fall', 2024, '2024-09-01', '2024-12-15'),

(602, 202, 102, 'English Composition', 'Spring', 2024, '2024-01-15', '2024-05-30'),

(603, 203, 103, 'Calculus I', 'Fall', 2024, '2024-09-01', '2024-12-15'),

(604, 204, 104, 'History of Art', 'Spring', 2024, '2024-01-15', '2024-05-30'),

(605, 205, 105, 'Chemistry 101', 'Fall', 2024, '2024-09-01', '2024-12-15');

# Part 2.3: Implementing the given queries SQL Code.

### **Q1: Print the name of student(s) who has/have enrolled in the course with**

### **course id xxx.**

*SQL Code:*

SELECT s.name AS StudentName

FROM Student s

JOIN CourseOffering co ON s.studentID = co.studentID

WHERE co.cid = 'xxx';

### **Q2: Print the maximal number of speakers that the student with name xxx can borrow. The student is enrolled in the course with course id yyy. Note: speaker is a category.**

*SQL Code:*

SELECT MAX(p.maxItems) AS MaxSpeakers

FROM Student s

JOIN CourseOffering co ON s.studentID = co.studentID

JOIN Reservation r ON s.memberID = r.memberID

JOIN Resource res ON r.resourceID = res.resourceID

JOIN Category cat ON res.categoryID = cat.categoryID

JOIN Privilege p ON cat.privilegeID = p.privilegeID

WHERE s.name = 'xxx' AND co.cid = 'yyy' AND cat.name = 'speaker';

### **Q3: For a staff member with id number xxx, print his/her name and phone number, the total number of reservations that the staff had made in 2022.**

*SQL Code:*

SELECT m.name AS StaffName, m.phone AS StaffPhone,

COUNT(r.reservationID) AS TotalReservations

FROM Member m

JOIN Staff s ON m.memberID = s.memberID

JOIN Reservation r ON s.memberID = r.memberID

WHERE s.staffID = xxx

AND YEAR(r.dateTimeReserved) = 2022

GROUP BY m.name, m.phone;

### **Q4: Print the name(s) of the student member(s) who has/have borrowed the category with the name of camera, of which the model is xxx, in this year. Note: camera is a category, model attribute must be in movable table, and “this year” must be decided by the system.**

*SQL Code:*

SELECT DISTINCT m.name AS StudentName

FROM Member m

JOIN Student s ON m.memberID = s.memberID

JOIN Loan l ON s.memberID = l.memberID

JOIN Movable mv ON l.resourceID = mv.resourceID

JOIN Resource r ON mv.resourceID = r.resourceID

JOIN Category c ON r.categoryID = c.categoryID

WHERE c.name = 'camera' AND mv.model = 'xxx' AND YEAR(l.dateTimeBorrowed) = YEAR(GETDATE());

### **Q5: Find the moveable resource that is the mostly loaned in current year. Print the resource id and resource name. Note: “current year” must be decided by the system.**

*SQL Code:*

SELECT TOP 1 r.resourceID, mv.name AS ResourceName

FROM Loan l

JOIN Movable mv ON l.resourceID = mv.resourceID

JOIN Resource r ON mv.resourceID = r.resourceID

WHERE YEAR(l.dateTimeBorrowed) = YEAR(GETDATE())

GROUP BY r.resourceID, mv.name

ORDER BY COUNT(\*) DESC;

### **Q6: For each of the three days, including May 1, 2024, June 5, 2024 and September 19, 2024, print the date, the name of the room, and the total number of reservations made for the room xxx on each day.**

*SQL Code:*

SELECT

DATE(dateTimeReserved) AS ReservationDate,

im.Room AS RoomName,

COUNT(r.reservationID) AS TotalReservations

FROM

Reservation r

JOIN

Resource res ON r.resourceID = res.resourceID

JOIN

Immovable im ON res.resourceID = im.resourceID

WHERE

im.Room = 'xxx'

AND (

DATE(dateTimeReserved) = '2024-05-01'

OR DATE(dateTimeReserved) = '2024-06-05'

OR DATE(dateTimeReserved) = '2024-09-19'

)

GROUP BY

DATE(dateTimeReserved), im.Room;

Summary

We planned an information base mapping for the SCS Asset The executives Undertaking, illustrating tables like Part, Staff, Understudy, Securing, Credit, Reservation, Asset, Versatile, Relentless, Classification, Honor, and CourseOffering. Connections, cardinalities, and unfamiliar key imperatives were characterized to guarantee information honesty. The legacy structure for Staff and Understudy from Part was laid out. SQL code was given to make the data set tables, including credits and unfamiliar key requirements. Genuine information was embedded into the tables to populate the data set, reflecting true situations. Inquiries were figured out to recover explicit data, for example, understudy names, staff subtleties, credited assets, and reservations, meeting different standards. Testing was directed to guarantee the inquiries delivered significant outcomes and stuck to the predetermined necessities.