

**INFO6001: Database Management 1**

**Assignment 3: SCS Resource Management Database Project**

**Logical Database Design**

# **Preface**

This document details the requirements analysis and conceptual design for developing the database of SCS Resource Management. The requirements analysis involves identifying data needs, transaction requirements, and business rules. This will ensure that all essential data is identified, managed, and monitored in compliance with the store's operational protocols. The conceptual design includes a detailed entity relationship diagram in UML that outlines the required entities, their characteristics, and their connections. A data dictionary is included to offer additional information on the entities, characteristics, and connections. This conceptual model was developed by analyzing needs. This document serves as an introduction to the project titled "Database Design of SCS Resource Management: Normalization process. We were tasked with conceptualizing and developing a bespoke database system tailored to the needs of the school database. The assignment comprises two distinct elements, both essential for achieving the goal of resource optimization.

Last month, database management assessment 1 involved a comprehensive requirement analysis procedure. Our task was to determine the crucial data requirements, intricate transactional details, and fundamental business principles that will establish the basis of the proposed database. We thoroughly comprehended the system's operational requirements through interviews, consultations, and a detailed examination of the available documentation.

We emphasized the conceptual design step, where students convert obtained insights into a tangible entity-relationship model. We may utilize the Unified Modelling Language (UML) syntax to construct a sophisticated network of entities, linkages, and attributes that depict the SCS Resource Management ecosystem. We are encouraged to tackle this task with determination, creativity, and a steadfast commitment to excellence since they are the custodians of information and innovation. By using their analytical capabilities, technical competence, and teamwork abilities, we can create a database system that meets present requirements and lays a solid foundation for a future marked by effectiveness and efficiency.

* 1. **Reflect on assignment 2**

Assignment 2 contains errors in the relational mapping, primarily due to erroneous foreign keys in each table. These foreign keys may be either excessive or insufficient according to the requirements. Part 3 should be attempted and submitted individually. Part 2 consists solely of a comprehensive inventory of unnormalized relations in DBDL derived from EER. Part 3 necessitates providing remarks on each table on its normalization form and the procedure of normalization for the unnormalized table. For assessment 2, our objective is to standardize the EER. Data normalization reduces data duplication, enhances data consistency, and streamlines data processing. The next part provides a comprehensive explanation of the normalization procedures implemented on the Entity-Relationship Diagram (ERD) that was established during the conceptual design stage. The normalization process will be executed in a sequential manner, gradually converting the initial Entity-Relationship Diagram (ERD) into a more advanced normal form. Our primary objective is to get Boyce-Codd Normal Form (BCNF), which guarantees the presence of minimal redundancy and facilitates excellent data integrity. Functional dependencies occur when there is a unique relationship between one attribute (or a group of attributes) in a database and another attribute (or group of attributes) in the same table. Subsequently, in accordance with the established functional dependencies, redundant attributes will be eliminated from the tables. Foreign keys can be used to retrieve these attributes from related tables. This measure minimizes data replication and enhances data uniformity. If any tables fail to meet the BCNF criteria due to partial or transitive dependencies, we will break them down into smaller tables that have appropriate foreign key associations. This guarantees that every entry in a table corresponds to a singular fact and reduces the likelihood of data irregularities. All entities, properties, primary keys, foreign keys, and relationships will be thoroughly described in the final normalized schema. This documentation functions as a point of reference for the implementation and upkeep of databases and provide SQL script to finish what are we going to work with.

1. **Requirement specification**

**Data Requirements**

Acquisition

Members can suggest acquisition. Acquisition will be prioritized for research & teaching. Acquisition table should include resourceName, description, make, model, year, urgency, status, fundCode, VendorCode, price, notes.

Member:

Both staff and students are called members. Members can access resources based on their privilege, which identify by their type of member and degree offered if member are students. The members can reserve the resource (if available to make sure they can use reserved resources for a given time, it will be on a first come first serve basis). Members should include the name, address, phone, email address, status of the member and comments of members.

Staff

Staff can be identified by a unique member identification code. Staff can borrow resources with unlimited privileges. The staff table is mandatory to belong to member’s table. Staff table should include staffID

Student

Students can be identified by a unique member identification code. Students can borrow resources depending on the degree they are studying for. The student table should include studentID and pointsEarned and should be mandatory belong to member of table.

CourseOffering

CourseOffering can be identified by unique course unique number. The course offered will cause the difference of privilege of student. Students can enroll in many courses at the same time. CourseOffering should include offerID, cid, course, semester, year, startDate, endDate.

Resource:

Resources can be identified by resource unique number. Each resource is separated by category which is identified by a unique code, name, description, and max time allowed. Resource basically categorizes as movable and immovable. Resources should include resourceID, description and status.

Movable

All movables are stored in the resource center using unique numbers. Moveable resources can be loaned to student or staff members based on their privilege. Moveable resources table is mandatory to resource’s table and should include name, manufacturer, model, year, assetValue, buildingID

Immovable

Immovable resources are considered as rooms located in different buildings. Immovable should include room, building, capacity, campus.

Category

Category is identified by a unique code, name, description, and max time allowed to borrow/book. Category is type of resource. For example, computer, laptop, mouse could be considered as “ICT”. Notes, pen, erase could be considered as “stationary.” Category should include code {PK}, name, description, durationDays, durationHours.

Privilege

The privilege is the right to borrow a specific resource. A staff has a no-limited privilege. Privilege for staff can be identified by memberID equal to “staff.” Student privilege is based on CourseOffering. Privilege depends on the type of category of student. For example, some students from coursework degree can’t use the research room but some students from research degree can use research room even they are in the same course. Privilege should include name, description, maxItems.

Loan

Loans can be created for resources for members to borrow and return. Loans depend on privilege requirements as mentioned. Loans should include dateTimeBorrowed, dateTimeReturn, dateTimeDue.

Reservation

Reservation can only be created for immovable resource to member. Reservation can be authorized based on first come first served. Reservation should include dateTimeReserved, dateTimeDue)

**Transaction Requirements**

Insert, update, and delete Acquisition (AcqID, resourceName, description, make, model, year, urgency, status, fundCode, VendorCode, price, notes)

Insert, update, and delete Student (studentID, pointsEarned)

Insert, update, and delete Staff (staffID)

Insert, update, and delete Member (MemberID, name, address, phone, email, status, comments)

Insert, update, and delete Loan (LoanID, dataTimeBorrowed, dateTimeReturn, dateTimeDue)

Insert, update, and delete Reservation (reservationID, dateTimeBorrowed, dateTimeReturned, dateTimeDue)

Insert, update, and delete Moveable (name, manufacturer, model, year, assestValue, BuildingID)

Insert, update, and delete Immovable (capacity, Room, building, campus)

Insert, update, and delete details of CourseOffering (offerID, cid, course, semester, year, dateBegin, dateEnd)

Insert, update, and delete Privilege (PrivilegeID, name, description)

Insert, update, and delete Resource (ResourceID, description, status)

Insert, update, and delete Category (code, name, description, durationDays, durationHours)

**Data queries:**

List total number of resources.

List the name, privilege of member.

Identify maximum borrow hour of resource.

List name of privilege relate to loan.

Identify the maximum borrow hour no longer than 7 days.

Identify all loan links to moveable resources.

Identify all reservation links to immovable resources.

Search for movable resource by loanID

Search for immovable resource by BuildingID

Search for any resource by resourceID

Search for category by code.

**Business Rules**

Movable resources are located by BuildingID

All resources have unique ID.

Staff have unlimited privileges.

Students have privileges based on their course studying.

Students can enrolls a maximum of 3 course.

Course offer to students might have duplicated privileges.

The members can reserve resources for their wishing time.

Overdue loan will be notified.

All members have limited time for loans.

All members have limited time for reservation.

All members can make acquisitions for research and learning.

All members can borrow a maximum of 20 items.

Students can only borrow a maximum of 5 items.

All members can’t borrow over 30 days.

Acquisition for teaching & research will be prioritized.

1. **EER Model**

A computer screen shot of a diagram

Description automatically generated

All the requirements need to be adjusted based on EER diagram. In assessment 1, deduction was addressed for this reason: there should not be any FK in the ERD and Data Dictionary, there should not be any PK in the sub-entities, there are multiplicity problems with some of the entities. Please, refer to the complete ERD solution given at the end of the feedback report to see. These mistakes need to be fixed in order to meet the requirement of assessment 2.

A computer screen shot of a diagram

Description automatically generated

**Entity**

Note: xxx entities (including sup & sub)

|  |  |  |  |
| --- | --- | --- | --- |
| Entity Name | Description | Aliases – Others way to call Entity name | Occurrence |
| Member | Describing member of school | User | When new member join school |
| Student | Describing student at school |  | When new student joins new course |
| Staff | Describing staff of school |  | When new staff joins school |
| CourseOffering | Describing course students are studying |  | When new student enrols new course |
| Privilege | Describing the right of resource can be borrowed for each member | Accessibility | When member is created with specific privilege |
| Loan | Describing loans that made by member | Resource borrowed | When a loan is made by a member |
| Movable | Describing moveable resource |  | When a moveable resource is created. |
| Immovable | Describing immoveable resource |  | When a reservation is created. |
| Resource | Describing resource details | Asset | When a resource is created |
| Category | Describing type of category |  | When a resource is created |
| Acquisition | Describing details of acquisition | Resource request | When a member request acquisition |
| Reservation | Describing details of reservation | Resource reserve | When a member request reservation |

Relationships

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Member | 1..1 | has | 0.. \* | Loan |
|  | 1..1 | make | 0.. \* | Acquisition |
|  | 1.. 1 | has | 0.. \* | Reservation |
| Student | 1.. \* | enrols | 0.. \* | CourseOffering |
|  | Subclass of |  | Member |
| Staff |  | Subclass of |  | Member |
| Loan | 0.. \* | to | 1..1 | Movable |
| Movable |  | Subclass of |  | Resource |
| Immovable |  | Subclass of |  | Resource |
| Resource | 0.. \* | Belongs to | 1..1 |  |
| Reservation | 0.. \* | Reserve for | 1..1 | Resource |
| CourseOffering | 0.. \* | enrols | 0.. \* | Privilege |
| Privilege | 0.. \* | belongsTo | 1…1 | Category |

Attributes:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Acquisition | AcqID | Acquisition identification code | Char | FALSE | FALSE | FALSE |  |
|  | resourceName | Name of resource | Char | FALSE | FALSE | FALSE |  |
|  | description | Description of resource | Varchar | FALSE | FALSE | FALSE |  |
|  | make | Material of resource | Varchar | FALSE | FALSE | FALSE |  |
|  | Model | Model of resource | Varchar | FALSE | FALSE | FALSE |  |
|  | Year | Year of resource | YearTime | FALSE | FALSE | FALSE |  |
|  | Urgency | Urgency of resource | Char | FALSE | FALSE | FALSE |  |
|  | status | Status of acquisition | Char | FALSE | FALSE | FALSE |  |
|  | fundCode | Funding information of resource | Varchar | FALSE | FALSE | FALSE |  |
|  | VendorCode | Vendor of resource | Varchar | FALSE | FALSE | FALSE |  |
|  | Price | Price of resource | Numberic | FALSE | FALSE | FALSE |  |
|  | notes | Notes for resource | Char | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Member | memberID | Unique member ID | Char | FALSE | FALSE | FALSE |  |
|  | name | Name of member | Varchar | FALSE | FALSE | FALSE |  |
|  | address | Address of member | Varchar | FALSE | FALSE | FALSE |  |
|  | phone | Phone number of members | Varchar | FALSE | FALSE | FALSE |  |
|  | email | Email of members | Varchar | FALSE | FALSE | FALSE |  |
|  | status | Status of member | Array | FALSE | FALSE | FALSE |  |
|  | Comments | Comments for member | Char | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Student | StudentID | Student unique number | Char | FALSE | FALSE | FALSE |  |
|  | pointsEarned | Point of student | Numberic | FALSE | FALSE | FALSE | 12 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Staff | StaffID | Staff unique identification number | Char | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| CourseOffering | offerID | Course identification code | Char | FALSE | FALSE | FALSE |  |
|  | cid | Course name | Varchar | FALSE | FALSE | FALSE |  |
|  | course | Name of course | Varchar | FALSE | FALSE | FALSE |  |
|  | semester | Description of course | Varchar | FALSE | FALSE | FALSE |  |
|  | year | Year |  | FALSE | FALSE | FALSE |  |
|  | dateBegin | Date of starting course | dateTime | FALSE | FALSE | FALSE |  |
|  | dateEnd | Date of ending course | dateTime | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Privilege | PrivilegeID | Privilege identification code | Varchar | FALSE | FALSE | FALSE |  |
|  | name | Name of privilege | Varchar | FALSE | FALSE | FALSE |  |
|  | description | Description of privilege | Varchar | FALSE | FALSE | FALSE |  |
|  | maxItems | Maximum of items | Varchar | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Loan | LoanID | Unique loan identity number | Char | FALSE | FALSE | FALSE |  |
|  | dateTimeBorrowed | Date of starting loan | date | FALSE | FALSE | FALSE |  |
|  | dateTimeReturn | Date of return resource, finishing off loan | date | FALSE | FALSE | FALSE |  |
|  | dateTimeDue | Date after mandatory return day | date | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Moveable | name | Unique item identification code | Char | FALSE | FALSE | FALSE |  |
|  | manufacturer | Building identification code | Char | FALSE | FALSE | FALSE |  |
|  | model | Name of the movable resource | Varchar | FALSE | FALSE | FALSE |  |
|  | year | Year of resource | Varchar | FALSE | FALSE | FALSE |  |
|  | assetValue | Value of resource | Varchar | FALSE | FALSE | FALSE |  |
|  | BuildingID | Building unique identification number | Varchar | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Immovable | room | Unique room identification code | char | FALSE | FALSE | FALSE |  |
|  | Building | Building identification code | char | FALSE | FALSE | FALSE |  |
|  | capacity | Space of the room | numberic | FALSE | FALSE | FALSE |  |
|  | campus | Campus of the room | array | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Resource | ResourceID | ResourceID unique code | Char | FALSE | FALSE | FALSE |  |
|  | description | Description of resource | Char | FALSE | FALSE | FALSE |  |
|  | status | Status of resource | Array | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Category | Code | Category identification code | Char | FALSE | FALSE | FALSE |  |
|  | Name | Name of category | Varchar | FALSE | FALSE | FALSE |  |
|  | description | Description of category | Varchar | FALSE | FALSE | FALSE |  |
|  | DurationDays | Duration of day | Time | FALSE | FALSE | FALSE |  |
|  | DurationHours | Duration of Hours | Time | FALSE | FALSE | FALSE |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Reservation | ReservationID | Acquistion identification code | Char | FALSE | FALSE | FALSE |  |
|  | dataTimeReserved | Beginning date of reservation | Time | FALSE | FALSE | FALSE |  |
|  | dateTimeDue | Due date of reservation | Time | FALSE | FALSE | FALSE |  |

1. **The relational model mapped from EER.**

**Relational Schema:**

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

**Primary Key** acqID

**Foreign Key** MemberID **reference** Member (MemberID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Member** (memberID {PK}, name, address, phone, email, status, comments)

**Primary Key** memberID

**StaffID** (memberID{PK}, name, address, phone, email, status, comments)

**Primary Key** MemberID

**Foreign Key** MemberID **reference** Member (MemberID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Student** (memberID{PK}, name, address, phone, email, status, comments)

**Primary Key** MemberID

**Foreign Key** MemberID **reference** Member (MemberID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Loan** (LoanID{PK}, MemberID{FK}, resourceID{FK}, dateTimeBorrowed, dateTimeReturned, dateTimeDue)

**Primary Key** LoanID

**Foreign Key** memberID **references** Member(memberID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** resourceID **references** Movable(resourceID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Reservation** (ReservationID{PK}, MemberID{FK}, dateTimeReserved, dateTimeDue)

**Primary Key** ReservationID

**Foreign Key** MemberID **reference** Member (MemberID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ResourceID **reference** Resource (ResourceID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Resource** (ResourceID, name, description, status)

**Primary Key** ResourceID

**Movable** (ResourceID {PK}, description, status, name, manufacturer, model, year, assetValue, buildingID)

**Primary Key** ResourceID

**Foreign Key** BuildingID **reference** Movable (buildingID)

**Immovable** (ResourceID{PK}, description, status, capacity, room, building, campus)

**Primary Key** ResourceID

**Foreign Key** BuildingID **reference** Movable (buildingID)

**Category** (Code {PK}, resourceID{FK}, name, description, durationDays, durationHours)

**Primary Key** Code

**Foreign Key** ResourceID reference Resource (ResourceID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Privilege** (privilID {PK}, name, description, maxItems)

**Primary Key** privilID

**Foreign key** OfferID **reference** CourseOffering(OfferID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**CourseOffering** (offerID{PK}, studentID{FK}, cid, course, semester, year, dateBegin, dateEnd)

Primary Key privilID

Foreign Key studentID

* 1. **Normalized Relational Schema in DBDL**

**Functional Dependencies (FDs):**

**Acquisition:** acqID -> memberID, resourceName, description, make, model, year, urgency, status, fundCode, VendorCode, price, notes

**Member:** memberID -> name, address, phone, email, status, comments

**Staff** (**assuming Staff inherits from Member)**: memberID -> name, address, phone, email, status, comments.

**Student** (**assuming Staff inherits from Member)**: memberID -> name, address, phone, email, status, comments.

**Loan:** LoanID -> memberID, resourceID, dateTimeBorrowed, dateTimeReturned, dateTimeDue

**Reservation:** ReservationID -> memberID, resourceID, dateTimeReserved, dateTimeDue

**Resource:** resourceID -> name, description, status

**Movable** (assuming movable is a part of Resource):resourceID -> description, status, name, manufacturer, model, year, assetValue, buildingID

**Immovable** (assuming movable is a part of Resource)**:** resourceID -> description, status, capacity, room, building, campus.

**Category:** Code -> resourceID, name, description, durationDays, durationHours

**Privilege:** privilID -> OfferID, name, description, maxItems

**CourseOffering:** offerID -> studentID, cid, course, semester, year, dateBegin, dateEnd

1. **Normalization Steps:**

**Member** (memberID {PK}, name, address, phone, email, status, comments): remains the same. This is already BCNF. No partial dependencies exist. Primary key (memberID) determines all other attributes.

**Staff** (staffID {PK}): separate table for Staff due to inheritance. Only memberID is removed. Separate tables were created for Staff due to inheritance from Member.

**Student** (studentID {PK}): separate table for Student due to inheritance. Only memberID is removed. Separate table was created for Student due to inheritance from Member.

**Acquisition** (acqID {PK}, memberID {FK}, vendorCode {FK}, resourceID {FK}, price, dateTimeReceived, notes): remove redundant attributes: resourceName (assuming this can be retrieved from Resource), description, make, model, year, urgency, status, and fundCode (assuming these might be related to another entity). A separate table was created for acquired resources to capture detaild information about each resource.

**Loan** (LoanID {PK}, memberID {FK}, resourceID {FK}, dateTimeBorrowed, dateTimeReturned, dateTimeDue): remains the same. This is already BCNF. No partial dependencies exist. Primary key (LoanID) determines all other attributes.

**Reservation** (ReservationID {PK}, memberID {FK}, resourceID {FK}, dateTimeReserved, dateTimeDue): remains the same. This is already BCNF. No partial dependencies exist. Primary key (ReservationID) determines all other attributes.

**Resource** (resourceID {PK}, name, status): remove redundant attribute: description (assuming this might be in another table).

**Movable** (resourceID {PK}, manufacturer, model, year, assetValue) remove redundant attributes: description, status, and name (assuming these can be retrieved from Resource), and buildingID (assuming this might be related to another entity).

**Immovable** (resourceID {PK}, capacity, room, building, campus) remove redundant attributes: description and status (assuming these can be retrieved from Resource).

**Category** (Code {PK}, name, description, durationDays, durationHours): remove foreign key to Resource. Category might not be directly related to a specific resource.

**Privilege** (privilID {PK}, name, description, maxItems): remove foreign key to CourseOffering. Privilege might not be specific to a course offering.

**CourseOffering** (offerID {PK}, cid, course, semester, year, dateBegin, dateEnd): remove foreign key to Student. Enrollment information might be on a separate table.

* **Additional Tables**

**CourseOfferingToPrivilege** (MemberID (FK), offerID (FK), privilID (PK))

**StaffPrivilege** (MemberID(PK), privilID (FK)) (Many-to-Many relationship)

**StudentEnrollCourseOffering** (StudentID,OfferID)

1. **SQL to create normalized SCS Resource Management database**

|  |
| --- |
| DROP DATABASE IF EXISTS NormalisedSCS  GO  CREATE DATABASE NormalisedSCS  GO  USE NormalisedSCS  GO  drop table if EXISTS StudentEnrollCourseOffering;  drop table if EXISTS CourseOfferingToPrivilege;  drop table if EXISTS CourseOffering;  drop table if EXISTS StaffPrivilege;  drop table if EXISTS Privilege;  drop table IF EXISTS Reservation;  drop table if EXISTS Loan;  drop table if EXISTS Immovable;  drop table if EXISTS Movable;  drop table if EXISTS Acquisition;  drop table if EXISTS Category;  drop table if EXISTS Resource;  drop table if EXISTS Student;  drop table if EXISTS Staff;  drop table if EXISTS Member;  CREATE TABLE Member (  memberID VARCHAR(255) PRIMARY KEY ,  name VARCHAR(255) NOT NULL,  address TEXT,  phone VARCHAR(20),  email VARCHAR(255),  status VARCHAR(50) NOT NULL,  comments TEXT  );  CREATE TABLE CourseOffering (  offerID VARCHAR(255) PRIMARY KEY,  cid VARCHAR(255) NOT NULL,  course VARCHAR(255) NOT NULL,  semester VARCHAR(50) NOT NULL,  year INT NOT NULL,  dateBegin DATE NOT NULL,  dateEnd DATE NOT NULL  );  CREATE TABLE Resource (  resourceID VARCHAR(255) PRIMARY KEY NOT NULL,  name VARCHAR(255) NOT NULL,  status VARCHAR(50) NOT NULL  );  CREATE TABLE Category (  Code VARCHAR(255) PRIMARY KEY NOT NULL,  ResourceID VARCHAR(255),  name VARCHAR(255) NOT NULL,  description TEXT,  durationDays INT,  durationHours INT,  FOREIGN KEY (ResourceID) REFERENCES Resource(resourceID)  );  CREATE Table Privilege  (  privilID VARCHAR(255) PRIMARY KEY NOT NULL,  Code VARCHAR(255) NOT NULL,  name VARCHAR(255) NOT NULL,  description TEXT,  maxItems INT,  FOREIGN KEY (Code) REFERENCES Category(Code)  );  CREATE TABLE Student (  StudentID VARCHAR(255) PRIMARY KEY NOT NULL,  PointEearned INT DEFAULT 12,  major VARCHAR(255) NOT NULL,  name VARCHAR(255) NOT NULL,  address TEXT,  phone VARCHAR(20),  email VARCHAR(255),  status VARCHAR(50) NOT NULL,  comments TEXT,  FOREIGN KEY (StudentID) REFERENCES Member(memberID) ON UPDATE CASCADE ON DELETE CASCADE  );  CREATE TABLE Staff (  StaffID VARCHAR(255) PRIMARY KEY NOT NULL,  title VARCHAR(255) NOT NULL,  name VARCHAR(255) NOT NULL,  address TEXT,  phone VARCHAR(20),  email VARCHAR(255),  status VARCHAR(50) NOT NULL,  comments TEXT  FOREIGN KEY (StaffID) REFERENCES Member(memberID) ON UPDATE CASCADE ON DELETE CASCADE  );  CREATE TABLE CourseOfferingToPrivilege (  StudentID VARCHAR(255),  offerID VARCHAR(255),  privilID VARCHAR(255),  FOREIGN KEY (studentID) REFERENCES Student(StudentID) ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY (offerID) REFERENCES CourseOffering(offerID) ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY (privilID) REFERENCES Privilege(privilID) ON DELETE CASCADE ON UPDATE CASCADE  );  CREATE TABLE StudentEnrollCourseOffering (  StudentID VARCHAR(255),  offerID VARCHAR(255),  PRIMARY KEY (StudentID, offerID),  FOREIGN KEY (StudentID) REFERENCES Student(StudentID) ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY (offerID) REFERENCES CourseOffering(offerID) ON DELETE CASCADE ON UPDATE CASCADE  );  CREATE TABLE StaffPrivilege  (  StaffID VARCHAR(255),  privilID VARCHAR(255),  PRIMARY KEY (StaffID, privilID),  FOREIGN KEY (StaffID) REFERENCES Member(memberID) ON UPDATE CASCADE ON DELETE CASCADE,  FOREIGN KEY (privilID) REFERENCES Privilege(privilID)  );  CREATE TABLE Acquisition (  acqID VARCHAR(255) PRIMARY KEY NOT NULL,  memberID VARCHAR(255) NOT NULL,  vendorCode VARCHAR(255) NOT NULL,  resourceID VARCHAR(255) NOT NULL,  price DECIMAL(5,2) NOT NULL,  dateTimeReceived DATETIME NOT NULL,  notes TEXT,  FOREIGN KEY (memberID) REFERENCES Member(memberID),  FOREIGN KEY (resourceID) REFERENCES Resource(resourceID)  );  CREATE TABLE Loan (  LoanID VARCHAR(255) PRIMARY KEY NOT NULL,  memberID VARCHAR(255) NOT NULL,  resourceID VARCHAR(255) NOT NULL,  dateTimeBorrowed DATETIME NOT NULL,  dateTimeReturned DATETIME DEFAULT NULL,  dateTimeDue DATETIME NOT NULL,  FOREIGN KEY (memberID) REFERENCES Member(memberID) ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON DELETE CASCADE ON UPDATE CASCADE  );  CREATE TABLE Reservation (  ReservationID VARCHAR(255) PRIMARY KEY NOT NULL,  memberID VARCHAR(255) NOT NULL,  resourceID VARCHAR(255) NOT NULL,  dateTimeReserved DATETIME NOT NULL,  dateTimeDue DATETIME NOT NULL,  FOREIGN KEY (memberID) REFERENCES Member(memberID),  FOREIGN KEY (resourceID) REFERENCES Resource(resourceID)  );  CREATE TABLE Movable (  resourceID VARCHAR(255) PRIMARY KEY NOT NULL,  manufacturer VARCHAR(255),  model VARCHAR(255),  year INT,  assetValue DECIMAL(10,2) NOT NULL,  CONSTRAINT FK\_Movable\_Resource FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON DELETE CASCADE ON UPDATE CASCADE  );  CREATE TABLE Immovable (  resourceID VARCHAR(255) PRIMARY KEY NOT NULL,  capacity INT,  room VARCHAR(255),  building VARCHAR(255),  campus VARCHAR(255),  CONSTRAINT FK\_Immovable\_Resource FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON DELETE CASCADE ON UPDATE CASCADE  ); |

1. **SQL script**
   1. **INSERT DATA**

|  |
| --- |
| -- Insert data into the Member table  INSERT INTO Member (memberID, name, address, phone, email, status, comments)  VALUES ('STUDENT1', 'John Doe', '123 Main St, Anytown, USA', '555-1234', 'johndoe@example.com', 'Active', 'No comments'),  ('STUDENT2', 'Jane Smith', '456 Elm St, Anytown, USA', '555-5678', 'Janesmith@example.com', 'Active', 'No comments'),  ('STUDENT3', 'Jim Brown', '789 Oak St, Anytown, USA', '555-9012', 'jimbrown@gmail.com', 'Active', 'No comments'),  ('STUDENT4', 'Jill White', '101 Pine St, Anytown, USA', '555-3456', 'jillwhite@gmail.com', 'Active', 'No comments'),  ('STUDENT5', 'Jack Black', '111 Maple St, Anytown, USA', '555-7890', 'Jackblack@example.com', 'Active', 'No comments'),  ('STAFF1', 'Sue Johnson', '123 Main St, Anytown, USA', '555-1234', 'Suejohnson@uon.edu.au', 'Active', 'No comments'),  ('STAFF2', 'Tom Brown', '456 Elm St, Anytown, USA', '555-5678', 'Tombrown@uon.edu.au', 'Active', 'No comments'),  ('STAFF3', 'Mary White', '789 Oak St, Anytown, USA', '555-9012', 'Marywhite@uon.edu.au', 'Active', 'No comments'),  ('STAFF4', 'Joe Black', '101 Pine St, Anytown, USA', '555-3456', 'Joeblack@uon.edu.au', 'Active', 'No comments'),  ('STAFF5', 'Sally Green', '111 Maple St, Anytown, USA', '555-7890', 'Sallygreen@uon.edu.au', 'Active', 'No comments');  -- Insert data into the Student table  INSERT INTO Student (studentID, major, name, address, phone, email, status, comments)  VALUES ('STUDENT1', 'Computer Science', 'John Doe', '123 Main St, Anytown, USA', '555-1234', 'johndoe@example.com', 'Active', 'No comments'),  ('STUDENT2', 'Mathematics', 'Jane Smith', '456 Elm St, Anytown, USA', '555-5678', 'Janesmith@example.com', 'Active', 'No comments'),  ('STUDENT3', 'Physics', 'Jim Brown', '789 Oak St, Anytown, USA', '555-9012', 'jimbrown@gmail.com', 'Active', 'No comments'),  ('STUDENT4', 'Biology', 'Jill White', '101 Pine St, Anytown, USA', '555-3456', 'jillwhite@gmail.com', 'Active', 'No comments'),  ('STUDENT5', 'Chemistry', 'Jack Black', '111 Maple St, Anytown, USA', '555-7890', 'Jackblack@example.com', 'Active', 'No comments');  INSERT INTO Staff (staffID, title, name, address, phone, email, status, comments)  VALUES ('STAFF1', 'Librarian', 'Sue Johnson', '123 Main St, Anytown, USA', '555-1234', 'Suejohnson@uon.edu.au', 'Active', 'No comments'),  ('STAFF2', 'Library Assistant', 'Tom Brown', '456 Elm St, Anytown, USA', '555-5678', 'Tombrown@uon.edu.au', 'Active', 'No comments'),  ('STAFF3', 'Library Assistant', 'Mary White', '789 Oak St, Anytown, USA', '555-9012', 'Marywhite@uon.edu.au', 'Active', 'No comments'),  ('STAFF4', 'Library Assistant', 'Joe Black', '101 Pine St, Anytown, USA', '555-3456', 'Joeblack@uon.edu.au', 'Active', 'No comments'),  ('STAFF5', 'Library Assistant', 'Sally Green', '111 Maple St, Anytown, USA', '555-7890', 'Sallygreen@uon.edu.au', 'Active', 'No comments');  INSERT INTO CourseOffering (offerID, cid, course, semester, year, dateBegin, dateEnd)  VALUES ('CO1', 'CSC101', 'Introduction to Computer Science', 'Spring', 2023, '2023-01-01', '2027-05-01'),  ('CO2', 'MAT101', 'Introduction to Mathematics', 'Spring', 2024, '2024-01-01', '2026-05-01'),  ('CO3', 'PHY101', 'Introduction to Physics', 'Spring', 2022, '2022-01-01', '2025-05-01'),  ('CO4', 'BIO101', 'Introduction to Biology', 'Spring', 2023, '2023-01-01', '2024-05-01'),  ('CO5', 'CHE101', 'Introduction to Chemistry', 'Spring', 2020, '2020-01-01', '2025-05-01');  INSERT INTO Resource (resourceID, name, status)  VALUES ('RES1', 'Book', 'Available'),  ('RES2', 'Journal', 'Available'),  ('RES3', 'DVD', 'Available'),  ('RES4', 'Laptop', 'Available'),  ('RES5', 'Camera', 'Available'),  ('RES6', 'Book', 'Available'),  ('RES7', 'Journal', 'Available'),  ('RES8', 'DVD', 'Available'),  ('RES9', 'Laptop', 'Available'),  ('RES10', 'Speaker', 'Available'),  ('RES11', 'Book', 'Available'),  ('RES12', 'Journal', 'Available'),  ('RES13', 'DVD', 'Available'),  ('RES14', 'Speaker', 'Available'),  ('RES15', 'Camera', 'Available'),  ('RES16', 'Room', 'Available'),  ('RES17', 'Room', 'Available'),  ('RES18', 'Room', 'Available'),  ('RES19', 'Room', 'Available'),  ('RES20', 'Room', 'Available');  INSERT INTO Movable (resourceID, manufacturer, model, year, assetValue)  VALUES ('RES1', 'Apple', 'MacBook Pro', 2020, 1000.00),  ('RES2', 'Dell', 'Inspiron', 2021, 800.00),  ('RES3', 'HP', 'Pavilion', 2019, 700.00),  ('RES4', 'Lenovo', 'ThinkPad', 2018, 600.00),  ('RES5', 'Canon', 'EOS 5D Mark IV', 2017, 500.00),  ('RES6', 'Apple', 'MacBook Pro', 2020, 1000.00),  ('RES7', 'Dell', 'Inspiron', 2021, 800.00),  ('RES8', 'HP', 'Pavilion', 2019, 700.00),  ('RES9', 'Lenovo', 'ThinkPad', 2018, 600.00),  ('RES10', 'Canon', 'EOS 5D Mark IV', 2017, 500.00),  ('RES11', 'Apple', 'MacBook Pro', 2020, 1000.00),  ('RES12', 'Dell', 'Inspiron', 2021, 800.00),  ('RES13', 'HP', 'Pavilion', 2019, 700.00),  ('RES14', 'Lenovo', 'ThinkPad', 2018, 600.00),  ('RES15', 'Canon', 'EOS 5D Mark IV', 2017, 500.00);  INSERT INTO Immovable (resourceID, capacity, room, building, campus)  VALUES ('RES16', 50, 'Room 1', 'Building A', 'Campus 1'),  ('RES17', 100, 'Room 1', 'Building B', 'Campus 2'),  ('RES18', 150, 'Room 1', 'Building C', 'Campus 3'),  ('RES19', 200, 'Room 2', 'Building D', 'Campus 4'),  ('RES20', 250, 'Room 3', 'Building E', 'Campus 5');  Insert INTO Acquisition (acqID, memberID, vendorCode, resourceID, price, dateTimeReceived, notes)  VALUES ('ACQ1', 'STUDENT1', 'VENDOR1', 'RES1', 50.00, '2023-01-01', 'No notes'),  ('ACQ2', 'STUDENT2', 'VENDOR2', 'RES2', 60.00, '2024-01-01', 'No notes'),  ('ACQ3', 'STUDENT3', 'VENDOR3', 'RES3', 70.00, '2022-01-01', 'No notes'),  ('ACQ4', 'STUDENT4', 'VENDOR4', 'RES4', 80.00, '2023-01-01', 'No notes'),  ('ACQ5', 'STUDENT5', 'VENDOR5', 'RES5', 90.00, '2020-01-01', 'No notes'),  ('ACQ6', 'STAFF1', 'VENDOR1', 'RES6', 50.00, '2023-01-01', 'No notes'),  ('ACQ7', 'STAFF2', 'VENDOR2', 'RES7', 60.00, '2024-01-01', 'No notes'),  ('ACQ8', 'STAFF3', 'VENDOR3', 'RES8', 70.00, '2022-01-01', 'No notes'),  ('ACQ9', 'STAFF4', 'VENDOR4', 'RES9', 80.00, '2023-01-01', 'No notes'),  ('ACQ10', 'STAFF5', 'VENDOR5', 'RES10', 90.00, '2020-01-01', 'No notes'),  ('ACQ11', 'STUDENT1', 'VENDOR1', 'RES11', 50.00, '2023-01-01', 'No notes'),  ('ACQ12', 'STUDENT2', 'VENDOR2', 'RES12', 60.00, '2024-01-01', 'No notes'),  ('ACQ13', 'STUDENT3', 'VENDOR3', 'RES13', 70.00, '2022-01-01', 'No notes');  INSERT INTO Category (Code, ResourceID, name, description, durationDays, durationHours)  VALUES ('CAT1', 'RES1', 'Speaker', 'Book Category', 30, 0),  ('CAT2', 'RES2', 'Speaker', 'Journal Category', 30, 0),  ('CAT3', 'RES3', 'DVD', 'DVD Category', 30, 0),  ('CAT4', 'RES4', 'Laptop', 'Laptop Category', 30, 0),  ('CAT5', 'RES5', 'Camera', 'Camera Category', 30, 0),  ('CAT6', 'RES6', 'Book', 'Book Category', 30, 0),  ('CAT7', 'RES7', 'Journal', 'Journal Category', 30, 0),  ('CAT8', 'RES8', 'DVD', 'DVD Category', 30, 0),  ('CAT9', 'RES9', 'Laptop', 'Laptop Category', 30, 0),  ('CAT10', 'RES10', 'Speaker', 'Speaker Category', 30, 0),  ('CAT11', 'RES11', 'Book', 'Book Category', 30, 0),  ('CAT12', 'RES12', 'Journal', 'Journal Category', 30, 0),  ('CAT13', 'RES13', 'DVD', 'DVD Category', 30, 0),  ('CAT14', 'RES14', 'Speaker', 'Speaker Category', 30, 0),  ('CAT15', 'RES15', 'Camera', 'Camera Category', 30, 0),  ('CAT16', 'RES16', 'Room', 'Room Category', 30, 0),  ('CAT17', 'RES17', 'Room', 'Room Category', 30, 0),  ('CAT18', 'RES18', 'Room', 'Room Category', 30, 0),  ('CAT19', 'RES19', 'Room', 'Room Category', 30, 0),  ('CAT20', 'RES20', 'Room', 'Room Category', 30, 0);  INSERT INTO Privilege (privilID, Code, name, description, maxItems)  VALUES ('PRIV1', 'CAT1', 'Student', 'Student Privilege', 5),  ('PRIV2', 'CAT2', 'Staff', 'Staff Privilege', 10),  ('PRIV3', 'CAT3', 'Student', 'Student Privilege', 5),  ('PRIV4', 'CAT4', 'Staff', 'Staff Privilege', 10),  ('PRIV5', 'CAT5', 'Student', 'Student Privilege', 5),  ('PRIV6', 'CAT6', 'Staff', 'Staff Privilege', 10),  ('PRIV7', 'CAT7', 'Student', 'Student Privilege', 5),  ('PRIV8', 'CAT8', 'Staff', 'Staff Privilege', 10),  ('PRIV9', 'CAT9', 'Student', 'Student Privilege', 5),  ('PRIV10', 'CAT10', 'Staff', 'Staff Privilege', 10),  ('PRIV11', 'CAT11', 'Student', 'Student Privilege', 5),  ('PRIV12', 'CAT12', 'Staff', 'Staff Privilege', 10),  ('PRIV13', 'CAT13', 'Student', 'Student Privilege', 5),  ('PRIV14', 'CAT14', 'Staff', 'Staff Privilege', 10),  ('PRIV15', 'CAT15', 'Student', 'Student Privilege', 5),  ('PRIV16', 'CAT16', 'Staff', 'Staff Privilege', 10),  ('PRIV17', 'CAT17', 'Student', 'Student Privilege', 5),  ('PRIV18', 'CAT18', 'Staff', 'Staff Privilege', 10),  ('PRIV19', 'CAT19', 'Student', 'Student Privilege', 5),  ('PRIV20', 'CAT20', 'Staff', 'Staff Privilege', 10);  INSERT INTO CourseOfferingToPrivilege (studentID, offerID, privilID)  VALUES ('STUDENT1', 'CO1', 'PRIV1'),  ('STUDENT2', 'CO2', 'PRIV3'),  ('STUDENT3', 'CO3', 'PRIV5'),  ('STUDENT4', 'CO4', 'PRIV7'),  ('STUDENT5', 'CO5', 'PRIV9');  INSERT INTO Loan(LoanID, memberID, resourceID, dateTimeBorrowed, dateTimeReturned, dateTimeDue)  VALUES ('LOAN1', 'STUDENT1', 'RES1', '2023-01-01', '2023-01-15', '2023-01-15'),  ('LOAN2', 'STUDENT2', 'RES2', '2024-01-01', '2024-01-15', '2024-01-15'),  ('LOAN3', 'STUDENT3', 'RES3', '2022-01-01', '2022-01-15', '2022-01-15'),  ('LOAN4', 'STUDENT4', 'RES4', '2023-01-01', '2023-01-15', '2023-01-15'),  ('LOAN5', 'STUDENT5', 'RES5', '2020-01-01', '2020-01-15', '2020-01-15'),  ('LOAN6', 'STAFF1', 'RES6', '2023-01-01', '2023-01-15', '2023-01-15'),  ('LOAN7', 'STAFF2', 'RES7', '2024-01-01', '2024-01-15', '2024-01-15'),  ('LOAN8', 'STAFF3', 'RES8', '2022-01-01', '2022-01-15', '2022-01-15'),  ('LOAN9', 'STAFF4', 'RES9', '2023-01-01', '2023-01-15', '2023-01-15'),  ('LOAN10', 'STAFF5', 'RES10', '2020-01-01', '2020-01-15', '2020-01-15'),  ('LOAN11', 'STUDENT1', 'RES11', '2023-01-01', '2023-01-15', '2023-01-15'),  ('LOAN12', 'STUDENT2', 'RES12', '2024-01-01', '2024-01-15', '2024-01-15'),  ('LOAN13', 'STUDENT3', 'RES13', '2022-01-01', '2022-01-15', '2022-01-15'),  ('LOAN14', 'STUDENT4', 'RES14', '2023-01-01', '2023-01-15', '2023-01-15'),  ('LOAN15', 'STUDENT5', 'RES15', '2024-01-01', '2024-01-30', '2024-01-30'),  ('LOAN16', 'STAFF1', 'RES16', '2024-05-01', '2023-05-10', '2023-01-15'),  ('LOAN17', 'STAFF2', 'RES17', '2024-06-05', '2024-06-10', '2024-01-15'),  ('LOAN18', 'STAFF3', 'RES18', '2022-09-19', '2022-09-30', '2022-01-15'),  ('LOAN19', 'STAFF4', 'RES19', '2023-01-01', '2023-01-15', '2023-01-15'),  ('LOAN20', 'STAFF5', 'RES20', '2020-01-01', '2020-01-15', '2020-01-15');  INSERT INTO Reservation (ReservationID, memberID, resourceID, dateTimeReserved, dateTimeDue)  VALUES ('RESV1', 'STUDENT1', 'RES1', '2023-01-01', '2023-01-15'),  ('RESV2', 'STUDENT2', 'RES2', '2024-01-01', '2024-01-15'),  ('RESV3', 'STUDENT3', 'RES3', '2022-01-01', '2022-01-15'),  ('RESV4', 'STUDENT4', 'RES4', '2023-01-01', '2023-01-15'),  ('RESV5', 'STUDENT5', 'RES5', '2020-01-01', '2020-01-15'),  ('RESV6', 'STAFF1', 'RES6', '2022-01-01', '2022-01-15'),  ('RESV7', 'STAFF2', 'RES7', '2024-01-01', '2024-01-15'),  ('RESV8', 'STAFF3', 'RES8', '2022-01-01', '2022-01-15'),  ('RESV9', 'STAFF4', 'RES9', '2023-01-01', '2023-01-15'),  ('RESV10', 'STAFF5', 'RES10', '2020-01-01', '2020-01-15'),  ('RESV11', 'STUDENT1', 'RES11', '2023-01-01', '2023-01-15'),  ('RESV12', 'STUDENT2', 'RES12', '2024-01-01', '2024-01-15'),  ('RESV13', 'STUDENT3', 'RES13', '2022-01-01', '2022-01-15'),  ('RESV14', 'STUDENT4', 'RES14', '2023-01-01', '2023-01-15'),  ('RESV15', 'STUDENT5', 'RES15', '2024-01-01', '2024-01-30'),  ('RESV16', 'STAFF1', 'RES16', '2024-05-01', '2023-05-10'),  ('RESV17', 'STAFF2', 'RES17', '2024-06-05', '2024-06-10'),  ('RESV18', 'STAFF3', 'RES18', '2024-09-19', '2024-09-30'),  ('RESV19', 'STAFF4', 'RES19', '2023-01-01', '2023-01-15'),  ('RESV20', 'STAFF5', 'RES20', '2020-01-01', '2020-01-15'); |

* 1. **SQL DATA ANALYSIS**

|  |
| --- |
| --Q1  SELECT s.name  FROM Student s  JOIN CourseOfferingToPrivilege cp ON s.studentID = cp.StudentID  JOIN CourseOffering c ON cp.offerID = c.offerID  WHERE c.cid = 'CSC101';  --Q2  SELECT p.maxItems AS MaxSpeakers  from Privilege p  join CourseOfferingToPrivilege cp on cp.privilID = p.privilID  join CourseOffering co on cp.offerID = co.offerID  join Student s on s.StudentID = cp.StudentID  join Category c on c.Code = p.Code  where co.cid='CSC101' and s.name ='John Doe' and c.name ='Speaker'  --Q3  SELECT m.name, m.phone, COUNT(\*) AS TotalReservations  FROM Member m  JOIN Reservation r ON m.memberID = r.memberID  WHERE m.memberID = 'STAFF1' AND YEAR(r.dateTimeReserved) = 2022  GROUP BY m.name, m.phone;  --Q4 --  SELECT s.name  FROM student s  JOIN loan l ON s.studentID = l.memberID  JOIN resource r ON l.resourceID = r.resourceID  JOIN movable mov ON r.resourceID = mov.resourceID  JOIN category c ON r.resourceID = c.resourceID  WHERE c.name = 'camera' AND mov.model = 'EOS 5D Mark IV' AND YEAR(l.dateTimeBorrowed)= YEAR(CURRENT\_TIMESTAMP);    --Q5 --  SELECT TOP 1 r.resourceID, r.name  FROM Reservation res  JOIN Movable mov ON res.resourceID = mov.resourceID  JOIN Resource r ON mov.resourceID = r.resourceID  WHERE YEAR(res.dateTimeReserved) = YEAR(GETDATE())  GROUP BY r.resourceID, r.name  ORDER BY COUNT(res.resourceID) DESC;  -- Q6  SELECT CONVERT(date, r.dateTimeReserved) AS Date, ro.room AS RoomName, COUNT(r.ReservationID) AS TotalReservations  FROM Reservation r  JOIN Immovable ro ON r.resourceID = ro.resourceID  WHERE ro.room = 'Room 1' AND CONVERT(date, r.dateTimeReserved) IN ('2024-05-01', '2024-06-05', '2024-09-19')  GROUP BY CONVERT(date, r.dateTimeReserved), ro.room; |