SDS Resource Management Database Design Project

By Patrick Holt

Assignment 3

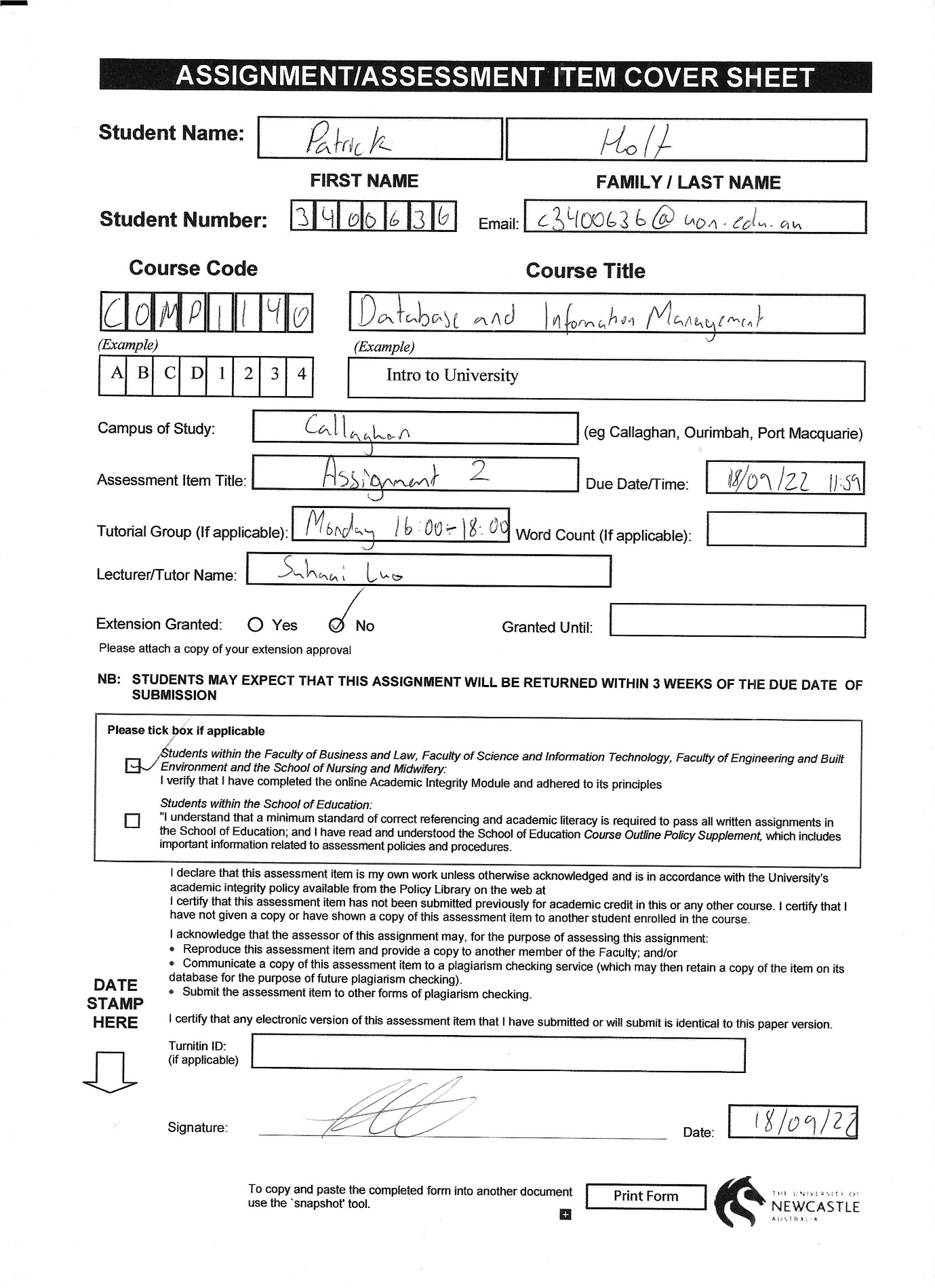


Table of Contents

[1. Reflection 5](#_Toc116934851)

[2. Requirement Specification 5](#_Toc116934852)

[2.1 Data Requirements 5](#_Toc116934853)

[2.1.1 Member 5](#_Toc116934854)

[2.1.2 Staff 5](#_Toc116934855)

[2.1.3 Student 6](#_Toc116934856)

[2.1.4 StudentCourseOffering 6](#_Toc116934857)

[2.1.5 CourseOfferingPrivilege 6](#_Toc116934858)

[2.1.6 Resource 6](#_Toc116934859)

[2.1.7 Moveable Resource 6](#_Toc116934860)

[2.1.8 Immovable Resource 6](#_Toc116934861)

[2.1.9 Location 6](#_Toc116934862)

[2.1.10 Category 7](#_Toc116934863)

[2.1.11 Loan 7](#_Toc116934864)

[2.1.12 Acquisition 7](#_Toc116934865)

[2.1.13 Reservation 7](#_Toc116934866)

[2.2 Transaction Requirements 7](#_Toc116934867)

[2.2.1.1 Data Manipulation Operations 7](#_Toc116934868)

[2.2.1.2 Data Queries 8](#_Toc116934869)

[2.3 Business Rules 8](#_Toc116934870)

[2.3.1 Student Member Rules 8](#_Toc116934871)

[2.3.2 Staff Member Rules 8](#_Toc116934872)

[2.3.3 Loaning rules 8](#_Toc116934873)

[2.3.4 StudentCourseOffering rules 8](#_Toc116934874)

[2.3.4 Reservation Rules 8](#_Toc116934875)

[3. EER Diagram 10](#_Toc116934876)

[4. Data Dictionary 11](#_Toc116934877)

[4.1 Entity Types 11](#_Toc116934878)

[4.2 Relationship Types 12](#_Toc116934879)

[4.3 Attributes 12](#_Toc116934880)

[5. Relation Model (Pre-Normalisation) 15](#_Toc116934881)

[5.1 Member 15](#_Toc116934882)

[5.2 Staff 15](#_Toc116934883)

[5.3 Student 16](#_Toc116934884)

[5.4 Reservation 16](#_Toc116934885)

[5.5 Loan 16](#_Toc116934886)

[5.6 Acquisition 16](#_Toc116934887)

[5.7 Moveable 17](#_Toc116934888)

[5.8 Immovable 17](#_Toc116934889)

[5.9 Resource 17](#_Toc116934890)

[5.10 Location 17](#_Toc116934891)

[5.11 Category 17](#_Toc116934892)

[5.12 CourseOfferingPrivilege 17](#_Toc116934893)

[5.13 StudentCourseOffering 18](#_Toc116934894)

[6. Normalised Relational Schema (DBDL) 18](#_Toc116934895)

[6.1 Member 18](#_Toc116934896)

[6.2 Staff 18](#_Toc116934897)

[6.3 Student 19](#_Toc116934898)

[6.4 Reservation 20](#_Toc116934899)

[6.5 Loan 20](#_Toc116934900)

[6.6 Acquisition 21](#_Toc116934901)

[6.7 Moveable 22](#_Toc116934902)

[6.8 Immovable 22](#_Toc116934903)

[6.9 Resource 23](#_Toc116934904)

[6.11 Category 23](#_Toc116934905)

[6.12 CourseOfferingPrivilege 24](#_Toc116934906)

[6.13 StudentCourseOffering 24](#_Toc116934907)

[6.14 Conclusion 25](#_Toc116934908)

[6.15 Boyce-Codd form conversion Example 25](#_Toc116934909)

[6.15.1 Resource BCNF Conversion 25](#_Toc116934910)

[6.15.1 StudentCourseOffering BCNF Conversion 27](#_Toc116934911)

[7. SQL Script 28](#_Toc116934912)

[7.1 Normalised SDS Database 28](#_Toc116934913)

[GO8.SQL Statements 31](#_Toc116934914)

[8.1 SQL Proper Data 31](#_Toc116934915)

[8.2 SQL Queries 33](#_Toc116934916)

[8.2.1 Query 1 33](#_Toc116934917)

[8.2.2 Query 2 33](#_Toc116934918)

[8.2.3 Query 3 33](#_Toc116934919)

[8.2.4 Query 4 33](#_Toc116934920)

[8.2.5 Query 5 33](#_Toc116934921)

[8.2.6 Query 6 33](#_Toc116934922)

# Reflection

For assignment 2, I greatly improved my overall score from 3/15 to 13.8/15. However, I knew I would still get things wrong. This time, I will solely use the feedback to make improvements based on the marker’s comments:

* “No relation between acquisition and resource needed”
* Removed relation between acquisition and resource in the relationship types table and in EER diagram
* Relationship type table mistake with {mandatory, or}
* Fixed this using recommendation from feedback
* Location attributes and relational model deleted
* Removed Location entity throughout assignment and added some attributes to resource entity to compensate
* resourceID should be foreign key for Immovable and Moveable
* Added resourceID as a foreign key for Immovable and Moveable
* Resource needs category FK
* Added categoryID FK to Resource
* Loan needs Moveable FK
* Added moveableID FK to Loan
* Missed Tables (5 for each table - StudentCourseOffering, CourseOfferingPrivilege)
* Replaced names of tables with names/ideas given in instructions
* Unnecessary Foreign Keys
* Removed all unnecessary foreign keys
* Two examples of normalization up to BCNF (9 for each example)

# Requirement Specification

## Data Requirements

### 2.1.1 Member

A member is a person in the database and is the superclass that contains two subclasses, student, and class (discussed in their respective data requirements section). Therefore, to become a member, you must be a staff member, or be enrolled in a course. Members can make acquisitions for new resources, reservations for resources, and loan resources using the databases search and acquisition functions. The data stored for members are their unique identification code (memberID), their name (name), their address (address), their phone number (phone), their email (email), any additional comments for their profile (comments), and their status (status), i.e. “active” or “inactive” members (defaulted to “active”).

### 2.1.2 Staff

Staff are workers of the university in the database and are a subclass of the superclass, member. Staff members have priority over acquisitions, reservations, and loans and can borrow as many resources as they acquire, however, they should try to return resources on time to allow other staff to use the resources. They will not be punished if they are overdue on their return, but staff should be made aware of their expectation to return items on time. The data uniquely stored for staff members is their job roles (role). As staff is a subclass of the superclass member, it also contains the data stored under member.

### 2.1.3 Student

Students a people enrolled in a course at the university in the database and are a subclass of the superclass, member. Student members have course offerings which give them privileges to borrow a category of resources for their study/assignments. The data stored uniquely for student members are the amount of resource borrowed (borrowed) that is limited to the privilege’s maximum borrowing capacity, total money owed (totalOwed) in the event a resource is overdue (defaulted to “$0.00”), and the status of the student (studentStatus) i.e., “Disabled” if any amount is owed and “Enabled” if no amount is owed, defaulted to “Enabled”. As student is a subclass of the superclass member, it also contains the data stored under member.

### 2.1.4 StudentCourseOffering

A course offer is what students enrol in which give them privileges on certain categories of resources. The data stored for course offerings are the course offer identification code (offerID), the start date of the course offer (dateStart), and the end date of the course offer(dateEnd). Adding the course offer dates limit the privileges of student members to these time frames. A resources due date can’t be longer than the end date of the course offer.

### 2.1.5 CourseOfferingPrivilege

A privilege is a student’s right to use the database to reserve, acquisition, and loan certain resources. This is determined by the courses they are enrolled in. The data stored for privileges are a unique identification code (privID), a detailed description of what the student is allowed to borrow and why (privilegeDescription), and a maximum borrowing capacity (borrowCapacity) which determines the student’s maximum allowed resources.

### 2.1.6 Resource

Resources are the main entity being interacted with by members in the database and is a superclass that contains two subclasses, moveable and immovable (discussed in their respective data requirements section). The resource belongs to a category based on what the resource is and resources also have a location it belongs to. The data uniquely stored for resources includes a unique identification code (resourceID), a short description of the item (resourceDescription), and its status (status) i.e. “Available”, “Unavailable”, “Reserved”, “Damaged”, or “Lost” (defaulted to (“Available”).

### 2.1.7 Moveable Resource

Moveable is a type of resource that is not a room, or other immovable resources (if any), and is a subclass of the superclass, resource. Moveable resources can be loaned, reserved, and acquisitioned by student members and staff members. The data uniquely stored under moveable includes its name (name), its manufacturer (manufacturer), its model (model), its year (year), and the value of the asset (assetValue). As moveable is a subclass of the superclass resource, it also contains the data stored under resource.

### 2.1.8 Immovable Resource

Immovable is a type of resource that can’t be moved, such as a room, and is a subclass of the superclass, resource. Immovable resources can be loaned, reserved, and acquisitioned by student members and staff members. The data uniquely stored for immovable resources are the maximum person capacity of the room (capacity). As immoveable is a subclass of the superclass resource, it also contains the data stored under resource.

### 2.1.9 Category

Category places resources into specific tags that can make them easier to search for in the database. The data stored for category is a category identification code (categoryID), a name of the category (name), a description of the category (categoryDescription), maximum loan time in days (durationDays), and maximum loan time in hours (durationHours).

### 2.1.10 Loan

Loans are an action that can be made by staff members or student members to acquire resources for a given amount of time. Staff members are given priority over resource loans. The data stored for loans are a loan identification code (loanID), the date and time the item was borrowed (dateTimeBorrowed), the date and time the item was returned (dateTimeReturned), and the date and time the resource is due (dateTimeDue) based upon the resource’s category’s maximum loan time.

### 2.1.11 Acquisition

Acquisitions are an action that can made by staff members or student members to acquire new resources for the database. Staff members are given priority over acquisitions. The data stored for acquisitions are an acquisition identification code (acquisitionID), acquisitioned resource’s name (name), acquisitioned resource’s manufacturer (manufacturer), acquisitioned resource’s model (model), acquisitioned resource’s year of model (year), acquisitioned resource’s price (price), reasoning for the acquisition (reasoning), and urgency for the item (urgency), defaulted to “Not Urgent”.

### 2.1.12 Reservation

Reservations are an action that can be made by staff members and student members to reserve resources for a given date and time. The data stored for reservations are a reservation identification code (reservationID), and the date and time the resource is reserved for (reservedDateTime).

## Transaction Requirements

### Data Manipulation Operations

* Insert/Update/Delete an immovable resource and their information
* Insert/Update/Delete a movable resource and their information
* Insert/Update/Delete a category and its information
* Insert/Update/Delete members (students or staff) and their information
* Insert/Update/Delete a loan and its information
* Insert/Update/Delete an acquisition
* Insert/Update/Delete a reservation
* Insert/Update/Delete a course offering
* Insert/Update/Delete a course offering privilege
* Insert/Update/Delete a location and its information

### Data Queries

* Search for a moveable resource based on a resourceID, locationID, name, manufacturer, model, year, asset value, description, status, campus, building, or room.
* Search for an immovable resource based on resourceID, capacity, description, status, campus, building, or room
* Search for a resource based on categories
* List a staff/student member’s details
* List all staff/student member’s loaned resources and if they are overdue
* List all staff/student member’s reserved resources
* List all staff/student member’s acquisitions
* List all loaned resources and who they are loaned by
* List all non-loaned resources
* List all reserved resources and who they are reserved by
* List all non-reserved resources
* List all acquisitions and who acquisitioned them
* List all overdue items

## Business Rules

### 2.3.1 Student Member Rules

1. If a student member has money owing, or an overdue resource, their status changes to “Disabled” and they are unable to borrow more resources until they have returned the item or they have paid off their fees.

### 2.3.2 Staff Member Rules

1. Staff members can reserve, loan, and acquire as many resources as they require.

2. Staff members should return resources on the due date or as soon as possible after the due date

3. Staff members should provide notice if they are unable to return a resource at the due date

4. Staff members have priority over student members

### 2.3.3 Loaning rules

1. Immovable resources can’t move and, therefore, can’t be loaned.

2. Loaning time is determined by the resource’s category.

### 2.3.4 StudentCourseOffering rules

1. Student members are unable to loan or reserve resources until they are enrolled in a course

2. Student members are unable to loan or reserve a resource they aren’t within the start and end dates of the course offer

3. They must return any borrowed resource after they are outside the start and end dates of the course offer

### 2.3.4 Reservation Rules

1. A reserved resource must be picked up on the reserved date or their reservation will be cancelled, and the resource will become “Available”.

2. Students can incur a fee if they do not pick up the resource without sufficient notice.

3. Reservations can’t take place within the same date and time.

# EER Diagram

# Data Dictionary

## Entity Types

|  |  |  |  |
| --- | --- | --- | --- |
| Entity Name | Description | Aliases | Occurrence |
| Member | Describes staff or students | Staff, student, borrower | Inserted when a staff is hired, or a student enrols and used when a they acquires/loans/reserves a resource |
| Staff | Provides additional information on staff | Member, borrower | Inserted when a staff member is hired and used when they acquires/loan/reserves a resource |
| Student | Provides additional information on students | Member, borrower | Inserted when a student member enrols and used when they acquires/loan/reservation |
| Reservation | Allows staff/student members to reserve resources | Reserve | Inserted when staff or students reserves a moveable/immovable resource |
| Loan | Allows borrowing of resources | Borrow | Inserted when a student member or staff member loans a resource |
| Acquisition | Allows acquiring of new resources | Acquire | Inserted when a student or staff member |
| MoveableResource | Includes cameras, speakers, software, phones, and other moveable resources | Camera, speaker, software, or phone | Is loaned, borrowed, or reserved by student members or staff members. Inserted when acquired. |
| ImmovableResource | Includes rooms | Rooms | Is loaned, borrowed, or reserved by student/staff members. Inserted when built. |
| Category | Provides categories for resources that are similar, allows for easy searching | Camera, speaker, software, or phone | Resources inserted are given a category. Used in searched by staff members or student members. |
| CourseOfferingPrivilege | Given to student members to allow access to certain category of resources | Access, privilege | Inserted when students are given access to certain category of resources |
| StudentCourseOffering | Describes a course offer a student is enrolled in at the university | Course, course offer, class, subject | Inserted when students are offered enrolment in a course which gives them privileges to certain resources. |

## Relationship Types

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Staff | Man, Or | Generalisation | Man, Or | Member |
| Student | Man, Or | Generalisation | Man, Or | Member |
| 0..\* | Enrols in | 0..\* | StudentCourseOffering |
| Staff Member | 1..1 | Makes | 0..\* | Acquisition |
| 1..1 | Has | 0..\* | Loan |
| 1..1 | Has | 0..\* | Reservation |
| Student Member | 1..1 | Makes | 0..\* | Acquisition |
| 1..1 | Has | 0..\* | Loan |
| 1..1 | Has | 0..\* | Reservation |
| Loan | 0..\* | To | 1..1 | Moveable Resource |
| 0..\* | To | 1..1 | Immovable Resource |
| Reservation | 0…\* | For | 1..1 | Moveable Resource |
| 0..\* | For | 1..1 | Immovable Resource |
| Moveable Resource | 0..\* | Belongs To | 1..1 | Category |
| Immoveable Resource | 0..\* | Belongs To | 1..1 | Category |
| StudentCourseOffering | 0..\* | Has | 0..\* | CourseOfferingPrivilege |
| CourseOfferingPrivilege | 0..\* | For | 1..1 | Category |

## Attributes

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity Name | Attributes | Description | Data Type Length | Null | Multi-valued | Derived | Default |
| Staff | role | Job role of the staff member | Varchar (40) | N | N | N |  |
| Student | borrowed | Number of borrowed resources of the student | Int | N | N | N | “0” |
| totalOwed | Amount owed for overdue resources | Varchar (8) | Y | N | N |  |
| studentStatus | Status of the student | Varchar (8) | N | N | N | “Enabled” |
| Member | memberID | Identification code of the member | Varchar (10) | N | N | N |  |
| name | Name of the member | Varchar (30) | N | N | N |  |
| address | Address of the member | Varchar (50) | Y | N | N |  |
| phone | Phone number of the member | Varchar (13) | Y | N | N |  |
| email | Email address of the member | Varchar (30) | Y | N | N |  |
| status | Status of the member | Varchar (10) | N | N | N | “Active” |
| comments | Additional comments for the member | Varchar (150) | Y | N | N |  |
| Loan | loanID | Identification code of a loan | Varchar (10) | N | N | N |  |
| dateTimeBorrowed | Date and time the resource was borrowed | Datetime | N | N | N |  |
| dateTimeReturned | Date and time the resource was returned | Datetime | Y | N | N |  |
| dateTimeDue | Date and time the resource is due | Datetime | N | N | N |  |
| Moveable | name | Name of the resource | Varchar (40) | N | N | N |  |
| manufacturer | Manufacturer of the resource | Varchar (25) | N | N | N |  |
| model | Model of the resource | Varchar (40) | N | N | N |  |
| year | Year of the resource | Int | N | N | N |  |
| assetValue | Value of the resource | Varchar (8) | N | N | N |  |
| Immovable | capacity | Resource’s maximum capacity | Int | N | N | N |  |
| Resource | resourceID | Identification code of the resource | Varchar (10) | N | N | N |  |
| resourceDescription | Description of the resource | Varchar (100) | Y | N | N |  |
| resourceStatus | Status of the resource | Varchar (11) | N | N | N | “Available” |
| campus | Campus a resource is located | Varchar (30) | N | N | N | “SDS” |
| building | Building a resource is located | Varchar (30) | N | N | N | “BSD” |
| room | Room a resource is located | Varchar (25) | N | N | N | “School Resource Centre” |
| Category | categoryID | Identification code of the category | Varchar (10) | N | N | N |  |
| name | Name of the category | Varchar (20) | N | N | N |  |
| categoryDescription | Description of the contents of the category | varchar (100) | N | N | N |  |
| durationDays | Days in which a category can be loaned | int | N | N | N |  |
| durationHours | Hours in which a category can be loaned | int | N | N | N |  |
| CourseOfferingPrivilege | privID | Identification code of the privilege | Varchar (10) | N | N | N |  |
| privilegeDescription | Description of the | Varchar (100) | Y | N | N |  |
| borrowCapacity | Maximum capacity of resources a student member can borrow | int | N | N | N | “3” |
| StudentCourseOffering | offerID | Identification code of the course offer | Varchar (10) | N | N | N |  |
| courseName | Name of the course | Varchar (40) | N | N | N |  |
| dateStart | Start date of course offer/privilege | Datetime | N | N | N |  |
| dateEnd | End date of course offer/privilege | Datetime | N | N | N |  |
| Acquisition | acquisitionID | Identification code of the acquisition | Varchar (10) | N | N | N |  |
| name | Name of the resource | Varchar (40) | N | N | N |  |
| manufacturer | manufacturer of the resource | Varchar (25) | N | N | N |  |
| model | Model of the resource | Varchar (40) | N | N | N |  |
| year | year of the model of the resource | Int | N | N | N |  |
| price | Price of the resource | Varchar (8) | N | N | N |  |
| reasoning | Reasoning for purchasing this resource | Varchar (100) | N | N | N |  |
| urgency | Urgency of purchasing this resource | Varchar (10) | N | N | N | “Not Urgent” |
| Reservation | reservationID | Identification code of a reservation | Varchar (10) | N | N | N |  |
| reservedForDateTime | When the resource is reserved for | Datetime | N | N | N |  |

# Relation Model (Pre-Normalisation)

## 5.1 Member

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

**Primary Key** memberID

## 5.2 Staff

Staff (memberID, role)

**Primary Key** memberID

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

**ON UPDATE CASCADE, ON DELETE CASCADE**

## 5.3 Student

Student (memberID, borrowed, totalOwed, studentStatus, offerID)

**Primary Key** memberID

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

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** offerID references StudentCourseOffering(offerID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

## 5.4 Reservation

Reservation (reservationID, reservedForDateTime, memberID, resourceID)

**Primary Key** reservationID

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

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** resourceID references Resource(resourceID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

## 5.5 Loan

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

**Primary Key** loanID

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

**ON UPDATE CASCADE, ON DELETE NO ACTION**

**Foreign Key** resourceID references Resource(resourceID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

## 5.6 Acquisition

Acquisition (acquisitionID, name, manufacturer, model, year, price, reasoning, urgency, memberID)

**Primary Key** acquisitionID

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

**ON UPDATE CASCADE, ON DELETE NO ACTION**

## 5.7 Moveable

Moveable (resourceID, name, manufacturer, model, year, assetValue)

**Primary Key** resourceID

**Foreign Key** resourceID references Resource(resourceID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

## 5.8 Immovable

Immovable (resourceID, capacity)

**Primary Key** resourceID

**Foreign Key** resourceID references Resource(resourceID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

## 5.9 Resource

Resource (resourceID, resourceDescription, resourceStatus, campus, building, room, categoryID)

**Primary Key** resourceID

**Foreign Key** categoryID references category(categoryID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

## 5.10 Category

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

**Primary Key** categoryID

## 5.11 CourseOfferingPrivilege

CourseOfferingPrivilege (privID, courseID, borrowCapacity, categoryID)

**Primary Key** privID

**Foreign Key** categoryID references Category(categoryID)

## 5.12 StudentCourseOffering

StudentCourseOffering (offerID, courseName, dateStart, dateEnd)

**Primary Key** offerID

# Normalised Relational Schema (DBDL)

All schemas ARE NOT in 1st normal form because they as they all have primary keys, contain single values, and are atomic.

## 6.1 Member

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

**Primary Key** memberID

Functional Dependency

* memberID  name, address, phone, email, status, comments

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Member is in Boyce-Codd Form

## 6.2 Staff

Staff (memberID, role)

**Primary Key** memberID

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

**ON UPDATE CASCADE, ON DELETE CASCADE**

Functional Dependency

* memberID  role

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Staff is in Boyce-Codd Form

## 6.3 Student

Student (memberID, borrowed, totalOwed, studentStatus, offerID)

**Primary Key** memberID

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

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** offerID references StudentCourseOffering(offerID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

Functional Dependency

* memberID  borrowed, totalOwed, studentStatus, offerID

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Student is in Boyce-Codd Form

## 6.4 Reservation

Reservation (reservationID, reservedForDateTime, memberID, resourceID)

**Primary Key** reservationID

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

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** resourceID references Resource(resourceID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

Functional Dependency

* reservationID, memberID, resourceID  reservedForDateTime

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Reservation is in Boyce-Codd Form

## 6.5 Loan

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

**Primary Key** loanID

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

**ON UPDATE CASCADE, ON DELETE NO ACTION**

**Foreign Key** resourceID references Resource(resourceID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

Functional Dependency

* loanID, memberID  dateTimeBorrowed, dateTimeReturned, dateTimeDue

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Loan is in Boyce-Codd Form

## 6.6 Acquisition

Acquisition (acquisitionID, name, manufacturer, model, year, price, reasoning, urgency, memberID)

**Primary Key** acquisitionID

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

**ON UPDATE CASCADE, ON DELETE NO ACTION**

Functional Dependency

* acquisitionID, memberID  name, manufacturer, model, year, price, reasoning, urgency

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Acquisition is in Boyce-Codd Form

## 6.7 Moveable

Moveable (resourceID, name, manufacturer, model, year, assetValue)

**Primary Key** resourceID

**Foreign Key** resourceID references Resource(resourceID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

Functional Dependency

* resourceID  name, manufacturer, model, year, assetValue, description, resourceStatus

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Moveable is in Boyce-Codd Form

## 6.8 Immovable

Immovable (resourceID, capacity)

**Primary Key** resourceID

**Foreign Key** resourceID references Resource(resourceID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

Functional Dependency

* resourceID  capacity, description, resourceStatus

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Immovable is in Boyce-Codd Form

## 6.9 Resource

Resource (resourceID, resourceDescription, resourceStatus, campus, building, room, categoryID)

**Primary Key** resourceID

**Foreign Key** categoryID references category(categoryID)

**ON UPDATE CASCADE, ON DELETE NO ACTION**

Functional Dependency

* resourceID  description, resourceStatus, campus, building, room, categoryID

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Resource is in Boyce-Codd Form

## 6.11 Category

Category (categoryID, name, categoryDescription, durationDays, durationHours)

**Primary Key** categoryID

Functional Dependency

* categoryID, resourceID  name, categoryDescription, durationDays, durationHours

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Category is in Boyce-Codd Form

## 6.12 CourseOfferingPrivilege

CourseOfferingPrivilege (privID, categoryID, privilegeDescription, borrowCapacity)

**Primary Key** privID

**Foreign Key** categoryID references Category(categoryID)

Functional Dependency

* privID, offerID  privilegeDescription, borrowCapacity

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* CourseOfferingPrivilege is in Boyce-Codd Form

## 6.13 StudentCourseOffering

StudentCourseOffering (offerID, courseName, dateStart, dateEnd)

**Primary Key** offerID

Functional Dependency

* offerID, memberID  dateStart, dateEnd

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* StudentCourseOffering is in Boyce-Codd Form

## 6.14 Conclusion

|  |  |
| --- | --- |
| **Relation** | **Form** |
| Member | Boyce-Codd Form |
| Staff | Boyce-Codd Form |
| Student | Boyce-Codd Form |
| Reservation | Boyce-Codd Form |
| Loan | Boyce-Codd Form |
| Acquisition | Boyce-Codd Form |
| Moveable | Boyce-Codd Form |
| Immovable | Boyce-Codd Form |
| Resource | Boyce-Codd Form |
| Location | Boyce-Codd Form |
| Category | Boyce-Codd Form |
| CourseOfferingPrivilege | Boyce-Codd Form |
| StudentCourseOffering | Boyce-Codd Form |

## 

## 6.15 BCNF conversion Example

In these examples, I will be making assumptions of functional dependencies in Resource relation and student

### 6.15.1 Resource BCNF Conversion

Resource (resourceID, resourceDescription, resourceStatus, campus, building, room)

**Primary Key** resourceID

Functional Dependency

* resourceID  resourceDescription, resourceStatus
* campus  building
* building  room

2nd Normal Form

* In 1st form
* Contains partial dependencies
* Therefore, needs to be changed

I did this by removing the partial dependencies. Moving campus, building, and room into another table allowed the removal of partial dependencies leaving:

Resource (resourceID, resourceDescription, resourceStatus)

**Primary Key** resourceID

Functional Dependency

* resourceID  resourceDescription, resourceStatus

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Resource is in Boyce-Codd Form

Then the new relation, Location is:

Location (campus, building, room)

Functional Dependency

* campus  building
* building  room

1st Normal Form

* Is not uniquely identifiable as no primary key
* Therefore in 1st form

By adding a unique identifier as a primary key, we get:

Location (locationID, campus, building, room)

**Primary Key** locationID

Functional Dependency

* locationID  campus, building, room

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Location is in Boyce-Codd Form

### 6.15.2 Student BCNF Conversion

As above, I have made assumptions that there are unnecessary functional dependencies in the Student entity that don’t allow it to be in BCNF form. In this example, we assume that when a student’s status is disabled a payment is owed and the account can only become enabled when that money is paid. This is shown below:

Student (memberID, borrowed, totalOwed, studentStatus, offerID)

**Primary Key** memberID

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

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** offerID references StudentCourseOffering(offerID)

Functional Dependency (with assumptions):

* memberID borrowing, studentStatus, offerID)
* studentStatus totalOwed

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains partial dependencies and is therefore not able to move to 2nd Normal form
* partial dependency:
* studentStatus totalOwed

Because there is a partial dependency, we must change Student relation by creating a status relation and moving the partial dependency attributes there whist making student status a composite key:

Student (memberID, borrowed, offerID, statusID)

**Primary Key** memberID

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

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** offerID references StudentCourseOffering(offerID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** statusID references Status(statusID)

**ON UPDATE CASCADE, ON DELETE CASCADE**

Functional Dependency

* memberID  borrowed, offerID, statusID

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Student is in Boyce-Codd Form

And the new relation is:

Status (studentStatus, totalOwed)

Functional Dependency

* studentStatus  totalOwed

1st Normal Form

* Is not uniquely identifiable as no primary key
* Therefore in 1st form

By adding a unique identifier as a primary key, we get:

Status (statusID, studentStatus, totalOwed)

**Primary Key** statusID

Functional Dependency

* statusID, studentStatus  totalOwed

1st Normal Form

* All rows are uniquely identifiable
* Each cell contains a single value
* Each value is atomic

2nd Normal Form

* In 1st form
* Contains no partial dependencies

3rd Normal Form

* In 1st, and 2nd form
* Contains no transitive dependencies

Boyce-Codd Normal Form

* In 1st, 2nd, and 3rd form
* Every functional dependency is a candidate key

Therefore:

* Status is in Boyce-Codd Form

Therefore, both relations are now in BCFN form.

# SQL Script

## 7.1 Normalised SDS Database

DROP TABLE Staff

DROP TABLE Student

DROP TABLE Moveable

DROP TABLE Immovable

DROP TABLE Loan

DROP TABLE CourseOfferingPrivilege

DROP TABLE StudentCourseOffering

DROP TABLE Acquisition

DROP TABLE Reservation

DROP TABLE Member

DROP TABLE Resource

DROP TABLE Category

DROP DATABASE SCS

CREATE DATABASE SCS

CREATE TABLE Member(

memberID Varchar(10) NOT NULL,

name Varchar(30) NOT NULL,

address Varchar(50),

phone Varchar(13),

email Varchar(30),

status Varchar(10) default 'Active' CHECK (status IN ('Active', 'Inactive')) NOT NULL,

comments Varchar(150),

PRIMARY KEY (memberID),

);

GO

CREATE TABLE Staff(

memberID Varchar(10) NOT NULL,

role Varchar(40) NOT NULL,

PRIMARY KEY (memberID),

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

CREATE TABLE StudentCourseOffering(

offerID Varchar(10) NOT NULL,

courseName Varchar(40) NOT NULL,

dateStart Datetime NOT NULL,

dateEnd Datetime NOT NULL,

PRIMARY KEY (offerID),

);

GO

CREATE TABLE Student(

memberID Varchar(10) NOT NULL,

borrowed Int NOT NULL default '0',

totalOwed Varchar(8),

studentStatus Varchar(8) default 'Enabled' CHECK (studentStatus IN ('Enabled', 'Disabled')) NOT NULL,

offerID Varchar(10) NOT NULL,

PRIMARY KEY (memberID),

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON UPDATE CASCADE ON DELETE NO ACTION,

FOREIGN KEY (offerID) REFERENCES StudentCourseOffering(offerID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

GO

CREATE TABLE Category(

categoryID Varchar(10) NOT NULL,

name Varchar(20) NOT NULL,

categoryDescription Varchar(100) NOT NULL,

durationDays Int NOT NULL,

durationHours Int NOT NULL,

PRIMARY KEY (categoryID),

);

GO

CREATE TABLE Resource(

resourceID Varchar(10) NOT NULL,

categoryID Varchar(10) NOT NULL,

resourceDescription Varchar(100),

resourceStatus Varchar(11) NOT NULL default 'Available' CHECK (resourceStatus IN ('Available', 'Occupied', 'Damaged')),

campus Varchar(30) default 'SDS' NOT NULL,

building Varchar(30) default 'BSD' NOT NULL,

room Varchar(25) default 'School Resource Centre' NOT NULL,

PRIMARY KEY (resourceID),

FOREIGN KEY (categoryID) REFERENCES Category(categoryID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

GO

CREATE TABLE Moveable(

resourceID Varchar(10) NOT NULL,

name Varchar(40) NOT NULL,

manufacturer Varchar(25) NOT NULL,

model Varchar(40) NOT NULL,

year Int NOT NULL,

assetValue Varchar(8) NOT NULL,

PRIMARY KEY (resourceID),

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

GO

CREATE TABLE Immovable(

resourceID Varchar(10) NOT NULL,

capacity Int NOT NULL,

PRIMARY KEY (resourceID),

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

GO

CREATE TABLE Loan(

loanID Varchar(20) NOT NULL,

dateTimeBorrowed Datetime NOT NULL,

dateTimeReturned Datetime,

dateTimeDue Datetime NOT NULL,

memberID Varchar(10) NOT NULL,

resourceID Varchar(10) NOT NULL,

PRIMARY KEY (loanID),

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON UPDATE CASCADE ON DELETE NO ACTION,

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

GO

CREATE TABLE CourseOfferingPrivilege(

privID Varchar(10) NOT NULL,

privilegeDescription Varchar(100),

borrowCapacity Int default '3' NOT NULL,

categoryID Varchar(10) NOT NULL,

offerID Varchar(10) NOT NULL,

PRIMARY KEY (privID),

FOREIGN KEY (categoryID) REFERENCES Category(categoryID) ON UPDATE CASCADE ON DELETE NO ACTION,

FOREIGN KEY (offerID) REFERENCES StudentCourseOffering(offerID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

GO

CREATE TABLE Acquisition(

acquisitionID Varchar(10) NOT NULL,

name Varchar(40) NOT NULL,

manufacturer Varchar(25) NOT NULL,

model Varchar(40) NOT NULL,

year Int NOT NULL,

price Varchar(8) NOT NULL,

reasoning Varchar(100) NOT NULL,

urgency Varchar(10) NOT NULL default 'Not Urgent' CHECK (urgency IN ('Not Urgent', 'Urgent')),

memberID Varchar(10) NOT NULL,

PRIMARY KEY (acquisitionID),

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

GO

CREATE TABLE Reservation(

reservationID Varchar(10) NOT NULL,

reservedForDateTime datetime NOT NULL,

memberID Varchar(10) NOT NULL,

resourceID Varchar(10) NOT NULL,

PRIMARY KEY (reservationID),

FOREIGN KEY (memberID) REFERENCES Member(memberID) ON UPDATE CASCADE ON DELETE NO ACTION,

FOREIGN KEY (resourceID) REFERENCES Resource(resourceID) ON UPDATE CASCADE ON DELETE NO ACTION,

);

# GO8.SQL Statements

## SQL Proper Data

INSERT INTO StudentCourseOffering(offerID, courseName, dateStart, dateEnd) values('10100-3399', 'Introduction To Photography', '2021-12-10 00:00:01', '2022-12-10 00:00:01')

INSERT INTO StudentCourseOffering(offerID, courseName, dateStart, dateEnd) values('10100-2135', 'Introduction To Sound Design', '2021-12-10 00:00:01', '2022-12-10 00:00:01')

INSERT INTO StudentCourseOffering(offerID, courseName, dateStart, dateEnd) values('10100-4890', 'Software Engineering Fundamentals', '2021-12-10 00:00:01', '2022-12-10 00:00:01')

INSERT INTO StudentCourseOffering(offerID, courseName, dateStart, dateEnd) values('10100-6011', 'Primary School Teaching', '2021-12-10 00:00:01', '2022-12-10 00:00:01')

INSERT INTO Member(memberID, name, address, phone, email, comments) values('9400-61910', 'Johnny Knox', '590/12 Booket Pl', '0947521700', 'johnnyknox123@gmail.com', 'Failed to return resource on time 11/09/2020')

INSERT INTO Member(memberID, name, address, phone, email, comments) values('9400-82257', 'Samual Web', '120 Smith St', '0911844292', 'sammy2241@hotmail.com', 'Returned a resource damaged, willing to pay so can continue to borrow')

INSERT INTO Member(memberID, name, address, phone, email, comments) values('9400-29537', 'Rebecca Tyka', '17 Myst St', '0933582001', 'tykatyka1212@yahoo.com', 'Good standing')

INSERT INTO Member(memberID, name, address, phone, email) values('9400-68021', 'Alanna Simmons', '31 Kira Cl', '0998627450', 'asimmy@gmail.com')

INSERT INTO Member(memberID, name, address, phone, email, comments) values('9400-28461', 'Mick Burkett', '12 Silly St', '0922893811', 'm.burkett1@sds.edu.au.com', 'Requires extra time on some resources')

INSERT INTO Member(memberID, name, address, phone, email) values('9400-89211', 'Bill Sader', '21/50 Buckingwood Pl', '0935938100', 'b.sader@sds.edu.au.com')

INSERT INTO Member(memberID, name, address, phone, email, comments) values('9400-96032', 'Shaun Ying', '3 Sizzle Cl', '0931012186', 'shaun.ying@sds.edu.au.com', 'Need to update phone number, Likes to reserve the same room on Tuesdays')

INSERT INTO Member(memberID, name, address, phone, email) values('9400-69346', 'Keeley Skarsgard', '22 Meak St', '0922483302', 'kskars@sds.edu.au.com')

INSERT INTO Staff(MemberID, role) values('9400-28461', 'Senior Photography Coordinator')

INSERT INTO Staff(MemberID, role) values('9400-89211', 'Music Professor')

INSERT INTO Staff(MemberID, role) values('9400-96032', 'Tutor')

INSERT INTO Staff(MemberID, role) values('9400-69346', 'Information Technology Professor')

INSERT INTO Student(MemberID, totalOwed, studentStatus, offerID) values('9400-61910', '$12.50', 'Disabled', '10100-3399')

INSERT INTO Student(MemberID, totalOwed, offerID) values('9400-82257', '$59.99', '10100-2135')

INSERT INTO Student(MemberID, borrowed, offerID) values('9400-29537', '2', '10100-4890')

INSERT INTO Student(MemberID, offerID) values('9400-68021', '10100-6011')

INSERT INTO Category(categoryID, name, categoryDescription, durationDays, durationHours) values('1200100-12', 'Room', 'A room that can be used for tutoring, lectures, or studying', '0', '3')

INSERT INTO Category(categoryID, name, categoryDescription, durationDays, durationHours) values('1200100-90', 'Camera', 'A device to take photos, mostly used by photography students and staff', '7', '0')

INSERT INTO Category(categoryID, name, categoryDescription, durationDays, durationHours) values('1200100-21', 'Software', 'Downloadable software given to students for computer based courses', '150', '0')

INSERT INTO Category(categoryID, name, categoryDescription, durationDays, durationHours) values('1200100-56', 'Speaker', 'A device used that produces sound that are mostly used by music students and staff', '0', '3')

INSERT INTO Resource(resourceID, categoryID, resourceDescription) values('11100-2122', '1200100-21', 'Course-recommended diagram designer')

INSERT INTO Resource(resourceID, categoryID, resourceDescription) values('11100-2371', '1200100-90', 'A high-tech camera used for high-definition photography')

INSERT INTO Resource(resourceID, categoryID, resourceDescription) values('11100-3217', '1200100-21', 'Course-recommended source-code editor subscription')

INSERT INTO Resource(resourceID, categoryID, resourceDescription) values('11100-4190', '1200100-56', 'Large guitar amp')

INSERT INTO Resource(resourceID, categoryID, resourceDescription, building, room) values('1110-00101', '1200100-12', 'Main lecture hall in B01', 'B01', '101')

INSERT INTO Resource(resourceID, categoryID, resourceDescription, building, room) values('1110-00205', '1200100-12', 'Small classroom in B01 on 2nd floor', 'B01', '205')

INSERT INTO Resource(resourceID, categoryID, resourceDescription, building, room) values('1110-00325', '1200100-12', 'Small classroom in B01 on 3rd floor', 'B01', '325')

INSERT INTO Resource(resourceID, categoryID, resourceDescription, building, room) values('1110-01102', '1200100-12', 'Second largest lecture hall in B02', 'B02', '102')

INSERT INTO Moveable(resourceID, name, manufacturer, model, year, assetValue) values('11100-2122', 'Visia', 'Microsoft', 'N/A', '2021', '$6.90/m')

INSERT INTO Moveable(resourceID, name, manufacturer, model, year, assetValue) values('11100-2371', 'Super Frame', 'Nokia', 'NSF200051', '2022', '$1200')

INSERT INTO Moveable(resourceID, name, manufacturer, model, year, assetValue) values('11100-3217', 'Visual Studio Code Premium', 'Microsoft', 'N/A', '2015', '$40/m')

INSERT INTO Moveable(resourceID, name, manufacturer, model, year, assetValue) values('11100-4190', 'Super Loud GAmp', 'Samsung', 'SGA221345', '2017', '$350')

INSERT INTO Immovable(resourceID, capacity) values('1110-00101', '200')

INSERT INTO Immovable(resourceID, capacity) values('1110-00205', '30')

INSERT INTO Immovable(resourceID, capacity) values('1110-00325', '30')

INSERT INTO Immovable(resourceID, capacity) values('1110-01102', '50')

INSERT INTO Loan(loanID, dateTimeBorrowed, dateTimeDue, memberID, resourceID) values('7011-02122', '2021-12-21 12:50:55', '2022-05-20 12:50:55', '9400-29537', '11100-2122')

INSERT INTO Loan(loanID, dateTimeBorrowed, dateTimeDue, memberID, resourceID) values('7011-02123', '2021-12-25 16:32:12', '2022-05-24 16:32:12', '9400-29537', '11100-3217')

INSERT INTO Loan(loanID, dateTimeBorrowed, dateTimeReturned, dateTimeDue, memberID, resourceID) values('7011-06900', '2022-10-01 09:12:42', '2022-10-07 12:12:53', '2022-10-08 09:12:42', '9400-61910', '11100-2371')

INSERT INTO Loan(loanID, dateTimeBorrowed, dateTimeReturned, dateTimeDue, memberID, resourceID) values('7011-04235', '2022-03-11 10:25:40', '2022-03-11 13:01:22', '2022-03-11 13:25:40', '9400-89211', '11100-4190')

INSERT INTO CourseOfferingPrivilege(privID, privilegeDescription, borrowCapacity, categoryID, offerID) values('10211-3399', 'Photography students may borrow 1 cameras for their study', '1', '1200100-90', '10100-3399')

INSERT INTO CourseOfferingPrivilege(privID, privilegeDescription, borrowCapacity, categoryID, offerID) values('10211-2135', 'Music students may borrow 1 speaker for their study', '1', '1200100-56', '10100-2135')

INSERT INTO CourseOfferingPrivilege(privID, privilegeDescription, borrowCapacity, categoryID, offerID) values('10211-4890', 'Software engineering students may borrow 3 software subscriptions or products for their study', '3', '1200100-21', '10100-4890')

INSERT INTO CourseOfferingPrivilege(privID, privilegeDescription, borrowCapacity, categoryID, offerID) values('10211-6011', 'Primary school teachers can loan rooms for study', '1', '1200100-12', '10100-6011')

INSERT INTO Acquisition(acquisitionID, name, manufacturer, model, year, price, reasoning, urgency, memberID) values('2112-01554', 'Extra Super Load GAmp NEW', 'Samsung', 'SGA3221555', '2022', '$499.99', 'Super Loud GAmp is aged and needs replacing', 'Urgent', '9400-89211')

INSERT INTO Acquisition(acquisitionID, name, manufacturer, model, year, price, reasoning, memberID) values('2112-82157', 'BlueJ Premium', 'Michael Kölling', 'NA', '2022', '$12.55/m', 'Hard to program without this', '9400-29537')

INSERT INTO Acquisition(acquisitionID, name, manufacturer, model, year, price, reasoning, memberID) values('2112-37432', 'Teaching For Dummies', 'John Simone', 'NA', '2021', '$55.00', 'Require new textbooks for teaching', '9400-68021')

INSERT INTO Acquisition(acquisitionID, name, manufacturer, model, year, price, reasoning, urgency, memberID) values('2112-23144', 'Masters Film Camera', 'Toshiba', 'Film Pro', '2022', '$4500.00', 'Need film camera for next years movie making course', 'Urgent', '9400-28461')

INSERT INTO Reservation(reservationID, reservedForDateTime, memberID, resourceID) values('6990-01561', '2022-05-01 06:22:11', '9400-96032', '1110-00325')

INSERT INTO Reservation(reservationID, reservedForDateTime, memberID, resourceID) values('6990-02122', '2022-06-05 09:40:59', '9400-96032', '1110-00325')

INSERT INTO Reservation(reservationID, reservedForDateTime, memberID, resourceID) values('6990-03288', '2022-09-19 15:22:14', '9400-28461', '11100-2371')

INSERT INTO Reservation(reservationID, reservedForDateTime, memberID, resourceID) values('6990-03167', '2022-11-20 1:00:00', '9400-89211', '11100-4190')

## SQL Queries

### 8.2.1 Query 1

SELECT m.name

FROM Member m JOIN Student s

ON (m.memberID = s.memberID)

WHERE s.offerID = '10100-4890’

### 8.2.2 Query 2

SELECT p.borrowCapacity

FROM Member m JOIN Student s

ON (m.memberID = s.memberID)

JOIN CourseOfferingPrivilege p

ON (s.offerID = p.offerID)

WHERE m.name = 'Samual Web' AND p.offerID = '10100-2135'

### 8.2.3 Query 3

SELECT m.name, m.phone, COUNT(r.reservationID) AS totalReservations

FROM Member m JOIN Reservation r

ON (m.memberID = r.memberID)

WHERE m.memberID = '9400-96032' AND YEAR(r.reservedForDateTime) = 2022

GROUP BY m.name, m.phone;

### 8.2.4 Query 4

SELECT m.name

FROM Member m JOIN Loan l

ON (m.memberID = l.memberID)

JOIN Moveable mr

ON (l.resourceID = mr.resourceID)

JOIN Resource r

ON (r.resourceID = mr.resourceID)

JOIN Category c

ON (c.categoryID = r.categoryID)

WHERE c.name = 'Camera' AND mr.model = 'NSF200051' AND YEAR(l.dateTimeBorrowed) = YEAR(GETDATE())

### 8.2.5 Query 5

SELECT TOP 1 m.name, m.resourceID

FROM Loan l JOIN Moveable m

ON (m.resourceID = l.resourceID)

JOIN Resource r

ON (m.resourceID = r.resourceID)

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

GROUP BY m.name, m.resourceID

### 8.2.6 Query 6

SELECT r.reservedForDateTime, COUNT(r.reservationID) as reservationsMade, re.room

FROM Reservation r JOIN Resource re

ON (r.resourceID = re.resourceID)

WHERE DAY(r.reservedForDateTime) = 01 AND MONTH(r.reservedForDateTime) = 05 AND YEAR(r.reservedForDateTime) = 2022 OR

DAY(r.reservedForDateTime) = 05 AND MONTH(r.reservedForDateTime) = 06 AND YEAR(r.reservedForDateTime) = 2022 OR

DAY(r.reservedForDateTime) = 19 AND MONTH(r.reservedForDateTime) = 09 AND YEAR(r.reservedForDateTime) = 2022

GROUP BY r.reservedForDateTime, re.campus, re.building, re.room

## Print Tables

SELECT \*

FROM Member

SELECT \*

FROM Staff

SELECT \*

FROM Student

SELECT \*

FROM StudentCourseOffering

SELECT \*

FROM CourseOfferingPrivilege

SELECT \*

FROM Resource

SELECT \*

FROM Moveable

SELECT \*

FROM Immovable

SELECT \*

FROM Category

SELECT \*

FROM Loan

SELECT \*

FROM Acquisition

SELECT \*

FROM Reservation