# **The School of Data Science (SDS) at the University of Northwest Technology Resource Management Logical and Physical Database Design Concept**

## Final Report

Ebrahim Gulamali

# Preface

The School of Data Science (SDS) at the University of Northwest Technology manages and lends resources to staff and students for assignment and project purposes. The resources include rooms, cameras, speakers, software, phones, etc.

[Preface 1](#_Toc139810566)

[Requirements Analysis 2](#_Toc139810567)

[Data Requirements 2](#_Toc139810568)

[Transaction Requirements 4](#_Toc139810569)

[Business Rules 5](#_Toc139810570)

[Data Dictionary 6](#_Toc139810571)

[Entity Types 6](#_Toc139810572)

[Attributes 7](#_Toc139810573)

[Relationship Types 10](#_Toc139810574)

[Enhanced Entity Relationship Diagram 11](#_Toc139810575)

[Sample EER Model of Library Database 11](#_Toc139810576)

[(without attributes shown) 11](#_Toc139810577)

[EER Model 12](#_Toc139810578)

[(Entities and attributes - without relationships) 12](#_Toc139810579)

[Relational Model 13](#_Toc139810580)

[Boyce-Codd Normal Form Normalisation 14](#_Toc139810581)

[Relation Normal Form Judgement 15](#_Toc139810582)

[Relation Normalisation Verdict 18](#_Toc139810583)

[Normalisation Process 19](#_Toc139810584)

[List of relations in BCNF: 20](#_Toc139810585)

[SQL Script: 21](#_Toc139810586)

# Requirements Analysis

## Data Requirements

1. Catalogue Service

Resources

The school of Data Sciences has two types of resources which are offered to be lended, movable and immovable objects; for example a camera or a room which is recorded for every item. Each resource is uniquely identified by its Id (Primary Key). A brief description of the resource purpose along with its status of availability (‘In Use’, ‘Maintenance’, ‘Available’ ‘Borrowed’, ‘Lost’, ‘Damaged’, etc.) is included as data describing every resource in the system. Every resource is also placed into its respective category.

Movable (subset)

A resource which is recorded as movable also has its own data set attributed in the catalogue. This includes the name of the resource, the make, manufacturer and model name/number of the resource. The year it was produced, its asset value and it’s building BDS is also recorded.

Immovable (subset)

A resource record as immovable will be lab rooms, classrooms or other related resources which are offered by The School of Data Science. The data stored regarding each immovable resource will include the capacity which describes the maximum amount of people that can be accommodated, the room Id, Building name/Id and the campus.

Category

When a resource is placed into its respective category, the process of cataloguing becomes more efficient, each category also has its own set of attributes to further increase efficiency. This includes a unique code (Primary Key), name, description and max time allowed to borrow/book (in days and/or hours).

1. Loan Service

Members

Member describes the collection of staff and students enrolled in courses offered by SDS. They are entitled to lending and reservation rights and therefore must be recorded in the system. Each member is associated by a unique Id (Primary Key), first and last name, residential address, contact phone number, email, status (disabled or active) and optional comment. Studentmember will have varying degrees of privileges which affect loaning and reservation.

StudentMember (subset)

Student detail will include the course offering the student is enrolled in and the points. Only students who have valid points may be allowed to reserve or loan resources.

StaffMember (subset)

Staff detail will include their position in the school and a flag if he/she is an administrator. Staff members have no restrictions on the amount of resources that can be booked/ loaned.

CourseOffering

Student members will be enrolled in courses and information about course offerings and student enrollment is maintained in the system. Each course is identified by a offering id (Primary Key), a course id, name of the course, semester offered, year offered and the start and end date of the course.

Privellage

Each student is granted privileges based on the course they are enrolled in, this affects their loaning limits. Each course is assigned privileges to different categories of resources. A privillege will have a data set including; privilege id (primary key), the name, a description, a category to which the privilege is granted for and maximum number of resources that can be borrowed or booked at any given time from the category.

Loan

When a resource is loaned, load id (primary key), information about the resource loaned, member lending the resource, resource condition, date and time loaned, due date and time, and date and time returned are all maintained. This is to ensure an accurate log of resources to account for any damaged/lost items and to allow for the active status to be updated.

1. Reservation Service

Reservation

When a member has the privileges to borrow/book a resource they can also make a reservation to allow for the item to be booked for pickup by the member on the requested date and time. Information stored about a reservation request includes the date and time the item is required and a due date & time. Reservations must not conflict with one another.

1. Acquisition Service

Acquisition

New acquisitions to the school can be requested by each member. When a request is made data about the person requesting acquisition, acquisition id (primary key), item name, make, manufacturer, model, year produced, a description of the required item and its urgency (priority based on an int 0-5) is all stored within the system. The administrator of the system will also assign an item status (“acquired”, “pending”, etc.), a fund code, vendor code, price and any other notes pertaining to the request.

## Transaction Requirements

**Data Manipulation Operations**

* Insert or Update existing Loan
* Insert or Update existing acquisition
* Insert or Update existing reservation
* Insert or Update existing member (staff or student)
* Insert or Update existing book or a resource in the system
* Delete a book or a resource in the system
* Delete a member (staff or student)
* Delete a storeroom
* Remove a reservation after 24 hours of due date
* Remove an acquisition request after it has been cancelled

**Queries**

* List all loans due on a specific date
* List all reservations on a specific date
* List all acquisitions with a high urgency
* List all available immovable objects on a campus
* List all available loanable items in a particular category
* Search a member based on their unique Id
* Search a loaned item based on an employee number, on a particular date.
* Find the student with a late returned item number.
* List all the reservations for a particular item
* Report of points earned for a particular student during certain period
* List all loans due after a particular date
* List all students with overdue loans
* List all categories
* List all administrator staff
* List all courses offered in a specific year

## 

## 

## Business Rules

**Expiration of student member access**

A student’s borrowing privileges are automatically taken away when the current date is later than end date of all his/her enrolled course offerings. The status of student member is set to

“disabled” .

**Maximum items loaned or reserved at any one time**

* A member cannot borrow or reserve more than the maximum number of items specified in his/her privileges at any given time.
* In general there is a maximum limit of 10 items
* Staff members have no limit

**Penalty for late returns by students**

* Each student member has a default set of points earned (12 at the beginning)
* A penalty of 3 points is incurred for each overdue day.
* When the point is reduced to 0, member status is disabled, disallowing borrowing/reservation privileges.
* The administrator has rights to reset/amend points

**Cancellation of Reservations**

* A reserved item is cancelled if it is not picked up after a day of the required date or due date (whichever is earlier)
* Non cancellation of reservation by member, then 1 demerit point
* Also, the administrator holds the right to cancel any reservation.

**Borrowing/Reservation**

* The duration of borrowing/reservation periods (either number of days or hours) are determined by the category to which the item belongs
* No two reservations should conflict!

## 

# Data Dictionary

## Entity Types

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Resource | Immovable/movable object that can be loaned | object | When a resource is being requested by a member |
| Movable | A movable resource that can be loaned | MovableResource | When a movable item is being loaned |
| Immovable | A immovable resource with a capacity | Room | When an immovable item is being loaned |
| Member | Staff or Students with varying privilege levels | Borrowers | When a new member is registered in the database |
| StaffMember | Staff working for the university | Teacher | When a new staff member is registered in the database |
| StudentMember | Students enrolled in courses offered by the university | Scholar | When a new student is enrolled in a course. |
| Privilege | Students enrolled in certain courses allowing them to request and loan additional resources | StudentCourse, Courseloan, Courseoffering | When a student makes a loan past the default student limits |
| Category | Category of resource | Catagorycode | When a resource is being loaned |
| Loan | Details regarding a loan taken out by a member. | lendedItem | When a resource is being loaned |
| Acquisition | Details of outstanding Acquisition request made by staff or students | Acquisition, request | When someone needs a new resource to be added |
| CourseOffering | Details regarding course offererings and their respective dates for assigning privelages | CourseOffer | When a privilege is being assigned |
| Reservation | Information about resources that are put on hold for a certain period of time by a member | hold | When a member makes a hold request |

## Attributes

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| **Member** | **MemberId** | **Student Id/Staff Id** | **VarChar(15)** | **N** | **N** | **N** |  |
| **firstName** | **First name** | **VarChar(20)** | **N** | **N** | **N** |  |
| **lastName** | **Last name** | **VarChar(20)** | **N** | **N** | **N** |  |
| **PhoneNo** | **Contact Phone number** | **INT** | **Y** | **N** | **N** |  |
| **ActiveStatus** | **Status of activity** | **VarChar(08)** | **N** | **N** | **N** | **‘active’** |
| **Address** | **Residential addy** | **VarChar(50)** | **Y** | **N** | **N** |  |
| **Comments** | **Extra comments** | **VarChar(300)** | **Y** | **Y** | **N** |  |
| **Email** | **Email address** | **VarChar(50)** | **Y** | **N** | **N** |  |
| **StudentMember** | **MemberId** | **Student ID** | **VarChar(15)** | **N** | **N** | **Y** |  |
| **HoldPoint** | **Number of Hold points** | **INT** | **N** | **N** | **N** | **‘12’** |
| **StaffMember** | **MemberId** | **Staff ID** | **VarChar(15)** | **N** | **N** | **N** |  |
| **position** | **Role in Uni** | **VarChar(20)** | **Y** | **N** | **N** |  |
| **adminFlag** | **Flag** | **Varchar(10)** | **N** | **N** | **N** |  |
| **Loan** | **LoanId** | **Unique loan identifier** | **Int (10)** | **N** | **N** | **N** |  |
| **DateTimeLoaned** | **The date & time the loan is made** | **date** | **N** | **N** | **N** |  |
| **DateTime returned** | **The date & time the loan is returned** | **date** | **Y** | **N** | **N** |  |
| **DateTimedue** | **The date & time the loan is due** | **date** | **N** | **N** | **N** |  |
| **Resource** | **ResourceId** | **Unique numerical Identifier** | **VarChar (10)** | **N** | **N** | **Y** |  |
| **Description** | **Cost of the total order** | **Varchar(50)** | **N** | **N** | **N** |  |
| **status** | **Status of resource** | **Varchar(20)** | **N** | **Y** | **Y** | **‘Available’** |
| **Movable** | **ResourceId** | **Unique Identifier** | **VarChar (10)** | **N** | **N** | **N** |  |
| **Name** | **The name of the . movable resource** | **Varchar(50)** | **N** | **N** | **N** |  |
| **Make** | **The Make of the movable resource** | **Varchar(50)** | **Y** | **Y** | **N** |  |
| **Manufacturer** | **The manufacturer of .the movable resource** | **Varchar(50)** | **Y** | **N** | **N** |  |
| **Model** | **The model of the .movable resource** | **Varchar(50)** | **Y** | **N** | **N** |  |
| **Year** | **The year model of the .movable resource** | **int** | **Y** | **N** | **N** |  |
| **AssetValue** | **The asset value of the .movable resource** | **Money** | **N** | **N** | **N** |  |
| **Immovable** | **ResourceId** | **Unique Identifier** | **VarChar (10)** | **N** | **N** | **N** |  |
| **Capacity** | **The capacity of .members the immovable ,object can hold** | **Int** | **N** | **N** | **N** |  |
| **RoomId** | **Identifier for Room** | **Int** | **N** | **N** | **N** |  |
| **BuildingId** | **Identifier for Building** | **VarChar(10)** | **N** | **N** | **N** |  |
| **Campus** | **Campus of resource** | **VarChar(20)** | **N** | **N** | **N** |  |
| **CourseOffering** | **CourseId** | **Course Identification code** | **VarChar(8)** | **N** | **N** | **N** |  |
| **Name** | **Name of Course** | **VarChar (20)** | **N** | **N** | **N** |  |
| **SemesterOffered** | **Estimated delivery time, etc.** | **VarChar (10)** | **N** | **N** | **N** |  |
| **YearOffered** | **Year course runs** | **Numeric (4)** | **Y** | **N** | **Y** |  |
| **DateBegins** | **Start date** | **Datetime** | **N** | **N** | **Y** |  |
| **DateEnds** | **End date** | **Datetime** | **N** | **N** | **N** |  |
| **Privilege** | **PrivilegeId** | **Unique Id for privilege** | **VarChar (20)** | **N** | **N** | **N** |  |
| **Name** | **Name of privilege granted is granted for** | **VarChar (20)** | **N** | **N** | **N** |  |
| **Description** | **Description of privilege** | **VarChar (300)** | **N** | **N** | **N** |  |
| **MaxItems** | **Max number of items allowed** | **INT** | **N** | **N** | **N** |  |
| **Acquisition** | **acquisitionId** | **Unique Id for acquisition** | **INT** | **N** | **N** | **N** |  |
| **ItemName** | **Resource Name** | **VarChar(20)** | **N** | **N** | **N** |  |
| **Make** | **Make of resource** | **VarChar(20)** | **Y** | **N** | **N** |  |
| **Manufacturer** | **Manufacturer Name** | **VarChar(20)** | **Y** | **N** | **N** |  |
| **Model** | **Model Number/code** | **VarChar(20)** | **Y** | **N** | **N** |  |
| **Year** | **Year produced** | **Numeric(4)** | **Y** | **N** | **N** |  |
| **Description** | **Brief description on purpose** | **VarChar(300)** | **Y** | **N** | **N** |  |
| **Urgency** | **How quick the resource is needed** | **VarChar(20)** | **Y** | **N** | **N** |  |
| **Status** | **Present Status of Acquisition request** | **VarChar(20)** | **Y** | **N** | **N** | **“Pending”** |
| **FundCode** | **Code of fund** | **VarChar(20)** | **N** | **N** | **N** |  |
| **VendorCode** | **Code of Vendor** | **VarChar(20)** | **N** | **N** | **N** |  |
| **Price** | **Price of resource** | **Money** | **N** | **N** | **N** |  |
| **Notes** | **Any extra notes** | **VarChar(300)** | **Y** | **N** | **N** |  |
| **Reservation** | **reservationId** | **Unique numerical identification, ensuring no duplicate reservations** | **Int** | **N** | **N** | **N** |  |
| **DateTimeRequired** | **24 hour format** | **datetime** | **N** | **N** | **N** |  |
| **DateTimeDue** | **Reservation cancelled 24 hours after this date and time (24 hour format)** | **datetime** | **N** | **N** | **N** |  |
| **Category** | **categoryId** | **Unique numerical code** | **INT** | **N** | **N** | **N** |  |
| **Name** | **Name of category** | **VarChar(20)** | **N** | **N** | **N** |  |
| **Description** | **Description of category** | **VarChar(300)** | **N** | **N** | **N** |  |
| **Max time** | **Max amount of time to loan** | **VarChar(20)** | **N** | **N** | **N** |  |

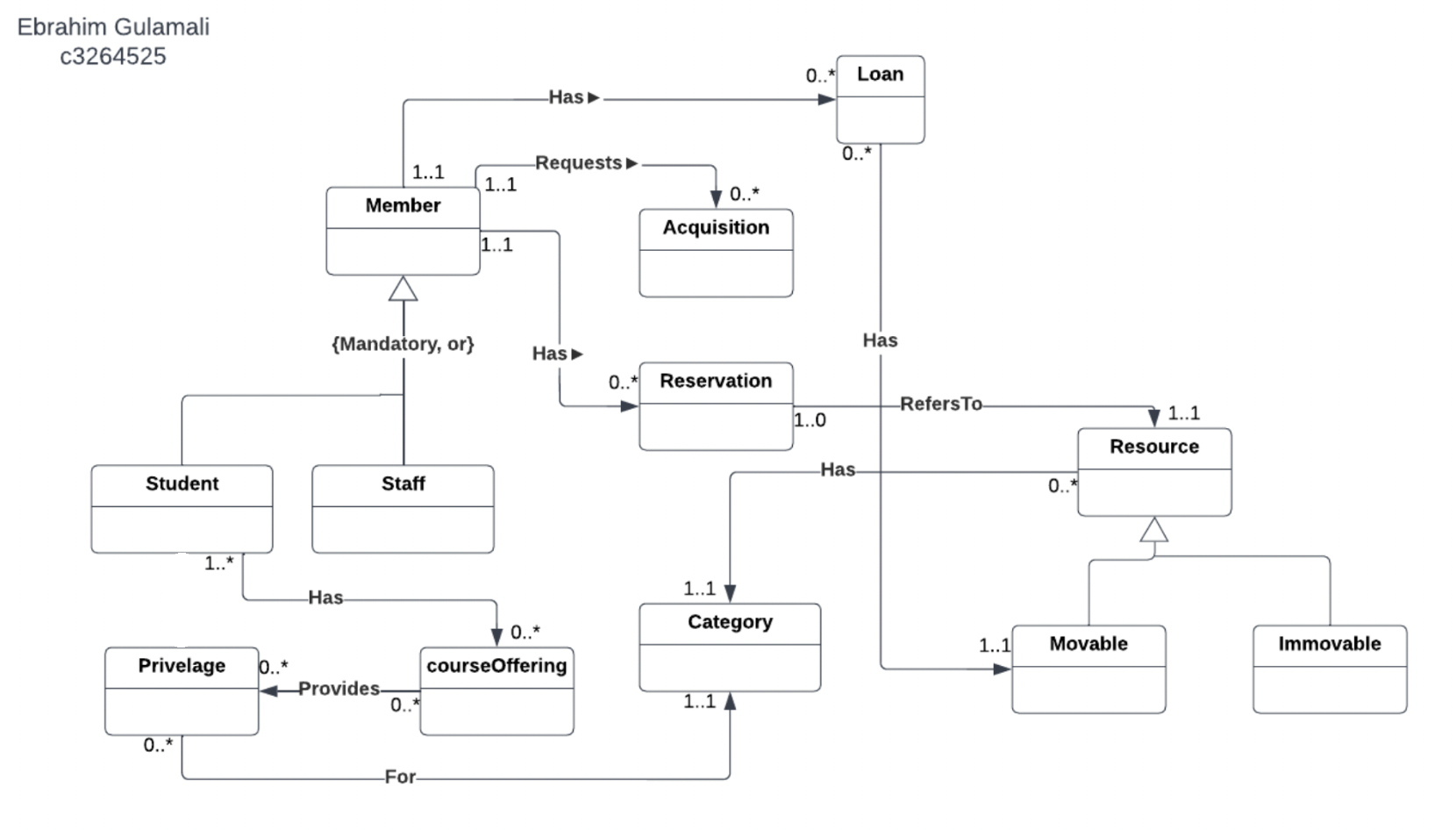
## Relationship Types

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name** | **Multiplicity** | **Relationship** | **Multiplicity** | **Entity Name** |
| Member | 1..1 | has | 0..\* | Loan |
| 1..1 | has | 0..\* | Reservation |
| 1..1 | requests | 0..\* | Acquisition |
| (Man,Or) | Generalisation | (Man,Or) | Staff |
| (Man,Or) | Generalisation | (Man,Or) | Student |
| Student | 1..1 | Has | 0..\* | courseOffering |
| Loan | 0..\* | Has | 1..1 | Movable |
| reservation | 1..\* | RefersTo | 1..1 | Resource |
| Resource | 0..\* | Has | 1..1 | Category |
| (Man,Or) | Generalisation | (Man,Or) | Immovable |
| (Man,Or) | Generalisation | (Man,Or) | Movabe |
| courseOffering | 1..1 | Provides | 1..1 | Privilege |
| Privilege | 0..\* | For | 1..1 | Category |

# Enhanced Entity Relationship Diagram

## Sample EER Model of Library Database

## (without attributes shown)

****

## EER Model

## (Entities and attributes - without relationships)

# Relational Model

**Member(**MemberId, firstName, lastName, address(streetName, streetNumber, suburb, postcode), phone, email, activeStatus, comments**)**

**Primary Key**: MemberId

**StudentMember(**MemberId, holdPoint**)**

​**Primary Key**: MemberId

**Foreign Key**: MemberId **references** Member(MemberId)

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

**StaffMember(**MemberId, position, adminFlag**)**

​**Primary Key**: MemberId

**Foreign Key**: MemberId **references** Member(MemberId)

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

**Loan(**LoanId, dateTimeLoaned, MemberId, ResourceId, dateTimeReturned, dateTimeDue**)**

**Primary Key:** LoanId

**Foreign Key:** MemberId **references** Member(MemberId)

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

**Foreign Key:** ResourceId **references** Movable(ResourceId)

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

**Resource(**ResourceId, description, categoryId, status**)**

**Primary Key:** ResourceId

**Foreign Key**: CatergoryId **references** Category(categoryId)

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

**Immovable(**ResourceId, Capacity, roomId, buildingId, campus**)**

​**Primary Key**: ResourceId

​**Foreign Key**: ResourceId **references** Resource(ResourceId)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Movable(**ResourceId, name, make, manufacturer, model, year, assetValue**)**

​**Primary Key**: ResourceId

**Foreign Key**: ResourceId **references** Resource(ResourceId)

**ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key**: CategoryId **references** Category(categoryId)

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

**Acquisition(**AcquisitionId, MemberId, Name, make, manufacturer, model, year, description, urgency, Status, FundCode, VendorCode, Price, Note**)**

**Primary Key:** AcquisitionId

**Foreign Key**: MemberId **references** Member(MemberId)

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

**courseOffering(**OfferingId, courseId, name, semesterOffered, yearOffered, dateBegins, dateEnds**)**

**Primary Key:** OfferingId

**Candidate Key:** CourseId, yearOffered, semesterOffered

**Privilege(**PrivilegeId, CategoryId, name, description, maxItems**)**

**Primary Key:** PrivilegeId

**Foreign Key**: CatergoryId **references** Category(categoryId)

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

**Category(**CategoryId, name, description, MaxTime**)**

**Primary Key:** CatgeoryId

**Reservation(**ReservationId,dateTimeRequired, MemberId, ResourceId, dateTimeDue**)**

**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**

**Student\_CourseOffering**(StudentId, OfferingId)

**Primary key: (**MemberId, OfferingId)

**Foreign Key:** MemberId **references** Student(MemberId)

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

**Foreign Key**: OfferingId **references** Course(OfferingId)

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

**CourseOffering\_Privilege: (**PrivilageId, OfferingID)

**Primary Key:** (PrivilageId, OfferingID)

**Foreign Key**: PrivilageId **references** Privilege(PrivilegeId)

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

**Foreign Key**: OfferingId **references** Course(OfferingId)

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

# Boyce-Codd Normal Form Normalisation

**By definition of relation model every relation is in 1st Normal Form (Atomic attributes)**

**Second Normal Form Check:**  Is every ​​non-candidate key attribute fully

functionally dependent on a candidate key?

**Third Normal Form Check:** No non-candidate-key attribute is

transitively dependent on a candidate key

**BCNF Check:** The determinant (left side) of every functional dependency (FD) is a candidate key

## Relation Normal Form Judgement

**Member**

**Member(**MemberId, firstName, lastName, streetName, address, phone, email, activeStatus, comments**)**

**Primary Key**: MemberId

**Functional Dependencies:**

MemberId → firstName, lastName, address, phone, email, activeStatus, comments

* This FD is the only FD in the relation and the determinant is PK.
* Therefore its in BCNF

**StudentMember**

**StudentMember(**MemberId, holdPoint**)**

​**Primary Key**: StudentId

**Foreign Key**: MemberId **references** Member(MemberId)

**Functional Dependencies:**

StudentId → holdPoint, MemberId

* The key attributes are StudentId and MemberId which are also the candidate keys
* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

**StaffMember**

**StaffMember(**MemberId, position, adminFlag**)**

​**Primary Key**: StaffId

**Foreign Key**: MemberId **references** Member(MemberId)

**Functional Dependencies:**

StaffId → position, adminflag, MemberId

* Key attributes: StaffId, MemberId
* Candidate Keys: StaffId, MemberId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

**Loan**

**Loan(**LoanId, dateTimeLoaned, MemberId, ResourceId, dateTimeReturned, dateTimeDue**)**

**Primary Key:** LoanId

**Foreign Key:** MemberId **references** Member(MemberId)

**Foreign Key:** ResourceId **references** Resource(ResourceId)

**Functional Dependencies:**

LoanId, MemberId, ResourceId → DateTimeLoaned, DateTimeReturned, DateTimeDue

* Key attributes: LoanId, MemberId, ResourceId
* Candidate Keys: LoanId, MemberId, ResourceId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

**Resource**

**Resource(**ResourceId, description, status**)**

**Primary Key:** ResourceId

**Functional Dependencies:**

ResourceId → Description, Status

* Key attributes: ResourceId
* Candidate Keys: ResourceId
* The left side of the FD is a candidate key and nothing on the right side is a key attribute
* Therefore its in BCNF

**Immovable**

**Immovable(**ResourceId, Capacity, roomId, buildingId, campus**)**

​**Primary Key**: ImmovableId

​**Foreign Key**: ResourceId **references** Resource(ResourceId)

**Functional Dependencies:**

ImmovableId, ResourceId → Capacity, RoomId, BuildingId, Campus

* Key attributes: ImmovableId, ResourceId
* Candidate Keys: ImmovableId, ResourceId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

**Movable**

**Movable(**ResourceId, name, make, manufacturer, model, year, assetValue**)**

​**Primary Key**: MovableId

**Foreign Key**: ResourceId **references** Resource(ResourceId)

**Functional Dependencies:**

ResourceId → name, make, manufacturer, model, year, assetValue(PK)

* Key attributes: MovableId, ResourceId
* Candidate Keys: MovableId, ResourceId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

**Acquisition**

**Acquisition(**AcquisitionId, MemberId, Name, make, manufacturer, model, year, description, urgency, Status, FundCode, VendorCode, Price, Note**)**

**Primary Key:** AcquisitionId

**Foreign Key**: MemberId **references** Member(MemberId)

**Functional Dependencies:**

AcquisitionId→ Name, make, manufacturer, model, year, description, urgency, Status, FundCode, VendorCode, Price, Note (PK)

* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

**CourseOffering**

**courseOffering(**OfferingId, courseId, name, semesterOffered, yearOffered, dateBegins, dateEnds**)**

**Primary Key:** OfferingId

**Candidate Key:** CourseId, yearOffered, semesterOffered

**Functional Dependencies:**

OfferingId→ courseId, name, semesterOffered, yearOffered, dateBegins, dateEnds

semesterOffered → dateBegins, dateEnds

* Key attributes: OfferingId, StudentId
* Candidate Keys: OfferingId, StudentId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute for the first FD however this is not true for the second
* Therefore it is in 2nd form normalisation

**Privilege**

**Privilege(**PrivilegeId, OfferingId, name, description, maxItems**)**

**Primary Key:** PrivilegeId

**Foreign Key**: OfferingId **references** Course(OfferingId)

**Functional Dependencies:**

PrivilegeId, OfferingId → name, description, maxItems

* Key attributes: PrivilegeId, OfferingId
* Candidate Keys: PrivilegeId, OfferingId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

**Category**

**Category(**CategoryId, PrivilegeId, ResourceId, name, description, MaxTime**)**

**Primary Key:** CategoryId

**Foreign Key**: PrivilegeId **references** Privilege(PrivilegeId)

**Foreign Key**: ResourceId **references** Resource(ResourceId)

**Functional Dependencies:**

Category Code, PrivilegeId, ResourceId → Name, description, MaxTime

* Key attributes: Catergorycode, PrivilegeId, ResourceId
* Candidate Keys: Catergorycode, PrivilegeId, ResourceId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

**Reservation**

**Reservation(**ReservationId,dateTimeRequired, MemberId, ResourceId, dateTimeDue**)**

**Primary Key:** ReservationId **references** Reservation(dateTimeReserved, dateTimeDue)

**Foreign Key:** MemberId **references** Member(MemberId)

**Foreign Key:** ResourceId **references** Resource(ResourceId)

**Functional Dependencies:**

ReservationId, MemberId, ResourceId → DateTimeRequired, DateTimeDue

* Key attributes: ReservationId, MemberId, ResourceId
* Candidate Keys: ReservationId, MemberId, ResourceId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute
* Therefore its in BCNF

## 

## Relation Normalisation Verdict

In conclusion, after checking each relation, every relation except courseOffering is in Boyce-Codd Normal Form. This was done through identifying and analysing the functional dependencies found in the relational model. Therefore only courseOffering needs to be normalised further.

## Normalisation Process

**Example 1:**

**CourseOffering**

**courseOffering(**OfferingId, StudentId, courseId, name, semesterOffered, yearOffered, dateBegins, dateEnds**)**

**Primary Key:** OfferingId

**Foreign Key**: StudentId **references** Student(StudentId)

**Functional Dependencies (FD):**

1. OfferingId → courseId, name, semesterOffered, yearOffered, dateBegins, dateEnds
2. CourseId, yearOffered, semesterOffered → OfferingId, name, dateBegins, dateEnds (CK)
3. CourseId → name

**R1: CourseDetails:**

(x,y) of FD3: CourseDetails(courselD, name)

**Primary Key:**  courselD

**R2**:**CourseOfferingNew:**

R-y of FD3: CourseOfferingNew (offeringID, courselD, yearOffered, semesterOffered, dateBegins, dateEnds)

**Primary Key:** offeringId

**Foreign Key:** CourseId references CourseDetails(courseId)

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

* Key attributes: OfferingId, StudentId
* Candidate Keys: OfferingId, StudentId
* The left side of the FD are candidate keys and nothing on the right side is a key attribute for the first FD however this is not true for the second
* Therefore it is in 2nd form normalisation

**Converting to BCNF:**

Since courseOffering has two functional dependencies and the second one does not have a candidate key on the left hand side, the relation needs to be broken up into two.

## 

## List of relations in BCNF:

* Member
* StudentMember
* StaffMember
* Loan
* Resource
* Immovable
* Movable
* Acquisition
* Privilege
* Category
* Reservation
* CourseOfferingNew
* CourseDetails

# 

# SQL Script:

--create DATABASE a3

--go

--use a3

Drop TABLE Loan

drop table Reservation

drop table Acquisition

Drop Table Student\_CourseOffering

drop table CourseDetails

Drop Table CourseOffering\_Privilege

Drop TABLE Moveable

drop table Immovable

Drop TABLE Resource

drop table CourseOfferingNew

drop table privilege

Drop TABLE Category

Drop TABLE StudentMember

drop table StaffMember

Drop TABLE Member

go

CREATE TABLE Member(

memberID VARCHAR(15) NOT NULL,

FirstName VARCHAR(20) NOT NULL,

LastName VARCHAR(20) NOT NULL,

PhoneNo INT,

ActiveStatus VARCHAR(8) DEFAULT 'active' CHECK (ActiveStatus IN ('active', 'expire')) NOT NULL,

Address VARCHAR(50),

Comments VARCHAR(300),

Email VARCHAR(50) NOT NULL,

PRIMARY KEY (memberID),

);

go

CREATE TABLE StudentMember(

memberID VARCHAR(15) NOT NULL,

HoldPoint INT DEFAULT 12,

PRIMARY KEY (memberID),

Foreign Key (memberID) references Member(memberId) On Update Cascade On

Delete CASCADE,

);

go

create table StaffMember(

MemberID VARCHAR(15) NOT NULL,

Position VARCHAR(20),

AdminFlag VARCHAR(10) NOT NULL,

primary key (MemberID),

foreign key (MemberID) references Member(MemberId) on update cascade on delete NO ACTION,

);

go

CREATE TABLE Category(

CategoryId INT NOT NULL,

name VARCHAR(20) NOT NULL,

Description varchar(300) NOT NULL,

maxTime VarChar(20) NOT NULL,

PRIMARY KEY (categoryID),

);

go

Create Table Resource(

resourceID VARCHAR (10) NOT NULL,

Description VARCHAR(50) NOT NULL,

Status VARCHAR(20) DEFAULT 'available' CHECK (Status IN ('available', 'occupied', 'damaged')) NOT NULL,

categoryID INT,

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 (50) NOT NULL,

Make VARCHAR(50),

Manufacturer VARCHAR(50),

Model VARCHAR(50),

Year int,

AssetValue Money NOT NULL,

PRIMARY KEY (resourceID),

FOREIGN KEY (resourceID) REFERENCES resource(resourceID) ON UPDATE CASCADE ON

DELETE CASCADE

);

go

create table Immovable(

ResourceID varchar (10) NOT NULL,

Capacity INT,

RoomId INT,

BuildingId varchar(10),

Campus varchar(20),

primary key (ResourceID),

foreign key (ResourceID) references resource(ResourceID) on update cascade on delete CASCADE

);

go

CREATE TABLE Loan(

loanID INT,

dateTimeLoaned DATETIME NOT NULL,

dateTimeReturned DATEtime,

dateTimeDue DATEtime NOT NULL,

resourceID VARCHAR(10) NOT NULL,

memberID VARCHAR(15) NOT NULL,

PRIMARY KEY (loanID),

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

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

);

go

create table Reservation(

ReservationID int NOT NULL,

MemberId VARCHAR(15) NOT NULL,

ResourceId VARCHAR(10) NOT NULL,

DateTimeRequired datetime NOT NULL,

DateTimeDue datetime 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,

);

go

create table Acquisition(

AcquisitionID INT NOT NULL,

ItemName VARCHAR(20) NOT NULL,

Status VarChar(20) DEFAULT 'available',

fundCode VarChar(20) NOT NULL,

VendorCode VarChar(20) NOT NULL,

Price money NOT NULL,

Make VarChar(20),

Manufacturer VarChar(20),

Model VarChar(20),

Year Numeric(4),

Description VarChar(300),

Urgency VarChar(20),

Notes VarChar(300),

MemberId VarChar(15),

PRIMARY KEY (AcquisitionID),

foreign key (MemberID) references Member(MemberID) on update cascade on delete no action

);

go

CREATE TABLE privilege(

privilegeID varchar(20) NOT NULL,

name varchar(20) NOT NULL,

description varchar(300) NOT NULL,

CategoryID int NOT NULL,

maxItems int NOT NULL,

PRIMARY KEY (privilegeID),

foreign key (CategoryId) references Category(CategoryID) on update cascade on delete no action,

);

go

CREATE TABLE CourseDetails(

CourseId VARCHAR(8) NOT NULL,

Name VARCHAR(50),

PRIMARY KEY (CourseId),

);

go

CREATE TABLE CourseOfferingNew(

offeringID VarChar(8) NOT NULL,

courseID VarChar(8) NOT NULL,

yearOffered numeric(4),

semesterOffered int NOT NULL,

dateBegins date NOT NULL,

dateEnds date NOT NULL,

PRIMARY KEY (OfferingId),

);

go

CREATE TABLE student\_CourseOffering(

MemberID varchar(15) NOT NULL,

OfferingID varchar(8) NOT NULL,

PRIMARY KEY (MemberID, OfferingID),

FOREIGN KEY (MemberID) REFERENCES studentMember(MemberID) ON UPDATE CASCADE ON DELETE NO ACTION,

FOREIGN KEY (OfferingID) REFERENCES courseOfferingNEW(offeringID) ON UPDATE CASCADE ON DELETE NO ACTION

);

go

CREATE TABLE CourseOffering\_Privilege(

privilegeID varchar(20) NOT NULL,

OfferingID varchar(8) NOT NULL,

PRIMARY KEY (privilegeID, OfferingID),

FOREIGN KEY (privilegeID) REFERENCES privilege(privilegeID) ON UPDATE CASCADE ON DELETE NO ACTION,

FOREIGN KEY (OfferingID) REFERENCES courseOfferingNEW(offeringID) ON UPDATE CASCADE ON DELETE NO ACTION

);

go

--Input proper data:

INSERT INTO Member VALUES('C3212345', 'Hassan', 'Ali', 0478612345, 'active', '2 Jannat Al-Baqi, Medina, Saudi Arabia', 'Student of Uon', 'C3212345@uon.edu.au')

INSERT INTO Member VALUES('C3212344', 'Hussain', 'Ali', 0478612355, 'active', '1 Karbala 56001, Iraq', 'Student of Uon', 'C3212344@uon.edu.au')

INSERT INTO Member VALUES('C3212333', 'Abbas', 'Ali', 0478612333, 'active', '2 Karbala 56001, Iraq', 'Student of Uon', 'C3212333@uon.edu.au')

INSERT INTO Member VALUES('C3212222', 'Zaynab', 'Ali', 0478672333, 'active', 'El Sayeda Zeinab, Cairo Governorate, Egypt', 'student of uon', 'C3212222@uon.edu.au')

INSERT INTO Member VALUES('C1111111', 'Mohammed', 'Mustafa', 0478672111, 'active', 'Al Haram, Medina 42311, Saudi Arabia', 'Teacher of Uon callaghan and Nuspace', 'C1111111@uon.edu.au')

INSERT INTO Member VALUES('C1111112', 'Ali', 'Murtaza', 0478672112, 'active', 'Imam Sadiq St, Najaf, Iraq', 'Teacher of callaghan', 'C1111112@uon.edu.au')

INSERT INTO Member VALUES('C1111113', 'Fatema', 'Mohammed', 0478672113, 'active', '1 Jannat Al-Baqi, Medina, Saudi Arabia', 'Teacher at uon nuspace', 'C1111113@uon.edu.au')

INSERT INTO Member VALUES('C1111114', 'Adam', 'Ibrahimi', 0431338123, 'expire', '1 University Dr, Callaghan NSW 2308', 'Personal Trainer at the Forum uon', 'C1111114@uon.edu.au')

GO

INSERT INTO StudentMember VALUES('C3212345', 11)

INSERT INTO StudentMember VALUES('C3212344', 12)

INSERT INTO StudentMember VALUES('C3212333', 4)

INSERT INTO StudentMember VALUES('C3212222', 1)

/\*Select \* from Member M, StudentMember s

where m.memberID = s.memberID ; \*/

INSERT INTO StaffMember VALUES('C1111111', 'Head Teacher', 'Admin')

INSERT INTO StaffMember VALUES('C1111112', 'Second Head Teacher', 'Admin')

INSERT INTO StaffMember VALUES('C1111113', 'Teacher', 'Not Admin')

INSERT INTO StaffMember VALUES('C1111114', 'Personal Trainer', 'Not Admin')

GO

INSERT INTO Category VALUES(12345, 'Camera', 'Photography and Videography equipment', 'one week')

INSERT INTO Category VALUES(54321, 'TextBooks', 'TextBook resources', 'one week')

INSERT INTO Category VALUES(53214, 'Calculators', 'Casio mathematical calculators', '2 days')

INSERT INTO Category VALUES(82234, 'BasketBalls', 'Spalding Balls for Basketball', '5 hours'),

(83465, 'Speakers', 'Sony Speaker system (Purple)', '2 Days')

GO

INSERT INTO Resource VALUES('ABC1234567', 'Canon EOS 3000D DSLR Camera', 'occupied', 12345)

INSERT INTO Resource(resourceID, Description, categoryID) VALUES('ABC2234567', 'Introduction to Algebra', 54321)

INSERT INTO Resource VALUES('DEF4543465', 'Casio fx-82AU PLUS', 'damaged', 53214)

INSERT INTO Resource(resourceID, Description, categoryID) VALUES('BBL9999999', 'BasketBall 001', 82234)

INSERT INTO Resource(resourceID, Description) VALUES('EVT4985367', 'ComputerLab')

INSERT INTO Resource(resourceID, Description, Status) VALUES('SRT4298976', 'Health and Disease Research Lab', 'Occupied')

INSERT INTO Resource(resourceID, Description, Status) VALUES('GHD4534636', 'Lecture Teather', 'Occupied')

INSERT INTO Resource(resourceID, Description, Status) VALUES('NUG3408963', 'Mathematics Room', 'Occupied')

GO

INSERT INTO Moveable VALUES('ABC1234567', 'DSLR Camera', 'EOS', 'Canon', '3000D', 2015, 5640.35)

INSERT INTO Moveable(resourceID, name, AssetValue) VALUES('ABC2234567', 'Introduction to Algebra', 346.25)

INSERT INTO Moveable VALUES('DEF4543465', 'Casio Scientifc and graphic calculator', 'Plus', 'Casio', 'fx-82AU', 2013, 50.00)

INSERT INTO Moveable(resourceID, name, Manufacturer, AssetValue) VALUES('BBL9999999', 'BasketBall 001', 'Spalding', 50.00)

GO

INSERT INTO Immovable VALUES('EVT4985367', 30, 435, 'CAL', 'NUSpace')

INSERT INTO Immovable VALUES('SRT4298976', 30, 685, 'HDL', 'Callaghan')

INSERT INTO Immovable VALUES('GHD4534636', 500, 112, 'LA2', 'Callaghan')

INSERT INTO Immovable VALUES('NUG3408963', 20, 546, 'MATH03', 'Singapore')

GO

INSERT INTO Loan VALUES(11111, '2022-10-01 01:02:03', '2022-10-02 01:02:43', '2022-10-03 01:02:00', 'ABC1234567', 'C3212345')

INSERT INTO Loan VALUES(11112, '2022-10-03 01:02:03', '2022-10-03 02:02:56', '2022-10-03 10:00:00', 'DEF4543465', 'C3212344')

INSERT INTO Loan VALUES(11113, '2022-10-04 01:02:03', '2022-10-05 01:02:55', '2022-10-06 11:00:00', 'ABC1234567', 'C3212333')

INSERT INTO Loan VALUES(11114, '2022-10-05 01:02:03', '2022-10-06 01:02:15', '2022-10-07 11:00:00', 'DEF4543465', 'C3212222')

INSERT INTO Loan VALUES(11115, '2022-09-06 01:02:03', '2022-09-07 01:02:10', '2022-09-08 04:02:00', 'ABC2234567', 'C3212345')

INSERT INTO Loan VALUES(11116, '2022-08-07 01:02:03', '2022-08-08 01:02:11', '2022-08-09 06:02:00', 'DEF4543465', 'C3212222')

INSERT INTO Loan VALUES(11117, '2022-06-08 01:02:03', '2022-06-09 01:02:22', '2022-06-10 23:59:59', 'BBL9999999', 'C1111114')

INSERT INTO Loan VALUES(11118, '2022-08-09 01:02:03', '2022-08-10 01:02:33', '2022-08-10 10:02:00', 'DEF4543465', 'C3212345')

INSERT INTO Loan VALUES(11119, '2022-10-10 01:02:03', '2022-10-22 01:02:44', '2022-10-11 01:02:03', 'ABC1234567', 'C3212344')

GO

INSERT INTO Reservation VALUES(2342551, 'C1111112', 'NUG3408963', '2021-11-11 12:04:22', '2022-12-11 12:04:22')

INSERT INTO Reservation VALUES(2342555, 'C1111113', 'GHD4534636', '2021-11-11 12:04:00', '2022-12-11 12:04:00')

INSERT INTO Reservation VALUES(2342251, 'C1111111', 'SRT4298976', '2022-11-11 12:03:00', '2022-12-13 11:03:00')

INSERT INTO Reservation VALUES(7950549, 'C1111112', 'EVT4985367', '2022-11-20 01:20:00', '2022-12-20 04:04:22')

INSERT INTO Reservation VALUES(7951549, 'C1111113', 'SRT4298976', '2022-05-01 14:20:00', '2022-05-02 04:04:22')

INSERT INTO Reservation VALUES(7952549, 'C1111112', 'EVT4985367', '2022-06-05 06:20:00', '2022-06-06 06:00:00')

INSERT INTO Reservation VALUES(7952849, 'C1111112', 'EVT4985367', '2022-09-19 05:50:00', '2022-09-20 06:00:00')

GO

INSERT INTO Acquisition(AcquisitionID, ItemName, [Status], fundCode, VendorCode, Price, Urgency, Notes, MemberId) VALUES(278346, 'Coffee Cups', 'available', 'HJF435', 'FES345', 23.30, 'high', 'Need for annual staff meeting', 'C1111111')

INSERT INTO Acquisition VALUES(2352837, 'Microphone', 'Available', 'MIC123', 'MIC321', 120.50, 'ABC123', 'Sony', 'XYZ321', 2022, 'Sony Wireless Microphone (Black)', 'Low', 'Need as one is broken', 'C1111113')

INSERT INTO Acquisition VALUES(4352875, 'Speaker', 'Available', 'SPK232', 'SPK342', 500, 'SWF234', 'Rhode', 'POP123', 2022, 'Rhode Subwoofer Speaker system (Green)', 'High', 'Need Very urgent', 'C1111114')

INSERT INTO Acquisition VALUES(4375863, 'Printer', 'Available', 'BEU425', 'EIF325', 1200.00, 'GJB423', 'NOCAP', 'PRI888', 2022, 'All fax no cap', 'Low', 'Replacemnt printer for staff room', 'C1111114')

GO

INSERT INTO privilege VALUES('PRIV54321111', 'Basketball Team', 'Basketball resources for the Unis home team', 82234, 20)

INSERT INTO privilege VALUES('PRIV54322222', 'MATH1110', 'Access to Calculators for Math class', 82234, 20)

INSERT INTO privilege VALUES('PRIV54333333', 'PHTO1111', 'Access to Cameras for Photography class', 12345, 5)

INSERT INTO privilege VALUES('PRIV33333333', 'ENGG2500', 'Access to Enginering textbooks', 54321, 7),

('PRIV44444444', 'PHTO1111', 'Photography class speaker loan', 83465, 3)

GO

INSERT INTO CourseDetails VALUES('MATH1110', 'Maths for Engineering')

INSERT INTO CourseDetails VALUES('SENG2250', 'System and Netowrk Security')

INSERT INTO CourseDetails VALUES('SENG1540', 'Software Engineering and Management')

INSERT INTO CourseDetails VALUES('PHTO1111', 'Introduction to Photographics techniques')

GO

INSERT INTO CourseOfferingNew VALUES('EUE235', 'MATH1110', 2022, 2, '01-01-2022', '02-02-2022')

INSERT INTO CourseOfferingNew VALUES('OUB976', 'BBAL2323', 2022, 1, '10-10-2022', '12-12-2022')

INSERT INTO CourseOfferingNew VALUES('BAL324', 'SENG2500', 2021, 1, '11-11-2021', '12-12-2021')

INSERT INTO CourseOfferingNew VALUES('UHB833', 'PHTO1111', 2022, 2, '01-01-2022', '01-03-2022')

GO

INSERT INTO student\_CourseOffering VALUES('C3212345', 'EUE235')

INSERT INTO student\_CourseOffering VALUES('C3212344', 'EUE235')

INSERT INTO student\_CourseOffering VALUES('C3212333', 'EUE235')

INSERT INTO student\_CourseOffering VALUES('C3212345', 'UHB833')

GO

INSERT INTO CourseOffering\_Privilege VALUES('PRIV54321111', 'OUB976')

INSERT INTO CourseOffering\_Privilege VALUES('PRIV54322222', 'EUE235')

INSERT INTO CourseOffering\_Privilege VALUES('PRIV54333333', 'UHB833')

INSERT INTO CourseOffering\_Privilege VALUES('PRIV33333333', 'BAL324')

INSERT INTO CourseOffering\_Privilege VALUES('PRIV44444444', 'UHB833')

GO

--Q1: Print the name of student(s) who has/have enrolled in the course with course id xxx. (5)

SELECT m.FirstName, m.LastName, CON.courseID AS [Enrolled In:]

FROM Member m, studentMember student, CourseOfferingNew con, student\_CourseOffering studenttocourse

where m.memberID = student.MemberID and student.MemberID = studenttocourse.MemberID and studenttocourse.OfferingID = con.offeringID and con.courseID = 'MATH1110'

--Q2: Print the maximal number of speakers that the student with name Hassan Ali can borrow. The student is enrolled in the course with course id PHTO1111. Note: speaker is a category. (5)

SELECT mem.FirstName, mem.LastName, c.name AS [Category], priv.maxItems AS [Maximal number that can be loaned]

FROM privilege priv, member mem, studentMember student, student\_CourseOffering studenttocourse, courseOffering\_Privilege coursetopriv, courseOfferingNew con, Category c

where mem.memberID = student.memberID and mem.FirstName = 'Hassan' and mem.LastName = 'Ali' and student.memberID = studenttocourse.memberID and studenttocourse.OfferingID = coursetopriv.offeringID and coursetopriv.privilegeID = priv.privilegeID and c.CategoryId = priv.CategoryID and c.name = 'Speakers' and coursetopriv.offeringID = con.offeringID and con.courseID = 'PHTO1111'

--Q3: For a staff member with id number C1111112, print his/her name and phone number, the total number of reservations that the staff had made in 2022. (10)

SELECT mem.memberID, mem.firstname, mem.LastName, mem.phoneNo, count( distinct res.reservationID) as [Total Reservations in 2022]

FROM member mem, staffMember staff, acquisition acq, reservation res

where mem.MemberID = Staff.memberID and staff.MemberID = 'C1111112' and (mem.memberID = res.memberID and res.DateTimeRequired > '2021-12-31 23:59:59' and res.DateTimeRequired < '2023-01-01 00:00:00')

group by mem.memberID, mem.firstname, mem.LastName, mem.phoneNo

--Q4: Print the name(s) of the student member(s) who has/have borrowed the category with the name of camera, of which the model is xxx, in this year. Note: camera is a category, model attribute must be in movable table, and “this year” must be decided by the system. (10)

SELECT mem.Firstname, mem.Lastname

FROM Member mem, studentMember student, loan l, moveable m, category c, resource r

WHERE student.memberID = l.memberID and mem.memberID = student.memberID and l.resourceID = m.ResourceID and m.resourceID = r.resourceID and r.categoryID = c.categoryId and c.name = 'Camera' and m.model = '3000D' and year(l.dateTimeLoaned) = Year(SYSDATETIME ())

--Q5: Find the moveable resource that is the mostly loaned in current month. Print the resource id and resource name. Note: “current month” must be decided by the system. (10)

Select r.resourceID, r.Name

From Loan l, moveable r

Where l.resourceID = r.resourceID and Month(l.dateTimeLoaned) = month(SYSDATETIME ())

And year(l.dateTimeLoaned) = Year(SYSDATETIME ())

GROUP BY r.resourceID, r.name

HAVING COUNT(\*) >=ALL

      (SELECT COUNT (\*)

      FROM Loan l, Moveable mr

      WHERE l.resourceID = mr.resourceID and Month(l.dateTimeLoaned) = month(SYSDATETIME ())

GROUP BY mr.resourceID)

/\*

--Q6: For each of the three days, including May 1, 2022, June 5, 2022, and September 19, 2022, print the date, the name of the room, and the total number of reservations made for the room xxx on each day. (10)

\*/

SELECT res.DateTimerequired AS [Date], r.[Description] as [Room Name], count(ReservationID) as [Number of reservations]

FROM reservation res, Resource R, Immovable I

where res.resourceID = R.resourceID and r.resourceId = I.ResourceID and I.RoomId = 685 and ((res.DateTimerequired > '2022-04-30 23:59:59' and res.DateTimerequired < '2022-05-01 23:59:59') or (res.DateTimerequired > '2022-06-04 23:59:59' and res.DateTimerequired < '2022-06-05 23:59:59') or (res.DateTimerequired > '2022-09-18 23:59:59' and res.DateTimerequired < '2022-09-19 23:59:59'))

group by res.DateTimerequired, r.[Description]