

COMP122 – 009

Term Project – Database for MyStudyRoom

Professor:

Sarajlic, Zoran

Student:

Rafael Rodrigues de Aguiar - ID: 301041266

Tiago Franco Finotti – ID: 301041366

Contents

[Case Study: MyStudyRoom 3](#_Toc16114529)

[Introduction 3](#_Toc16114530)

[Project Scope: 3](#_Toc16114531)

[Business Requirements: 3](#_Toc16114532)

[Additional Requirements: 4](#_Toc16114533)

[Entities and relations: 4](#_Toc16114534)

[Normalization: 4](#_Toc16114535)

[Entity-Relationship Model: 5](#_Toc16114536)

[Specific data input formats: 6](#_Toc16114537)

[CHECKLIST – Database Term project: 7](#_Toc16114538)

[References: 8](#_Toc16114539)

# Case Study: MyStudyRoom

## Introduction

The study rooms booking software currently implemented by the Centennial College's Libraries provides good support for desktop web browsers but shows significant limitations to mobile users because the pages are not designed for smartphones' screens, nor are conceived for supporting touch interactions. This inevitably leads to critical usability issues that can directly hinder the fruition of the service. The MyStudyRoom App is a mobile application that aims to solve the abovementioned problem by offering to Centennial College students an easy-to-use and enticing interface that will enable them to easily schedule and manage study rooms.

## Project Scope:

MyStudyRoom is a mobile application available for Android and iOS smartphones, that will assist college students by providing a user-friendly interface through which will be possible to see all available rooms in a specific time-span, choose a suitable one and book it quickly and easily. Another important feature included will be a share button that enables the student who booked the room, to share with others some of the details of the reservation such as room location, date, time and even the Google Calendar event associated. Furthermore, the end-user will receive notifications as a reminder of their meetings; a pop-up message will ask them if they can make it or if they prefer to cancel the reservation. All the above-mentioned functionalities will enhance the booking service; thus, contributing in a better management of the libraries' spaces and availabilities.

The Scope of this documentation is to document and implement an Oracle database that attends to the business requirements for this application.

## Business Requirements:

* User can “book” the room on main screen using date/time and selecting the available desired room
* User can filter the available rooms by number of seats, multimedia capabilities and room location
* User can “cancel” the booked room as desired, by selecting his/her booking and canceling the reservation.
* User can “share reservation details” with other users, since these are multi-person study rooms. The share will be performed by searching the name or e-mail information for the users on the existing users table and sending a single notification to all listed users.
* Notified users must be stored on a table so when a reservation is cancelled, all users from the reservation must be notified about the cancelation.
* System will add the user to a waitlist whenever there is no available room for the desired date/time and/or room specifications
* System will create a reservation for a waitlisted user when a room with the required specifications is made available and the user accepts it
* System books the room for 1-hour slots automatically, in order for all students to have equal access to the rooms. A student can book more than 1 hour by creating multiple reservations.

## Additional Requirements:

* There are 3 types of users: Student, Faculty and Librarian
  + Student – Active Centennial Students
  + Faculty – Employees who are currently teaching courses in the present semester
  + Librarian – Employee from library who can schedule and cancel room reservations in behalf of the students

## Entities and relations:

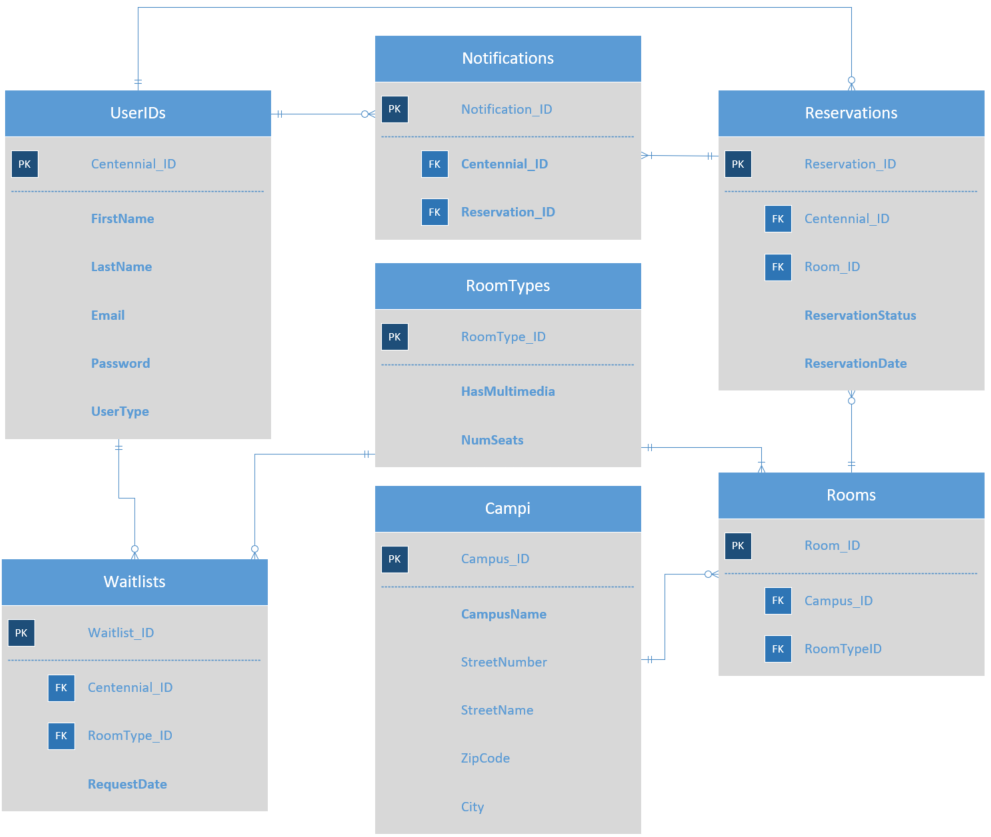
By the business requirements, 5 entities are easily identified:

* Users
  + Primary key: Centennial\_ID
  + Relationships: Reservations, Notifications and Wailists
* Reservations
  + Primary key: Reservation\_ID
  + Relationships: Users, Rooms, Notifications
* Room Types
  + Primary Key: RoomType\_ID
  + Relationships: Waitlists, Rooms
* Notifications
  + Primary key: Notification\_ID
  + Relationships: Users and Reservations
* Rooms
  + Primary key: Room\_ID
  + Relationships: Reservations and Waitlists
    - Found a N-to-N relationship, fixed with a RoomTypes entitiy between them.
    - Found a breach on 3NF (dependency on non-key attribute), fixed with new table Campi.
* Waitlists
  + Primary key: Waitlist\_ID
  + Relationships: Users and Rooms
    - Found a N-to-N relationship, fixed with a RoomTypes entity between them.
* Campi
  + Primary key: Campus\_ID
  + Relationships: Rooms

## Normalization:

* 1NF -> No “divisible” values on the rows
* 2NF -> No data redundancy – no N-to-N relationships
* 3NF -> No “non-key” attributes depend on other “non-key” attributes

## Entity-Relationship Model:



## Specific data input formats:

To attend business requirements and promote quality of the data stored on the database, some fields will have specific data input formats, with some of them enEnforced by constraints.

Table: Users

* Usertype: 9-character string with three and only three possible values: student, faculty or librarian; enEnforced through constraint.

Table: Reservations

* Reservation\_ID: 10-character number in the following format: YYYYMMXXXX – where YYYY is the 4-digit year, MM is a 2-digit month and XXXX is the reservation number from 0001 to 9999.
* Reservation\_Status: 1-character string with two and only two possible contents: “Y” – for valid reservation and “N” – for canceled reservation. Enforced through constraint.

Table: Rooms

* Room\_ID: Primary Key, 5-character string, will have the format: LocationFloor-RoomNumber. I.e.: L2-34; Enforced through constraint.

Table: Campi

* Campus\_ID: Primary Key, 5-character string with the format: [a-z][a-z][a-z][0-9][0-9]. I.e.: SCB01 as in Scarborough Campus 01, PCK01 as in Pickering Campus 01.
* Campus\_Name: 60-character string with the full, unique, Campus Name. I.e.: Centennial College Theatre Arts and Performance

Table: RoomTypes

* RoomType\_ID: 4-digit number, where the first three digits represents the number of available seats (1 – 999), and the last digit represents the multimedia capabilities (0 or 1).
* HasMultimedia: 1-character string with two and only two possible contents: “Y” – for a room with multimedia available and “N” – for rooms with no multimedia available. Enforced through constraint.

Table: Notifications

* Notification\_ID: 11-character number in the following format: YYYYMMXXXXX – where YYYY is the 4-digit year, MM is a 2-digit month and XXXXX is the notification number from 00001 to 99999.

Table: Waitlists

* Waitlist\_ID: 11-character number in the following format: YYYYMMXXXXX – where YYYY is the 4-digit year, MM is a 2-digit month and XXXXX is the waitlist number from 00001 to 99999.

## CHECKLIST – Database Term project:

* 1 - Define scope – what will be the business illustrated – business rules
* 2 – Define entities (minimum 5 maximum 10), primary keys and attributes
* 3 – Map and document relations
* 4 – Normalization – check and execute until 3rd normal form
* 5 – E.R. Model – build
* 6 – create tables and constraints, commit
* 7 – load data into database
* 8 – finish word document
* 9 – provide the .sql file with the commands to create and load the database with data.

# References:

<https://docs.oracle.com/>