A Sports and Recreation Club Database: *a broad specification*

You are going to create a database that will hold details of clubs and organisations involved with recreational activities and their members

The eventual application would support the following functions:

- The ability to register as a user of the system. There are two types of users:
    - End users, and
    - Administrators
- The ability to register a club with all of its details
- The ability to update a club with all of its details (only the owner and administrators can do this)
- The ability to delete a club (only the owner and administrators can do this)
- The ability to search for clubs based on a range of tags, names, location (location should use suburb, town, within radius of location)
- The ability to list details of the club
- The ability to display the list of clubs
- The ability to do the above with members of the club

Although the intention is not to have a fully working application, all of the database functionality needs to be done with the basic functionality

What follows are the basic entities. You need to flesh this out and decide what other entities you need in your data model.

Users	**Club**	**Members**	**Membership**
• Username • Password • Name • Email Address • Date Registered • User Type	• Name • Address • Type • Location (map coordinates) • Number Members • Creation Date • Recruiting Members • Contact • Description • Facebook Page • Twitter Account	• Name • Address • Email Address • Facebook • Twitter • Interests	• Club • Member • When Joined • Last Paid

Feel free to extend the model. For example, you could allow people to rate the clubs or give them "likes"

You need to think of sensible data types and attributes.

Members can belong to more than one club - how do you model this?
Think of the constraints that need to be added to the database

Partial deliverable 1
By the end of Week 8, you should have created either a UML diagram or an ERD and the final schema defined - i.e. the DDL scripts to create the database and populate with some test data. You don't need to submit it, just have it completed.

Project Specification II

By this stage, you should have developed the design of the database including the data definition language to create the database and tables.

Have you checked that the database is in BCNF and 4th Normal Form? If not, then check that now.

You will need to write a 1-2 page description of why you designed the database as you did and the process you went through to make sure it was in BCNF and 4NF.

Functionality
Now we can move onto the actual functionality of the system.

1. For all of the tables you have in the database, create sql scripts that will populate the tables with test data
2. Add tables to your model that will provide overall statistics for the system. The tables should have totals for:
    1. Number of users
    2. Number of clubs broken down by type
    3. Number of members per club

Create database triggers that will update the totals whenever something is inserted or deleted that will affect these totals

For the application functionality, you can either write a command line application in any language that interacts with the database, or one with a user interface. Either web-based, mobile or desk-top based.

You can use the underlying object relational model if the language or framework you are using supports it. You do not have to write the database code using SQL but you may if you wish. Remember, this is not a programming test - but it is useful to be able to use the database functionality in a context.

As a minimum, the application should:

1. Provide a login - don't worry how you check passwords - we are not concerned with security
2. Allow registration of a user of the system. There are two types of users: [a] end users and [b] administrators
3. Register a club with all of its details
4. Search for clubs based on a range of tags, names, location (location should use suburb, town, within radius of location)
5. The ability to list details of the club

Deliverables and Marking Guide

There are 100 marks in total for this project. The project is worth 20% of the overall unit mark

1. 40 are for the database design, setup scripts and documentation
2. 10 are for the database install scripts
3. 50 is for the functionality accessing the database (the app)

You will be asked to submit this work in a zip file via cssubmit. A document called team.txt needs to specify the student numbers of the students involved in producing the work.

Only one member of the team is required to submit the work. Running examples of the code should be detailed in the submission along with instructions about how to access, view, test.

Final submission date is Friday 18th October at 17:00

Item	**Comments**	**Marks**	**Marking Notes**
UML or ERD Diagram of Database Design (pdf format)	Use any suitable drawing tool to produce a diagram of the data model	25	-2 for each missing or incorrect table relationship multiplicity -5 for missing entity -5 for incorrect relationship -5 for not being normalised
Data dictionary	Document detailing individual tables and their attributes including data types	5	-1 for any invalid data type or missing attribute
Install Scripts	Document detailing individual tables and their attributes including data types	5	-1 missing entities or attributes -1 runtime errors
Database design document	1 - 2 page document detailing any thoughts behind the design including the process of normalisation	5	-2 missing or inadequate discussion on normalisation
Database Install Scripts	SQL scripts that will install dummy data into the tables	10	-2 missing script for installation of a given entity -2 any runtime errors
Application	Application that demonstrates interaction with the database and provides the minimum functionality detailed in the specification. You will submit the source code, instructions on how to run and a description of the different components. If the application is a web app, you can point to the running app	50	-5 missing function -2 any runtime errors -10 App doesn't run

A bonus of 5 extra overall percentage marks for the course is available if the application shows originality or effort - or is just a really nice app