IN605 Data Model Assignment

**Learning Outcomes**

At the successful completion of this course, students will be able to:

1. Apply the normalization rules to the creation of a relational database.
2. Construct an appropriate data and database (ERD) model for a specified problem and build the corresponding database

# Overview

Design use cases and a conceptual data model for the development of a database for a local sports club. This assignment is worth 20% of your final grade. This assignment may be done in pairs.

The assignment has 100 marks. It is due at **11:59pm Friday 17th September.**  A late penalty of 10 marks per day is applied for each day the assignment is late (00:01am 18th September counts as a day!)

# Deliverables

A ***single*** document should be submitted to GitHub Classroom (you will be sent a link), (yes there will be marks deducted for multiple files!)

* Use cases (30%)
* A conceptual data model (60%)
* Data dictionary (10%)

**The problem:**

We organise a local team competition and we want an information system that will

help us record the details and results of matches in the competition. The system

needs to handle only one season of competition.

We have many players that are formed into teams. Each player can be a member of

only one team at one time. A team can have up to 18 players. Players may join and

leave a team, so we need to know their joining date and leaving date of any team.

We need to know the player’s name, email address, phone number and date of birth.

Each team has a name, a contact person, contact person phone number and a contact

email address.

We need to keep a record of injuries for ACC purposes. During a match, players may

be injured. We need to know which player was injured, the time when they were

injured and an injury code. Injury codes and the injuries they refer to are a standard

list received from ACC, so the system must be able to store that list.

In a match two teams play against each other at a ground. We have a limited number

of grounds; we need to know the name and address of the ground so we can inform

teams. At match time not all 18 players in the team can play; the team must provide a

squad of players. A squad is made up of between 11 and 16 players. We need to keep

a record of who was in each squad at match time. During a match, players can score

goals. We need to know which player scored the goals and the time in the match

when they scored. We also need to know the shirt number the player was wearing for

the match. It is possible that same two teams will play against each other more than

once in a season.

We also have a list of umpires. We need to know which umpires (there are two per

match) that will be controlling the match. The system must hold the date and time the

match will start. Ask the lecturer follow up questions if you require more information, a discussion document will be provided on Teams (DesignAssignmentFAQ.doc).

# Requirements – Learning Outcomes 1, 2

From this description and any discussion, identify and document the use cases and create a suitable data model. I recommend that you use the “seven question” processes discussed in week two. However your final report should be formatted with the following sections:

* Title page
* Main objectives
* Use Cases
* Data Model
* Data Dictionary

## Title Page

This should be a single page clearly indicating the course name, the assignment name and your name.

## Main Objective

Identify the main objectives of this database. This should be a maximum of a page detailing what the database is designed to do and stipulating anything that you wish noted as being explicitly ***not*** in scope.

## Use Cases

Develop use cases (diagram and explanatory text) for the tasks outlined in the interview.

## Conceptual Data Model

Design a conceptual model of the data used in this organisation.

You should:

* Use the ERD diagram’s as covered in this course. Use crows-feet to indicate optionality & cardinality.
* Show the class and attributes in a box. It is not necessary to show a third section which indicates methods
* Include verbs/descriptions on the relationship line when it is needed.

## Data Dictionary

Create a data dictionary for each class. This should take the form of:

* The table name (major heading – bold)
* A separate line for each attribute on which you should identify:
  + the attribute name
  + the attribute data type – be generic and descriptive (‘String’ rather than varchar(20), ‘Integer’ rather than smallint, longint, etc). This may be a general description rather than a standard datatype (eg “graphic”, “audio file” etc)
  + A short description or explanation. This is not necessary for all attributes – include where the name, on its own, may not explain the purpose or use of the field.

Marking Rubric

This is given as an indicative guide. The detail may be adjusted at any stage.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CATEGORY | Perfect  100% | Excellent  90% | Good  75% | Satisfactory  55% | Needs Improvement  30% | Attempted  15% |
| Objectives | 10) Clear comprehensive yet concise description of the major objectives - no issues at all | 9) Clear comprehensive yet concise description of the major objectives | 7.5) Good coverage of the objectives. Possibly unclear about certain issues or not concise. | 5.5) Reasonable description of objectives. Missing or unclear about some issues. Or not concise | 3) The objectives were not clearly or adequately described | 1.5) Real attempt is evident but is not adequate |
| Use Case | 20) Comprehensive documentation of actors and tasks with excellent description – no issues at all | 18) Comprehensive documentation of actors and tasks with excellent description | 15) Good set of actors and tasks are identified. Explanations are clear | 10) Adequate identification of actors and classes. Some gaps evident. Descriptions lack completeness | 6) Insufficient identification of actors, tasks and/or inadequate descriptions | 3) Real attempt is evident but is not adequate |
| Classes | 30) Comprehensive set of classes have been identified. No redundancy. – no issues at all | 27) Comprehensive set of classes have been identified. No redundancy.  Sub-types must be well designed | 22.5) Good set of classes. Little redundancy. 3NF should not be difficult.  Reasonable identification of subtypes | 17) Reasonable set of classes. Some redundancy. Substantial revision to move to 3NF at Logical level | 10) Insufficient identification of classes | 4.5) Real attempt is evident but is not adequate |
| Relationships | 30) Excellent identification of relationships. Cardinalities are well designed | 27) Good identification of relationships. Cardinalities are well designed | 22.5) Good identification of relationships and cardinality. | 17) Reasonable identification of relationships. Some problems with relationships and/or cardinality | 10) Inadequate relationships developed | 4.5) Real attempt is evident but is not adequate |
| Data Dictionary | 10) Excellent identification of attributes and data types with very clear descriptions – no issues at all | 9) Excellent identification of attributes and data types with very clear descriptions | 7.5) Reasonable identification and definition of attributes | 5.5) Attributes are adequately identified | 3) Poor selection or definition of attributes | 1.5) Real attempt is evident but is not adequate |