IFB130 Databases Management
Project Part A

Project Overview
The IFB130 project gives you an opportunity to apply all the concepts and skills you acquire in the unit to a "realistic" database design scenario and reflect on the data requirements of an organisation.

The submission is divided into two parts due at different times during the semester. These parts will cover:
A. Design of a Database
B. Creation and Use of a Database

The Task for Part A
You are to design a solution for a database management system that addresses the needs of an online augmented reality game.

In this part you will design a database by:
1. Creating an ER model for the database based on a use case;
2. Creating a relational schema for the database; and,
3. Validating the relational schema for the database using normalisation.

Weighting
Part A is worth 15 marks, for 15% of the unit.

Groups
This assignment can be done individually or in pairs.

If you choose to work in a pair, only one student is to submit the assignment. Please provide the name and student number of the person you worked with on the front page. No consideration will be given to students who claim they did more work in their pair than the other student because this assignment can be done individually.

Doing the assignment in undeclared groups, or groups larger than two students, will be treated as plagiarism. Pairs that work together and then split due to difficulties must not submit any of the same work, or it will be treated as plagiarism.

Due Date
Monday September 4th at 1.59pm (beginning of week 7, before the lecture).

Submission
For this assessment you should submit a professionally presented word document containing your solution to each task. ER models should be created using a computer drawing program, such as draw.io or Lucid Chart. Partial or fully handwritten submissions will not be accepted. This needs to be uploaded to the *TurnItIn* link on Blackboard.

Late Submission
Assessment work submitted after the due date will be marked only with an approved extension (MOPP E/6.8.2). Assessment work submitted after the due date without an approved extension or, where an extension has been granted, after the extended due date, will not be marked and a grade of 1 or 0% will be awarded against the assessment item.

Please note: extensions will not be granted for group submissions, regardless of the reason.

Background

You have been hired by the Brisbane based company Aggity Games. Aggity are passionate about bringing augmented reality games to the masses. After some success with their previous game, Bush Rangers, they are keen to get you on board to design the database for their new game, which is called Treasure Hunters! In Treasure Hunters players will travel to different locations in the real world in the hunt for treasure. Their smartphones will act as a gateway into the treasure hunter world, with real objects taking on different roles within the game, such as beacons that unlock quests or the elusive treasure.

After hours of meetings and a lot of emails the company have agreed on the following scenario. It is now your task to **design the database.**

The Scenario – *Treasure Hunters,* an augmented reality online game.

In order to play the game, each **player** creates their account. They must choose a unique username, provide their name (first and last names), gender (female, male, other or prefer not to disclose), date of birth, email and postal address (street number, street name, suburb, city and postcode) and up to three phone numbers. The date and time of account creation and the player's total points will also be associated with their account.

The game revolves around players embarking on **quests** and hunting for treasure. Multiple players can go on the same quest and find the same treasure. There are hundreds of quests, new quests are added every week and some of the more advanced quests can contain up to five mini quests. Each quest needs to be identifiable using a unique id, and they also have a name. Players initiate quests by finding objects in the real world, which act as beacons, and each quest is assigned a **beacon**. Once the player finds a beacon they will be presented with a story, which provides some background on the quest, and a list of the treasure they must hunt down in order to finish the quest. The database needs to keep track of each player's progress (active, complete, inactive) on each quest.

There are different types of **treasure** (common, uncommon, rare, ultra-rare and elite). The players are directed to a webpage where they can obtain various clues to find the treasure (we only wish to store the web address in our database however, not the clues themselves). Each treasure has one webpage. When a player finds the treasure, they are awarded points. Each piece of treasure rewards a different amount of points. When a player unlocks the list of treasures for a quest, they can see a description for each piece.

Players can use their points to purchase **badges** from virtual **stores**. All stores have a name, and they are only open at certain times of the day, so their opening and closing times need to be recorded.

In order to purchase **badges** from the stores, players can use their points. There are a lot of different badges that players can purchase. To help players decide what to purchase they will be able to read the badge names and descriptions. The price of the badges varies depending on demand, and is re-calculated when a player reaches a store. This means that the price itself is not stored in the database. However, when a player purchases a badge from a store, the store name, date and time of purchase and the cost of the purchase need to be recorded, so that they can see a detailed list of their purchases at any time.

Part A Tasks

Part A requires you to complete a number of sequential tasks to fulfill the requirements of the scenario. In this part you will design a database by:

1. Creating an ER model for the database;
2. Deriving the relational schema for the new database;
3. Demonstrating the constraints of the relational schema through examples; and
4. Verifying a previous (different) relational schema for the database using normalization;

Task 1 [5 marks]

Considering the use case provided, create an **Entity Relationship Diagram** that correctly models the data requirements of *Treasure Hunters.* You must ensure that everything that happens in the game (or in relation to the game) is recorded. Your ER diagram (using UML notation) needs to show:

- the **entity types** (0.5 marks),
- the **binary relationship types** (1 mark),
- the **complex relationship type** (1 mark),
- the **multiplicity** of relationship types (1 mark),
- the **attributes** (and if applicable their **domains**) of entity types (0.5 marks),
- the **attributes** (and if applicable their **domains**) of relationship types (0.5 marks), and
- the **primary key** of each entity type (0.5 marks).

State any assumptions you make.

Task 2 [5 marks]

Derive a **Relational Model** from the conceptual schema you provided in Task 1 (it should match exactly). *For the purpose of this assignment, only derive the part of the database that stores the information about players, quests, beacons and treasures (considering a sub-ERD), and leave aside the information relating to stores and badges*.

1. List all the relations derived from the entity types of the ERD, and their initial attributes. (1 mark)
2. For each relationship type in the sub-ERD justify whether it leads to a new relation, a new attribute in a relation derived from an entity type, or to a merger of two relations derived from two entity types. (1 marks)
3. List all the relations in the final relational model, and all their attributes (1 mark)
4. List all the constraints included in the conceptual schema in addition to the relevant primary keys and foreign keys. (2 marks)

Task 3 [2 marks]

Create four rows of sample data for each of the relations you identified in Task 2.

Based on your relational model and sample data, specify an example of each of the following, and explain why you selected it:

- A delete operation that would run successfully
- An update operation that would run successfully
- An update operation that would <u>not</u> run successfully
- An insert operation that would <u>not</u> run successfully

Task 4 [3 marks]

Aggity Games had an intern who prototyped the database for the players and quests, but they could never implement it, and they have already been told this is really **poor design**. This is what it looks like, with some sample data:

Player

Name	Quests	Points	Address_id
Amy	Q33 Q12 Q25	674	2
Michael	Q54 Q25	78	2
Emily	Q33	NULL	5
Tom	Q45	54	6

Treasure

Name	Description	Quest_id	Quest Name	Beacon
Diamond	The tree's diamonds	Q25	Urban Elves	Toohey Forest Sign
Heart	The Big Gum's heart	Q25	Urban Elves	Toohey Forest Sign
Diamond	The river's diamonds	Q33	Pirates of Brisbane River	City Cat (Broncos)
Gold	The Captain's chest	Q33	Pirates of Brisbane River	City Cat (Broncos)

Address

Id	Street number	Street name	city	postcode
2	15	Queen Street	Brisbane	4000
3	2	Edward Street	Noosa	4230
5	6	Alice Street	Gympie	5533
6	28	George Street	Bundaberg	4332

For each of the 3 relations proposed (Player, Treasure, Address), explain which normal form (0NF, 1NF or 2NF) it is in, and why.