Assignment 3
Full Stack Application Development and Software Workshop 2

Pieter Joubert

April 19, 2023

Contents

1 Introduction

> **Warning!**
>
> Please note that to receive any marks for this Assignment you will need to complete the Quiz on Canvas. Just completing the Tasks will result in ZERO marks being awarded. Also make sure that your database can pass all the tests given at the end of the Assignment before you start the Quiz.

1.1 Case Study

For this Assignment you will be designing and developing a Database for a fantasy base building game. One example of such a game is Townsmen. For this Assignment you do not need to worry about implementing the game at all, we are just focusing on the database.

The game is multiplayer with multiple *players*. Each player has a *base*, in which they can construct various *buildings*. These buildings produce *resources* which can be used to construct other buildings or can be sold for *currency*. As this is designed to be an online game the construction of a building can take several hours or days in real time.

You will be given an initial ERD (Entity Relationship Diagram) as well as a *.zip* file containing test data stored as *.csv* (comma-separated format) files. From this information, and following the Tasks below, you will expand the database design, populate the database with test data, test the data, and answer questions in an online quiz for your final mark.

1.2 Initial ERD

The ERD in Figure 1 is an initial ERD, that includes *some* of the entities required in the database. This ERD also excludes any information regarding the fields of these entities.
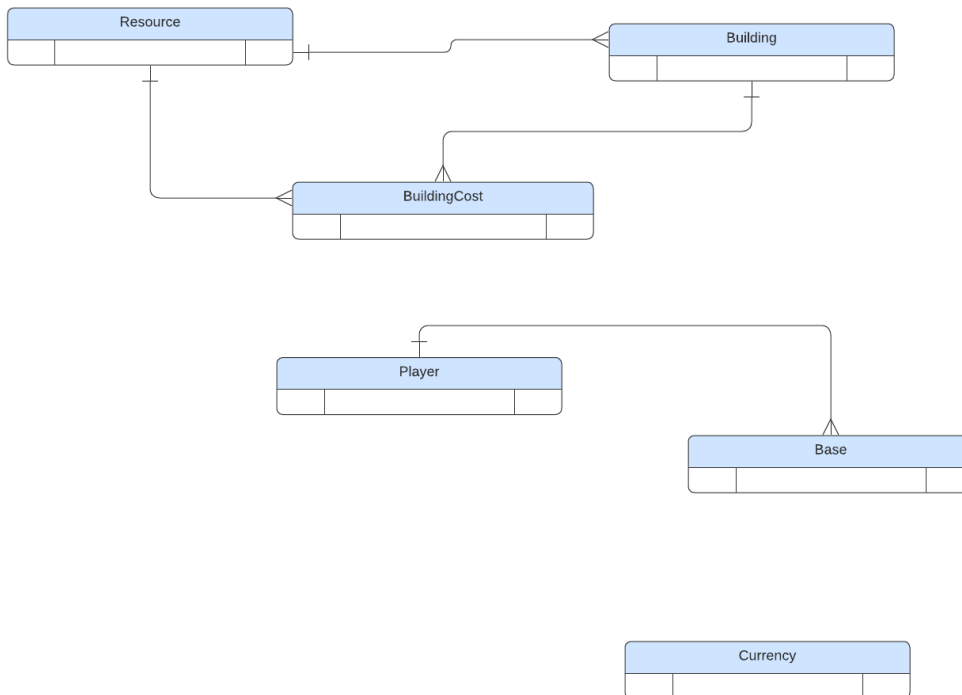

Figure 1: Entity-relationship diagram

2   Tasks

> **Warning!**
>
> Make sure that each step is completed correctly before moving to the next step. If necessary DROP or TRUNCATE entities with incorrect data before fixing your code and carrying on with the Assignment.

2.1   Task 1 - Setup the database

1. Download the *test-data.zip* from Canvas and extract the *csv* files in it.

2. Create a new file named *assignment3.sql*. In this file you will save all the sql code that you write.

3. Create a new postgres user named *fsad*. (Note: If you are using the lab machines, the username and database that has already been created for you is fine to use).

4. Create a new database named *assignment3_<studentId>* (where studentId is your email prefix, e.g. the abc123 part of abc123@student.bham.ac.uk).

5. Make sure the *fsad* user has full privileges on the *assignment3* database.

2.2   Task 2 - Update existing entities

1. For each of the tables listed in Figure 1, create a corresponding table in your database. Use the csv files to guide you.

2. NOTE that the csv files includes a header row, which should NOT be inserted into the database.

3. Ensure that you match the datatype *exactly* to the test-data. HINT: The premium column in the Currency table should be a *bool* datatype, and the build_time column BuildingCost table should be an *interval* datatype.

4. Create any required Primary Key Constraints.

5. Create any required Foreign Key constraints.

2.3   Task 3 - Add new Entities

1. Update your database to add any missing entities based on the csv files.

2. NOTE that the csv files includes a header row, which should NOT be inserted into the database.

3. Ensure that you match the datatype *exactly* to the test-data.

4. Create any required Primary Key Constraints.

5. Create any required Foreign Key constraints.

2.4   Task 4 - Populate test-data

1. For each table import the corresponding values from the csv file.

2. NOTE that the csv files includes a header row, which should NOT be inserted into the database.

3. Ensure that all the tables contain the correct data.

2.5 Task 4 - Create Views

1. Create a view for each building named *<building_info>* that displays the following columns: Name, Description, Resources Generated, Required Gold, Build Time.

2. Create a view for each Player named *<player_name>buildings* that displays all the buildings (by name) the player has built in their base.

3. Create a view for each Player named *<player_>resources* that displays the following columns: Name, Food Amount, Wood Amount, Stone Amount.

4. Create a view for each Player named *<player_>currencies* that displays the following columns: Name, Gold Amount, Ethereal Silver Amount, Diamonds Amount.

2.6 Task 5 - Create Code Block

> **Warning!**
>
> Before you create and execute the code block make a backup of your database. Once the code block has been tested and is working RESTORE the database from your backup.

Create a code (DO) block that essentially *fast-forwards* any outstanding constructions by doing the following:

1. Declares a variable name *test_date*.

2. Assigns this variable to the 10th of April 2023.

3. Checks each row in the construction_queue table.

4. If the construction has NOT been completed (based on the start_date plus build_time being after the test_date) it should be removed from the construction_queue table and inserted into the base_building table.

5. Once the DO block is working correctly restore your database back to its state before running the DO block.

3 Tests

The following tests should give you basic idea if you have set up the database correctly. Passing all the tests DOES NOT mean that your database is setup correctly, but failing ANY of the tests means your database is setup INCORRECTLY.

3.1 Test 1

3.1.1 Query

```
01 |  SELECT * FROM player ORDER BY player_name ASC LIMIT 1;
```

3.1.2 Expected Result

```
01 |   player_id | player_name | player_password
02 |  -----------+-------------+----------------
03 |           6 | AishaKhan   | password6
04 |  (1 row)
```

3.2 Test 2

3.2.1 Query

```
01 |  SELECT COUNT(*) FROM constructionqueue;
```

3.2.2 Expected Result

```
01 |    count
02 |  -------
03 |       4
04 |  (1 row)
```

3.3 Test 3

3.3.1 Query

```
01 |  SELECT building_id, build_time FROM buildingcost ORDER BY build_time DESC LIMIT
          4 ;
```

3.3.2 Expected Result

```
01 |    building_id | build_time
02 |  ------------+------------
03 |             5 | 14 days
04 |             4 | 7 days
05 |             3 | 5 days
06 |             1 | 3 days
07 |  (4 rows)
```

3.4 Test 4

3.4.1 Query

```
01 |  SELECT * from currency WHERE premium = true;
```

3.4.2 Expected Result

```
01 |    currency_id | currency_name | premium
02 |  ------------+---------------+--------
03 |             3 | Diamonds      | t
04 |  (1 row)
```

4 Quiz Instructions

Before attempting the quiz make sure you have completed the following:

1. Most of the required content for assignment 3 has been provided in Weeks 6, 7 and 8 on Canvas. You might need to do some of your own research using the links provided on Canvas to complete some of the Tasks.

2. Complete all the Tasks listed above. This should take you approximately six (6) hours.

3. VERY IMPORTANT: Do not run Task 5 (the DO block) until you get asked to do so in the quiz.

4. Run all the tests listed above, and ensure they pass.

5. Run your *own* tests and make sure they pass. Make sure you do not change the structure or dta of the database when doing so.

6. Ensure you have a backup of the database you can quickly run if you break your database during the quiz.

7. Ensure you understand the structure of the database.

8. Make sure you attempt the quiz on a device that has postgres installed and working properly. Also ensure that the device has the *Assignment3* database already loaded.

9. You will need to write a number of queries on the database to answer the quiz. Make sure you can easily run any required queries.

10. You will have a limited amount of time for the quiz (60 minutes) so make sure you understand how to write psql, sql and plpgsql commands *before* you start the quiz.

11. Please note that NO late submissions will be accepted.