Comp115 Assignment 1: Designing a database

Please download this file and open it in a current version of Adobe Acrobat Reader. This will make the form items active. Viewing it in a web browser will not activate the form.

On this assignment, **you are encouraged to work in groups of up to 6 people.** These assignments are **designed to be done in groups.** Working alone is permissible but not recommended. Please enter the names and login names of all group members below.

Student 1 Name	David Taus	Student 1 login name	dtaus01
Student 2 Name	John Casey	Student 2 login name	jcasey03
Student 3 Name	Sam Duncan	Student 3 login name	sdunca01
Student 4 Name	Rachel Marison	Student 4 login name	rmaris01
Student 5 Name		Student 5 login name	
Student 6 Name		Student 6 login name	

Please paste your solutions to the following problems into the provided regions on the subsequent pages. Please leave the "Grade" and "Comments" boxes for use by your graders. Place your own comments in the "Solution" boxes.

Keep in mind that **you will be quizzed on this assignment after it is due,** so all members of the group should strive to understand the solution.

When you have finished this form, please assign one group member to provide it. To provide it, browse to .the web-based provide submission page for the class.

You can also provide the solution by uploading this form to linux and typing:

```
provide comp115 a1 a1.pdf
```

This is equivalent to using the web form.

Results will be returned electronically to all group members. To see your results, please browse to the web-based grade reporting page.

Accepting assignments in this format is a bold experiment for us, and we would appreciate your comments on difficulties you might have in submitting assignments, understanding your grade, or any other aspects of this grading system.

Problem 1: You have a drink machine to monitor via a database. Drinks are stocked periodically, purchased, and sometimes run out. You want to track:
- Individual sales of drinks, including time of day.
- Times at which each drink sold out.
- Times at which each drink was restocked.

Give the SQL statements that will create tables to track this information. Utilize relationships where possible to avoid storing redundant information. Hint: use your postgresql accounts to test your statements for correct function!

Solution 1

```
CREATE TABLE drinks
(
id SERIAL PRIMARY KEY,
name VARCHAR(50) UNIQUE NOT NULL,
price DECIMAL CHECK(price>=0) NOT NULL,
quantity INT CHECK(quantity>=0) NOT NULL
);

CREATE TABLE sales
(
id SERIAL PRIMARY KEY,
sale_time TIMESTAMP NOT NULL,
drink_id INT REFERENCES drinks(id) NOT NULL
);

CREATE TABLE sellouts
(
id SERIAL PRIMARY KEY,
sellout_sale_id INT REFERENCES sales(id) NOT NULL
);

CREATE TABLE restocks
```

Grade 1

- ◯ Exceeds expectations
- ◯ Meets expectations
- ◯ Minor issues
- ◯ Major problems

Comments 1

Problem 2: In English, describe the model operations (database changes) that would be necessary in order to track the drink machine data in this database.

Solution 2

There are two primary operations: Drink sales and Restocks.

When a drink is sold the number of drinks in the drink table is decremented provided it is 1 or greater, then a sale is added to the sales table. If that sale resulted in that specific drink being sold out the sale is added to the sellout table. This procedure fails when there are zero drinks in the machine as the drink table cannot be decremented.

A restock increases the quantity of a specific drink in the drinks table and adds a row to the restock table.

Grade 2

- ◯ Exceeds expectations
- ◯ Meets expectations
- ◯ Minor issues
- ◯ Major problems

Comments 2

Problem 3: Give SQL templates for each operation above, where $1, $2... denote variables, and define the meanings of those variables. Hint: you might want to look up CURRENT_TIME() and CURRENT_DATE() PostgreSQL functions.

Solution 3

Restock operation:

```
START TRANSACTION;
INSERT INTO restocks (DEFAULT, CURRENT_TIMESTAMP(), $1, $2);
UPDATE drinks SET quantity=quantity+$2 WHERE id=$1;
COMMIT TRANSACTION;
```

Sale operation:

```
START TRANSACTION;
INSERT INTO sales (DEFAULT, CURRENT_TIMESTAMP(), $1);
UPDATE drinks SET quantity=quantity-1 WHERE id=$1;
SELECT quantity,
  CASE WHEN quantity=0 THEN INSERT INTO sellouts (DEFAULT, SELECT LAST(id) from
sales)
  END
FROM drinks WHERE id=$1;
COMMIT TRANSACTION;
```

$1 represents drink_id
$2 represents the quantity restocked

Grade 3

- ◯ Exceeds expectations
- ◯ Meets expectations
- ◯ Minor issues
- ◯ Major problems

Comments 3