# Database Design: Using Postgresql and Web Tools to Implement the Collective Coupons Tracking Database System

This project is to use postgreSQL and web to implement the RBMS application. In this case, you will design a relational database for a company that markets and sells coupons to groups on the Internet. After your database design is completed and correct, you will create database tables and populate them with data. You will produce one form with a subform that allows you to book coupon deals. You will also create queries to help the company answer some important questions: Which deals are available for a specified maximum price? How good a bargain is a deal compared with the regular price? Which deals have at least 100 people signed up so the deal can run? What is the most popular deal? Finally, you will create a query and then a report that lists all customers who have signed up for current deals.

## 1.BACKGROUND

The Internet is a great avenue for marketing and sales, and e-commerce has exploded since the mid-1990s. Recently, new companies have begun to offer discounted coupons to registered members. These coupon deals originate in local businesses, which offer products and services for about half the normal price if the coupon company can get 100 or more people to sign up for a deal. For example, suppose that a local restaurant wants to increase its patronage during the middle of the week, when the dining room usually is not full. The restaurant will offer the coupon company half-price deals on mid-week dinners, such as a $100 gift certificate for $50, if the coupon company can get at least 100 people to sign up for the deal. By using e-mail and Facebook, coupon companies can often find 100 people who want to take advantage of a particular bargain.

You have been hired as a summer intern to create a database tracking system for an Internet coupon company called Collective Coupons. The owner, Charlene Clayton, saw your resume on your university Web site. She thinks that your experience with database design, implementation, and Microsoft Access makes you a perfect fit for the job. Charlene wants you to design the database first and then implement a number of forms, queries, and a report.

On your first day at work, you interview Charlene to find out what she wants from the database:

YOU: What sort of information do you need to keep about your business?

CHARLENE: We have a spreadsheet of all our customers, including their names and e-mail addresses. Each record is designated with a customer number so we can keep them straight. So far we have over 3,000 customers registered with us, so the business is really taking off well! We don't need additional information about the customers because payment for the deals occurs only after 100 people sign up and the deal is accepted. Those payments are handled by another system we already have in place. But we would like an easy way to record who signs up for each deal.

YOU: A form with a subform would be a good way to do that. What information do you need to keep about the deals you solicit from local businesses?

CHARLENE: We are starting to have so many deals that we need to keep them straight. Some deals have the same because the business might run the same deal later in the year. We keep a description of the deal, the city where the deal is offered and run, the available dates and the Price of the deal versus the regular price so customers can see what a bargain it is.

YOU: How do you track who signs up for the deals?

CHARLENE: Again, using a spreadsheet, we keep a list of customers who sign up for the deals. But there isn't an easy way to count all the deals, and we need to be able to do that.

YOU: You can easily count all the customers who sign up for the deals by using a query. What other information do you want to get out of this database?

CHARLENE: I'd like to know what deals are available for a certain price. We like to advertise that price; people e-mail us with that question all the time. Also, we want to be able to calculate the savings that customers are getting with a deal. We would advertise these savings heavily since our customers love good bargains. Of course, we need to know which deals are running, so figuring out which deals have at least 100 sign-ups is essential. We'd also like to know which deals are the most popular so we can ask those businesses for more future deals.

YOU: Queries can handle all those questions easily. Do you need any reports?

CHARLENE: We would like to see a report of everyone who has signed up for the current deals.

## 1. Building the database (15 points)

Use the SQL DDL statements to create the tables required for this project. Please also note that the tables are created in certain order such that by the time when a foreign key needs to be created, the corresponding primary key also need to be created.

(1) First, determine the tables you will need by listing the name of each table and the fields it should contain. Avoid data redundancy. Do not create a field if it can be created by a “calculated field” in a query.
(2)You will need to record the transactions in a separate table. Avoid duplicating data.
(3)You must mark the appropriate primary key field(s) or foreign key field(s) for each table.
(4)Create at least five deals offered by Collective Coupons.
(5)Create more than 100 customers.
(6)Make sure that at least 100 customers sign up for some of the deals.
(7)Appropriately limit the size of the text fields; for example, a customer number does not need the default length of 255 characters.

## 2. PostgreSQL Implementation (55 points)

You need to write SQL queries, stored procedures/functions, and triggers to implement this project. The following requirements and functionalities need to be implemented.

1. (6 points) Write a stored procedure Maximum Price that prompts the user for a maximum price and then display fields for the Description, Location, Deal Price, Available Date, and Ending Date of all deals under the specified price. For example, if you enter $50 at the prompt, your output should resemble that in Table 1, although your data will be different.

Table 1 Maximum Price

| Maximum Price | | | | |
|---|---|---|---|---|
| Description | Location | Deal Price | Available Date | Ending Date |
| 18 Hole Golf Heritage Golf Center | Chicago, IL | **$28.00** | 2012/9/2 | 2012/9/15 |
| Hamburgers, Etc Restaurant | Chicago, IL | **$12.00** | 2012/9/20 | 2012/9/22 |
| Segway City Tour - 3 hours | Chicago, IL | **$35.00** | 2012/9/18 | 2012/9/19 |
| Spas Unlimited | Chicago, IL | **$25.00** | 2012/10/1 | 2012/10/10 |

2. (6 points) Write a stored procedure called Percentage Bargain. List all the available deals, including their Deal Number, Description, Location, Deal Price, and Original Price, and then calculate a percentage in the Bargain column. The bargain is the percentage difference between the Deal Price and Original Price. Your output should look like that in Table 2, although your data will be different.

Table 2 Percentage Bargain

| Percentage Bargain | | | | | |
|---|---|---|---|---|---|
| Deal Number | Description | Location | Deal Price | Original Price | Bargain |
| 101 | Fall Foliage Helicopter Tour | Chicago, IL | 189.00 | 450.00 | 58.00% |
| 102 | 18 Hole Golf Heritage Golf Center | Chicago, IL | 28.00 | 64.00 | 56.25% |
| 103 | Hamburgers, Etc Restaurant | Chicago, IL | 12.00 | 25.00 | 52.00% |
| 104 | Segway City Tour - 3 hours | Chicago, IL | 35.00 | 70.00 | 50.00% |
| 105 | Spas Unlimited | Chicago, IL | 25.00 | 50.00 | 50.00% |

3. (7 points)Create a query called Sign-ups Equal to or Over 100. In this query, you need to determine which deals have at least 100 members signed up. Display columns only for the Deal Number and Description in your output.

4. (8 points) Write a procedure called Most Popular Deal. List a description of the deal and how many people have signed up for it. Sort the output to list the most popular deals first. Note the column heading change from the default setting provided by the query generator. Your output should resemble the format shown in Table 3, but the data will be different.

Table 3 Most Popular Deal

| **Most Popular Deal** | |
|---|---|
| Description | Number Signed Up |
| Spas Unlimited | 129 |
| Fall Foliage Helicopter Tour | 117 |
| Hamburgers, Etc Restaurant | 114 |
| Segway City Tour - 3 hours | 100 |
| 18 Hole Golf Heritage Golf Center | 68 |

5. (8 points) Write two procedures add tuples into the Coupons table and the customers table. As an example, you can use a procedure, say **add_coupons** to add a tuple in the Coupons table, where **Deal Number, Description, Location, etc** are parameters of the procedure. Note that **Available Date** should be set as the current date (use current_timestamp).

6. (15 points) Set that each coupon (deal) could be used by maximum 120 customers. If the number of customers exceed the limitation, your program should perform the following tasks: (1) print a message indicating **the problem,** (b) using **triggers** to extend the maximum number of customers, and (c) using a **log table** to record the update operation on the tables (Add a tuple to the log table automatically whenever any table is modified).

7. (5 points) You need to make your code user friendly by designing and displaying appropriate messages for all exceptions. For example, if someone wants to use a coupon, but entered a non-existent Deal number, your program should report the problem clearly.

## 3. Interface (20 points)

Implement a Web interactive interface using any program language. Your interface program should utilize as many of your database stored procedures/functions as possible.

## 4. Documentation (10 points)

Documentation consists of the following aspects:

1. Each procedure and function and every other object you create for your project needs to be explained clearly regarding its objective and usage.
2. Your code needs to be well documented with in-line comments.

## 5. Hand-ins, Demo and Grading

1. You will also need to submit your source code along with your documentation to the Blackboard.
2. It is required to demonstrate your project to the instructor using tuples created by the instructor.

3. The grading will be based on the quality of your code, the documentation and on how successful of your demo is.