Milestone 4: Database schema diagram and SQL to create and drop the system's tables and constraints in SQL Server

CPS 353: Internet Programming
Web Development Project
Due Wednesday, October 21, 2015 by the start of class

Introduction

The next step to making our mock ups dynamic is to develop a database that will power the UI we have designed. Have a solid database that is flexible enough for unexpected new requirements but also quick to respond to queries is critical in web based systems. In this milestone, you will create a schema diagram for the ecommerce database. You will also write files containing the SQL statements necessary to create and drop the database tables and constraints used by your web application.

Your schema will include details about the following tables:

- Users
- Categories
- Products
- Promotions
- Carts
- CartProducts

Each of these components is discussed in detail in the specifications below.

In addition to these tables, other tables may need to be created to support the product details page that you mocked up. For example, if your product can be sold in 5 different colors, you might need an attributes table with columns like:

AttibuteId	AttributeGroupName (or maybe just AttributeGroupId)	AttributeValue	DisplayOrder
1	Color	Red	3
2	Color	Green	1
3	Size	Small	1
4	Size	Medium	2
5	Size	Large	3
6	Color	Pink	2

Plus a table joining products to attributes:

ProductId	AttributeId
1	1
1	2
2	5
1	3

In this example, Product 1 can only be bought in red or green and size small, Product 2 does not have a color but it is only available in size Large.

These are just examples of other tables that may be needed to support your UI. Do not limit this assignment to the tables described. For example, if your UI has related products on the product detail page, you will need those tables to be included.

Specifications

Users

The users table holds information about the users in our system. Users can be admins or shoppers. A user can only have one shopping cart. The users table has the following required fields:

- An identity column that contains an id that is automatically assigned
- The date and time the user record was created
- The user's name, email and password
- The type of user (admin or shopper)

Categories

The categories table contains a list of categories that will be shown in the menu and is used to categorize products. The categories table has the following required fields:

- An identity column that contains an id that is automatically assigned
- A name column containing the name of the category
- An integer column to define the order the category will appear in the menu

Products

The products table contains all the details about the product. The products table has the following fields, none of which can be null except for some of the product details if they only exist for a subset of your products:

- An identity column that contains an id that is automatically assigned
- The name, description, image url, price and other details about your product. Each field that is in your product details mockup will need a column to contain that data
- The category id that the product belongs to
- A bit to indicate if it is the featured product so that it will be displayed on the home page

Promotions

The promotions table contains details about promotions. We have not created a UI for creating promotions, but we will still have them in our application. Promotions provide discounts either the product item level or the cart level. Our promotions table will handle both types of promotions. The promotions table has the following fields:

- An identity column that contains an id that is automatically assigned
- A name, description, start date and end date
- The date the promotion was created
- A column to indicate the sale price of the product. This can be null.
- A column to indicate the overall discount amount if this promotion is not product specific. This can be null.
- A column to define the zip code in which the promotion is valid. This can be null
- A column to indicate the type of promotion: product, free shipping (or service fee or those sites that do not have shippable products)

Carts

The carts table is used to track the details of the user's cart. It should contain the following columns:

- An identity column that contains an id that is automatically assigned
- A status column to track if the cart is new, abandoned or purchased
- The date the cart was created
- The user id for the user that owns the cart
- The zip code in which this cart was created. This can be null.

CartProducts

The CartProducts table contains a list of all the products that are in the cart. It should contain the following columns:

- An identity column that contains an id that is automatically assigned
- An id for the cart the item belongs to
- The id of the product in the cart
- The quantity of that item

CartPromotions

The CartPromotions table contains a list of all the promotions that have been applied to the cart. It should contain the following columns:

- An identity column that contains an id that is automatically assigned
- A column for the promotion id
- A column for the product id that the promotion is applied to. This should be optional since the promotion may be a discount of some other kind

Technical Requirements

- Schema Diagram
    - Use the sql server schema diagram generator to create a diagram of your tables and how they relate to each other. Take a screenshot of this and include it in the zip file you send in.
- SQL

- Create create_tables.sql and drop_tables.sql containing the SQL statements necessary to create and drop your tables, respectively. Observe the following guidelines when constructing your SQL statements.
    - Use the plural form of an entity's name for its table (i.e. a users table holds user records).
    - Each table must have a primary key defined on it.
    - Columns whose values are required should have not null constraints declared on them.
    - Columns containing unique values (other than a primary key) should have a unique constraint declared on it.
    - Two tables connected by a relationship should have an appropriate foreign key declared between their connecting columns. Generally, the table containing entities that "belong to" entities in the other table has the foreign key defined on it.
    - Note that the order of your create table and drop table statements will be important. For instance, a table on which a foreign key constraint is to be defined needs to be created before the table that declares the constraint. Similarly, a table defining a foreign key constraint must be dropped before the table against which the constraint is declared.

Your create_tables.sql and drop_tables.sql files must execute correctly and without errors or warnings when run against a database.

Turn in the screenshot of the database schema relationship and the two sql files before the start of class.