Designing Relational Databases

Build A Menu For Bytes Of China

A new restaurant, Bytes of China, has just moved to a busy street in Chinatown from a previous location. Bytes of China has hired you to design its database schema so that it can display its menu on its new website, It has provided you with a copy of its current menu. 

Bytes of China describes itself on its website as follows:

Bytes of China, delectable Chinese cuisine  
2020 Busy Street, Chinatown, MA 02139  
Telephone: 617-555-1212  
Hours: Mon - Fri 9:00 am to 9:00 pm,  Weekends 10:00 am to 11:00 pm  
Ratings: 3.9 out of 200 reviews  
Google Map

A web page called Menu should have the following categories:

Appetizers, Soup, Chicken, Beef, Veal, Vegetarian, Rice and Noodles, Luncheon Specials, House Specials

Under each category should be a list of dishes in this sample format:

Category  Name  
Code1. Dish Name 1        Price 1  
           Dish Description 1  
Code2. Dish Name 2 \*        Price 2  
           Dish Description 2

Certain dishes have an asterisk, \*, to mark it as hot and spicy.

Certain categories such as Luncheon Specials provide additional information to the customer but all the dishes have the same price.:

Luncheon Specials, $8.95  
Served with Hot and Sour Soup or Egg Drop Soup and Fried or Steamed Rice  between 11:00 am and 3:00 pm from Monday to Friday.

In addition to a menu, a web page, Reviews, contains a collection of reviews from customers.

5.0  stars Awesome service. Would love to host another birthday party at Bytes of China! 05/22/2020  
4.5  stars Other than a small mix-up, I would give it a 5.0!  04/01/2020  
3.9  stars A reasonable place to eat for lunch, if you are in a rush!  03/15/2020  
...

As a database designer, you get to design a database schema and perform the following tasks:

* create tables
* define relationships between tables
* designate appropriate columns as keys
* insert sample data and
* make queries from the database

Let’s get started.

**Tasks**

17/18 Complete

Mark the tasks as complete by checking them off

**Task Group 1: Create Tables and Primary Keys**

1.

The first step towards designing a database schema is to create tables with the appropriate columns and primary keys.

Open **script.sql**. Create a restaurant table and an address table with columns that make sense based on the description above. Refer to our hint to see how to create these tables. Then, click SAVE to see the created table schema on the SQL browser.

The syntax for creating a table in PostgreSQL is:

CREATE TABLE table\_name (  
   column\_name1 <datatype>,  
   column\_name2 <datatype>,  
  ...  
  column\_nameN <datatype>  
);

Our restaurant table contains the following columns:

* id of integer
* name of varchar(20)
* description of varchar(100)
* rating of decimal
* telephone of char(10)
* hours of varchar(100)

Our address table contains the following columns:

* id of integer
* street\_number of varchar(10)
* street\_name of varchar(20)
* city of varchar(20)
* state or varchar(15)
* google\_map\_link of varchar(50)

2.

In **script.sql**, assign a primary key to a column for each of the tables, restaurant and address. Then, write queries to validate that these primary keys exist for these tables.

The primary key is one or more columns that contain unique value(s) which can be used to uniquely identify every row in the table. The id column for both restaurant and address tables should make an ideal primary key. Type the keyword PRIMARY KEY next to the column designated to be a primary key. For example:

CREATE TABLE recipe (  
  id integer PRIMARY KEY,  
  name varchar(20),  
  ...  
);

Use the information\_schema.key\_column\_usage view to display the constraint\_name and column\_name where table\_name is the name of the table.

3.

Next, we focus on the restaurant menu. For Bytes of China, its menu is split into many categories. Some categories (Appetizers and House Specials) simply have a name while others (Luncheon Specials) have more information such as price, hours and choice of side dishes served along with the main dish.

In **script.sql**, create a category table. Check the hint for how to do this.

A category’s id is a 2-character identifier such as ‘C’ for Chicken, ‘HS’ for House Specials and ‘LS’ for Luncheon Specials. For simplicity, the description column can contain information including hours of availability and other miscellaneous information. For most categories, only the id and name columns are sufficient and the description would be null.

Click SAVE to see the created table schema on the SQL browser.

Add a primary key to the correct column. Write a query to validate the primary key in this table.

Let’s add a category table with the following columns:

* id of char(2)
* name of varchar(20)
* description of varchar(200)

The id column will make an ideal primary key.

4.

Next, we focus on the dishes inside a category. A dish has a name, price, description and an indicator if it’s hot and spicy.

In **script.sql**, create a dish table. Check the hint to see how to create the dish table.

Click SAVE to see the created table schema on the SQL browser.

Add a primary key for the dish table. Write a query to validate the primary key in this table.

Create the dish table with these columns:

* id of integer
* name of varchar(50)
* description of varchar(200)
* hot\_and\_spicy of boolean

The id column will make an ideal primary key.

5.

Lastly, we create a table, review which corresponds to a customer review of the restaurant.

In **script.sql**, create a review table with columns that make sense. Check the hint for how to create this table.

Click SAVE to see the created table schema on the SQL browser. Add a primary key for this table. Write a query to validate the primary key in this table.

Create a review table with the following columns:

* id of integer
* rating of decimal
* description of varchar(100)
* date of date

**Task Group 2: Define Relationships and Foreign Keys**

6.

There are three types of database relationships: one-to-one, one-to-many and many-to-many. Of the five tables you have created, identify the relationships between any pair of tables.

A one-to-one relationship exists when one row in a table links to exactly one row in another table and vice-versa. Which two tables in our schema perfectly address a one-to-one relationship between them?

After identifying the two tables that exhibit a one-to-one relationship between them, implement this relationship by adding a foreign key in one of the tables. Write a query to validate the existence of this foreign key.

In our restaurant schema, the restaurant table has a one-to-one relationship with the address table.

To implement a one-to-one relationship, add a foreign key to the address table and designate it with the UNIQUE keyword.

To implement a foreign key in PostgreSQL, add a column in the child table that references the primary key of the parent table via the REFERENCES keyword. For example:

  columnname integer REFERENCES parent\_table(parent\_id)

7.

A one-to-many relationship exists when one row in a table links to many rows in another table. Which two tables perfectly address a one-to-many relationship between them? After identifying the two tables that exhibit a one-to-many relationship between them, implement this relationship by adding a foreign key in one of the tables. Write a query to validate the existence of this foreign key.

In our restaurant schema, a restaurant can have many customer reviews. To implement a one-to-many relationship, the primary key of the one table is referenced by the many table as a foreign key.

Add a restaurant\_id column in the review table as a foreign key referencing the restaurant table’s primary key, id.

8.

A many-to-many relationship comprises two one-to-many relationships. Which two tables perfectly address a many-to-many relationship between them?

How would you implement this relationship? To implement a many-to-many relationship, a third cross-reference table is required that contains two foreign keys referencing the primary keys of the member tables as well as a composite primary key from these two keys.

If you identify the category and dish tables as having a many-to-many relationship between them, you would be correct. A menu category consists of many dishes, however, the same dish may belong to more than one category. Hence, these two tables, category and dish exhibit a many-to-many relationship between them.

Create a third cross-reference table, categories\_dishes, to implement this relationship and assign the appropriate primary and foreign keys that are needed in this table. In addition to the keys mentioned above, add a price column of type money in this table. The price column is needed in this table because the cost of a dish depends on its category. For example, Chicken with Broccoli costs $6.95 as a main dish but $8.95 if it is part of Luncheon Specials which includes side dishes.

Click SAVE to see the created table schema on the SQL browser. Write a query to validate the primary and foreign keys that exist in this table.

A composite primary key is implemented as follows:

PRIMARY KEY (column\_one, column\_two)

**Task Group 3: Insert Sample Data**

9.

Congratulations on defining the restaurant menu schema. Now, it’s time to populate the schema with our sample data.

Open **projectdata.sql**. Study the various INSERT statements to populate the various tables. Copy the content from **projectdata.sql** and paste it to the end of **script.sql** and click SAVE to populate the tables.

If you didn’t make your tables exactly as described, you might get a syntax error when copying and pasting these INSERT statements. However, this is perfectly alright. Should you get a syntax error when trying to insert data, try the following tips:

* Inspect the error message.
* Inspect the data you’re trying to insert.
* Inspect the table schema to match the data you’re inserting.

**Task Group 4: Make Sample Queries**

10.

Once you have successfully imported the sample data in **projectdata.sql**, you can start making queries to the database. The SELECT, AS, FROM, WHERE, ORDER BY, HAVING and GROUP BY keywords will be useful here as well as a couple of functions.

In **script.sql**. Type in a query that displays the restaurant name, its address (street number and name) and telephone number. Then, click SAVE to run the query.

11.

In **script.sql**, write a query to get the best rating the restaurant ever received. Display the rating as best\_rating. Then, click SAVE to run the query.

To display a column using a custom header use the AS keyword as follows:

SELECT id AS custom\_name  
FROM table\_name;

To display the highest value in a column, use the MAX(column) function For example:

SELECT MAX(price) FROM item;

will display only the maximum value for the price column in the item table.

12.

Open **script.sql**. Write a query to display a dish name, its price and category sorted by the dish name. Your results should have the following header:

| **dish\_name** | **price** | **category** |
| --- | --- | --- |

You should get 8 rows of results.

You will be making this query from three tables: categories\_dishes, dish and category. Use the ORDER BY keyword to sort the results by a specific column.

13.

Instead of sorting the results by dish name, type in a new query to display the results as follows, sorted by category name.

| **category** | **dish\_name** | **price** |
| --- | --- | --- |

14.

Next, type a query in **script.sql** that displays all the spicy dishes, their prices and category. The header should look like this:

| **spicy\_dish\_name** | **category** | **price** |
| --- | --- | --- |

You should get 3 rows of results.

You will be making this query from three tables: categories\_dishes, dish and category.

15.

In a complete menu, there will be dishes that belong to more than one category. In our sample menu, only Chicken with Brocolli is assigned to two different categories - Luncheon Specials and Chicken. How do we query the database to find dishes that span multiple categories?

We could use a database function, COUNT(column\_name) to help us. For instance if we have a table whose non-primary key column appears multiple times in results, we can count how many times the row appears.

Write a query that displays the dish\_id and COUNT(dish\_id) as dish\_count from the categories\_dishes table. When we are displaying dish\_id along with an aggregate function such as COUNT(), we have to also add a GROUP BY clause which includes dish\_id.

To display column\_name in addition to an aggregate function such as COUNT(column\_name), add the column\_name to the GROUP BY clause as follows:

SELECT column\_name, COUNT(column\_name)  
FROM table\_name  
GROUP BY column\_name;

16.

Great work! Try to adjust the previous query to display only the dish(es) from the categories\_dishes table which appears more than once. We can use the aggregate function, COUNT() as a condition. But instead of using the WHERE clause, we use the HAVING clause together with COUNT().

To add a condition using an aggregate function such as COUNT(column\_name) > 1, do the following:

SELECT column\_name, COUNT(column\_name)  
FROM table\_name  
GROUP BY column\_name  
HAVING COUNT(column\_name) > 1;

17.

Excellent! The previous two queries only give us a dish\_id which is not very informative. We should write a better query which tells us exactly the name(s) of the dish that appears more than once in the categories\_dishes table. Write a query that incorporates multiple tables that display the dish name as dish\_name and dish count as dish\_count.

Add another condition to the HAVING clause that is based on the primary keys categories\_dishes.dish\_id and dish.id. Then, add dish.name and dish.id to the GROUP BY clause.

SELECT  
  table\_one.column\_two, COUNT(table\_two.column\_one)   
FROM  
  table\_one, table\_two  
GROUP BY  
  table\_one.column\_one, table\_one.column\_two, table\_two.column\_one  
HAVING  
  COUNT(table\_two.column\_one) > 1 and table\_two.column\_one = table\_one.column\_one;

18.

Our last task is an improvement on Task 11 which was to display the highest rating from the review table using an aggregate function, MAX(column\_name). Since the result returned only one column, it is not very informative.

| **best\_rating** |
| --- |
| 5.0 |
| (1 row) |

It would be better if we can also see the actual review itself. Write a query that displays the best rating as best\_rating and the description too. In order to do this correctly, we need to have a nested query or subquery. We would place this query in the WHERE clause.

SELECT column\_one, column\_two  
FROM table\_name  
WHERE  column\_one = ( SELECT MAX(column\_one) from table\_name );

Type your last query in **script.sql**.

You have completed this project by writing tables, defining constraints and relationships between tables. You have populated the tables and written interesting and challenging queries to the database.

Congratulations for a project well done!