Final Project: Database Design and Implementation

Estimated time needed: 90 minutes

Congratulations! You have finished the modules. Now is the time to put your skills to the test. Read through the scenario below.

Scenario

In this scenario, you have recently been hired as a Data Engineer by a New York-based coffee shop chain looking to expand nationally by opening several franchise locations. They want to streamline operations and revamp their data infrastructure as part of their expansion process.

Your job is to design their relational database systems for improved operational efficiencies and make it easier for their executives to make data-driven decisions.

Currently, their data resides in several systems: accounting software, supplier databases, point of sales (POS) systems, and even spreadsheets. You will review the data in all of these systems and design a central database to house all of the data. You will then create the database objects and load them with source data. Finally, you will create subsets of data your business partners require, export them, and load them into staging databases using several RDBMS.

Software used in this project

In this project, you will use PostgreSQL Database, MySQL. These are all relational database management systems (RDBMS) designed to store, manipulate, and retrieve the data efficiently.

Data used in this project

In this project, you will be working with a subset of data from the Coffee shop sample data.

You will use a modified version of the data for the project, so to succeed in the project, download the linked files when prompted in the instructions. You do not need to use any data from the source.

In your scenario, you will be working with data from the following sources:

- Staff information held in a spreadsheet at headquarters (HQ)
- Sales outlet information held in a spreadsheet at HQ
- Sales data output as a CSV file from the POS system in the sales outlets
- Customer data output as a CSV file from a custom customer relationship management system
- Product information maintained in a spreadsheet exported from your supplier's database

Objectives

After completing this lab, you will be able to:

- Identify entities
- Identity attributes
- Create an entity relationship diagram (ERD) using the pgAdmin ERD tool
- Normalize tables
- Define keys and relationships
- Create database objects by generating and running the SQL script from the ERD tool
- Create a view and export the data
- Create a materialized view and export the data
    - Import data into a MySQL database using phpMyAdmin GUI tool
    - Import data into a MySQL database

Task 1: Identify entities

The first step when designing a new database is to review any existing data and identify the entities for your new system.

1. The following image shows sample data from each source you will be working with to design your new central database. Review the image and identify the entities you plan to create.

    - Note: You can download a copy of this image or open it in another browser tab for reference later in the lab.

2. Make a list of the entities you have identified. Take a screenshot and save it as Task1.jpg or Task1.png.

Task 2: Identify attributes

In this task, you will identify the attributes of one of the entities you plan to create.

1. Using the information from the sample data in the image from Task 1, identify the entity's attributes that will store the sales transaction data.

2. Make a list of the sales transaction attributes that you identified. Take a screenshot and save it as Task2.jpg or Task2.png.

Task 3: Create an ERD

Now that you have defined some of your attributes and entities, you can determine the tables and columns for them and create an entity-relationship diagram (ERD).

1. Open a new terminal from the side-by-side Cloud IDE.

2. Use the button below to start a PostgreSQL service session in the Cloud IDE.

Open and Start PostgreSQL in IDE

3. Use the pgAdmin weblink to open pgAdmin in a new tab in your browser.

4. Create a new database named `COFFEE`, view the schemas in the new `COFFEE` database, and then start a new ERD project.

5. Add a table to the ERD for the sale transactions entity using the information in the following table. Consider the naming convention to use so that your colleagues can understand your data and ensure that the names are valid in other RDBMS. Use the sample data shown in the image in Task 1 to determine appropriate data types for each column.

6. Take a screenshot of your ERD and save it as Task3A.png or Task3A.jpg.

7. Add a table to the ERD for the product entity using the information in the following table.

8. Take a screenshot of your ERD and save it as Task3B.png or Task3B.jpg.

Task 4: Normalize tables

When reviewing your ERD, you notice it does not conform to the second normal form. In this task, you will normalize some of the tables within the database.

1. Review the data in the sales transaction table. Note that the transaction id column does not contain unique values because some transactions include multiple products.

2. Determine which columns should be stored in a separate table to remove the repeating rows and to put this table into second normal form.

3. Add a new table named `sales_detail` to the ERD, define the columns in the new table, and delete the moved columns from the sales transaction table, leaving a matching column in each of the two tables to create a relationship between them later.

4. Take a screenshot of your ERD and save it as Task4A.png or Task4A.jpg.

5. Review the data in the product table. Note that the product category and product type columns contain redundant data.

6. Determine which columns should be stored in a separate table to reduce redundant data and to put this table into a second normal form.

7. Add a new table named `product_type` to the ERD, define the columns in the new table, and delete the moved columns from the product table, leaving a matching column in each of the two tables to create a relationship between them later.

8. Take a screenshot of your ERD and save it as Task4B.png or Task4B.jpg.

Task 5: Define keys and relationships

After normalizing your tables, you can define their primary keys and relationships between the tables in your ERD.

1. Identify an appropriate column in each table to be a primary key and create the primary keys in the tables in your entity-relationship diagram (ERD).

2. Take a screenshot of your ERD and save it as Task5A.png or Task5A.jpg.

3. Identify the relationships between the following pairs of tables and then create the relationships in your ERD:

    - `sales_detail` to `sales_transaction`
    - `sales_detail` to `product`
    - `product` to `product_type`

4. Take a screenshot of your ERD and save it as Task5B.png or Task5B.jpg.

Task 6: Create database objects by generating and running the SQL script from the ERD tool

Now that your design is complete, you will generate an SQL script from your ERD, which you can use to create your database schema. For this project, you will then use a given SQL script to ensure that you can load the sample data into the schema. Finally, you will load the existing data from various sources into your new database schema.

1. Use the Generate SQL functionality in the ERD tool to create an SQL script from your ERD.

2. Download the following `GeneratedScript.sql` file to your local computer.

    - GeneratedScript.sql

3. In pgAdmin, open the query tool, upload and open the `GeneratedScript.sql` file from your local computer, and then run the script to create the tables defined in the ERD. Verify that the tables exist in the COFFEE database\'s public schema now.

4. Take a screenshot of the tables shown in the tree-view pane on the left side of the page and save it as Task6A.png or Task6A.jpg.

5. Download the following `CoffeeData.sql` file to your local computer.

    - CoffeeData.sql

6. In pgAdmin, open another instance of the Query tool, upload and open the `CoffeeData.sql` file from your local computer, and then run the script to populate the tables you just created.

7. In pgAdmin, view the first 100 rows of the `sales_detail` table.

8. Take a screenshot of the Data Output pane and save it as Task6B.png or Task6B.jpg.

Task 7: Create a view and export the data

The external payroll company has requested a list of employees and the locations at which they work. This list should not include the CEO or CFO who owns the company. In this task, you will create a view in your PostgreSQL database that returns this information and export the results to a CSV file.

1. In your `COFFEE` database, create a new view named `staff_locations_view` using the following SQL:

```
1. 1
2. 2
3. 3
```

```
4. 4
5. 5
6. 6
1. SELECT staff.staff_id,
2. staff.first_name,
3. staff.last_name,
4. staff.location
5. FROM staff
6. WHERE "position" NOT IN ('CEO', 'CFO');
```

Copied!

2. View all the rows returned from the view.
3. Save the query results to a file named `staff_locations_view.csv` on your local computer.
4. Take a screenshot of the view shown in the tree-view pane on the left side of the page with the results in the Data Output pane, and save it as Task7.png or Task7.jpg.

Task 8: Create a materialized view and export the data

A marketing consultant requires access to your product data in their MySQL database for a marketing campaign. You will create a materialized view in your PostgreSQL database that returns this information and export the results to a CSV file.

1. In your `COFFEE` database, create a new materialized view named `product_info_m-view` using the following SQL:

```
1. 1
2. 2
3. 3
4. 4
1. SELECT product.product_name, product.description, product_type.product_category
2. FROM product
3. JOIN product_type
4. ON product.product_type_id = product_type.product_type_id;
```

Copied!

2. Refresh the materialized view with data.
3. View all the rows returned from the view.
4. Save the query results to a file named `product_info_m-view.csv` on your local computer.
5. Take a screenshot of the view shown in the tree-view pane on the left side of the page with the results in the Data Output pane, and save it as Task8.png or Task8.jpg.

Task 9: Import staff_location data into a MySQL database

The external payroll company has asked you to upload the staff location information to their MySQL database.

1. Click the button below to start a MySQL instance in the Cloud IDE.

Open and Start MySQL in IDE

2. Open phpMyAdmin in a new tab in your browser.
3. In phpMyAdmin, create a new database named `STAFF_LOCATIONS`, then import the location information saved in the `staff_locations_view.csv` file you exported from the view you created in Task 7.
4. Explore the new table and then view the data in it.
5. Take a screenshot of the contents of the new table and save it as Task9.png or Task9.jpg.

Task 10: Import coffee_shop_products data into a MySQL database

The marketing consultant has asked you to upload the product information to their MySQL database.

1. In phpMyAdmin, create a new database named `coffee_shop_products`, and then import the product information saved in the `product_info_m-view.csv` file from your materialized view into a new table in the `coffee_shop_products` database.
2. Browse the contents of the new table.
3. Take a screenshot of the contents of the new table and save it as Task10.png or Task10.jpg.

Task 11[Optional]: Import staff_location and coffee_shop_products into SQLite OR Db2 database

The external payroll company has asked you to upload the staff location information to their Db2 database.

1. In a new browser tab, go to cloud.ibm.com/login, log in using your credentials, and then open a console for your Db2 on the Cloud instance you created earlier in this course.
2. Use the Load Data feature to load a new table named `STAFF_LOCATIONS` with the staff location information saved in the `staff_locations_view.csv` file you exported from the view you created in Task 7.
3. Explore the new table and then view the data in it.
4. Take a screenshot of the contents of the new table and save it as Task9.png or Task9.jpg.

Recap

In this project, you created the objects and views that you exported in a MySQL database.

Author(s)

- Lin Joyner
- Pratiksha Verma