# MIDS W205

|  |  |  |  |
| --- | --- | --- | --- |
| **Lab #** | 5 | **Lab Title** | Working with Relational Databases |
| **Related Module(s)** | 5 | **Goal** | Get you introduced to a RDBMS (PostgreSQL) |
| **Last Updated** | 1/6/17 | **Expected duration** | 60 min |

## Introduction, Resources

While our initial investigations have dealt with Hive and SparkSQL, often as a Data Scientist, you will encounter relational databases like PostgreSQL. In this lab, we will learn about the following:

1. How to create a database in PostgreSQL
2. How to load data into PostgreSQL
3. How to run queries on PostgreSQL
4. How queries are transformed into plans for DAGs in PostgreSQL

|  |  |
| --- | --- |
| Resource | What |
| <http://www.postgresql.org/docs/9.5/static/index.html> | PostgreSQL Documentation |
| <http://www.postgresql.org/docs/9.5/static/sql.html> | The SQL Language |

## Step 1. Setup the environment

We need to setup an EC2 instance and make sure that PostgreSQL is up and running. Do the following:

1. Launch an instance of UCB W205 Spring 2016
2. Attach your EBS volume from Lab 2. Note that PostgreSQL should be installed after you finish step 3.4 of Lab 2
3. Check whether PostgreSQL is up and running: bash ps auxw | grep postgres
4. If not, change your current path to /data directory bash cd /data and start Postgres bash start\_postgres.sh

## Step 2. Getting the Data

1. We need some data in order to create a database, schema and, ultimately, query. The data we'll consider is a toy dataset called "DVD rental".
2. Navigate to the /data directory on your AWS instance and download the Pagila data as follows: bash wget -O pagila.zip http://pgfoundry.org/frs/download.php/1719/pagila-0.10.1.zip
3. Unzip the data: bash unzip pagila.zip
4. Connecting to the PostgreSQL instance: Log into postgres as the postgres user: bash psql -U postgres
5. Create a database: bash create database dvdrental;
6. Connect to the database: bash \c dvdrental
7. Import the data: Load the data using the \i command. Starting a line with \i runs .sql scripts in Postgres. bash \i pagila-0.10.1/pagila-schema.sql \i pagila-0.10.1/pagila-insert-data.sql \i pagila-0.10.1/pagila-data.sql
8. Check to see if it worked:

At this point the data should be loaded. Examine the database schema by typing: bash \dt

Examine the schema of a table using the \d command bash \d <table name>

Question 1: What is the output of ?

Question 2: What is the schema for the customer table?

## Step 3. Running Queries and Understanding EXPLAIN plans

We want to understand not only what queries we can issue against data, but also how that query maps to an execution plan. For each of the following sections, run the queries provided, and generate their explain plans using: bash EXPLAIN <sql query here>

Run the following simple queries, then generate their explain plans.

1. **Projection:**

sql SELECT customer\_id, first\_name, last\_name FROM customer; 1. **Projection and Selection #1:**

sql SELECT customer\_id, amount, payment\_date FROM payment WHERE amount <= 1 OR amount >= 8;

1. **Projection and Selection #2:**

sql SELECT customer\_id, payment\_id, amount FROM payment WHERE amount BETWEEN 5 AND 9; > Question 3: What similarities do you see in the explain plains for these 3 queries?

1. **Merging Data: UNIONs:** Run the following 2 statements:

Union of 2 tables: sql SELECT u.customer\_id, sum(u.amount) FROM ( SELECT \* FROM payment\_p2007\_01 UNION SELECT \* FROM payment\_p2007\_02) u WHERE u.payment\_date <= '2007-02-01 00:00:00'::TIMESTAMP WITHOUT time ZONE GROUP BY u.customer\_id ; Partition a Table: sql SELECT customer\_id, sum(amount) FROM payment WHERE payment\_date <= '2007-02-01 00:00:00'::TIMESTAMP WITHOUT time ZONE GROUP BY customer\_id ; > Question 4: What is the difference between the plans for the Partitioned table and the union query? Why do you think this difference exists?

1. **Merging Data: JOINs:** sql SELECT customer.customer\_id, first\_name, last\_name, email, amount, payment\_date FROM customer INNER JOIN payment ON payment.customer\_id = customer.customer\_id; > Question 5: What join algorithm is used for the inner join?
2. Finally, disconnect from postgres: bash \q

## Submissions

Submit your answers to the questions through ISVC as a text file, docx file, or PDF.