CS 327E Lab 1: Exploring the IMDB dataset through SQL

Prerequisites:
1. Formed a group
2. Created a personal GitHub account
3. Received a GitHub repo in our class org
4. Installed a git client locally
5. Created an AWS account and received $100 in education credits
6. Created an IAM account with admin privileges in your AWS account
7. Created a LucidChart account with an education upgrade

Step 0. Follow the steps in our snippets wiki to satisfy prerequisites 4-7 [1].

Step 1. Using your local git client, clone your GitHub repo to your laptop. Create a new folder called **lab1** in your local repo. Add all your work for this lab to this lab1 folder. Commit your changes as you go and push them to the remote repo on GitHub as you go. Very important: make lots of small incremental commits as opposed to one large commit right before the deadline!

Step 2. Get the sample imdb files from this download link: http://cs327e-fall2017-imdb.s3.amazonaws.com/sample.zip and open the zip file. You do **not** need to add the samples file to your git repo.

Step 3. From the AWS console, open Athena. In the Athena Query Editor, create an imdb database using the command:

```
create database imdb;
```

Put all the tables that you will create into this database.

Step 4. For each file in the imdb dataset, create an external table for this file. Use the following S3 paths for the input files. For example, to create the Directors table, use the input path s3://cs327e-fall2017-imdb/athena/directors/

s3://cs327e-fall2017-imdb/athena/directors/
s3://cs327e-fall2017-imdb/athena/person_basics/
s3://cs327e-fall2017-imdb/athena/person_professions/
s3://cs327e-fall2017-imdb/athena/principals/
s3://cs327e-fall2017-imdb/athena/stars/
s3://cs327e-fall2017-imdb/athena/title_basics/

s3://cs327e-fall2017-imdb/athena/title_episodes/
s3://cs327e-fall2017-imdb/athena/title_genres/
s3://cs327e-fall2017-imdb/athena/title_ratings/
s3://cs327e-fall2017-imdb/athena/writers/

Note: The files that are stored in S3 contain the full imdb dataset.

Choose CSV for the data format and open the sample files to figure out what the column names and datatypes should be. One important caveat, a char and varchar datatype are known as a string in Athena. You will see a drop-down that gives the list of supported column types. Use numeric to store numbers with decimals.

As you create each table, copy and paste the generated DDL (i.e. create table statement) into a file called imdb_tables.sql. **Add** this file to your **git repo**.

Step 5. Explore the imdb data through SQL. First, preview each table to get a basic understanding of the data. Next, run some queries to understand what the primary key is for each table, and how the tables relate to each other via primary and foreign keys. Also, identify the datatype for each field in the tables. Once you have defined the schema, think of 20 interesting questions to ask that are non-trivial. Write those 20 questions in English and in SQL. Together, the 20 queries should meet the following minimum criteria:

- Each table must be accessed at least once
- There must be at least 10 `inner joins`
- There must be at least 10 `where` clauses
- There must be at least 10 `order by` clauses

Use the following format to document the 20 queries:

/* Query 1: Find all movies with Tom Hanks sorted by year, from newest to oldest */

```
select tb.title_id, tb.primary_title, tb.start_year
from title_basics tb join principals p on tb.title_id = p.title_id
join person_basics pb on pb.person_id = p.person_id
where pb.primary_name = 'Tom Hanks'
and tb.title_type = 'movie'
and tb.start_year is not null
order by tb.start_year desc;
```

Notice that the comment above the SQL provides a brief explanation of the query in English. Place the 20 queries in a file called imdb_queries.sql and **add** it to your **git repo**.

Step 6. Using LucidChart, draw an ER diagram of the imdb schema. We will be using this diagram in the upcoming lab project to create the tables in Postgres, so be sure to use valid constraints and datatypes. For numbers with decimals, use the numeric datatype. For char and varchar columns, you should get the

max length of the characters in those columns to decide what the limit of the char or varchar should be. For example, to find out what Title_Basics.primary_title should be, run the query:

```
select max(length(primary_title)) from Title_Basics;
```

This returns a max length of 292, which we will round up to the nearest 10. So the datatype for `Title_Basics.primary_title` will be varchar(300).

Download the diagram from Lucid in png format and choose the option "Crop to Content". Name the file imdb_erd.png and **add** it to your **git repo**.

Step 7. Create a Stache entry for storing AWS admin credentials that we will need for grading your project. Give the Stache entry the same name as your repo name. In the secret field, put the following json string:

```
{
  "aws-username": "admin",
  "aws-password": "mypassword",
  "aws-console-link": "myconsolelink"
}
```

Replace mypassword and myconsolelink with your actual admin account password and console link, respectively.

The Stache entry should also have a read-only API endpoint and read key. Select allow this item to be accessed by read-only API call to generate this endpoint and key. Make sure to save the Stache entry.

Step 8. Locate the commit id that you will be using for your submission. This is a long 40-character that shows up on your main GitHub repo page next to the heading "Latest commit" (e.g. commit 6ca6f695bca36f7fc2c33485d1080ae30f8b9928). Locate the link to your GitHub repo (e.g. https://github.com/cs327e-fall2017/xyz.git where xyz is your repo name). Go back to the Stache entry and locate the read-only API endpoint and read key.

Replace the commit id, repo link, API endpoint, and read key in the json string below with your own:

```
{
  "repository-link": "https://github.com/cs327e-spring2017/xyz.git",
  "commit-id": "6ca6f695bca36f7fc2c33485d1080ae30f8b9928",
  "stache-endpoint": "/api/v1/item/read/61515",
  "stache-read-key": "b2eacb0387a919e33b27e7c03a6c5d84b71234795732be33eb28711ec16f0e21"
}
```

Create a file called submission.json that contains your modified json string. Click on the Lab 1 Assignment in Canvas and upload submission.json. **Do not add submission.json to your git repo**.

This submission is due by **Friday, 09/22 at 11:59pm**. If it's late, there will be a 10% grade reduction per late day. This late policy is also documented in the syllabus.

References:

[1] Snippets Wiki: https://github.com/cs327e-fall2017/snippets/wiki
[2] Lab 1 Grading Rubric: http://www.cs.utexas.edu/~scohen/projects/lab1-rubric.pdf
[3] IMDB sample dataset: http://cs327e-fall2017-imdb.s3.amazonaws.com/sample.zip