Update mini2.md
Brandon Haynes authored just now

fbf75d06

mini2.md 5.12 KB

Mini-Homework #2

Due date: December 10, 2019

Objectives:

Execute queries on Vertica!

Assignment tools

Docker on EC2 or your local machine with Vertica. See the section notes for details on installation.

What to Turn In

Turn in command and results of a query using Vertica on a toy dataset. Submit everything as a single markdown, notebook, or PDF.

How to submit the assignment

In your GitLab repository, you should see a directory called `Homeworks/mini-hw2` . Put your report in that directory. Remember to `git add, git commit, and git push` . You can add your report early and keep updating it and pushing it as you do more work. We will collect the final version after the deadline passes. If you need extra time on an assignment, let us know. This is a graduate course, so we are reasonably flexible with deadlines but please do not overuse this flexibility. Use extra time only when you truly need it.

Assignment Dataset

Lobster Dataset

The data we will use for Vertica is a toy dataset for website http://lobste.rs, which is a Hacker News clone. We'll create relations for this dataset, ingest data, and then run queries.

Create Schema

Create relations for this dataset as follows:

```
CREATE SCHEMA lobsters;

CREATE TABLE lobsters.tags ( id integer NOT NULL, tag varchar(64));

CREATE TABLE lobsters.taggings (id integer NOT NULL, story_id integer NOT NULL, tag_id integer NOT NULL);

CREATE TABLE lobsters.hiddens (id integer NOT NULL, user_id integer NOT NULL, story_id integer NOT NULL);

CREATE TABLE lobsters.stories (
  id integer NOT NULL,
  created_at TIMESTAMP,
  description varchar(4095),
  hotness float,
  markeddown_description varchar(4095),
  short_id varchar(255),
  title varchar(1023),
  upvotes integer,
  downvotes integer,
  url varchar(255),
  user_id integer);
```

Download data

Next, download the data for these relations:

- hidden.csv
- tags.csv
- taggingcsv
- stories.csv

Ingesting data into the database

Finally, use the following Python script to load data into your Vertica database:

```
import vertica_python as vp
import json

with open("args.json") as f:
    args = json.load(f)
conn = vp.connect(**args)
cur = conn.cursor()
cur.execute("select * from sample_table")
cur.fetchone()
cur.fetchall()

with open("taggings.csv", "rb") as f:
    cur.copy("COPY lobsters.taggings from stdin DELIMITER ','", f)
with open("hidden.csv", "rb") as f:
    cur.copy("COPY lobsters.hiddens from stdin DELIMITER ','", f)
with open("stories.csv", "rb") as f:
    cur.copy("COPY lobsters.stories from stdin DELIMITER ','", f)
```

Sample Queries

To make sure things are working, try executing the following queries from your VSQL console:

```
select count(*) from lobsters.stories where user_id = 1;

select t.tag_id, count(*) from lobsters.stories as s inner join lobsters.taggings as t on s.hotness > 0 and t.story_id = s.i

select s.id, s.title, s.upvotes, s.downvotes from lobsters.stories as s inner join lobsters.hiddens as h on created_at > '20

select id, title from lobsters.stories
where id in (( select story_id from lobsters.taggings where tag_id not in (2, 6, 7) ) union (select story_id from lobsters.h
```

Assignment Details

In this Assignment you will be required to access a Vertica DBMS either via EC2 or locally. In both cases we recommend installing Vertica in a Docker container as described in the section notes.

In this homework you will write a few queries on the Lobsters dataset. For query A and B below, return the SQL and the rows returned. You may restrict the columns you list for these two queries.

For the last bullet, run the queries a couple times and return mean runtime for each of the queries as well as the discussion of the resulting query times. For this query there is no need to return number of rows or actual rows from the queries.

- Describe in one sentence the purpose of each of the queries in the previous section. Which (if any) of these queries are especially well-suited for a column store?
- Query A: List the top-10 stories ranked by 'hotness' (10 points)
- Query B: List top 10 stories with the highest number of up-votes and lowest number of down-votes. How does this list compare with the results from the previous query? (10 points)
- For the `stories` relation run the following queries, note the time to run these queries and discuss your intuition and observations about their runtimes (10 points):
  - `select * from stories;`
  - `select id from stories;`
  - `select id, title from stories;`