IT Hiring Take Home Problem 1 - Query Tool

This programming challenge entails creating a highly simplified version of the typical Rentrak software platform for measuring media. The challenge consists of four sections. You are neither required nor expected to complete all four sections. The purpose of the exercise is to give you to get a sense of the type of systems you would work on at Rentrak and allow you to provide a small sample of the level and style of your coding ability. By allowing you to do this at your own leisure rather than in a stressful interview situation we hope to get a better sense of your ability and subsequently allow you to spend your onsite interview time discussing your solution and learning about Rentrak rather than going through a stressful marathon series of live coding challenges. As such, we ask you not to invest any more time than you feel is necessary to meet these goals. Each section is designed to take around 2 hours. Generally, completing section 1 and 2.1 should be sufficient if you implement this challenge in a general purpose programming language.

1.) Importer and Datastore

You will be provided with a pipe separated file in the following format.

File Sample:

STB|TITLE|PROVIDER|DATE|REV|VIEW\_TIME

stb1|the matrix|warner bros|2014-04-01|4.00|1:30

stb1|unbreakable|buena vista|2014-04-03|6.00|2:05

stb2|the hobbit|warner bros|2014-04-02|8.00|2:45

stb3|the matrix|warner bros|2014-04-02|4.00|1:05

Field Descriptions:

STB - The set top box id on which the media asset was viewed. (Text, max size 64 char)

TITLE - The title of the media asset. (Text, max size 64 char)

PROVIDER - The distributor of the media asset. (Text, max size 64 char)

DATE - The local date on which the content was leased by through the STB (A date in YYYY-MM-DD format)

REV - The price incurred by the STB to lease the asset. (Price in US dollars and cents)

VIEW\_TIME - The amount of time the STB played the asset. (Time in hours:minutes)

Your first task is to parse and import the file into a simple datastore. You may use any file format that you want to implement to store the data. For this exercise, you must write your own datastore instead of reusing an existing one such as a SQL database or a NoSQL store. Also, you must assume that after importing many files the entire datastore will be too large to fit in memory. Records in the datastore should be unique by STB, TITLE and DATE. Subsequent imports with the same logical record should overwrite the earlier records.

2.) Query tool

2.1) select, order and filter

The next task is to create a query tool that can execute simple queries against the datastore you created in step one. The tool should accept command line args for SELECT, ORDER and FILTER functions:

$ ./query -s TITLE,REV,DATE -o DATE,TITLE

the matrix,4.00,2014-04-01

the hobbit,8.00,2014-04-02

the matrix,4.00,2014-04-02

unbreakable,6.00,2014-04-03

$ ./query -s TITLE,REV,DATE -f DATE=2014-04-01

the matrix,4.00,2014-04-01

2.2) group and aggregate functions

The next step is to add group by and aggregate functions to your query tool. Your tool should support the following aggregates:

MIN: select the minimum value from a column

MAX: select the maximum value from a column

SUM: select the summation of all values in a column

COUNT: count the distinct values in a column

COLLECT: collect the distinct values in a column

$ ./query -s TITLE,REV:sum,STB:collect -g TITLE

the matrix,8.00,[stb1,stb3]

the hobbit,8.00,[stb2]

unbreakable,6.00,[stb1]

2.3) advanced filter function

Add a filter function which evaluates boolean AND and OR expressions in the following format:

STB="stb1" AND TITLE="the hobbit" OR TITLE="unbreakable"

Assume AND has higher precedence than OR. Parens can be added to change the above statement to the more logical:

STB="stb1" AND (TITLE="the hobbit" OR TITLE="unbreakable")

Example:

$ ./query -s TITLE,REV -f 'TITLE="the hobbit" OR TITLE="the matrix"'

the matrix,4.00

the hobbit,8.00

the matrix,4.00