Data Science Course Exercise Workbook (Inc. Answers)

Hive: Course Material

Adam Kershaw, April 2017

Data Science – Hive

Adam Kershaw, April 2017

08

**Fall**

Contents

[Summary and Intro 2](#_Toc483554178)

[Section 1 - Stocks Data: 3](#_Toc483554179)

[1.1 Database &Table Creation 3](#_Toc483554180)

[1.2 Querying 101: 4](#_Toc483554181)

[1.3 Further Querying: 5](#_Toc483554182)

[1.4 Querying across datasets: 6](#_Toc483554183)

[Section 2 – San Francisco Salaries Mini Project: 9](#_Toc483554184)

[2.1 Scenario & Data: 9](#_Toc483554185)

[2.2 Task: 9](#_Toc483554186)

[Additional Research Task 10](#_Toc483554187)

[Summary 10](#_Toc483554188)

## Summary and Intro

This workbooks aim is to aid and facilitate Data Science trainees learning. This workbook will cover aspects about using apache Hive, to manage and explore large datasets residing in distributed storage.

1. Stocks – Hive Exercises
2. San Fran Salaries - Hive Mini Project

This workbook and subsequent workbooks will act as a record of the trainees work, learnings and understanding.

You should aim to complete all questions. Sections relate to course slides, so the workbook can be completed in stages or once all theory has been delivered.

# Section 1 - Stocks Data:

This section will focus on the Stock data files, please complete all exercises.

**Scenario:**

Come up with some sort of case for this other than the basic “getting to grips”.

**Data:**

For the following exercises in this section we will be using the Stocks data. This data is in 6 CSV files: appl\_daily, face\_daily, nasdaq\_daily, nflx\_daily, twtr\_daily, yhoo\_daily.

Each of these files has the same following attributes: Date, Open, High, Low, Close, Volume, Adj\_Close.

It is recommended to use ‘Geaney’ (a unix alternative to notepad++) to develop your hiveQL code, rather than develop in the command line.

## 1.1 Database &Table Creation

Write a hiveQL statement to create database location for the stock data.

1. Create a Table inside of your database for apple data and load in the appl\_daily .csv file.
2. You can create a Hive Managed or External Table, It is up to you but make sure you can justify your decision

CREATE DATABASE stocks;

--External

CREATE EXTERNAL TABLE IF NOT EXISTS appl\_stock

(stock\_date date, open int, high int, low int, close int, adj\_close int, volume int)

COMMENT 'Apple Stock, Source: Yahoo Finance'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION '/user/adam.kershaw/stock\_data/apple'

tblproperties("skip.header.line.count"="1");

1. In order to test your table is set up correctly; use your SQL knowledge to retrieve the data.
2. Once completed write a script to create tables for the remaining stock datasets (6 Tables in total) **Ensure to use hive Command Line functions to execute your script.**

CREATE EXTERNAL TABLE IF NOT EXISTS fb\_stock

(stock\_date date, open int, high int, low int, close int, adj\_close int, volume int)

COMMENT 'Facebook Stock, Source: Yahoo Finance'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION '/user/adam.kershaw/stock\_data/facebook'

tblproperties("skip.header.line.count"="1");

CREATE EXTERNAL TABLE IF NOT EXISTS nasdaq\_stock

(stock\_date date, open int, high int, low int, close int, adj\_close int, volume int)

COMMENT 'Nasdaq Stock, Source: Yahoo Finance'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION '/user/adam.kershaw/stock\_data/nasdaq'

tblproperties("skip.header.line.count"="1");

CREATE EXTERNAL TABLE IF NOT EXISTS nffx\_stock

(stock\_date date, open int, high int, low int, close int, adj\_close int, volume int)

COMMENT 'Netflix Stock, Source: Yahoo Finance'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION '/user/adam.kershaw/stock\_data/netflix'

tblproperties("skip.header.line.count"="1");

CREATE EXTERNAL TABLE IF NOT EXISTS twtr\_stock

(stock\_date date, open int, high int, low int, close int, adj\_close int, volume int)

COMMENT 'Twitter Stock, Source: Yahoo Finance'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION '/user/adam.kershaw/stock\_data/twitter'

tblproperties("skip.header.line.count"="1");

CREATE EXTERNAL TABLE IF NOT EXISTS yhoo\_stock

(stock\_date date, open int, high int, low int, close int, adj\_close int, volume int)

COMMENT 'Yahoo Stock, Source: Yahoo Finance'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION '/user/adam.kershaw/stock\_data/yahoo'

tblproperties("skip.header.line.count"="1");

--Code to execute script

Hive –f ‘insert/path/here’

## 1.2 Querying 101:

This section will test your ability to write efficient SELECT statements in hiveQL, making use of the WHERE clause.

Write a hiveQL query to determine:

1. How the Highs of apple change over time (The entire data set)

SELECT \*

FROM appl\_stock;

1. Apple’s stock data for the last 100 days recorded

SELECT \* FROM appl\_stock

ORDER BY stock\_date DESC

LIMIT 100;

1. Apple released the 1st generation IPhone in September 2007, extract the stock data for this month.

SELECT \*

FROM appl\_stock

WHERE year(stock\_date) = 2007

AND month(stock\_date) = 9;

SELECT \*

FROM appl\_stock

WHERE date LIKE '2007-09%';

1. Apple’s earliest recorded stock data is from 1980, extract the data for 1980’s last day of trading.

SELECT \*

FROM appl\_stock

WHERE year(stock\_date) = 1980

ORDER BY stock\_date DESC

LIMIT 1;

1. Steve Jobs joined Apple as CEO in 1977, and left Apple in January 2011, to focus on his health. Extract the stock data the end of January 2011. Comment on how the stock price changed under his lead (1980 – 2011)?

**SELECT \***

**FROM appl\_Stock**

**WHERE year(stock\_date) = 2011**

**ORDER BY stock\_date DESC**

**LIMIT 1;**

## 1.3 Further Querying:

This section will test your ability to use Sub queries, aggregations & GROUP BY in hiveQL.

Write a hiveQL query to determine:

1. **What was apples stock when it was 1st traded (Earliest existing record)**

**SELECT \***

**FROM appl\_stock**

**ORDER BY stock\_date ASC**

**LIMIT 1;**

**SELECT \***

**FROM appl\_stock a**

**WHERE a.stock\_date IN(**

**SELECT MIN(b.stock\_date)**

**FROM apple\_stock b);**

1. **When was apples highest close stock value**

**SELECT stock\_date, MAX(close)**

**FROM appl\_stock**

**GROUP BY stock\_date**

**ORDER BY close DESC;**

1. **What date was apple's greatest volume traded**

**SELECT stock\_date**

**FROM appl\_Stock a**

**WHERE a.volume IN (**

**SELECT MAX(b.volume)**

**FROM appl\_stock b);**

1. **What is the average difference between high and adj close for apple**

**SELECT AVG(a.difference)**

**FROM (SELECT high-close AS difference**

**FROM appl\_Stock) a;**

1. **Determine the total volume traded each month for apple for 2015.**

**SELECT month(stock\_date), SUM(volume) As 'Total Trades'**

**FROM appl\_stock**

**WHERE year(stock\_date) = 2015**

**GROUP BY month(stock\_date)**

**ORDER BY month(stock\_date);**

1. **Break this out by years 2011-15, is there any kind of year on year trend/ pattern? or anomalies? Look into apples history to suggest why these occur?**

**SELECT year(stock\_date), month(stock\_date), SUM(volume) As 'Total Trades'**

**FROM appl\_stock**

**WHERE year(stock\_date) >= 2011**

**GROUP BY year(stock\_date), month(stock\_date)**

**ORDER BY 1,2;**

## 1.4 Querying across datasets:

This section will test your ability to write hiveQL queries across tables; you may need to use a combination of UNION, JOINS and built in hive functions.

You have been tasked with comparing the historic stock data for two of the social media industries biggest players (Facebook & Twitter).

Use a single query to answer the following questions:

Note: You will need to use joins or union logic

1. Retrieve and output to a file Facebook and Twitters stock price data for the calendar year 2015.

SELECT 'twtr' as company, \*

FROM twtr\_stock

WHERE year(stock\_date) = 2015

UNION

SELECT 'fb' as company, \*

FROM fb\_stock

WHERE year(stock\_date) = 2015;

Hive –f ‘insert/path/here’ > results.txt

1. Use a single query to identify the month of the year with the highest volume of trades for each company in 2015?

SELECT \*

FROM (

SELECT 'fb' as company, month(stock\_date), sum(volume) rank() over (order by sum(volume) DESC) rank

FROM fb\_stock

WHERE year(stock\_date) = 2015

GROUP BY month(stock\_date)) a

WHERE a.rank =1

UNION

SELECT \*

FROM (

SELECT 'twtr' as company, month(stock\_date), sum(volume) rank() over (order by sum(volume) DESC) rank

FROM twtr\_stock

WHERE year(stock\_date) = 2015

GROUP BY month(stock\_date)) a

WHERE a.rank =1;

1. Use a single query to identify the month of the year with the lowest volume of trades for each company in 2015?

SELECT \*

FROM (

SELECT 'fb' as company, month(stock\_date), sum(volume) rank() over (order by sum(volume) ASC) rank

FROM fb\_stock

WHERE year(stock\_date) = 2015

GROUP BY month(stock\_date)) a

WHERE a.rank =1

UNION

SELECT \*

FROM (

SELECT 'twtr' as company, month(stock\_date), sum(volume) rank() over (order by sum(volume) ASC) rank

FROM twtr\_stock

WHERE year(stock\_date) = 2015

GROUP BY month(stock\_date)) a

WHERE a.rank =1;

1. Alter your query to output the average volume of trades for each company & month of the year. Order the result set by company, average volume of trades.

SELECT \*

FROM (

SELECT 'fb' as company, month(stock\_date), avg(volume) rank() over (order by avg(volume) DESC) rank

FROM fb\_stock

WHERE year(stock\_date) = 2015

GROUP BY month(stock\_date)) a

UNION

SELECT \*

FROM (

SELECT 'twtr' as company, month(stock\_date), avg(volume) rank() over (order by avg(volume) DESC) rank

FROM twtr\_stock

WHERE year(stock\_date) = 2015

GROUP BY month(stock\_date)) a

ORDER BY company, rank;

1. Output question 4’s result set to a file & ensuring the query runs in silent mode

Hive –S –f ‘insert/path/here’ > queryOutput.txt

OR

Hive –S –e 'set hive.cli.print.header=true;

SELECT \*

FROM (

SELECT 'fb' as company, month(stock\_date), avg(volume) rank() over (order by avg(volume) DESC) rank

FROM fb\_stock

WHERE year(stock\_date) = 2015

GROUP BY month(stock\_date)) a

UNION

SELECT \*

FROM (

SELECT 'twtr' as company, month(stock\_date), avg(volume) rank() over (order by avg(volume) DESC) rank

FROM twtr\_stock

WHERE year(stock\_date) = 2015

GROUP BY month(stock\_date)) a

ORDER BY company, rank;' > queryOutput.txt

1. How do the results differ between the two companies?
2. Adapt your query to include the years 2013 & 2014. Is there any year or year trend/ pattern? Or anomalies? (Hint: Look for typically high/ low trading months)

**Hive –S –e 'set hive.cli.print.header=true;**

**SELECT \***

**FROM (**

**SELECT 'fb' as company, year(stock\_date) as year, month(stock\_date), avg(volume) rank() over (partition by year(stock\_date) order by avg(volume) DESC) rank FROM fb\_stock**

**WHERE year(stock\_date) > 2012**

**GROUP BY year(stock\_date), month(stock\_date)) a**

**UNION**

**SELECT \***

**FROM (**

**SELECT 'twtr' as company, year(stock\_date) as year, month(stock\_date), avg(volume) rank() over (partition by year(stock\_date) order by avg(volume) DESC) rank**

**FROM twtr\_stock**

**WHERE year(stock\_date) > 2012**

**GROUP BY year(stock\_date), month(stock\_date)) a**

**ORDER BY company, year, rank;'**

**> queryOutput.txt**

**Select \*,**

**CASE WHEN vol > LAG(vol) over(ORDER BY yr, mon)**

**THEN ‘Increase’**

**ELSE ‘Decrease’**

**END**

**FROM ( SELECT Month(stock\_date) AS mon,**

**YEAR(stock\_date) AS yr,**

**AVG(volume) AS vol**

**FROM apple**

**WHERE YEAR(stock\_date) IN (‘2013’,’2014’,’2015’)**

**GROUP BY MONTH(stock\_date),**

**YEAR(stock\_date)**

**)a**

**ORDER BY yr,**

**mon**

**UNION for Twitter**

# **Section 2 – San Francisco Salaries Mini Project:**

## 2.1 Scenario & Data:

One way to understand how a city government works is by looking at who it employs and how its employees are compensated.

The San\_Fran\_Salaries dataset contains the names, job title, and annual income data for San Francisco City employees from 2011 to 2014.

## 2.2 Task:

Put your new hive skills into practise to explore the San Francisco salaries dataset. You can focus on a specific idea, or keep it general. The angle for analysis is up to you!

The dataset is not perfect so you’ll have to watch out for and work around missing values and duplicate records.

To help get you started, here are some data exploration ideas:

• How have salaries changed over time between different groups of people (Job Titles/ Regions/ Job Type)?

• How are base pay, overtime pay, and benefits allocated between different groups?

• Is there any evidence of pay discrimination based on gender in this dataset?

Prepare a 5-10 minute presentation to present to the class at the end of the day, on your findings and angle for analysis. This task is intended to be completed individually, unless otherwise stated by the trainer.

CREATE EXTERNAL TABLE IF NOT EXISTS sf\_salaries(

Id int, EmpId int, EmpName varchar(50),Salutation char(5), JobTitle varchar(100), BasePay decimal(8,2),

OvertimePay decimal(8,2), OtherPay decimal(8,2), Benefits decimal(8,2), TotalPay decimal(8,2), TotalPayBenefits decimal(8,2),

Year int, Notes varchar(25), Agency varchar(25), status char(2), Region varchar(20))

COMMENT 'Source: Kaggle.com'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION 'user/adam.kershaw/san\_fran\_salaries'

tblproperties("skip.header.line.count"="1");

SELECT EmpName, JobTitle, TotalPay

FROM sf\_salaries

WHERE salutation <> 'Mr.'

AND TotalPay IS NOT NULL

ORDER BY TotalPay DESC

LIMIT 1;

## Additional Research Task

Note: This is only to be attempted if you have completed all previous tasks

In hive it’s possible to organise tables into **partitions and buckets**, to allow for more efficient querying.

* Research the use cases of partitioned tables, and how to create them
* Test out partitioning by re-creating the San\_Fran\_Salaries table
  + Partitions can be added when a table is created or altered

# Summary

Upon completion of this workbook’s exercises and hive mini project:

* Created Databases and Tables in Hive
* Projected structure upon data held in Hadoops Distributed File System
* Used hiveQL queries to analyse two large datasets from two large real-world datasets (Stocks, San Francisco Salaries)
* Put into practice hiveQL’s aggregation, filter, joins and union logic functionality.