SQL Workshop Final Review

UC Davis DSI

2018

1 Review

These questions are meant to be more open ended. Challenge yourself by answering each one in two different ways. This requires three types of knowledge: knowledge of the data / domain (finance), knowledge of the tool (SQL), and critical thinking. This is the next step beyond following instructions.

1. What are the most profitable companies?

Solution:

1. We can compute standard metrics that don't exist in the database such as return on assets (ROA). `DISTINCT` addresses a data quality issue by removing the duplicated rows.

```
SELECT DISTINCT company_name, net_income / asset as return_on_assets
FROM company_info
ORDER BY  return_on_assets DESC
;
```

2. Return on equity directly measures profitability for investors.

```
SELECT cn.sandp_company_name, fr.return_on_equity
FROM company_name as cn
    INNER JOIN financial_ratios as fr
ON cn.ticker = fr.ticker
ORDER BY fr.return_on_equity DESC
;
```

2. What are the most valuable sectors?

Solution: This is a question about sectors, so you should be thinking about aggregating figures up to the sector level, which means using `GROUP BY sector`. Use `ORDER BY` so that the most valuable show up on top.

1. Showing the assets in each sector in trillions of dollars rather than single units lets us readily read and compare them.

```
SELECT sector, SUM(asset) / 1e12 as trillions
FROM company_info
GROUP BY sector
ORDER BY trillions DESC
;
```

2. This computes the average earnings per share in each industry

```
SELECT sector, AVG(earning_per_share) as avg_earning_per_share
FROM company_info
GROUP BY sector
ORDER BY  avg_earning_per_share DESC
;
```

2 Joins

From Wikipedia:

> The Standard Industrial Classification (SIC) is a system for classifying industries by a four-digit code. Established in the United States in 1937, it is used by government agencies to classify industry areas.

Single Joins

1. Select the ticker, company name, and SIC code with corresponding description for companies that have SIC codes.

Solution:

```
SELECT ticker, company_name, sic_code, Description
FROM company_info
INNER JOIN sic
    ON company_info.sic_code = sic.SIC
;
```

2. Select the ticker, company name, and SIC code with corresponding description (possibly NULL) for all companies in the `company_info` table.

Solution:

```
SELECT ticker, company_name, sic_code, Description
FROM company_info
LEFT JOIN sic
    ON company_info.sic_code = sic.SIC
;
```

3. Which 4 digit SIC codes have the most companies?

Solution:

```
SELECT sic_code, Description, count(*) as cnt
FROM company_info
LEFT JOIN sic
    ON company_info.sic_code = sic.SIC
GROUP BY sic_code
ORDER BY cnt DESC
;
```

4. Modify the query above to only return SIC codes with at least 5 companies.

Solution:

```
SELECT sic_code, Description, count(*) as cnt
FROM company_info
LEFT JOIN sic
    ON company_info.sic_code = sic.SIC
GROUP BY sic_code
HAVING cnt >= 5
ORDER BY cnt DESC
;
```

Multiple Joins

5. Use inner joins to produce a table with columns for company name, state, SIC code, and SIC description.

Solution:

```
SELECT n.sandp_company_name, l.state, i.sic_code, s.Description
FROM company_info as i
INNER JOIN company_name as n
    ON n.ticker = i.ticker
INNER JOIN sic as s
    ON i.sic_code = s.SIC
INNER JOIN company_locations as l
    ON l.ticker = i.ticker
;
```

Or you may prefer this syntax:

```
SELECT n.sandp_company_name, l.state, i.sic_code, s.Description
FROM company_info as i
    , company_name as n
    , sic as s
    , company_locations as l
WHERE n.ticker = i.ticker
AND i.sic_code = s.SIC
AND l.ticker = i.ticker
;
```

6. Modify the above query to produce a table with columns for state, SIC code, SIC description, and count of companies located in that state with that SIC code.

Solution:

We no longer need company name, so we can remove the join to that table. Then we add a `GROUP BY` clause.

```
SELECT l.state,  i.sic_code, s.Description, count(*) as cnt
FROM company_info as i
INNER JOIN sic as s
    ON i.sic_code = s.SIC
INNER JOIN company_locations as l
    ON l.ticker = i.ticker
GROUP BY i.sic_code, l.state
```

```
ORDER BY l.state ASC, cnt DESC
;
```

7. Modify the query above to only return rows with a count of at least 2 companies.

Solution:

```
SELECT l.state,  i.sic_code, s.Description, count(*) as cnt
FROM company_info as i
INNER JOIN sic as s
    ON i.sic_code = s.SIC
INNER JOIN company_locations as l
    ON l.ticker = i.ticker
GROUP BY i.sic_code, l.state
HAVING cnt >= 2
ORDER BY l.state ASC, cnt DESC
;
```

3 Putting It All Together

1. Did the Brexit vote affect share prices for commercial banks in the United States? How?

   Use Google Sheets to show this with a plot.

Solution: First we need to find out when the Brexit vote took place. We can look online to find June 23, 2016. So we need to get share prices around this time.

We also need to identify the commercial banks. There are several ways to do this. We can make a list of banks, or use SIC codes (since the codes identify each company's industry). To find out the right SIC codes, we can look online or we examine a well-known bank in the database.

For example, the ticker for Citigroup is "C", so we can run the query

```
SELECT *
FROM company_info
WHERE ticker = 'C'
;
```

This gives us SIC code 6021. We can check all the companies with that SIC code with the query

```
SELECT *
FROM company_info
WHERE sic_code = 6021
;
```

The results appear to all be banks, so we can use 6021, but what if some banks use a different SIC code? To check this, we can look for companies in the "Banks" industry that have a different SIC code. A query to do this is

```
SELECT *
FROM company_info
WHERE industry = 'Banks'
  AND sic_code != 6021
;
```

This leads to three banks: Fannie Mae, Fifth Third Bancorp, and Citizens Financial. Fannie Mae is not a commercial bank (we can find this information online, for example on Wikipedia). Both Fifth Third Bancorp and Citizens Financial are commercial banks, and both have SIC code 6022. So we should also check 6022.

```
SELECT *
FROM company_info
WHERE sic_code = 6022
;
```

This query finds two more banks. So we can use all the companies listed under SIC code 6021 and 6022 as banks.

An even more careful strategy is to search for SIC codes that mention banks in the sic table.

Now we can get relevant share prices. First we select the tickers for banks:

```
SELECT ticker
FROM company_info
WHERE sic_code IN (6021, 6022)
;
```

Now we use this query as a subquery to get all the share prices around June 23, 2016:

```
SELECT date, ticker, open, close
FROM daily_share_prices
WHERE ticker IN (
  SELECT ticker
  FROM company_info
  WHERE sic_code IN (6021, 6022)
)
AND date BETWEEN '2016-06-16' AND '2016-07-23'
;
```

We could've used `JOIN` to find these instead of a subquery. This has the advantage that we could also include information that isn't in the daily share prices table (such as company names). Here's what the join looks like:

```
SELECT date, dsp.ticker, open, close
FROM company_info
JOIN daily_share_prices as dsp
    ON company_info.ticker = dsp.ticker
WHERE sic_code IN (6021, 6022)
AND date BETWEEN '2016-06-16' AND '2016-07-23'
;
```

Now we can save the share prices from our query to a comma-seperated values (CSV) file using the menus in the database browser. Google Sheets and Microsoft Excel can read CSV files, so we can use either to create plots of the data.

2. Based on the information in the database, what are the dominant industries in New York? How does New York compare to other states?

Solution: First we need to find where each company is located, so we can look at the company locations table. Since that table doesn't have the industry for each company, we need to join its information to the company info table. So we can write the first draft of our query as

```
SELECT L.ticker, L.company, L.state, R.industry
FROM
  company_location AS L
```

```
LEFT JOIN
  company_info as R
ON L.ticker = R.ticker
```

Since the goal is to compare industries in different states, we need information per state and industry, rather than per company. So we can change the query to group by state and industry to count the number of companies. Since we are aggregating, it no longer makes sense to keep the ticker and company name. The second draft of our query is

```
SELECT L.state, R.industry, COUNT(*) as freq
FROM
  company_location AS L
LEFT JOIN
  company_info as R
ON L.ticker = R.ticker
GROUP BY state, industry
```

Finally, the result is a lot easier to read if we sort by state and by the number of companies within each state. The third draft of our query is

```
SELECT L.state, R.industry, COUNT(*) as freq
FROM
  company_locations AS L
LEFT JOIN
  company_info as R
ON L.ticker = R.ticker
GROUP BY state, industry
ORDER BY state, freq DESC
```

This shows that New York state doesn't really seem to have a dominant industry, but there are 2 companies each for apparel, investment banking, insurance, and pharmaceuticals. There are also 2 companies with no industry listed (NULL) in the database. Of course, the database we're using only has a small amount of data so it probably doesn't reflect the real situation in New York.

Most other states also have only 1 or 2 companies in each industry. We can sort by number of companies (but not state) to see if any states do have a dominant industry. The query to do this is

```
SELECT L.state, R.industry, COUNT(*) as freq
FROM
  company_locations AS L
LEFT JOIN
  company_info as R
ON L.ticker = R.ticker
GROUP BY state, industry
ORDER BY freq DESC
```

The result shows that the oil industry is dominant in Texas (8 companies), while internet services and computer hardware are dominant in California (7 companies). California also has 5 companies with no industry listed. No other states show clearly dominant industries.