SELECT company.name

-- Table(s) to select from

from company

inner join fortune500

on company.ticker=fortune500.ticker;

==

-- Count the number of tags with each type

SELECT type, count(tag) AS count

FROM tag\_type

-- To get the count for each type, what do you need to do?

group by type

-- Order the results with the most common

-- tag types listed first

order by count desc;

==

-- Select the 3 columns desired

SELECT name, tag\_type.tag, tag\_type.type

FROM company

-- Join to the tag\_company table

inner JOIN tag\_company

ON company.id = tag\_company.company\_id

-- Join to the tag\_type table

inner JOIN tag\_type

ON tag\_company.tag = tag\_type.tag

-- Filter to most common type

WHERE type='cloud';

==

-- Use coalesce

SELECT coalesce(industry, sector, 'Unknown') AS industry2,

-- Don't forget to count!

count(\*)

FROM fortune500

-- Group by what? (What are you counting by?)

GROUP BY industry2

-- Order results to see most common first

order by count(\*) desc

-- Limit results to get just the one value you want

limit 1;

==

SELECT company\_original.name, title, rank

-- Start with original company information

FROM company AS company\_original

-- Join to another copy of company with parent

-- company information

LEFT JOIN company AS company\_parent

ON company\_original.parent\_id = company\_parent.id

-- Join to fortune500, only keep rows that match

inner JOIN fortune500

-- Use parent ticker if there is one,

-- otherwise original ticker

ON coalesce(company\_parent.ticker,

company\_original.ticker) =

fortune500.ticker

-- For clarity, order by rank

ORDER BY rank;

==

-- Select the original value

SELECT profits\_change,

-- Cast profits\_change

CAST(profits\_change as integer) AS profits\_change\_int

FROM fortune500;

==

-- Divide 10 by 3

SELECT 10/3,

-- Divide 10 cast as numeric by 3

10::numeric/3;

==

-- Select the count of each revenues\_change integer value

SELECT revenues\_change::integer, count(\*)

FROM fortune500

group by revenues\_change::integer

-- order by the values of revenues\_change

ORDER BY revenues\_change;

==

-- Count rows

SELECT count(\*)

FROM fortune500

-- Where...

WHERE revenues\_change > 0;

==

-- Select average revenue per employee by sector

SELECT sector,

avg(revenues/employees::numeric) AS avg\_rev\_employee

FROM fortune500

GROUP BY sector

-- Use the column alias to order the results

ORDER BY avg\_rev\_employee;

==

-- Divide unanswered\_count by question\_count

SELECT unanswered\_count/question\_count::numeric AS computed\_pct,

-- What are you comparing the above quantity to?

unanswered\_pct

FROM stackoverflow

-- eliminate rows where question\_count is not 0

WHERE question\_count != 0

limit 10;

==

-- Select sector and summary measures of fortune500 profits

SELECT min(profits),

avg(profits),

max(profits),

stddev(profits),

sector

FROM fortune500

-- What to group by?

GROUP BY sector

-- Order by the average profits

ORDER BY avg(profits);

==

-- Compute standard deviation of maximum values

SELECT stddev(maxval),

-- min

min(maxval),

-- max

max(maxval),

-- avg

avg(maxval)

-- Subquery to compute max of question\_count by tag

FROM (SELECT max(question\_count) AS maxval

FROM stackoverflow

-- Compute max by...

GROUP BY tag) AS max\_results; -- alias for subquery

==

-- Truncate employees

SELECT trunc(employees, -5) AS employee\_bin,

-- Count number of companies with each truncated value

count(\*)

FROM fortune500

-- Use alias to group

GROUP BY employee\_bin

-- Use alias to order

ORDER BY employee\_bin;

==

-- Truncate employees

SELECT trunc(employees, -4) AS employee\_bin,

-- Count number of companies with each truncated value

count(\*)

FROM fortune500

-- Limit to which companies?

WHERE employees < 100000

-- Use alias to group

GROUP BY employee\_bin

-- Use alias to order

ORDER BY employee\_bin;

==

-- Select the min and max of question\_count

SELECT min(question\_count),

max(question\_count)

-- From what table?

FROM stackoverflow

-- For tag dropbox

where tag = 'dropbox';

==

-- Create lower and upper bounds of bins

SELECT generate\_series(2200, 3050, 50) AS lower,

generate\_series(2250, 3100, 50) AS upper;

==

-- Bins created in previous step

WITH bins AS (

SELECT generate\_series(2200, 3050, 50) AS lower,

generate\_series(2250, 3100, 50) AS upper),

-- subset stackoverflow to just tag dropbox

dropbox AS (

SELECT question\_count

FROM stackoverflow

WHERE tag='dropbox')

-- select lower, upper, and count(\*)

SELECT lower, upper, count(\*)

-- from bins created above

FROM bins

-- join to dropbox and keep all rows from bins

left JOIN dropbox

-- Compare question\_count to lower and upper

ON question\_count >= lower

AND question\_count < upper

-- Group by lower and upper to count values in each bin

GROUP BY lower, upper

-- Order by lower to put bins in order

ORDER BY lower;

==

-- What groups are you computing statistics by?

SELECT sector,

-- Select the mean of assets with the avg function

avg(assets) AS mean,

-- Select the median

percentile\_disc(0.5) within group (order by assets) AS median

FROM fortune500

-- Computing statistics for each what?

GROUP BY sector

-- Order results by a value of interest

ORDER BY mean;

==

-- To clear table if it already exists;

-- fill in name of temp table

DROP TABLE IF EXISTS profit80;

-- Create the temporary table

create temp table profit80 AS

-- Select the two columns you need; alias as needed

SELECT sector,

percentile\_disc(0.8) within group (order by profits) AS pct80

-- What table are you getting the data from?

from fortune500

-- What do you need to group by?

group by sector;

-- See what you created: select all columns and rows

-- from the table you created

SELECT \*

FROM profit80;

==

-- Code from previous step

DROP TABLE IF EXISTS profit80;

CREATE TEMP TABLE profit80 AS

SELECT sector,

percentile\_disc(0.8) WITHIN GROUP (ORDER BY profits) AS pct80

FROM fortune500

GROUP BY sector;

-- Select columns, aliasing as needed

SELECT title, fortune500.sector,

profits, profits/pct80 AS ratio

-- What tables do you need to join? What type of join?

FROM fortune500

left JOIN profit80

-- How are the tables joined?

ON fortune500.sector=profit80.sector

-- What rows do you want to select?

WHERE profits > pct80;

==

-- To clear table if it already exists

DROP TABLE IF EXISTS startdates;

-- Create temp table syntax

CREATE temp table startdates AS

-- Compute the minimum date for each what?

SELECT tag,

min(date) AS mindate

FROM stackoverflow

-- What do you need to add to get a date for each tag?

group by tag;

-- Look at the table you created

SELECT \*

FROM startdates;

==

-- To clear table if it already exists

DROP TABLE IF EXISTS startdates;

CREATE TEMP TABLE startdates AS

SELECT tag, min(date) AS mindate

FROM stackoverflow

GROUP BY tag;

-- Select tag and mindate

SELECT startdates.tag,

mindate,

-- Select question count on the first and last days

a.question\_count AS min\_date\_question\_count,

b.question\_count AS max\_date\_question\_count,

-- Compute the difference of above

b.question\_count - a.question\_count AS change

-- Join startdates and one copy of stackoverflow

FROM startdates

INNER JOIN stackoverflow AS a

ON startdates.tag=a.tag

-- Condition for matching mindate

AND startdates.mindate=a.date

-- Join other copy of stackoverflow

INNER JOIN stackoverflow AS b

ON startdates.tag=b.tag

-- Condition for matching last date

AND b.date='2018-09-25';

==

DROP TABLE IF EXISTS correlations;

-- Create table

create temp table correlations AS

-- Select each correlation

SELECT 'profits'::varchar AS measure,

-- Compute correlations

corr(profits,profits) AS profits,

corr(profits,profits\_change) AS profits\_change,

corr(profits,revenues\_change) AS revenues\_change

FROM fortune500;

==

DROP TABLE IF EXISTS correlations;

CREATE TEMP TABLE correlations AS

SELECT 'profits'::varchar AS measure,

corr(profits, profits) AS profits,

corr(profits, profits\_change) AS profits\_change,

corr(profits, revenues\_change) AS revenues\_change

FROM fortune500;

-- Add a row for profits\_change

-- Insert into what table?

INSERT INTO correlations

-- Follow the pattern of the select statement above

SELECT 'profits\_change'::varchar AS measure,

corr(profits\_change, profits) AS profits,

corr(profits\_change, profits\_change) AS profits\_change,

corr(profits\_change, revenues\_change) AS revenues\_change

FROM fortune500;

-- Repeat the above, but for revenues\_change

INSERT INTO correlations

SELECT 'revenues\_change'::varchar AS measure,

corr(revenues\_change, profits) AS profits,

corr(revenues\_change, profits\_change) AS profits\_change,

corr(revenues\_change, revenues\_change) AS revenues\_change

FROM fortune500;

==

DROP TABLE IF EXISTS correlations;

CREATE TEMP TABLE correlations AS

SELECT 'profits'::varchar AS measure,

corr(profits, profits) AS profits,

corr(profits, profits\_change) AS profits\_change,

corr(profits, revenues\_change) AS revenues\_change

FROM fortune500;

INSERT INTO correlations

SELECT 'profits\_change'::varchar AS measure,

corr(profits\_change, profits) AS profits,

corr(profits\_change, profits\_change) AS profits\_change,

corr(profits\_change, revenues\_change) AS revenues\_change

FROM fortune500;

INSERT INTO correlations

SELECT 'revenues\_change'::varchar AS measure,

corr(revenues\_change, profits) AS profits,

corr(revenues\_change, profits\_change) AS profits\_change,

corr(revenues\_change, revenues\_change) AS revenues\_change

FROM fortune500;

-- Select each column, rounding the correlations

SELECT measure,

round(profits::numeric,2) AS profits,

round(profits\_change::numeric,2) AS profits\_change,

round(revenues\_change::numeric,2) AS revenues\_change

FROM correlations;

/\* Always need to cast double as numeric for round/trunc \*/

==