Thank you for a lovely take-home exercise! I hope this gives a solid sample of the type of work I try to do.

Data Scientist Product Analytics Exercise

January 4, 2022

Katt Kennedy

Scenario 1

TABLE_PURCHASES is a table that contains the following three fields: user_id, product, date_purchased.

Here is a sample of records from TABLE_PURCHASES:

TABLE_PURCHASES

user_id	product	date_purchased
00001	A	2015-01-10
00001	B	2014-11-23
00001	C	2015-05-01
00002	A	2014-10-01
00002	C	2014-12-23
00003	B	2015-02-15
00003	D	2014-09-23
00003	E	2014-06-01
00004	E	2014-12-14
00004	F	2015-03-03
.	.	.
.	.	.
.	.	.

QUESTION 1:

Write the SQL to generate the count of unique users who have purchased product B but have never purchased product C.

My goal here was to create two subqueries to filter out users; this is not the most performant solution, but it will get the job done. An alternative would be to create a CTE with columns totaling the count of each item purchased, which is preferable if this query needs to be repeated on a regular basis.

```
SELECT DISTINCT tp.user_id
FROM table_purchases AS tp
WHERE tp.user_id IN ( -sub query of users buying >=1 B
      SELECT COUNT(tp.product) AS count
      FROM table_purchases as tp
      WHERE tp.product = "B"
      AND count >= 1)
AND tp.user_id IN ( -sub query to create a user list who have
purchased exactly zero C
      SELECT COUNT(tp.product) AS count
      FROM table_purchases as tp
      WHERE tp.product = "C" AND count = 0)
```

TABLE_PRICES is a table that contains the following two fields: product, price.

Here is a sample of records from TABLE_PRICES:

TABLE_PRICES

product	price
A	$9.99
B	$4.99
C	$19.99
D	$14.99
E	$1.99
.	.
.	.
.	.

QUESTION 2:

Write the SQL to generate the count of unique users who purchased a product priced > $10 during May 2015.

The goal of this code is to add the price of each product to the base table, then filter out all transactions where the product was not more than $10. This assumes all transactions have only one product associated to the customer, and if a customer purchases multiple items totaling more than $10 than they should still not be included.

```
SELECT DISTINCT COUNT(tp.user_id) AS count_of_users
FROM table_purchases AS tpu
     LEFT JOIN table_prices AS tpr ON tpu.product = tpr.product
WHERE tpr.price > $10
```

Scenario 2

TABLE_WINS contains the numbers of games each Major League Baseball team has won during every season.

TABLE_WINS contains the following three fields: team, season, games_won.

Here is a sample of records from TABLE_WINS:

TABLE_WINS

team	season	games_won
San Francisco Giants	2010	92
San Francisco Giants	2011	86
San Francisco Giants	2012	94
San Francisco Giants	2013	76
San Francisco Giants	2014	88
Los Angeles Dodgers	2010	80
Los Angeles Dodgers	2011	82
Los Angeles Dodgers	2012	86
Los Angeles Dodgers	2013	92
Los Angeles Dodgers	2014	94
.	.	.
.	.	.
.	.	.

QUESTION 3:

Write the SQL to generate the following output, where the second column is the season in which each team had the most wins. The Giants & Dodgers are shown for illustrative purposes below, but the SQL should pull all teams at once:

team	season_with_most_wins
San Francisco Giants	????
Los Angeles Dodgers	????
.	.
.	.
.	.

Here, a subquery finds the season with the most total wins for each team, then returns that season year. This does not take into account win/loss ratios, as this was not an explicit part of the exercise.

```
SELECT DISTINCT tw.team
, ws.season AS season_with_most_wins
FROM table_wins tw
LEFT JOIN (SELECT team, season, MAX(games_won)
```

```
        FROM table_wins AS wins
        GROUP BY team, season
        -adding season here b/c all terms must be here or in an
aggregate)
ON tw.team = ws.team  AS ws
```

QUESTION 4:

Write the SQL to generate the following output, where the fourth column ("season_rank") is based on numbers of wins and is calculated by team. In other words for a given team the season with the most wins will be ranked 1, season with the next most wins will be ranked 2, and so on.

team	season	games_won	season_rank
San Francisco Giants	2010	92	?
San Francisco Giants	2011	86	?
San Francisco Giants	2012	94	?
San Francisco Giants	2013	76	?
San Francisco Giants	2014	88	?
Los Angeles Dodgers	2010	80	?
Los Angeles Dodgers	2011	82	?
Los Angeles Dodgers	2012	86	?
Los Angeles Dodgers	2013	92	?
Los Angeles Dodgers	2014	94	?
.	.	.	.
.	.	.	.
.	.	.	.

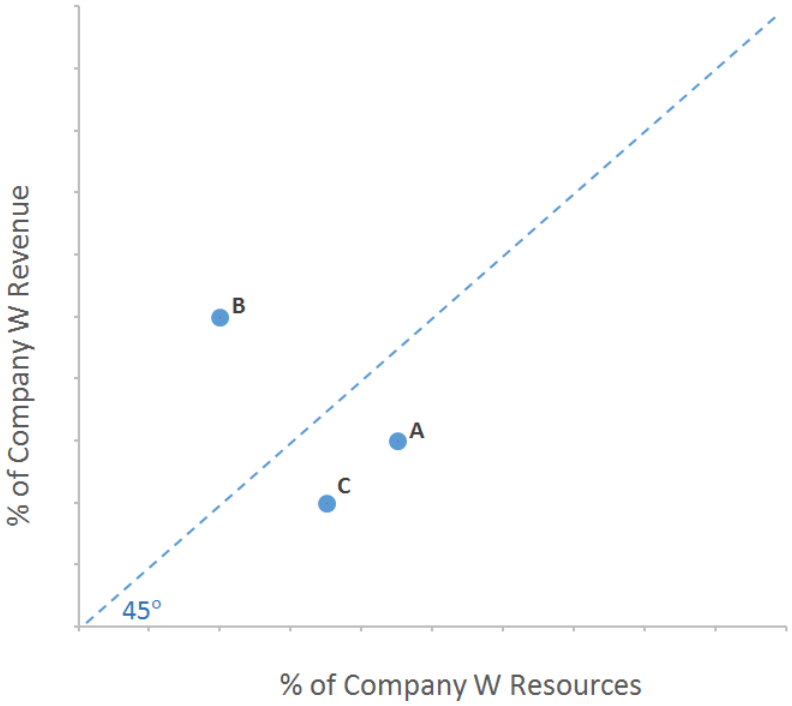
This is the tricky one! The trick is to use OVER PARTITION and RANK, so that:

1. *A partition is created by each season*
2. *The partition is ordered by the number of games won, descending*
3. *The rank/order is taken for each partition and returned to the overall query*

```
SELECT tw.team, tw.season, tw.games_won
      ,RANK() OVER(PARTITION BY tw.season ORDER BY games_won
DESC) AS season_rank
FROM table_wins AS tw
```

Scenario 3

Company W produces and sells exactly three products: Product A, Product B and Product C. The scatter plot below displays all three products. The x-axis displays the percent of Company W's resources allocated to each product. The y-axis displays the percent of Company W's revenue generated by each product.



QUESTION 5:

A. Assume you are the product manager of one of the products. Would you rather your product be below or above the 45 degree line? Explain.

Above the 45 degree line; this would mean the product has a larger than average share of company revenue and lower than average share of company resources, instead of the reverse.

B. Is it possible for all three products to be below the 45 degree line? Or all three to be above the 45 degree line? Explain.

It is not possible for all three products to be above or below the 45 degree line unless other products exist.

The line represents the average ratio of company resources to company revenue for a product line, and how each product relates to this ratio. While it is possible for all products to be profitable (or unprofitable) given different absolute values of Gross Revenue and Gross Resources, it is not possible for all three products to be under or above the line unless other products exist which have different ratios of Revenue: Resources.