**SITUATION**

You are a business analyst working at Deloitte. You already know SQL, and you have both a more junior analyst and a more senior analyst working on your team. In this exercise, you will help the junior analyst make their SQL commands work correctly and help the more senior analyst finish off some of their harder queries, and write some queries of your own.

Your client, the state of Iowa, has some urgent questions - questions that need answering. They are considering cracking down on alcohol consumption, but they don’t really understand the dynamics of the alcohol market in their state.

They hire your team of SQL experts and business problem solvers to analyze the data they have (conveniently located in SQLite databases) to solve the problem. Your team divvies up the work, and assigns some specific analyses to you.

**EXERCISES**

**QUESTION 1:**

The junior analyst wants to get the item numbers and descriptions of products with a case cost of at least 100 and pack size at least 12. She writes:

SELECT item\_no item\_description

FROM products

WHERE case\_cost >= 100

WHERE pack >= 12

Why isn’t this query working?

Needs a comma between item\_no and item\_description, second WHERE shoud be an AND

SELECT item\_no

, item\_description

FROM products

WHERE case\_cost >= 100

AND pack >= 12

**QUESTION 2:**

She now wants to get a list of all combinations of “category” and “vendor” in the data. She knows that she can select these two columns using:

SELECT category\_name, vendor\_name

FROM products

What else does she need to add to accomplish her goal?

A semi-colon?

**QUESTION 3:**

Her query to get all the products where proof is greater than 85 seems right, but it’s generating an error:

SELECT \* FROM products WHERE proof > 85;

How could you fix it?

Need to cast Proof to an integer:

CAST(proof as int)

**QUESTION 4:**

This analyst likes sweet drinks, and she wants to write a query that shows all the vendors that have products in a few categories she has identified:

‘PEACH BRANDIES’

‘FLAVORED VODKA’

‘FLAVORED RUM’

Her SQL has improved based on your feedback, so she correctly writes:

SELECT DISTINCT vendor\_name

FROM products

WHERE category\_name = “PEACH BRANDIES”

OR category\_name = “FLAVORED VODKA”

OR category\_name = “FLAVORED RUM”

How could you make this query shorter and more consistent with SQL best practices?

SELECT DISTINCT vendor\_name

, category\_name

FROM products

WHERE category\_name IN('PEACH BRANDIES','FLAVORED VODKA', 'FLAVORED RUM');

**QUESTION 5:**

Now you are given an area to analyze independently: the state of Iowa now wants you to do analysis on all “IMPORTED” products, as they want to know which foreign products are coming into their state.

First, select all products with “IMPORTED” in the category name.

SELECT DISTINCT vendor\_name

, category\_name

FROM products

WHERE category\_name LIKE '%IMPORTED%';

SELECT \*

FROM products

WHERE category\_name LIKE ‘%IMPORTED%’;

**QUESTION 6:**

*Pro Tip: The following several analyses are extremely common among analysts in the real world. Practice up!*

Now, get the top 10 vendors of imported products, ranked by sales.

SELECT total

,store

,category\_name

,description

,vendor

FROM sales

WHERE category\_name LIKE '%IMPORTED%'

ORDER BY total desc

LIMIT 10;

SELECT DISTINCT vendor

,total

,description

FROM SALES

ORDER by total desc

LIMIT 10;

ANSWER:

SELECT vendor

,SUM(total)

FROM sales

WHERE category\_name LIKE '%IMPORTED%'

GROUP BY vendor

ORDER BY 2 desc

LIMIT 10;

"Pernod Ricard USA/Austin Nichols"

"Constellation Wine Company Inc."

"Pernod Ricard USA/Austin Nichols"

"Pernod Ricard USA/Austin Nichols"

"Pernod Ricard USA/Austin Nichols"

"Pernod Ricard USA/Austin Nichols"

"Jim Beam Brands"

"Pernod Ricard USA/Austin Nichols"

"Bacardi U.S.A. Inc."

"Diageo Americas"

But we can’t do the DISTINCT, and the ORDER BY at the same time?

**QUESTION 7:**

Now, of the vendors that have greater than $100,000 in sales, which one has the highest average sales price? Order these vendors by highest to lowest average sales price.

//we’re going to have to get the # by multiplying the sales# with the price

//limit to over 100,000

//get the avg?

//order

SELECT vendor

,total

,btl\_price

, CAST(total \* CAST(btl\_price as decimal) as money)

,store

,category\_name

,description

FROM sales

WHERE (total \* CAST(btl\_price as decimal)) > 100000.00

ORDER BY money desc

LIMIT 45;

HOW COME SELECT DISTINCT ISN’T WORKING?!?!

It does here:

SELECT DISTINCT vendor

FROM sales

WHERE (total \* CAST(btl\_price as decimal)) > 100000.00

LIMIT 45;

What’s the way to name a column?

SELECT DISTINCT vendor

,ROUND(AVG(CAST(total as decimal)),2)

FROM sales

WHERE (total \* CAST(btl\_price as decimal)) > 100000.00

GROUP BY vendor

ORDER BY 2 desc

LIMIT 45;

**QUESTION 8:**

Now, the senior analyst wants you to work with her on a separate assignment: using the ‘fy17p’ and ‘fy18p’ databases, she wants to know, of all the stores in the database, which ones showed the largest increases and decreases in sales from fiscal year 2017 to fiscal year 2018 (Iowa has mislabeled the years in its dataset).

To do this, she knows you’ll have to go through several steps, so she has outlined them for you and given hints along the way:

First, create a new column in the sales\_2017 table that sums up the sales for all four “weeks” in the dataset, and give it an alias of “sales\_2017” by writing “AS sales\_2017” after the query:

select store\_name, sum(week1+week2+week3+week4) as sales\_2017

from fy17p

group by store\_name

Now do the same for 2018:

select store\_name, sum(week1+week2+week3+week4) as sales\_2018

from fy18p

group by store\_name

Now that you’ve done this, write a query that joins these two “sub”-tables on their common store\_name field. The outline for that, given to you by the senior analyst, will look like this:

SELECT \*

FROM (<code to generate table 1>) a

INNER JOIN (<code to generate table 2>) b

ON <your code here>

Now, for the final set of steps to complete the query: select the columns store\_name, sales\_2017 (from table 1), sales\_2018 (from table 2), and create a new variable “delta” that represents the change in sales from 2017 to 2018.

Finally, sort the final query by the new “delta” variable and you’re done!

**STRETCH QUESTION:**

You are proud of this work and you want to share it with the rest of the team. However, Question 8 results in a complicated query that would be tough for someone who wasn’t an advanced SQL user to pick up and understand.

Take the query above and add some comments explaining how it works, using the correct syntax for commenting in SQL. Aim to write a couple of full sentences at the beginning of the query explaining how it works, and add at least 3 shorter, 1 line comments throughout the code.