SELECT \*

FROM sales

LIMIT 100; -- store, county, item FK; total (numeric)

SELECT \*

FROM stores

LIMIT 100; -- store PK

SELECT \*

FROM products

LIMIT 100; -- item\_no PK; bottle\_price (money)

SELECT

sales.store,

ROUND(AVG(products.bottle\_price::numeric),2) AS avg\_btl\_price,

AVG(sales.total)::money AS avg\_total

FROM

sales

JOIN

products

ON products.item\_no = sales.item

JOIN

stores

ON stores.store = sales.store

WHERE

sales.category\_name ILIKE '%tequila%'

AND stores.store\_address ILIKE '%mason city%'

AND stores.store\_status = 'A'

GROUP BY sales.store

LIMIT 100;

SELECT \*

FROM products

LIMIT 100;

SELECT upc::bigint + scc::bigint -- null + non-null = null

FROM produtcs

LIMIT 100;

SELECT upc::bigint / scc:bigint

FROM products

LIMIT 100;

SELECT 20/NULLIF(5, 0);

SELECT NULLIF(0, 0);

SELECT 20 + COALESCE(NULL, 0);

SELECT COALESCE(upc, 'none given')

FROM products

LIMIT 100;

SELECT \*

FROM products

WHERE upc = NULL

LIMIT 100;

SELECT

county,

population,

CASE

WHEN population IS NULL THEN 'unknown'

WHEN population < 100000 THEN 'small'

WHEN population < 400000 THEN 'medium'

ELSE 'large'

END AS size\_class

FROM counties;

SELECT ROUND(AVG(

CASE

WHEN category\_name LIKE '%WHISK%' THEN 1

ELSE 0

END

) \* 100, 2) AS AverageWhiskyPct

FROM products;

SELECT

SUM(

CASE

WHEN category\_name LIKE '%WHISK%' THEN 1

ELSE 0

END

) AS whiskey\_products,

COUNT(1) AS total\_products,

(SUM(

CASE

WHEN category\_name LIKE '%WHISK%' THEN 1

ELSE 0

END

)::numeric

) / COUNT(1) \* 100 AS pct\_whiskey\_products

FROM products

SELECT

sales.total,

average.mean

FROM

sales

CROSS JOIN

(SELECT AVG(total) AS mean

FROM sales) AS average

LIMIT 100;

SELECT AVG(total) AS mean

FROM sales;

SELECT

sales.item,

COUNT( DISTINCT sales.store ) As store\_count

FROM sales

LEFT JOIN products

ON sales.item = products.item\_no

WHERE products.item\_no IS NULL

GROUP BY sales.item

LIMIT 100;

SELECT DISTINCT sales.store

FROM sales

JOIN stores

ON sales.store = stores.store

WHERE stores.store IS NULL

LIMIT 100;

/\*We want to see all of the information we can get on inactive

stores (if there are any) for sales, as well as their addresses.\*/

SELECT

stores.store\_address,

sales.\*

FROM

stores

LEFT JOIN

sales

USING(store)

WHERE stores.store\_status = 'I';

--or

SELECT

stores.store\_address,

sales.\*

FROM

sales

RIGHT JOIN

stores

USING(store)

WHERE stores.store\_status = 'I';

/\*Create one query that:

1. Identifies the items that were sold (by item number) that don’t have matching

reference information in the Products table (based on item\_no NULLs).

a. In the output, include a count of how many distinct stores sold these

items.

\*/

SELECT

sales.item,

products.item\_no,

COUNT(DISTINCT sales.store) AS number\_distinct\_stores

FROM

sales

LEFT JOIN

products

ON sales.item = products.item\_no

WHERE products.item\_no IS NULL

GROUP BY sales.item, products.item\_no

ORDER BY 3 DESC

LIMIT 100;

--or, but doesn't get the count of distinct stores

SELECT item FROM sales

EXCEPT

SELECT item\_no FROM products;

/\*Create another query that:

2. Lists the distinct store numbers with recorded sales but no listing in the

Stores table.

b. Use a CASE and COALESCE combination to add a column that

labels these records as either a liquor store or convenience store.

\*/

SELECT DISTINCT

sales.store,

CASE

WHEN COALESCE(sales.convenience\_store, 'N') = 'Y' THEN 'Convenience'

ELSE 'Liquor'

END AS case\_coalesce,

CASE

WHEN sales.convenience\_store = 'Y' THEN 'Convenience'

ELSE 'Liquor'

END AS case\_alone

FROM

sales

LEFT JOIN

stores

USING(store)

WHERE stores.store IS NULL

LIMIT 100;

/\*

1. Show the sales in the database completed at an active store.

a. Try connecting the keys with “USING.”

b. Limit to 1,000 rows.

c. Experiment with grouping and order.

\*/

SELECT \*

FROM sales

INNER JOIN stores

USING(store)

WHERE stores.store\_status = 'A'

LIMIT 1000;

/\*

2. Which sales included tequila products?

\*/

SELECT \*

FROM sales

WHERE sales.category\_name LIKE '%TEQUILA%'

LIMIT 1000;

/\*

3. Which tequila products were not sold?

\*/

SELECT item\_description

FROM products

LEFT JOIN sales

ON products.item\_no = sales.item

WHERE products.category\_name LIKE '%TEQUILA%' AND sales.store is NULL

LIMIT 1000;

/\*

4. Which distinct products were sold in Mason City, IA?

\*/

SELECT DISTINCT ON(sales.category\_name)

sales.category\_name,

stores.store\_address

FROM sales

INNER JOIN stores

USING(store)

WHERE stores.store\_address LIKE '%Mason City%'

LIMIT 1000;

/\*

5. Which Scotch whiskies were sold in Mason City, IA?

\*/

SELECT DISTINCT

description,

category\_name

FROM sales a

INNER JOIN stores b

USING(store)

WHERE store\_address LIKE '%Mason City%' AND category\_name LIKE '%SCOTCH WHISKIES%'

LIMIT 1000;

/\*

6. Which unique products, other than whiskies, were sold in Mason City, IA?

\*/

SELECT DISTINCT

a.description,

a.category\_name

FROM sales a

INNER JOIN stores b

USING(store)

WHERE a.category\_name NOT LIKE '%WHISKIES%' AND store\_address LIKE '%Mason City%'

LIMIT 1000;

/\*

7. As a check for data consistency, were there any sales of products that are not listed in the

Products table?

\*/

SELECT

a.description

FROM sales a

EXCEPT

SELECT

b.item\_description

FROM products b

LIMIT 1000;

/\*

8. As another check for data consistency, were there any sales at a store that doesn’t exist?

\*/

SELECT

store

FROM sales

EXCEPT

SELECT

store

FROM stores

LIMIT 1000;