In-Class Practice – (WHERE, ORDER BY)

Ordering

1. Select ID, name, address, city, state, zip from vendors and sort by state ascending. **Validate that you're getting three Arizona vendors and then California next. Note: it's okay if your first CA vendor isn't the same.**
2. Sort by state Z-A. **What's the 3rd vendor on the list? Compare with your partner**
3. Sort by state, city, and vendor name all ascending using column names
4. Update previous query to sort by column position numbers
5. Try adding a column alias and then sorting by that
6. Okay, if you are confused about what FETCH/OFFSET does, skip to #25 now. After that return to #7 below

Using WHERE clause to filter based on columns

7. Make a copy of your previously made vendor query and update it to only pull vendors in the states of NY and NJ without using an IN operator. **How many are there? Compare methods**
8. Make a copy of the previous query and update it to pull the same data but using an IN operator. **Did you get the same results? Discuss with partner which version you preferred.**
9. Make an updated version of the previous query to pull vendors ***not*** in NY and NJ

Using ROWNUM in WHERE clause

10. Pull the first 7 records from the Invoices table and then sort them from highest to lowest by invoice_total

Using expressions in WHERE clause

11. Select invoice_ids where "Balance Due" = 0. NOTE: "Balance Due" is invoice_total – payment_total – credit_total
12. Select customers from Customers_OM where the concatenation of First_Name " " Last Name = 'Korah Blanca'
13. Select customers from Customers_OM where the last name starts with 'Dam'. Try doing this with and without SUBSTR.
14. Go pull all invoices that are due in the next 120 days. *Hint: You want all invoices that have an invoice date between today and (today+120).*

WHERE with multiple conditions *using AND, OR*

15. Update previous query to remove invoices where payment_total isn't 0 (i.e. payment hasn't been made)
16. Select all vendors from where city is 'Sacramento' AND state is 'CA'
17. What will happen when we use OR?
18. What would the following return?

```
select *
from vendors
where not (vendor_state = 'CA' AND vendor_city = 'Sacramento')
```

Using expressions in LIKE/NOT keywords

19. Select all vendors where Vendor_phone is null
20. Select all vendors that contain the word "Gas" in their name

21. Select all vendors that contain the word “gas” in their name (note difference)

22. Select all vendors whose names start with “B”

23. What would the following return?
    select *
    from vendors
    where vendor_phone IS NOT NULL

Above and beyond

Using IN operator with a subquery

24. Select all vendors that have an invoice_total greater than $1000. Hint: you will want a subquery that selects vendor ID from invoices table then use the results of this as a subquery in the WHERE clause of a query that selects all vendors that are in this subquery results.

25. Update the previous query to only those vendors in CA.

FETCH & OFFSET

26. Run the 1st and 2nd queries and notice the difference in the rows being return. In the 1st query the WHERE clause filters the first 5 rows of the table and then sorts them. In the 2nd query the sort actually happens first and then the filter of the first 5 rows happens. In the last query we add in an OFFSET to skip the first 2 rows in the returned results and only return next 5 rows. The goal of this is to start to learn the order of operations in SQL. More can be learned here.

```
--Query using ROWNUM in WHERE
SELECT *
FROM invoices
WHERE ROWNUM <=5
ORDER BY invoice_id DESC;
```

```
--Query using FETCH after ORDER BY
SELECT *
FROM invoices
ORDER BY invoice_id DESC
FETCH FIRST 5 ROWS ONLY;
```

```
--Query using FETCH by OFFSET filter to start after 2 rows are skipped
SELECT *
FROM invoices
ORDER BY invoice_id DESC
OFFSET 2 ROWS FETCH NEXT 5 ROWS ONLY;
```