In-Class Practice (Multi-Table Selects)

Basic inner joins with table alias: The goal is to pull records from two different tables at once

1. Pull a list of all *vendors* and their associated *invoices*. Return all columns from both tables.

2. Pull a list of all *customer_om* and their associated *orders* but…this time **use a table alias** for both tables.

3. Update query from question #2 to only include customer_id, customer_name, order_id, and the order date.

Joining multiple table: The goal here is to join more than two (i.e. 3) tables at once

4. Make a copy of the query from #2 and then update that query to join TO ORDER_DETAILS and just show customer_om.customer_id, order_id, order_date, plus *order_details.item_id* and *order_details.order_qty*.

Outer joins

5. Pull all vendors and their associated invoice_ids even if we have no invoices from that vendor

6. Pull a query that joins Employees and Projects tables and returns the employee_id, first and last name, and the project number. Join tables with full outer join and then discuss what the null values mean with your partner(s)
   6.1 Now change to a left join, rerun the query, and discuss what results mean.
        6.1.1 Bonus: How would you update 7.1 to only show rows for employees without a project?
   6.2 Now change to a right join, rerun the query, and discuss what results mean
        6.2.1 Bonus: How would you update 7.2 to only show rows for projects without employees?

Unions

7. Select Customer_ID, City, and State from CUSTOMER_OM table where state is 'CA' or 'AZ'
   7.1 Add a literal string column with value of 'West' and give it a column alias of "Region"
   7.2 Create addition statements in same format for 'East' that contains states in 'DC','MD', 'NC', 'NY', 'NJ' and union it to the first statement
   7.3 Create a third statement in same format for the "Midwest" region that contains 'IA','OH', 'WI' and union it to the first two queries
   7.4 Lastly, add an order by column position 4 in ascending order

Intersect and Minus

8. Write a select statement of the distinct list of vendor_ids from Vendors

9. Write a 2nd statement of the distinct list of vendor_ids from Invoices

10. Minus these two statements. What is the result telling us?

11. Intersect these two statements. What is the result telling us?

In case you want more practice….

12. **Join more 3 or more tables** – Make a copy of the query from #1 and then update it to include only the vendor_name, the invoice_id, and invoice_date of each vendor along with invoice_sequence & line_item_amt from ITEM_LINE_ITEMS.

13. **Outer join** - If you want to practice more, pull all customers_om.customers_ids, customer_name, and their associated orders, even if they don't have any.

14. Let's see if there is any overlap between employees and customers_ex by comparing the first name and last name columns in both tables. NOTE: You can compare these datasets using multiple methods you've practiced in this class today.

15. Use the UNION operator to return a report that looks something like the one below that returns the vendor and their phone if it is known. If the vendor has no phone it just shows 'No phone'.

VENDOR_NAME	PHONE
1 ASC Signs	No phone
2 AT&T	No phone
3 Abbey Office Furnishings	(559) 555-8300
4 American Booksellers Assoc	(800) 555-0037
5 American Express	(800) 555-3344
6 Ascom Hasler Mailing Systems	No phone
7 Aztek Label	(714) 555-9000
8 BFI Industries	(559) 555-1551

16. Return a query that lists each vendor that has an invoice with their address, invoice date, invoice_total, and the term description and due date.