Database Administration Assignment 4

Breydon Brennan

CIT-4770 Data & Database Management

1. List those product (i.e., description, finish, standard price) made of ‘ASH’ Wood

```
SELECT PRODUCT_ID, PRODUCT_DESCRIPTION, PRODUCT_FINISH, STANDARD_PRICE FROM
PRODUCT_T
    WHERE PRODUCT_FINISH LIKE '%Ash';
```

	PRODUCT_ID	PRODUCT_DESCRIPTION	PRODUCT_FINISH	STANDARD_PRICE
1	2	Coffe Table	Natural Ash	200
2	3	Computer Desk	Natural Ash	375
3	6	8-Drawer Desk	White Ash	750
4	7	Dining Table	Natural Ash	800

2. List product lines and the average prices of product lines.

```
SELECT AVG(STANDARD_PRICE) AS "Average Price" FROM PRODUCT_T
    GROUP BY PRODUCT_LINE_ID;
```

	PRODUCT_LINE_ID	Average Price
1	1	250
2	2	531.25
3	3	450

3. How many orders did each customer place in November 2004?

```
-- selecting the cust id (CUSTOMER_T), oid (ORDER_T) and date (ORDER_T) with
INNER JOIN ON each of their id's
SELECT C.CUSTOMER_NAME, O.ORDER_ID, O.ORDER_DATE FROM ORDER_T O
    INNER JOIN CUSTOMER_T C ON O.CUSTOMER_ID = C.CUSTOMER_ID
        WHERE TO_CHAR(O.ORDER_DATE, 'MON-YY') LIKE '%NOV-04'; --simple
search with datatype conversion
```

	CUSTOMER_NAME	ORDER_ID	ORDER_DATE
1	Eastern Furniture	1009	05-NOV-04
2	Contemporary Casuals	1010	05-NOV-04

4. List product finishes whose average price exceeds $500

```
SELECT PRODUCT_FINISH, AVG(STANDARD_PRICE) FROM PRODUCT_T
    GROUP BY PRODUCT_FINISH
        HAVING AVG(STANDARD_PRICE) > 500;
```

	PRODUCT_FINISH	AVG(STANDARD_PRICE)
1	White Ash	750
2	Natural Maple	650

5. List, in alphabetical order, the product finish and the average standard price for each finish for selected finishes (Cherry, Natural Ash, Natural Maple, White Ash) having an average standard price less than $750

```
SELECT PRODUCT_FINISH, AVG(STANDARD_PRICE) FROM PRODUCT_T
    -- first check for selected finishes
    WHERE PRODUCT_FINISH IN ('Cherry', 'Natural Ash', 'Natural Maple',
'White Ash')
        -- group all instance of prod_fin having an average std_price over
750
        GROUP BY PRODUCT_FINISH
            HAVING AVG(STANDARD_PRICE) < 750
            -- alphabetical order
                ORDER BY PRODUCT_FINISH;
```

	PRODUCT_FINISH	AVG(STANDARD_PRICE)
1	Cherry	250
2	Natural Ash	458.333333333333333333333333333333333333
3	Natural Maple	650

6. Which customers (ID and Name) have not placed any orders

```
-- we use full outer join beause we need order date and cust info
SELECT C.CUSTOMER_ID, C.CUSTOMER_NAME FROM CUSTOMER_T C
    FULL OUTER JOIN ORDER_T O ON C.CUSTOMER_ID = O.CUSTOMER_ID
        WHERE ORDER_DATE IS NULL;
```

	CUSTOMER_ID	CUSTOMER_NAME
1	6	Furniture Gallery
2	7	Period Furniture
3	9	M and H Casual Furniture
4	10	Seminole Interiors
5	13	Heritage Furnishings
6	14	Kaneohe Homes

7. Display the order ID, customer ID, order date, and items (product ID and description) ordered for order number 1001

```
-- use order_line table to connect all the dependent tables through a number
of joins
SELECT O.ORDER_ID, C.CUSTOMER_ID, O.ORDER_DATE, P.PRODUCT_ID,
P.PRODUCT_DESCRIPTION FROM ORDER_LINE_T OL
    INNER JOIN ORDER_T O ON OL.ORDER_ID = O.ORDER_ID
    INNER JOIN PRODUCT_T P ON OL.PRODUCT_ID = P.PRODUCT_ID
    INNER JOIN CUSTOMER_T C ON O.CUSTOMER_ID = C.CUSTOMER_ID
        WHERE O.ORDER_ID = 1001; -- find order number
```

	ORDER_ID	CUSTOMER_ID	ORDER_DATE	PRODUCT_ID	PRODUCT_DESCRIPTION
1	1001	1	21-OCT-04	1	End Table
2	1001	1	21-OCT-04	2	Coffe Table
3	1001	1	21-OCT-04	4	Entertainment Center

8. List the products (ID and description) and the number of times the product has been purchased (label it as Purchase_Time), but only show those products that were purchased at least 3 times? (Hint: if a product was purchased 6 times, then this product will be associated with 6 orderlines)

```
-- We count the number of distinc OL_ID's to get the quantity of each
product in the order line
SELECT P.PRODUCT_ID, P.PRODUCT_DESCRIPTION, COUNT(DISTINCT OL.OL_ID) AS
"PURCHASE_TIME" FROM ORDER_LINE_T OL
    LEFT JOIN PRODUCT_T P ON OL.PRODUCT_ID = P.PRODUCT_ID -- left join to
get prod_id and desc
        GROUP BY P.PRODUCT_ID, P.PRODUCT_DESCRIPTION -- use group by to get
items from prev left join
            HAVING COUNT(DISTINCT OL.OL_ID) >= 3;
```

	PRODUCT_ID	PRODUCT_DESCRIPTION	PURCHASE_TIME
1	8	Computer Desk	3
2	3	Computer Desk	3
3	4	Entertainment Center	4

9. Find the products (ID and description) that were purchased by at least 3 customers

```
SELECT P.PRODUCT_ID, P.PRODUCT_DESCRIPTION FROM PRODUCT_T P
    RIGHT OUTER JOIN ORDER_LINE_T OL ON P.PRODUCT_ID = OL.PRODUCT_ID
        INNER JOIN ORDER_T O ON O.ORDER_ID = OL.ORDER_ID
            GROUP BY P.PRODUCT_ID, P.PRODUCT_DESCRIPTION
                HAVING COUNT(DISTINCT O.CUSTOMER_ID) >= 3
                    ORDER BY PRODUCT_ID;
```

	PRODUCT_ID	PRODUCT_DESCRIPTION
1	3	Computer Desk
2	4	Entertainment Center
3	8	Computer Desk

10. Which customers (ID and Name) purchased at least 3 different types of items in one order? (Hint: you may need to use subquery; if a customer purchased 5 different types of items in one order, then there will be 5 orderlines in that order)

```
SELECT C.CUSTOMER_ID, C.CUSTOMER_NAME FROM CUSTOMER_T C
    -- check if the customer is in subquery
    WHERE C.CUSTOMER_ID IN (
        SELECT O.CUSTOMER_ID FROM ORDER_T O
            JOIN ORDER_LINE_T OL ON O.ORDER_ID = OL.ORDER_ID
                GROUP BY O.ORDER_ID, O.CUSTOMER_ID
                    HAVING COUNT(DISTINCT OL.PRODUCT_ID) >= 3
    );
```

	CUSTOMER_ID	CUSTOMER_NAME
1	1	Contemporary Casuals
2	2	Value Furniture