Katrina Ying
Database Management Systems
Assignment 2
SQL – Due Sunday 1/28/24

1. **Retrieve a list of the employees in each state who were hired before the most recently hired person in that state.**
   Hint: In this query, you can list each employee information for each state in the order of minimum date to max date to simplify the query.

```
SELECT EmployeeId, EmployeeName, EmployeeAddress, EmployeeCity, EmployeeState,
EmployeeZip, EmployeeDateHired,
EmployeeBirthDate, EmployeeSupervisor
FROM employee_t
WHERE EmployeeDateHired < (
   SELECT
   MAX(EmployeeDateHired)
   FROM
   employee_t
   WHERE
   EmployeeState = EmployeeState )
   ORDER BY
   EmployeeState,
   EmployeeDateHired;
```

2. **Display the Employee ID and Employee Name for those employees who do not possess the skill Router. Display the results in order by Employee Name.**
   Hint: You can write a subquery here.

```
/* Query 2 - Display Employee ID and Employee Name for employees without the skill
Router. Use subquery.
Need to use employee_t and employeeskills_t and skill_t, also get rid of null values */
```

```
SELECT E.EmployeeID, E.EmployeeName
FROM employee_t E
WHERE NOT EXISTS (
     SELECT 1
     FROM employeeskills_t ES
   JOIN skill_t ST ON ES.SkillID = ST.SkillID
   WHERE ST.SkillDescription = 'Router'
   AND E.EmployeeID = ES.EmployeeID
```

```
)
ORDER BY E.EmployeeName;
```

3. **Display the salesperson name, product finish, and total quantity sold (label as TotalSales) for each finish by each salesperson.**
   Hint: The solution for this question requires multiple joins

```
SELECT SalespersonName, ProductFinish, SUM(OrderedQuantity) AS 'TotalSales'
FROM salesperson_t, order_t, customer_t, product_t, territory_t, doesbusinessin_t,
orderline_t
WHERE product_t.ProductID = orderline_t.ProductID
AND salesperson_t.SalesTerritoryID = territory_t.TerritoryID
AND territory_t.TerritoryID = doesbusinessin_t.TerritoryID
AND doesbusinessin_t.CustomerID = customer_t.CustomerID
And customer_t.CustomerID = order_t.CustomerID
AND order_t.OrderID = orderline_t.OrderID
GROUP BY SalespersonName, ProductFinish
ORDER BY TotalSales ASC;
```

4. **How many orders are there in the order table?**
   Hint: This is a very simple query.

```
/* Query 4 - How many orders in the order table? */

SELECT COUNT(*) OrderID
FROM order_t;
```

5. **From the product table, list the cheapest prices for specific product finishes such as the cheapest Cherry product, the cheapest Natural Ash, etc.**
   Hint: You can use a window function to solve this question.

```
/* Query 5 - From the product table, list the cheapest prices for specific product finishes
such as the cheapest Cherry product, the cheapest Natural Ash, etc.
Hint: You can use a window function to solve this question. Attempt 1, Rank did not work
below.

SELECT ProductID, ProductLineID, ProductDescription, ProductFinish, ProductStandardPrice
FROM (
    SELECT ProductID, ProductLineID, ProductDescription, ProductFinish,
ProductStandardPrice,
```

```
    RANK() OVER (PARTITION BY ProductFinish ORDER BY ProductStandardPrice) AS Rank
    FROM product_t
    ) RankedProducts
    WHERE
            Rank = 1; */

SELECT ProductFinish,
MIN(ProductStandardPrice) AS CheapestPrice
FROM product_t
GROUP BY ProductFinish
ORDER BY CheapestPrice ASC;
```

6. **What is the most expensive materialid in the supplies table?**

```
/* Query 6 - Most Expensive MaterialID in Supplies Table */

SELECT MaterialID,
MAX(SupplyUnitPrice) AS MostExpensive
FROM supplies_t
GROUP BY MaterialID
ORDER BY MostExpensive DESC
LIMIT 1;
```

7. **Retrieve a list of the raw materials and quantities required to produce a cherry end table (product with product description 'cherry end table'). The query should output the material ID, material name, material standard price, and quantity of raw material required. Sort the results with the most expensive raw materials listed first.**

```
/* Query 7 - Retrieve a list of the raw materials and quantities required to produce
a cherry end table (product with product description 'cherry end table').
The query should output the material ID, material name, material standard price, and quantity
of raw material required.
 Sort the results with the most expensive raw materials listed first.  Join rawmaterial_t,
product_t, uses_t */

 SELECT RM.MaterialID, RM.MaterialName, RM.MaterialStandardPrice, UT.QuantityRequired
 FROM rawmaterial_t RM
 JOIN uses_t UT ON RM.MaterialID = UT.MaterialID
 JOIN product_t PT ON UT.ProductID = PT.ProductID
 WHERE PT.ProductDescription = 'End Table' AND PT.ProductFinish = 'Cherry'
 ORDER BY RM.MaterialStandardPrice DESC;
```

8. **Retrieve the count of the materials with supply unit price less than 5 supplied by each vendor. Output the vendor name and the count of materials supplied that meets this supply unit price criterion. Sort in descending order of materials supplied.**

```
/* Query 8 - Retrieve the count of the materials with supply unit price less than 5 supplied by each vendor.
Output the vendor name and the count of materials supplied that meets this supply unit price criterion.
 Sort in descending order of materials supplied. Join vendor_t with supplies_t */


SELECT VendorName, COUNT(MaterialID) AS 'MaterialsCount'
 FROM vendor_T VT
 JOIN supplies_t ST ON VT.VendorID = ST.VendorID
 WHERE ST.SupplyUnitPrice < 5
 GROUP BY VT.VendorName
 ORDER BY MaterialsCount DESC;
```

9. **Retrieve the average price, maximum price, and minimum price of products within a product line, considering product lines 3 or 4. Output the product line ID, and the average, maximum, and minimum prices of all products within that product line. Sort in ascending order of product line ID.**

```
/* Query 9 - Retrieve the average price, maximum price, and minimum price of products within a product line,
considering product lines 3 or 4. Output the product line ID, and the average, maximum, and minimum prices of all products
within that product line. Sort in ascending order of product line ID. */

SELECT ProductLineID,
AVG(ProductStandardPrice) AS 'Average Price',
MAX(ProductStandardPrice) AS 'Max Price',
MIN(ProductStandardPrice) AS 'Min Price'
FROM product_t
WHERE ProductLineID IN (3, 4)
GROUP BY ProductLineID
ORDER BY ProductLineID ASC;
```

10. **Can you find the median supplyunitprice in the supplies table? Please explain how? If you can come up with the correct SQL query, that is great. If you cannot, please explain your thinking process only.**

```
WITH RankedSupplies AS (
```

```
  SELECT
    VendorID,
    MaterialID,
    SupplyUnitPrice,
    ROW_NUMBER() OVER (ORDER BY SupplyUnitPrice) AS RowAsc,
    ROW_NUMBER() OVER (ORDER BY SupplyUnitPrice DESC) AS RowDesc
  FROM supplies_t
)
SELECT AVG(SupplyUnitPrice) AS median_supplyunitprice
FROM RankedSupplies
WHERE RowAsc = RowDesc OR RowAsc + 1 = RowDesc OR RowAsc = RowDesc + 1;
```

```
/* Query 10 - Can you find the median supplyunitprice in the supplies table? Please explain
how? */

/* First count the number of rows - 1433 rows is an odd number. This requires me to calculate
the median manually.
Cross out numbers from each end of the set until you get the middle pair. Then get the average
of the middle pair */

SELECT COUNT(*)
FROM supplies_t;

/* Now calculate the median. First, create an alias for MedianPrice and get the average of
SupplyUnitPrice.
Next, the subquery orders the rows by ascending and descending which helps calculate the
median from each end of the set.
The line of code with "rows" makes sure that the rows are even, so that the median calculation
is as accurate as possible.
*/

SELECT
  AVG(SupplyUnitPrice) AS MedianPrice
FROM (
  SELECT
    SupplyUnitPrice,
    ROW_NUMBER() OVER (ORDER BY SupplyUnitPrice) AS RowAsc,
    ROW_NUMBER() OVER (ORDER BY SupplyUnitPrice DESC) AS RowDesc
  FROM
    supplies_t
) AS CalcMedian
WHERE
  RowAsc = RowDesc OR RowAsc + 1 = RowDesc OR RowAsc = RowDesc + 1;
```