Chapter 6

Exercises

All queries written for Chapter 6 exercises must be created using subqueries and not joins. The use of joins will result in a zero for the exercise.

1. Your accounting department has asked: Which products have a list price that’s greater than the average list price for all products? Create a report that contains the name of the product and the list price for each item. The report should be sorted by price, descending.

|  |
| --- |
| SELECT ProductName, ListPrice  FROM Products  WHERE ListPrice >  (SELECT AVG(ListPrice) AS AvgListPrice  FROM Products)  ORDER BY ListPrice DESC |
|  |
| 2 rows |
| We had to get the average list price and then grab list prices that are higher than that |
| In class notes |

1. A manager asks how many categories are available that have no related products. Write a subquery that gets a list of these categories.

Note: Use the NOT EXISTS operator

|  |
| --- |
| SELECT CategoryName  FROM Categories  WHERE NOT EXISTS  (SELECT CategoryID  FROM Products  WHERE Products.CategoryID = Categories.CategoryID) |
|  |
| 1 row |
| We got all the categories that had a product attached and then used not exists to get all the ones that don’t have that. |
| In class notes |

1. You are writing a report on wholesale prices and need to know the how many discount percentages are in use by only one product. Write a query that returns those unique discount percentages and the associated product name.

|  |
| --- |
| SELECT ProductName, DiscountPercent  FROM Products  WHERE DiscountPercent != (SELECT DiscountPercent  FROM Products  GROUP BY DiscountPercent  HAVING COUNT(\*) > 1) |
|  |
| 6 rows |
| We needed to get the discount percents that were not unique and use all of the other ones |
| Google and in class notes |

1. An account manager is trying to find out his customer’s oldest orders. Create a query for the report that will return one row per customer that contains the customer’s full name as a single column, customer’s email address, orderID, and OrderDate for the customer’s oldest order.

|  |
| --- |
| SELECT DISTINCT (FirstName + ' ' + LastName) AS FullName, EmailAddress,  (SELECT MIN(OrderDate) FROM Orders  WHERE Orders.CustomerID = Customers.CustomerID) AS OldestOrder  FROM Customers JOIN Orders  ON Customers.CustomerID = Orders.CustomerID  ORDER BY OldestOrder DESC; |
|  |
| 35 rows |
| We needed first and last name and email and a subquery to get the oldest order date |
| Joe Ghan’s help and in class notes |