SQL-4 Exercises

1. Using Subqueries

The **orion.Order_Fact** table contains information about orders that were placed by Orion Star Sales staff. Create a report that lists the Sales staff whose average quantity of items sold exceeds the company average quantity of items sold.

a. Write a query that displays the average **Quantity** for all rows in the table.

- Use **AVG(Quantity)** to calculate the average.
- Use the **orion.Order_Fact** table.

PROC SQL Output

```
_____________
1.747164
```

b. Write a query that displays **Employee_ID** and **AVG(Quantity)** for those employees whose average exceeds the company average. The query should do the following:

- Display the values for **Employee_ID** and **AVG(Quantity)**. Name the second column **MeanQuantity**.
- Use the **orion.Order_Fact** table.
- Group the data by **Employee_ID**.
- Include only groups where the employee's average quantity of items sold exceeds the company average. Use the query from step **1.a.** as a subquery in the HAVING clause.
- Add a title to the report as shown.

Partial PROC SQL Output

```
Employees whose Average Quantity Items Sold
  Exceeds the Company's Average Items Sold

 Employee_ID  MeanQuantity
_____________________________
      120127          2.20
      120128          2.00
      120134          1.83
```

2. Using a Noncorrelated Subquery

Each month a memo is posted that lists the employees who have employment anniversaries for that month. Create the report for February and list **Employee_ID** and the first and last names for all employees hired during the month of February of any year.

You can find **Employee_Name** in the **orion.Employee_Addresses** table and **Employee_Hire_Date** in the **orion.Employee_Payroll** table. Both tables contain the column **Employee_ID**. Order the report by an employee's last name.

a. Create a query that returns a list of employee IDs for employees with a February anniversary. The query should do the following:

- Display **Employee_ID** numbers.
- Use the **orion.Employee_Payroll** table.
- Return only employees whose **Employee_Hire_Date** isin February.
- Add a title to the report as shown.

Employee IDs for February Anniversaries

Employee_ID
120107
120116
120136
120162
120164
120167
120177
120194
120267
120658
120667
120671
120677
120715
120719
120750
120778
120806
121005
121007
121022
121030
121053
121070
121090
121106
121130

b. Using the query in step **2.a.** as a noncorrelated subquery, write a query that displays the employee IDs and names of employees who have February anniversaries. The final query should do the following:

- Display **Employee_ID** and split **Employee_Name** into two new columns: **FirstName** and **LastName**. Both new columns should have a length of $15 and appropriate labels. (See the report below.) The original **Employee_Name** is stored as **Lastname, Firstname**.
- Use the **orion.Employee_Addresses** table.
- Select only employee IDs for employees who had February anniversary months.
- Order the final results by **LastName**.
- Create an appropriate title.

Employees with February Anniversaries

Employee_ID	First Name	Last Name
121030	Jeryl	Areu
121007	John	Banaszak
120667	Edwin	Droste
120778	Angela	Gardner
120194	Reece	Harwood
121130	Gary	Herndon
121106	James	Hilburger
121070	Agnieszka	Holthouse
120658	Kenneth	Kennedy
120177	Franca	Kierce
121090	Betty	Klibbe
120671	William	Latty
120136	Atul	Leyden
121053	Tywanna	Mcdade
121005	Yuh-Lang	Mclamb
120715	Angelia	Neal
120806	Lorna	Ousley
120116	Austen	Ralston
120719	Roya	Ridley
120267	Belanda	Rink
120162	Randal	Scordia
120107	Sherie	Sheedy
120677	Suad	Sochacki
120164	Ranj	Stamalis
121022	Robert	Stevens
120167	Kimiko	Tilley
120750	Connie	Woods

3. Creating Subqueries Using the ALL Keyword

In most companies, you can assume that the higher-level job titles have employees that are older than employees with a lower-level job title. Using the **orion.Staff** table, determine whether there are any lower-level purchasing agents (Purchasing Agent I and Purchasing Agent II) that are older than all the higher-level purchasing agents (Purchasing Agent III). The final report should display **Employee_ID**, **Job_Title**, **Birth_Date**, and a calculated **Age** column for the employee as of 24Nov2007.

Hint: Use the SAS date constant (`'24Nov2007'd`) in the calculation for **Age**. If you use the TODAY function to calculate the age, the values might differ from the results below:

Level I or II Purchasing Agents
Who are older than ALL Purchasing Agent IIIs

Employee ID	Employee Job Title	Employee Birth Date	Age
120742	Purchasing Agent I	04FEB1948	59

4. Using Nested Subqueries

Orion Star Sales managers are interested in rewarding the top sales person at the company. The **orion.Order_Fact** table contains information about all sales, including the employee ID of the staff member responsible for making the sale. The **orion.Employee_Addresses** table contains the ID and name of every employee in the company.

Generate a report that shows **Employee_ID** and the respective staff member's calculated total sales figures from the **orion.Order_Fact** table. Calculate the total sales figures by summing the product of **Total_retail_price***Quantity**. The **Employee_ID** number 99999999 is a generic employee ID number that indicates an Internet sale for which no staff member can take credit. Exclude the **Employee_ID** number 99999999 when you determine the employee with the highest total sales.

a. Generate a report that shows **Employee_ID** and calculated **Total_Sales** from the **orion.Order_Fact** table.

Employee with the Highest Total Sales

Employee_ID	Total_Sales
121045	$8,446.70

b. Generate another report that displays **Employee_ID** and **Employee_Name** of the employee with the highest sales. The **orion.Employee_Addresses** table contains the employee names.

Name of the Employee with the Highest Total Sales

Employee_ID	Employee_Name
121045	Hampton, Cascile

c. Write a query that combines the two queries above in order to generate a report that adds the **Total_Sales** column with **Employee_ID** and **Employee_Name** and the calculated **Total_Sales** column.

Hint: To solve this problem, you can use an in-line view.

Employee with the Highest Sales

Employee Identification Number	Employee Name	Total Sales
121045	Hampton, Cascile	$8,446.70

5. Creating a Simple Correlated Subquery

Create a report showing **Employee_ID** and the birth month (calculated as **month(Birth_date)**) for all Australian employees, using a correlated subquery.

The table **orion.Employee_Payroll** contains **Employee_ID** and **Birth_Date**.

In the subquery, select only **Country** from **orion.Employee_Addresses**. Use a WHERE clause to return only rows where the **EmployeeID** in **orion.Employee_Addresses** matches the **EmployeeID** in **orion.Employee_Payroll**.

Order the report by birth month.

Partial PROC SQL Output

Australian Employees' Birth Months

Employee_ID	Birth Month
120145	1
120198	1
120147	1
120127	1
120191	1

6. Using a Correlated Subquery

Generate a report that displays **Employee_ID**, **Employee_Gender**, and **Marital_Status** for all employees who donate more than 0.02 of their salary. The table **orion.Employee_donations** contains **Employee_ID**, quarterly donations (**Qtr1**-**Qtr4**), and charities (**Recipients**). The **orion.Employee_Payroll** table contains **Employee_ID** and **Salary** information.

Partial PROC SQL Output

Employees With Donations > 0.02 Of Their Salary

Employee_ID	Employee_ Gender	Marital_ Status
120267	F	M
120680	F	S
120686	F	M
120689	F	S
120753	M	S