Lab 05 – Multi-Table Queries and Views

# Objective:

The purpose of this lab is to introduce students to querying data from multiple tables. Relationships are used in relational databases to reduce redundant and repetitive data, but it is necessary to reconnect these tables when extracting data and obtaining information. Student will be able to:

* produce query results containing data from multiple tables using ANSI-92 joins and demonstrate their knowledge of inner, outer, and full joins.
* To actively troubleshoot queries to handle potentially ambiguous fields across multiple tables using aliases.
* Students learn to create and modify views.

# Submission:

***Your submission will be a single WORD file with the solutions provided.***

Your submission needs to follow the same question order and clearly indicate the answers to each question. Make sure every SQL statement terminates with a semicolon.

**ALL questions must be answered using ANSI-92 JOINs unless otherwise stated**. ANSI-89 are obsolete and should not be used in new query derivations. We only teach them in case you see them in the workplace, that you know what they are and how they work.

# Tasks:

## **In this lab, you will use the 8 tables created in lab-1 and tables from Airlines script.**

## **Please download and run the Airline script given in Lab-5.**

## **If you have deleted any tables from lab-1, Please download and run the lab-1 retail application script given in Lab-5**

## **Hint:**

## Default JOIN statements should be ANSI-92, unless asked to do in ANSI-89

## Use table aliases and field aliases wherever necessary.

## Don’t use \* for all the SELECT, please use only the field name asked and in the order given. If no field names mentioned use all fieldnames.

## Use SELECT SYSDATE FROM DUAL; to find your date format to use in all SELECT OR INSERT OR UPDATE queries.

## Please use TO\_CHAR(column name,'DD-MON-YYYY: HH24:MI:SS') to display in SELECT and use TO\_DATE(column name,'DD-MON-YYYY: HH24:MI:SS') to insert date and time together

## Provide SQL query in text form followed by screenshots of the results for all the questions

## Select data from multiple tables

## Display the following details of the flight who have an Airline company whose destination have a letter ‘O’ in them.

## FlightID, FlightNumber, Origin, Destination, Departure Time, ArrivalTime, AirlineName and Country.

## SELECT

## Flights.FlightID,

## Flights.FlightNumber,

## Flights.Origin,

## Flights.Destination,

## Flights.DepartureTime,

## Flights.ArrivalTime,

## Airlines.AirlineName,

## Airlines.Country

## FROM

## Flights

## JOIN

## Airlines ON Flights.AirlineID = Airlines.AirlineID

## WHERE

## UPPER(Flights.Destination) LIKE '%O%';

## 

## Display the following details of the flight who have an Airline company or not

## FlightID, FlightNumber, Origin, Destination, Departure Time, ArrivalTime, AirlineName and Country.

## SELECT

## Flights.FlightID,

## Flights.FlightNumber,

## Flights.Origin,

## Flights.Destination,

## Flights.DepartureTime,

## Flights.ArrivalTime,

## Airlines.AirlineName,

## Airlines.Country

## FROM

## Flights

## LEFT OUTER JOIN

## Airlines ON Flights.AirlineID = Airlines.AirlineID;

## 

## Display the following details of all the Airline company who have operating flights under them or not.

## FlightID, FlightNumber, Origin, Destination, Departure Time, ArrivalTime, AirlineName and Country.

## SELECT

## Flights.FlightID,

## Flights.FlightNumber,

## Flights.Origin,

## Flights.Destination,

## Flights.DepartureTime,

## Flights.ArrivalTime,

## Airlines.AirlineName,

## Airlines.Country

## FROM

## Flights

## RIGHT OUTER JOIN

## Airlines ON Flights.AirlineID = Airlines.AirlineID;

## 

## Display the following details of the flights who have not been assigned to an Airline company

## FlightID, FlightNumber, Origin, Destination, DepartureTime, ArrivalTime, AirlineName and Country.

SELECT

    Flights.FlightID,

    Flights.FlightNumber,

    Flights.Origin,

    Flights.Destination,

    Flights.DepartureTime,

    Flights.ArrivalTime,

    Airlines.AirlineName,

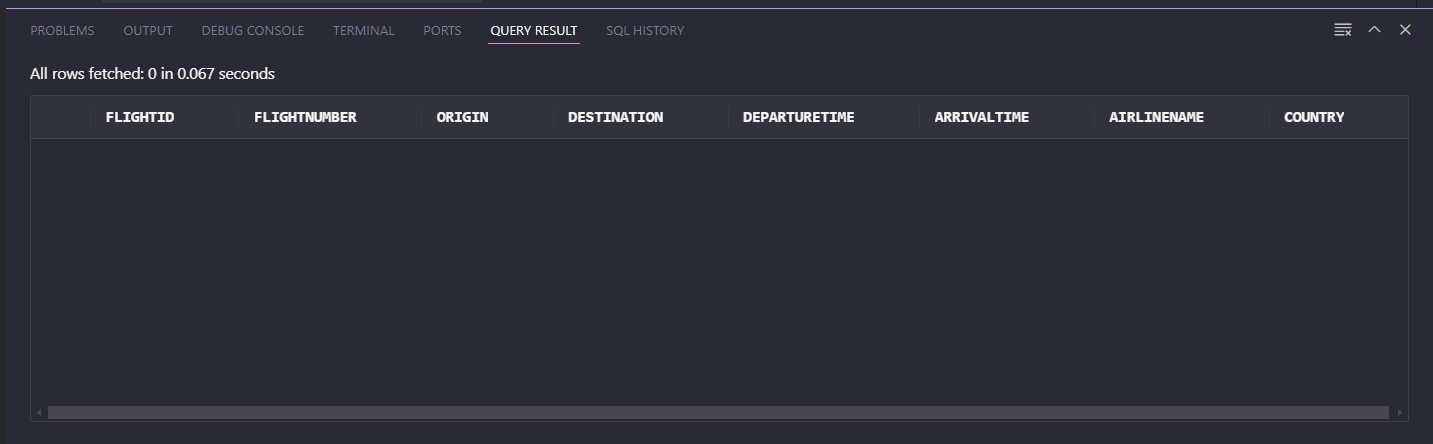
    Airlines.Country

FROM

    Flights

LEFT OUTER JOIN

    Airlines ON Flights.AirlineID = Airlines.AirlineID

WHERE Flights.AirlineID IS NULL;

## No record is found as each flight has be assigned with an Airline

## Display the following details of the Airline company who don’t have flights assigned to them

## AirlineName, Country FlightID, FlightNumber, Origin, Destination, DepartureTime, ArrivalTime

SELECT

    Flights.FlightID,

    Flights.FlightNumber,

    Flights.Origin,

    Flights.Destination,

    Flights.DepartureTime,

    Flights.ArrivalTime,

    Airlines.AirlineName,

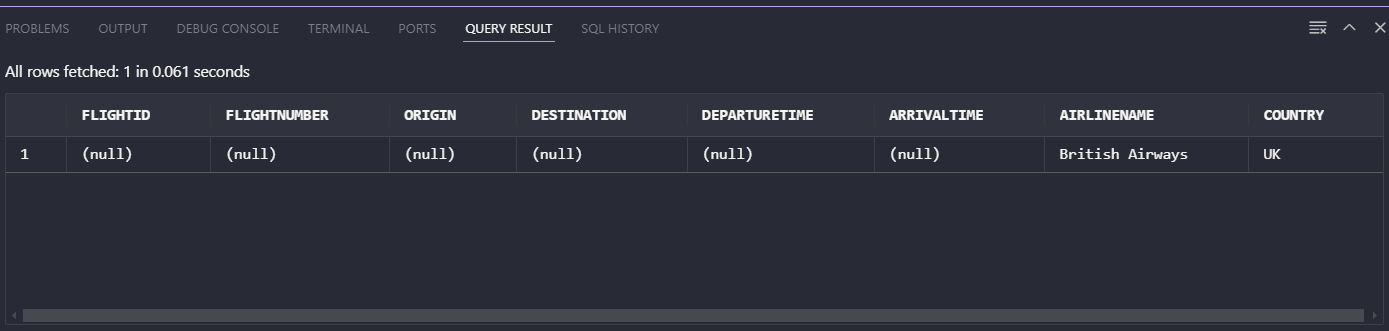
    Airlines.Country

FROM

    Flights

RIGHT OUTER JOIN

    Airlines ON Flights.AirlineID = Airlines.AirlineID

WHERE Flights.AirlineID IS NULL;

## Display the following details of the passengers who have been assigned a flight

## firstname, last name, seat number, origin, destination, departure time, arrival time

SELECT p.firstname, p.lastname, p.seatnumber, f.origin, f.destination, f.departuretime, f.arrivaltime

FROM passengers p

JOIN flights f ON p.flightid = f.flightid

## 

## Display the following details of the passengers who have been assigned a flight or not been assigned a flight

## First name, last name, seat number, origin, destination, departure time, arrival time

SELECT p.firstname, p.lastname, p.seatnumber, f.origin, f.destination, f.departuretime, f.arrivaltime

FROM passengers p

LEFT JOIN flights f ON p.flightid = f.flightid;

## 

## 

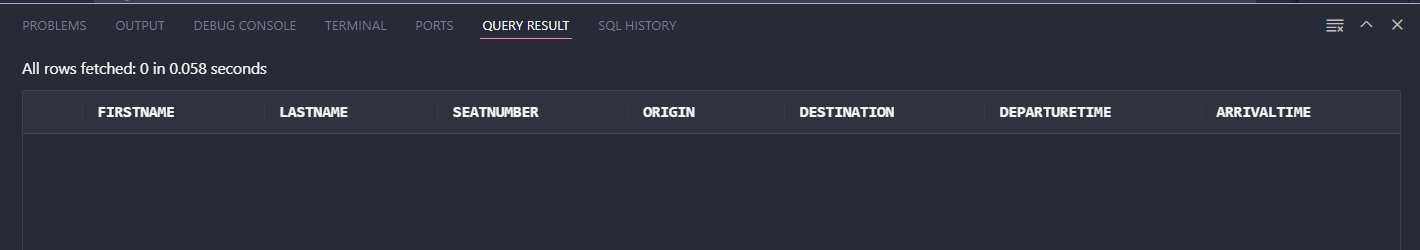
## Display the following details of the passengers who have not been assigned a flight

## First name, last name, seat number, origin, destination, departure time, arrival time

SELECT p.firstname, p.lastname, p.seatnumber, f.origin, f.destination, f.departuretime, f.arrivaltime

FROM passengers p

LEFT JOIN flights f ON p.flightid = f.flightid

WHERE p.flightid IS NULL; 

## No record is found as every passenger is assigned with a flight

## Display the following details of the flight who have been assigned a passenger along with airline information

## AirlineName, Country, first name, last name, seat number, origin, destination, departure time, arrival time

SELECT a.AirlineName, a.Country, p.firstname, p.lastname, p.seatnumber, f.origin, f.destination, f.departuretime, f.arrivaltime

FROM Airlines a

JOIN FLIGHTS f ON a.AIRLINEID = f.AIRLINEID

JOIN PASSENGERS p ON p.flightid = f.flightid;



## Display the following details of the passenger who have not been assigned a flight or airline

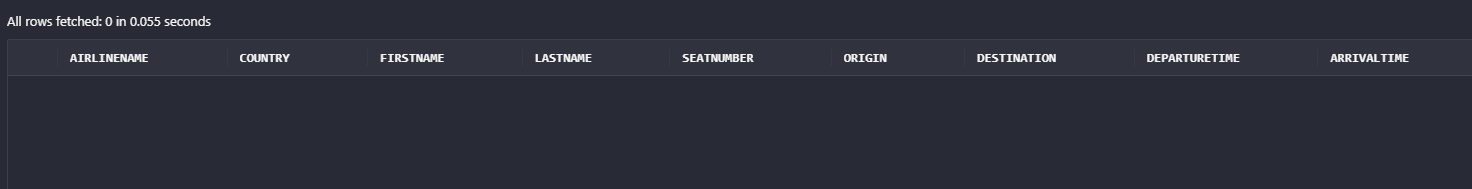
## AirlineName, Country, first name, last name, seat number, origin, destination, departure time, arrival time

SELECT a.AirlineName, a.Country, p.firstname, p.lastname, p.seatnumber, f.origin, f.destination, f.departuretime, f.arrivaltime

FROM Airlines a

JOIN FLIGHTS f ON a.AIRLINEID = f.AIRLINEID

RIGHT JOIN PASSENGERS p ON p.flightid = f.flightid

WHERE p.flightid IS NULL; 

## No record is found as every passenger is assigned with a flight

## Create a View of the SELECT query as “Passenger\_Airline\_Flight” in question 9. Display the data of the view using SELECT query.

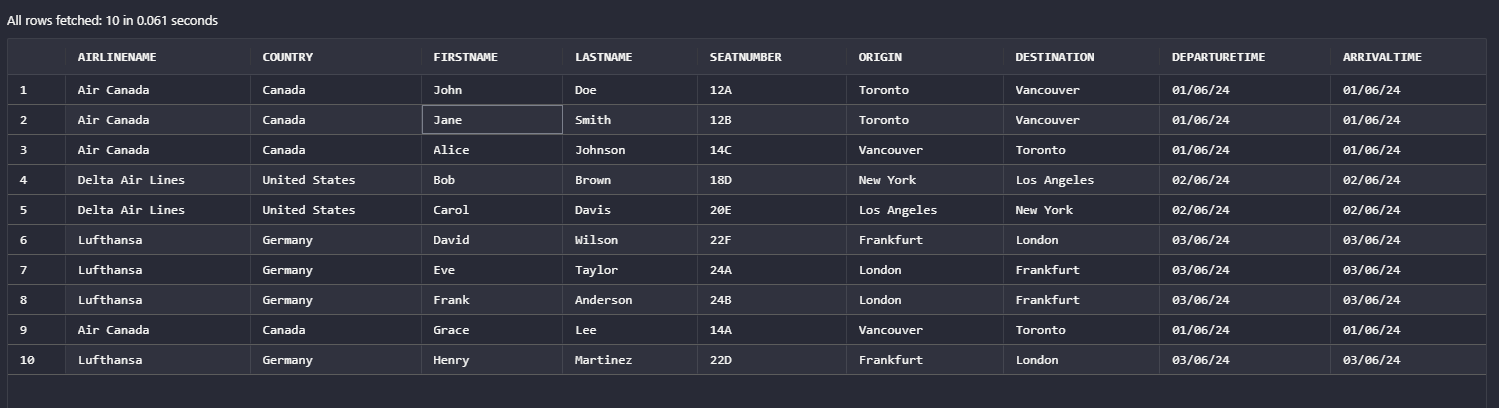
CREATE VIEW Passenger\_Airline\_Flight AS

SELECT a.AirlineName, a.Country, p.firstname, p.lastname, p.seatnumber, f.origin, f.destination, f.departuretime, f.arrivaltime

FROM Airlines a

JOIN FLIGHTS f ON a.AIRLINEID = f.AIRLINEID

JOIN PASSENGERS p ON p.flightid = f.flightid;

SELECT \* FROM Passenger\_Airline\_Flight; 

## Create a query that shows retail customers first name and last name along with their sales rep employee number and their first name, last name, city, phone number and postal code for all retail customers who live in Singapore.

## Answer this question using an ANSI-89 Join

SELECT rc.CONTACTLASTNAME, rc.CONTACTFIRSTNAME,

    re.EMPLOYEENUMBER, re.FIRSTNAME, re.LASTNAME, rO.CITY, rO.PHONE, rO.POSTALCODE

FROM RETAILCUSTOMERS rc, RETAILEMPLOYEES re, RETAILOFFICES ro

WHERE rc.SALESREPEMPLOYEENUMBER = re.EMPLOYEENUMBER

AND re.OFFICECODE = rO.OFFICECODE

AND rc.CITY = 'Singapore';

## Answer this question using an ANSI-92 Join

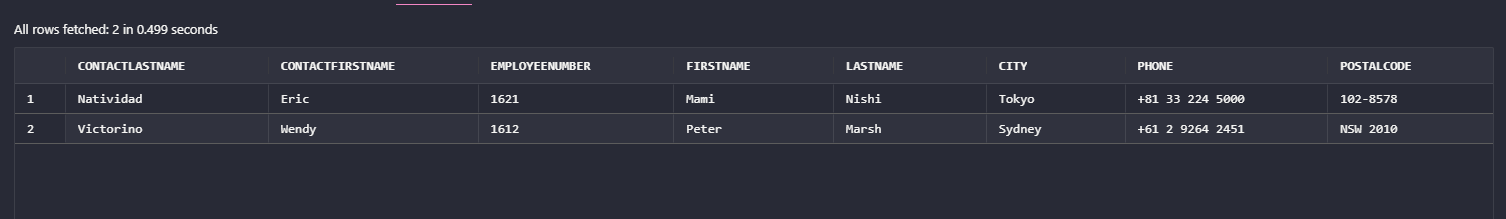
SELECT rc.CONTACTLASTNAME, rc.CONTACTFIRSTNAME,

    re.EMPLOYEENUMBER, re.FIRSTNAME,re.LASTNAME, rO.CITY, rO.PHONE, rO.POSTALCODE

FROM RETAILCUSTOMERS rc

JOIN RETAILEMPLOYEES re ON rc.SALESREPEMPLOYEENUMBER = re.EMPLOYEENUMBER

JOIN  RETAILOFFICES ro ON re.OFFICECODE = rO.OFFICECODE

WHERE rc.CITY = 'Singapore'; 

## Create a query that displays all retail payments made by retail customers from USA.

## Sort the output by Customer Number. Only display the Customer Number, Customer Name, Country, Payment Date and amount.

## Make sure the payment date is displayed clearly in this format 4-Feb-2019.

SELECT  rc.CUSTOMERNUMBER, rc.CONTACTFIRSTNAME || ' ' || rc.CONTACTLASTNAME AS CUSTOMERNAME,

    rc.COUNTRY, TO\_CHAR(rp.PAYMENTDATE, 'DD-MON-YY') AS PAYMENTDATE, rp.AMOUNT

FROM RETAILCUSTOMERS rc

JOIN RETAILPAYMENTS rp ON rc.CUSTOMERNUMBER = rp.CUSTOMERNUMBER

WHERE UPPER(rc.COUNTRY) = 'USA'

ORDER BY rc.CUSTOMERNUMBER ASC; 

## Create a query that shows all Canada retail customers who have or have not made a payment. Display only the customer number, customer name, amount sorted by customer number.

SELECT  rc.CUSTOMERNUMBER, rc.CONTACTFIRSTNAME || ' ' || rc.CONTACTLASTNAME AS CUSTOMERNAME,

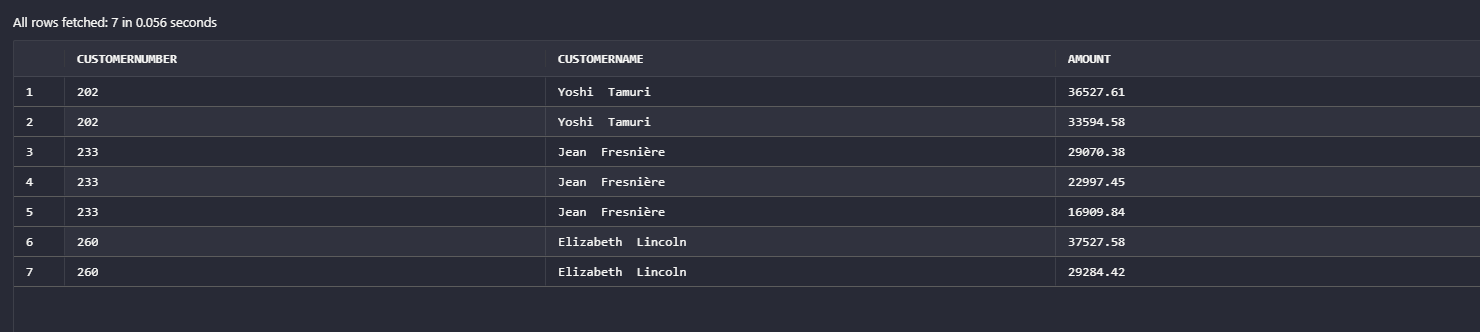
    rp.AMOUNT

FROM RETAILCUSTOMERS rc

LEFT JOIN RETAILPAYMENTS rp ON rc.CUSTOMERNUMBER = rp.CUSTOMERNUMBER

WHERE LOWER(rc.COUNTRY) = 'canada'

ORDER BY rc.CUSTOMERNUMBER ASC;



## Display all the retail orders with quantity ordered, price of each item, who have their order shipped and who live in Denmark.

SELECT  od.ORDERNUMBER, od.QUANTITYORDERED, od.PRICEEACH, rc.CUSTOMERNUMBER

FROM ORDERDETAILS od

JOIN RETAILORDERS ro ON od.ORDERNUMBER = ro.ORDERNUMBER

JOIN RETAILCUSTOMERS rc ON ro.CUSTOMERNUMBER = rc.CUSTOMERNUMBER

WHERE ro.STATUS = 'Shipped'

AND

LOWER(rc.COUNTRY) = 'denmark';



## Create a view (vwProductOrder) to list all the retail products with the following data:

## Product code, product name, msrp, buy price, quantity ordered, and price for each product in every order.

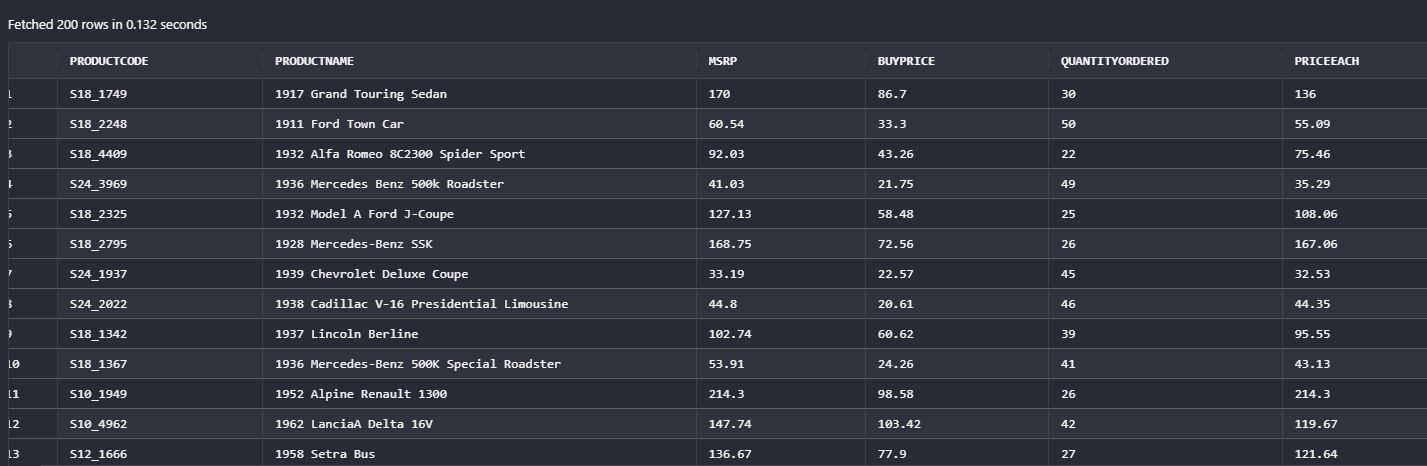
## b) Write a statement to view the results of the view just created.

CREATE VIEW vwProductOrder AS

SELECT rp.PRODUCTCODE, rp.PRODUCTNAME, rp.MSRP, rp.BUYPRICE, od.QUANTITYORDERED, od.PRICEEACH

FROM RETAILPRODUCTS rp

JOIN ORDERDETAILS od ON rp.PRODUCTCODE = od.PRODUCTCODE;

SELECT \* FROM vwProductOrder; 

## Using the vwProductOrder view, display the product order information with product name, buyprice ,order line number anwhose buy price is in the range from $30 to $40 and whose product code starts with ‘s32’. Sort the output based on product name and then buy price. (Hint: orderLineNumber is not in the view then how can you get in this query?)

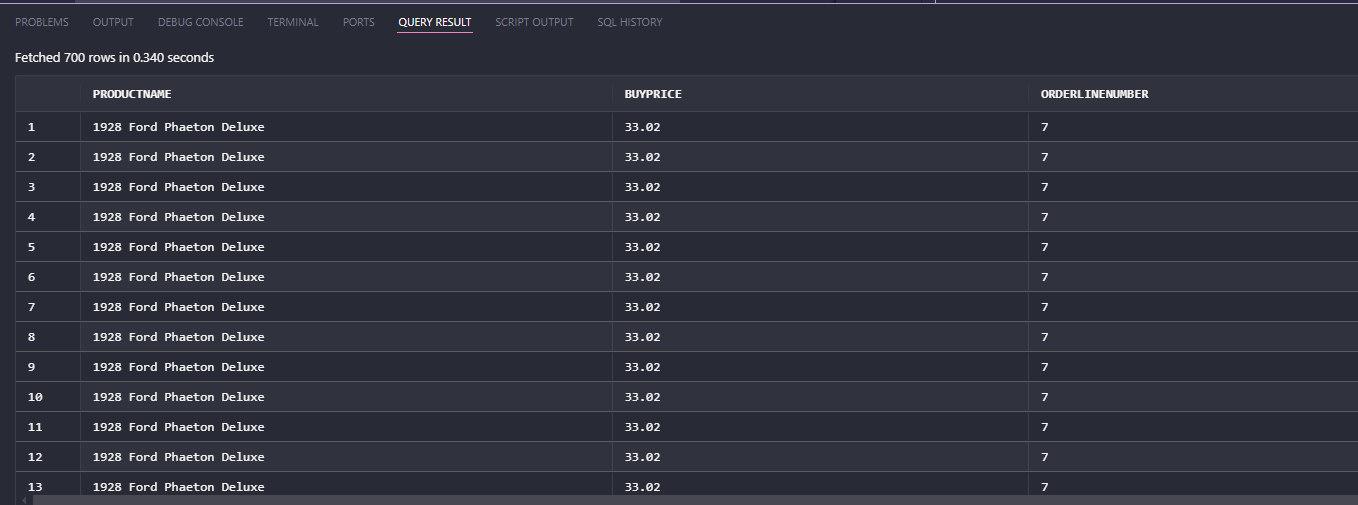
SELECT vwp.PRODUCTNAME, vwp.BUYPRICE, od.ORDERLINENUMBER

FROM vwProductOrder vwp

JOIN ORDERDETAILS od ON vwp.PRODUCTCODE = od.PRODUCTCODE

WHERE vwp.BUYPRICE BETWEEN 30 AND 40

AND vwp.PRODUCTCODE LIKE 'S32%'

ORDER BY vwp.PRODUCTNAME ASC, vwp.BUYPRICE ASC; 

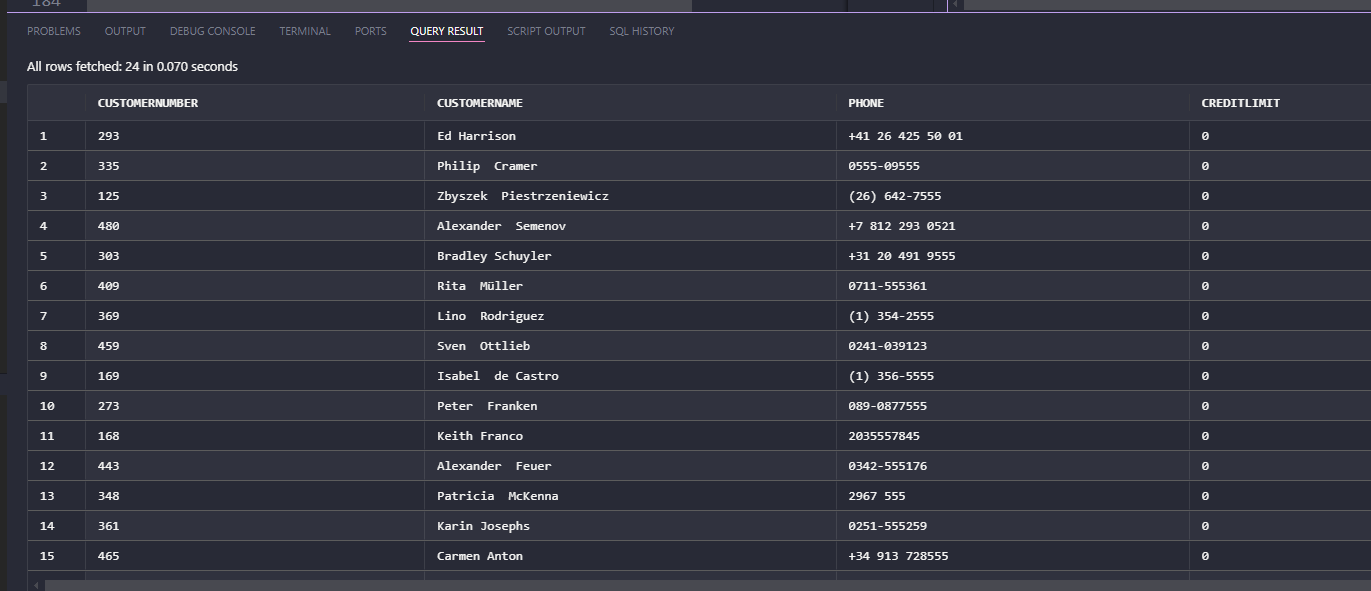
## Create a query that displays the customer order information with customer number, first name, last name, phone, and credit limits for all customers who do not have any orders.

SELECT  rc.CUSTOMERNUMBER, rc.CONTACTFIRSTNAME || ' ' || rc.CONTACTLASTNAME AS CUSTOMERNAME,

    rc.PHONE, rc.CREDITLIMIT

FROM RETAILCUSTOMERS rc

LEFT JOIN RETAILORDERS ro ON rc.CUSTOMERNUMBER = rO.CUSTOMERNUMBER

WHERE ro.ORDERNUMBER IS NULL; 

## Create a view (vwEmployeeManager) to display the information of all retail employees first name and last name and their managers first name and managers last name if there is any manager that the employee reports to. Include all employees, including those who do not report to anyone.

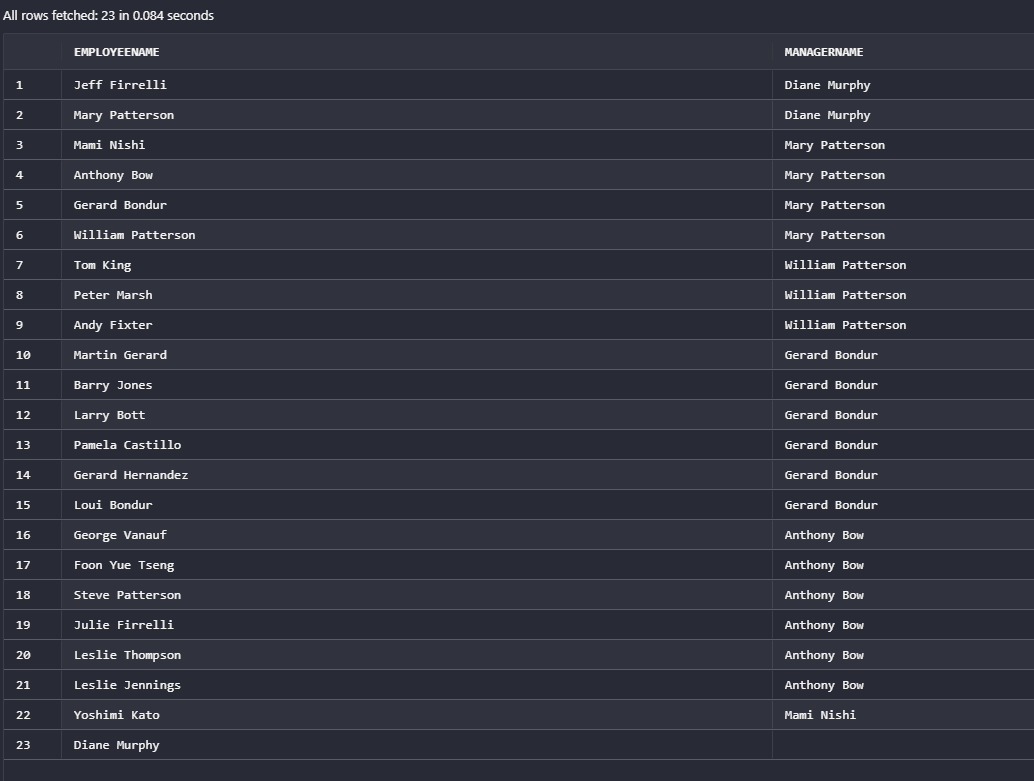
CREATE VIEW vwEmployeeManager AS

SELECT re.FIRSTNAME || ' ' || re.LASTNAME AS EMPLOYEENAME,

    rm.FIRSTNAME || ' ' || rm.LASTNAME AS MANAGERNAME

FROM RETAILEMPLOYEES re

LEFT JOIN RETAILEMPLOYEES rm ON re.REPORTSTO = rm.EMPLOYEENUMBER;

SELECT \* FROM vwEmployeeManager; 

## Modify the vwEmployeeManager view so the view returns only employee information for employees who have a manager. Do not DROP and recreate the view – modify it. (Google is your friend).

CREATE OR REPLACE VIEW vwEmployeeManager AS

SELECT re.FIRSTNAME || ' ' || re.LASTNAME AS EMPLOYEENAME,

    rm.FIRSTNAME || ' ' || rm.LASTNAME AS MANAGERNAME

FROM RETAILEMPLOYEES re

LEFT JOIN RETAILEMPLOYEES rm ON re.REPORTSTO = rm.EMPLOYEENUMBER

WHERE re.REPORTSTO IS NOT NULL;

SELECT \* FROM vwEmployeeManager; 