

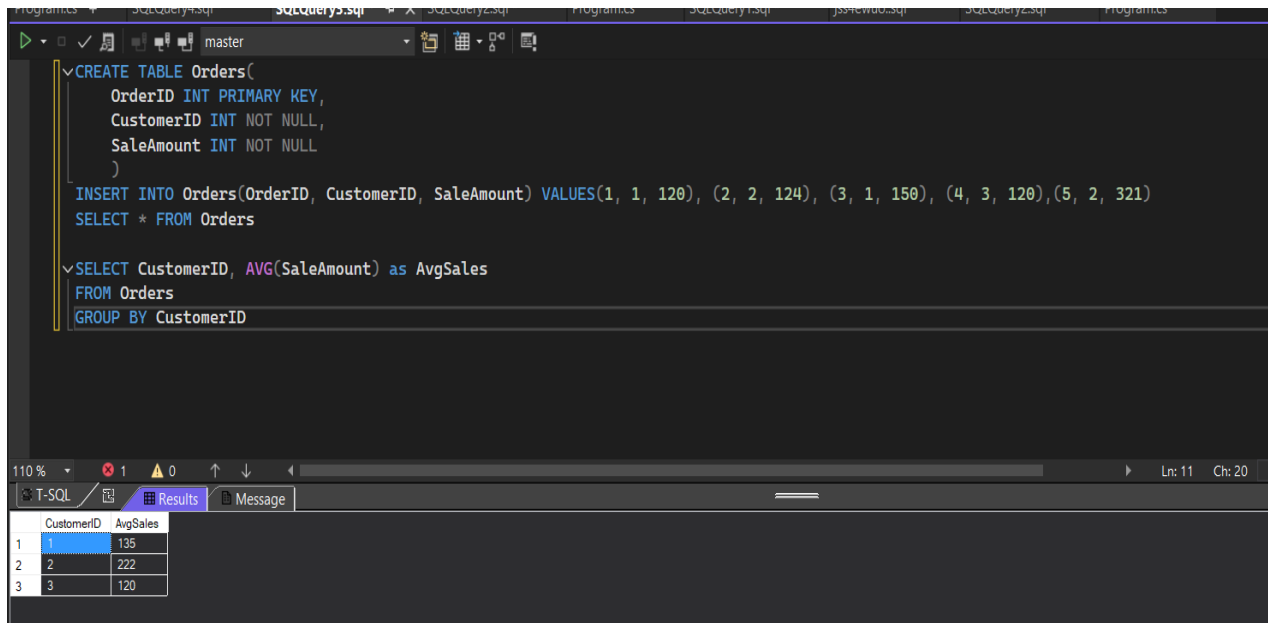
1. Find the Average Sales Per Customer

Problem:

Given an `Orders` table

with columns `OrderID`, `CustomerID`, and `SaleAmount`,

calculate the average sales amount per customer.



The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results pane. The query editor contains the following T-SQL code:

```
CREATE TABLE Orders(  
    OrderID INT PRIMARY KEY,  
    CustomerID INT NOT NULL,  
    SaleAmount INT NOT NULL  
)  
  
INSERT INTO Orders(OrderID, CustomerID, SaleAmount) VALUES(1, 1, 120), (2, 2, 124), (3, 1, 150), (4, 3, 120), (5, 2, 321)  
  
SELECT * FROM Orders  
  
SELECT CustomerID, AVG(SaleAmount) as AvgSales  
FROM Orders  
GROUP BY CustomerID
```

The results pane shows the output of the query, which is a table with two columns: CustomerID and AvgSales. The data is as follows:

CustomerID	AvgSales
1	135
2	222
3	120

2. Find Employees with No Manager

Problem:

Given an `Employees` table

with columns `EmployeeID`, `Name`, and `ManagerID`

(which refers to `EmployeeID` of the manager),

find all employees who do not have a manager.

Table Structure:

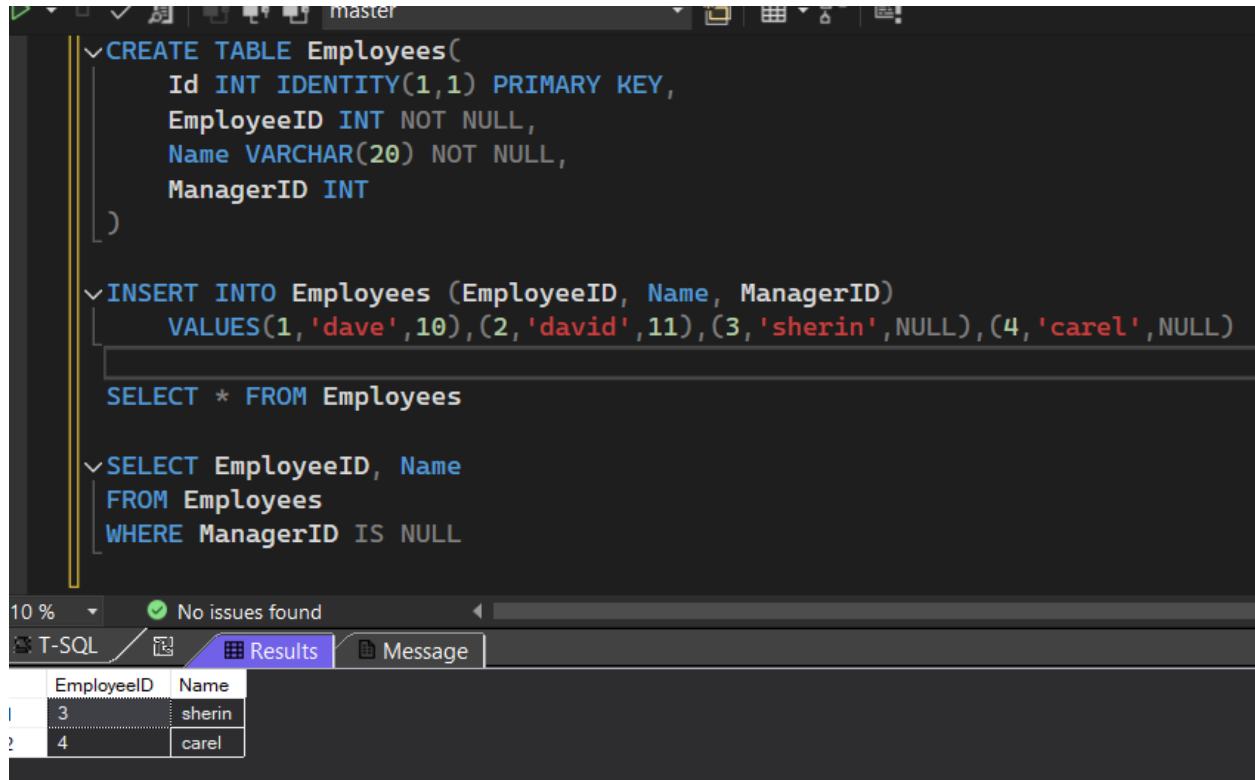
...

CREATE TABLE Employees (
EmployeeID INT,

Name VARCHAR(100),

ManagerID INT

);



The screenshot shows a SQL Server Enterprise Manager interface. The T-SQL tab is active, displaying the following code:

```
CREATE TABLE Employees(  
    Id INT IDENTITY(1,1) PRIMARY KEY,  
    EmployeeID INT NOT NULL,  
    Name VARCHAR(20) NOT NULL,  
    ManagerID INT  
)  
  
INSERT INTO Employees (EmployeeID, Name, ManagerID)  
VALUES(1, 'dave', 10), (2, 'david', 11), (3, 'sherin', NULL), (4, 'carel', NULL)  
  
SELECT * FROM Employees  
  
SELECT EmployeeID, Name  
FROM Employees  
WHERE ManagerID IS NULL
```

Below the code, the Results tab shows the output of the last query. It displays a table with two columns: EmployeeID and Name. The results are as follows:

EmployeeID	Name
3	sherin
4	carel

3. Find the Oldest and Youngest Employees

Problem: Given an `Employees` table

with columns `EmployeeID`, `Name`, and `DateOfBirth`,

find the oldest and youngest employees.

```
master
CREATE TABLE Employees(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    EmployeeID INT NOT NULL,
    Name VARCHAR(20) NOT NULL,
    DateOfBirth DATE
)
INSERT INTO Employees (EmployeeID, Name, DateOfBirth)
VALUES(1, 'dave', '2001-10-12'), (2, 'david', '2005-11-8'), (3, 'sherin', '1998-4-4'), (4, 'carel', '2000-3-7')
SELECT * FROM Employees
WHERE DateOfBirth IN ((SELECT MIN(DateOfBirth) FROM Employees), (SELECT MAX(DateOfBirth) FROM Employees))
```

110 % No issues found

	Id	EmployeeID	Name	DateOfBirth
1	5	2	david	2005-11-08
2	6	3	sherin	1998-04-04

4. Find the Most Recent Order for Each Customer

Problem: Given an `Orders` table

with columns `OrderID`, `CustomerID`, and `OrderDate`,

find the most recent order date for each customer.

Table Structure:

```

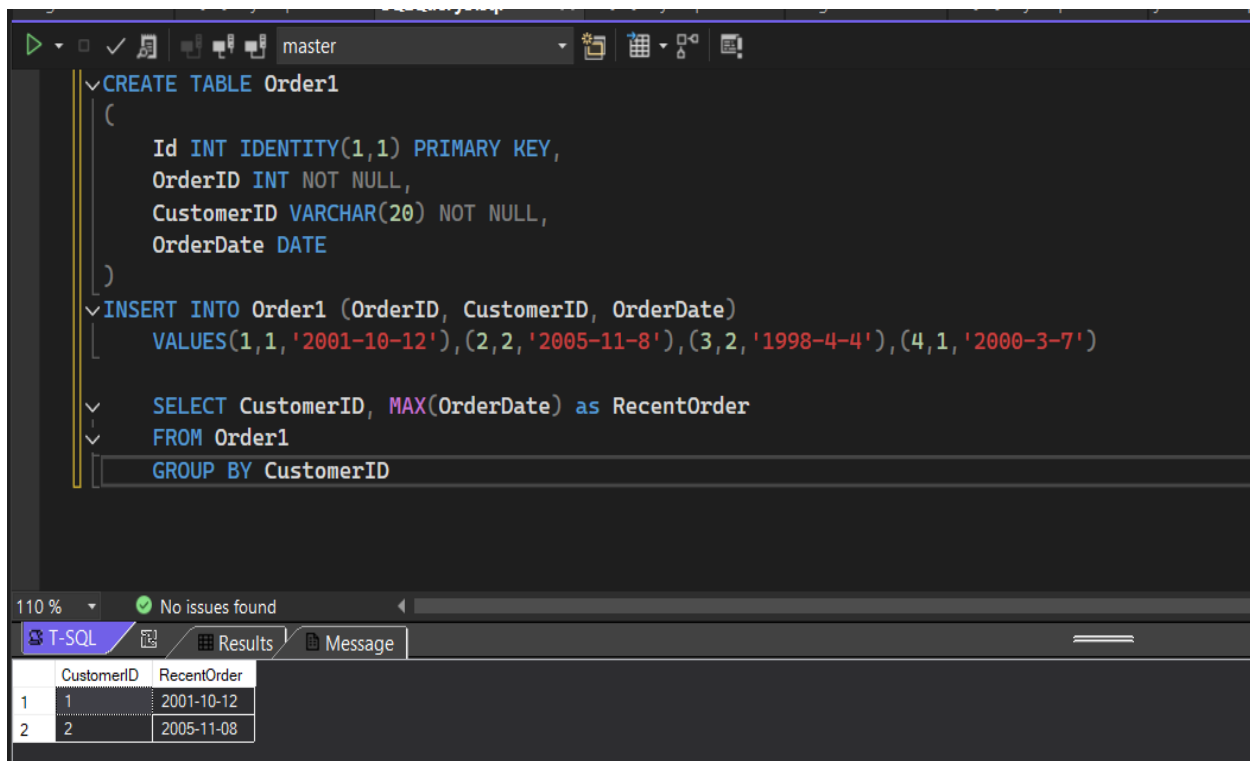
CREATE TABLE Orders (

OrderID INT,

CustomerID INT,

OrderDate DATE

);



The screenshot shows the SQL Server Enterprise Manager interface. The main pane displays T-SQL code for creating a table, inserting data, and querying it. The bottom pane shows the results of the query in a table format.

```
CREATE TABLE Order1
(
 Id INT IDENTITY(1,1) PRIMARY KEY,
 OrderID INT NOT NULL,
 CustomerID VARCHAR(20) NOT NULL,
 OrderDate DATE
)
INSERT INTO Order1 (OrderID, CustomerID, OrderDate)
VALUES(1,1, '2001-10-12'), (2,2, '2005-11-8'), (3,2, '1998-4-4'), (4,1, '2000-3-7')

SELECT CustomerID, MAX(OrderDate) as RecentOrder
FROM Order1
GROUP BY CustomerID
```

|   | CustomerID | RecentOrder |
|---|------------|-------------|
| 1 | 1          | 2001-10-12  |
| 2 | 2          | 2005-11-08  |

## 5. Find Employees Who Report to More Than One Manager

Problem: Given an `Employees` table

with columns `EmployeeID`, `Name`, and `ManagerID`,

find employees who report to more than one manager.

Student

```
CREATE TABLE Employee (
 id INT PRIMARY KEY IDENTITY,
 Name NVARCHAR(50) NOT NULL,
 ManagerID INT NOT NULL UNIQUE
)

INSERT INTO Employee (Name, ManagerID)
VALUES
('dave',101), ('anu',102), ('dave', 103), ('carel',104), ('Aish',105), ('carel', 106), ('carel',107);

SELECT * FROM Employee

SELECT Name, COUNT(ManagerID) as ManagerNumber
FROM Employee
GROUP BY Name
HAVING COUNT(ManagerID) >= 2
```

110 % No issues found

T-SQL Results Message

|   | Name  | ManagerNumber |
|---|-------|---------------|
| 1 | carel | 3             |
| 2 | dave  | 2             |