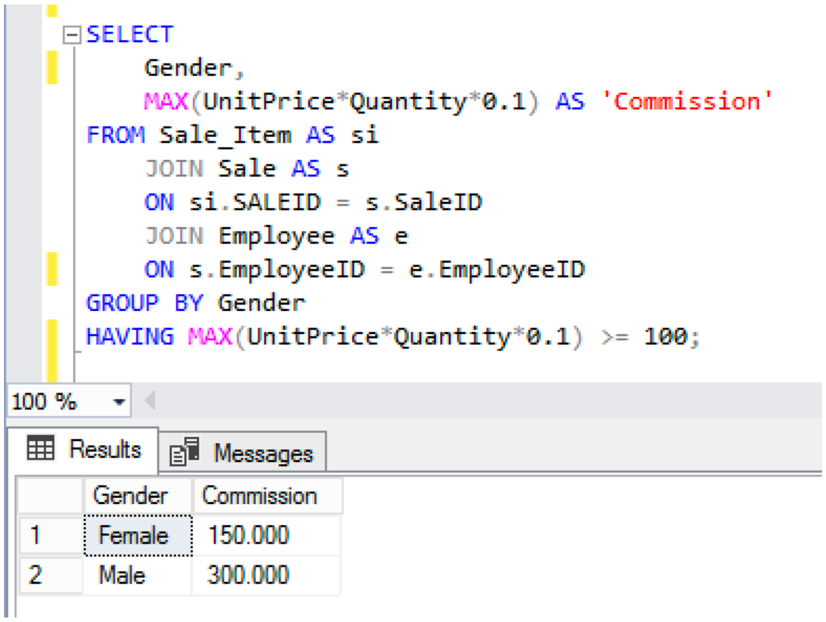
Lab 2 - Joins & subqueries (Chapter 3 & 4)

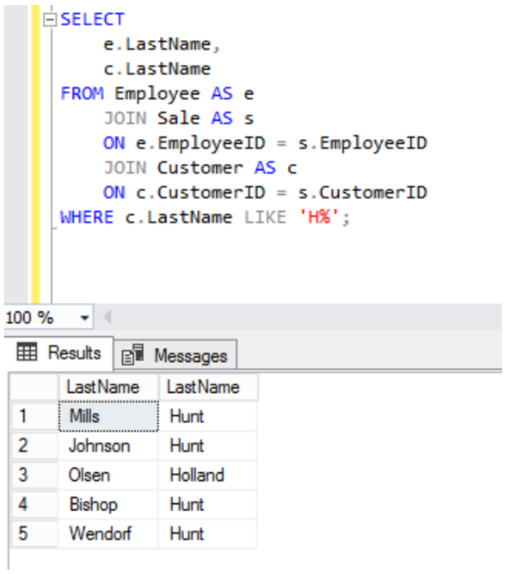
1) Write a query that shows the highest employee commission broken down by gender. Don't include groups with less than $100 max commission.



```
SELECT
    Gender,
    MAX(UnitPrice*Quantity*0.1) AS 'Commission'
FROM Sale_Item AS si
    JOIN Sale AS s
    ON si.SALEID = s.SaleID
    JOIN Employee AS e
    ON s.EmployeeID = e.EmployeeID
GROUP BY Gender
HAVING MAX(UnitPrice*Quantity*0.1) >= 100;
```

	Gender	Commission
1	Female	150.000
2	Male	300.000

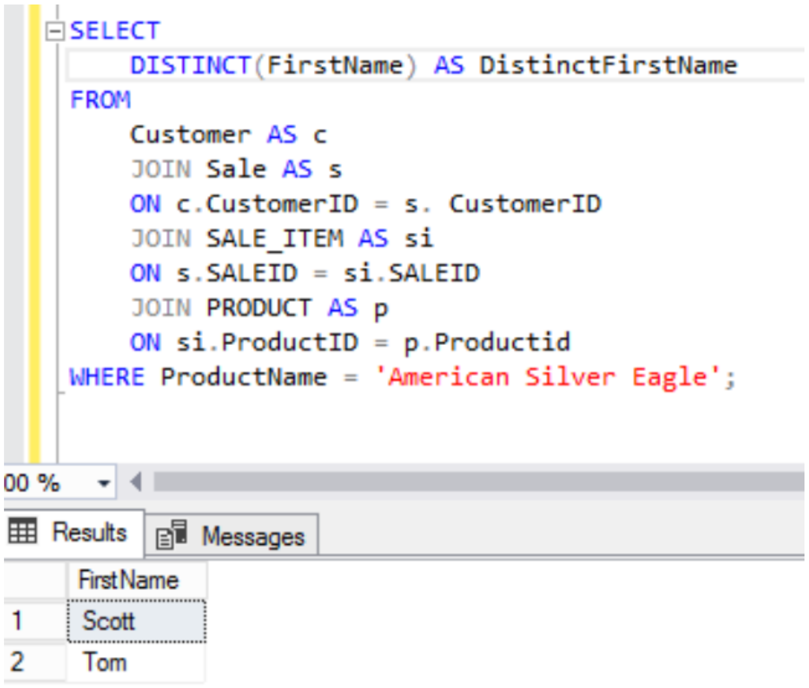
2. List the last name of both the last name of the customer and employee where an employee sold an item to a customer with a last name starting with the letter 'H'. Use an appropriate alias even though you are not including an aggregate function.



```
SELECT
    e.LastName,
    c.LastName
FROM Employee AS e
    JOIN Sale AS s
    ON e.EmployeeID = s.EmployeeID
    JOIN Customer AS c
    ON c.CustomerID = s.CustomerID
WHERE c.LastName LIKE 'H%';
```

	LastName	LastName
1	Mills	Hunt
2	Johnson	Hunt
3	Olsen	Holland
4	Bishop	Hunt
5	Wendorf	Hunt

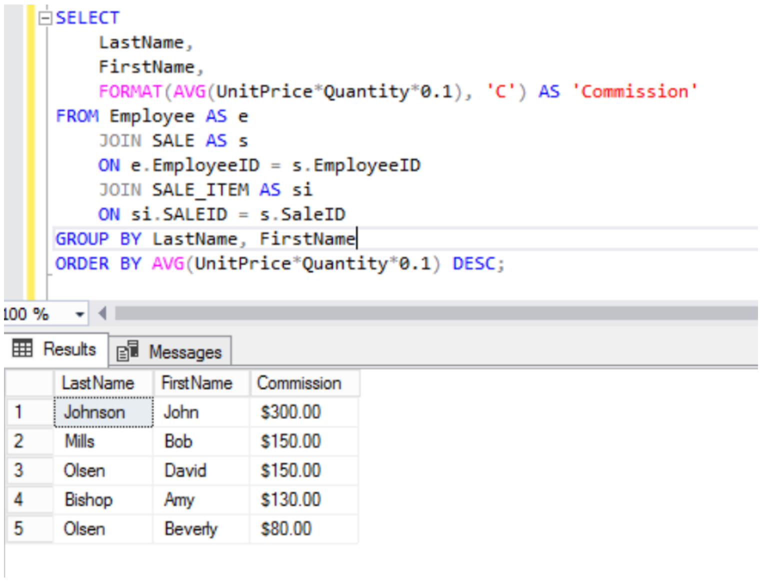
3. Write a query that shows the first name of the customer who bought a product with a product name of 'American Silver Eagle'. Do not include duplicate values.



```
SELECT
    DISTINCT(FirstName) AS DistinctFirstName
FROM
    Customer AS c
    JOIN Sale AS s
    ON c.CustomerID = s. CustomerID
    JOIN SALE_ITEM AS si
    ON s.SALEID = si.SALEID
    JOIN PRODUCT AS p
    ON si.ProductID = p.Productid
WHERE ProductName = 'American Silver Eagle';
```

	FirstName
1	Scott
2	Tom

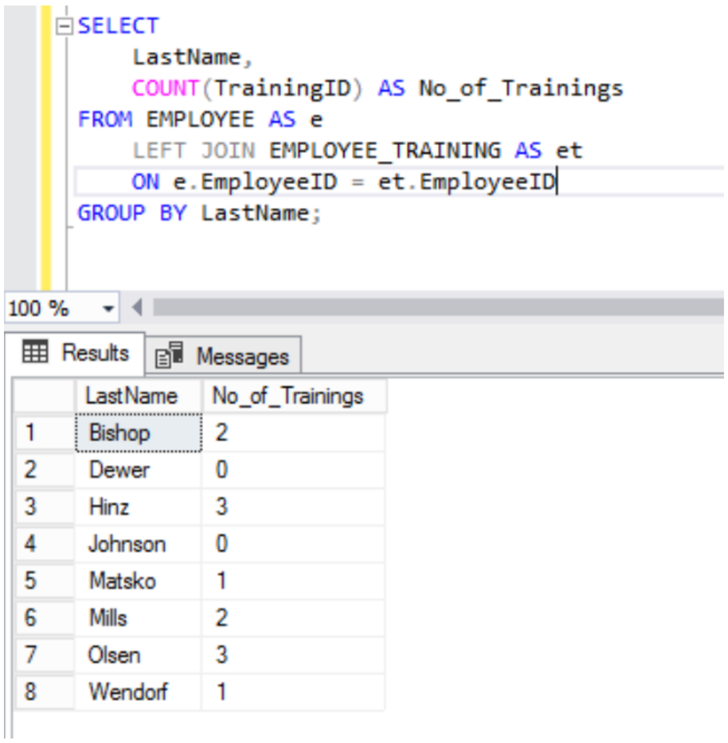
4. List the Last Name, First Name, and average commission of each employee. Sort from highest commission to lowest. Use an alias where appropriate. Make the commission output in currency format. If you encounter an error, it's probably related to the ORDER BY clause. Try doing an order by without using an alias. (Hint: This will require a 3 table join, Commission is calculated by (quantity*UnitPrice*0.1).



```
SELECT
    LastName,
    FirstName,
    FORMAT(AVG(UnitPrice*Quantity*0.1), 'C') AS 'Commission'
FROM Employee AS e
    JOIN SALE AS s
    ON e.EmployeeID = s.EmployeeID
    JOIN SALE_ITEM AS si
    ON si.SALEID = s.SaleID
GROUP BY LastName, FirstName
ORDER BY AVG(UnitPrice*Quantity*0.1) DESC;
```

	LastName	FirstName	Commission
1	Johnson	John	$300.00
2	Mills	Bob	$150.00
3	Olsen	David	$150.00
4	Bishop	Amy	$130.00
5	Olsen	Beverly	$80.00

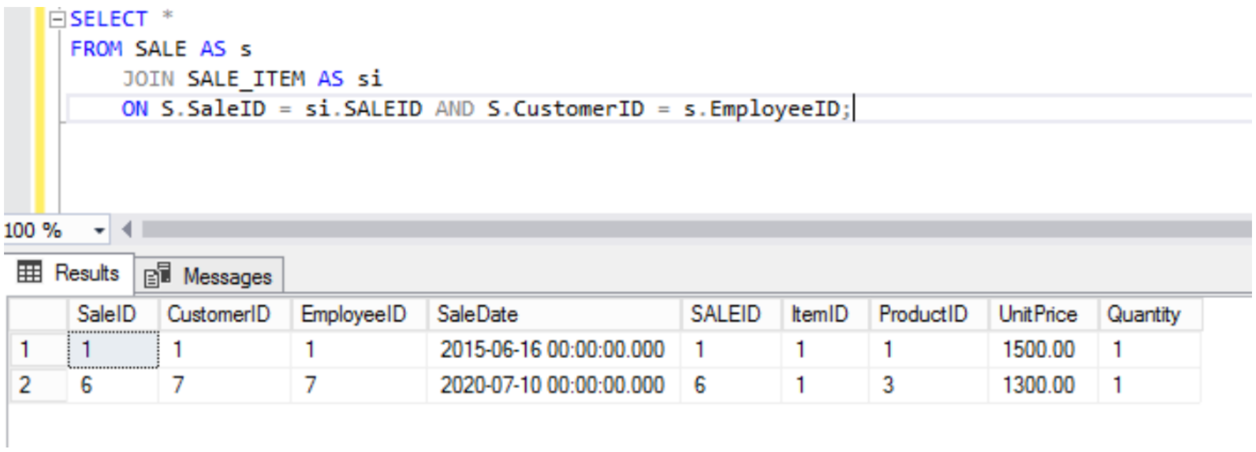
5. Write a count aggregate with a Left OUTER JOIN to show each employees' last name, and the number of times they have enrolled in a training course. Use the EMPLOYEE and EMPOLYEE_TRAINING Tables to code the solution.



```
SELECT
    LastName,
    COUNT(TrainingID) AS No_of_Trainings
FROM EMPLOYEE AS e
    LEFT JOIN EMPLOYEE_TRAINING AS et
    ON e.EmployeeID = et.EmployeeID
GROUP BY LastName;
```

	LastName	No_of_Trainings
1	Bishop	2
2	Dewer	0
3	Hinz	3
4	Johnson	0
5	Matsko	1
6	Mills	2
7	Olsen	3
8	Wendorf	1

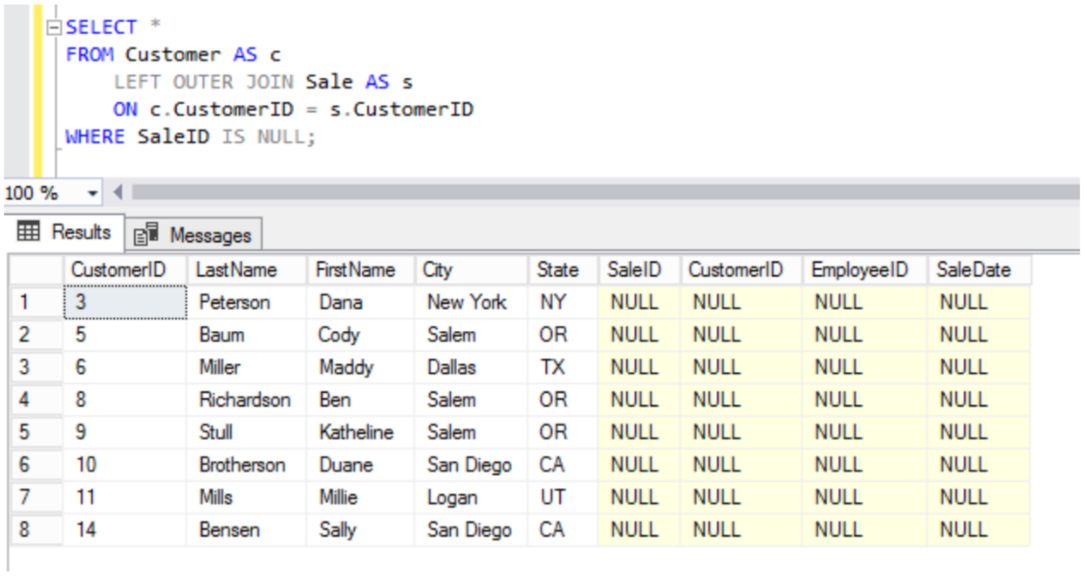
6. First, write an INNER JOIN that connects the SALE and SALE_Item tables. Then write an additional ON clause shows any sales where customer and employees IDs are the same to create a COMPOSITE JOIN.



```
SELECT *
FROM SALE AS s
    JOIN SALE_ITEM AS si
    ON S.SaleID = si.SALEID AND S.CustomerID = s.EmployeeID;
```

	SaleID	CustomerID	EmployeeID	SaleDate	SALEID	ItemID	ProductID	UnitPrice	Quantity
1	1	1	1	2015-06-16 00:00:00.000	1	1	1	1500.00	1
2	6	7	7	2020-07-10 00:00:00.000	6	1	3	1300.00	1

7. Return customers who have not yet made a purchase (Use the Customer and Sale Table) and use a LEFT OUTER JOIN.



```
SELECT *
FROM Customer AS c
    LEFT OUTER JOIN Sale AS s
    ON c.CustomerID = s.CustomerID
WHERE SaleID IS NULL;
```

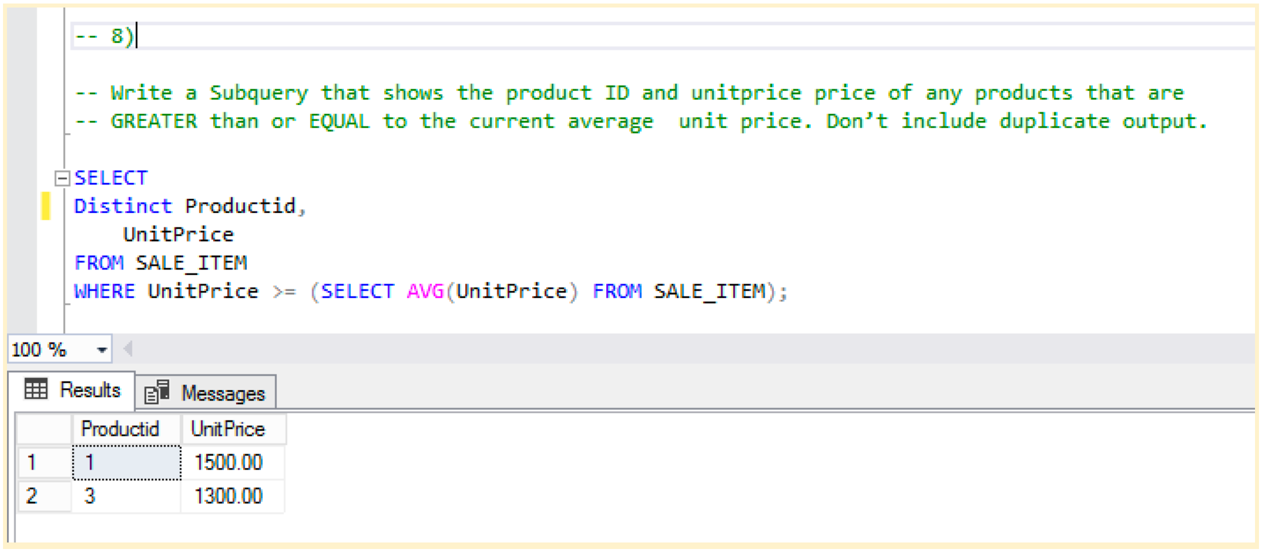
100 %

Results | Messages

	CustomerID	LastName	FirstName	City	State	SaleID	CustomerID	EmployeeID	SaleDate
1	3	Peterson	Dana	New York	NY	NULL	NULL	NULL	NULL
2	5	Baum	Cody	Salem	OR	NULL	NULL	NULL	NULL
3	6	Miller	Maddy	Dallas	TX	NULL	NULL	NULL	NULL
4	8	Richardson	Ben	Salem	OR	NULL	NULL	NULL	NULL
5	9	Stull	Katheline	Salem	OR	NULL	NULL	NULL	NULL
6	10	Brotherson	Duane	San Diego	CA	NULL	NULL	NULL	NULL
7	11	Mills	Millie	Logan	UT	NULL	NULL	NULL	NULL
8	14	Bensen	Sally	San Diego	CA	NULL	NULL	NULL	NULL

PART 2

8) Write a Subquery that shows the product ID and unit price of any products that are GREATER than or EQUAL to the current average unit price. Don't include duplicate output.



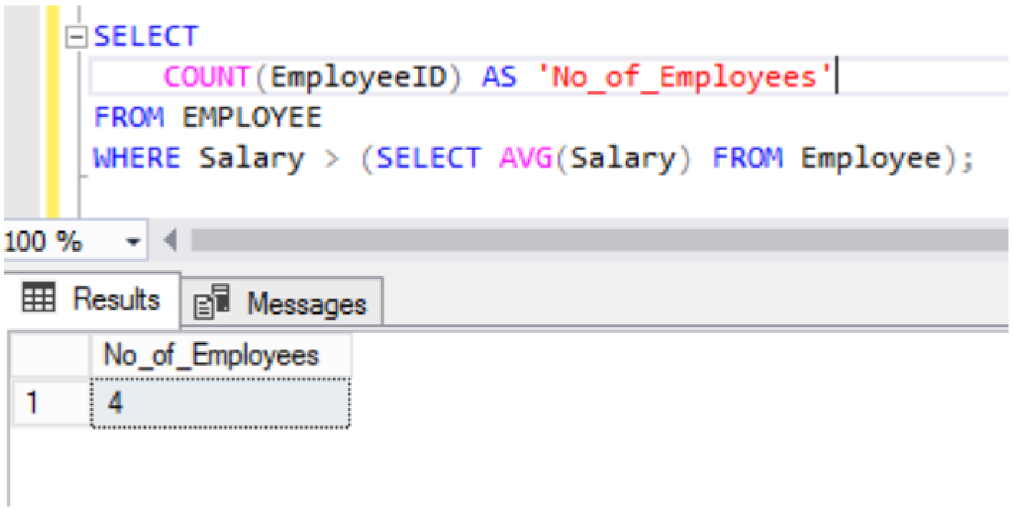
```
-- 8)

-- Write a Subquery that shows the product ID and unitprice price of any products that are
-- GREATER than or EQUAL to the current average  unit price. Don’t include duplicate output.

SELECT
Distinct Productid,
    UnitPrice
FROM SALE_ITEM
WHERE UnitPrice >= (SELECT AVG(UnitPrice) FROM SALE_ITEM);
```

100 %

Results | Messages

```
   Productid  UnitPrice
1  1          1500.00
2  3          1300.00
```

9) Write an SQL statement that counts the number of employees that make an above average salary.



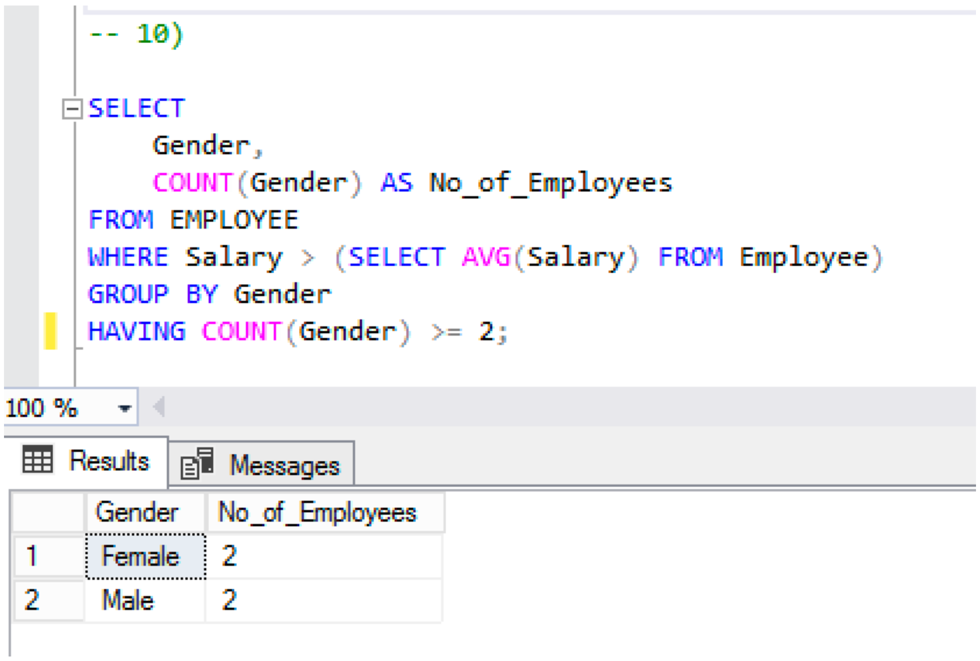
```
SELECT
    COUNT(EmployeeID) AS 'No_of_Employees'
FROM EMPLOYEE
WHERE Salary > (SELECT AVG(Salary) FROM Employee);
```

100 %

Results | Messages

```
   No_of_Employees
1  4
```

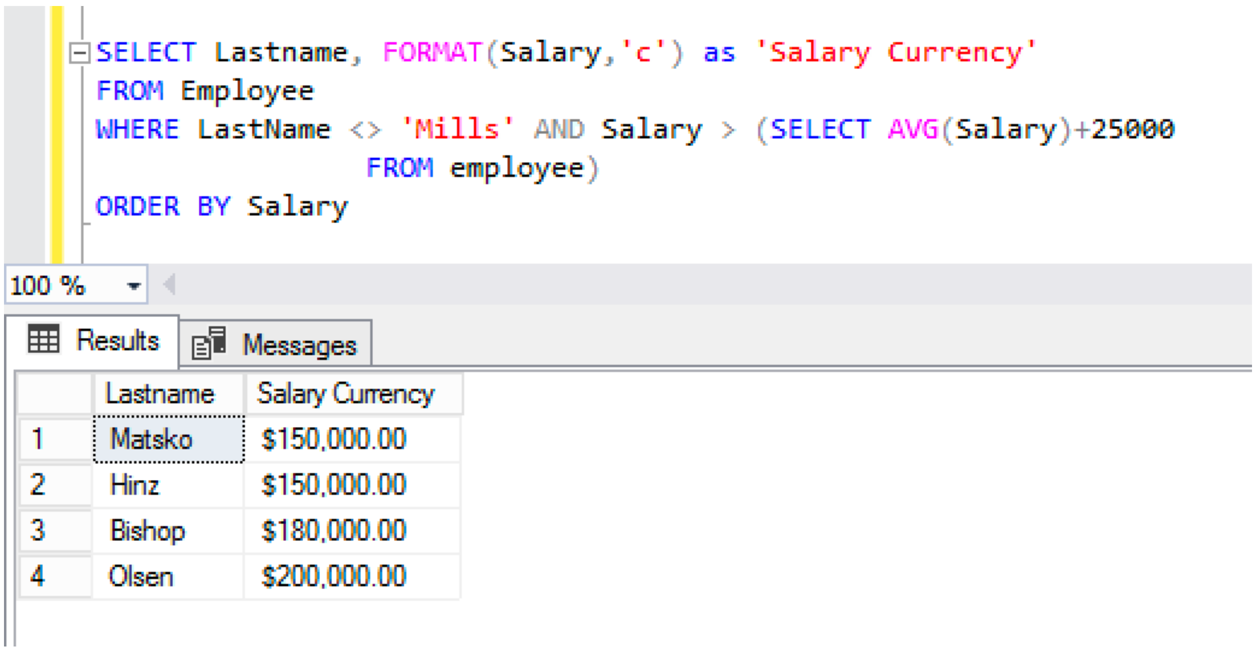
10) Write an SQL statement that displays gender, and separately counts and displays the number of male and female (use a GROUP BY) employees that make an above average salary. Don't include groups that are less than 2.



```
-- 10)

SELECT
    Gender,
    COUNT(Gender) AS No_of_Employees
FROM EMPLOYEE
WHERE Salary > (SELECT AVG(Salary) FROM Employee)
GROUP BY Gender
HAVING COUNT(Gender) >= 2;
```

	Gender	No_of_Employees
1	Female	2
2	Male	2

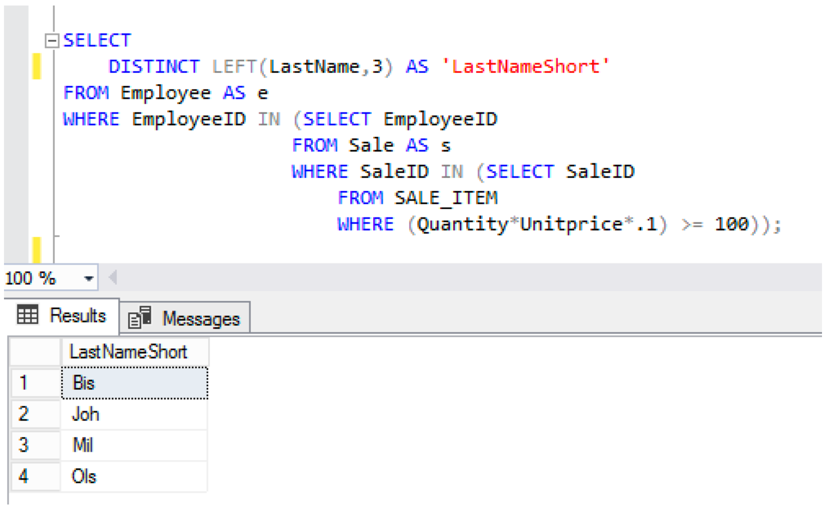
11) Write a Subquery that shows the last name and salary of employees that make over $25,000 the minimum salary of all employees. Don't include the employee with the last name 'Mills' in the output. Format the Salary output to currency (i.e., $100,000.00) and order output from smallest to largest. If you encounter an error, it's probably related to the ORDER BY clause. Try doing an order by without using an alias.



```sql
SELECT Lastname, FORMAT(Salary,'c') as 'Salary Currency'
FROM Employee
WHERE LastName <> 'Mills' AND Salary > (SELECT AVG(Salary)+25000
                FROM employee)
ORDER BY Salary
```

	Lastname	Salary Currency
1	Matsko	$150,000.00
2	Hinz	$150,000.00
3	Bishop	$180,000.00
4	Olsen	$200,000.00

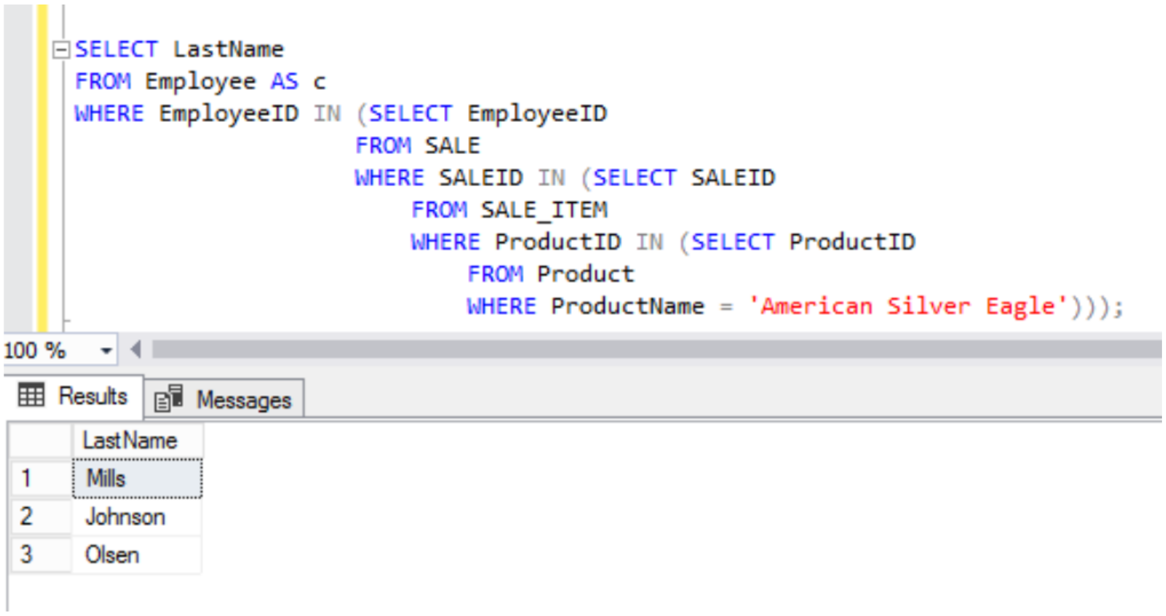
12) Write a Subquery that shows the first three letters of the employees' last name that earned a commission over 100. (Hint: (Quantity*Unitprice*.1)>=100).



```sql
SELECT
    DISTINCT LEFT(LastName,3) AS 'LastNameShort'
FROM Employee AS e
WHERE EmployeeID IN (SELECT EmployeeID
                    FROM Sale AS s
                    WHERE SaleID IN (SELECT SaleID
                        FROM SALE_ITEM
                        WHERE (Quantity*Unitprice*.1) >= 100));
```

	LastNameShort
1	Bis
2	Joh
3	Mil
4	Ols

13) Write a Subquery that lists the Employees Last Name that have sold a product with the name "American Silver Eagle". Question 7 above may provide some help with this question.



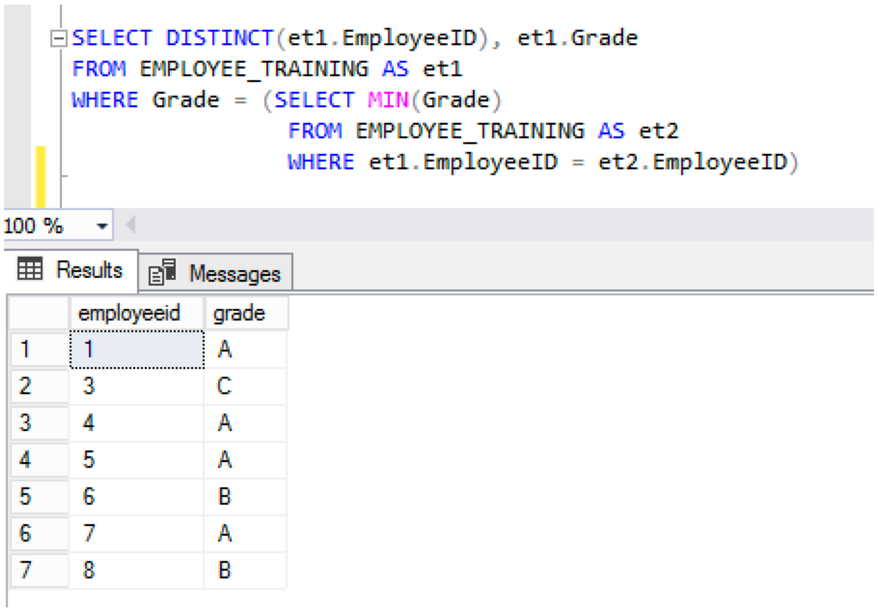
```
SELECT LastName
FROM Employee AS c
WHERE EmployeeID IN (SELECT EmployeeID
                    FROM SALE
                    WHERE SALEID IN (SELECT SALEID
                        FROM SALE_ITEM
                        WHERE ProductID IN (SELECT ProductID
                            FROM Product
                            WHERE ProductName = 'American Silver Eagle')));
```

100 %

Results | Messages

	LastName
1	Mills
2	Johnson
3	Olsen

14) Write a Correlated subquery that shows the best grade earned by each student. Use the EMPLOYEE_Training table to --code this problem. Use DISTINCT if needed. Output should include the employeeID and highest grade. Also, would you use MIN(GRADE) or MAX(GRADE)? *Note, grades for this training only include A, B, C, D, and F.



```
SELECT DISTINCT(et1.EmployeeID), et1.Grade
FROM EMPLOYEE_TRAINING AS et1
WHERE Grade = (SELECT MIN(Grade)
               FROM EMPLOYEE_TRAINING AS et2
               WHERE et1.EmployeeID = et2.EmployeeID)
```

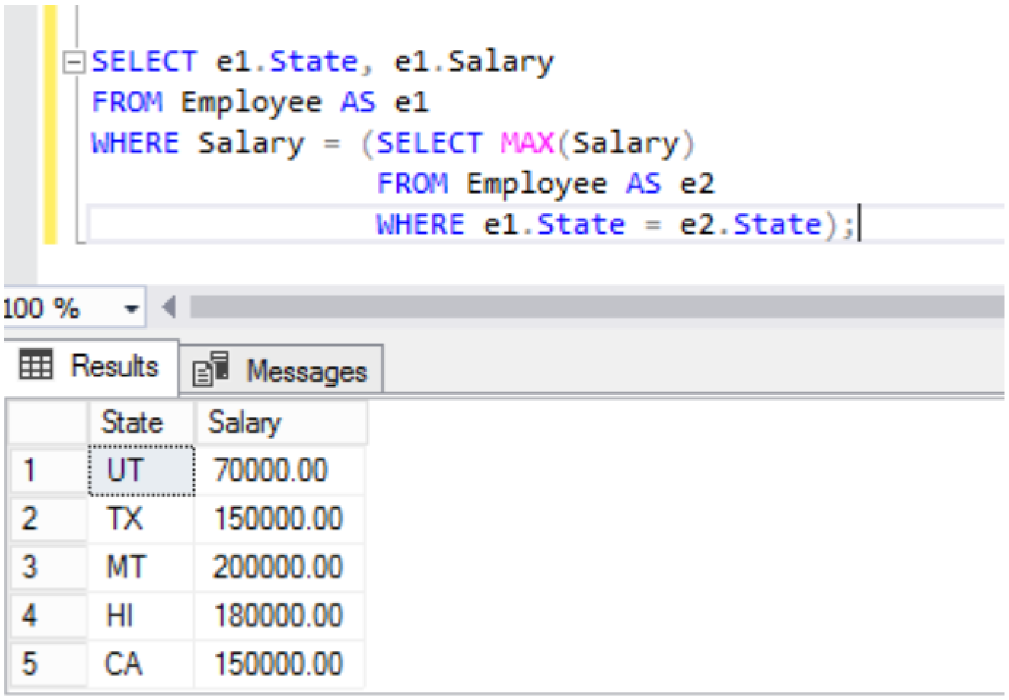
100 %

Results | Messages

	employeeid	grade
1	1	A
2	3	C
3	4	A
4	5	A
5	6	B
6	7	A
7	8	B

We will use Min(grade) because that's the highest grade that each person has received in a class.

15) **Write a correlated subquery that shows the highest salary by state. Use the EMPLOYEE table to code the problem.** *Your final WHERE clause in the correlated subquery DOESNT use Employeeid at all in the problem… so don't write… (WHERE e1.EmployeeID = e2.EmployeeID).



```
SELECT e1.State, e1.Salary
FROM Employee AS e1
WHERE Salary = (SELECT MAX(Salary)
                FROM Employee AS e2
                WHERE e1.State = e2.State);
```

100 %

Results | Messages

	State	Salary
1	UT	70000.00
2	TX	150000.00
3	MT	200000.00
4	HI	180000.00
5	CA	150000.00