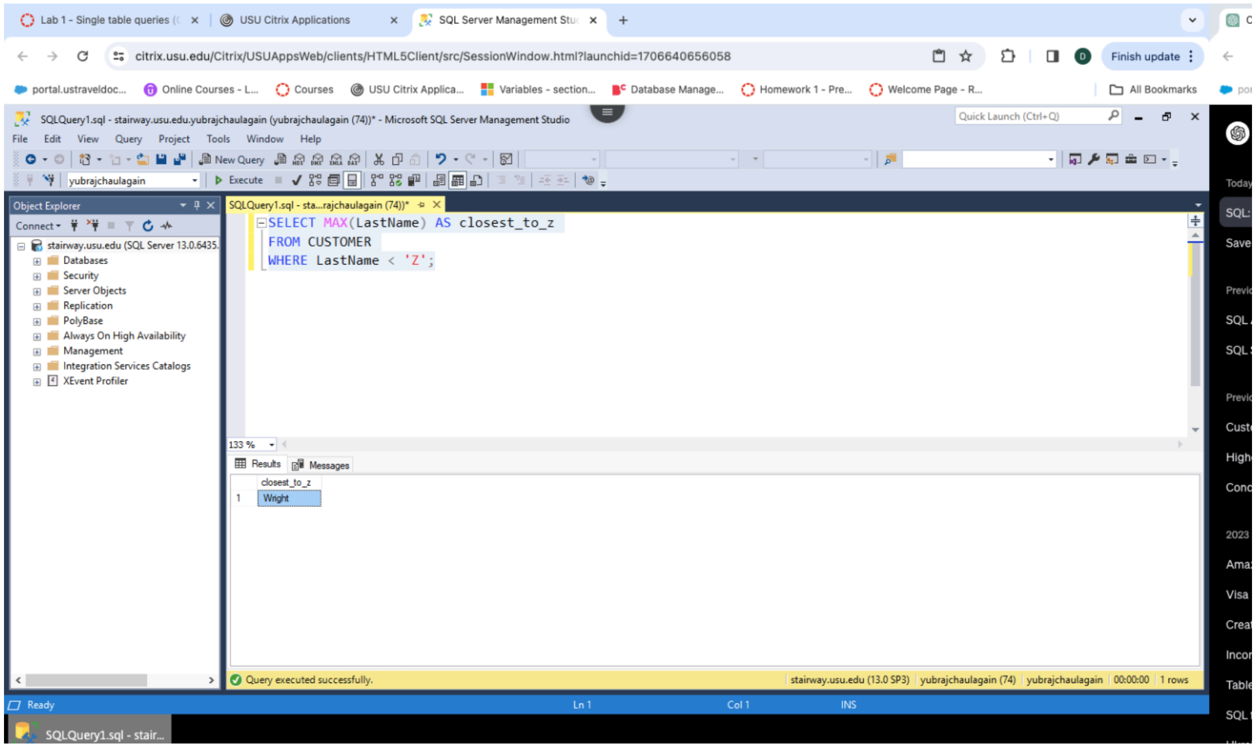
Part 1

1. Using the MAX Function, write an SQL statement that shows the customer's last name closest to Z. Include an Aliases when appropriate.

SELECT MAX(LastName) AS ClosestToZ FROM CUSTOMER WHERE   LastName < 'Z' ORDER BY MAX(LastName) DESC.



```
SELECT MAX(LastName) AS closest_to_z
FROM CUSTOMER
WHERE LastName < 'Z';
```

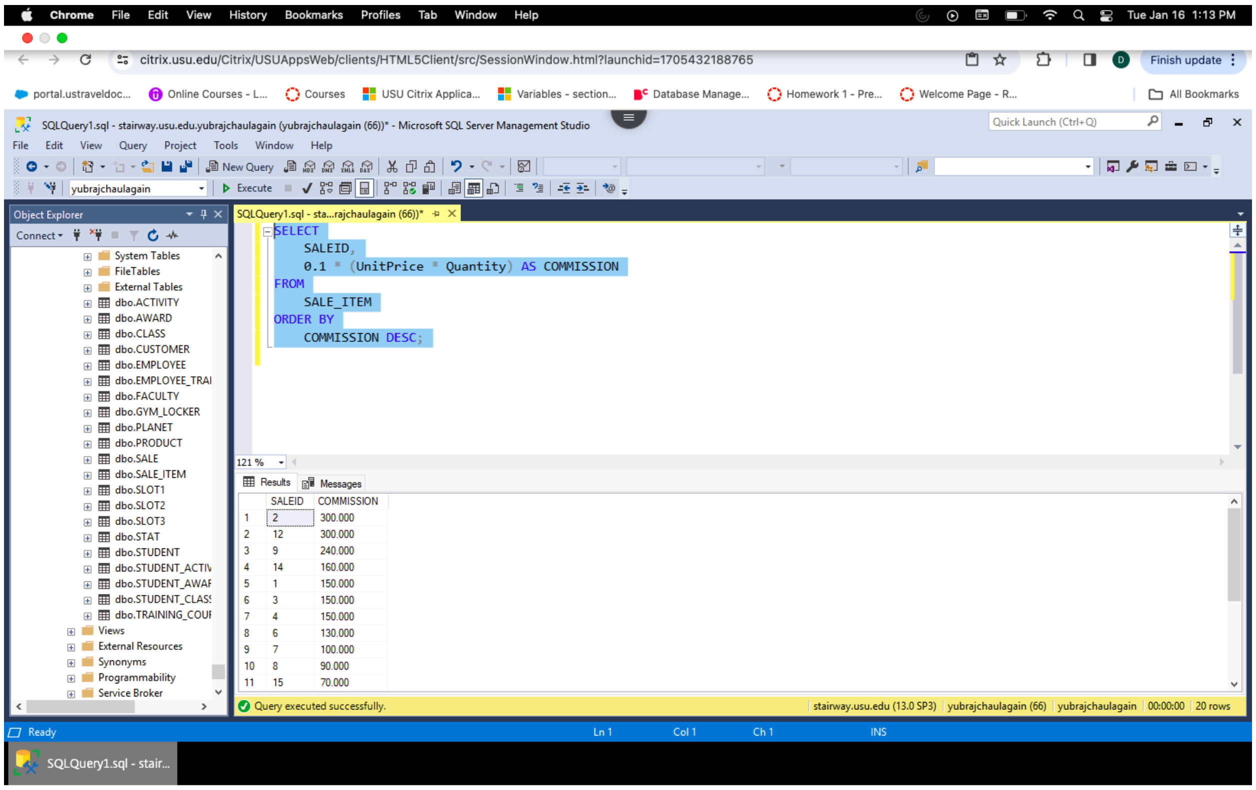
```
   closest_to_z
1  Wright
```

```
Query executed successfully.    stairway.usu.edu (13.0 SP3) | yubrajchaulagain (74) | yubrajchaulagain | 00:00:00 | 1 rows
```

2. Using the SALE_ITEM Table, write an SQL statement that shows the SALEID and COMMISSION for each sale. Commission is a calculated field based on 10% of the (UnitPrice * Quantity). Order the results from highest to lowest commission and include an alias. (If your SALE_ITEM table does not have a quantity row then you need to

update your tables by going into the canvas files under Antique dataset, drop the tables and then add the new ones back!!)

SELECT SALEID, 0.1 * (UnitPrice * Quantity) AS COMMISSION FROM SALE_ITEM ORDER BY COMMISSION DESC;



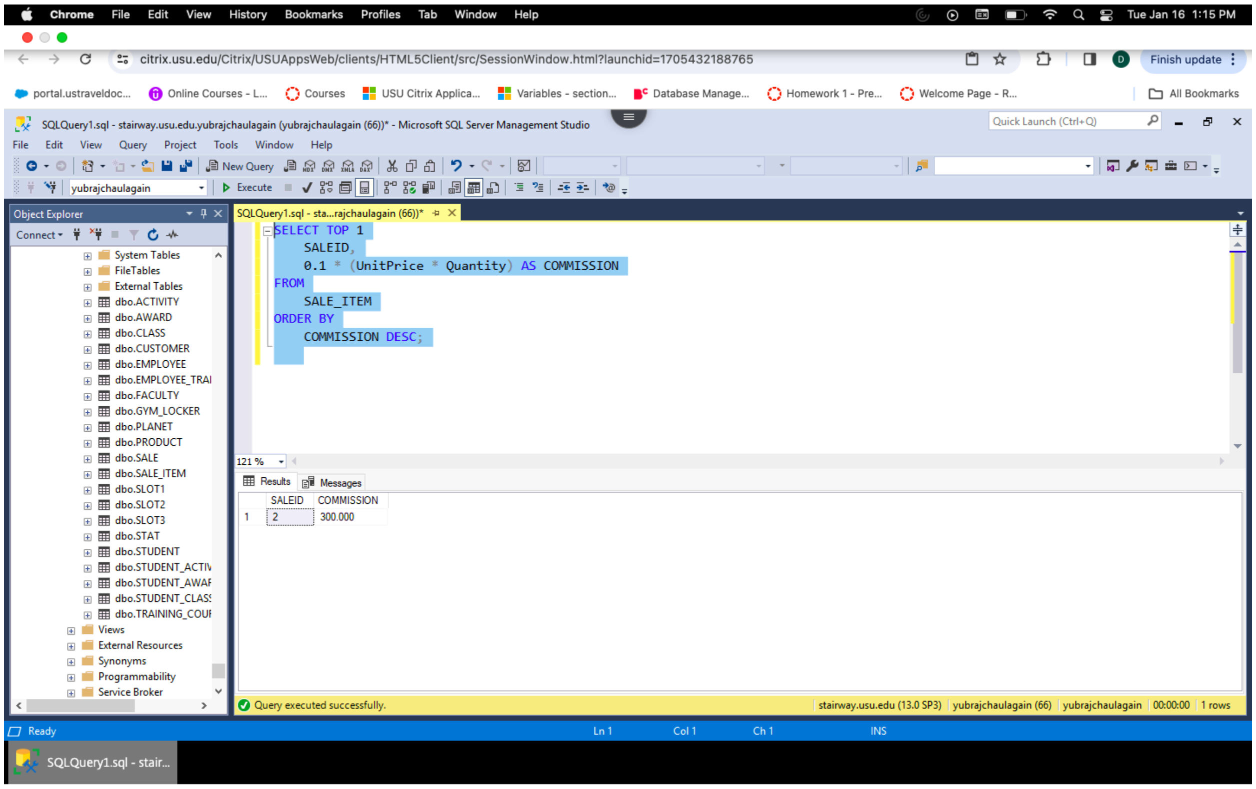
```
SELECT
    SALEID,
    0.1 * (UnitPrice * Quantity) AS COMMISSION
FROM
    SALE_ITEM
ORDER BY
    COMMISSION DESC;
```

```
   SALEID  COMMISSION
1  2       300.000
2  12      300.000
3  9       240.000
4  14      160.000
5  1       150.000
6  3       150.000
7  4       150.000
8  6       130.000
9  7       100.000
10 8       90.000
11 15      70.000
```

Query executed successfully.   stairway.usu.edu (13.0 SP3)   yubrajchaulagain (66)   yubrajchaulagain   00:00:00   20 rows

3.Modify the prior question to only show the highest SaleID and Commission using the TOP Function.

SELECT TOP 1 SALEID, 0.1 * (UnitPrice * Quantity) AS COMMISSION FROM SALE_ITEM ORDER BY COMMISSION



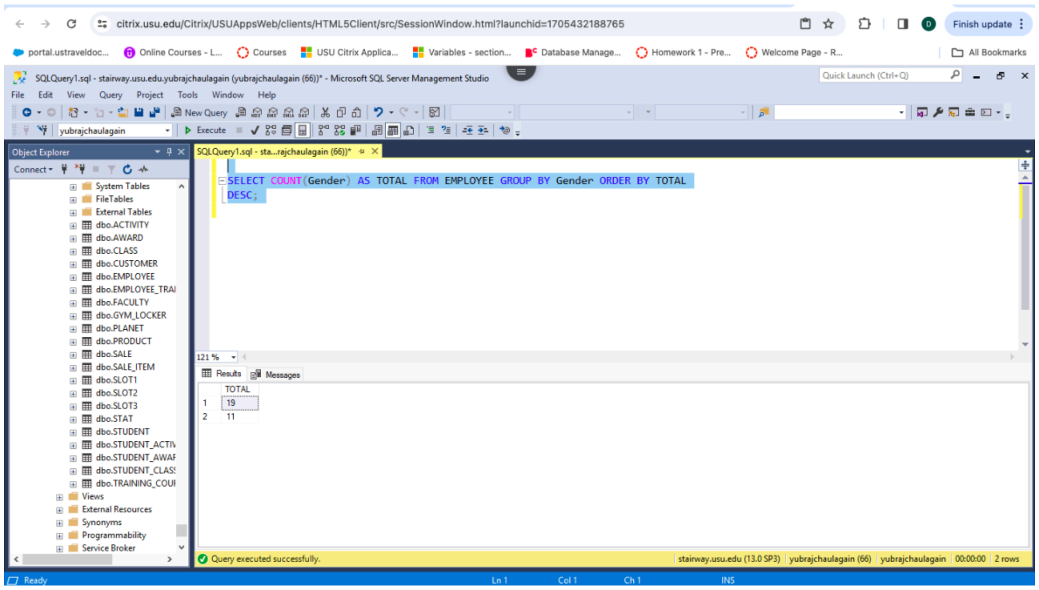
```
SELECT TOP 1
    SALEID,
    0.1 * (UnitPrice * Quantity) AS COMMISSION
FROM
    SALE_ITEM
ORDER BY
    COMMISSION DESC;
```

```
   SALEID   COMMISSION
1  2        300.000
```

Query executed successfully.

4. Write an SQL statement that shows Gender, and the total number of men and women that work for DPC Antiques. Order the output by the total from highest to lowest. Include an Aliases when appropriate.

SELECT COUNT(Gender) AS TOTAL FROM EMPLOYEE GROUP BY Gender ORDER BY TOTAL DESC;



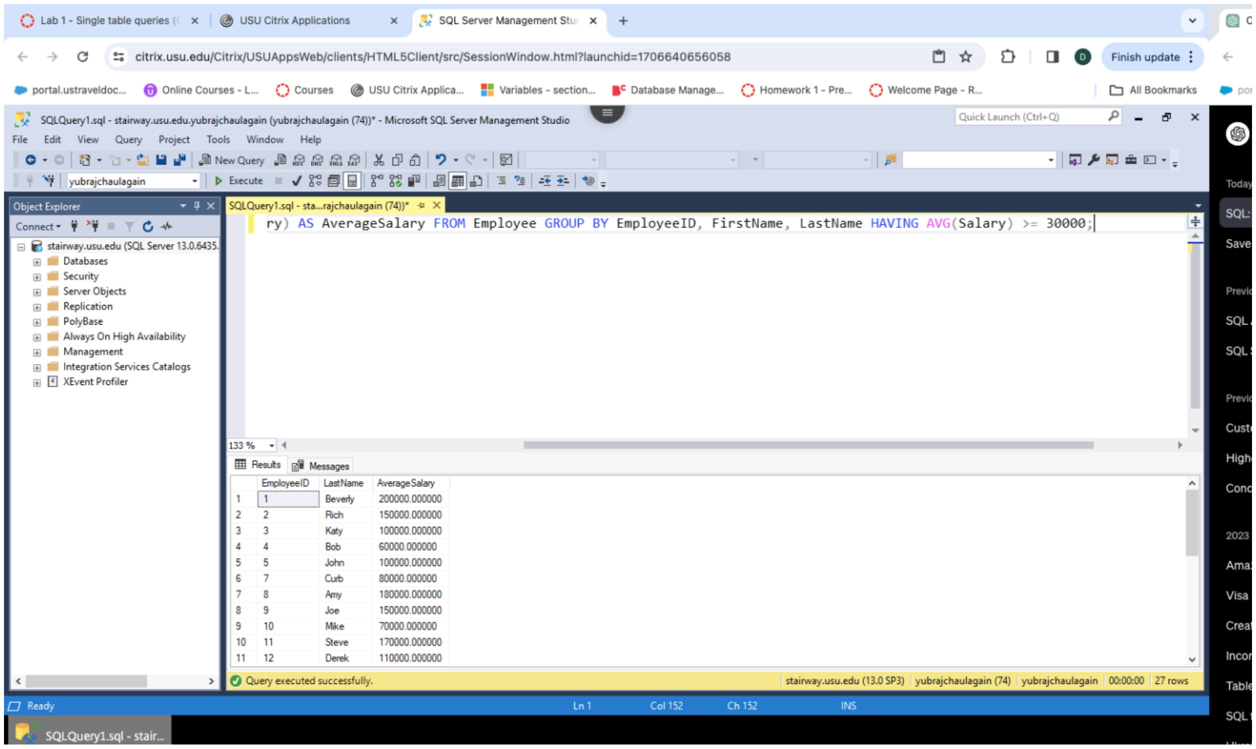
```
SELECT COUNT(Gender) AS TOTAL FROM EMPLOYEE GROUP BY Gender ORDER BY TOTAL
DESC;
```

```
   TOTAL
1  19
2  11
```

Query executed successfully.

5. Write an SQL statement that shows the average salary of each employee. Don't include salaries below 30000 in the calculation. Include an Aliases where appropriate. *Describe why this question doesn't need to be answered using the AVG(Salary) aggregate function.

SELECT EmployeeID, FirstName LastName, AVG(Salary) AS AverageSalary FROM Employee GROUP BY EmployeeID, FirstName, LastName HAVING AVG(Salary) >= 30000;



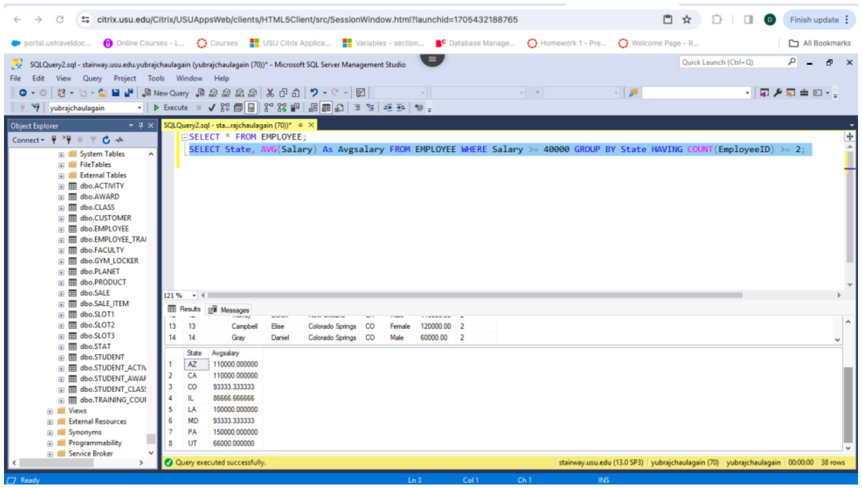
```
ry) AS AverageSalary FROM Employee GROUP BY EmployeeID, FirstName, LastName HAVING AVG(Salary) >= 30000;
```

```
    EmployeeID  LastName  AverageSalary
1   1           Beverly   200000.000000
2   2           Rich      150000.000000
3   3           Katy      100000.000000
4   4           Bob       60000.000000
5   5           John      100000.000000
6   7           Curb      80000.000000
7   8           Amy       180000.000000
8   9           Joe       150000.000000
9   10          Mike      70000.000000
10  11          Steve     170000.000000
11  12          Derek     110000.000000
```

Query executed successfully.

6. Write an SQL statement that shows the state and average salary of each state. Don't include salaries below 40000 in the calculation. Include an Aliases when appropriate. In addition, group the averages by State. Don't include groups with less than 2 employees.

SELECT State, AVG(Salary) As Avgsalary FROM EMPLOYEE WHERE Salary >= 40000 GROUP BY State HAVING COUNT(EmployeeID) >= 2;



```
SELECT * FROM EMPLOYEE;
SELECT State, AVG(Salary) As Avgsalary FROM EMPLOYEE WHERE Salary >= 40000 GROUP BY State HAVING COUNT(EmployeeID) >= 2;
```

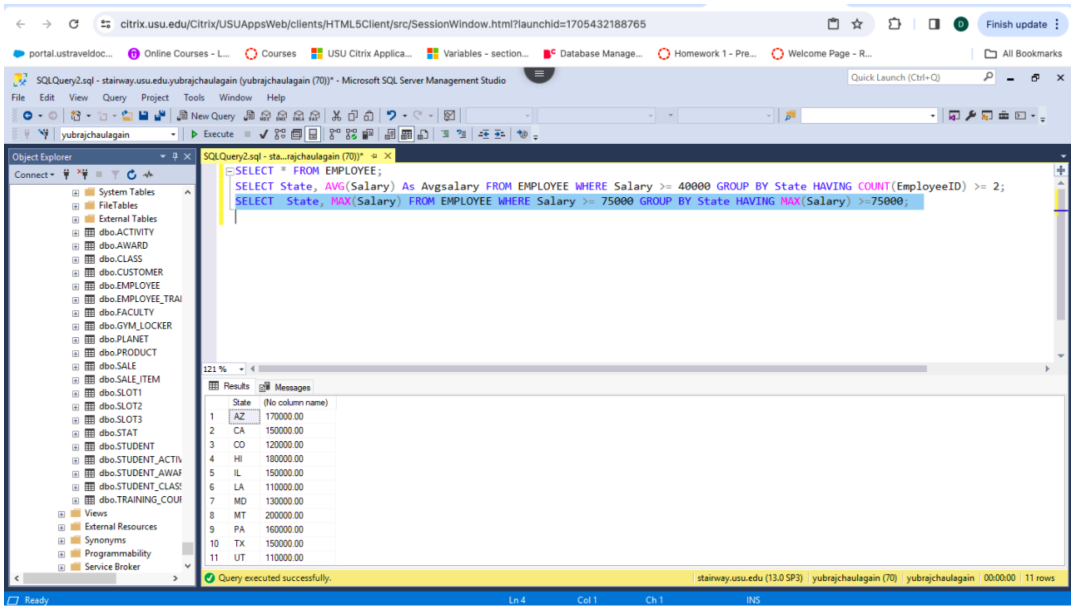
```
13  13   Campbell  Elise   Colorado Springs  CO  Female  120000.00  2
14  14   Gray      Daniel  Colorado Springs  CO  Male    60000.00   2
```

```
   State  Avgsalary
1  AZ     110000.000000
2  CA     110000.000000
3  CO     93333.333333
4  IL     86666.666666
5  LA     100000.000000
6  MD     93333.333333
7  PA     150000.000000
8  UT     66000.000000
```

Query executed successfully. stairway.usu.edu (13.0 SP3) | yubrajchaulagain (70) | yubrajchaulagain | 00:00:00 | 38 rows

7. Write an SQL statement that shows the Employees' maximum salary offered in each state. Don't include states (groups) with a maximum salary less than $75,000.

SELECT State, MAX(Salary) FROM EMPLOYEE WHERE Salary >= 75000 GROUP BY State HAVING MAX(Salary) >=75000;



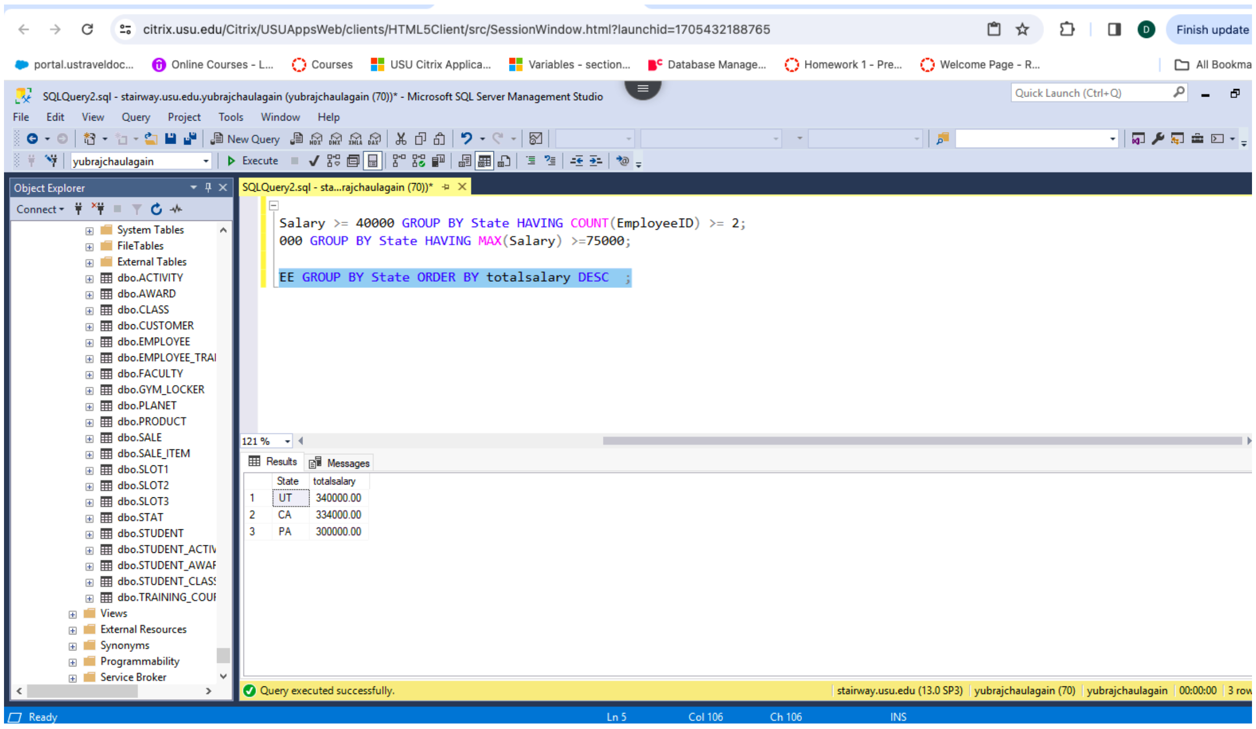
```
SELECT * FROM EMPLOYEE;
SELECT State, AVG(Salary) As Avgsalary FROM EMPLOYEE WHERE Salary >= 40000 GROUP BY State HAVING COUNT(EmployeeID) >= 2;
SELECT  State, MAX(Salary) FROM EMPLOYEE WHERE Salary >= 75000 GROUP BY State HAVING MAX(Salary) >=75000;
```

```
    State  (No column name)
1   AZ     170000.00
2   CA     150000.00
3   CO     120000.00
4   HI     180000.00
5   IL     150000.00
6   LA     110000.00
7   MD     130000.00
8   MT     200000.00
9   PA     160000.00
10  TX     150000.00
11  UT     110000.00
```

Query executed successfully. stairway.usu.edu (13.0 SP3) | yubrajchaulagain (70) | yubrajchaulagain | 00:00:00 | 11 rows

8. Using a GROUP BY Clause, write an SQL statement that shows the TOP 3 hometowns of Employees based on total salaries being paid. Use the TOP and ORDER BY Clause to solve.

SELECT TOP 3 State, SUM(Salary) as totalsalary FROM EMPLOYEE GROUP BY State ORDER BY totalsalary DESC;



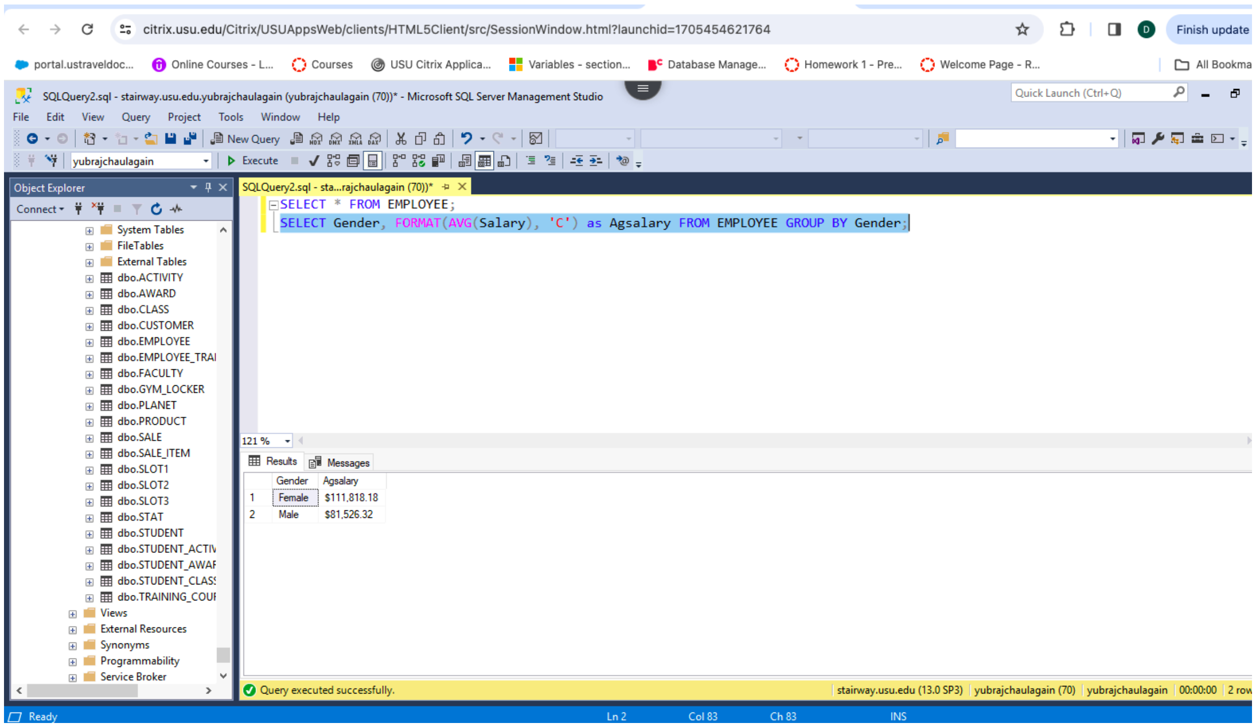
```
Salary >= 40000 GROUP BY State HAVING COUNT(EmployeeID) >= 2;
000 GROUP BY State HAVING MAX(Salary) >=75000;

EE GROUP BY State ORDER BY totalsalary DESC  ;
```

	State	totalsalary
1	UT	340000.00
2	CA	334000.00
3	PA	300000.00

Query executed successfully.

9. Write an SQL statement that illustrates whether men or women employees have higher average salaries. Format the average salary column to ‘currency’ (i.e, $34.55).

SELECT Gender, FORMAT(AVG(Salary), 'C') as Agsalary FROM EMPLOYEE GROUP BY Gender;



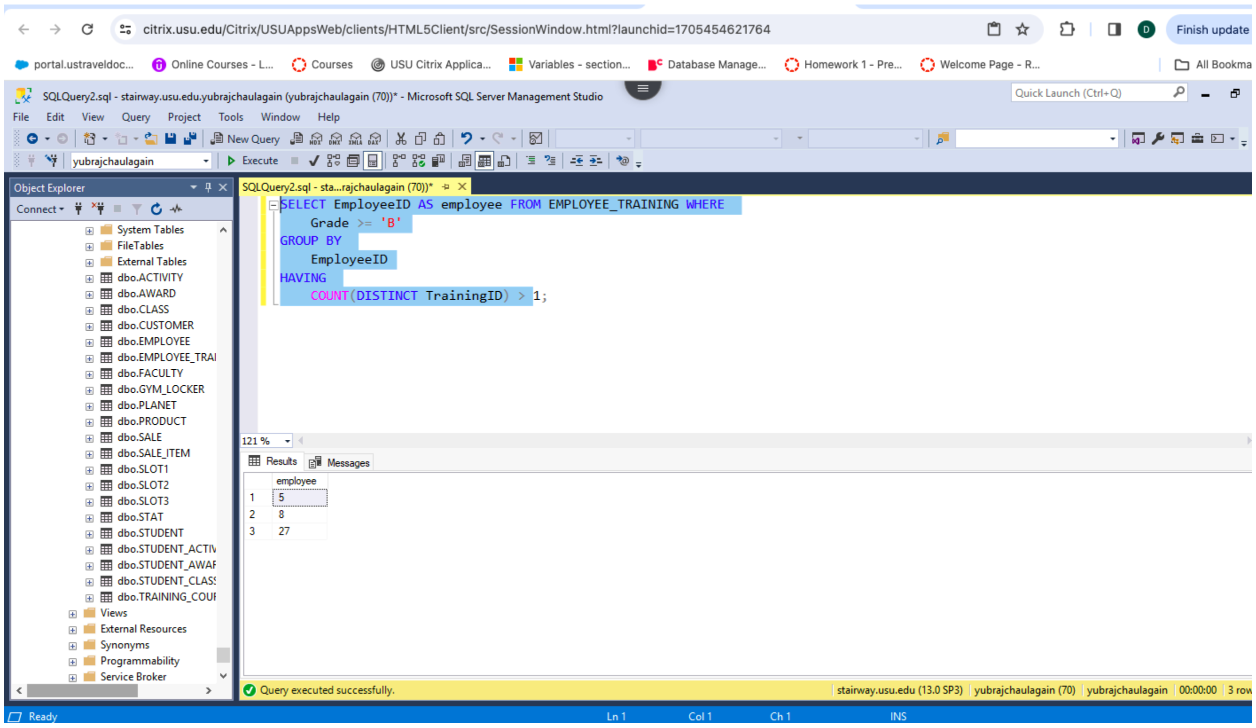
```
SELECT * FROM EMPLOYEE;
SELECT Gender, FORMAT(AVG(Salary), 'C') as Agsalary FROM EMPLOYEE GROUP BY Gender;
```

	Gender	Agsalary
1	Female	$111,818.18
2	Male	$81,526.32

Query executed successfully.

10. Write an SQL statement that shows employee IDs that have completed more than one class with a grade of B or better. Include an Aliases when appropriate. *This is a unique question where you might include a HAVING Clause to treat each employee as a group.

SELECT EmployeeID AS employee FROM EMPLOYEE_TRAINING WHERE Grade >= 'B' GROUP BY   EmployeeID HAVING COUNT(DISTINCT TrainingID) > 1;


SQLQuery2.sql - stairway.usu.edu.yubrajchaulagain (yubrajchaulagain (70))* - Microsoft SQL Server Management Studio
Object Explorer
Results
Messages
Ready

```
SELECT EmployeeID AS employee FROM EMPLOYEE_TRAINING WHERE
    Grade >= 'B'
GROUP BY
    EmployeeID
HAVING
    COUNT(DISTINCT TrainingID) > 1;
```

```
   employee
1  5
2  8
3  27
```

Query executed successfully.

11. Write an SQL statement that shows each Grade and the latest date where someone was assigned a grade using MAX(Completion Date) code in the SELECT statement of solution.

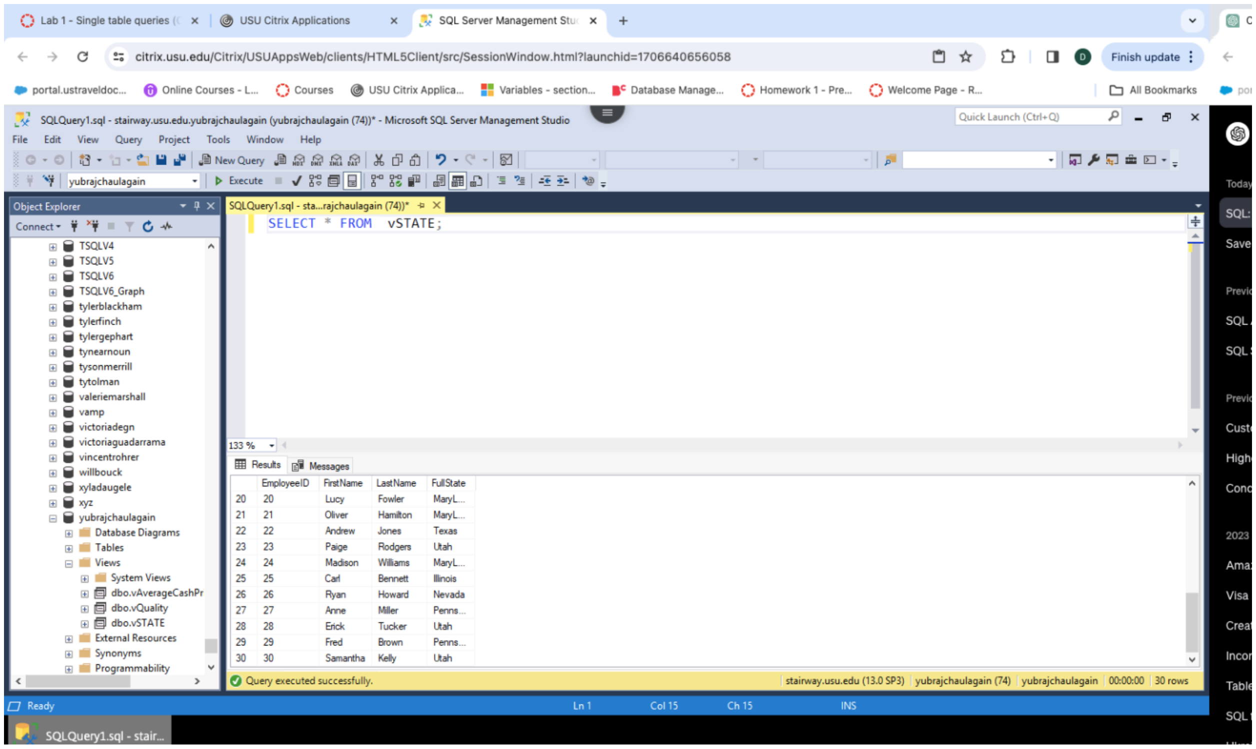
SELECT Grade, MAX(CompletionDate) AS LatestAssignmentDate FROM EMPLOYEE_TRAINING GROUP BY Grade;

Part 2

1. Curb uses the Employee data for corporate presentations. He would like his data output to show the full spelling for state information rather than abbreviations. The database administrator is unwilling to accommodate this request. Help Curb by creating a view called vSTATE that contains the full name of the employee along with the full spelling of the state using a CASE EXPRESSION. Just include the states that are included in the Employee Table.

```
CREATE VIEW vSTATE AS

SELECT    EmployeeID, FirstName,LastName,

    CASE State

        WHEN 'IL' THEN 'Illinois'

        WHEN 'NV' THEN 'Nevada'

        WHEN 'TX' THEN 'Texas'

        WHEN 'MD' THEN 'MaryLand'

        WHEN 'PA' THEN 'Pennsylvania'

         WHEN 'UT' THEN 'Utah'

        ELSE State

    END AS FullState

FROM
```

Employee;



```
SELECT * FROM vSTATE;
```

	EmployeeID	FirstName	LastName	FullState
20	20	Lucy	Fowler	MaryL...
21	21	Oliver	Hamilton	MaryL...
22	22	Andrew	Jones	Texas
23	23	Paige	Rodgers	Utah
24	24	Madison	Williams	MaryL...
25	25	Carl	Bennett	Illinois
26	26	Ryan	Howard	Nevada
27	27	Anne	Miller	Penns...
28	28	Erick	Tucker	Utah
29	29	Fred	Brown	Penns...
30	30	Samantha	Kelly	Utah

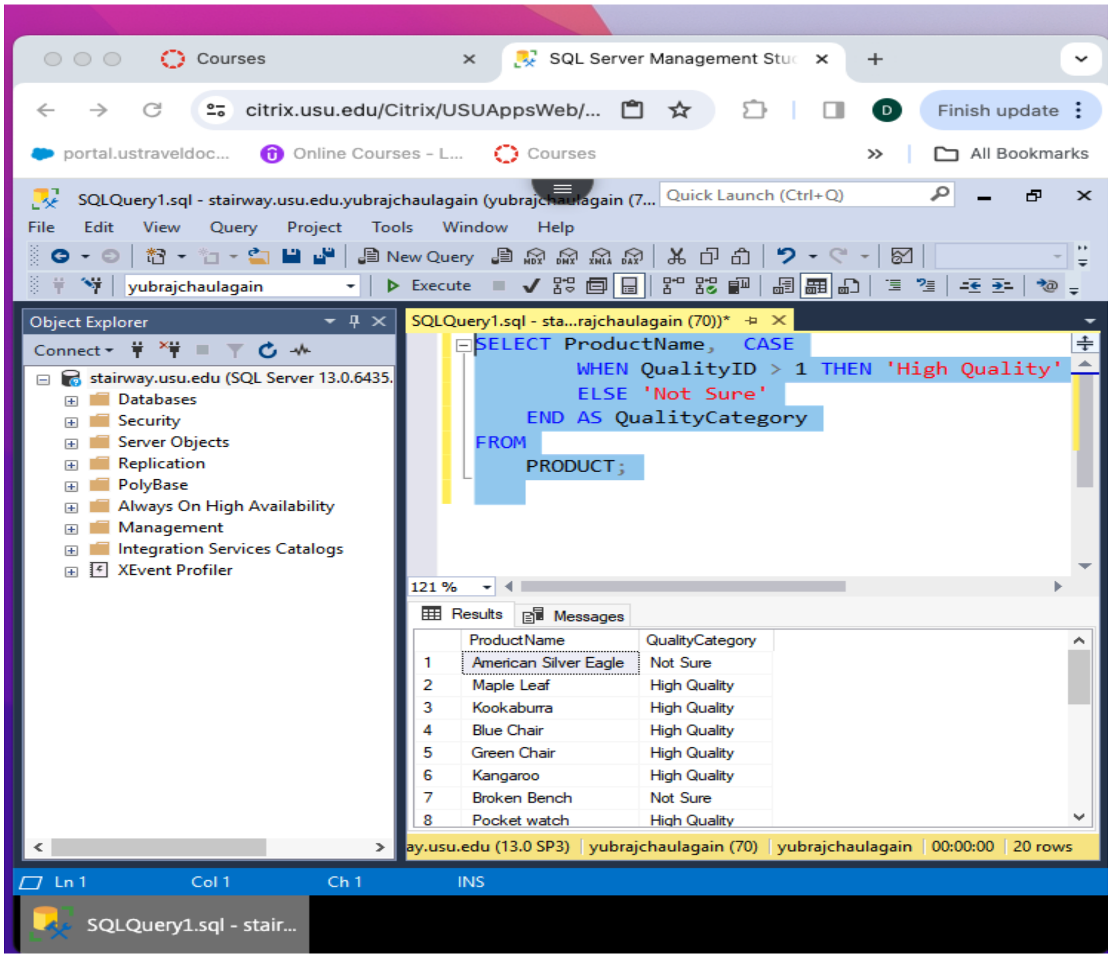
```
Query executed successfully.    stairway.usu.edu (13.0 SP3) | yubrajchaulagain (74) | yubrajchaulagain | 00:00:00 | 30 rows
```

2. For internal use, the antique store would like to create a view that provides additional information regarding product quality. In the current product database, if the quality is 1, the actual quality is Very Good. If the quality is 2, the actual quality is About Uncirculated. Finally, if the quality is 3, the actual quality is Mint State. Write an SQL view called vQuality that creates a new virtual column that includes actual quality descriptions. Show the ProductID, ProductName, and New Column.

3. Write a searched CASE Expression using the PRODUCT table that provides a new column that displays 'High Quality' if qualityID is > 1 and 'Not Sure' for any other possibilities (ELSE statement). Show the ProductName and New column.

```
 SELECT ProductName,  CASE

        WHEN QualityID > 1 THEN 'High Quality'

        ELSE 'Not Sure'

    END AS QualityCategory

FROM
```

PRODUCT;



```
SELECT ProductName,  CASE
        WHEN QualityID > 1 THEN 'High Quality'
        ELSE 'Not Sure'
    END AS QualityCategory
FROM
    PRODUCT;
```

	ProductName	QualityCategory
1	American Silver Eagle	Not Sure
2	Maple Leaf	High Quality
3	Kookaburra	High Quality
4	Blue Chair	High Quality
5	Green Chair	High Quality
6	Kangaroo	High Quality
7	Broken Bench	Not Sure
8	Pocket watch	High Quality

ay.usu.edu (13.0 SP3) | yubrajchaulagain (70) | yubrajchaulagain | 00:00:00 | 20 rows

4. Write your own SQL coding question and answer that requires a case expression. Indicate if the CASE Expression is Simple or Searched.

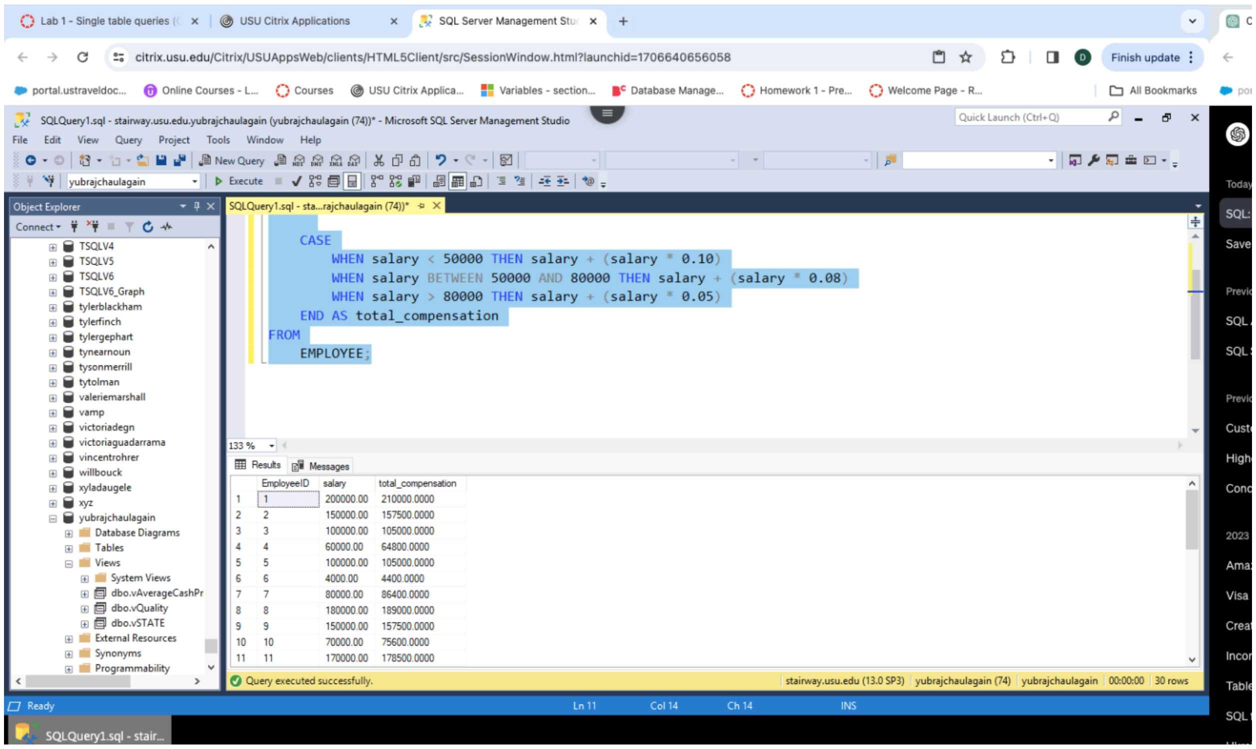
Question:

Consider a table named employees with columns employee_id, salary, Write an SQL query to create a new column named total_compensation based on the following conditions:

- If the employee's salary is less than 50000, add a bonus of 10%.

- If the employee's salary is between 50000 and 80000 (inclusive), add a bonus of 8%.
- If the employee's salary is greater than 80000, add a bonus of 5%.

```
SELECT
    EmployeeID,
    salary,

    CASE
        WHEN salary < 50000 THEN salary + (salary * 0.10)
        WHEN salary BETWEEN 50000 AND 80000 THEN salary + (salary * 0.08)
        WHEN salary > 80000 THEN salary + (salary * 0.05)
    END AS total_compensation
FROM
    EMPLOYEE;
```



```
    CASE
        WHEN salary < 50000 THEN salary + (salary * 0.10)
        WHEN salary BETWEEN 50000 AND 80000 THEN salary + (salary * 0.08)
        WHEN salary > 80000 THEN salary + (salary * 0.05)
    END AS total_compensation
FROM
    EMPLOYEE;
```

```
   EmployeeID  salary     total_compensation
1  1           200000.00  210000.0000
2  2           150000.00  157500.0000
3  3           100000.00  105000.0000
4  4           60000.00   64800.0000
5  5           100000.00  105000.0000
6  6           4000.00    4400.0000
7  7           80000.00   86400.0000
8  8           180000.00  189000.0000
9  9           150000.00  157500.0000
10 10          70000.00   75600.0000
11 11          170000.00  178500.0000
```

Query executed successfully.

Consider a hypothetical EMPLOYEE table with columns EmployeeID, FirstName, LastName, and YearsOfService. Write an SQL query that creates a new column called ServiceCategory using a CASE expression. If an employee has worked for the company for more than 5 years, categorize them as 'Long-Term'; if between 2 and 5 years, categorize them as 'Mid-Term'; otherwise, categorize them as 'Short-Term'.

```
SELECT

    EmployeeID,

    FirstName,

    LastName,

    YearsOfService,

    CASE

        WHEN YearsOfService > 5 THEN 'Long-Term'

        WHEN YearsOfService BETWEEN 2 AND 5 THEN 'Mid-Term'

        ELSE 'Short-Term'
```

END AS ServiceCategory

FROM

EMPLOYEE;

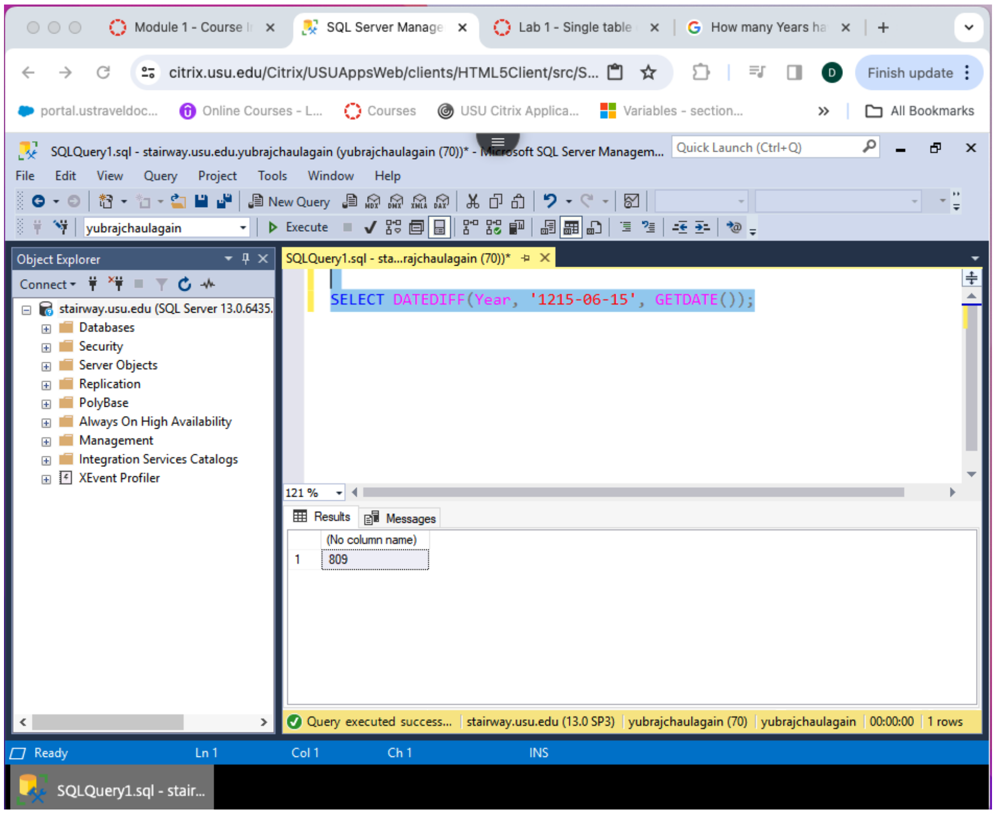
The CASE expression categorizes the years of service into different categories based on the specified conditions.

It is a Searched CASE Expression because it evaluates multiple conditions.

The result includes the original columns (EmployeeID, FirstName, LastName, YearsOfService) and a new column ServiceCategory

5. How many Years have passed since the Magna Carta was signed? Don't hardwire today's date. Use an Aliases when appropriate.

```
SELECT DATEDIFF(Year, '1215-06-15', GETDATE());
```



```
SELECT DATEDIFF(Year, '1215-06-15', GETDATE());
```

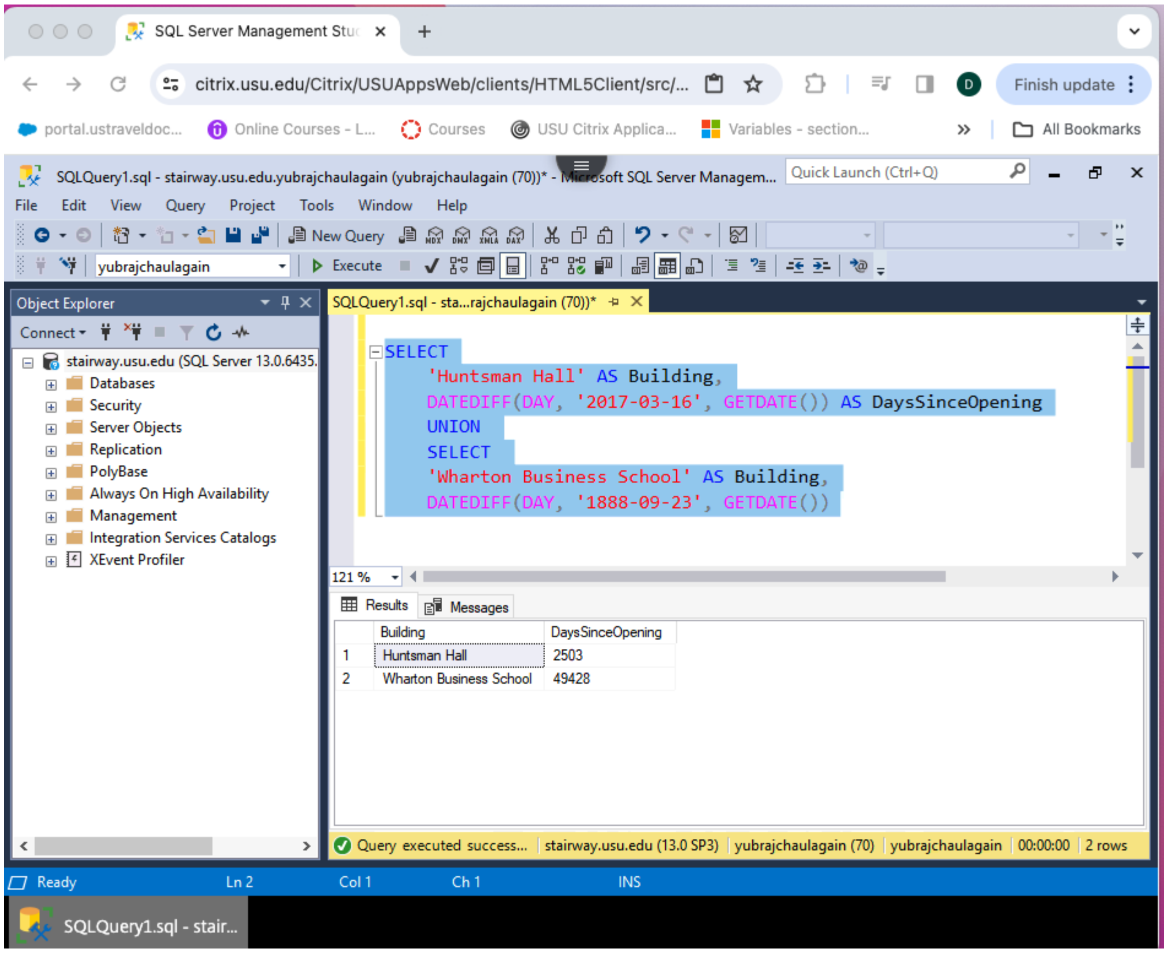
	(No column name)
1	809

Query executed success... | stairway.usu.edu (13.0 SP3) | yubrajchaulagain (70) | yubrajchaulagain | 00:00:00 | 1 rows

6. How many days has it been since Huntsman Hall opened at Utah State and Wharton Business School. For Wharton, you can include the year and make up the month and day? The output should include two columns and include aliases.

```
SELECT

   'Huntsman Hall' AS Building,

   DATEDIFF(DAY, '2017-03-16', GETDATE()) AS DaysSinceOpening

         UNION

         SELECT

         'Wharton Business School' AS Building,

         DATEDIFF(DAY, '1888-09-23', GETDATE())
```

X



```
SELECT
    'Huntsman Hall' AS Building,
    DATEDIFF(DAY, '2017-03-16', GETDATE()) AS DaysSinceOpening
    UNION
    SELECT
    'Wharton Business School' AS Building,
    DATEDIFF(DAY, '1888-09-23', GETDATE())
```

	Building	DaysSinceOpening
1	Huntsman Hall	2503
2	Wharton Business School	49428

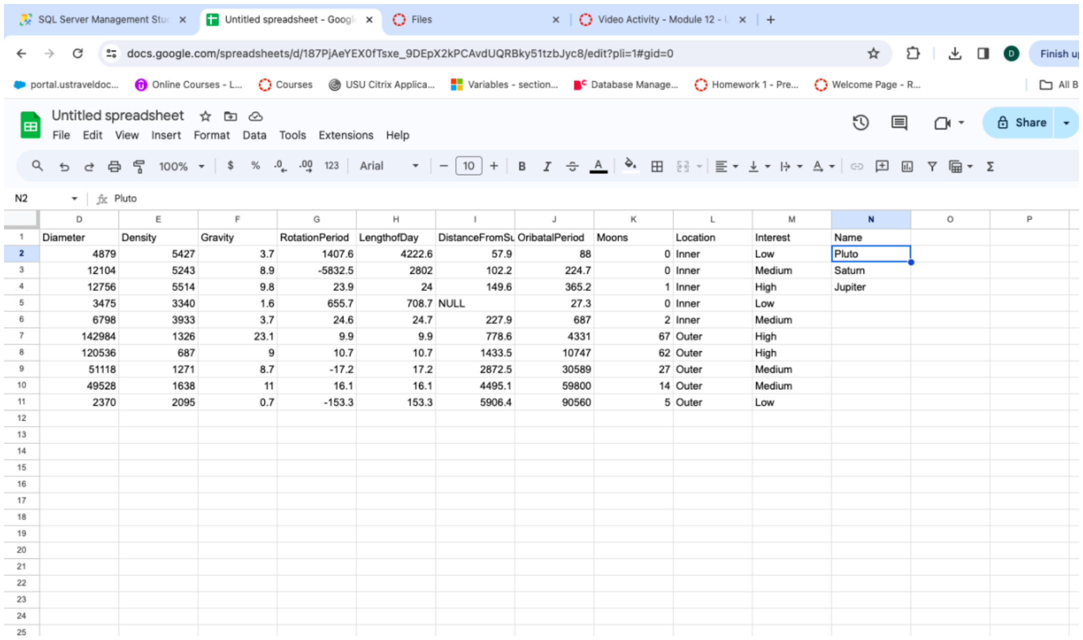
Query executed success... | stairway.usu.edu (13.0 SP3) | yubrajchaulagain (70) | yubrajchaulagain | 00:00:00 | 2 rows

Part 3 (SQL in Google Sheets)

Complete the two in-class video questions using Google Sheets. Complete the activity using the Planet spreadsheet in folder 0.5 - Google Sheet File under Files in Canvas. Submit a screen capture of your screen for each question.

IN CLASS # GS 1

```
=QUERY(A:M, "SELECT B WHERE D > 100000 OR B = 'Pluto' ORDER BY D",1)
```

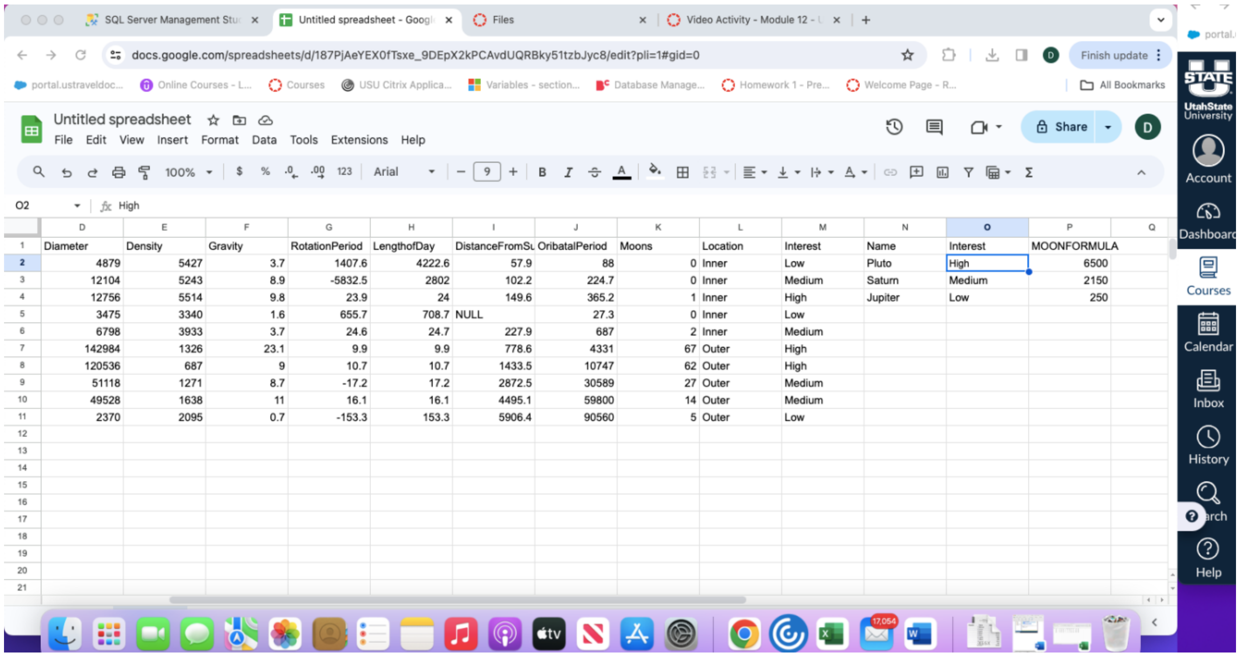


N2 Pluto

	D	E	F	G	H	I	J	K	L	M	N
1	Diameter	Density	Gravity	RotationPeriod	LengthofDay	DistanceFromSu	OribatalPeriod	Moons	Location	Interest	Name
2	4879	5427	3.7	1407.6	4222.6	57.9	88	0	Inner	Low	Pluto
3	12104	5243	8.9	-5832.5	2802	102.2	224.7	0	Inner	Medium	Saturn
4	12756	5514	9.8	23.9	24	149.6	365.2	1	Inner	High	Jupiter
5	3475	3340	1.6	655.7	708.7	NULL	27.3	0	Inner	Low	
6	6798	3933	3.7	24.6	24.7	227.9	687	2	Inner	Medium	
7	142984	1326	23.1	9.9	9.9	778.6	4331	67	Outer	High	
8	120536	687	9	10.7	10.7	1433.5	10747	62	Outer	High	
9	51118	1271	8.7	-17.2	17.2	2872.5	30589	27	Outer	Medium	
10	49528	1638	11	16.1	16.1	4495.1	59800	14	Outer	Medium	
11	2370	2095	0.7	-153.3	153.3	5906.4	90560	5	Outer	Low	

IN CLASS GS 2

```
=QUERY(A:M, "SELECT M, SUM(K) *50 GROUP BY M ORDER BY SUM(K) *50 DESC LABEL SUM(K) *50 'MOONFORMULA'",1)
```



O2 High

	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Diameter	Density	Gravity	RotationPeriod	LengthofDay	DistanceFromSu	OribatalPeriod	Moons	Location	Interest	Name	Interest	MOONFORMULA
2	4879	5427	3.7	1407.6	4222.6	57.9	88	0	Inner	Low	Pluto	High	6500
3	12104	5243	8.9	-5832.5	2802	102.2	224.7	0	Inner	Medium	Saturn	Medium	2150
4	12756	5514	9.8	23.9	24	149.6	365.2	1	Inner	High	Jupiter	Low	250
5	3475	3340	1.6	655.7	708.7	NULL	27.3	0	Inner	Low			
6	6798	3933	3.7	24.6	24.7	227.9	687	2	Inner	Medium			
7	142984	1326	23.1	9.9	9.9	778.6	4331	67	Outer	High			
8	120536	687	9	10.7	10.7	1433.5	10747	62	Outer	High			
9	51118	1271	8.7	-17.2	17.2	2872.5	30589	27	Outer	Medium			
10	49528	1638	11	16.1	16.1	4495.1	59800	14	Outer	Medium			
11	2370	2095	0.7	-153.3	153.3	5906.4	90560	5	Outer	Low			