Create a table for Employee as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EMPID | NAME | COUNTRY | SALARY | JOINING\_DATE | DEPARTMENT |
| 001 | Shippy | Jamaica | 100000 | 2022-02-20 09:00:00 | S/W Engg-A |
| 002 | Christopher | Jamaica | 800000 | 2022-06-11 09:00:00 | S/W Engg |
| 003 | ALANZO | India | 300000 | 2022-02-20 09:00:00 | S/W Engg |
| 004 | Ezra | Jamaica | 500000 | 2022-02-20 09:00:00 | S/W Engg-C |
| 005 | Grant | India | 500000 | 2022-06-11 09:00:00 | S/W Engg |
| 006 | Olando | Jamaica | 200000 | 2022-06-11 09:00:00 | S/W Engg |
| 007 | CAMI | India | 705000 | 2022-01-20 09:00:00 | S/W Engg-B |
| 008 | Donald | India | 900000 | 2022-04-11 09:00:00 | S/W Engg |

DROP TABLE IF EXISTS Employee;

CREATE TABLE Employee

(

EMPID INT PRIMARY KEY NOT NULL,

NAME VARCHAR(50) NOT NULL,

COUNTRY VARCHAR(50) NOT NULL,

SALARY DECIMAL(8,2) NOT NULL,

JOINING\_DATE DATETIME NOT NULL,

DEPARTMENT VARCHAR(50) NOT NULL

);

-- CREATING ANOTHER TABLE - Employee\_personal

-- Question 16 - Create a table Employee\_personal

DROP TABLE IF EXISTS Employee\_personal;

CREATE TABLE Employee\_personal

(

EMPID INT PRIMARY KEY NOT NULL,

NAME VARCHAR(50) NOT NULL,

PLACE VARCHAR(50) NOT NULL,

GENDER VARCHAR(50) NOT NULL,

AVAILABLE VARCHAR(50) NOT NULL,

PHNO VARCHAR(50) NOT NULL,

DEPARTMENT VARCHAR(50) NOT NULL

);

INSERT INTO

Employee

(EMPID, NAME, COUNTRY, SALARY, JOINING\_DATE, DEPARTMENT)

VALUES

(001, 'Shippy', 'Jamaica', 100000, '2022-02-20 09:00:00', 'S/W Engg-A'),

(002, 'Christopher', 'Jamaica', 800000, '2022-06-11 09:00:00', 'S/W Engg'),

(003, 'Alanzo', 'India', 300000, '2022-02-20 09:00:00', 'S/W Engg'),

(004, 'Ezra', 'Jamaica', 500000, '2022-02-20 09:00:00', 'S/W Engg-C'),

(005, 'Grant', 'India', 500000, '2022-06-11 09:00:00', 'S/W Engg'),

(006, 'Olando', 'Jamaica', 200000, '2022-06-11 09:00:00', 'S/W Engg'),

(007, 'Camille', 'India', 705000, '2022-01-20 09:00:00', 'S/W Engg-B'),

(008, 'Donald', 'India', 900000, '2022-04-11 09:00:00', 'S/W Engg');

INSERT INTO

Employee\_personal

(EMPID, NAME, PLACE, GENDER, AVAILABLE, PHNO, DEPARTMENT)

VALUES

(001, 'Shippy', 'Kingston', 'Male', 'Yes', '(876) 987-8346', 'S/W Engg-A'),

(002, 'Christopher', 'Montego Bay', 'Male', 'No', '(876) 785-6346', 'S/W Engg'),

(003, 'Alanzo', 'New Delhi', 'Male', 'Yes', '(876) 864-5346', 'S/W Engg'),

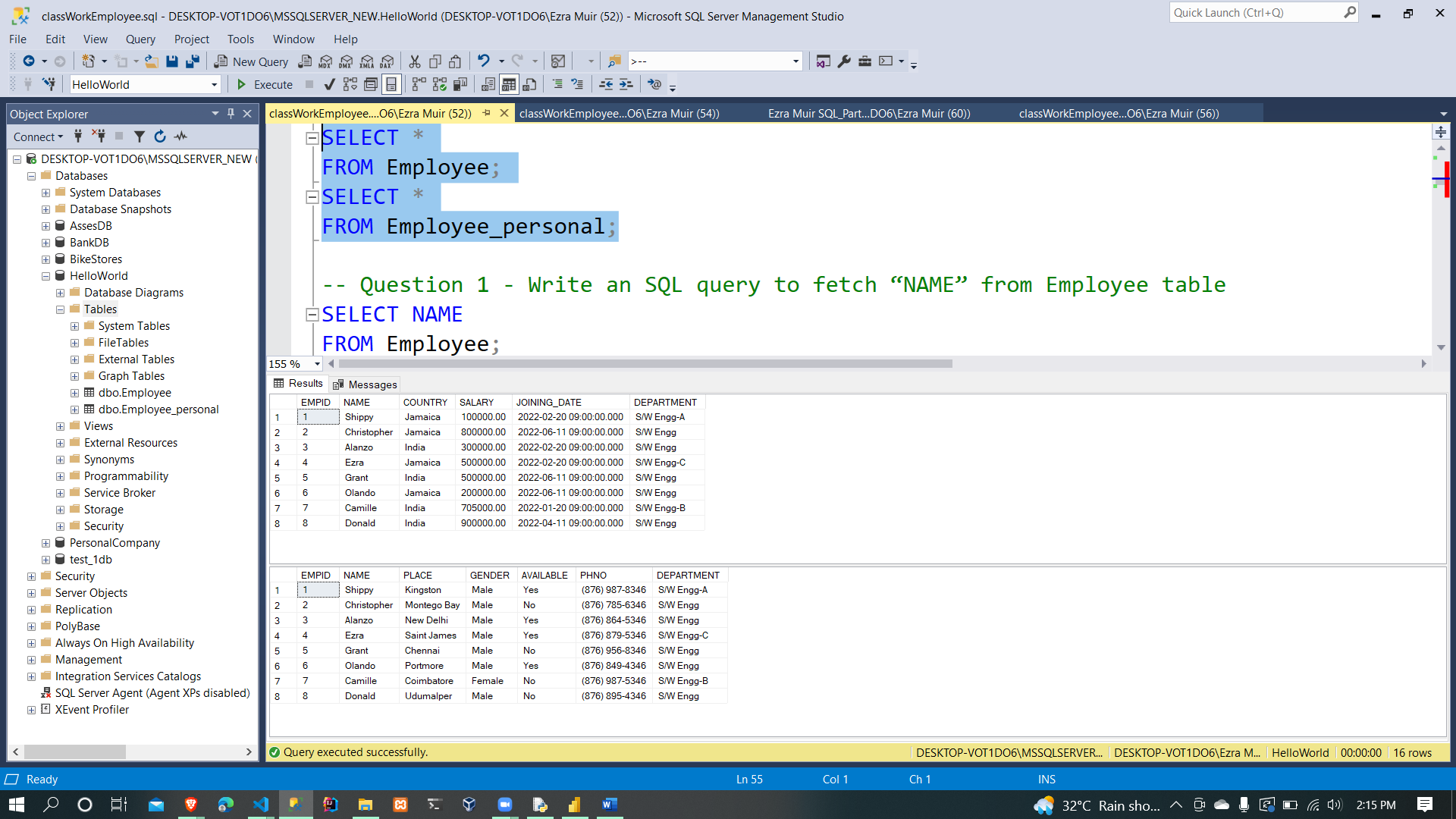
(004, 'Ezra', 'Saint James', 'Male', 'Yes', '(876) 879-5346', 'S/W Engg-C'),

(005, 'Grant', 'Chennai', 'Male', 'No', '(876) 956-8346', 'S/W Engg'),

(006, 'Olando', 'Portmore', 'Male', 'Yes', '(876) 849-4346', 'S/W Engg'),

(007, 'Camille', 'Coimbatore', 'Female', 'No', '(876) 987-5346', 'S/W Engg-B'),

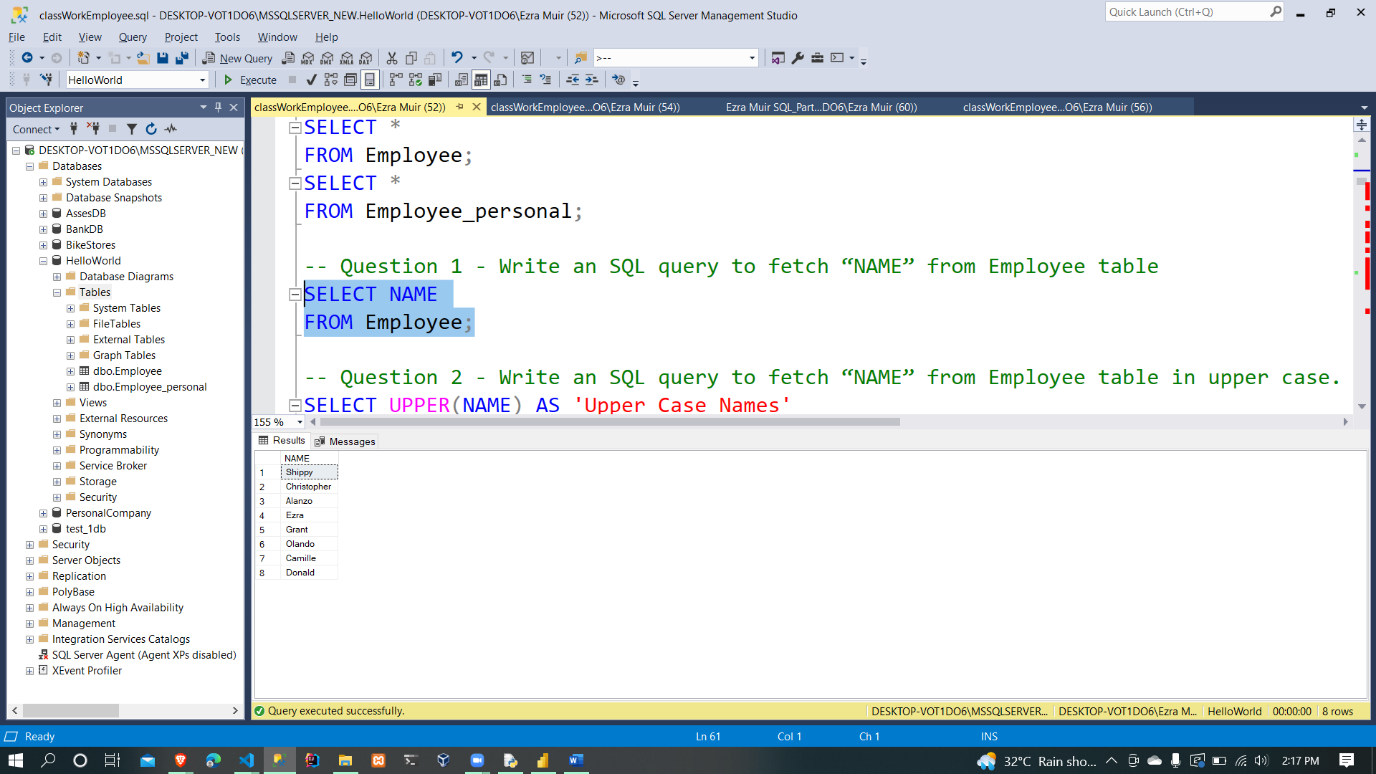
(008, 'Donald', 'Udumalper', 'Male', 'No', '(876) 895-4346', 'S/W Engg');



1. Write an SQL query to fetch “NAME” from Employee table

SELECT NAME

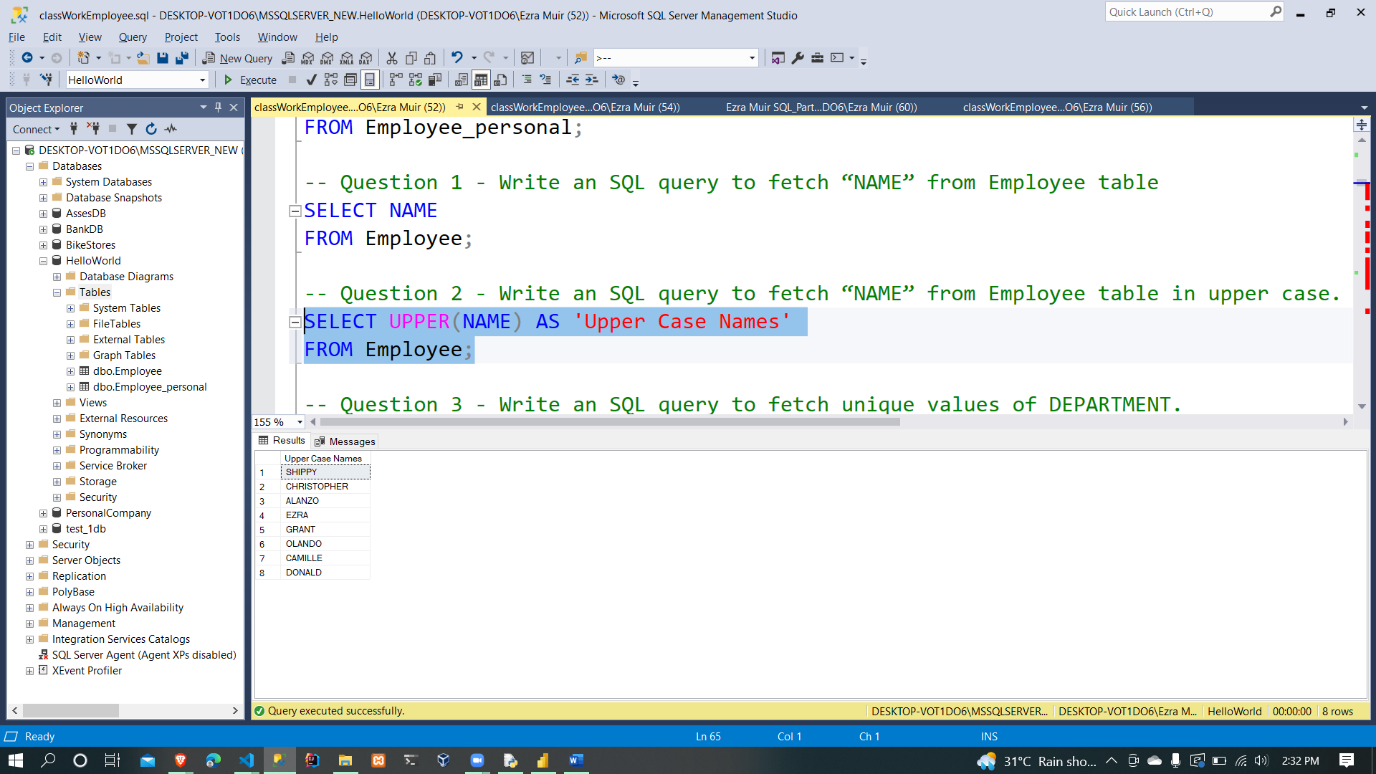
FROM Employee;



1. Write an SQL query to fetch “NAME” from Employee table in upper case.

SELECT UPPER(NAME) AS 'Upper Case Names'

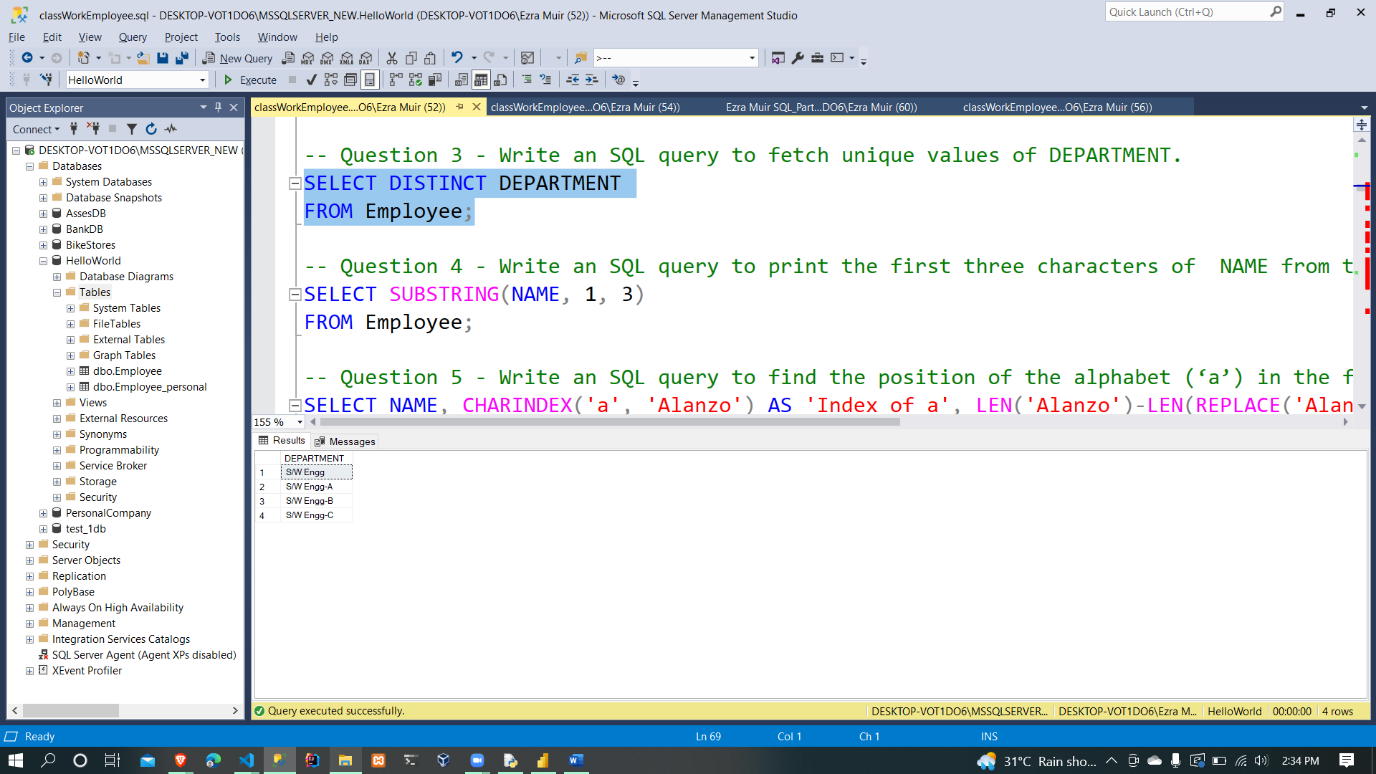
FROM Employee;



1. Write an SQL query to fetch unique values of DEPARTMENT.

SELECT DISTINCT DEPARTMENT

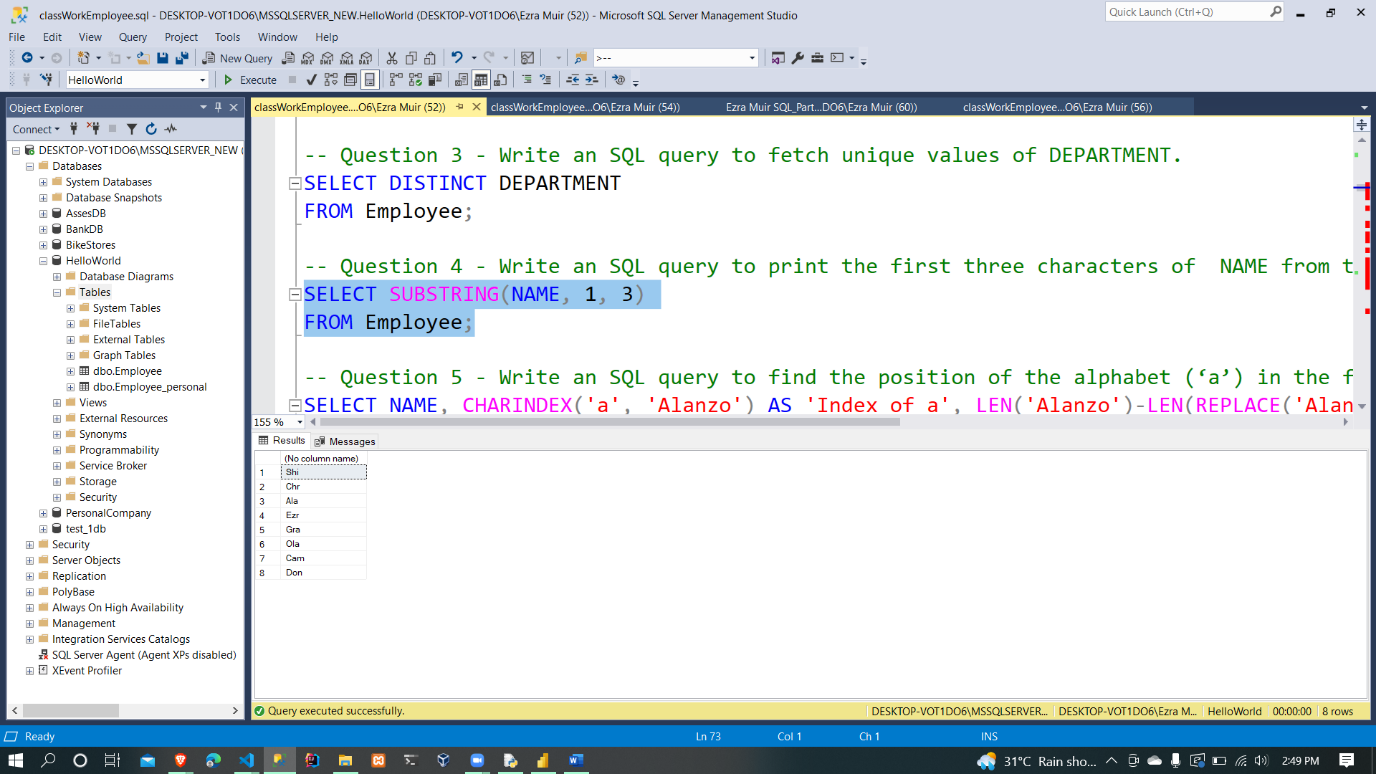
FROM Employee;



1. Write an SQL query to print the first three characters of  NAME from table.

SELECT SUBSTRING(NAME, 1, 3)

FROM Employee;

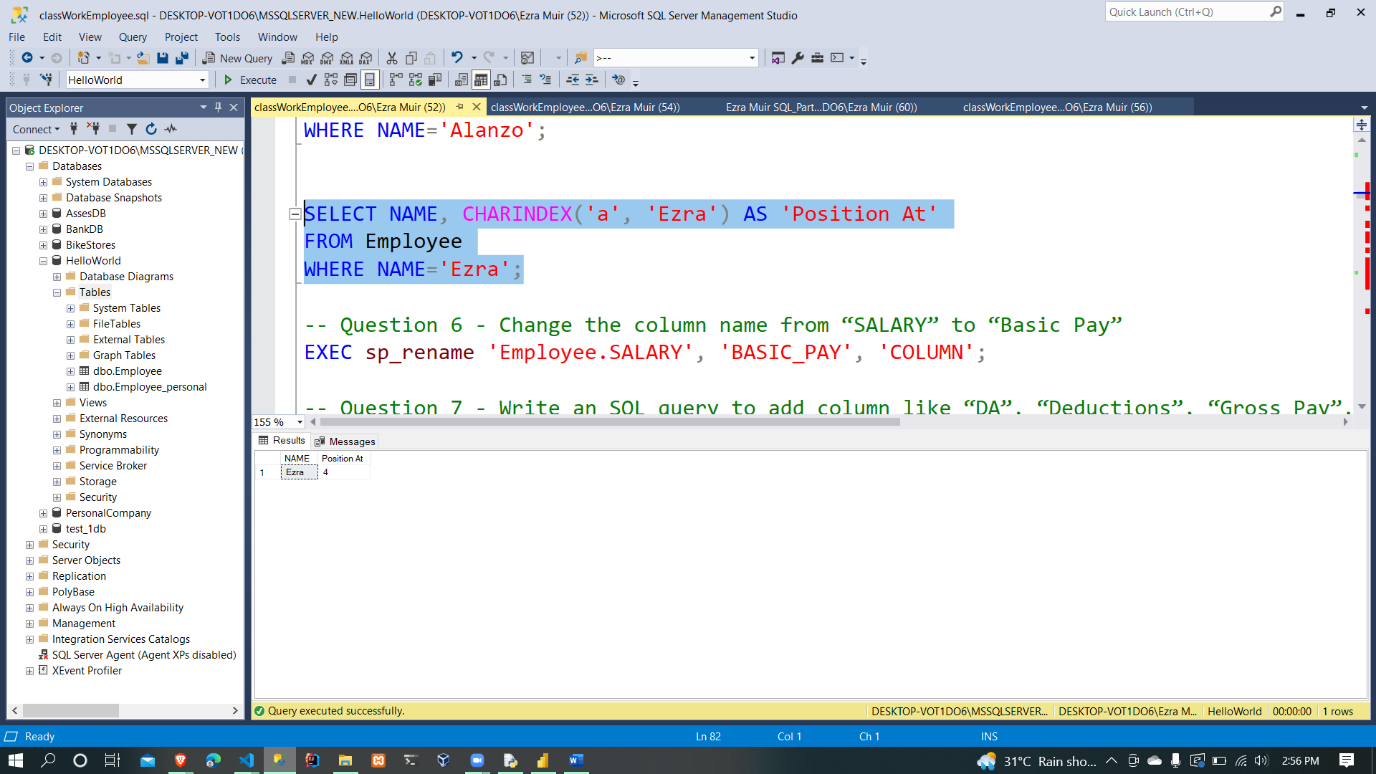


1. Write an SQL query to find the position of the alphabet (‘a’) in the first name column ‘Ezra from the table.

SELECT NAME, CHARINDEX('a', 'Ezra') AS 'Position At'

FROM Employee

WHERE NAME='Ezra';



1. Change the column name from “SALARY” to “Basic Pay”

EXEC sp\_rename 'Employee.SALARY', 'BASIC\_PAY', 'COLUMN';

1. Write an SQL query to add column like “DA”, “Deductions”, “Gross Pay”, “Net Pay”.

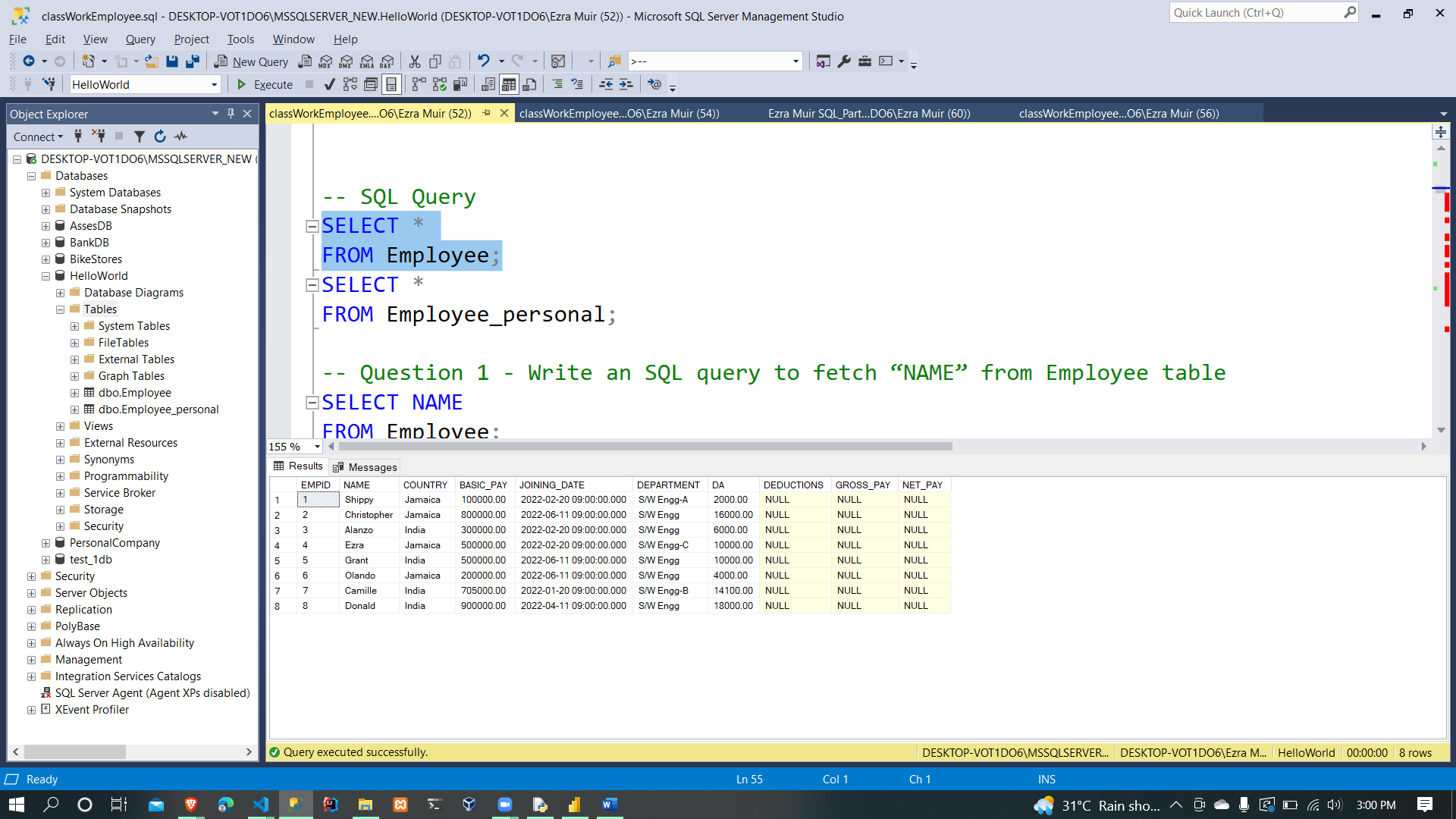
ALTER TABLE Employee

ADD DA DECIMAL(8,2),

DEDUCTIONS DECIMAL(8,2),

GROSS\_PAY DECIMAL(8,2),

NET\_PAY DECIMAL(8,2);



1. DA = 2% of Basic Pay

UPDATE Employee

SET DA = 0.02 \* BASIC\_PAY;

1. Deductions = 3% of Basic Pay

UPDATE Employee

SET DEDUCTIONS = 0.03 \* BASIC\_PAY;

1. Gross Pay =Basic Pay + DA

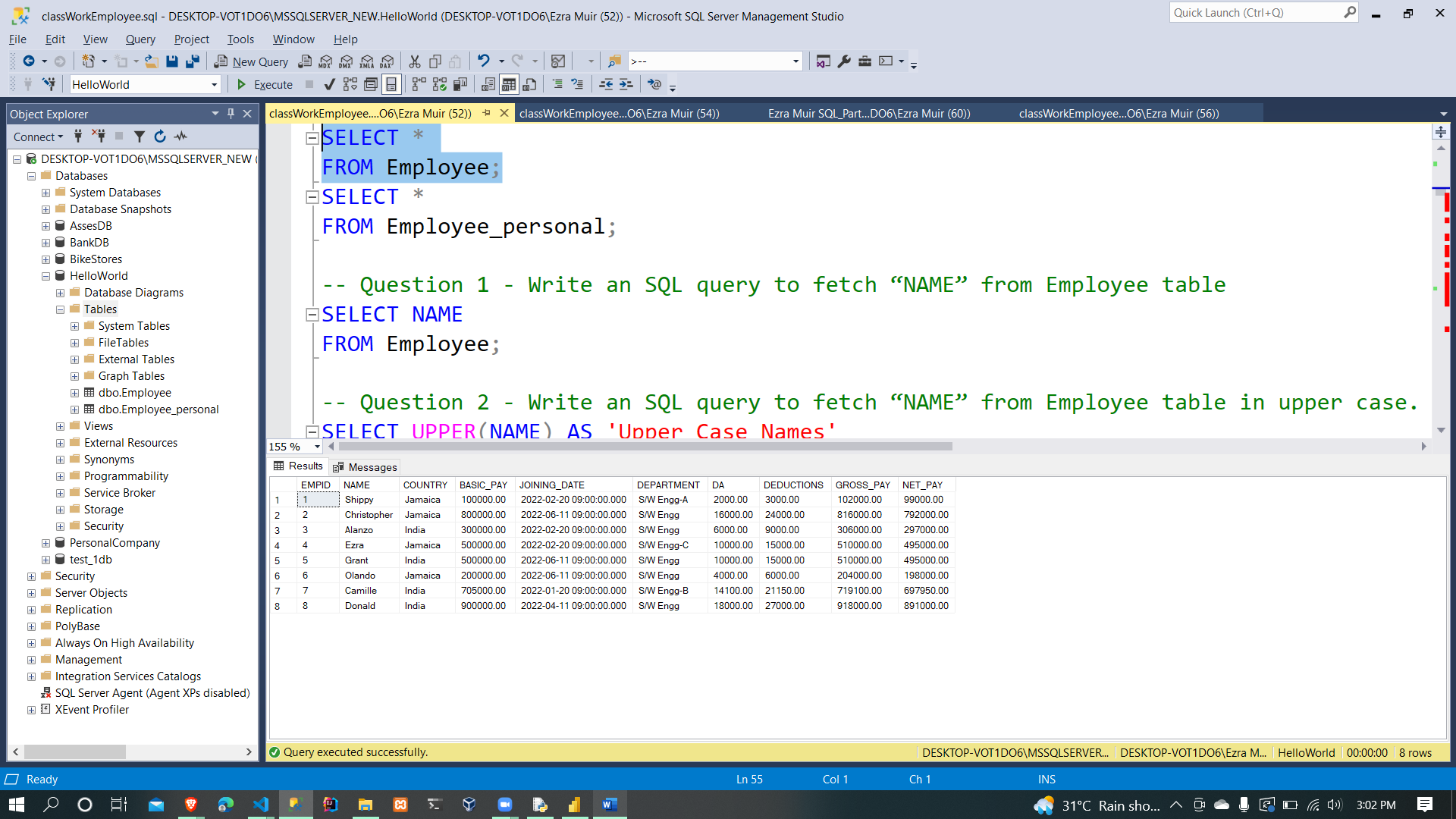
UPDATE Employee

SET GROSS\_PAY = BASIC\_PAY + DA;

1. Net Pay = Gross Pay - Deductions

UPDATE Employee

SET NET\_PAY = GROSS\_PAY - DEDUCTIONS;



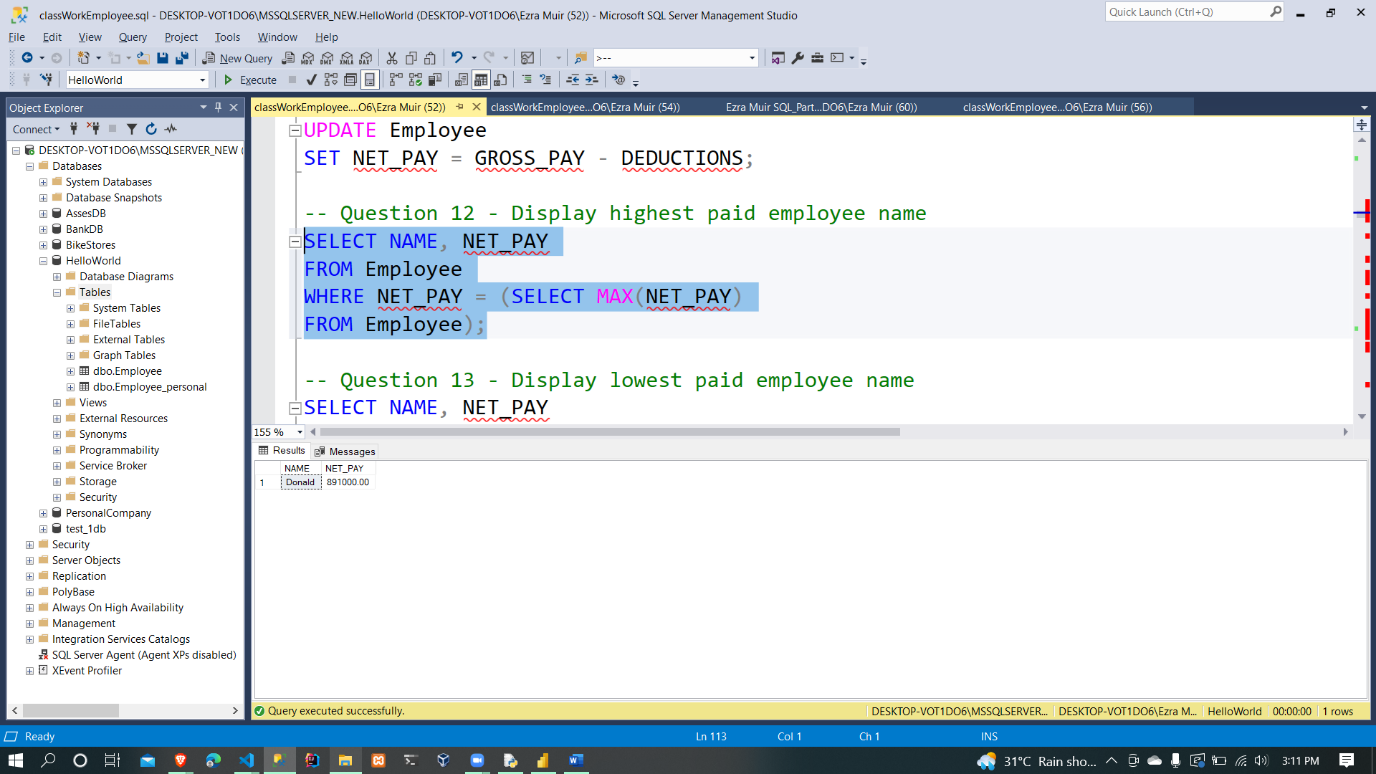
1. Display highest paid employee name

SELECT NAME, NET\_PAY

FROM Employee

WHERE NET\_PAY = (SELECT MAX(NET\_PAY)

FROM Employee);



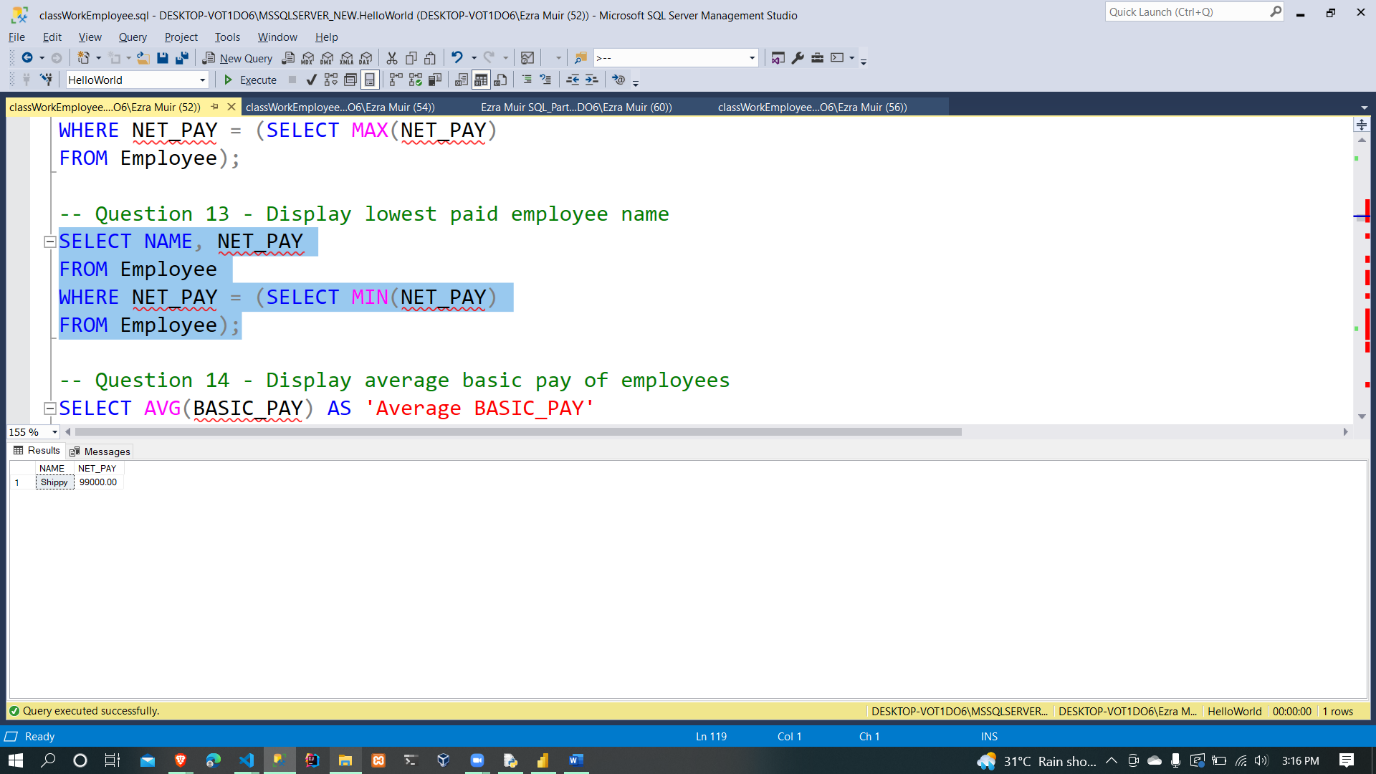
1. Display lowest paid employee name

SELECT NAME, NET\_PAY

FROM Employee

WHERE NET\_PAY = (SELECT MIN(NET\_PAY)

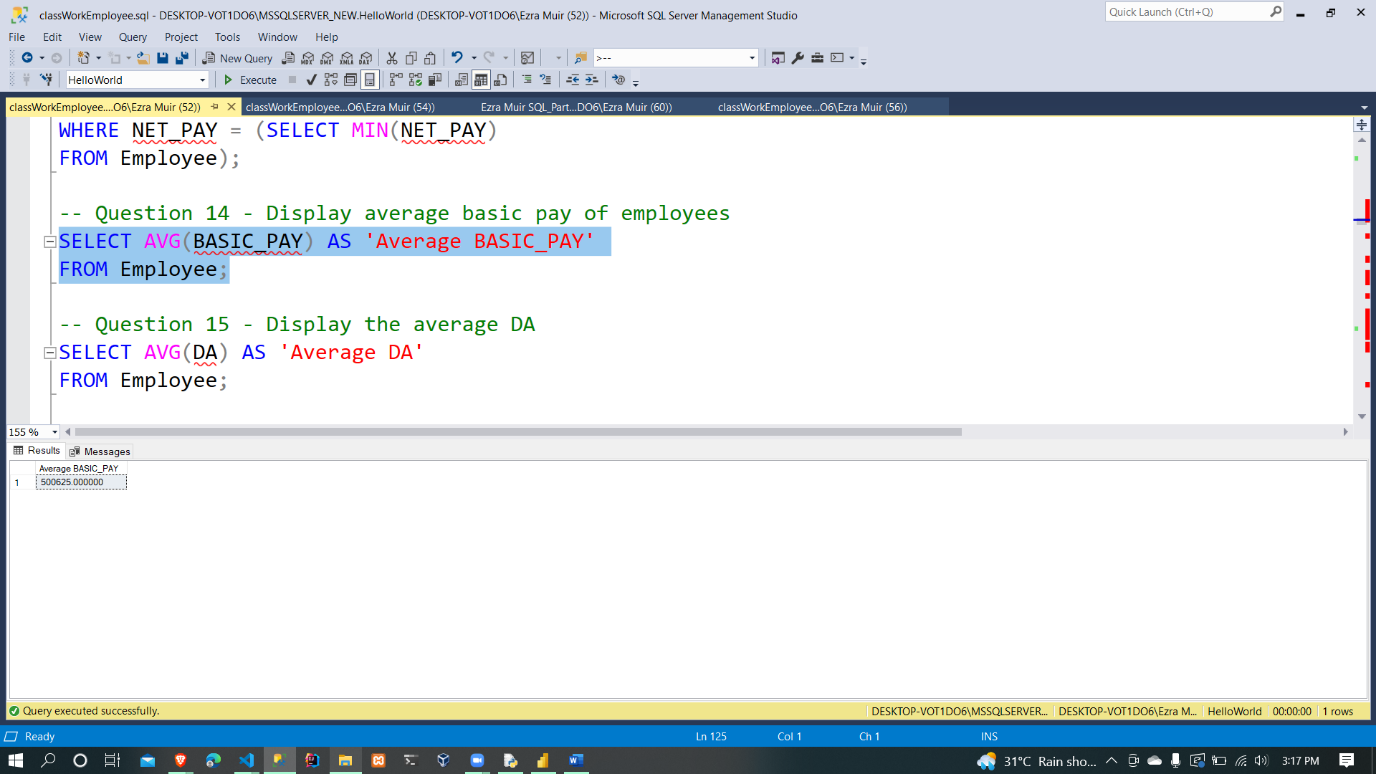
FROM Employee);



1. Display average basic pay of employees

SELECT AVG(BASIC\_PAY) AS 'Average BASIC\_PAY'

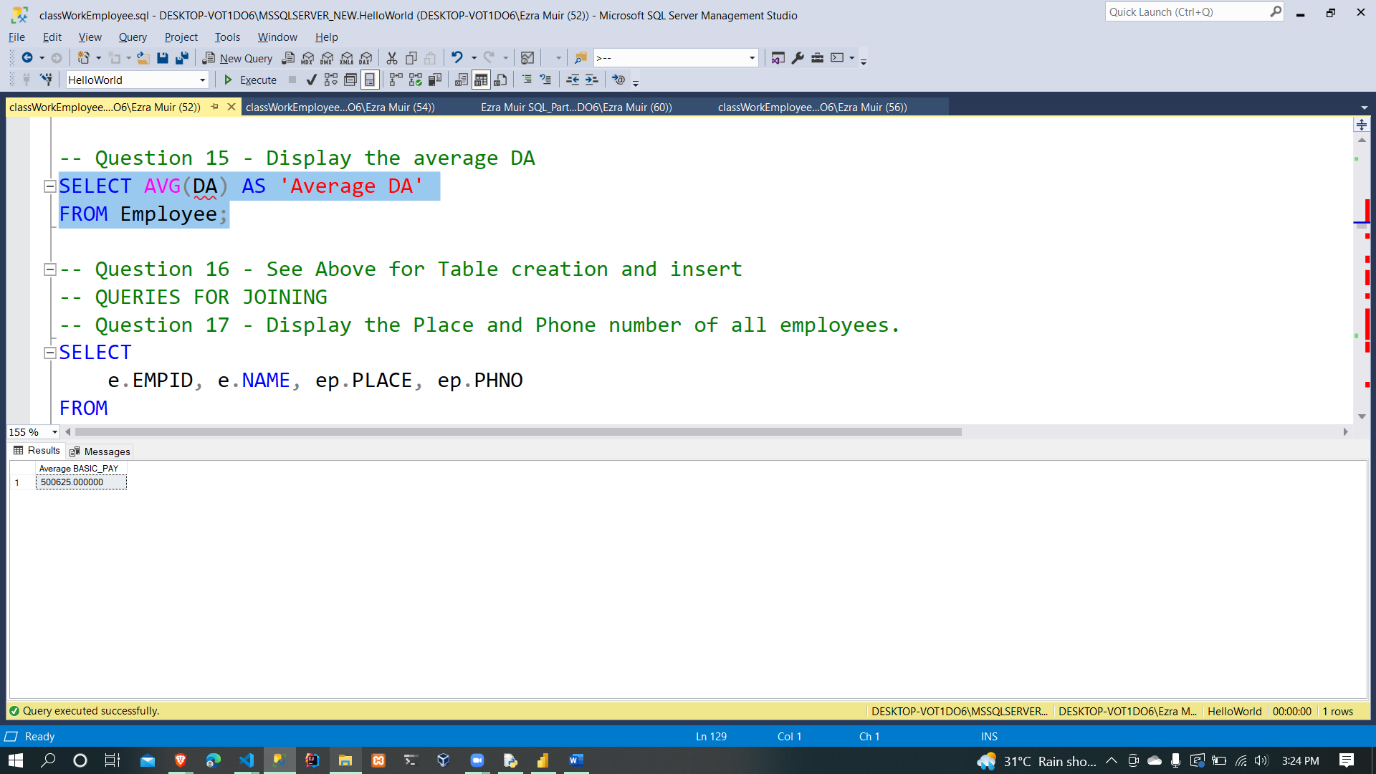
FROM Employee;



1. Display the average DA

SELECT AVG(DA) AS 'Average DA'

FROM Employee;



1. Create a table Employee\_personal

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| EMPID | NAME | PLACE | GENDER | AVAILABLE | Phno | DEPARTMENT |
| 001 | Shippy | Kingston | Male | Yes | 9878 | S/W Engg-A |
| 002 | Christopher | Montego Bay | Male | No | 7856 | S/W Engg |
| 003 | ALANZO | New Delhi | Male | Yes | 8645 | S/W Engg |
| 004 | Ezra | Saint James Parish | Male | Yes | 8795 | S/W Engg-C |
| 005 | Grant | Chennai | Male | No | 9568 | S/W Engg |
| 006 | Olando | Portmore | Male | Yes | 8494 | S/W Engg |
| 007 | CAMI | Coimbatore | Female | No | 9875 | S/W Engg-B |
| 008 | Donald | Udumalpet | Male | No | 8954 | S/W Engg |

1. Display the Place and Phone number of all employees.

SELECT

e.EMPID, e.NAME, ep.PLACE, ep.PHNO

FROM

Employee e

INNER JOIN Employee\_personal ep

ON e.NAME = ep.NAME



1. Display the Place and Country of employees with availability “Yes”

SELECT

e.NAME, e.COUNTRY, ep.PLACE, ep.AVAILABLE

FROM

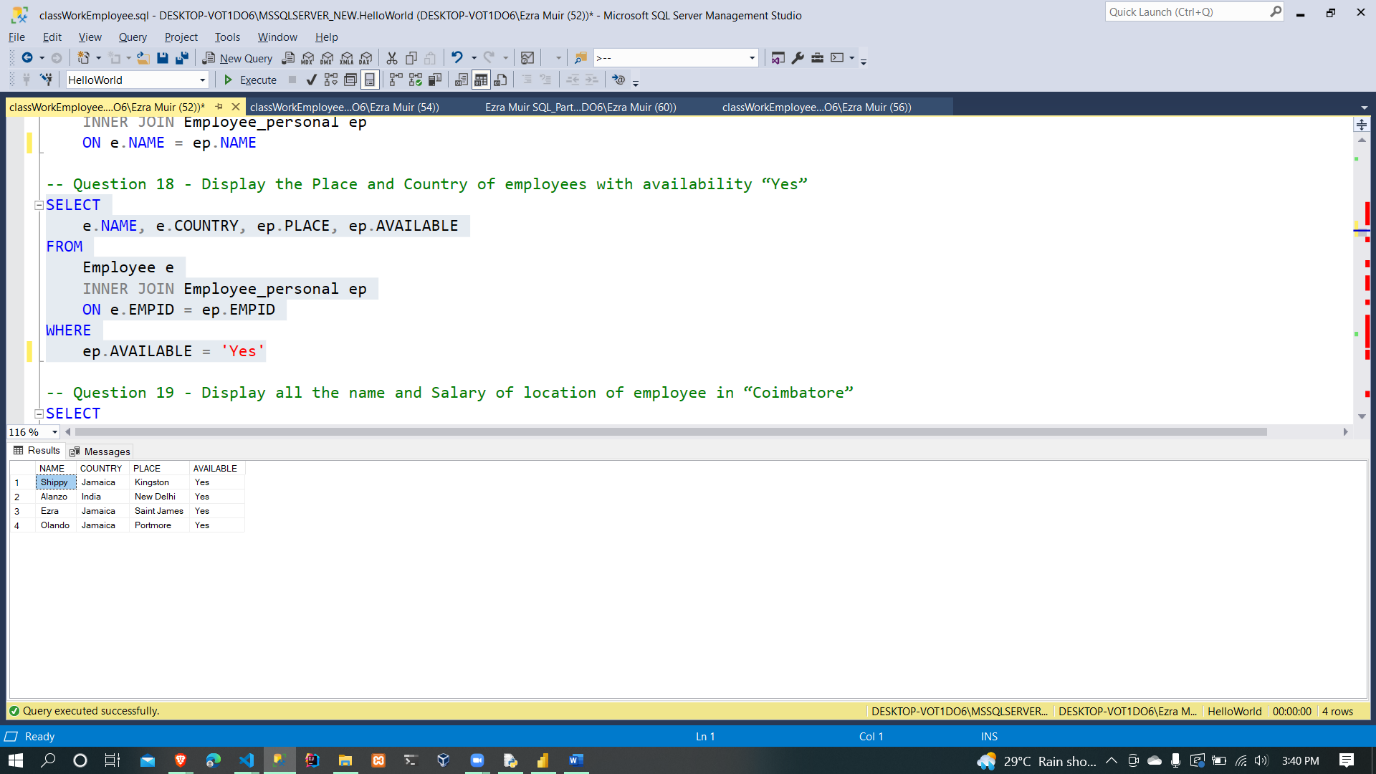
Employee e

INNER JOIN Employee\_personal ep

ON e.EMPID = ep.EMPID

WHERE

ep.AVAILABLE = 'Yes'



1. Display all the name and Salary of location of employee in “Coimbatore”

SELECT

e.NAME, e.BASIC\_PAY, e.DEPARTMENT, ep.PLACE

FROM

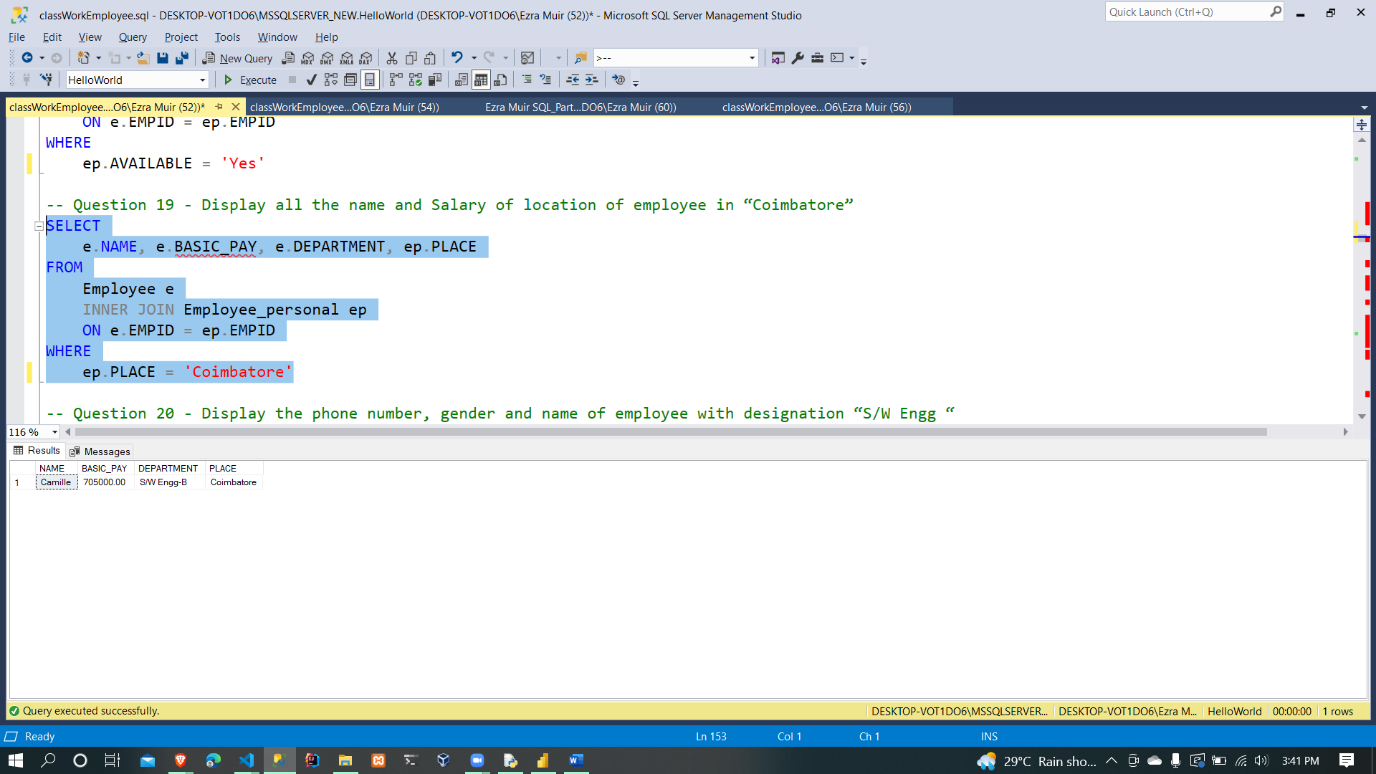
Employee e

INNER JOIN Employee\_personal ep

ON e.EMPID = ep.EMPID

WHERE

ep.PLACE = 'Coimbatore'



1. Display the phone number, gender and name of employee with designation “S/W Engg “

SELECT

e.NAME, ep.PHNO, ep.GENDER, e.DEPARTMENT

FROM

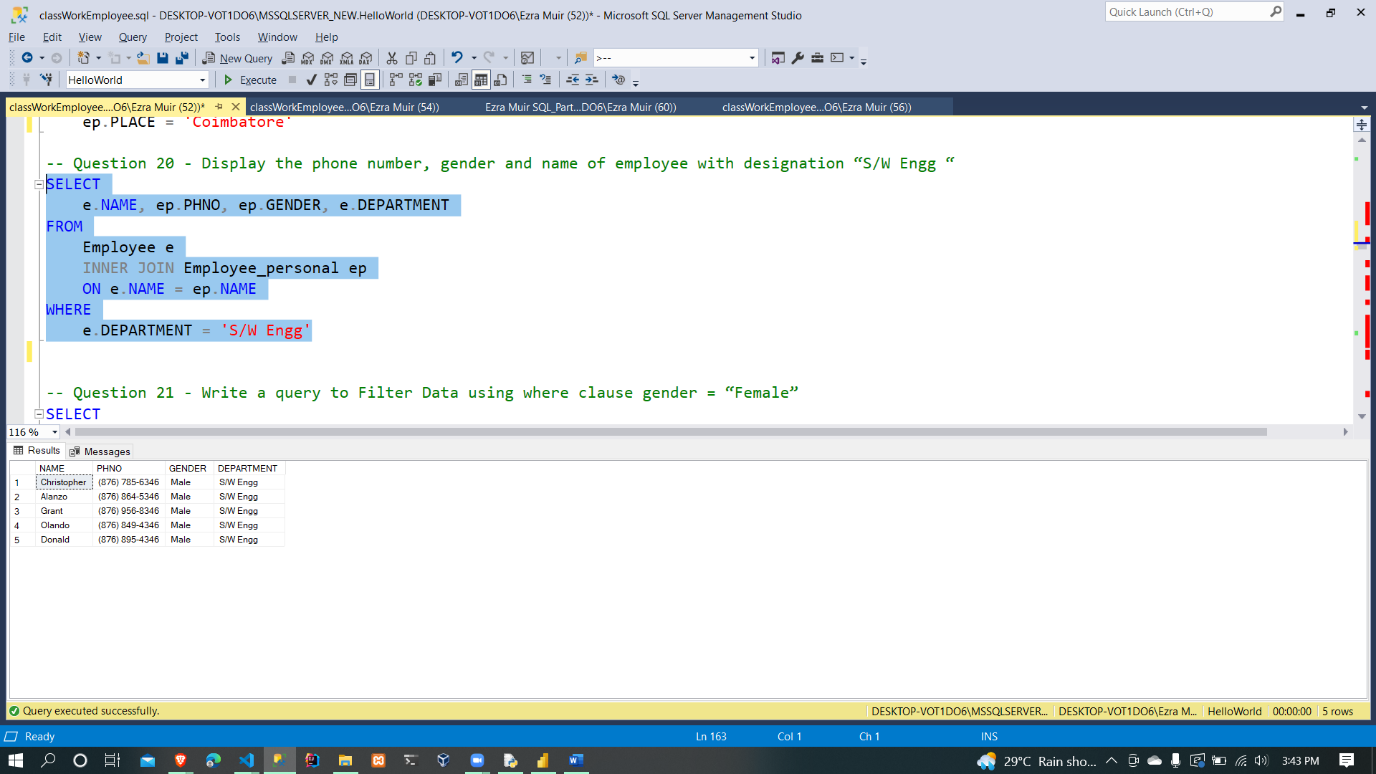
Employee e

INNER JOIN Employee\_personal ep

ON e.NAME = ep.NAME

WHERE

e.DEPARTMENT = 'S/W Engg'



1. Write a query to Filter Data using where clause gender = “Female”

SELECT

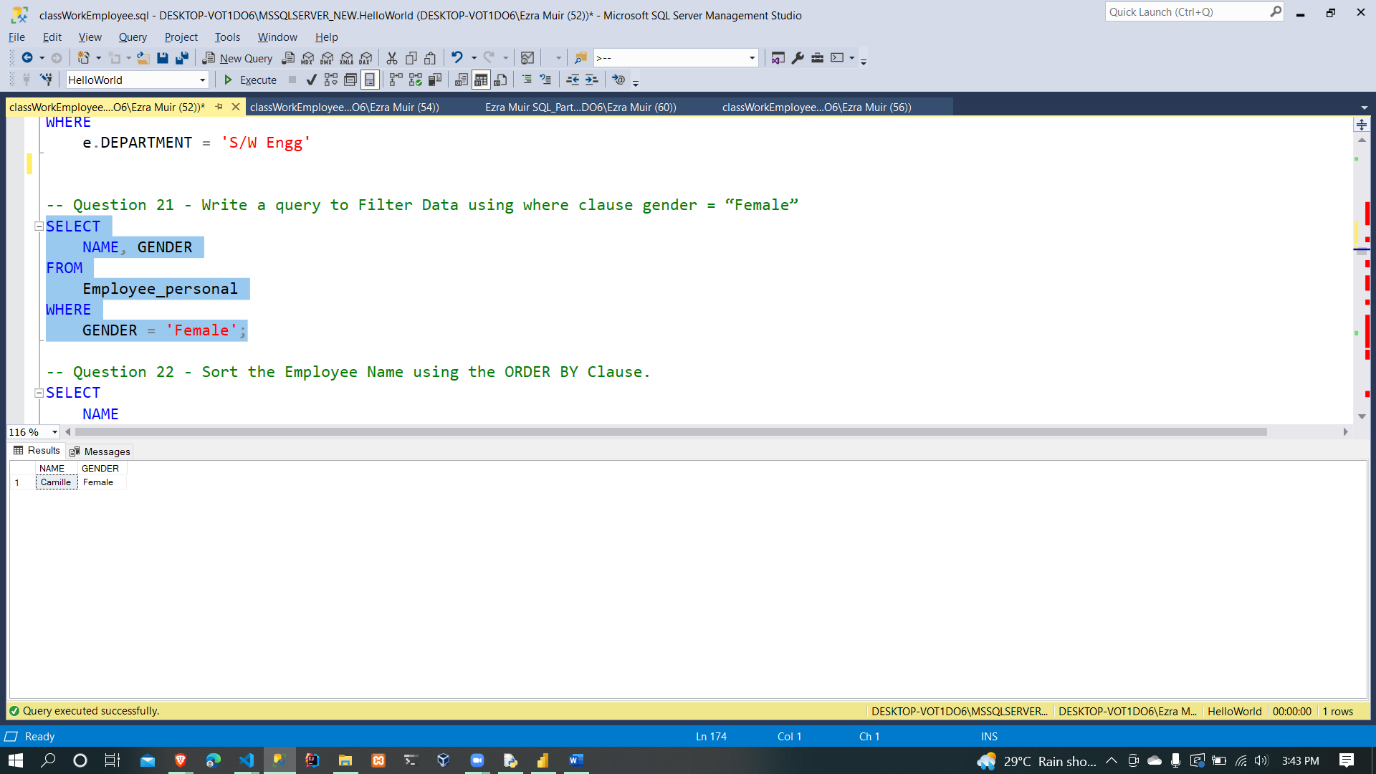
NAME, GENDER

FROM

Employee\_personal

WHERE

GENDER = 'Female';



1. Sort the Employee Name using the ORDER BY Clause.

SELECT

NAME

FROM

Employee

ORDER BY NAME;

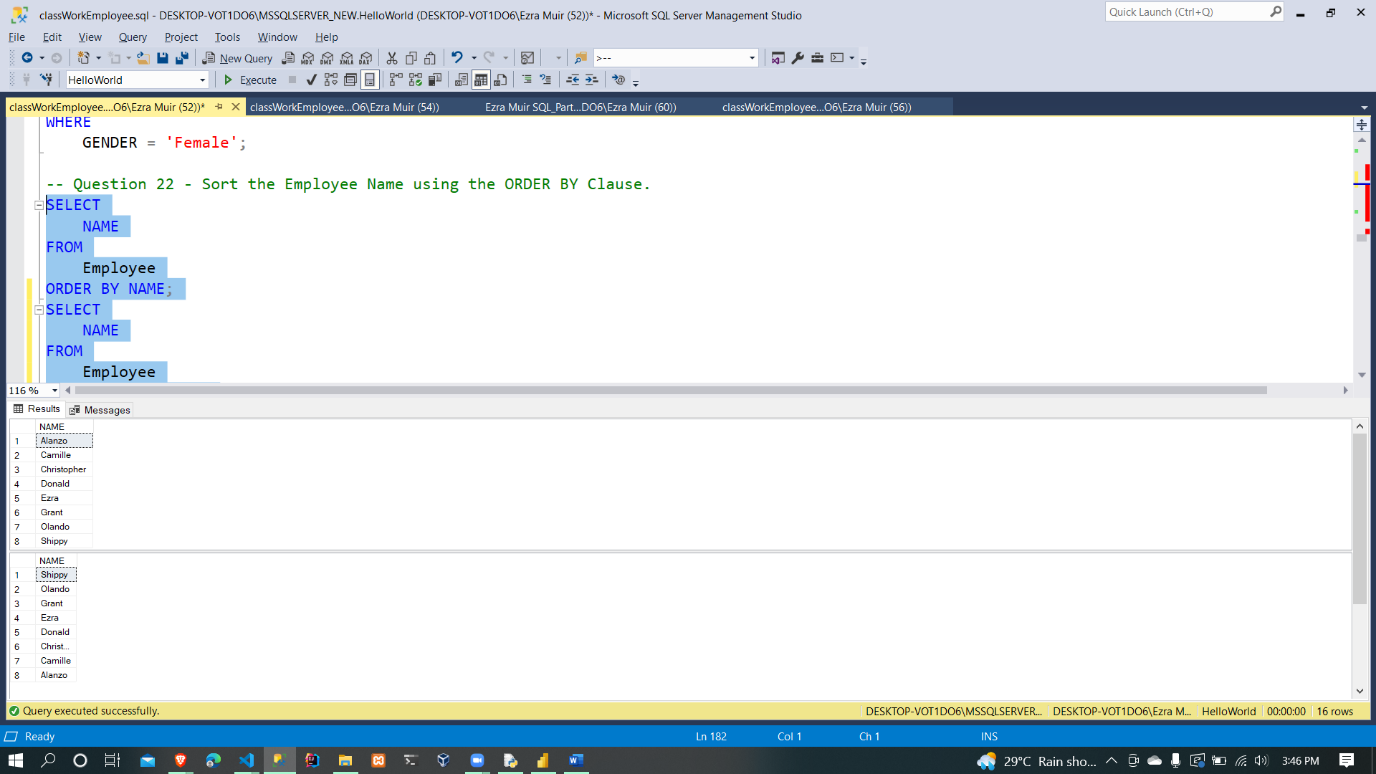
SELECT

NAME

FROM

Employee

ORDER BY NAME DESC;



1. Display the TOP (5) high salaried employees – display their name, salary and department.

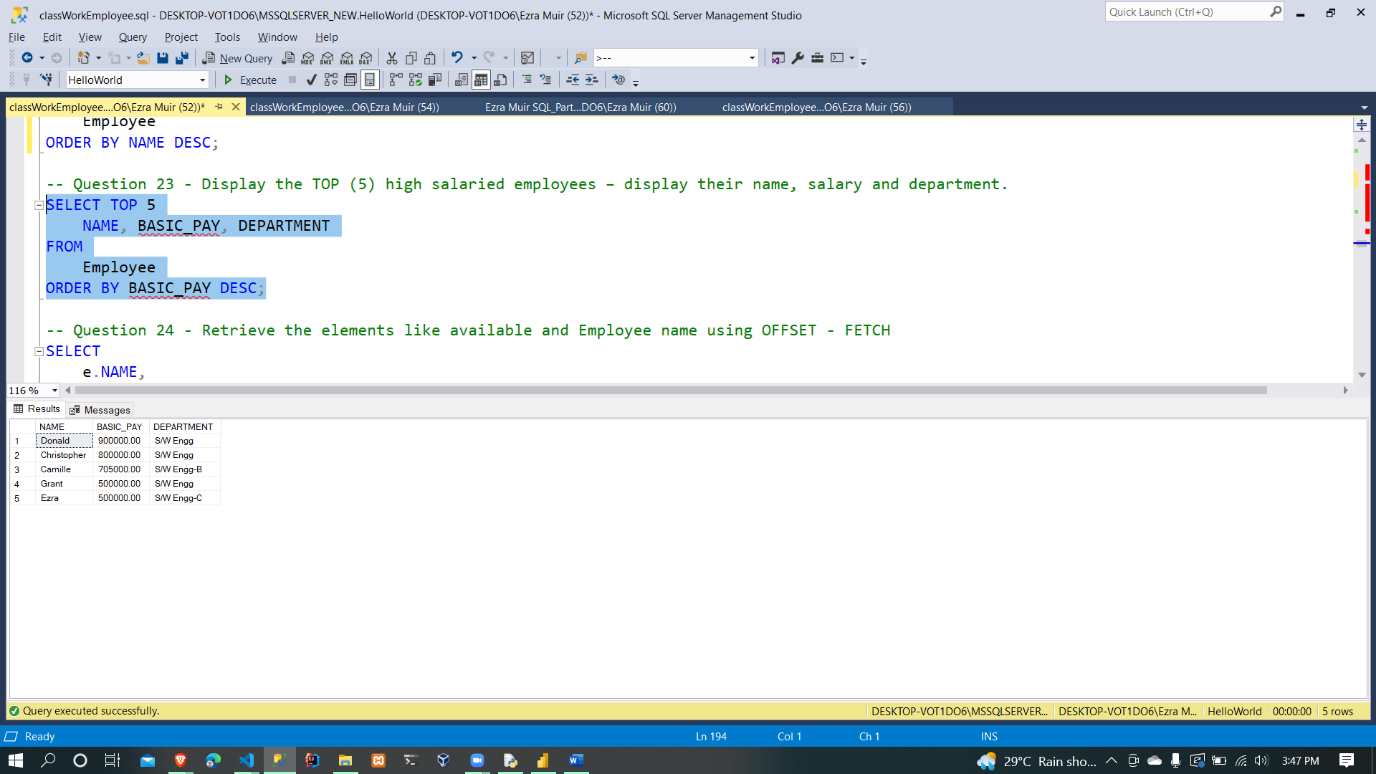
SELECT TOP 5

NAME, BASIC\_PAY, DEPARTMENT

FROM

Employee

ORDER BY BASIC\_PAY DESC;



1. Retrieve the elements like available and Employee name using OFFSET – FETCH

SELECT

e.NAME,

ep.AVAILABLE

FROM

Employee e

FULL OUTER JOIN Employee\_personal ep

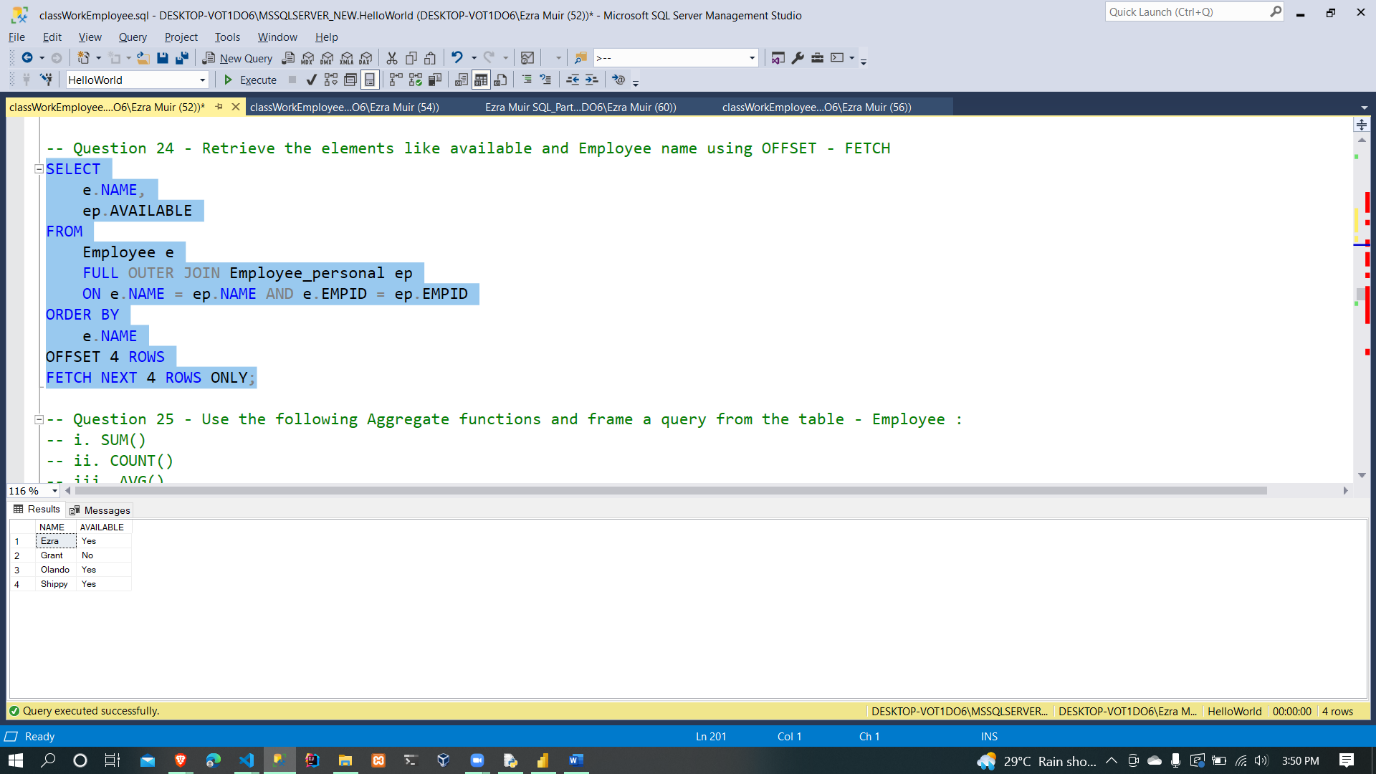
ON e.NAME = ep.NAME AND e.EMPID = ep.EMPID

ORDER BY

e.NAME

OFFSET 4 ROWS

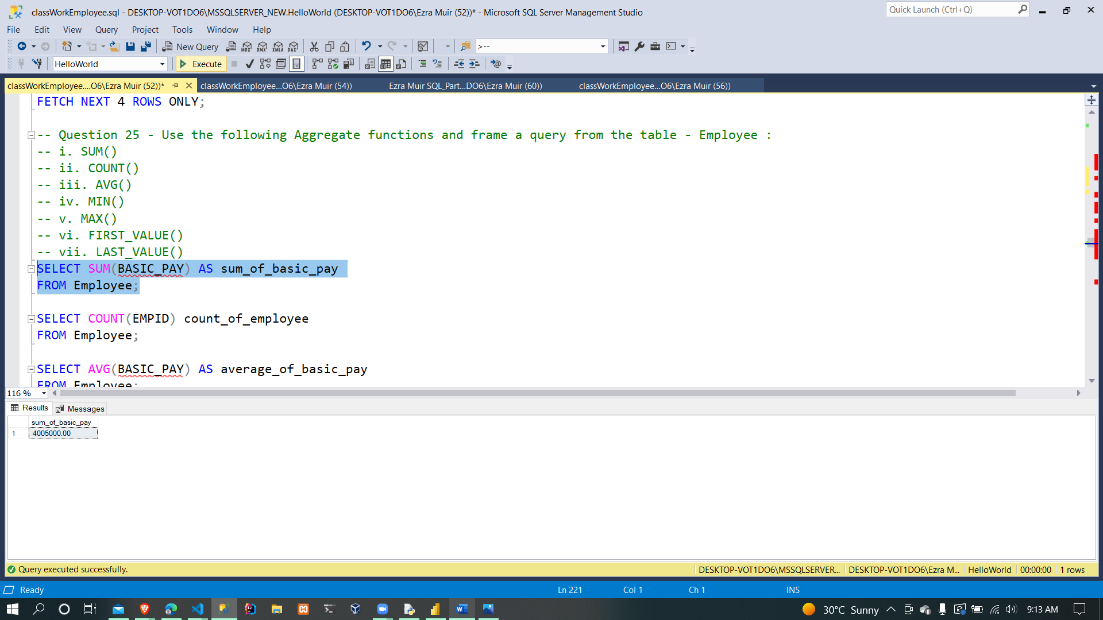
FETCH NEXT 4 ROWS ONLY;



1. Use the following Aggregate functions and frame a query from the table - Employee :
   * 1. [SUM()](https://www.edureka.co/blog/sql-functions#sum)

SELECT SUM(BASIC\_PAY) AS sum\_of\_basic\_pay

FROM Employee;



* + 1. [COUNT()](https://www.edureka.co/blog/sql-functions#count)

SELECT COUNT(EMPID) count\_of\_employee

FROM Employee;

* + 1. [AVG()](https://www.edureka.co/blog/sql-functions#avg)

SELECT AVG(BASIC\_PAY) AS average\_of\_basic\_pay

FROM Employee;

* + 1. [MIN()](https://www.edureka.co/blog/sql-functions#min)

SELECT MIN(BASIC\_PAY) AS minimum\_of\_basic\_pay

FROM Employee;

* + 1. [MAX()](https://www.edureka.co/blog/sql-functions#max)

SELECT MAX(BASIC\_PAY) AS maximum\_of\_basic\_pay

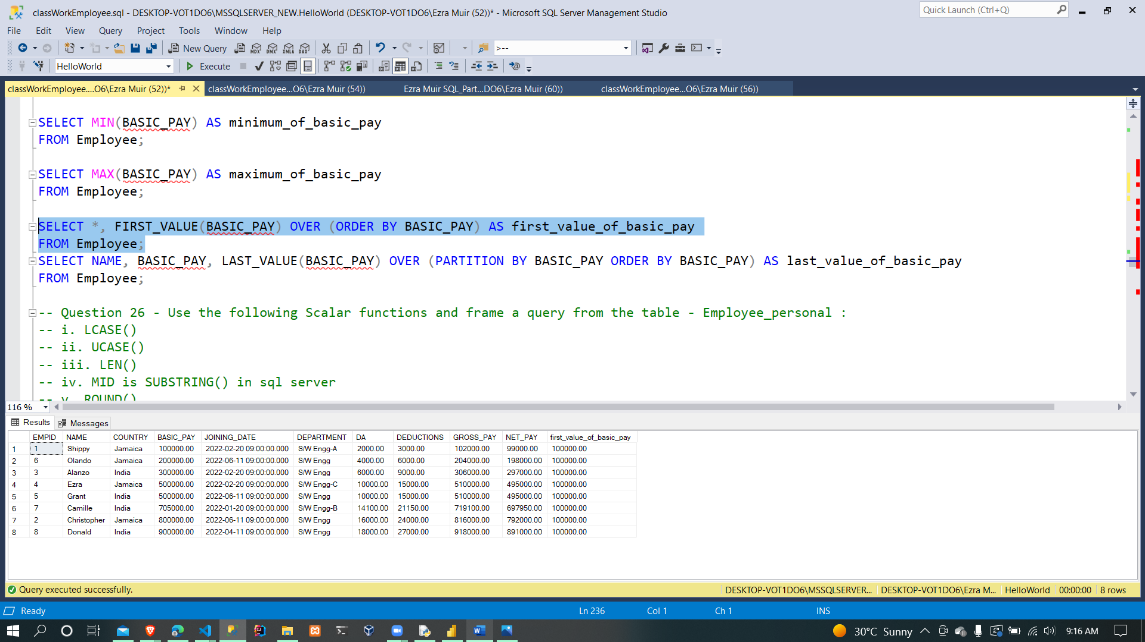
FROM Employee;

* + 1. [FIRST()](https://www.edureka.co/blog/sql-functions#first)

SELECT \*,

FIRST\_VALUE(BASIC\_PAY) OVER (ORDER BY BASIC\_PAY) AS first\_value\_of\_basic\_pay

FROM Employee;



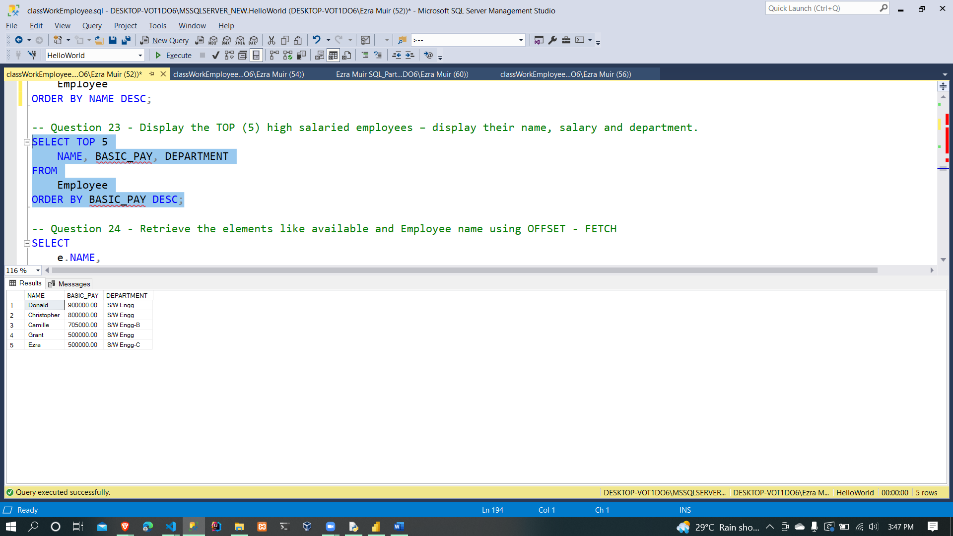
* + 1. [LAST()](https://www.edureka.co/blog/sql-functions#last)

SELECT NAME, BASIC\_PAY,

LAST\_VALUE(BASIC\_PAY) OVER (PARTITION BY BASIC\_PAY ORDER

BY BASIC\_PAY) AS last\_value\_of\_basic\_pay

FROM Employee;



1. Use the following Scalar functions and frame a query - Employee\_personal :
   * 1. [LCASE()](https://www.edureka.co/blog/sql-functions#lcase)

SELECT LOWER(NAME) AS upper\_case\_of\_name

FROM Employee\_personal;

* + 1. [UCASE()](https://www.edureka.co/blog/sql-functions#ucase)

SELECT UPPER(NAME) AS lower\_case\_of\_name

FROM Employee\_personal;

* + 1. [LEN()](https://www.edureka.co/blog/sql-functions#len)

SELECT NAME, LEN(NAME) AS length\_of\_name

FROM Employee\_personal;

* + 1. [MID()](https://www.edureka.co/blog/sql-functions#mid)

SELECT NAME, SUBSTRING(NAME, 1, 4) AS substring\_of\_name

FROM Employee\_personal;

* + 1. [ROUND()](https://www.edureka.co/blog/sql-functions#round)

SELECT AVG(e.BASIC\_PAY) AS average\_basic\_pay,

ROUND(AVG(e.BASIC\_PAY), 2) AS rounded\_basic\_pay

FROM Employee e

INNER JOIN Employee\_personal ep

ON e.NAME = ep.NAME;

* + 1. [NOW()](https://www.edureka.co/blog/sql-functions#now)

SELECT SYSDATETIME() AS current\_date\_time

FROM Employee\_personal

WHERE NAME = 'Ezra';

* + 1. [FORMAT()](https://www.edureka.co/blog/sql-functions#format)

SELECT FORMAT(SYSDATETIME(), 'dd-MM-yyyy hh:mm:ss') AS formatted\_date\_time

FROM Employee\_personal;

1. Update the following columns to the table

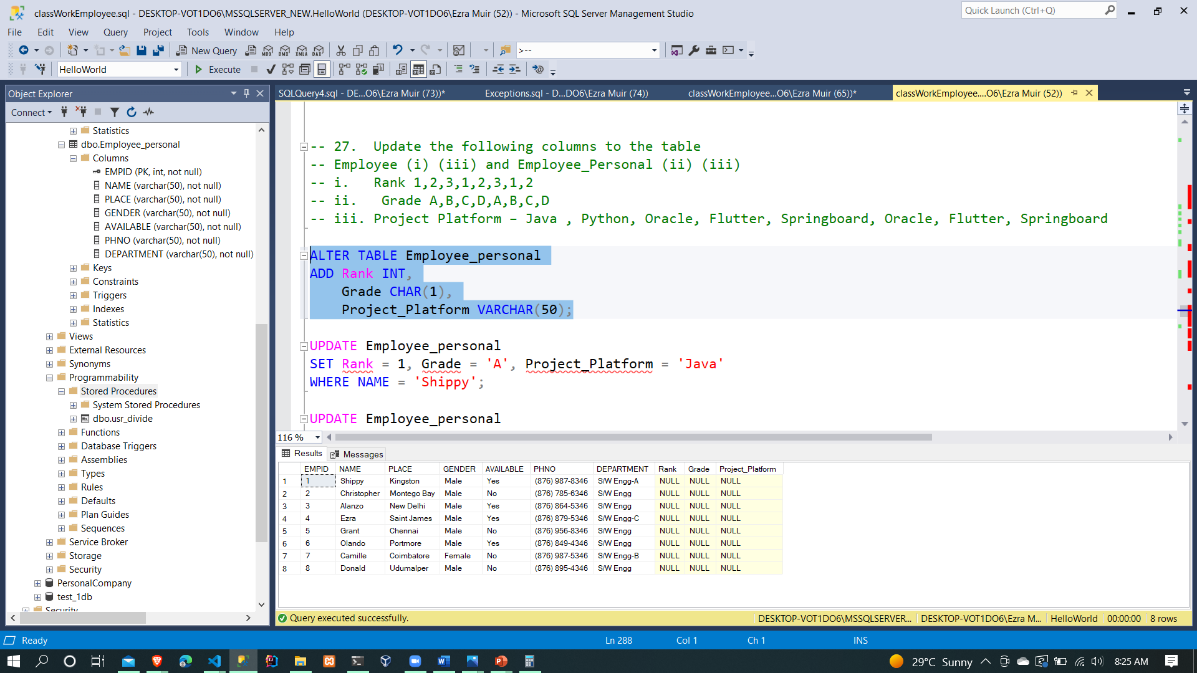
Employee (i) (iii) and Employee\_Personal (ii) (iii)

ALTER TABLE Employee\_personal

ADD Rank INT,

Grade CHAR(1),

Project\_Platform VARCHAR(50);



* + 1. Rank 1,2,3,1,2,3,1,2
    2. Grade A,B,C,D,A,B,C,D
    3. Project Platform – Java , Python, Oracle, Flutter, Springboard, Oracle, Flutter, Springboard

UPDATE Employee\_personal

SET Rank = 1, Grade = 'A', Project\_Platform = 'Java'

WHERE NAME = 'Shippy';

UPDATE Employee\_personal

SET Rank = 2, Grade = 'B', Project\_Platform = 'Python'

WHERE NAME = 'Christopher';

UPDATE Employee\_personal

SET Rank = 3, Grade = 'C', Project\_Platform = 'Oracle'

WHERE NAME = 'Alanzo';

UPDATE Employee\_personal

SET Rank = 4, Grade = 'D', Project\_Platform = 'Flutter'

WHERE NAME = 'Ezra';

UPDATE Employee\_personal

SET Rank = 5, Grade = 'E', Project\_Platform = 'Springboard'

WHERE NAME = 'Grant';

UPDATE Employee\_personal

SET Rank = 6, Grade = 'F', Project\_Platform = 'Oracle'

WHERE NAME = 'Olando';

UPDATE Employee\_personal

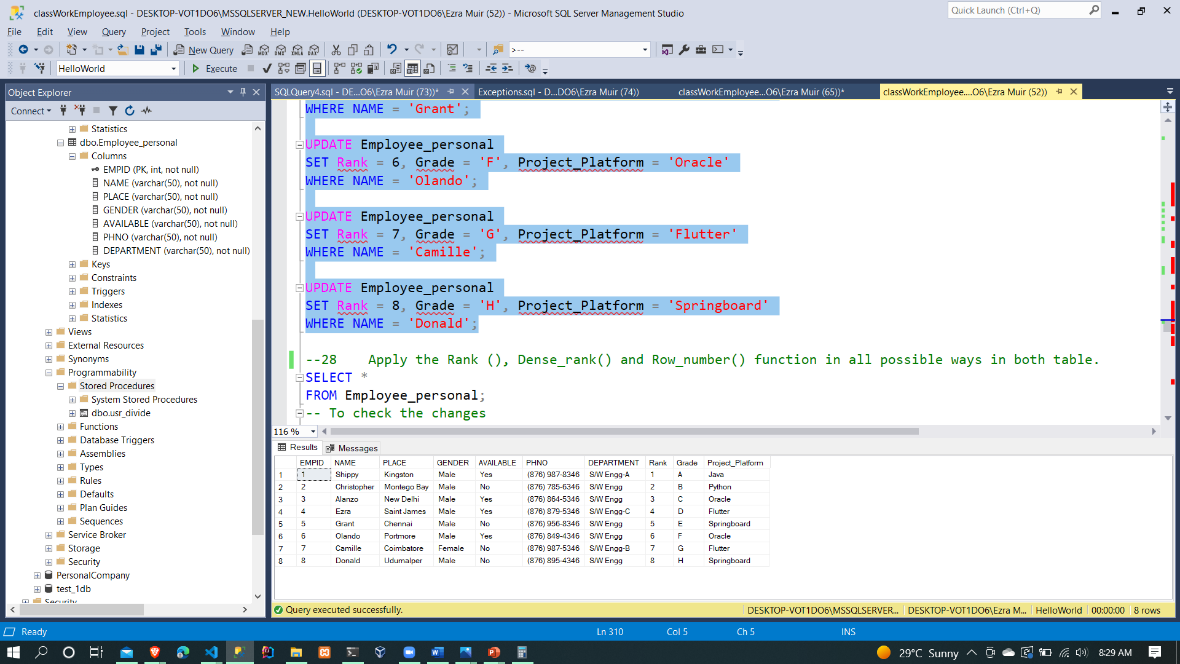
SET Rank = 7, Grade = 'G', Project\_Platform = 'Flutter'

WHERE NAME = 'Camille';

UPDATE Employee\_personal

SET Rank = 8, Grade = 'H', Project\_Platform = 'Springboard'

WHERE NAME = 'Donald';



1. Apply the Rank (), Dense\_rank() and Row\_number() function in all possible ways in both table.

-- RANK()

SELECT

NAME, Rank, Grade, Project\_Platform,

RANK() OVER (

ORDER BY Project\_Platform

) AS rank\_of\_project\_platform

FROM Employee\_personal;

-- RANK() with OUTER JOIN

SELECT

ep.NAME, Rank, Grade, Project\_Platform,

RANK() OVER (

PARTITION BY Project\_Platform

ORDER BY Project\_Platform

) AS rank\_of\_project\_platform

FROM Employee\_personal ep

FULL OUTER JOIN Employee e

ON ep.NAME = e.NAME;

-- DENSE\_RANK()

SELECT

NAME, Rank, Grade, Project\_Platform,

DENSE\_RANK() OVER (

ORDER BY Project\_Platform

) AS rank\_of\_project\_platform

FROM Employee\_personal;

-- ROW\_NUMBER()

SELECT

NAME, Rank, Grade, Project\_Platform, DEPARTMENT,

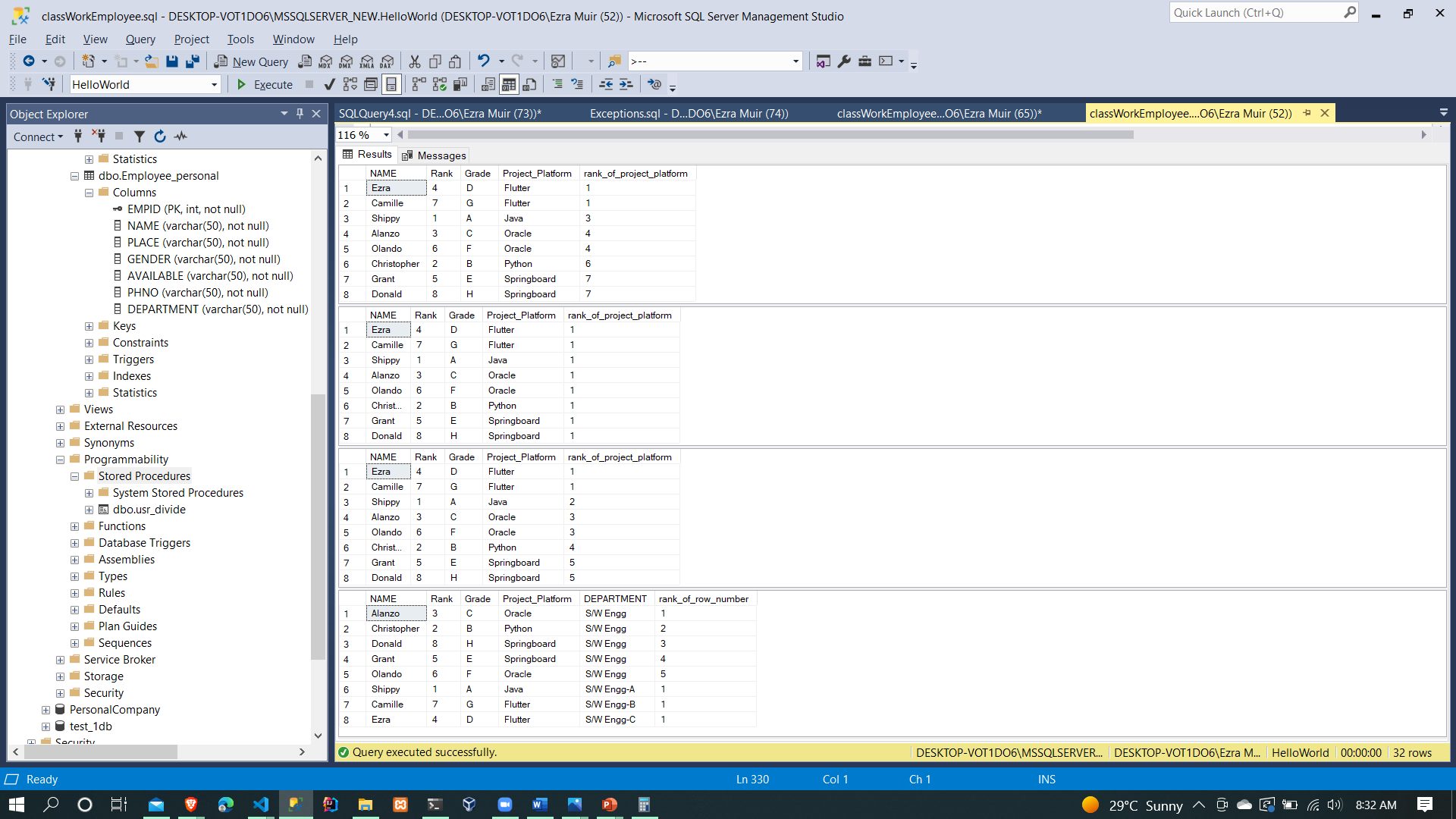
ROW\_NUMBER() OVER (

PARTITION BY DEPARTMENT

ORDER BY NAME

) AS rank\_of\_row\_number

FROM Employee\_personal;



1. Implement the following Set Operators in both the tables (write syntax and example with existing tables)
   * 1. UNION
     2. UNION ALL
     3. INTERSECT
     4. MINUS

-- i. UNION

SELECT

EMPID, NAME, DEPARTMENT

FROM Employee

UNION

SELECT

EMPID, NAME, DEPARTMENT

FROM Employee\_personal;

-- ii. UNION ALL

SELECT

EMPID, NAME, DEPARTMENT

FROM Employee

UNION ALL

SELECT

EMPID, NAME, DEPARTMENT

FROM Employee\_personal;

-- iii. INTERSECT

SELECT

EMPID, NAME, DEPARTMENT

FROM Employee

INTERSECT

SELECT

EMPID, NAME, DEPARTMENT

FROM Employee\_personal;

-- iv. MINUS (i.e. EXCEPT in sql server)

SELECT

EMPID, NAME, DEPARTMENT

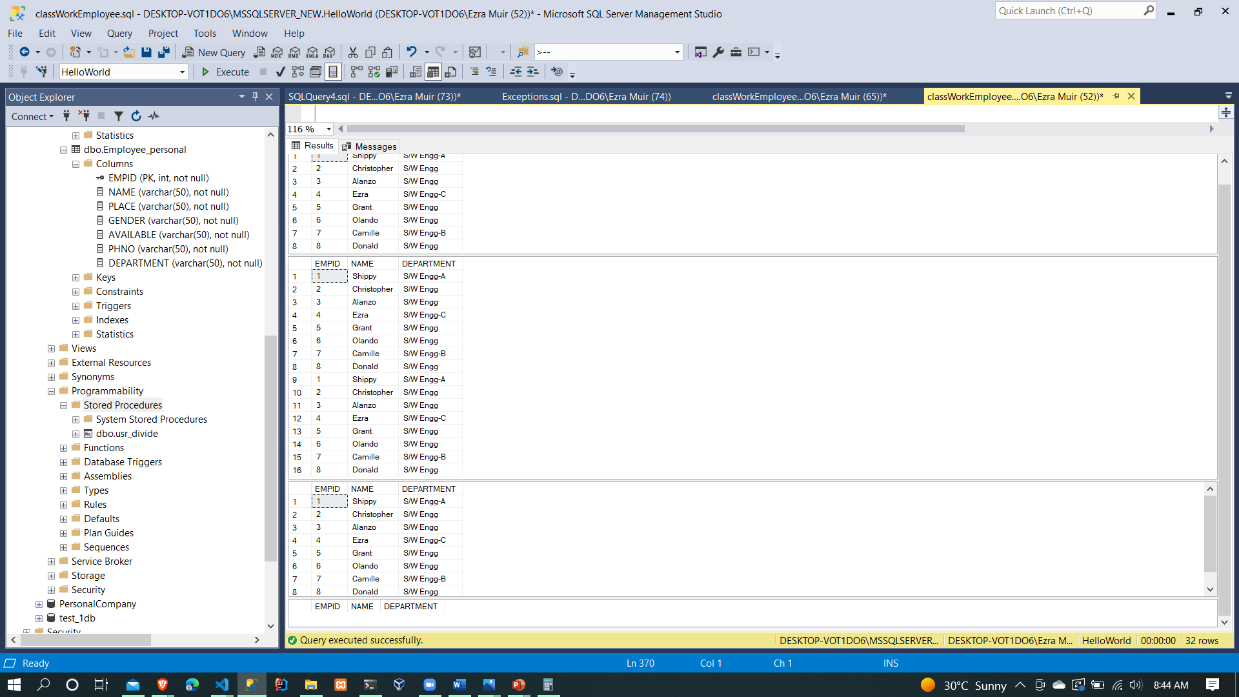
FROM Employee

EXCEPT

SELECT

EMPID, NAME, DEPARTMENT

FROM Employee\_personal;



1. Self-Learn topics 1:
   * 1. Data manipulation in a database
     2. Need for data manipulation
     3. Transaction
2. Self-Learn topics (write the concept, syntax and example with existing tables) 2:
   1. COMMIT & ROLLBACK statements
   2. Using the SAVEPOINT statement
   3. Automatic COMMIT behaviour in SQL
   4. MERGE statement
3. SQL Performance Tuning (write the concept, syntax and example with existing tables)
   * 1. Query Tuning by rewriting SQL’s
     2. Query Tuning by using Indexes Data dictionary views
4. Declare a variable Emp\_API\_Grade and assign the value “Yes” where rank is 1 or 2 and print the values Employee\_name and Department.
5. Declare the following variables and print the values:
   1. variable Emp\_dest\_place and assign the value “India” where rank is 1
   2. variable Emp\_wrk\_hrs and assign the value “5” where rank is 1.
   3. Self-Learn: Variable Types in SQL
6. Implement the following and form SQL Query.
   1. DECLARE
   2. SET
   3. SELECT
   4. PRINT
7. Self-Learn:
   1. Nested blocks – What? Scope?
   2. Types of blocks
   3. Structure
   4. Example using the employee table
   5. Nested loops
8. Using Nested blocks create a new column religion and add the following
   1. Create a variable and assign the values like a=”Islam”, b= “Hindu”, c= “Christian”
   2. Inside the nested blocks create the conditional statements and check the nation and allocate the religion.
   3. In next block count the number of people belongs to same religion.
   4. Print the results.
9. Read all data row by row with the help of the WHILE loop and display the table contents from table: Employee\_personal
10. Display the data from the table with the following condition: using WHILE loop
    1. The salary is greater than 300000
    2. The employee name while the employee gender is not null