



INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO



DATA BASE SELECTED TOPIC

Bocanegra Heziquio
Yestlanezi
No. Boleta: 201934009
3CV14

Ejercicios SQL Server Aerolineas

Profesora: Galeana Chávez Ing. María del Rosario

13 de octubre del 2023

1. Seleccionar el nombre de todos los tipos de aviones con más de 200 asientos.

```
SELECT TipoAvion FROM [TipoAvion] WHERE numAsientos > 200
```

SQLQuery1.sql - 5C...ezi Bocanegra (72))* X Ejercicios_DBST (2)...nezi Bocanegra (78))

```
SELECT TipoAvion FROM [TipoAvion] WHERE numAsientos > 200
```

100 %

Results Messages

| | TipoAvion |
|---|----------------|
| 1 | Airbus A320neo |
| 2 | A350 XWB |
| 3 | Tu-244 |

2. Seleccionar el nombre de todos los tipos de aviones con más de 200 asientos, con la siguiente leyenda: "El avión" <tipo avión> "tiene una capacidad de " <num asientos> "asientos".

```
SELECT 'El avión' + TipoAvion + 'tiene una capacidad de' +  
CAST(numAsientos AS varchar(10)) + 'asientos' AS leyenda  
FROM [TipoAvion]  
WHERE numAsientos > 200;
```

SQLQuery2.sql - 5C...ezi Bocanegra (54))* X SQLQuery1.sql - 5C...ezi Bocanegra (72)) Ejercicios_DBST (2)...nezi Bocanegra (78))

```
SELECT 'El avión' + TipoAvion + 'tiene una capacidad de' + CAST(numAsientos AS varchar(10)) + 'asientos' AS leyenda  
FROM [TipoAvion]  
WHERE numAsientos > 200;
```

100 %

Results Messages

| | leyenda |
|---|---|
| 1 | El aviónAirbus A320neotiene una capacidad de220a... |
| 2 | El aviónA350 XWBtiene una capacidad de300asientos |
| 3 | El aviónTu-244tiene una capacidad de269asientos |

3. Seleccionar el nombre, apellido paterno y apellido materno de todos los empleados de las aerolíneas.

```
SELECT p.nombre, p.paterno, p.materno  
  
FROM Empleado e  
  
JOIN Persona p ON e.idPersona = p.IdPersona  
  
JOIN Aerolinea a ON e.AerolineaId = a.AerolineaId;
```

SQLQuery3.sql - 5C...ezi Bocanegra (54)) * Ejercicios_DBST (2)...nezi Bocanegra (78))

```
SELECT p.nombre, p.paterno, p.materno  
FROM Empleado e  
JOIN Persona p ON e.idPersona = p.IdPersona  
JOIN Aerolinea a ON e.AerolineaId = a.AerolineaId;
```

| | nombre | paterno | materno |
|---|---------|-----------|-----------|
| 1 | Eduardo | Ortiz | NULL |
| 2 | Martín | Camona | Bautista |
| 3 | Victor | Sandoval | Escudero |
| 4 | Carlos | Tadeo | Almeida |
| 5 | Elsa | Fernández | Cabrera |
| 6 | Yuridia | Salazar | Figuerola |
| 7 | Edith | Parez | Granados |
| 8 | Beatriz | Pedraza | López |

4. Seleccionar el nombre, apellido paterno y apellido materno de todos los empleados de las aerolíneas concatenados en una sola columna con el alias nombre del empleado.

```
SELECT p.nombre + ' ' + p.paterno + ' ' + p.materno AS [nombre del empleado]  
FROM Empleado e  
JOIN Persona p ON e.idPersona = p.IdPersona  
JOIN Aerolinea a ON e.AerolineaId = a.AerolineaId;
```

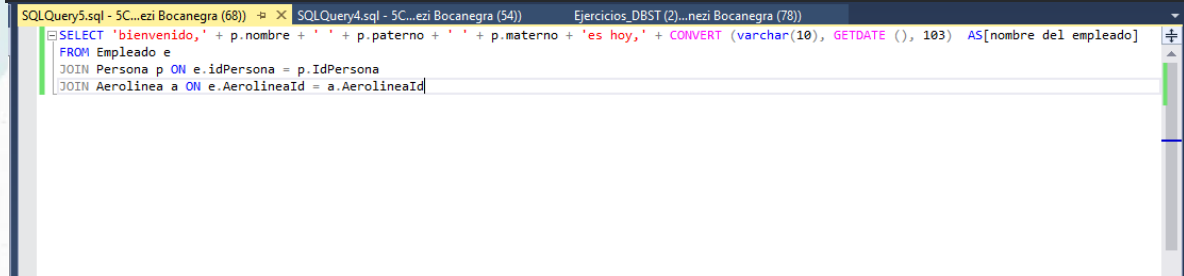
SQLQuery4.sql - 5C...ezi Bocanegra (54)) * Ejercicios_DBST (2)...nezi Bocanegra (78))

```
SELECT p.nombre + ' ' + p.paterno + ' ' + p.materno AS [nombre del empleado]  
FROM Empleado e  
JOIN Persona p ON e.idPersona = p.IdPersona  
JOIN Aerolinea a ON e.AerolineaId = a.AerolineaId;
```

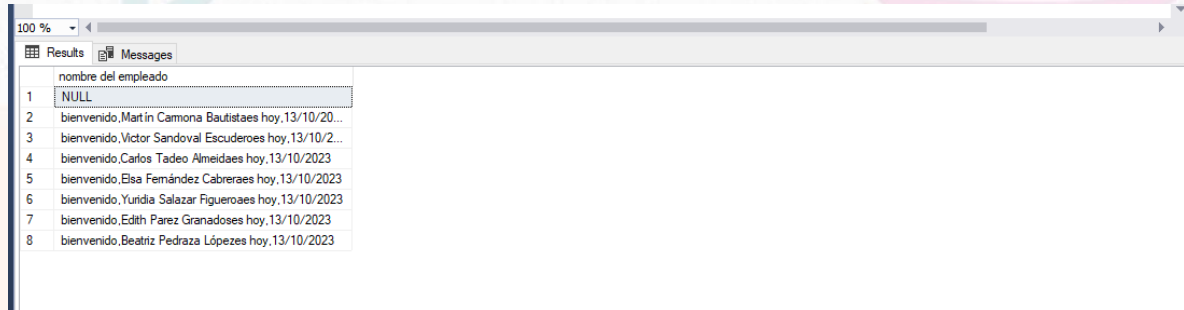
| | nombre del empleado |
|---|---------------------------|
| 1 | NULL |
| 2 | Martín Camona Bautista |
| 3 | Victor Sandoval Escudero |
| 4 | Carlos Tadeo Almeida |
| 5 | Elsa Fernández Cabrera |
| 6 | Yuridia Salazar Figuerola |
| 7 | Edith Parez Granados |
| 8 | Beatriz Pedraza López |

5. Seleccionar el nombre, apellido paterno y apellido materno de todos los empleados de las aerolíneas con la siguiente leyenda "bienvenido, " <nombre completo de la persona> ", hoy es " <fecha actual>

```
SELECT 'bienvenido,' + p.nombre + ' ' + p.paterno + ' ' + p.materno + 'es  
hoy,' + CONVERT (varchar(10), GETDATE (), 103) AS[nombre del empleado]  
FROM Empleado e  
JOIN Persona p ON e.idPersona = p.IdPersona  
JOIN Aerolinea a ON e.AerolineaId = a.AerolineaId
```



The screenshot shows the SQL Server Enterprise Manager interface. The 'SQLQuery5.sql' file is open, displaying the same SQL query as above. The query is: `SELECT 'bienvenido,' + p.nombre + ' ' + p.paterno + ' ' + p.materno + 'es hoy,' + CONVERT (varchar(10), GETDATE (), 103) AS[nombre del empleado] FROM Empleado e JOIN Persona p ON e.idPersona = p.IdPersona JOIN Aerolinea a ON e.AerolineaId = a.AerolineaId`

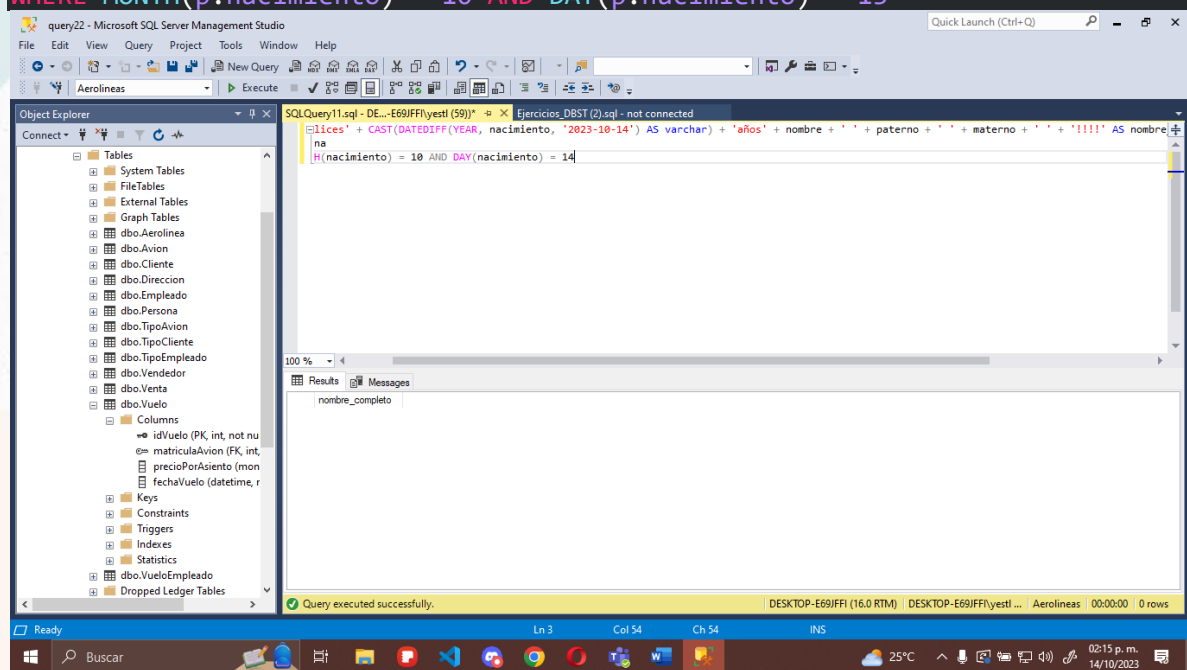


The screenshot shows the 'Results' pane of the SQL Server Enterprise Manager. It displays the output of the query, which is a list of 8 rows. The first row is NULL, and the subsequent 7 rows show the concatenated string 'bienvenido,' followed by the employee's name, 'es hoy,' followed by the current date in YYYY/MM/DD format.

| | nombre del empleado |
|---|--|
| 1 | NULL |
| 2 | bienvenido,Martin Camona Bautistaes hoy,13/10/20... |
| 3 | bienvenido,Victor Sandoval Escuderoes hoy,13/10/2... |
| 4 | bienvenido,Carlos Tadeo Almeidaes hoy,13/10/2023 |
| 5 | bienvenido,Elaa Fernández Cabreraes hoy,13/10/2023 |
| 6 | bienvenido,Yuridia Salazar Figueroaes hoy,13/10/2023 |
| 7 | bienvenido,Edith Perez Granadoses hoy,13/10/2023 |
| 8 | bienvenido,Beatriz Pedraza Lópezes hoy,13/10/2023 |

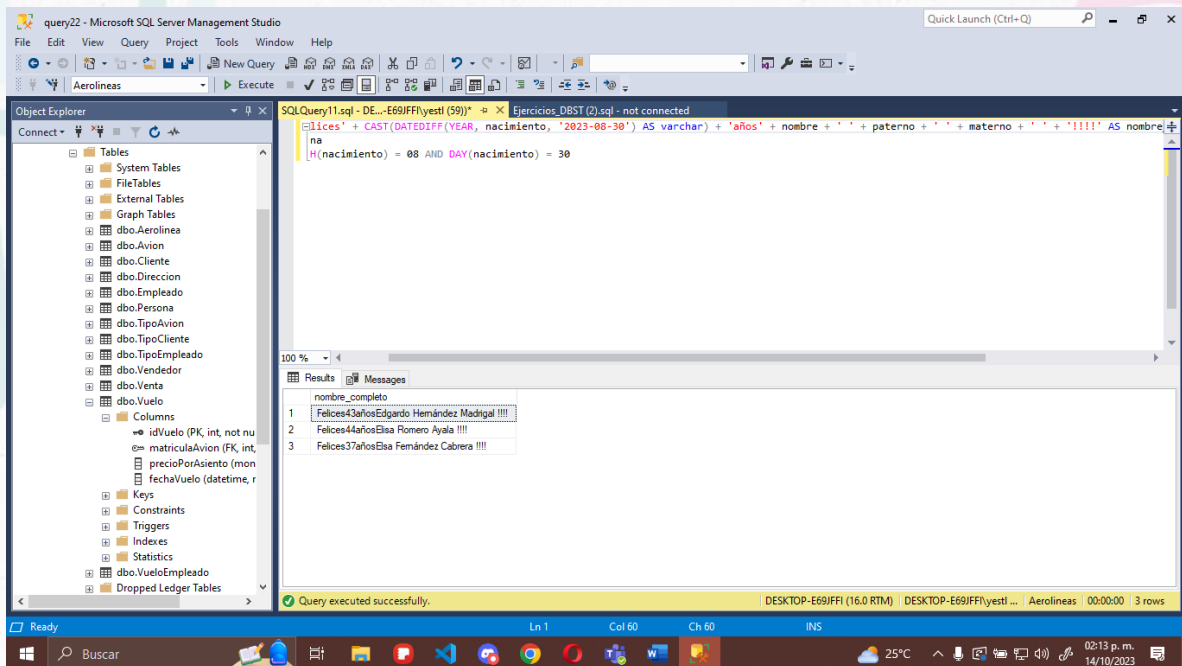
6. Seleccionar el nombre completo de todas las personas en la base de datos que cumpleaños el 21/04/2023 y mostrarlo como "Felices " <edad> " años " <nombre completo> "!!!"

```
SELECT 'Felices,' + CAST(DATEDIFF(YEAR, p.nacimiento, '2023-10-13') AS
varchar) + 'años,' + p.nombre + ' ' + p.paterno + ' ' + p.materno + '!!!'
AS nombre_completo
FROM Persona p
WHERE MONTH(p.nacimiento) = 10 AND DAY(p.nacimiento) = 13
```



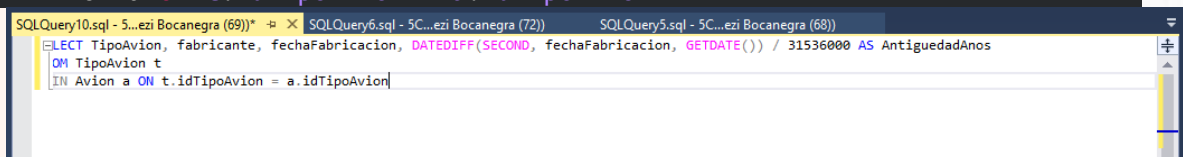
The screenshot shows the Microsoft SQL Server Enterprise Edition interface. The 'Object Explorer' on the left lists the database structure, including tables like 'dbo.Aerolinea', 'dbo.Avion', 'dbo.Cliente', 'dbo.Direccion', 'dbo.Emppleado', 'dbo.Persona', 'dbo.TipoAvion', 'dbo.TipoCliente', 'dbo.TipoEmpleado', 'dbo.Vendedor', 'dbo.Venta', and 'dbo.Vuelo'. The 'Query Editor' window displays the SQL query. The 'Results' pane at the bottom shows a single row with the value 'na' under the column 'nombre_completo'. A status bar at the bottom indicates 'Query executed successfully.' and '0 rows'.

Debido a que no hay nadie que cumpla años el día de hoy, la consulta no muestra un resultado, así que decidí probar la consulta, pero con otra fecha que, si estuviera en la base de datos, solo para comprobar que funcionaba.



7. mostrar la antigüedad en años de todos los aviones almacenados en la base de datos

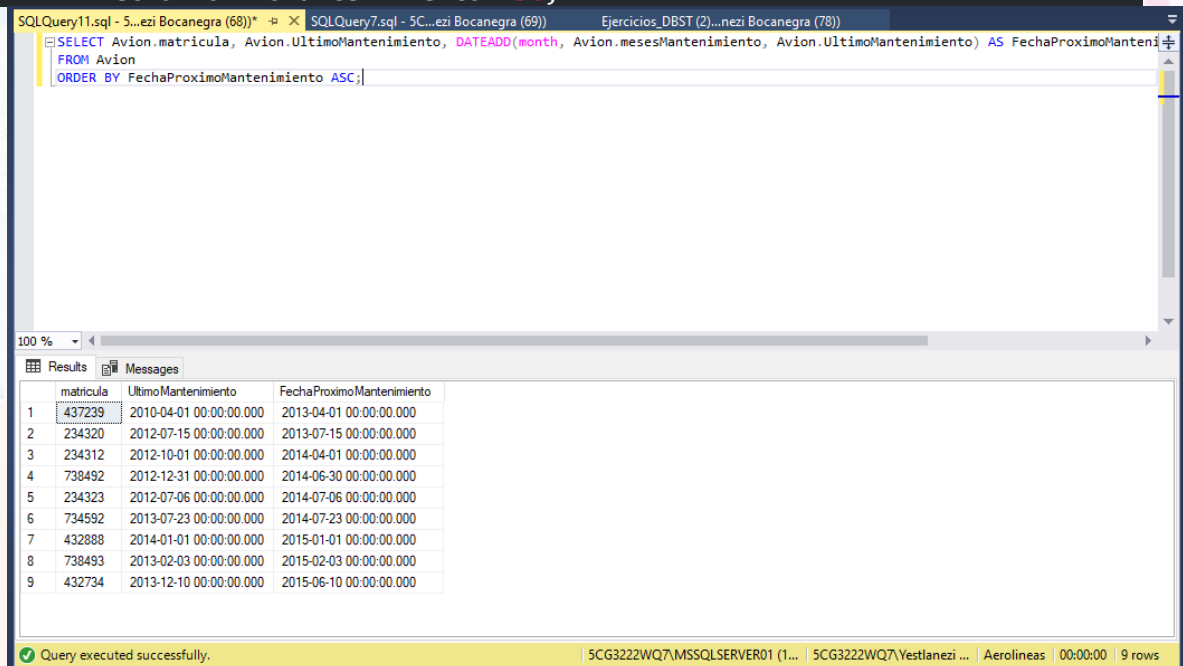
```
SELECT TipoAvion, fabricante, fechaFabricacion, DATEDIFF(SECOND,
fechaFabricacion, GETDATE()) / 31536000 AS AntiguedadAnos
FROM TipoAvion t
JOIN Avion a ON t.idTipoAvion = a.idTipoAvion
```



| | TipoAvion | fabricante | fechaFabricacion | AntiguedadAnos |
|---|------------------------|---------------------------------|-------------------------|----------------|
| 1 | Bombardier BD-500-1A10 | Bombardier Aerospace | 2010-04-01 00:00:00.000 | 13 |
| 2 | MRJ 90ER | Mitsubishi Aircraft Corporation | 2011-07-15 00:00:00.000 | 12 |
| 3 | Airbus A320neo | Airbus | 2010-07-06 00:00:00.000 | 13 |
| 4 | Boeing 737 MAX 9 | Boeing Commercial Airplanes | 2013-12-10 00:00:00.000 | 9 |
| 5 | Boeing 737 MAX 9 | Boeing Commercial Airplanes | 2014-01-01 00:00:00.000 | 9 |
| 6 | A350 XWB | Airbus | 2010-04-01 00:00:00.000 | 13 |
| 7 | MS-21-200 | United Aircraft Corporation | 2012-07-23 00:00:00.000 | 11 |
| 8 | Tu-244 | Tupolev | 2011-01-01 00:00:00.000 | 12 |
| 9 | COMAC C919 | COMAC | 2010-02-03 00:00:00.000 | 13 |

8. mostrar la fecha exacta del siguiente mantenimiento de los aviones tomando como referencia la fecha del último, ordenándolos por el que esté más próximo a vencer

```
SELECT Avion.matricula, Avion.UltimoMantenimiento, DATEADD(month,  
Avion.mesesMantenimiento, Avion.UltimoMantenimiento) AS  
FechaProximoMantenimiento  
FROM Avion  
ORDER BY FechaProximoMantenimiento ASC;
```



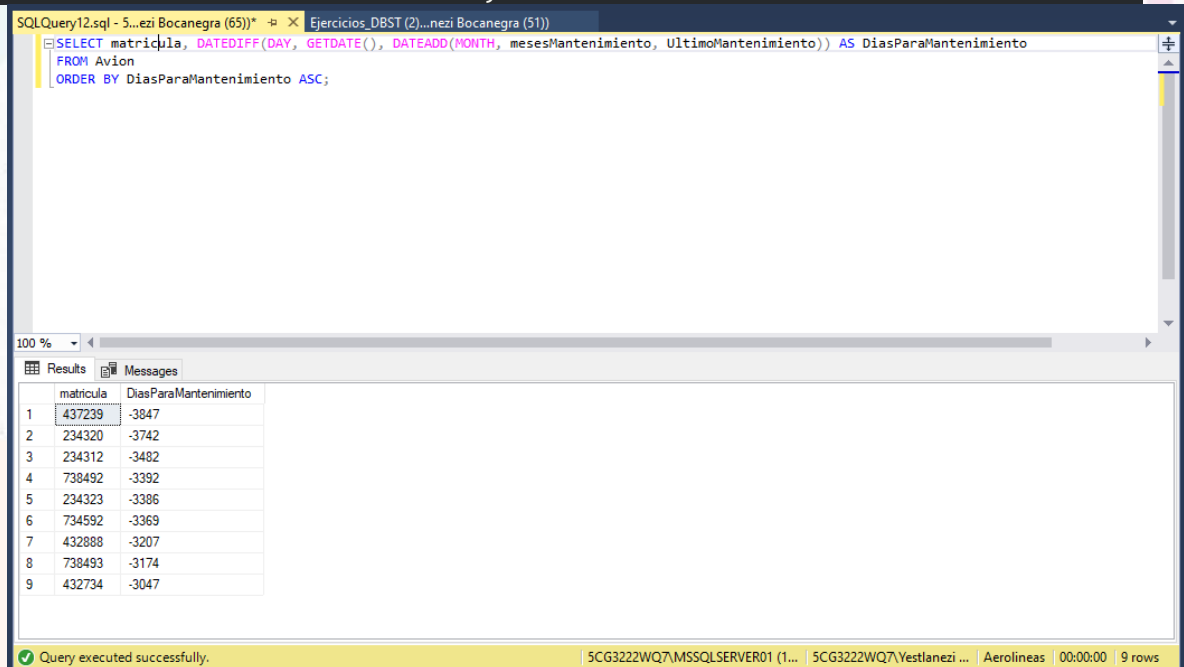
The screenshot shows a SQL Server Enterprise Manager window with a query executed successfully. The query is displayed in the top pane, and the results are shown in the bottom pane. The results table has three columns: matricula, UltimoMantenimiento, and FechaProximoMantenimiento. The data is ordered by FechaProximoMantenimiento in ascending order.

| | matricula | UltimoMantenimiento | FechaProximoMantenimiento |
|---|-----------|-------------------------|---------------------------|
| 1 | 437239 | 2010-04-01 00:00:00.000 | 2013-04-01 00:00:00.000 |
| 2 | 234320 | 2012-07-15 00:00:00.000 | 2013-07-15 00:00:00.000 |
| 3 | 234312 | 2012-10-01 00:00:00.000 | 2014-04-01 00:00:00.000 |
| 4 | 738492 | 2012-12-31 00:00:00.000 | 2014-06-30 00:00:00.000 |
| 5 | 234323 | 2012-07-06 00:00:00.000 | 2014-07-06 00:00:00.000 |
| 6 | 734592 | 2013-07-23 00:00:00.000 | 2014-07-23 00:00:00.000 |
| 7 | 432888 | 2014-01-01 00:00:00.000 | 2015-01-01 00:00:00.000 |
| 8 | 738493 | 2013-02-03 00:00:00.000 | 2015-02-03 00:00:00.000 |
| 9 | 432734 | 2013-12-10 00:00:00.000 | 2015-06-10 00:00:00.000 |

Query executed successfully. | 5CG3222WQ7\MSSQLSERVER01 (1... | 5CG3222WQ7\Vestlanezi ... | Aerolineas | 00:00:00 | 9 rows

9. mostrar el tiempo en días que falta para el mantenimiento de los aviones tomando como referencia la fecha del último, ordenándolos por el que esté más próximo a vencer

```
SELECT matricula, DATEDIFF(DAY, GETDATE(), DATEADD(MONTH, mesesMantenimiento, UltimoMantenimiento)) AS DiasParaMantenimiento
FROM Avion
ORDER BY DiasParaMantenimiento ASC;
```



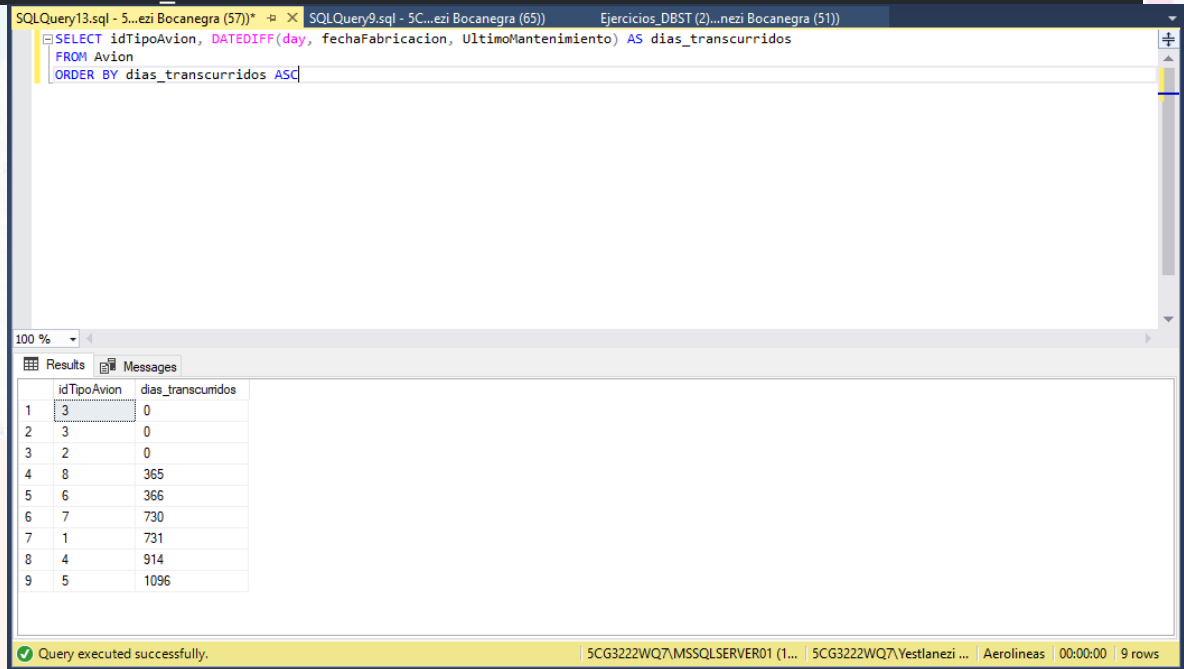
The screenshot shows a SQL Server Enterprise Manager window with a query executed successfully. The query is displayed in the SQL Query window, and the results are shown in the Results pane. The Results pane displays a table with two columns: 'matricula' and 'DiasParaMantenimiento'. The table contains 9 rows of data, ordered by 'DiasParaMantenimiento' in ascending order.

| | matricula | DiasParaMantenimiento |
|---|-----------|-----------------------|
| 1 | 437239 | -3847 |
| 2 | 234320 | -3742 |
| 3 | 234312 | -3482 |
| 4 | 738492 | -3392 |
| 5 | 234323 | -3386 |
| 6 | 734592 | -3369 |
| 7 | 432888 | -3207 |
| 8 | 738493 | -3174 |
| 9 | 432734 | -3047 |

Query executed successfully. 5CG3222WQ7\MSSQLSERVER01 (1... 5CG3222WQ7\Vestlanezi ... Aerolineas 00:00:00 9 rows

10. mostrar el tiempo en días por el que se atrasó o adelantó el último mantenimiento de los aviones tomando como referencia la fecha de adquisición.

```
SELECT idTipoAvion, DATEDIFF(day, fechaFabricacion, UltimoMantenimiento) AS
dias_transcurridos
FROM Avion
ORDER BY dias_transcurridos ASC
```



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

```
SELECT idTipoAvion, DATEDIFF(day, fechaFabricacion, UltimoMantenimiento) AS dias_transcurridos
FROM Avion
ORDER BY dias_transcurridos ASC
```

The results pane displays the following data:

| | idTipoAvion | dias_transcurridos |
|---|-------------|--------------------|
| 1 | 3 | 0 |
| 2 | 3 | 0 |
| 3 | 2 | 0 |
| 4 | 8 | 365 |
| 5 | 6 | 366 |
| 6 | 7 | 730 |
| 7 | 1 | 731 |
| 8 | 4 | 914 |
| 9 | 5 | 1096 |

The status bar at the bottom indicates: "Query executed successfully. 5CG3222WQ7\MSSQLSERVER01 (1... 5CG3222WQ7\Yestlanezi ... Aerolineas 00:00:00 9 rows

11. mostrar todos los vuelos y ordenarlos por los que tengan estén más próximos y tengan más lugares vacíos

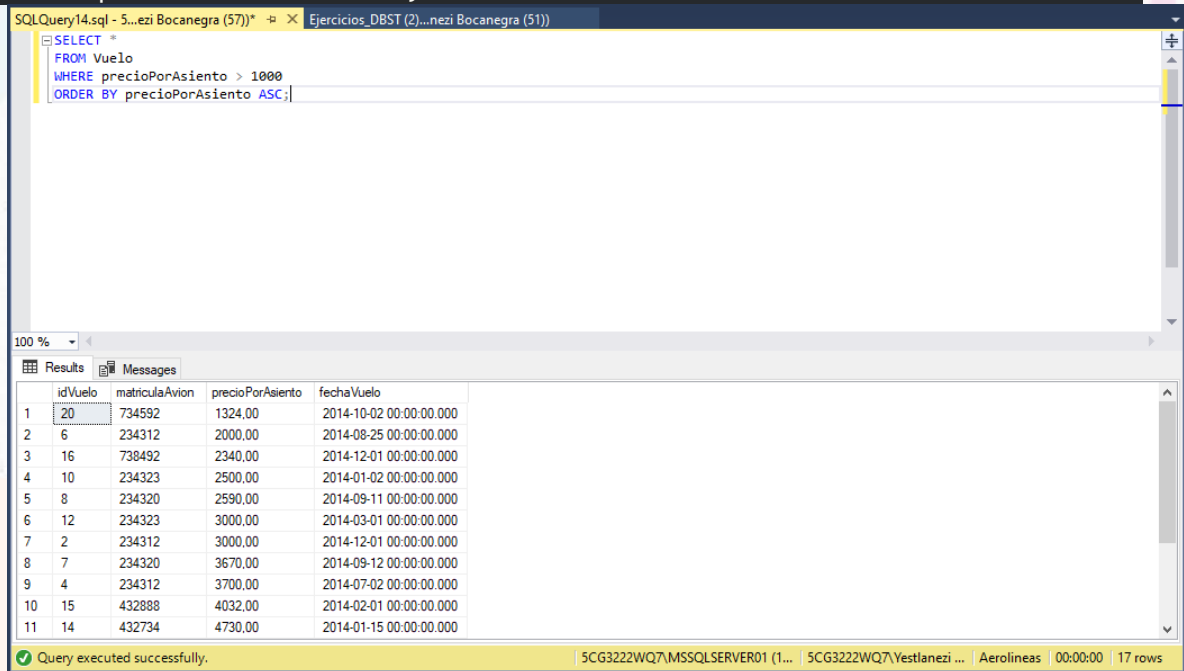
```
SELECT Vuelo.idVuelo, TIPOAVION.NUMASIENTOS -  
SUM(ISNULL(Venta.numAsientos,0) ) ASIENTOS, Vuelo.fechaVuelo  
FROM VUELO left JOIN AVION ON VUELO.matriculaAvion = AVION.MATRICULA left  
JOIN TIPOAVION ON AVION.IDTIPOAVION =  
TIPOAVION.IDTIPOAVION left JOIN VENTA ON VENTA.idVuelo = VUELO.idVuelo  
group by Vuelo.idVuelo, Vuelo.fechaVuelo,  
TIPOAVION.NUMASIENTOS  
order by ASIENTOS DESC , fechaVuelo asc ;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure, including tables like dbo.Aerolineas, dbo.Avion, and dbo.Venta. The central query window shows the SQL query from the previous block. The Results pane at the bottom displays the output of the query, which is a table with three columns: idVuelo, ASIENTOS, and fechaVuelo. The results are ordered by ASIENTOS in descending order and then by fechaVuelo in ascending order.

| idVuelo | ASIENTOS | fechaVuelo |
|---------|----------|-------------------------|
| 16 | 269 | 2014-12-01 00:00:00.000 |
| 17 | 269 | 2014-12-30 00:00:00.000 |
| 18 | 269 | 2014-12-31 00:00:00.000 |
| 10 | 218 | 2014-01-02 00:00:00.000 |
| 12 | 215 | 2014-03-01 00:00:00.000 |
| 14 | 178 | 2014-01-15 00:00:00.000 |
| 13 | 176 | 2014-01-01 00:00:00.000 |
| 15 | 163 | 2014-02-01 00:00:00.000 |
| 19 | 147 | 2014-08-20 00:00:00.000 |
| 20 | 147 | 2014-10-02 00:00:00.000 |
| 4 | 100 | 2014-07-02 00:00:00.000 |

12. mostrar los vuelos que tengan un precio por asiento mayor a 1000 pesos, ordenarlos de manera ascendente

```
SELECT *  
FROM Vuelo  
WHERE precioPorAsiento > 1000  
ORDER BY precioPorAsiento ASC;
```



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

```
SELECT *  
FROM Vuelo  
WHERE precioPorAsiento > 1000  
ORDER BY precioPorAsiento ASC;
```

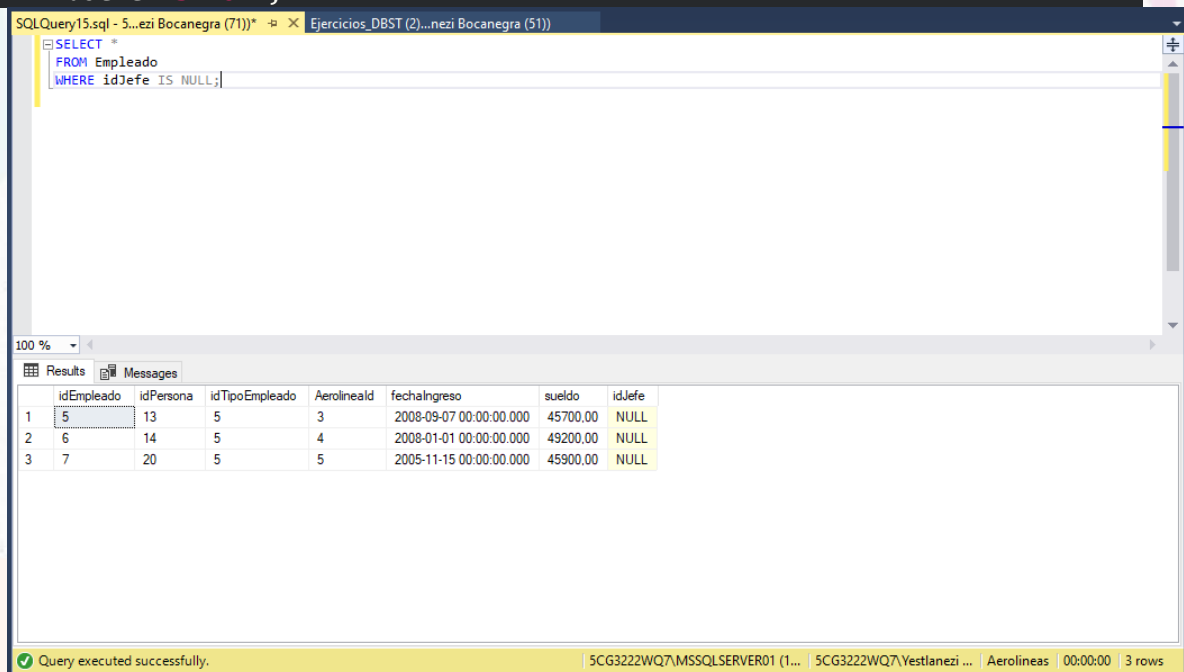
The results pane displays 17 rows of data. The first row is highlighted. The columns are: idVuelo, matriculaAvion, precioPorAsiento, and fechaVuelo.

| | idVuelo | matriculaAvion | precioPorAsiento | fechaVuelo |
|----|---------|----------------|------------------|-------------------------|
| 1 | 20 | 734592 | 1324,00 | 2014-10-02 00:00:00.000 |
| 2 | 6 | 234312 | 2000,00 | 2014-08-25 00:00:00.000 |
| 3 | 16 | 738492 | 2340,00 | 2014-12-01 00:00:00.000 |
| 4 | 10 | 234323 | 2500,00 | 2014-01-02 00:00:00.000 |
| 5 | 8 | 234320 | 2590,00 | 2014-09-11 00:00:00.000 |
| 6 | 12 | 234323 | 3000,00 | 2014-03-01 00:00:00.000 |
| 7 | 2 | 234312 | 3000,00 | 2014-12-01 00:00:00.000 |
| 8 | 7 | 234320 | 3670,00 | 2014-09-12 00:00:00.000 |
| 9 | 4 | 234312 | 3700,00 | 2014-07-02 00:00:00.000 |
| 10 | 15 | 432888 | 4032,00 | 2014-02-01 00:00:00.000 |
| 11 | 14 | 432734 | 4730,00 | 2014-01-15 00:00:00.000 |

The status bar at the bottom indicates: Query executed successfully. | 5CG3222WQ7\MSSQLSERVER01 (1... | 5CG3222WQ7\Vestlanezi ... | Aerolineas | 00:00:00 | 17 rows

13. Mostrar todos los empleados que no tengan jefe

```
SELECT *  
FROM Empleado  
WHERE idJefe IS NULL;
```



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

```
SELECT *  
FROM Empleado  
WHERE idJefe IS NULL;
```

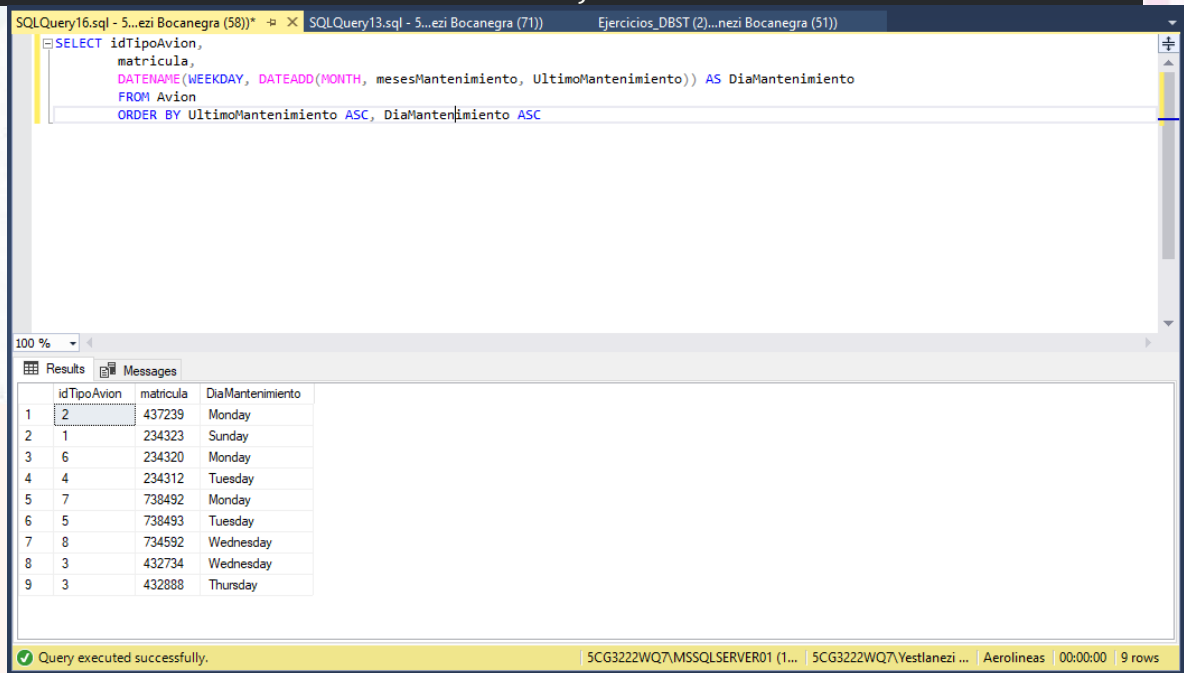
The results pane displays a table with 7 columns: idEmpleado, idPersona, idTipoEmpleado, Aerolineaid, fechaIngreso, sueldo, and idJefe. The table contains 3 rows of data, all with NULL values in the idJefe column.

| | idEmpleado | idPersona | idTipoEmpleado | Aerolineaid | fechaIngreso | sueldo | idJefe |
|---|------------|-----------|----------------|-------------|-------------------------|----------|--------|
| 1 | 5 | 13 | 5 | 3 | 2008-09-07 00:00:00.000 | 45700.00 | NULL |
| 2 | 6 | 14 | 5 | 4 | 2008-01-01 00:00:00.000 | 49200.00 | NULL |
| 3 | 7 | 20 | 5 | 5 | 2005-11-15 00:00:00.000 | 45900.00 | NULL |

At the bottom of the window, a status bar indicates: "Query executed successfully. 5CG3222WQ7\MSSQLSERVER01 (1... 5CG3222WQ7\Yestlanezi ... Aerolineas 00:00:00 3 rows".

14. mostrar el día de la semana en que le toca mantenimiento a cada uno de los aviones

```
SELECT idTipoAvion,  
       matricula,  
       DATENAME(WEEKDAY, DATEADD(MONTH, mesesMantenimiento,  
UltimoMantenimiento)) AS DiaMantenimiento  
FROM Avion  
ORDER BY UltimoMantenimiento ASC, DiaMantenimiento ASC
```



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

```
SELECT idTipoAvion,  
       matricula,  
       DATENAME(WEEKDAY, DATEADD(MONTH, mesesMantenimiento, UltimoMantenimiento)) AS DiaMantenimiento  
FROM Avion  
ORDER BY UltimoMantenimiento ASC, DiaMantenimiento ASC
```

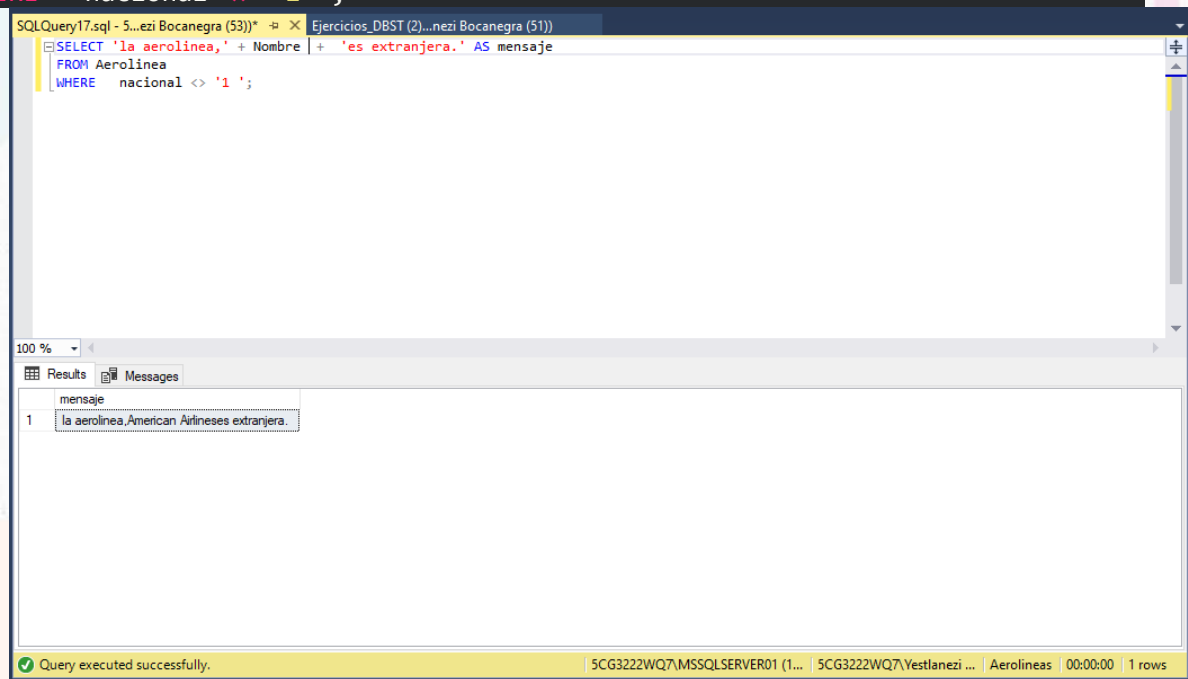
The results pane displays the following data:

| | idTipoAvion | matricula | DiaMantenimiento |
|---|-------------|-----------|------------------|
| 1 | 2 | 437239 | Monday |
| 2 | 1 | 234323 | Sunday |
| 3 | 6 | 234320 | Monday |
| 4 | 4 | 234312 | Tuesday |
| 5 | 7 | 738492 | Monday |
| 6 | 5 | 738493 | Tuesday |
| 7 | 8 | 734592 | Wednesday |
| 8 | 3 | 432734 | Wednesday |
| 9 | 3 | 432888 | Thursday |

The status bar at the bottom indicates: Query executed successfully. | 5CG3222WQ7\MSSQLSERVER01 (1... | 5CG3222WQ7\Yestlanezi ... | Aerolineas | 00:00:00 | 9 rows

15. Muestra las aerolíneas extranjeras, con el texto, la aerolínea <nombre> es extranjera

```
SELECT 'la aerolinea,' + Nombre + 'es extranjera.' AS mensaje
FROM Aerolinea
WHERE nacional <> '1';
```



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

```
SELECT 'la aerolinea,' + Nombre + 'es extranjera.' AS mensaje
FROM Aerolinea
WHERE nacional <> '1';
```

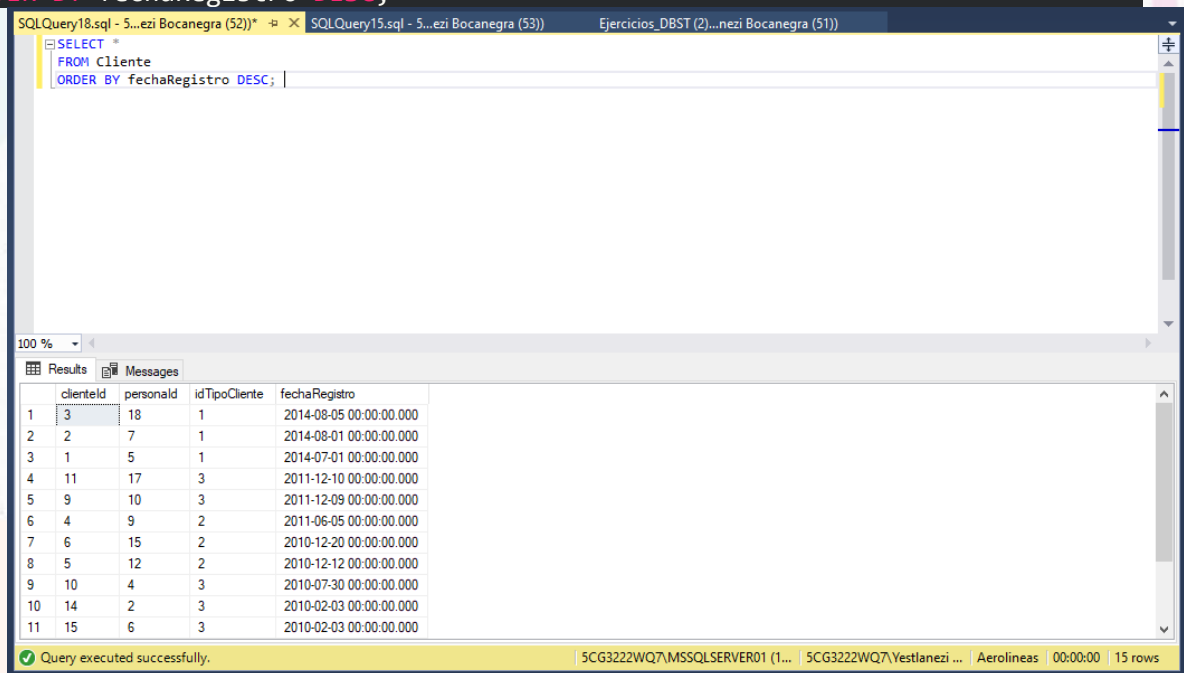
The results pane shows a single row with the following data:

| mensaje |
|--|
| la aerolinea,American Airlineses extranjera. |

The status bar at the bottom indicates: Query executed successfully. 5CG3222WQ7\MSSQLSERVER01 (1... 5CG3222WQ7\Yestlanezi ... Aerolineas 00:00:00 1 rows

16. Selecciona la tabla de clientes ordenándola por la fecha de registro de manera descendente

```
SELECT *  
FROM Cliente  
ORDER BY fechaRegistro DESC;
```

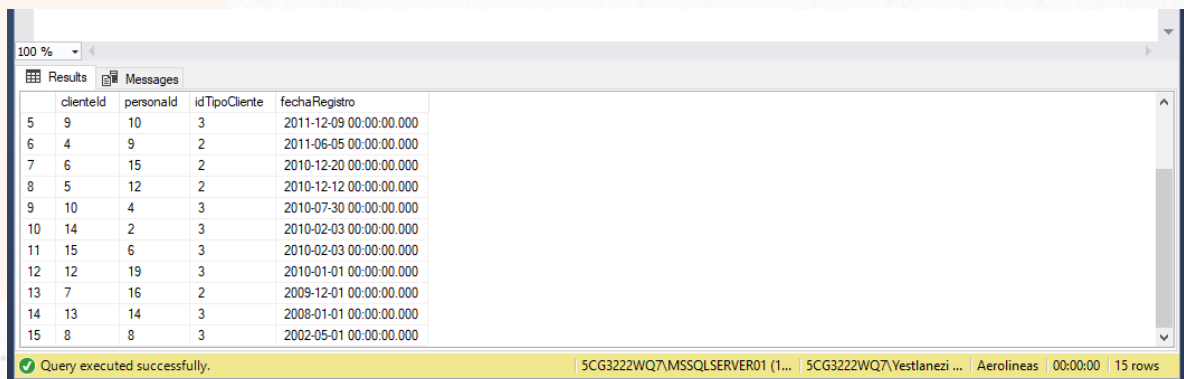


SQLQuery18.sql - 5...ezi Bocanegra (52))* SQLQuery15.sql - 5...ezi Bocanegra (53)) Ejercicios_DBST (2)...nezi Bocanegra (51))

```
SELECT *  
FROM Cliente  
ORDER BY fechaRegistro DESC;
```

| | clienteld | personald | idTipoCliente | fechaRegistro |
|----|-----------|-----------|---------------|-------------------------|
| 1 | 3 | 18 | 1 | 2014-08-05 00:00:00.000 |
| 2 | 2 | 7 | 1 | 2014-08-01 00:00:00.000 |
| 3 | 1 | 5 | 1 | 2014-07-01 00:00:00.000 |
| 4 | 11 | 17 | 3 | 2011-12-10 00:00:00.000 |
| 5 | 9 | 10 | 3 | 2011-12-09 00:00:00.000 |
| 6 | 4 | 9 | 2 | 2011-06-05 00:00:00.000 |
| 7 | 6 | 15 | 2 | 2010-12-20 00:00:00.000 |
| 8 | 5 | 12 | 2 | 2010-12-12 00:00:00.000 |
| 9 | 10 | 4 | 3 | 2010-07-30 00:00:00.000 |
| 10 | 14 | 2 | 3 | 2010-02-03 00:00:00.000 |
| 11 | 15 | 6 | 3 | 2010-02-03 00:00:00.000 |

Query executed successfully. 5CG3222WQ7\MSSQLSERVER01 (1... 5CG3222WQ7\Vestlanezi ... Aerolíneas 00:00:00 15 rows



SQLQuery18.sql - 5...ezi Bocanegra (52))* SQLQuery15.sql - 5...ezi Bocanegra (53)) Ejercicios_DBST (2)...nezi Bocanegra (51))

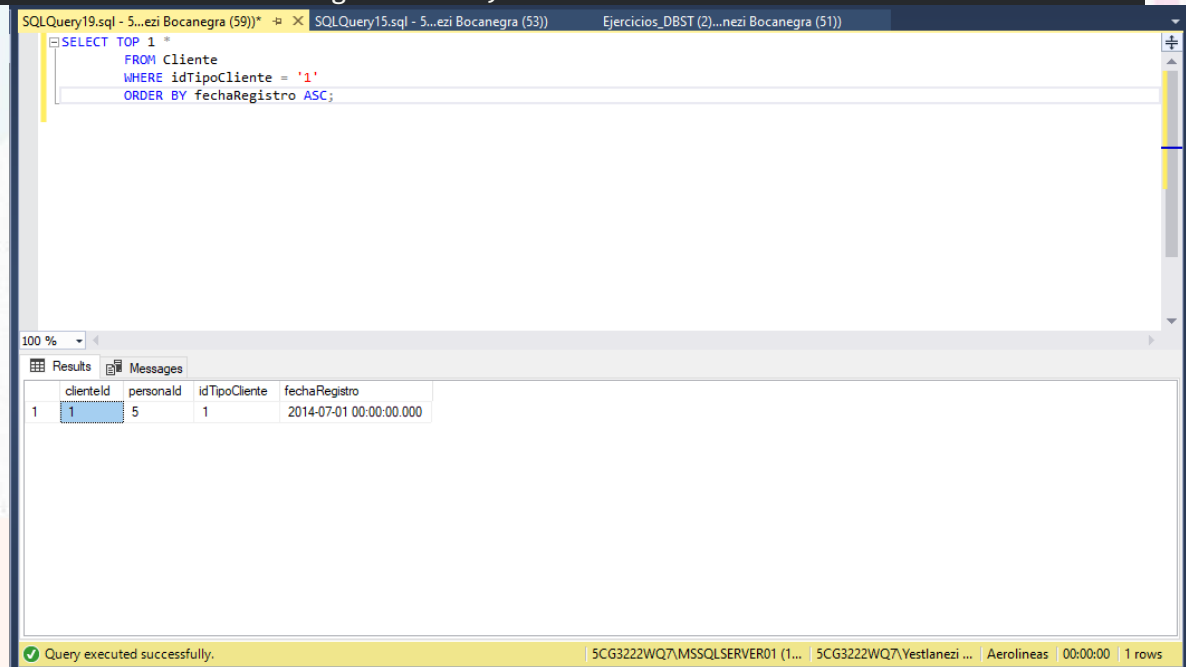
```
SELECT *  
FROM Cliente  
ORDER BY fechaRegistro DESC;
```

| | clienteld | personald | idTipoCliente | fechaRegistro |
|----|-----------|-----------|---------------|-------------------------|
| 5 | 9 | 10 | 3 | 2011-12-09 00:00:00.000 |
| 6 | 4 | 9 | 2 | 2011-06-05 00:00:00.000 |
| 7 | 6 | 15 | 2 | 2010-12-20 00:00:00.000 |
| 8 | 5 | 12 | 2 | 2010-12-12 00:00:00.000 |
| 9 | 10 | 4 | 3 | 2010-07-30 00:00:00.000 |
| 10 | 14 | 2 | 3 | 2010-02-03 00:00:00.000 |
| 11 | 15 | 6 | 3 | 2010-02-03 00:00:00.000 |
| 12 | 12 | 19 | 3 | 2010-01-01 00:00:00.000 |
| 13 | 7 | 16 | 2 | 2009-12-01 00:00:00.000 |
| 14 | 13 | 14 | 3 | 2008-01-01 00:00:00.000 |
| 15 | 8 | 8 | 3 | 2002-05-01 00:00:00.000 |

Query executed successfully. 5CG3222WQ7\MSSQLSERVER01 (1... 5CG3222WQ7\Vestlanezi ... Aerolíneas 00:00:00 15 rows

17. Selecciona al cliente tipo Oro que tenga más antigüedad de registro

```
SELECT TOP 1 *  
  FROM Cliente  
 WHERE idTipoCliente = '1'  
 ORDER BY fechaRegistro ASC;
```



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

```
SELECT TOP 1 *  
  FROM Cliente  
 WHERE idTipoCliente = '1'  
 ORDER BY fechaRegistro ASC;
```

The results pane displays a single row of data with the following columns: clienteld, personald, idTipoCliente, and fechaRegistro. The data values are 1, 5, 1, and 2014-07-01 00:00:00.000 respectively.

| clienteld | personald | idTipoCliente | fechaRegistro |
|-----------|-----------|---------------|-------------------------|
| 1 | 5 | 1 | 2014-07-01 00:00:00.000 |

The status bar at the bottom indicates "Query executed successfully." and "1 rows".

18. Selecciona todos los domicilios y Ordénalos por código postal, colonia y calle

```
SELECT
codigoPostal,
colonia,
calle
FROM Direccion
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL query:

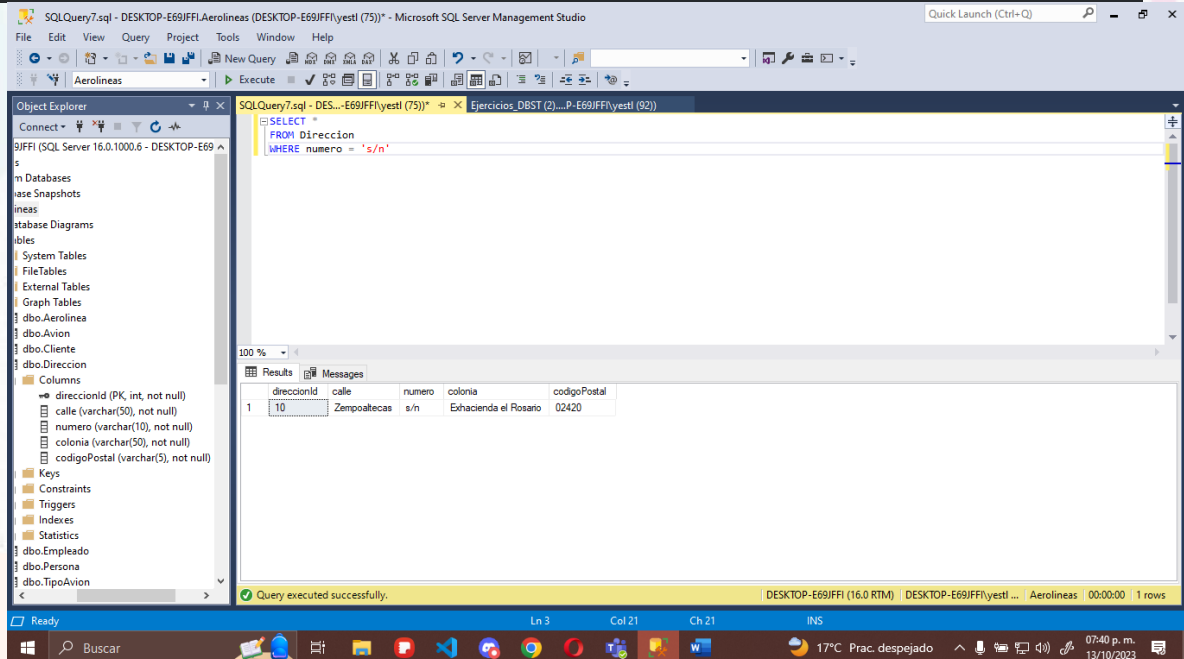
```
SELECT
codigoPostal,
colonia,
calle
FROM Direccion
```

The query has been executed successfully, and the results are displayed in the Results pane. The results are ordered by código postal, colonia, and calle. The status bar at the bottom indicates that the query was executed successfully and returned 20 rows.

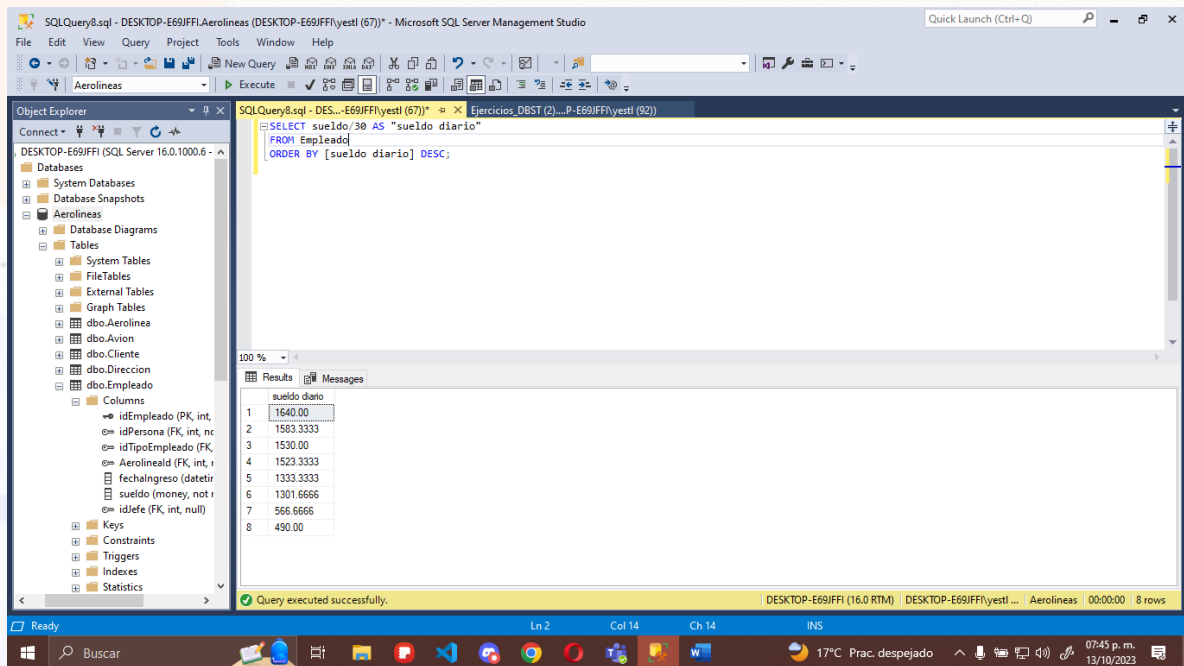
| | codigoPostal | colonia | calle |
|----|--------------|-----------------------|------------------------------|
| 1 | 06400 | Granjias | Canela |
| 2 | 06470 | San Rafael | Insurgentes Centro |
| 3 | 04470 | Presidentes Ejidales | Av. H. Escuela Naval Militar |
| 4 | 06500 | Cuauhtémoc | Rio Atoyac |
| 5 | 06600 | Juárez | Luzja |
| 6 | 06140 | Hípódromo Condesa | Alfonso Reyes |
| 7 | 06060 | Centro | Av. 20 de Noviembre |
| 8 | 06500 | Cuauhtémoc | Rio Amazonas |
| 9 | 02420 | Exhacienda el Rosario | Zemopoltecas |
| 10 | 06470 | San Rafael | Serapio Rendón |
| 11 | 06010 | Centro | Arcos de Belén |

19. Selecciona todos los domicilios cuyo número de casa no sea numérico

```
SELECT *  
FROM Direccion  
WHERE numero = 's/n'
```

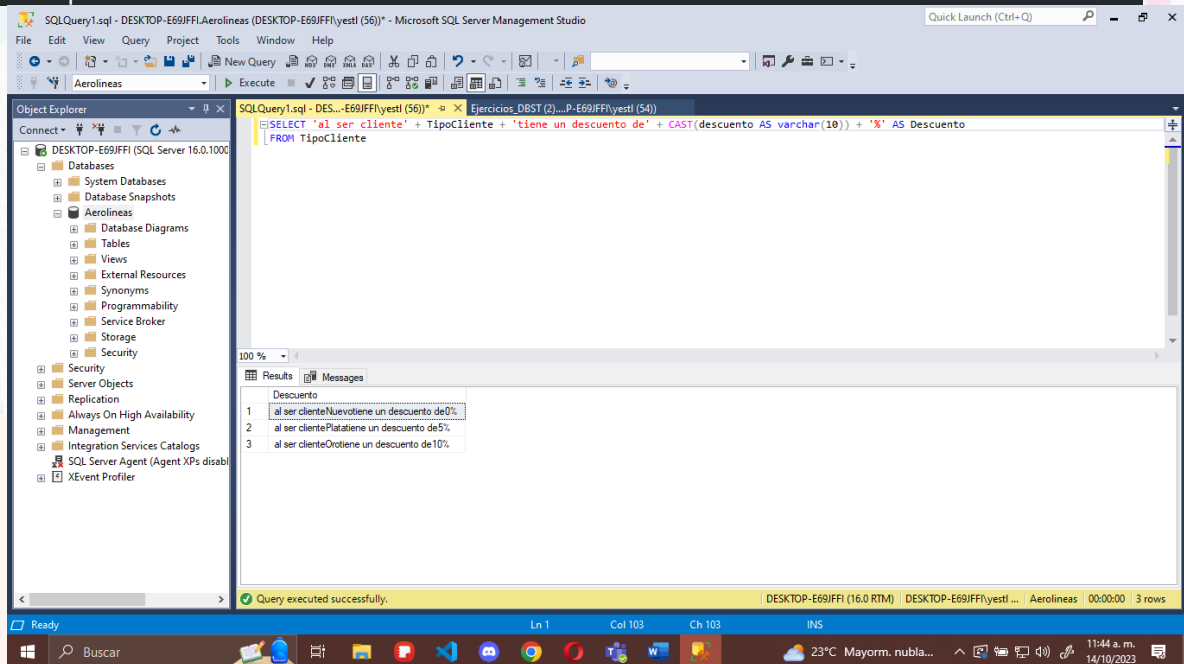


20. Selecciona el sueldo diario de los empleados considerando que todos los meses son de 30 días, agregar el alias "sueldo diario" y ordenarlo de manera descendente



21. Realiza una consulta a los tipos de clientes que muestre el siguiente texto "al ser cliente " <tipo cliente> " tienes un descuento de " <porcentaje descuento>

```
SELECT 'al ser cliente' + TipoCliente + 'tiene un descuento de' +  
CAST(descuento AS varchar(10)) + '%' AS Descuento  
FROM TipoCliente
```



22. selecciona los vuelos del mes de diciembre de 2014 que cuesten menos de 4000 pesos y ordénalos de forma ascendente en costo

```
SELECT *  
FROM Vuelo  
WHERE MONTH(fechaVuelo) = 12 AND YEAR(fechaVuelo) = 2014 AND  
precioPorAsiento < 4000  
ORDER BY precioPorAsiento ASC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL query:

```
SELECT *  
FROM Vuelo  
WHERE MONTH(fechaVuelo) = 12 AND YEAR(fechaVuelo) = 2014 AND precioPorAsiento < 4000  
ORDER BY precioPorAsiento ASC;
```

The Object Explorer on the left shows the database structure, including tables like `dbo.Aerolinea`, `dbo.Avion`, `dbo.Cliente`, `dbo.Direccion`, `dbo.Empleado`, `dbo.Persona`, `dbo.TipoAvion`, `dbo.TipoCliente`, `dbo.TipoEmpleado`, `dbo.Vendedor`, `dbo.Venta`, and `dbo.Vuelo`. The `dbo.Vuelo` table is selected, showing its columns: `idVuelo` (PK, int, not null), `matriculaAvion` (FK, int, not null), `precioPorAsiento` (money, not null), and `fechaVuelo` (datetime, not null).

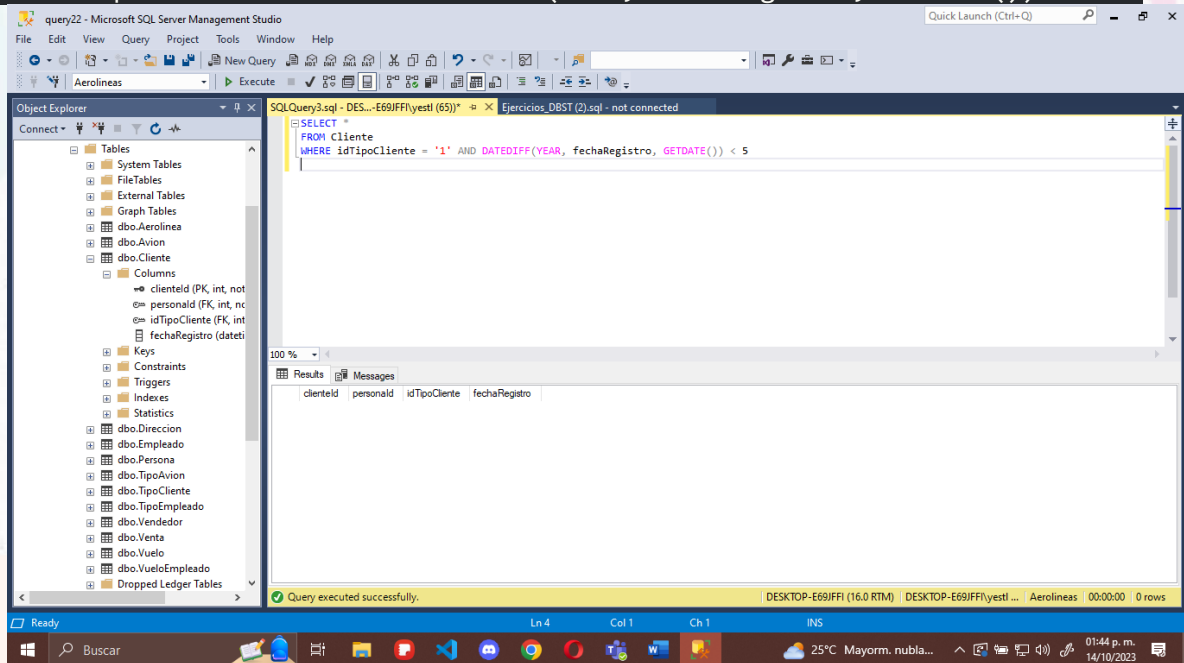
The Results pane shows the output of the query, displaying two rows of data:

| | idVuelo | matriculaAvion | precioPorAsiento | fechaVuelo |
|---|---------|----------------|------------------|-------------------------|
| 1 | 16 | 738492 | 2340.00 | 2014-12-01 00:00:00.000 |
| 2 | 2 | 234312 | 3000.00 | 2014-12-01 00:00:00.000 |

The status bar at the bottom indicates that the query was executed successfully, returning 2 rows in 00:00:00. The taskbar at the bottom shows the system clock as 01:42 p.m. on 14/10/2023.

23. selecciona los clientes tipo oro que tengan menos de 5 años de antigüedad

```
SELECT *  
FROM Cliente  
WHERE idTipoCliente = '1' AND DATEDIFF(YEAR, fechaRegistro, GETDATE()) < 5
```



24. muestra una lista de todos los tipos de aviones ordenandolos de manera ascendente por el nombre del fabricante y descendente por la cantidad de asientos

```
SELECT idTipoAvion, fabricante, numAsientos
FROM TipoAvion
ORDER BY fabricante ASC, numAsientos DESC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL query:

```
SELECT idTipoAvion, fabricante, numAsientos
FROM TipoAvion
ORDER BY fabricante ASC, numAsientos DESC;
```

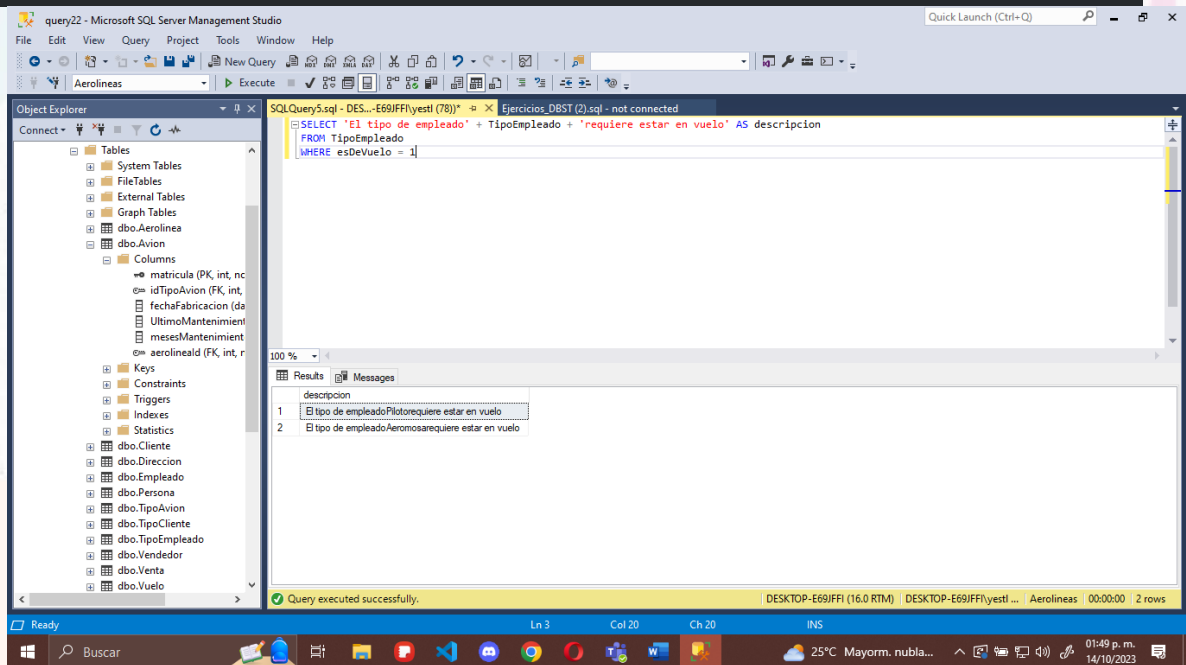
The Object Explorer on the left shows the database structure, including the 'dbo.TipoAvion' table. The Results pane on the right displays the query output as a table with 8 rows:

| | idTipoAvion | fabricante | numAsientos |
|---|-------------|---------------------------------|-------------|
| 1 | 4 | Bombardier Aerospace | 100 |
| 2 | 6 | Mitsubishi Aircraft Corporation | 96 |
| 3 | 2 | Airbus | 300 |
| 4 | 1 | Airbus | 220 |
| 5 | 3 | Boeing Commercial Airplanes | 180 |
| 6 | 5 | COMAC | 190 |
| 7 | 7 | Tupolev | 269 |
| 8 | 8 | United Aircraft Corporation | 150 |

The status bar at the bottom indicates that the query was executed successfully, returning 8 rows.

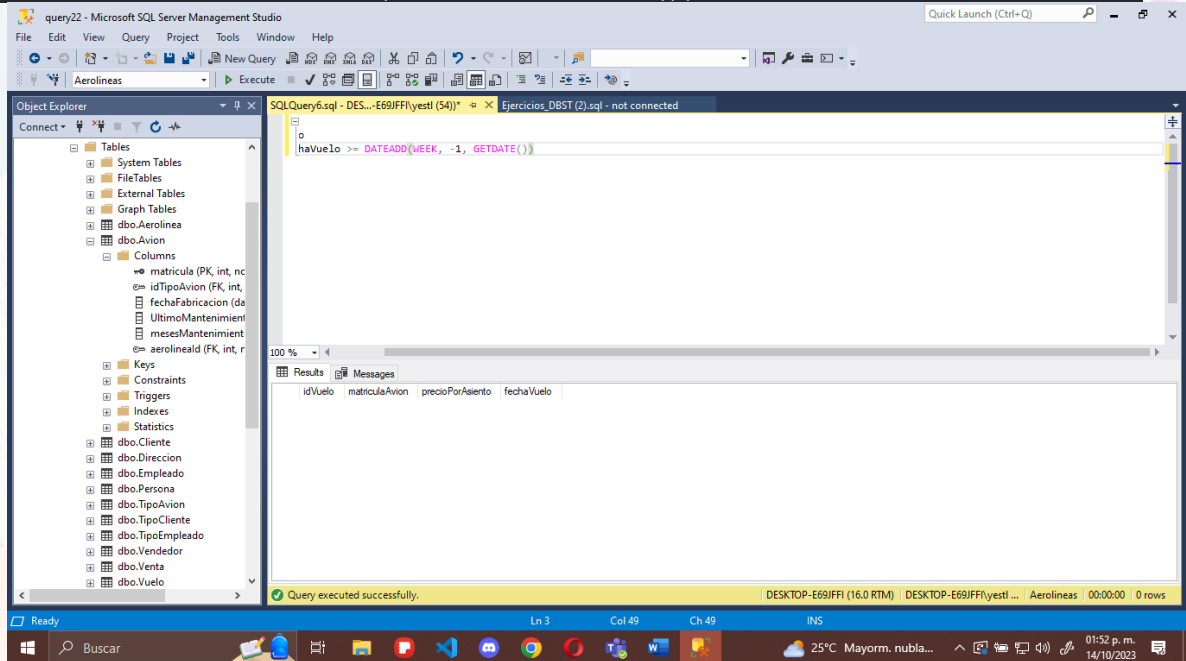
25. selecciona los tipos de empleado de vuelo con la siguiente leyenda "el tipo de empleado " + <nombre del tipo> + " requiere estar en vuelo"

```
SELECT 'El tipo de empleado' + TipoEmpleado + 'requiere estar en vuelo' AS
descripcion
FROM TipoEmpleado
WHERE esDeVuelo = 1
```



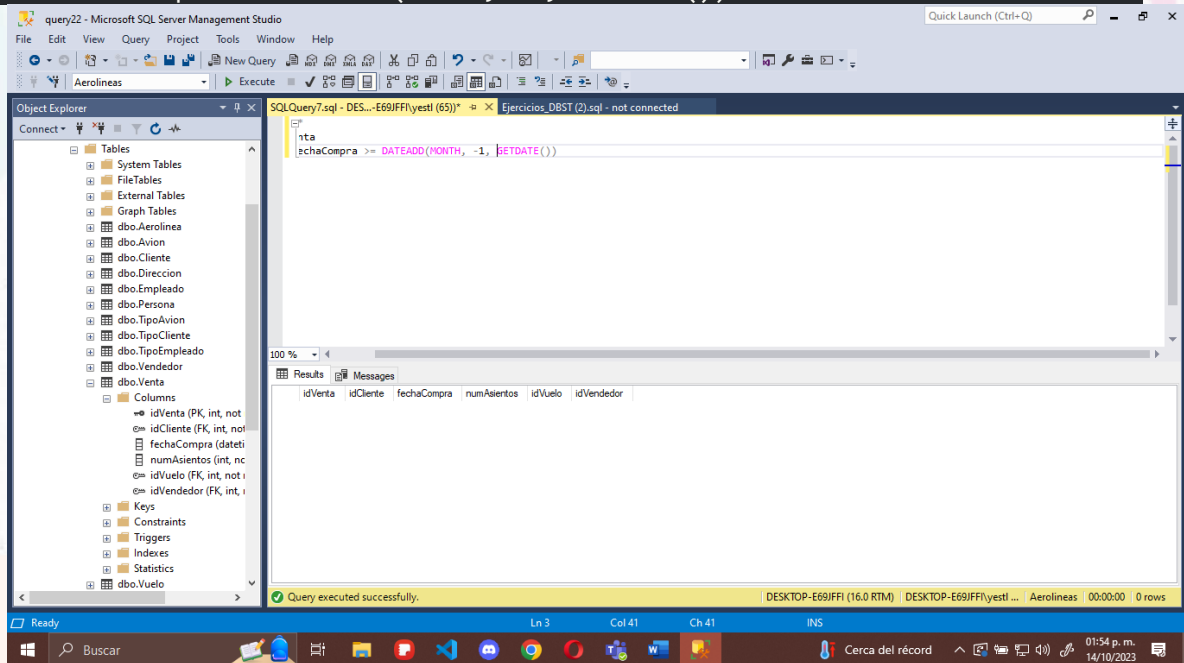
26. selecciona los vuelos que se realizaron esta última semana

```
SELECT *  
FROM Vuelo  
WHERE fechaVuelo >= DATEADD(WEEK, -1, GETDATE())
```



27. selecciona los vuelos que se compraron durante el último mes

```
SELECT *  
FROM Venta  
WHERE fechaCompra >= DATEADD(MONTH, -1, GETDATE())
```



28. selecciona a las personas cuya longitud del teléfono no sea exactamente igual a 8

```
SELECT *  
FROM Persona  
WHERE LEN(telefono) <> 8
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor contains the SQL statement: `SELECT * FROM Persona WHERE LEN(telefono) <> 8`. The query has been executed successfully, and the results are displayed in the Results pane. The results pane shows a table with 8 columns: IdPersona, nombre, paterno, materno, direccionid, telefono, nacimiento, and sexo. There are 5 rows of data. The status bar at the bottom indicates that the query was executed successfully and that there are 5 rows returned.

| | IdPersona | nombre | paterno | materno | direccionid | telefono | nacimiento | sexo |
|---|-----------|---------|-----------|----------|-------------|-------------|-------------------------|------|
| 1 | 1 | Edgardo | Hernández | Madrigal | 2 | 5561283734 | 1980-08-30 00:00:00.000 | 0 |
| 2 | 4 | Eduardo | Ortiz | NULL | 8 | 53123212231 | 1965-09-27 00:00:00.000 | 0 |
| 3 | 9 | Marco | Ramirez | Ortiz | 11 | 53247845232 | 1987-02-09 00:00:00.000 | 0 |
| 4 | 10 | Elisa | Romero | Ayala | 12 | 535221 | 1979-08-30 00:00:00.000 | 1 |
| 5 | 15 | Monica | Juárez | NULL | 17 | 591203 | 1979-12-09 00:00:00.000 | 1 |

29. Selecciona las compras con una antigüedad mayor a un año

```
SELECT *  
FROM Venta  
WHERE fechaCompra <= DATEADD(year, -1, GETDATE())
```

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure, including tables like dbo.Aerolinea, dbo.Avion, dbo.Cliente, dbo.Direccion, dboEmpleado, dbo.Persona, dbo.TipoAvion, dbo.TipoEmpleado, dbo.Vendedor, and dbo.Venta. The query window on the right contains the following SQL query:

```
SELECT *  
FROM Venta  
WHERE fechaCompra <= DATEADD(year, -1, GETDATE())
```

The Results pane shows the output of the query, displaying 11 rows of data. The status bar at the bottom indicates that the query was executed successfully and returned 23 rows.

| idVenta | idCliente | fechaCompra | numAsientos | idVuelo | idVendedor |
|---------|-----------|-------------------------|-------------|---------|------------|
| 1 | 1 | 2014-07-01 00:00:00.000 | 2 | 2 | 1 |
| 2 | 1 | 2014-08-07 00:00:00.000 | 1 | 7 | 2 |
| 3 | 1 | 2014-05-07 00:00:00.000 | 4 | 9 | 2 |
| 4 | 2 | 2014-03-24 00:00:00.000 | 2 | 2 | 4 |
| 5 | 3 | 2014-02-26 00:00:00.000 | 1 | 7 | 4 |
| 6 | 3 | 2014-02-26 00:00:00.000 | 2 | 10 | 4 |
| 7 | 4 | 2014-01-01 00:00:00.000 | 4 | 8 | 2 |
| 8 | 4 | 2014-02-20 00:00:00.000 | 2 | 9 | 1 |
| 9 | 5 | 2014-12-23 00:00:00.000 | 1 | 12 | 1 |
| 10 | 6 | 2013-09-11 00:00:00.000 | 2 | 14 | 1 |
| 11 | 7 | 2014-08-30 00:00:00.000 | 4 | 13 | 1 |

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure, including tables like dbo.Aerolinea, dbo.Avion, dbo.Cliente, dbo.Direccion, dboEmpleado, dbo.Persona, dbo.TipoAvion, dbo.TipoEmpleado, dbo.Vendedor, and dbo.Venta. The query window on the right contains the following SQL query:

```
SELECT *  
FROM Venta  
WHERE fechaCompra <= DATEADD(year, -1, GETDATE())
```

The Results pane shows the output of the query, displaying 11 rows of data. The status bar at the bottom indicates that the query was executed successfully and returned 23 rows.

| idVenta | idCliente | fechaCompra | numAsientos | idVuelo | idVendedor |
|---------|-----------|-------------------------|-------------|---------|------------|
| 13 | 15 | 2014-03-02 00:00:00.000 | 2 | 15 | 1 |
| 14 | 16 | 2014-05-09 00:00:00.000 | 4 | 15 | 2 |
| 15 | 17 | 2014-08-30 00:00:00.000 | 4 | 15 | 2 |
| 16 | 18 | 2014-07-23 00:00:00.000 | 4 | 12 | 2 |
| 17 | 19 | 2014-01-01 00:00:00.000 | 4 | 8 | 1 |
| 18 | 20 | 2013-11-05 00:00:00.000 | 2 | 8 | 1 |
| 19 | 21 | 2014-09-02 00:00:00.000 | 3 | 20 | 4 |
| 20 | 23 | 2014-09-02 00:00:00.000 | 3 | 19 | 4 |
| 21 | 24 | 2013-01-01 00:00:00.000 | 5 | 15 | 1 |
| 22 | 25 | 2014-11-25 00:00:00.000 | 4 | 8 | 1 |
| 23 | 26 | 2014-12-31 00:00:00.000 | 3 | 8 | 1 |

30. selecciona el día de la semana en que saldrán todos los vuelos.

```
SELECT DISTINCT DATENAME(WEEKDAY, fechaVuelo) AS dia_semana  
FROM Vuelo;
```

