

1. Consulta con GROUP BY: Escribe una consulta que muestre la cantidad de usuarios por edad, agrupados por rangos (por ejemplo, 18-25, 26-35, etc.) en la tabla "Usuarios".

```
SELECT
CASE
    WHEN Edad BETWEEN 18 AND 25 THEN '18-25'
    WHEN Edad BETWEEN 26 AND 35 THEN '26-35'
    WHEN Edad BETWEEN 36 AND 45 THEN '36-45'
    ELSE '46+'
END AS RangoEdad,
COUNT(*) AS CantidadAspirantes
FROM Aspirantes
GROUP BY
CASE
    WHEN Edad BETWEEN 18 AND 25 THEN '18-25'
    WHEN Edad BETWEEN 26 AND 35 THEN '26-35'
    WHEN Edad BETWEEN 36 AND 45 THEN '36-45'
    ELSE '46+'
END;
```

Esta consulta utiliza la cláusula CASE para asignar cada edad a un rango específico. Luego, utiliza la función de agregación COUNT(*) para contar la cantidad de aspirantes en cada rango. La consulta agrupa los resultados por el rango de edad y muestra la cantidad de aspirantes en cada uno.

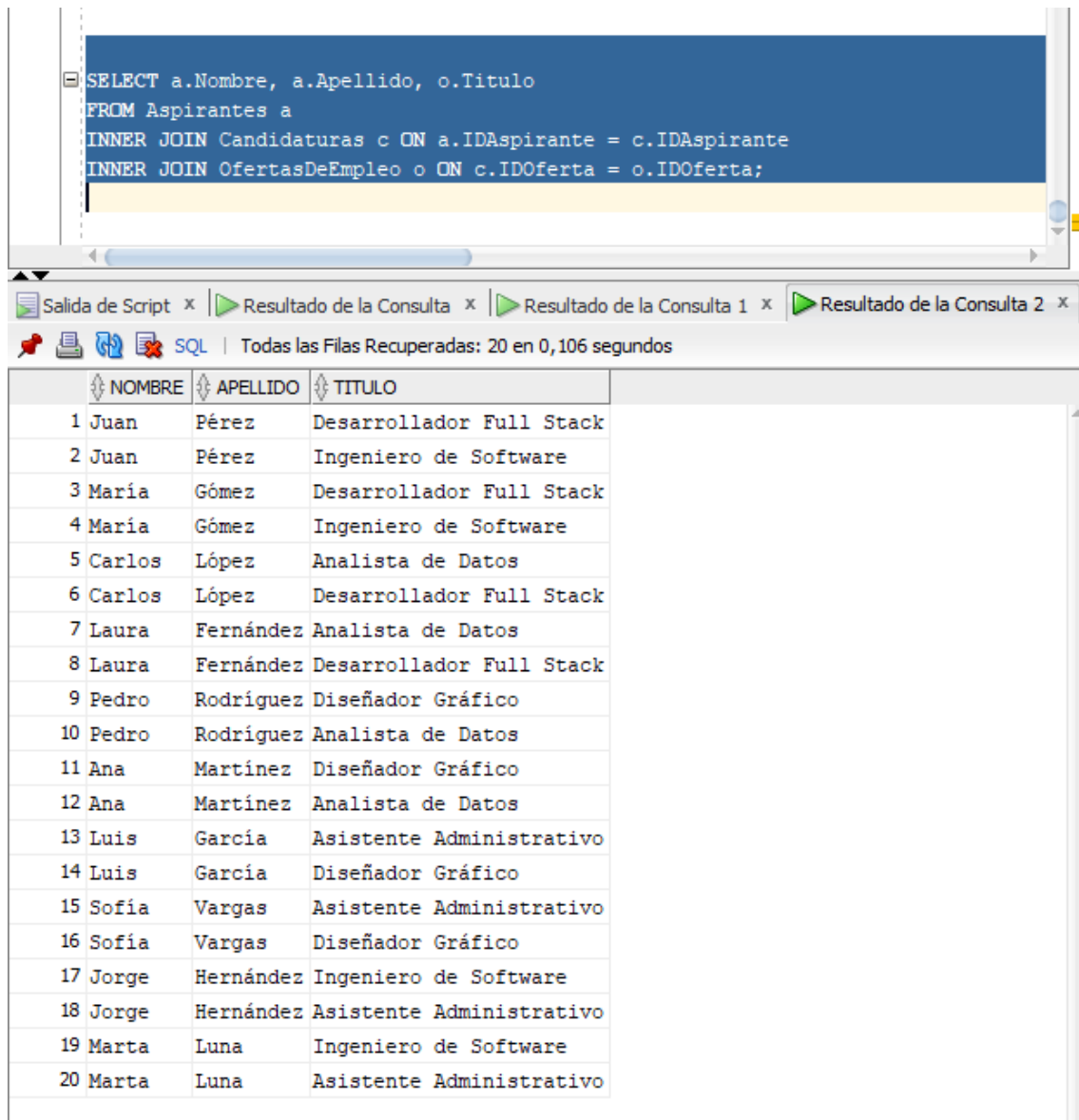
Esta consulta repetirá la expresión CASE en la cláusula GROUP BY, asegurando que se pueda agrupar correctamente por el rango de edad.

2. Consulta con INNER JOIN: Realiza una consulta que muestre el nombre de los usuarios y los títulos de los empleos a los que se han inscrito. Utiliza las tablas "Usuarios" y "Inscripciones" en la consulta.

```

SELECT a.Nombre, a.Apellido, o.Titulo
FROM Aspirantes a
INNER JOIN Candidaturas c ON a.IDAspirante = c.IDAspirante
INNER JOIN OfertasDeEmpleo o ON c.IDOferta = o.IDOferta;

```



Salida de Script x | Resultado de la Consulta x | Resultado de la Consulta 1 x | Resultado de la Consulta 2 x

SQL | Todas las Filas Recuperadas: 20 en 0,106 segundos

	NOMBRE	APELLIDO	TITULO
1	Juan	Pérez	Desarrollador Full Stack
2	Juan	Pérez	Ingeniero de Software
3	María	Gómez	Desarrollador Full Stack
4	María	Gómez	Ingeniero de Software
5	Carlos	López	Analista de Datos
6	Carlos	López	Desarrollador Full Stack
7	Laura	Fernández	Analista de Datos
8	Laura	Fernández	Desarrollador Full Stack
9	Pedro	Rodríguez	Diseñador Gráfico
10	Pedro	Rodríguez	Analista de Datos
11	Ana	Martínez	Diseñador Gráfico
12	Ana	Martínez	Analista de Datos
13	Luis	García	Asistente Administrativo
14	Luis	García	Diseñador Gráfico
15	Sofía	Vargas	Asistente Administrativo
16	Sofía	Vargas	Diseñador Gráfico
17	Jorge	Hernández	Ingeniero de Software
18	Jorge	Hernández	Asistente Administrativo
19	Marta	Luna	Ingeniero de Software
20	Marta	Luna	Asistente Administrativo

Esta consulta realiza un **INNER JOIN** entre las tablas "Aspirantes" y "Candidaturas" usando la columna **IDAspirante**, y luego realiza otro **INNER JOIN** entre las tablas "Candidaturas" y "OfertasDeEmpleo" usando la columna **IDOferta**. De esta manera, se relacionan los aspirantes con las candidaturas y las candidaturas con las ofertas de empleo.

La consulta selecciona el nombre y apellido de los aspirantes de la tabla "Aspirantes" y el título de las ofertas de la tabla "OfertasDeEmpleo" para las inscripciones correspondientes en la tabla "Candidaturas".

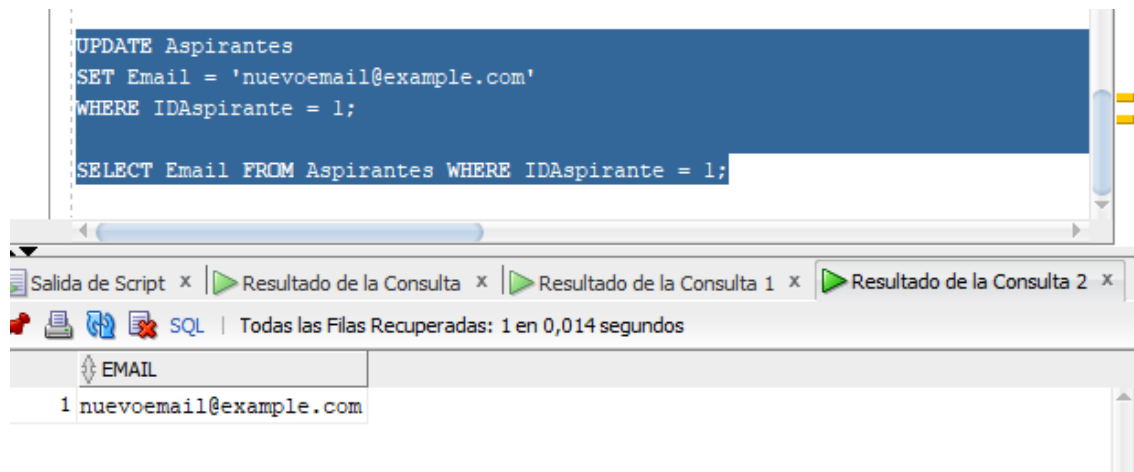
3. Actualización con UPDATE: Actualiza el campo "Email" del usuario con ID = 1 en la tabla "Usuarios" y cambia su valor a "nuevoemail@example.com".

```
UPDATE Aspirantes
```

```
SET Email = 'nuevoemail@example.com'
```

```
WHERE IDAspirante = 1;
```

```
SELECT Email FROM Aspirantes WHERE IDAspirante = 1;
```

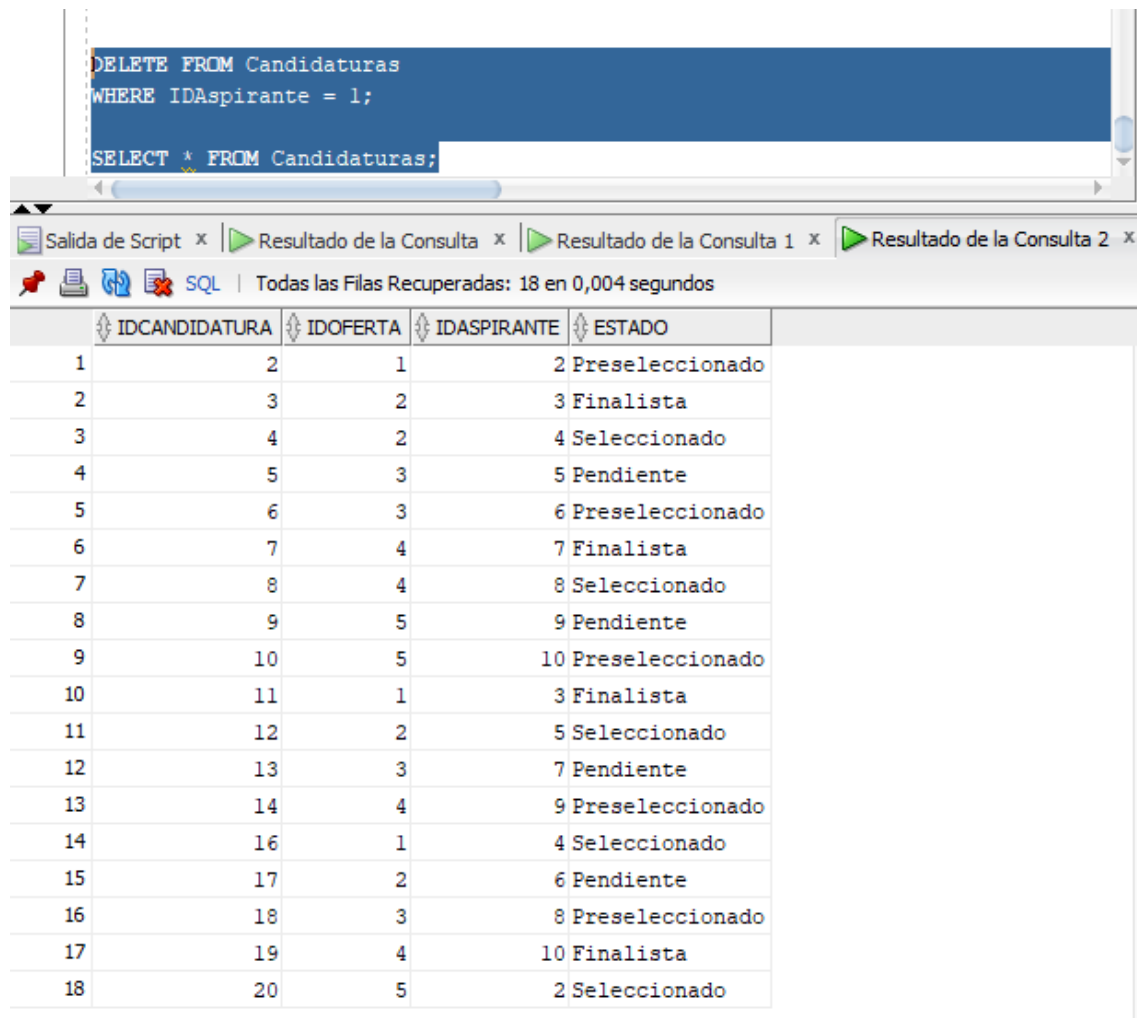


Esta consulta utiliza la sentencia **UPDATE** para modificar el campo "Email" en la tabla "Aspirantes". La cláusula **SET** establece el nuevo valor del campo y la cláusula **WHERE** especifica la condición para seleccionar el aspirante con IDAspirante = 1.

4. Eliminación con DELETE: Elimina todas las inscripciones de la tabla "Inscripciones" que correspondan al usuario con ID = 1.

```
DELETE FROM Candidaturas
```

```
WHERE IDAspirante = 1;
```



The screenshot shows a SQL IDE interface. At the top, a script editor contains the following SQL code:

```
DELETE FROM Candidaturas
WHERE IDAspirante = 1;

SELECT * FROM Candidaturas;
```

Below the script editor, the results pane displays the output of the second query. It shows a table with 18 rows and 4 columns: IDCANDIDATURA, IDOFERTA, IDASPIRANTE, and ESTADO. The status bar indicates that 18 rows were recovered in 0.004 seconds.

IDCANDIDATURA	IDOFERTA	IDASPIRANTE	ESTADO
1	2	1	2 Preseleccionado
2	3	2	3 Finalista
3	4	2	4 Seleccionado
4	5	3	5 Pendiente
5	6	3	6 Preseleccionado
6	7	4	7 Finalista
7	8	4	8 Seleccionado
8	9	5	9 Pendiente
9	10	5	10 Preseleccionado
10	11	1	3 Finalista
11	12	2	5 Seleccionado
12	13	3	7 Pendiente
13	14	4	9 Preseleccionado
14	16	1	4 Seleccionado
15	17	2	6 Pendiente
16	18	3	8 Preseleccionado
17	19	4	10 Finalista
18	20	5	2 Seleccionado

Esta consulta utiliza la sentencia **DELETE** para eliminar filas de la tabla "Candidaturas". La cláusula **WHERE** especifica la condición para seleccionar las inscripciones que corresponden al usuario con IDAspirante = 1.

5. Consulta con WHERE: Escribe una consulta que muestre los usuarios cuyo nombre empiece con la letra "A" en la tabla "Usuarios".

```
SELECT *
FROM Aspirantes
WHERE Nombre LIKE 'A%';
```



```
SELECT * FROM Candidaturas;

SELECT *
FROM Aspirantes
WHERE Nombre LIKE 'A%';
```

Salida de Script x | Resultado de la Consulta x | Resultado de la Consulta 1 x | Resultado de la Consulta 2 x

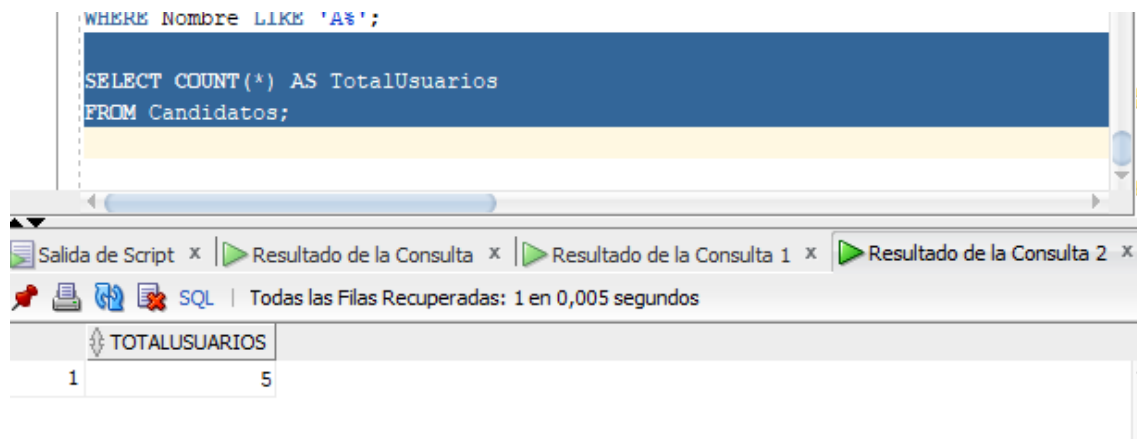
SQL | Todas las Filas Recuperadas: 1 en 0,006 segundos

	IDASPIRANTE	NOMBRE	APELLIDO	EDAD	EMAIL	TELEFONO	DIRECCION
1	6	Ana	Martinez	29	ana@example.com	8887776665	Calle Central 890

Esta consulta selecciona todas las columnas de la tabla "Aspirantes" donde el valor del campo "Nombre" comienza con la letra "A" seguida de cualquier número de caracteres. El símbolo "%" se utiliza como comodín para representar cualquier secuencia de caracteres.

6. Consulta con COUNT: Realiza una consulta que muestre el número total de usuarios en la tabla "Usuarios".

```
SELECT COUNT(*) AS TotalUsuarios
FROM Candidatos;
```



```
WHERE Nombre LIKE 'A%';

SELECT COUNT(*) AS TotalUsuarios
FROM Candidatos;
```

Salida de Script x | Resultado de la Consulta x | Resultado de la Consulta 1 x | Resultado de la Consulta 2 x

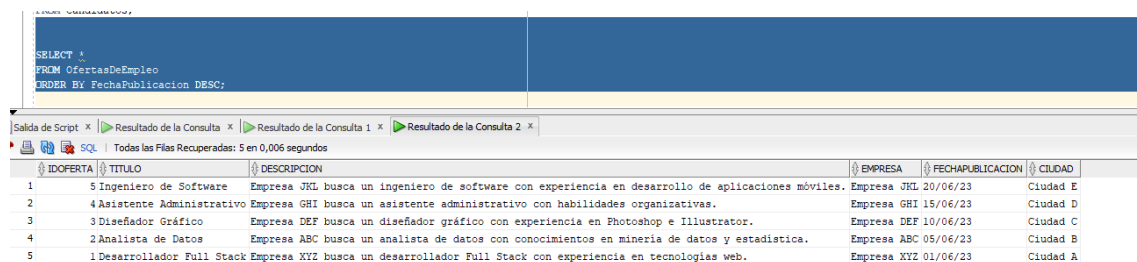
SQL | Todas las Filas Recuperadas: 1 en 0,005 segundos

	TOTALUSUARIOS
1	5

Esta consulta utiliza la función COUNT(*) para contar todas las filas de la tabla "Candidatos". El alias "TotalUsuarios" se utiliza para dar nombre al resultado obtenido.

7. Consulta con ORDER BY: Muestra los empleos de la tabla "Empleos" ordenados por fecha de publicación de forma descendente.

```
SELECT *  
  
FROM OfertasDeEmpleo  
  
ORDER BY FechaPublicacion DESC;
```



The screenshot shows a SQL query editor with the following query:

```
SELECT *  
FROM OfertasDeEmpleo  
ORDER BY FechaPublicacion DESC;
```

Below the query, the results are displayed in a table with 5 rows and 5 columns. The columns are: ID_OFERTA, TITULO, DESCRIPCION, EMPRESA, and FECHA_PUBLICACION. The results are ordered by FECHA_PUBLICACION in descending order.

ID_OFERTA	TITULO	DESCRIPCION	EMPRESA	FECHA_PUBLICACION
1	5 Ingeniero de Software	Empresa JKL busca un ingeniero de software con experiencia en desarrollo de aplicaciones móviles.	Empresa JKL	20/06/23
2	4 Asistente Administrativo	Empresa GHI busca un asistente administrativo con habilidades organizativas.	Empresa GHI	15/06/23
3	3 Diseñador Gráfico	Empresa DEF busca un diseñador gráfico con experiencia en Photoshop e Illustrator.	Empresa DEF	10/06/23
4	2 Analista de Datos	Empresa ABC busca un analista de datos con conocimientos en minería de datos y estadística.	Empresa ABC	05/06/23
5	1 Desarrollador Full Stack	Empresa XYZ busca un desarrollador Full Stack con experiencia en tecnologías web.	Empresa XYZ	01/06/23

Esta consulta selecciona todas las columnas de la tabla "OfertasDeEmpleo" y las ordena en función de la columna "FechaPublicacion" en orden descendente, lo que mostrará los empleos más recientes primero.

8. Consulta con HAVING: Escribe una consulta que muestre los rangos de edad y el número de usuarios por rango, pero solo para aquellos rangos que tengan más de 5 usuarios. Utiliza la tabla "Usuarios".

```
SELECT  
  
CASE  
  
    WHEN Edad BETWEEN 18 AND 25 THEN '18-25'  
  
    WHEN Edad BETWEEN 26 AND 35 THEN '26-35'  
  
    WHEN Edad BETWEEN 36 AND 45 THEN '36-45'  
  
    ELSE '46+'  
  
END AS RangoEdad,  
  
COUNT(*) AS NumeroAspirantes  
  
FROM Aspirantes
```

GROUP BY

CASE

WHEN Edad BETWEEN 18 AND 25 THEN '18-25'

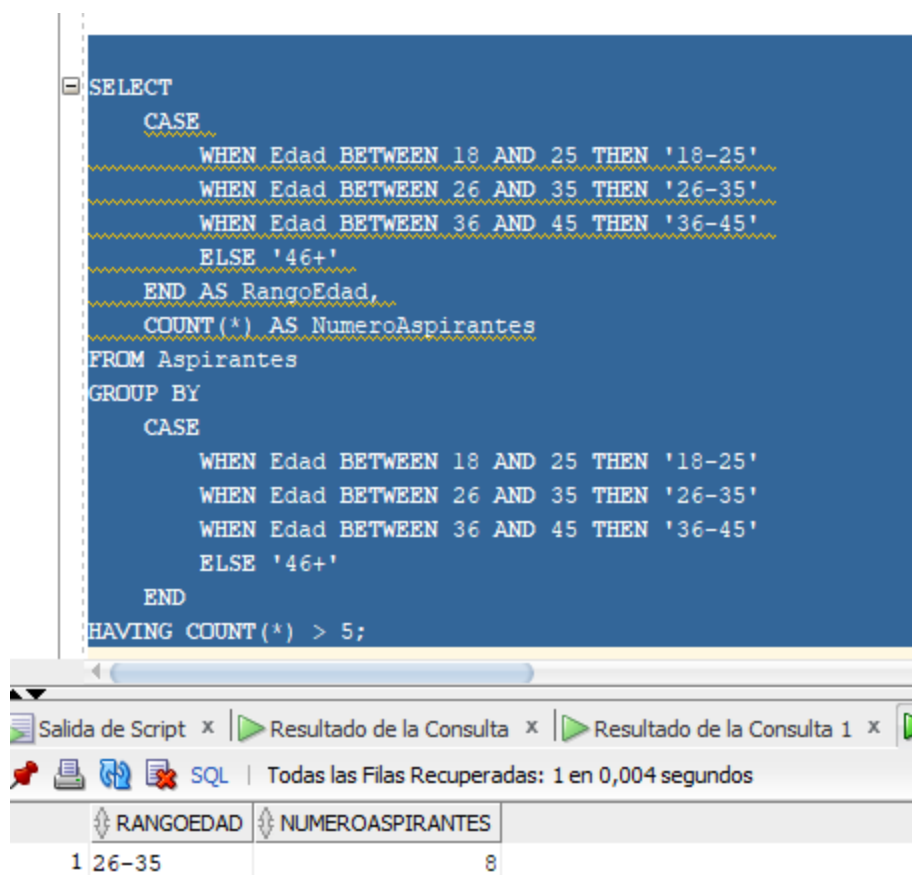
WHEN Edad BETWEEN 26 AND 35 THEN '26-35'

WHEN Edad BETWEEN 36 AND 45 THEN '36-45'

ELSE '46+'

END

HAVING COUNT(*) > 5;



```
SELECT
  CASE
    WHEN Edad BETWEEN 18 AND 25 THEN '18-25'
    WHEN Edad BETWEEN 26 AND 35 THEN '26-35'
    WHEN Edad BETWEEN 36 AND 45 THEN '36-45'
    ELSE '46+'
  END AS RangoEdad,
  COUNT(*) AS NumeroAspirantes
FROM Aspirantes
GROUP BY
  CASE
    WHEN Edad BETWEEN 18 AND 25 THEN '18-25'
    WHEN Edad BETWEEN 26 AND 35 THEN '26-35'
    WHEN Edad BETWEEN 36 AND 45 THEN '36-45'
    ELSE '46+'
  END
HAVING COUNT(*) > 5;
```

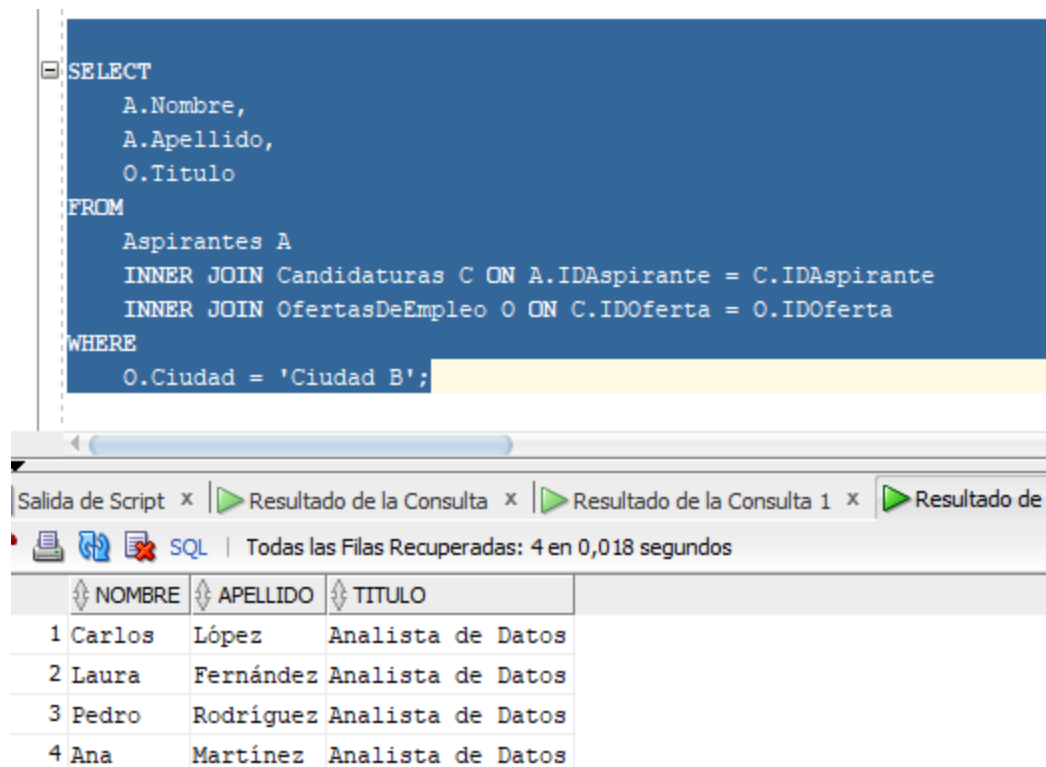
	RANGOEDAD	NUMEROASPIRANTES
1	26-35	8

9. Consulta con INNER JOIN y WHERE: Muestra los nombres de los usuarios y los títulos de los empleos a los que se han inscrito, pero solo para aquellos empleos que estén ubicados en una ciudad específica. Utiliza las tablas "Usuarios", "Inscripciones" y "Empleos".

```

SELECT
    A.Nombre,
    A.Apellido,
    O.Titulo
FROM
    Aspirantes A
    INNER JOIN Candidaturas C ON A.IDAspirante = C.IDAspirante
    INNER JOIN OfertasDeEmpleo O ON C.IDOferta = O.IDOferta
WHERE
    O.Ciudad = 'Ciudad B';

```



```

SELECT
    A.Nombre,
    A.Apellido,
    O.Titulo
FROM
    Aspirantes A
    INNER JOIN Candidaturas C ON A.IDAspirante = C.IDAspirante
    INNER JOIN OfertasDeEmpleo O ON C.IDOferta = O.IDOferta
WHERE
    O.Ciudad = 'Ciudad B';

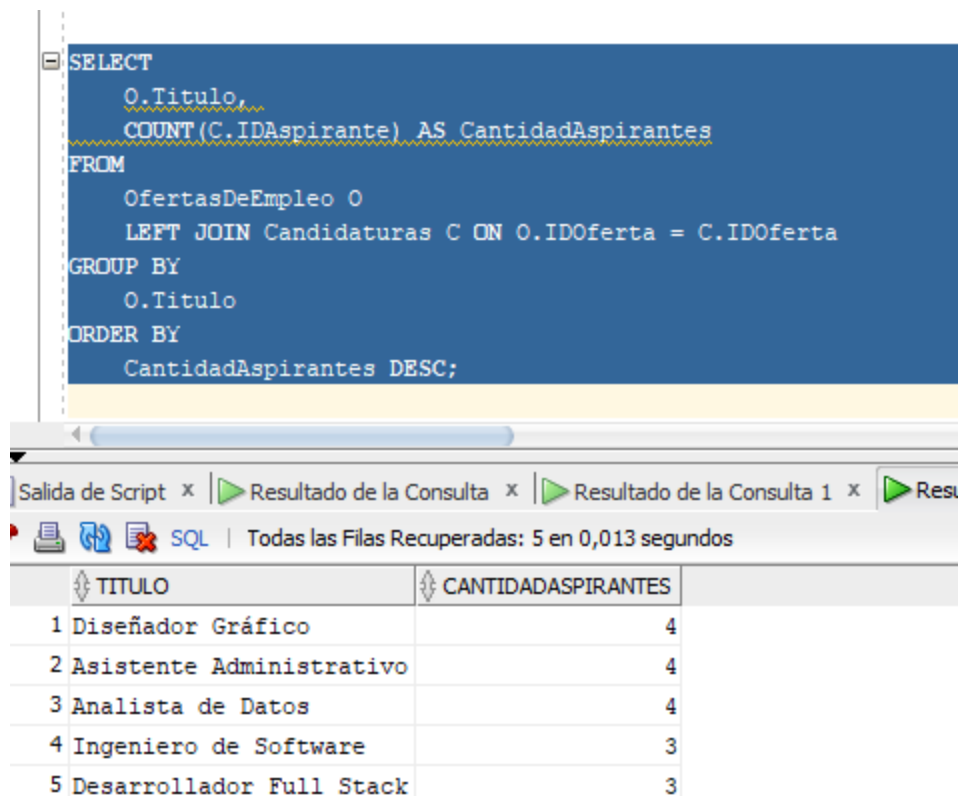
```

	NOMBRE	APELLIDO	TITULO
1	Carlos	López	Analista de Datos
2	Laura	Fernández	Analista de Datos
3	Pedro	Rodríguez	Analista de Datos
4	Ana	Martínez	Analista de Datos

En esta consulta, se realiza un **INNER JOIN** entre las tablas "Aspirantes", "Candidaturas" y "OfertasDeEmpleo" utilizando las claves primarias y foráneas correspondientes. Luego, se utiliza la cláusula **WHERE** para filtrar solo los empleos que estén ubicados en la ciudad específica que desees.

10. Consulta con GROUP BY y COUNT: Escribe una consulta que muestre los títulos de los empleos y la cantidad de usuarios inscritos en cada uno, ordenados de mayor a menor cantidad de inscripciones. Utiliza las tablas "Empleos" e "Inscripciones"

```
SELECT
    O.Titulo,
    COUNT(C.IDAspirante) AS CantidadAspirantes
FROM
    OfertasDeEmpleo O
    LEFT JOIN Candidaturas C ON O.IDOferta = C.IDOferta
GROUP BY
    O.Titulo
ORDER BY
    CantidadAspirantes DESC;
```



The screenshot shows a SQL query editor with a blue background for the query text. The query is as follows:

```
SELECT
    O.Titulo,
    COUNT(C.IDAspirante) AS CantidadAspirantes
FROM
    OfertasDeEmpleo O
    LEFT JOIN Candidaturas C ON O.IDOferta = C.IDOferta
GROUP BY
    O.Titulo
ORDER BY
    CantidadAspirantes DESC;
```

Below the query editor, there is a results pane with a tab labeled "Resultado de la Consulta 1". The results are displayed in a table with two columns: "TITULO" and "CANTIDADASPIRANTES". The table contains five rows of data, ordered by the number of aspirants in descending order.

TITULO	CANTIDADASPIRANTES
1 Diseñador Gráfico	4
2 Asistente Administrativo	4
3 Analista de Datos	4
4 Ingeniero de Software	3
5 Desarrollador Full Stack	3

En esta consulta, se utiliza un **LEFT JOIN** entre las tablas "OfertasDeEmpleo" y "Candidaturas" para obtener todas las ofertas de empleo, incluso si no tienen candidaturas asociadas. Luego, se utiliza la función de agregación **COUNT** para contar el número de aspirantes inscritos en cada oferta. La cláusula **GROUP BY** agrupa los resultados por el título de la oferta. Finalmente, la cláusula **ORDER BY** ordena los resultados en función de la cantidad de aspirantes de forma descendente.