

EJERCICIO GUIADO. BASES DE DATOS. SQL

Lenguaje de Consulta SQL

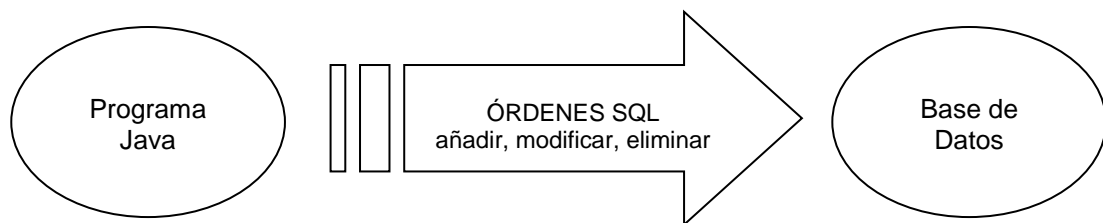
El lenguaje SQL (lenguaje de consulta estructurado) es un lenguaje que permite “actuar” sobre una base de datos.

Con este lenguaje se pueden construir órdenes que permiten hacer lo siguiente (entre otras cosas):

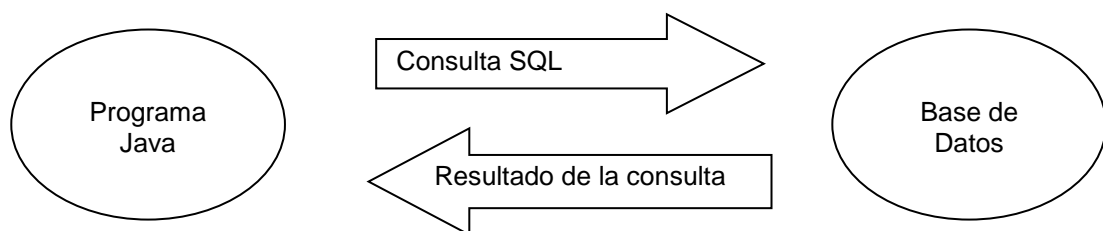
- Añadir registros a las tablas.
- Modificar registros de las tablas.
- Eliminar registros de las tablas.
- Realizar Consultas sobre las tablas.

Gracias a este lenguaje, se construirán órdenes desde nuestra aplicación java, que se aplicarán a la base de datos, actuando sobre ella.

Las órdenes de *añadir*, *modificar*, *eliminar* realizan cambios dentro de la base de datos, pero no devuelven nada al programa java.



Por otro lado, cuando se da una orden de *realizar una consulta*, la base de datos nos devuelve el resultado de dicha consulta:



Gracias a este lenguaje, nuestra aplicación tiene dominio total sobre la base de datos. Puede actuar sobre ella introduciendo nuevos datos, o modificando los que había, o eliminándolos. También puede extraer información de ella accediendo a las consultas de la base de datos o realizando nuevas consultas.

A continuación se comentarán las reglas básicas de este lenguaje.

Creación de consultas en SQL

Se empezará estudiando como realizar consultas sobre una base de datos usando el lenguaje SQL (más adelante se verá como realizar consultas de acción: añadir, modificar eliminar)

Código base en SQL para realizar consultas

Para consultar una base de datos usará un código general como el que sigue:

```
SELECT campos a visualizar
FROM tablas donde se encuentran dichos campos
WHERE condiciones que deben cumplir los registros
ORDER BY forma de ordenar la consulta;
```

Como puede ver, una consulta en SQL tiene cuatro partes (SELECT, FROM, WHERE y ORDER BY) de las cuales solo las dos primeras son obligatorias.

Se debe mantener el orden indicado. Es decir, primero SELECT, luego FROM, luego WHERE y luego ORDER BY.

Este código debe terminar siempre con punto y coma ;

A continuación se verán ejemplos de uso de este código general.

Visualizar una tabla entera (todos los campos y todos los registros)

Ejemplo: "Visualizar la tabla Clientes"

```
SELECT *
FROM clientes;
```

Observa, el * significa ver todos los campos. En el FROM se indica la tabla que se quiere ver. Observa como hay que terminar con un ;

Visualizar algunos campos de una tabla (algunos campos y todos los registros)

Ejemplo: "Visualizar CIF, nombre y Direccion de todos los clientes"

```
SELECT clientes.CIF, clientes.nombre, clientes.direccion
FROM clientes;
```

Observa como se indican los campos a visualizar en la cláusula SELECT. Se indica la tabla y luego el nombre del campo, separados por un punto.

Visualizar solo aquellos registros de la tabla que cumplan una condición

Ejemplo: “Visualizar todos los campos de aquellos trabajadores que cobren un sueldo superior a los 1000 euros”

```
SELECT *  
FROM trabajadores  
WHERE trabajadores.sueldo > 1000;
```

Observa el uso de la cláusula WHERE para aplicar una condición al resultado.

Ejemplo: “Visualizar el nombre, apellido y sueldo de aquellos trabajadores que cobren un sueldo entre 800 y 2000 euros”

```
SELECT trabajadores.nombre, trabajadores.apellidos, trabajadores.sueldo  
FROM trabajadores  
WHERE trabajadores.sueldo BETWEEN 800 AND 2000;
```

Observa el uso de BETWEEN – AND para indicar que el sueldo esté entre 800 y 2000

Nota. Más adelante en este ejercicio guiado se muestran las distintas posibilidades que tenemos a la hora de indicar criterios en la cláusula WHERE

Visualizar el contenido de una tabla ordenado

Ejemplo: “Visualizar la tabla de trabajadores ordenada por sueldo de menor a mayor”

```
SELECT *  
FROM trabajadores  
ORDER BY trabajadores.sueldo ASC;
```

Observa el uso de la cláusula ORDER BY para indicar que se ordene por sueldo. La palabra ASC indica “ascendente” (de menor a mayor)

Ejemplo: “Visualizar nombre, apellidos y sueldo de los trabajadores ordenados por sueldos de mayor a menor”

```
SELECT trabajadores.nombre, trabajadores.apellidos, trabajadores.sueldo  
FROM trabajadores  
ORDER BY trabajadores.sueldo DESC;
```

Observa el uso de DESC para indicar una ordenación descendente.

Ejemplo: “Visualizar nombre, apellidos y sueldo de los trabajadores que cobren más de 1000 euros, ordenados por apellidos y nombre”

```
SELECT trabajadores.nombre, trabajadores.apellidos, trabajadores.sueldo
FROM trabajadores
WHERE trabajadores.sueldo > 1000
ORDER BY trabajadores.apellidos ASC, trabajadores.nombre ASC;
```

Observa aquí como ordenar por dos campos, primero por apellidos y luego por nombre. Esto significa que aquellos trabajadores que tengan el mismo apellido serán ordenados por nombre.

Visualizar datos de varias tablas

Ejemplo: “Visualizar todos los servicios. Interesa que aparezca el nombre del trabajador que hizo el servicio, la fecha del servicio realizado y el tipo de servicio”

```
SELECT trabajadores.nombre, servicios.fecha, servicios.tipo
FROM trabajadores, servicios
WHERE trabajadores.DNI=servicios.DNI;
```

Observa aquí como se indica en la cláusula FROM las dos tablas de las que extraemos datos. Es importante que te fijas también en como se unen ambas tablas igualando en la cláusula WHERE el campo de unión de ambas tablas, que en el ejemplo es el DNI.

Ejemplo: “Visualizar todos los servicios. Interesa que aparezca el nombre del trabajador que hizo el servicio, el tipo de servicio y el nombre del cliente al que se le hizo el servicio”

```
SELECT trabajadores.nombre, servicios.tipo, clientes.nombre
FROM trabajadores, servicios, clientes
WHERE trabajadores.DNI=servicios.DNI AND clientes.CIF=servicios.CIF;
```

Observa aquí una consulta sobre tres tablas, las cuales aparecen en el FROM. Es necesario indicar en la cláusula WHERE los campos de unión. La tabla Trabajadores se relaciona con la tabla Servicios a través del campo DNI, y la tabla Trabajadores se relaciona con Clientes a través del campo CIF. Observa el uso de AND para unir varias condiciones.

Ejemplo: “Visualizar los servicios que hayan costado más de 200 euros. Interesa ver la fecha del servicio, el nombre del cliente y el coste ordenado por cantidad”

```
SELECT servicios.fecha, clientes.nombre, servicios.cantidad
FROM servicios, clientes
WHERE servicios.CIF=clientes.CIF AND servicios.cantidad>200
ORDER BY servicios.cantidad;
```

Observa como la cláusula WHERE contiene por un lado la condición de unión de ambas tablas y por otro lado la condición que se busca (cantidad > 200)

FORMA DE INDICAR CRITERIOS EN LA CLÁUSULA WHERE

Se van a indicar a continuación una serie de reglas que se deben seguir a la hora de crear condiciones en la cláusula WHERE de una consulta SQL

Operadores Relacionales

Operador	Significa...	Ejemplos
=	Igual que	WHERE cantidad = 200 WHERE tipo = 'Limpieza' WHERE fecha = #8-5-2006#
>	Mayor que (para números) Posterior a (para fechas)	WHERE cantidad > 200 WHERE fecha > #8-5-2006#
>=	Mayor o igual que (para números) Esa fecha o posterior (para fechas)	WHERE cantidad >= 200 WHERE fecha >= #8-5-2006#
<	Menor que (para números) Anterior a (para fechas)	WHERE cantidad < 200 WHERE fecha < #8-5-2006#
<=	Menor o igual que (para números) Esa fecha o anterior (para fechas)	WHERE cantidad <= 200 WHERE fecha <= #8-5-2006#
<>	Distinto de (para fechas, números y textos)	WHERE cantidad <> 200 WHERE fecha <> #8-5-2006# WHERE tipo <> 'Limpieza'
Between...and	Entre valor1 y valor2 (aplicable a números y fechas)	WHERE cantidad BETWEEN 100 AND 200 WHERE fecha BETWEEN #8-5-2006# AND #1-12-2006#
Like 'cadena*'	Que empiece por <i>cadena</i> (aplicable a textos)	WHERE nombre LIKE 'Jose*'
Like '*cadena'	Que termine por <i>cadena</i> (aplicable a textos)	WHERE nombre LIKE '*Jose'

Like <i>*cadena*</i>	Que contenga <i>cadena</i> (aplicable a textos)	WHERE nombre LIKE <i>*Jose*</i>
IS NULL	Que el campo esté vacío (aplicable a números, textos, fechas)	WHERE telefono IS NULL
NOT ... IS NULL	Que el campo no esté vacío (aplicable a números, textos, fechas)	WHERE NOT telefono IS NULL

Operadores Lógicos

Operador	Significa...	Ejemplos
AND	Obliga a que se cumplan las dos condiciones que une.	WHERE cantidad > 200 AND tipo = 'Limpieza' (Debe cumplirse que la cantidad sea mayor de 200 y que el tipo de servicio sea <i>Limpieza</i>)
OR	Basta con que se cumpla una sola de las dos condiciones que une.	WHERE cantidad > 200 OR tipo = 'Limpieza' (Basta con que la cantidad sea mayor de 200, o que el tipo de servicio sea <i>Limpieza</i> para que se cumpla la condición)
NOT	Si no se cumple la condición, la condición global se cumple.	WHERE NOT cantidad > 200 (Se cumple la condición si la cantidad NO es mayor de 200)

Forma de indicar los valores

Como puedes observar en los ejemplos anteriores, tendrás que tener en cuenta las siguientes reglas para indicar valores:

Valores numéricos

Indica los valores numéricos tal cual, teniendo en cuenta que debes usar el punto decimal cuando quieras representar decimales.

Ejemplo:

WHERE cantidad > 200.12

Valores de texto

Los valores de texto se indican rodeándolos entre comillas simples: ‘

Ejemplos:

WHERE nombre = 'Jose'

WHERE dirección LIKE '*avenida*'

Valores de fecha

Las fechas se indican rodeándolas entre almohadillas #. Se debe tener en cuenta que las fechas deben indicarse separadas por guiones – o barras / y que su formato debe ser el siguiente:

Mes – Dia – Año

Ejemplos:

WHERE fecha > #02-01-2005#

(Significa que la fecha debe ser posterior al 1 de febrero de 2005)

WHERE fecha <> #10-12-2006#

(Significa que la fecha debe ser distinta del 12 de Octubre de 2006)

Forma de indicar los campos

Normalmente los campos que se usan en el WHERE (y en otras cláusulas) se indican de la siguiente forma:

Tabla.Campo

Por ejemplo,

WHERE trabajadores.sueldo > 1000

(*Sueldo* es un campo de la tabla *trabajadores*)

Si tenemos la seguridad de que no existe otro campo de otra tabla que se llame igual, entonces podemos prescindir del nombre de la tabla.

Por ejemplo,

WHERE sueldo > 1000

(No existe otro campo sueldo en otras tablas de la consulta)

En el caso de que el nombre del campo contenga espacios, entonces tendremos que rodear el campo con corchetes.

Por ejemplo,

WHERE [sueldo del trabajador] > 1000

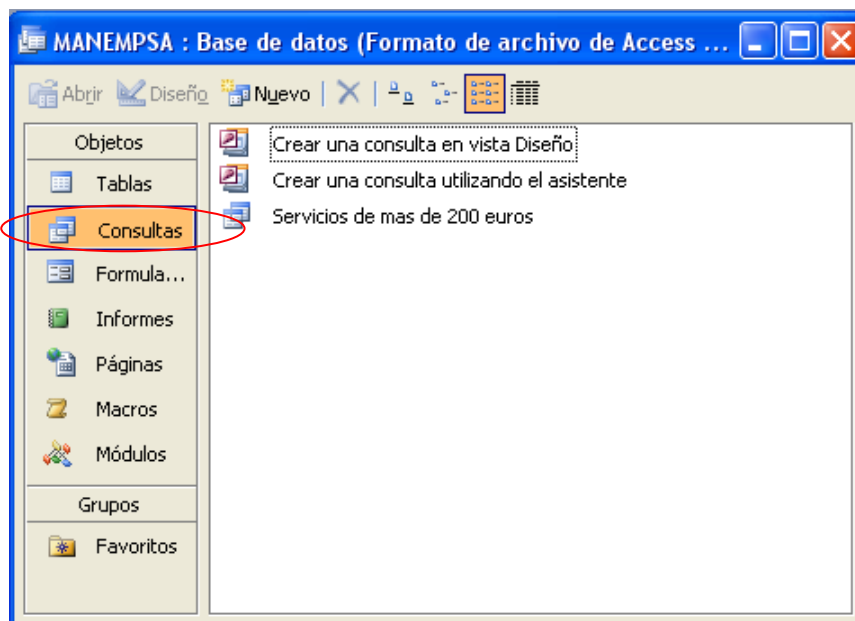
(El campo se llama *sueldo del trabajador*)

EJERCICIO GUIADO Nº 1

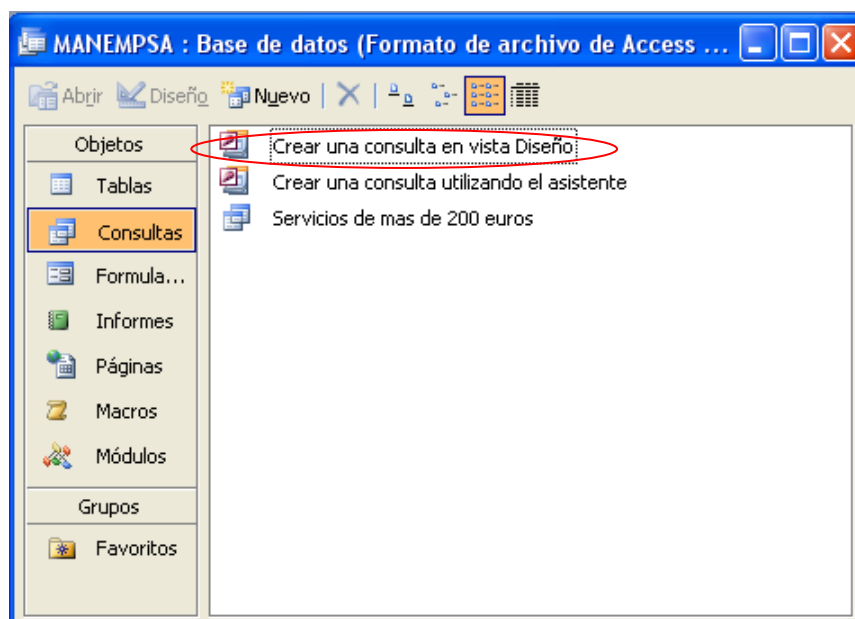
ACCESS permite la creación de consultas usando SQL. Esto nos permitirá practicar con el lenguaje SQL antes de que se aplique posteriormente en nuestras aplicaciones Java.

En este ejercicio guiado se explicará como crear una consulta usando el lenguaje SQL en Access.

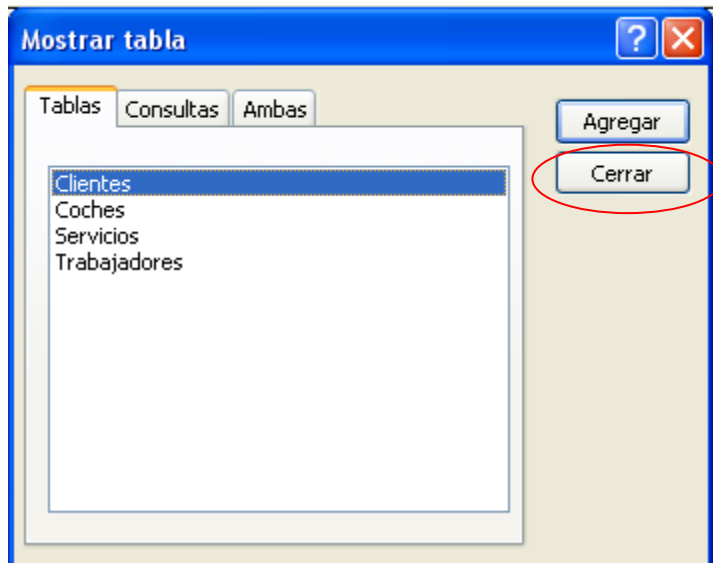
1. Entre en Mis Documentos. Allí verá la base de datos MANEMPSA. Ábrala.
2. Active la opción de *Consultas* para acceder al listado de consultas de la base de datos.



3. Vamos a crear una nueva consulta, por lo que tendrá que activar la opción *Crear una consulta en vista Diseño*:



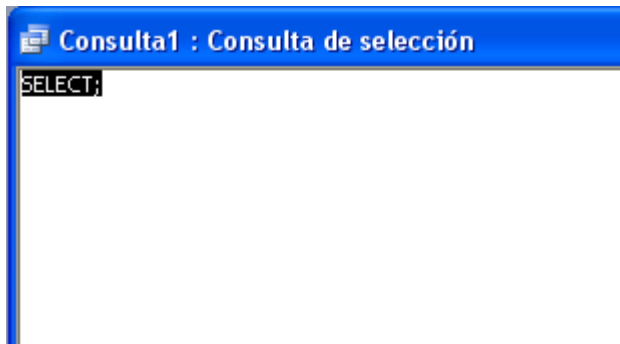
4. En el cuadro de elección de tablas no agregaremos ninguna. Así pues pulsaremos directamente el botón *Cerrar*.



5. El resultado es que aparece la ventana de diseño de consultas. En la parte superior izquierda de esa ventana podrás ver un botón con el texto SQL. Este botón nos permitirá crear la consulta usando el lenguaje SQL. Púlsalo.

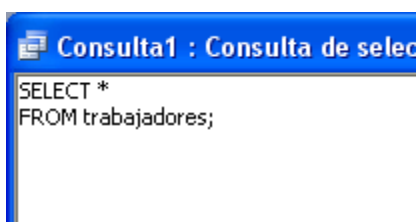


6. Aparecerá una pantalla casi en blanco donde escribirá la consulta SQL. Observe que ya aparece la cláusula *SELECT* escrita:

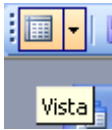


7. Para empezar mostraremos el contenido de la tabla *Trabajadores*. Para ello, debe escribir la siguiente consulta SQL:

```
SELECT *  
FROM trabajadores;
```



8. Para ver el resultado de esta consulta debe pulsar el botón *Vista* que se encuentra en la parte superior izquierda de la ventana:

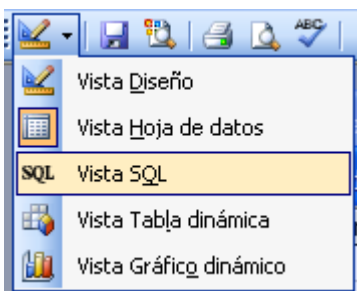


9. El resultado es que verá el contenido de la tabla *Trabajadores*:

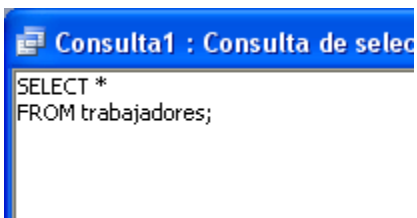
	DNI	Nombre	Apellidos	Sueldo	Fecha	Matricula
▶	12.321.567-B	Juan	Pérez	1120	04/05/2002	4433-ABB
	21.123.123-A	Ana	Ruiz	1200	02/03/2002	3322-ASR
	22.333.444-C	Francisco	López	1000	01/06/2006	1144-BBB
*				0		

Registro: 1 de 3

10. Acaba de crear una consulta dentro de Access usando el lenguaje SQL. Para modificar de nuevo la consulta SQL tendrá que desplegar el botón *Vista* y activar de nuevo la opción SQL:

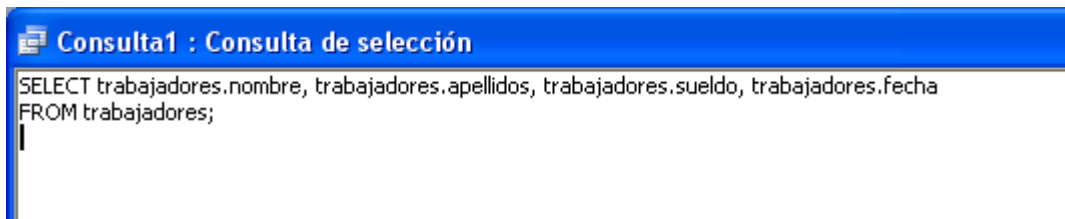


11. De esta manera entrará de nuevo en el editor de consultas SQL de Access, donde aún permanecerá la consulta creada:

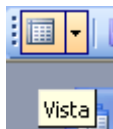


12. Ahora modificaremos la consulta para que solo nos muestre los campos *nombre*, *apellidos*, *sueldo* y *fecha*. Para ello tendrá que escribir la siguiente consulta:

```
SELECT trabajadores.nombre, trabajadores.apellidos, trabajadores.sueldo, trabajadores.fecha  
FROM trabajadores;
```



13. Usa el botón *Vista* para ver el resultado de la consulta. Observa como ahora solo vemos los campos indicados:



Consulta1 : Consulta de selección

	nombre	apellidos	sueldo	fecha
▶	Juan	Pérez	1120	04/05/2002
	Ana	Ruiz	1200	02/03/2002
	Francisco	López	1000	01/06/2006
*			0	

Registro: 1 de 3

14. Vuelve a la zona de edición de consultas SQL y cambia la consulta para que quede así:

```
SELECT trabajadores.nombre, trabajadores.apellidos, trabajadores.sueldo, trabajadores.fecha
FROM trabajadores
WHERE trabajadores.apellidos LIKE '*ez';
```

Como ves, hemos añadido una cláusula *where*, para buscar solo aquellos trabajadores cuyo apellido termine en *ez*.

15. Comprueba el resultado:

Consulta1 : Consulta de selección

	nombre	apellidos	sueldo	fecha
▶	Juan	Pérez	1120	04/05/2002
	Francisco	López	1000	01/06/2006
*			0	

Registro: 1 de 2

16. Vuelve a la zona de edición SQL. Vamos a complicar un poco la consulta.

Supongamos que queremos ver los servicios que han realizado cada uno de estos trabajadores. Para ello tendremos que incluir a la tabla *servicios* en la consulta.

Supongamos que queremos ver el tipo de servicio realizado y el coste de cada servicio. Por otro lado, de cada trabajador solo queremos ver el nombre y los apellidos.

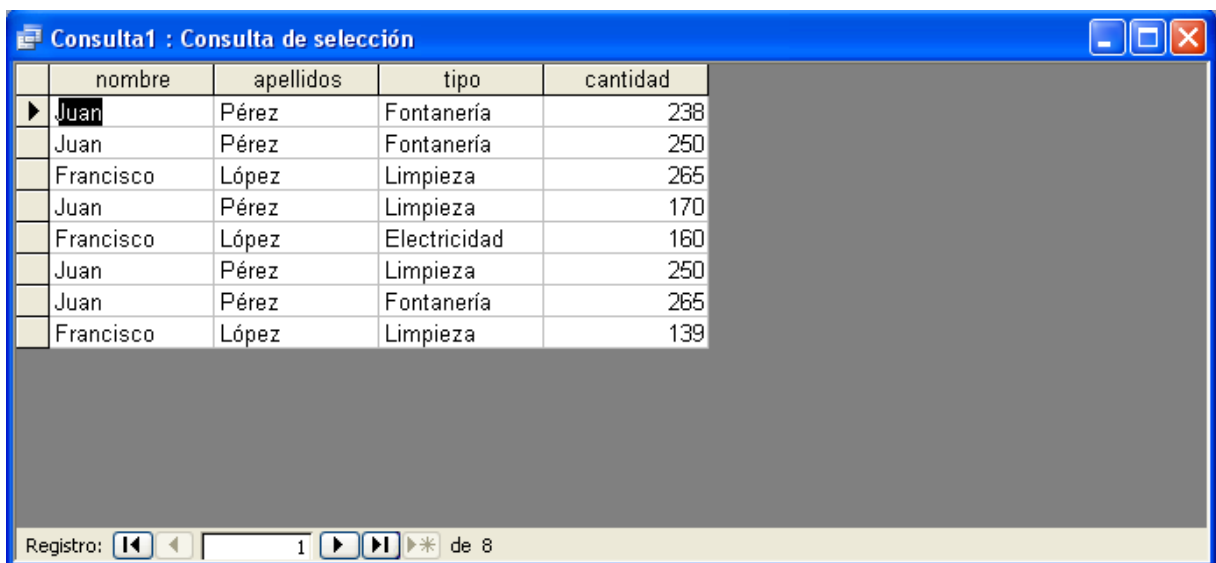
Hay que tener en cuenta que es necesario añadir en el WHERE una condición que iguale el campo de unión entre la tabla *trabajadores* y la tabla *servicios*. Este campo de unión es el DNI.

La consulta debe quedar así:

```
SELECT trabajadores.nombre, trabajadores.apellidos, servicios.tipo, servicios.cantidad
FROM trabajadores, servicios
WHERE trabajadores.apellidos LIKE '*ez' AND trabajadores.DNI=servicios.DNI;
```

(Observa la inclusión en el FROM de la tabla *servicios*, y como se ha añadido una condición de unión entre la tabla *trabajadores* y *servicios* a través del campo DNI)

17. Visualiza el resultado de la consulta (botón *Vista*) Aparecerá lo siguiente:



nombre	apellidos	tipo	cantidad
Juan	Pérez	Fontanería	238
Juan	Pérez	Fontanería	250
Francisco	López	Limpieza	265
Juan	Pérez	Limpieza	170
Francisco	López	Electricidad	160
Juan	Pérez	Limpieza	250
Juan	Pérez	Fontanería	265
Francisco	López	Limpieza	139

Como ves, aparecen todos los servicios y el nombre y apellidos del trabajador que hizo cada uno. Solo aparecen los servicios de los trabajadores cuyo apellido termine en 'ez'.

18. Vuelve a la vista de edición SQL. Vamos a complicar aún más la consulta. Esta vez vamos a hacer que aparezca el nombre del cliente al que se le hizo el servicio.

Esto implica la inclusión de una nueva tabla (*clientes*) y la necesidad de relacionar la tabla *servicios* con la tabla *clientes* según el modelo E-R de esta base de datos. El campo de unión es el CIF.

Modifica la consulta para que quede de la siguiente forma:

```
SELECT trabajadores.nombre, trabajadores.apellidos, servicios.tipo, servicios.cantidad,
clientes.nombre
FROM trabajadores, servicios, clientes
WHERE trabajadores.apellidos LIKE '*ez' AND trabajadores.DNI=servicios.DNI AND
servicios.CIF=clientes.CIF;
```

(Observa la inclusión de la tabla *clientes* en la cláusula FROM, y la adición de una nueva condición de unión, entre la tabla *servicios* y la tabla *clientes* en la cláusula WHERE. Esta condición de unión une ambas tablas usando el campo CIF)

19. Veamos el resultado de la consulta:



trabajadores.no	apellidos	tipo	cantidad	clientes.nombre
Juan	Pérez	Fontanería	238	Academia La Plata
Juan	Pérez	Fontanería	250	Seguros Segasa
Francisco	López	Limpieza	265	Seguros La Paz
Juan	Pérez	Limpieza	170	Seguros Cruces
Francisco	López	Electricidad	160	Academia de Pintura La Vid
Juan	Pérez	Limpieza	250	Seguros Cruces
Juan	Pérez	Fontanería	265	Academia de Pintura La Vid
Francisco	López	Limpieza	139	Seguros Cruces

Como se puede observar, aparece el listado de los servicios realizados por los trabajadores cuyo apellido termine en 'ez'. Aparece información incluida en tres tablas distintas: *trabajadores*, *servicios* y *clientes*.

20. Vuelve a la zona de diseño SQL. Añadiremos una última modificación a la consulta y la finalizaremos.

Ahora estableceremos un orden. Concretamente, ordenaremos el resultado de la consulta según apellidos del trabajador. Para ello tendremos que añadir una cláusula ORDER BY:

```
SELECT trabajadores.nombre, trabajadores.apellidos, servicios.tipo, servicios.cantidad,
clientes.nombre
FROM trabajadores, servicios, clientes
WHERE trabajadores.apellidos LIKE '*ez' AND trabajadores.DNI=servicios.DNI AND
servicios.CIF=clientes.CIF
ORDER BY trabajadores.apellidos ASC;
```

21. Ejecuta la consulta y observa el resultado. Ahora el listado de servicios aparece ordenado según el apellido del trabajador que lo realiza, alfabéticamente.



trabajadores.no	apellidos	tipo	cantidad	clientes.nombre
Francisco	López	Limpieza	139	Seguros Cruces
Francisco	López	Electricidad	160	Academia de Pintura La Vid
Francisco	López	Limpieza	265	Seguros La Paz
Juan	Pérez	Fontanería	265	Academia de Pintura La Vid
Juan	Pérez	Limpieza	250	Seguros Cruces
Juan	Pérez	Limpieza	170	Seguros Cruces
Juan	Pérez	Fontanería	250	Seguros Segasa
Juan	Pérez	Fontanería	238	Academia La Plata

22. Hemos finalizado con la consulta. Círrrela y guárdela. Puede ponerle de nombre *Servicios realizados por trabajadores ez*.
23. Este ha sido un ejemplo de consulta realizada en Access usando el lenguaje de consulta SQL. Este lenguaje será vital para hacer que nuestra aplicación java manipule la base de datos correspondiente.

NOTA.

En Internet se pueden encontrar múltiples manuales sobre SQL. Se recomienda buscar información sobre el tema para ampliar los conocimientos aquí expuestos.

La sintaxis del lenguaje SQL puede variar según el programa de base de datos que se use. En este ejercicio guiado se ha usado la sintaxis del SQL incluido en el programa Access. Tenga en cuenta que puede haber variaciones entre este SQL y el incluido en otro gestor de base de datos, aunque estas variaciones son mínimas.

CONCLUSIÓN

El lenguaje de consulta SQL permite manipular la base de datos.

A través de este lenguaje, se pueden crear órdenes que permitan:

- Introducir nuevos datos en la base de datos.**
- Eliminar datos de la base de datos.**
- Modificar datos de la base de datos.**
- Realizar consultas sobre la base de datos.**

Para realizar consultas en la base de datos usando el lenguaje SQL es necesario usar instrucciones SELECT, las cuales tienen la siguiente forma general:

SELECT campos a visualizar

FROM tablas a las que pertenecen dichos campos

WHERE condiciones a cumplir, condiciones de unión de tablas

ORDER BY campos por los que se ordena

El programa ACCESS permite crear consultas usando el lenguaje SQL, por lo que puede ser una buena herramienta para practicar este lenguaje.

Nuestro programa de bases de datos java, usará este lenguaje para manipular la base de datos correspondiente.