FUNDAMENTOS DE BASE DE DATOS.

Tema Nº8:Lenguaje para la manipulación de datos DML I.

Indicador de logro Nº8:Emplea sentencias optimizadas para Ingresar, Eliminar, Actualizar y Consultar datos de tablas.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº8:**

Lenguaje para la manipulación de datos DML I.

**Subtema 8.1:**

Insertar, actualizar, eliminar y consultar datos: INSERT, UPDATE y DELETE.

**¿Qué es DML?**

* El lenguaje de manipulación de datos (DML) es un conjunto de comandos utilizado para recuperar y trabajar con datos en base de datos. La mayoría también funciona en Data Warehouse.
* Estos comandos son usados para agregar, modificar, consultar o eliminar datos de una base de datos.

**IMPORTANTE**

Las operaciones de inserción, modificación, eliminación y consultas, solo trabajan con los datos de las tablas, mas no con la estructura. Es decir, la estructura de la tabla permanece intacta.

**COMANDOS DML:**

* INSERT
* DELETE
* UPDATE
* SELECT

**COMANDO INSERT INTO.**

* Permite la inserción de nuevos datos en una tabla, se realiza añadiendo filas enteras a la tabla.
* La inserción se puede realizar de una fila o de varias filas a la vez. Veremos las dos opciones por separado y empezaremos por la inserción de una fila.

SINTAXIS:

**FORMATO 1**

Inserción de datos sobre todas las columnas que tenga la tabla:

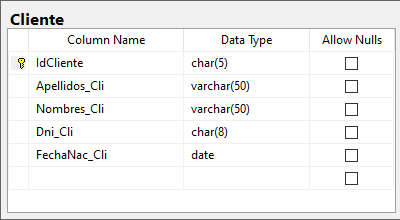
**INSERT INTO <tabla> VALUES (valor1, valor2, …, valor(n))**

**FORMATO 2**

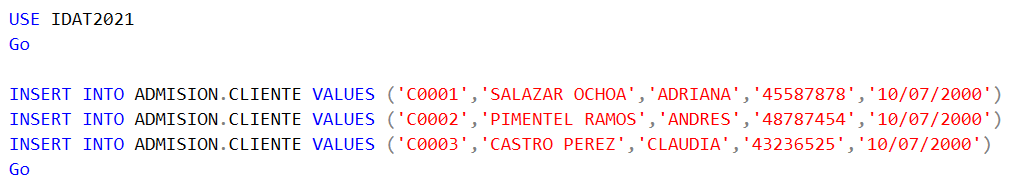
Inserción de datos sobre columnas específicas de la tabla.

**INSERT INTO <tabla> (col1, col2, …, col(n) VALUES (valor1, valor2, …, valor(n))**

Por ejemplo, tengo la tabla **CLIENTE** en el esquema **ADMISION**, con la siguiente estructura:



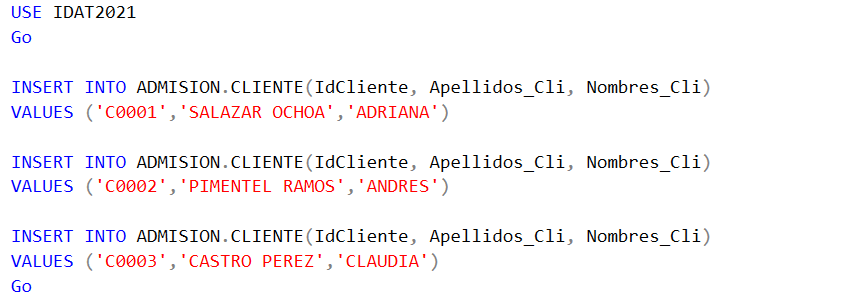
**Usando el formato 1**, insertar 3 registros en la tabla Cliente:



**IMPORTANTE:**

Como se observa se está insertando datos sobre todas las columnas de la tabla.

**Usando el formato 2**, insertar 3 registros en la tabla Cliente, específicamente en las columnas **IdCliente**, **Apellidos\_Cli**, **Nombres\_Cli**,



**IMPORTANTE:**

Como se observa se está insertando datos sobre ciertas columnas de la tabla CLIENTE, para que esto pueda funcionar correctamente, las otras columnas que no se han usado, deberán permitir NULL.

**COMANDO DELETE - FROM**

* Para borrar datos de una tabla, debemos utilizar la sentencia DELETE.
* Su sintaxis es la siguiente: **DELETE FROM** <Nombre\_Tabla> [**WHERE** {condición}]
* Dónde: Nombre\_Tabla nombre de la tabla donde se desea borrarlos datos. La cláusula **WHERE** sigue el mismo formato que la vista en la sentencia SELECT y determina qué registros se borrarán.

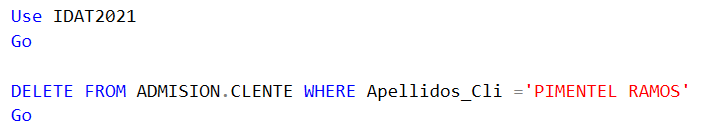
**Ejemplo 1:**

Borrar de la tabla **CLIENTE** del esquema **ADMISION**, todos sus registros.



**Ejemplo 2:**

Eliminar de la tabla **CLIENTE** del esquema **ADMISION**, todos aquellos clientes que sus apellidos sea ‘**PIMENTEL RAMOS’**.



**COMADO TRUNCATE**

Realiza un borrado completo de tabla, debemos considerar la posibilidad de utilizar la sentencia **TRUNCATE**, mucho más rápida que **DELETE**.

Su sintaxis es la siguiente: **TRUNCATE TABLE** <Nombre\_Tabla>

**A CONSIDERAR:**

* TRUNCATE no es transaccional. No se puede deshacer
* TRUNCATE no admite el uso de la cláusula **WHERE**. Borra toda la tabla

**Ejemplo 3:**

Eliminar los datos de la tabla **TRABAJADOR** del esquema **ADMISION.**



**COMANDO UPDATE.**

* Para la actualización de datos SQL dispone de la sentencia **UPDATE**
* La sentencia **UPDATE** permite la actualización de uno o varios registros de una única tabla, uso de la cláusula **WHERE**.

**SINTAXIS:**

**UPDATE** <Nombre\_tabla> **SET** nombre\_columna1 = expr1,

nombre\_columna2 = expr2,

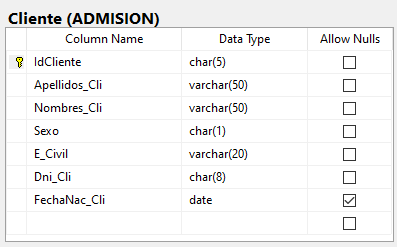
……

……

nombre\_columna(n) = expr(n)

[**WHERE** < condición>]

**Por Ejemplo**, se tiene la siguiente tabla en el esquema **ADMISION**.



**Datos de la tabla CLIENTE.**



**Ejemplo 1:** modificar una fila completa de datos de la tabla Cliente en el esquema ADMISION.



En este caso elegimos al registro de CASRO PEREZ, los cambios son los siguientes:

Nuevo Apellido : **ALCANTARA RAMOS**

Nuevo Nombre : **JUAN JOSE**

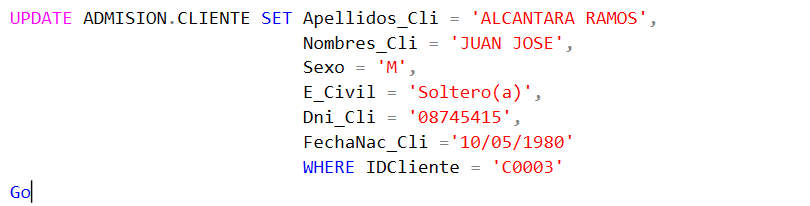
Nuevo Género : **M**

Nuevo Estado Civil : **Soltero(a)**

Nuevo Dni : **08745415**

Nueva Fecha de Nacimiento: **10/05/1980**

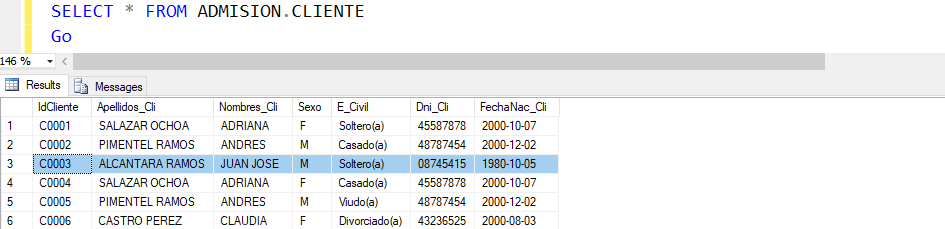
**SOLUCIÓN:**



La ejecución devuelve el siguiente mensaje:

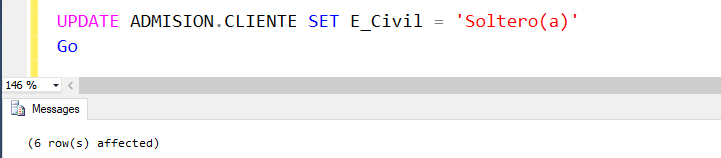


Verificando el cambio de datos:

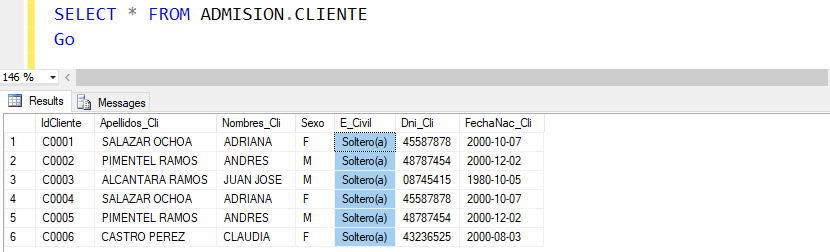


**Ejemplo 2:** Cambiar el estado civil de todos los clientes, por Soltero(a).

**SOLUCION:**

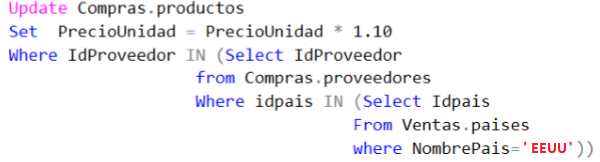


**Verificando los cambios:**



**EJEMPLO BASADO EN MULTI CONSULTA.**

**Ejemplo:** Actualizar el precio de los productos incrementando 10%, solo si han sido suministrados por proveedores de EEUU.



**Subtema 1.2:**

Recuperación de datos, empleo de la sentencia SELECT y cláusulas de Ordenamiento de registros ORDEBR BY, condiciones de datos WHERE, operadores lógicos (AND, OR y NOT), operadores de comparación <, <, =, <>, <=, <=, operador para el manejo de cadenas LIKE, Otros operadores IN, BETWEEN.

**Comando SELECT**

* Lenguaje de base de datos normalizado, utilizado por el motor de base de datos de Microsoft SQL Server.
* Se utiliza para indicar al motor de datos que devuelva información de las bases de datos en forma de conjunto de registros.
* La información es devuelta en forma de conjunto de registros que se pueden almacenar en un objeto Recordset.

**La sintaxis del comando SELECT es la siguiente:**

**SELECT** [**\***] [**DISTINCT**] [**TOP**]

**FROM** <Tabla(s)>

**WHERE** <Condición> [**AND**][**OR**]

**GROUP BY** <Nombre de Campos> **HAVING** <Condición> [**AND**][**OR**]

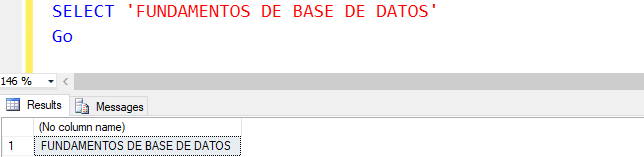
**ORDER BY** [Campo orden]

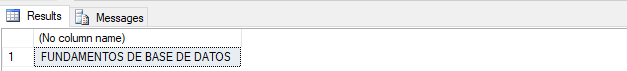
El comando SELECT utiliza predicados y cláusulas.

**Predicados**: \*, DISTINCT, TOP

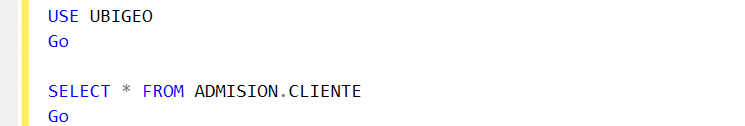
**Claúsulas**: FROM, WHERE, GROUP BY, HAVING, ORDER BY

**COMANDO SELECT**, Consulta sencilla.



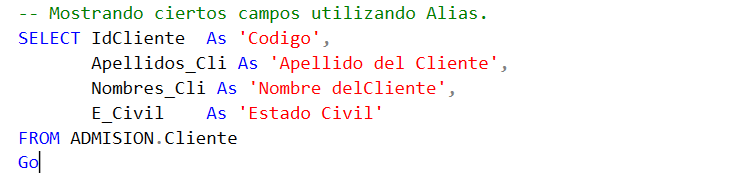


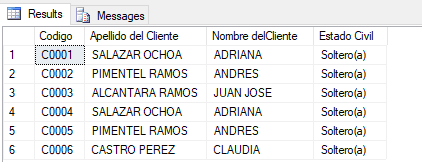
**COMANDO SELECT**, Consulta sencilla.





**COMANDO SELECT**, Consulta sencilla.





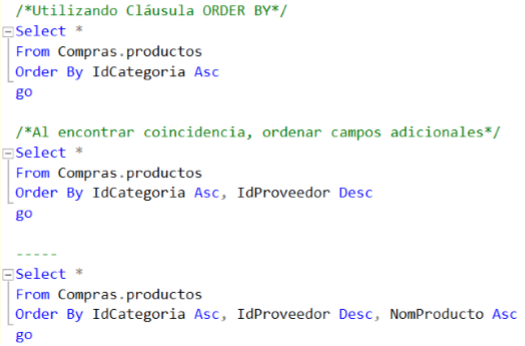
**ORDENAR REGISTROS.**

Se puede especificar el orden en que se desean recuperar los registros de las tablas mediante la cláusula:

**Sintaxis:**

**SELECT** \* **FROM** • **ORDER BY** [lista de campos] [**Asc** | **Desc**]

Se puede hacer ordenamiento con más de un campo cuando se encuentren coincidencias.



**PREDICADO.**

Consulta donde el predicado se incluye entre el comando **SELECT** y el primer nombre del campo a recuperar. son los siguientes:

|  |  |
| --- | --- |
| **Predicado** | **Descripción** |
| **ALL** | Devuelve todos los campos de la tabla. |
| **TOP** | Devuelve las primeras N especificaciones de filas de datos de la tabla. |
| **DISTINCT** | Omite duplicidad de datos. |

**CONSULTAS CON USO DE PREDICADOS.**

**PREDICADO ALL:**

* Si no se incluye ninguno de los predicados se asume ALL
* Este predicado obliga al motor de la base de datos a analizar la estructura de la tabla para averiguar los campos que contiene.
* Es más rápido listar los campos deseados.

**EJEMPLO: Muestra todos los campos de la tabla CLIENTE.**

**SELECT** **ALL** \* **FROM** ADMISION.CLIENTE

Go

**PREDICADO TOP:**

* Devuelve un cierto número de registros que entran al principio o al final de un rango especificado por una cláusula ORDER BY.

**EJEMPLO: Muestra a los 3 primeros clientes más jóvenes.**

**SELECT TOP 3 FROM** ADMISION**.**CLIENTE

**ORDER BY** FechaNac\_Cli **DESC**

**Go**

**PREDICADO DISTINCT.**

Omite los registros que contienen datos duplicados en los campos seleccionados.

**SELECT DISTINCT IdCliente FROM ADMISION.CLIENTE**

**Go**

**CONDICIONES EN LAS CONSULTAS.**

**CLAUSULA WHERE.**

* Determina qué registros de las tablas enumeradas en la cláusula FROM aparecerán en los resultados de la instrucción SELECT.
* En esta cláusula se deben especificar las condiciones de los datos a obtener.
* Si no se emplea esta cláusula, la consulta devolverá todas las filas de la tabla

**QUE OPERADORES SE PUEDE EMPLEAR EN LA CLAUSULA WHERE.**

* Operadores de Comparación o Relacionales
  + **> Mayor que.**
  + **< Menor que.**
  + **= Igual que.**
  + **>= Mayor e igual que.**
  + **<= Menor e igual que**
  + **< >** o **!=** **Diferencia**
* Operadores Lógicos **AND**, **OR**, **NOT**
* Operadores: Listas de valores. **IN**
* Operadores Aritméticos
  + +Suma.
  + - Resta.
  + \* Multiplicación.
  + / División.

**EJEMPLO DE CONSULTAS USANDO OPERADORES.**

Muestra los productos cuyo stock este comprendido entre 2 y 20.

**SELECT \* FROM PRODUCTO WHERE STOCK >= 2 AND STOCK <= 20**

Muestra a los Clientes cuyas edades se encuentren entre 18 y 25 años.

**SELECT \* FROM CLIENTE WHERE EDAD >= 18 AND EDAD <=25**

Muestra los libros de los géneros DRAMA o NOVELAS.

**SELECT \* FROM LIBRO**

**WHERE GENERO = ‘DRAMA’ OR GENERO=’NOVELAS’**

Muestra el nombre del empleado, su sueldo y el 50% del sueldo de todos los empleados.

**SELECT NOMBRE, SUELDO, SUELDO\*0.5 FROM EMPLEADO**

Muestra los productos que superen el precio de 2500.00

**SELECT \* FROM PRODUCTO WHERE PRECIO > 2500**

Mostrar a los empleados que sean SOLTEROS, VIUDOS y CASADOS.

**SELECT \* FROM EMPLEADO**

**WHERE E\_CIVIL IN (‘SOLTERO’, ‘VIUDO’,’CASADO’)**

**OPERADOR BETWEEN.**

Es un operador de intervalo.

**EJEMPLOS:**

Mediante una consulta mostrar los productos cuyos precios estén comprendidos entre 100 y 500.

**SELECT \* FROM PRODUCTO**

**WHERE PRECIO BETWEEN 100 AND 500**

Mediante una consulta mostrar los productos cuyos precios no estén comprendidos entre 100 y 500.

**SELECT \* FROM PRODUCTO**

**WHERE PRECIO NOT BETWEEN 100 AND 500**

**OPERADOR IN – NOT IN.**

Permite seleccionar las filas que coinciden con alguno de los valores de una lista

**EJEMPLOS:**

Mediante una consulta, mostrar a los empleados SOLTEROS, CASADOS y DIVORCIADOS.

**SELECT \* FROM EMPLEADO**

**WHERE E\_CIVIL IN (‘SOLERO’, ‘CASADO’,’DIVORCIADO’**

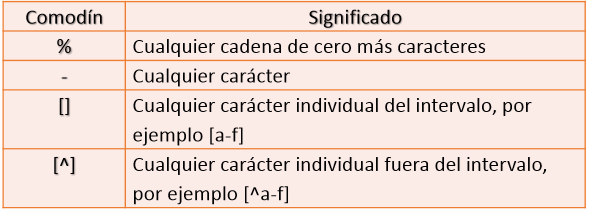
Mediante una consulta, mostrar a los empleados no sean SOLTEROS ni CASADOS.

**SELECT \* FROM EMPLEADO**

**WHERE NOT IN (‘SOLTERO’,’CASADO’)**

**OPERADOR DE COINCIDENCIA DE PATRON LIKE.**

* Permite buscar valores de cadenas de caracteres que coincidan con un patrón determinado.
* El operador de coincidencias es LIKE, la cual está compuesta por 4 caracteres comodín.



**EJEMPLOS:**

Muestre a los clientes cuyo nombre inicie con la letra “P”.

**SELECT \* FROM CLIENTE WHERE NOMBRE LIKE ‘P%’**

Muestre los empleados cuyo nombre empiece con la letra “A”, “B”,”C”.

**SELECT \* FROM EMPLEADO WHERE NOMBRE LIKE ‘[A-C]%’**

Muestre a los empleados cuyo nombre no empiecen con las letras “A”, “B”,”C”.

**SELECT \* FROM EMPLEADO WHERE NOMBRE LIKE ‘[^A-C]%’**

**Subtema 1.3:**

Funciones agregadas: SUM, MIN, MAX, AVG, COUNT, Agrupaciones de datos: GROUP BY y HAVING, Creación y ejecución de vistas.

**INTRODUCCIÓN A LAS FUNCIONES AGREGADAS O AGREGACIÓN.**

Una **función de agregación** es una función que genera un resume de las filas de un grupo en un solo valor, estos pueden ser: **SUM**, **AVG**, **COUNT**, **MIN** y **MAX** son ejemplos de funciones de agregación.

Estas funciones solo devuelven 1 solo resultado.



**FUNCIÓN SUM:**

Obtiene la sumatoria de los valores de una columna numérica de la tabla.

Sintaxis

**SUM**(Columna)

**FUNCIÓN AVG:**

Obtiene el promedio aritmético de los valores de una columna numérica de la tabla.

Sintaxis:

**AVG**(Columna)

**FUNCIÓN MAX:**

Obtiene el máximo de los valores de una columna numérica de la tabla.

Sintaxis:

**MAX**(Columna)

**FUNCIÓN MIN:**

Obtiene el mínimo de los valores de una columna numérica de la tabla.

Sintaxis:

**MIN**(Columna)

**FUNCIÓN COUNT:**

Obtiene el conteo de los valores de una columna de la tabla.

Sintaxis:

**COUNT**(Columna)

**EJEMPLOS:**

**SELECT SUM(Sueldo) FROM TRABAJADOR**

**SELECT AVG(Sueldo) FROM TRABAJADOR**

**SELECT MAX(Sueldo) FROM TRABAJADOR**

**SELECT MIN(Sueldo) FROM TRABAJADOR**

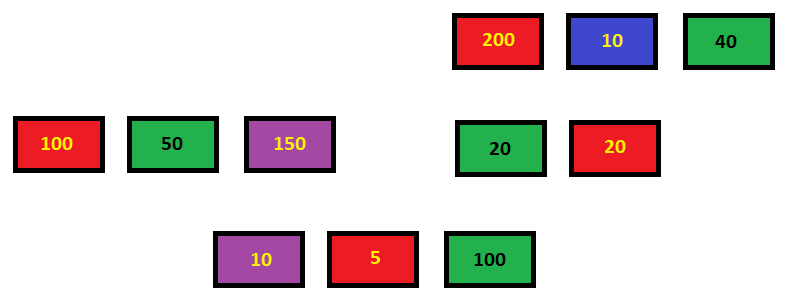
**SELECT COUNT(Sueldo) FROM TRABAJADOR**

**AGRUPACIONES: GROUP BY**

La GROUP BY declaración agrupa las filas que tienen los mismos valores en filas de resumen, como "encontrar el número de clientes en cada país".

La GROUP BY instrucción se utiliza a menudo con funciones de agregado COUNT(), MAX(), MIN(), SUM(), AVG() a grupo el conjunto de resultados de una o más columnas.

**PARA ENTENDER MEJOR QUE ES UN GRUPO, OBSERVA LA SIGUIENTE IMAGEN:**

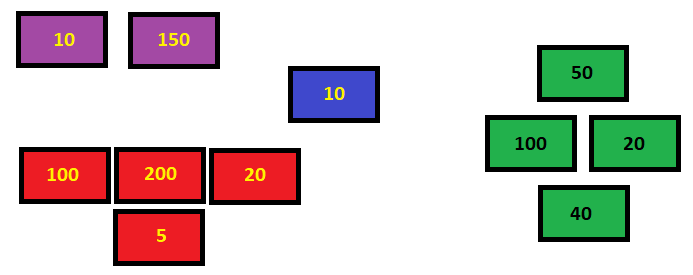


Se tienen varias cajitas de colores, ¿podemos adivinar cuantos grupos por coles existen?

Respuesta: Sí.

Cajas Verdes, Cajas Moradas, Cajas Rojas y una Caja Azul.

**Entonces las clasificamos:**



**Como resultado obtendremos 4 grupos de colores.**

**¿Qué podemos hacer con los grupos de colores?**

Respuesta:

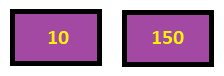
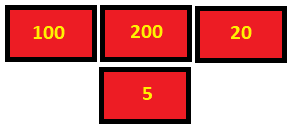
* Sumarlos.
* Contarlos.
* Obtener el Mayor.
* Obtener el Menor.
* Promediarlos.

**Por ejemplo:**

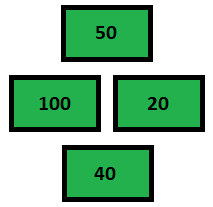
**DESDE LUEGO EL GRUPO ES COLOR.**

Sumar Morados Sumar los Rojo. Sumar Azules

Respuesta: 160 Respuesta: 325 Respuesta: 10



Sumar los Verdes.

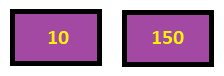
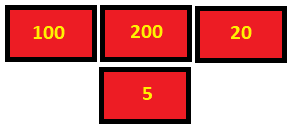
Respuesta: 210

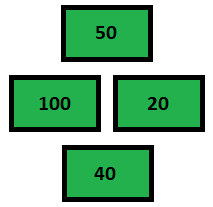
**Ejemplo:**

**DESDE LUEGO EL GRUPO ES COLOR.**

Contar Morados Contar los Rojo. Contar Azules

Respuesta: 2 Respuesta: 5 Respuesta: 1



Contar los Verdes.

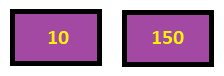
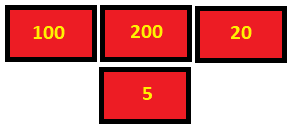
Respuesta: 4

**Ejemplo:**

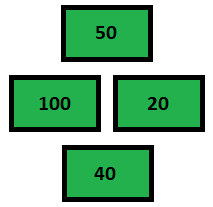
**DESDE LUEGO EL GRUPO ES COLOR.**

El Máximo Morados El Máximo Rojo. El Max Azul

Respuesta: 150 Respuesta: 200 Respuesta: 10



El Máximo de Verdes.

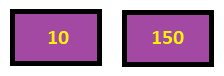
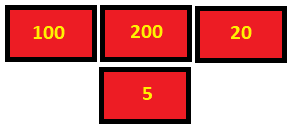
Respuesta: 100

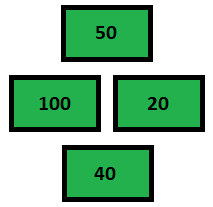
**Ejemplo:**

**DESDE LUEGO EL GRUPO ES COLOR.**

Mínimo de Morados Mínimo Rojo. Mínimo Azul

Respuesta: 10 Respuesta: 5 Respuesta: 10





Mínimo Verdes.

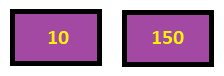
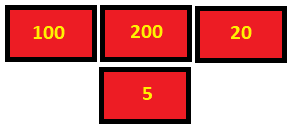
Respuesta: 20

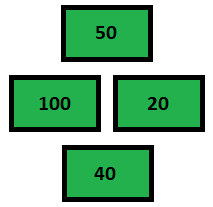
**Ejemplo:**

**DESDE LUEGO EL GRUPO ES COLOR.**

Promedio Morados Promedio Rojo. Promedio Azul

Respuesta: 80 Respuesta: 81.25 Respuesta: 10



Promedio Verdes.

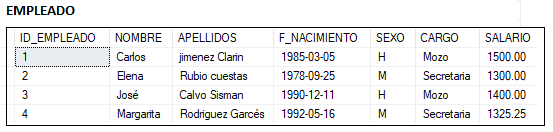
Respuesta: 52,5

Sintaxis:

**SELECT** column\_name(s)  
**FROM** table\_name  
**WHERE** condition  
**GROUP BY** column\_name(s)

http://deletesql.com/viewtopic.php?f=5&t=13

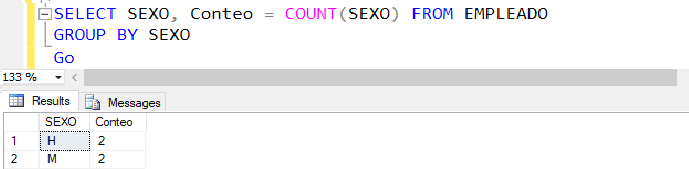
Para demostrar el uso de las agrupaciones, tenemos a la siguiente tabla.



**EJERCICIOS BASADOS EN GRUPOS**

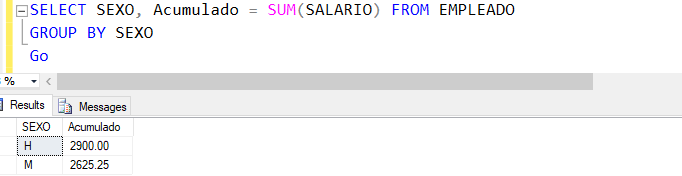
**Ejercicio 1:**

Contabilizar cuantos varones y damas existen.



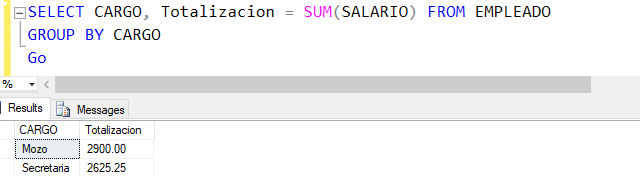
**Ejercicio 2:**

Obtener el total acumulado de sueldos, entre varones y damas.



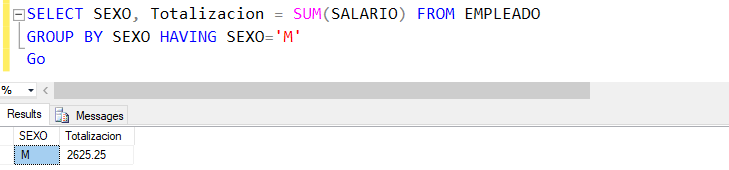
**Ejercicio 3:**

Totalizar los salarios por cargos.



**Ejercicio 4:**

Totalizar los sueldos por el género del empleado. Solo para el caso de las damas.



VISTAS:

Una vista es una alternativa para mostrar datos de varias tablas. Una vista es como una tabla virtual que almacena una consulta. Los datos accesibles a través de la vista no están almacenados en la base de datos como un objeto.

SINTAXIS:

**CREATE VIEW <Nombre Vista>**

**AS**

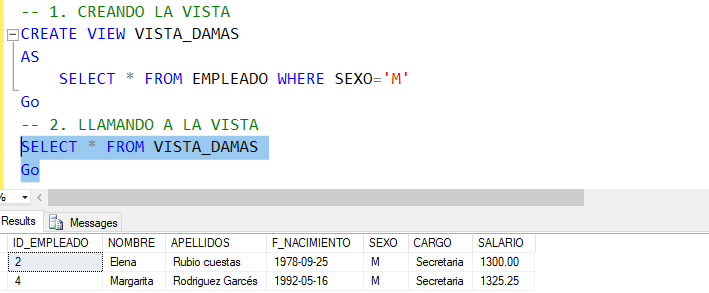
**[Consulta SQL SELECT]**

**Ejemplo 1:**

Crear una vista el cual permita mostrar los datos de todos los empleados.



**Ejemplo 2**: Crear una vista el cual permita consultar solo a las damas.



**CONCLUSIONES Y RECOMENDACIONES DE LA EXPERIENCIA**

* Las sentencias DML sirven para seleccionar, insertar, modificar, eliminar datos en nuestras tablas.
* Comprende el Lenguaje de Manipulación de Datos (DML) utilizando el lenguaje de manipulación de datos SELECT, INSERT, UPDATE y DELETE.

**ACTIVIDAD VIRTUAL**

Analizar y revisar los siguientes enlaces; luego desarrolla las actividades propuestas

<https://www.youtube.com/watch?v=fJqU1WGjGVo>

<https://www.youtube.com/watch?v=gPVb9Dq9K38>

* Mencionar las cláusulas del lenguaje de manipulación de datos
* Mencionar los comandos del lenguaje de manipulación de datos
* Mencionar cuáles son los operadores de comparación