Base de Datos

Unidad 2

Lenguaje SQL

SENTENCIAS DML

Select y Where

Introducción al SQL - SQL DML

- Sentencias de Manipulación de Datos

SELECT	INSERT
UPDATE	DELETE

Sentencia SELECT – Estructura Básica

SELECT ..... FROM .... WHERE

- SELECT corresponde a la operación Proyección (Π). Lista los atributos deseados en el resultado de la consulta.
- FROM corresponde a la operación Producto Cartesiano (X). Lista las relaciones que se deben analizar.
- WHERE corresponde al predicado Selección (σ), y se refiere a los atributos de las relaciones listadas en FROM.

Sentencia SELECT

[SELECT ...]
    [INTO ...]
[FROM ...]
[WHERE ...]
[GROUP BY …]
[HAVING ...]
[ORDER BY ...]
    [UNION ...]

Cláusula SELECT

- En la cláusula SELECT se especifican las columnas que van a ser mostradas en pantalla

SELECT [all | distinct]

{ * | tabla.* | [tabla.]campo1 [encab1] ,
[tabla.]campo2 [encab2] , ...}

Cláusula FROM

- Específica la(s) fuente(s) de donde van a ser tomadas las columnas mostradas y/o operadas en la sentencia Select.

[SELECT…]

FROM expresión tabla [as] t_alias [, ...]

Cláusula INTO

- La cláusula INTO, permite guardar los resultados intermedios de una consulta

```
[SELECT ...] INTO tabla_into
[FROM ...]
```

Cláusula WHERE

- Donde se especifican las condiciones de búsqueda y join para las filas que conforman el conjunto resultado.

  [SELECT ...]

  [FROM ...]

  WHERE *condiciones_de_búsqueda*

- Tipos de condiciones
  - Condiciones de comparación, de Join, o de Subquery.

Ejemplos de la Sentencia SELECT

SELECT distinct Cempleado, Fnacimiento, Ddireccion

FROM EMPLEADO
WHERE Dnombre = “Juan” AND Dapellido = “Perez”

$$\Pi_{\text{fecha\_nac, direccion}} (\sigma_{\text{nombre} = \text{“Juan”} \text{ AND } \text{apellido} = \text{“Perez”}} (\text{EMPLEADO}))$$

Nota: Los campos del where no son necesariamente los mismos campos de la consulta

Ejemplos de la Sentencia SELECT

Considerando que existe una tabla de los prestatarios y de los prestamos

'Mostrar la relación de clientes que han obtenido un préstamo en alguna sucursal de Miraflores'

```
SELECT    distinct  Dcliente
FROM      PRESTATARIO PRS, PRESTAMO PTR
WHERE     PRS.Cprestamo = PRT.Nprestamo
                AND  Dsucursal = "Miraflores"
```

Condiciones de Búsqueda

Las condiciones de búsqueda pueden incluir:

- ✓ Operadores de Comparación (=, <>, < y >).
- ✓ Porciones de cadenas de caracteres (SUBSTR)
- ✓ Rangos (BETWEEN y NOT BETWEEN).
- ✓ Listas (IN, NOT IN).
- ✓ Patrones de caracteres (LIKE y NOT LIKE).
- ✓ Valores desconocidos (IS NULL y IS NOT NULL).
- ✓ Combinaciones con conjunciones (AND, OR).

Condiciones de Búsqueda

- Operadores de comparación:
  - WHERE SueldoBásico > 1000

Porciones de cadenas de caracteres (SUBSTR)

  - WHERE SUBSTR(CodigoPostal, 1, 3)= “SAN”
- Rangos (BETWEEN / NOT BETWEEN):
  - WHERE SueldoBásico BETWEEN 1000 AND 5000
- Listas (IN / NOT IN):
  - WHERE Departamento NOT IN ('ADM', 'SIS')
- Patrones de caracteres (LIKE / NOT LIKE):
  - WHERE NombreCompleto LIKE “LET%”
- Valores desconocidos (IS NULL y IS NOT NULL) :
  - WHERE FechaDespacho IS NULL

Comodines (Wildcards) en la cláusula WHERE EN LA SENTENCIA LIKE

Wildcard	Significado
* %	Cualquier cadena de cero o mas caracteres.
?, #, _	Cualquier carácter/ número único.
[-]	Cualquier carácter único dentro de un rango.
[!]	Cualquier carácter único que no está dentro de un rango.

Operadores y Wildcards en la cláusula WHERE

….WHERE Nombre

- LIKE “Ma%” busca todos los nombres que comiencen con “Ma”
  (Ej.: María, Mariana, Manuel, Martín)

- LIKE “%ía” busca todos los nombres que terminen con “ía”.
  (Ej.: Sofía, María, Estefanía).

- LIKE “%ar%” busca todos los nombres que tengan las letras “ar”.
  (Ej.:Carlos, Arturo, Eleazar).

Operadores y Wildcards en la cláusula WHERE

- LIKE “_va” busca todos los nombres de tres letras que terminan en “va”. (Ej.: Eva, Iva, Ava).
- LIKE “[CM]arlo[ns]” busca todos los nombres: Carlon, Marlon, Carlos y Marlos.
- LIKE “[B-D]elia” busca todos los nombres que terminan en “elia” y que comiencen con las letras de la B a la D. (Ej.: Delia, Celia).
- LIKE “_ _ _” busca todas las cadenas de exactamente 3 caracteres.
- LIKE “_ _ _ %” busca las cadenas de al menos 3 caracteres.

CARACTERES DE ESCAPE:

Para que los patrones puedan contener los caracteres especiales comodín, se especifica un caracter de escape.

- LIKE “%ab_cd” escape “\” busca todas las cadenas que terminen con “ab_cd”.
- LIKE “ab_cd%” escape “\” busca las cadenas que empiecen con “ab_cd”.

Cláusula GROUP BY

- Especifica las columnas por las que las filas van a estar agrupadas o particionadas. Los resultados del query contienen un valor o conjunto de valores para cada conjunto de valores indicado por las *funciones_de_agregación* nombradas en la lista del Select

Cláusula GROUP BY

SELECT {{columnas_de_agrupación,…},
{función_de_agregación,…}}

FROM …

[WHERE …]

GROUP BY {columnas_de_agrupación,…}

Funciones de Agregación más usadas

SUM([ALL|DISTINCT] expresión)	Calcula el total de una expresión numérica para todas las filas o sólo las distintas.
AVG([ALL|DISTINCT] expresión)	Calcula el promedio de una expresión numérica para las filas involucradas.
MIN([ALL|DISTINCT] expresión)	Calcula el mínimo valor de una expresión numérica para las filas involucradas
MAX([ALL|DISTINCT] expresión)	Calcula el máximo valor de una expresión numérica para las filas involucradas.
COUNT([ALL|DISTINCT] expresión)	Número de veces que se repite el valor de la expresión.
COUNT(*)	Número de filas seleccionadas

Cláusula GROUP BY

‘Mostrar la suma de sueldo básico de los empleados activos, por Departamento’

```
SELECT Nombre_Depto, SUM(SueldoBasico)
FROM  EMPLEADO EM, DEPARTAMENTO DE
WHERE      EM.IDDepartamento = DE.IDDepartamento
    AND
               EstadoEmpleado= 'ACT’
GROUP BY Nombre_Depto
```

Cláusula GROUP BY

"Obtener el número de titulares de cuenta de cada sucursal"

```
SELECT nombre_sucursal, COUNT ( DISTINCT
     nombre-cliente)
FROM  TITULAR-CUENTA TC, CUENTA CUE
WHERE      TC.numero-cuenta = CUE. numero-cuenta
GROUP BY nombre-sucursal
```

Cláusula HAVING

- Especifica una restricción que aplica a las funciones de agregación de los grupos. Esto afecta a las filas que son devueltas como resultado y **no** al cálculo de las funciones de agregación.
- La cláusula WHERE **si** condiciona el número de filas que intervienen en el cálculo de las funciones de agregación.

Cláusula HAVING

SELECT {{columnas_de_agrupación,…},
{función_de_agregación,…}}

FROM …

[WHERE …]

GROUP BY {columnas_de_agrupación,…}

HAVING condiciones_de_búsqueda

GROUP BY y HAVING

"Mostrar el sueldo promedio de los Departamentos con promedio superior a 1000"

```
SELECT Nombre_Depto, AVG(SueldoBasico)
FROM  EMPLEADO EM, DEPARTAMENTO DE
WHERE      EM.IDDepartamento =
     DE.IDDepartamento AND
                EstadoEmpleado= 'ACT'
GROUP BY Nombre_Depto
HAVING     AVG(SueldoBasico) > 1000
```

GROUP BY y HAVING

"Saldo promedio de cada cliente de Surco que tiene como mínimo 3 cuentas"

```
SELECT TC. Nombre-cliente, AVG (saldo)
FROM  TITULAR-CUENTA TC, CUENTA CUE, CLIENTE CLI
WHERE      TC.numero-cuenta = CUE.numero-cuenta
     AND
          TC.nombre-cliente = CLI.nombre-cliente AND
          ciudad-cliente = "Surco"
GROUP BY TC.nombre-cliente
HAVING     COUNT (DISTINCT TC.numero-cuenta) >= 3
```

GROUP BY - HAVING - WHERE

Procedimiento:

- Si en una misma consulta aparece una cláusula WHERE y una cláusula HAVING, se aplica primero el predicado de la cláusula WHERE.
- Las tuplas que satisfagan la condición, se colocan en grupos según la cláusula GROUP BY.
- La cláusula HAVING se aplica luego a cada grupo. Los grupos que no la satisfagan se eliminan.
- La cláusula SELECT utiliza los grupos restantes para generar las tuplas resultado de la consulta.

Cláusula ORDER BY

- Ordena el resultado de los querys por los valores de las columnas mencionadas. Solamente se puede ordenar por las columnas especificadas en el SELECT.
  - ASC: Es el valor por defecto e indica que los resultados se van a presentar ascendentemente.
  - DESC: Debe especificarse al lado de la columna cuyo orden se desea ver en forma descendente.
- *Consideración*: *ordenar un gran número de tuplas puede ser costoso. Es conveniente ordenar sólo cuando sea estrictamente necesario*.

Cláusula ORDER BY

SELECT {columna 1, columna 2,…}

FROM …

[WHERE …]

[GROUP BY...]

[HAVING ...]

ORDER BY columna 1 [ASC|DESC],....

Cláusula ORDER BY

```
SELECT    Nombre 'Departamento',
          AVG(SueldoBasico) 'Promedio Sueldos'
FROM      EMPLEADO EM, DEPARTAMENTO DE
WHERE     EM.Departamento = DE.Departamento
GROUP BY Nombre
ORDER BY AVG(SueldoBasico) DESC
```