

Bases de Datos I

Cursada 2008

Clase 4: Sentencia SELECT SQL básica

Facultad de Ciencias Exactas
Universidad Nac. Centro de la Pcia. de Bs. As.

Lenguaje de consulta

- En una base de datos relacional los datos son almacenados en estructuras de almacenamiento o tablas.
- Las dos operaciones básicas a llevar a cabo sobre una base de datos relacional son la recuperación de los datos almacenados, y el almacenamiento en sí o actualización de datos (inserción, eliminación o modificación de datos).
- En una base de datos relacional, tanto para la recuperación de datos como para la actualización de los mismos, se utiliza el lenguaje SQL (Structured Query Language)
- La sentencia del lenguaje utilizada para la recuperación de datos a partir de la base de datos es la **sentencia SELECT**

2

Sentencias SELECT Básicas

```
SELECT *|{[DISTINCT] column|expression [alias],...}  
FROM table;
```

- SELECT identifica las columnas – **EL QUE** -
 - FROM identifica la tabla - **DE DONDE** -
- Selección de todas las columnas**

```
SELECT *  
FROM voluntario;
```

Selección SOLO de algunas columnas

```
SELECT nombre, apellido, horas_aptadas  
FROM voluntario ;
```

3

Escritura de Sentencias SQL

- Las sentencias SQL no son sensibles a mayúsculas/minúsculas.
- Las sentencias SQL pueden ocupar una o más líneas.
- Las palabras clave no se pueden abreviar ni dividir entre líneas.
- Las cláusulas suelen estar colocadas en líneas separadas.
- Los sangrados se utilizan para mejorar la legibilidad.

4

Filas Duplicadas

- La visualización por defecto de las consultas son todas las filas, incluidas las filas duplicadas.

```
SELECT max_horas  
FROM tarea;
```

5

Eliminar los valores repetidos

- Cuando el resultado de una consulta dará valores repetidos puede ser deseable eliminar los valores repetidos y solamente quedarse con aquellos distintos.

```
SELECT DISTINCT max_horas  
FROM tarea;
```

- DISTINCT se aplica a todas las columnas de la lista de la cláusula SELECT

6

Uso de Operadores Aritméticos

```
SELECT nombre_tarea, max_horas - min_horas
FROM tarea;
```

Prioridad de los Operadores

*** / + -**

- La multiplicación y la división tienen prioridad sobre la suma y la resta.
- Los operadores de idéntica prioridad se evalúan de izquierda a derecha.
- Los paréntesis se utilizan para forzar evaluaciones prioritarias y para clarificar sentencias.

7

Prioridad de los Operadores

```
SELECT nombre_tarea, 10 * max_horas - min_horas
FROM tarea;
```

Uso de Paréntesis

```
SELECT nombre_tarea, 10*(max_horas - min_horas)
FROM tarea;
```

8

Cadenas de Caracteres Literales

- Un literal es un carácter, un número o una fecha incluida en la lista `SELECT`.
- Los valores literales de caracteres y fecha se deben escribir entre comillas simples.
- Cada cadena de caracteres tiene una salida para cada fila devuelta.

9

Operador de Concatenación

Un operador de concatenación:

- Concatena columnas o cadenas de caracteres a otras columnas.
- Está representado por dos barras verticales (||).
- Crea una columna resultante que es una expresión de caracteres.

```
SELECT apellido || ', ' || nombre  
FROM voluntario;
```

10

Uso de Alias de Columna

```
SELECT apellido || ', ' || nombre AS  
        "Apellido y Nombre"  
FROM voluntario;
```

11

Restringir filas recuperadas

- Se pueden restringir las filas recuperadas usando la cláusula WHERE.
- Una cláusula WHERE contiene una condición lógica, la cual usa operadores de comparación y/o lógicos.
- Las filas procesadas son aquellas en que los datos que contienen satisfacen la/s condición/es lógicas

```
SELECT [DISTINCT] {*, columna [alias] }  
FROM tabla  
[WHERE condicion/es];
```

12

Restringir filas recuperadas

- Por ejemplo, para recuperar los voluntarios de nombre 'Alberto'

```
SELECT *  
FROM voluntario  
WHERE nombre = 'Alberto';
```

13

Condiciones de comparación

- Los operadores de comparación se emplean en la cláusula WHERE para comparar una expresión con otra.
- El resultado de la comparación puede ser
 - Verdadero (T)
 - Falso (F)
 - Desconocido (U)
- No siempre se conoce el valor a buscar o se desea consultar por valores que son iguales o distintos a nulos.
 - LIKE
 - IS [NOT] NULL

14

Operadores LIKE, IS [NOT] NULL

- No siempre se conoce el valor exacto a buscar.
- % : Representa cualquier secuencia de cero o más caracteres
- _ : Denota un solo caracter
- Cuando se necesita una coincidencia exacta para los comodines "%" y "_", se usa la opción ESCAPE. Dicha opción especifica cuál es el caracter ESCAPE.
- Seleccionar los voluntarios con nombres que empiecen con "A" y terminen con "r".

```
SELECT apellido || ', ' || nombre  
FROM voluntario  
WHERE nombre LIKE 'A%r';
```

15

Operadores LIKE, IS [NOT] NULL

- Los operadores IS NULL e IS NOT NULL testean valores que son nulos.
- Si se comparan valores nulos usando los otros operadores (=, >, etc.) el resultado será siempre FALSO porque un valor nulo no puede ser igual, mayor, distinto, etc. a otro valor.
- Seleccionar los voluntarios sin institución asociada.

```
SELECT apellido || ', ' || nombre
FROM voluntario
WHERE id_institucion is NULL;
```

16

Operadores Lógicos

- Un operador lógico combina los resultados de dos condiciones para producir un único resultado basado en ellos, o invertir el resultado de una condición.
- Los operadores AND y OR se pueden usar para componer expresiones lógicas.
- El operador AND retorna VERDADERO si ambas condiciones evaluadas son VERDADERAS, mientras que el operador OR retorna VERDADERO si alguna de las condiciones es VERDADERA.
- Listar los voluntarios con más de 5000 horas disponibles, pero que aportó menos del 50%.

```
SELECT apellido || ', ' || nombre
FROM voluntario
WHERE horas_aportadas > 5000 AND porcentaje < 50;
```

17

Valor Nulo

- Si una fila carece de un valor para una columna en particular, se dice que contiene un "null".
- NULL es un valor inaccesible, sin valor, desconocido o inaplicable. No representa ni un cero ni un espacio en blanco. El cero es un número y el espacio en blanco es un carácter.
- *Listar los voluntarios con más de 5000 horas disponibles, pero que aportó menos del 50%.*
- *!Los porcentajes que son NULL devuelven U y con el AND según la tabla es NULL, por lo que no lo selecciona!'*
- *Listar los voluntarios con menos de 5000 horas disponibles o que aportaron menos del 30%.*
- *!Los porcentajes que son NULL devuelven U y con el OR, en este caso según la tabla, recupera el registro si se cumple la primera condición!'*

```
SELECT apellido || ', ' || nombre
FROM voluntario
WHERE horas_aportadas > 5000 AND porcentaje < 50;
SELECT apellido || ', ' || nombre
FROM voluntario
WHERE horas_aportadas < 5000 OR porcentaje < 30;
```

18

Funciones

- De una sola fila
 - Funciones de caracteres
 - Funciones numéricas
 - Funciones para trabajar con nulos
 - Funciones de fechas
 - Función CASE
 - Función DECODE

19

¿Qué son Funciones de Grupo?

Las funciones de grupo operan sobre juegos de filas para proporcionar un resultado por grupo.

Voluntario

nombre	apellido	horas aportadas
Steven	King	24000
Neena	Kochhar	17000
Lex	De Haan	17000
John	Russell	14000
Karen	Partners	13500
Michael	Hartstein	13000
Nancy	Greenberg	12000
Alberto	Errazuriz	12000
Shelley	Higgins	12000

...

El voluntario que puede aportar mayor cantidad de horas

24000

```
SELECT MAX(horas aportadas)
FROM voluntario;
```

20

Sintaxis de las Funciones de Grupo

```
SELECT    [columna,] funcion de grupo(columna),
          ...
FROM      tabla
[WHERE    condicion]
```

Tipos de Funciones de Grupo

- AVG
- COUNT
- MAX
- MIN
- SUM

21

Uso de las Funciones AVG y SUM

Puede utilizar AVG y SUM para datos numéricos.

```
SELECT AVG(horas aportadas),  
       MAX(horas aportadas),  
       MIN(horas aportadas),  
       SUM(horas aportadas)  
FROM   voluntario  
WHERE  nro voluntario > 500;
```

Puede utilizar MIN y MAX para cualquier tipo de dato.

```
SELECT MIN(fecha nacimiento)  
FROM   voluntario;
```

22

Uso de la Función COUNT

COUNT (*) devuelve el número de filas de una tabla.

```
SELECT COUNT(*)  
FROM   voluntario  
WHERE  id_tarea = SH CLERK;
```

- COUNT(*expr*) devuelve el número de filas con valores no nulos para *expr*.
- Visualice el número de valores de departamento de la tabla EMPLOYEES, excluyendo los valores nulos.

```
SELECT COUNT(porcentaje)  
FROM   voluntario  
WHERE  id_tarea = SH CLERK;
```

23

Funciones de Grupo y Valores Nulos

Las funciones de grupo ignoran los valores nulos de la columna.

```
SELECT AVG(porcentaje)  
FROM   voluntario;
```

Uso de la Función NVL con Funciones de Grupo

La función NVL fuerza a las funciones de grupo a que incluyan valores nulos.

```
SELECT AVG(NVL(porcentaje, 0))  
FROM   voluntario;
```

24

Creación de Grupos de Datos

VOLUNTARIO			
APELLIDO	ID_TAREA	HORAS_APORTADAS	
Gietz	AC_ACCOUNT 8300	8300	La máxima cantidad de horas aportadas por voluntarios que realizan una misma tarea
Higgins	AC_MGR 12000	12000	
De Haan	AD_VP 17000	17000	
Kochhar	AD_VF 17000	17000	
Popp	FI_ACCOUNT 6900	6900	9000
Sciarra	FI_ACCOUNT 7700	7700	
Uman	FI_ACCOUNT 7800	7800	
Chen	FI_ACCOUNT 8200	8200	
Faviet	FI_ACCOUNT 9000	9000	

ID_TAREA	MAX(HORAS_APORTADAS)
AC_ACCOUNT 8300	
AC_MGR 12000	
AD_VP 17000	
FI_ACCOUNT 9000	

25

Creación de Grupos de Datos: Sintaxis de la Cláusula GROUP BY

```
SELECT      columna, funcion de grupo(columna)
FROM        tabla
[WHERE      condicion]
[GROUP BY  expresion de grupo]
[ORDER BY  columna];
```

Divida las filas de una tabla en grupos más pequeños utilizando la cláusula GROUP BY.

26

Uso de la Cláusula GROUP BY

Todas las columnas de la lista SELECT que no estén en las funciones de grupo deben estar en la cláusula GROUP BY.

```
SELECT id_tarea, MAX(horas_aportadas)
FROM voluntario
GROUP BY id_tarea;
```

La columna en la cláusula GROUP BY pueden no estar en la lista SELECT.

```
SELECT AVG(horas_aportadas)
FROM voluntario
GROUP BY id_tarea ;
```

27

Exclusión de Resultados de Grupo: La Cláusula **HAVING**

Utilice la cláusula **HAVING** para restringir grupos:

1. Las filas se agrupan.
2. Se aplica la función de grupo.
3. Se muestran los grupos que coinciden con la cláusula **HAVING**.

```
SELECT      column, group_function
FROM        table
[WHERE      condition]
[GROUP BY   group_by_expression]
[HAVING     group_condition]
[ORDER BY   column];
```

28
