

Tema 5:





Bases

De

Datos

Índice



- 1. Sentencia SELECT
- 2. Consultas multitabla JOIN
- 3. Consultas Resumen





Lenguaje de consulta de datos SQL

```
SELECT [DISTINCT | ALL | otras opciones] lista_selección
[INTO OUTFILE 'nom_fichero' opciones_exportación]
[FROM tabla_origen [,tabla_origen]...]
[WHERE condición_de_búsqueda]
[LIMIT m,n]
[GROUP BY expresión_de_agrupamiento]
[HAVING condición_de_búsqueda]
[ORDER BY expresión_order_by [ASC | DESC] ]
```

Florida

1.- Sentencia SELECT

- •Cláusula SELECT. Determina las columnas que se mostrarán.
- •Cláusula INTO. Determina si el resultado de la consulta se escribe en un fichero.
- •Cláusula FROM. Determina las tablas origen para la consulta.
- •Cláusula WHERE. Determina qué registros se seleccionan.
- •Cláusula GROUP BY. Determina cómo se agrupan los registros.
- •Cláusula HAVING. Determina que registros se van a mostrar.
- •Cláusula ORDER BY. Determina el orden en el que se muestran los registros resultantes.
- •Cláusula LIMIT. Permite seleccionar una sección de filas del resultado



EJEMPLOS DE USO

- 1. Uso de alias para columnas SELECT Nombre as NOMBRE_EMPLEADO FROM Empleado
- 2. Uso de alias obligatorio para funciones de agre gado o nombres ambiguos/duplicados

SELECT AVG(Edad) AS EDAD_MEDIA FROM Empleado

- 3. Obtener todos los campos **SELECT * FROM Empleado**
- 4. Obtener algunos campos **SELECT Nombre, Edad FROM Empleado**



- 5. Obtener los datos ordenados
- SELECT Nombre, Edad, Alojamiento FROM Empleado ORDER BY Edad
- SELECT Nombre, Edad , Alojamiento FROM Empleado ORDER BY Edad ASC
- SELECT Nombre, Edad, Alojamiento FROM Empleado ORDER BY Edad DESC
- 6. Obtener los datos agrupados
- SELECT Nombre, Edad, Alojamiento FROM Empleado GROUP BY Edad, Alojamiento
- 7. Eliminar filas duplicadas
- SELECT DISTINCT Alojamiento FROM Empleado



Orden de procesamiento con WHERE, GROUP BY y HAVING

- 1. WHERE excluye las filas que no cumplen la condición de búsqueda
- 2. GROUP BY recopila las filas seleccionadas en un grupo para cada valor exclusivo de la cláusula GROUP BY
- 3. Las funciones de agregado especificadas en la lista de seLección calculan los valores de resumen de cada grupo.
- 4. HAVING excluye las filas que no cumplen su condición de búsqueda.

8. Condición de búsqueda

SELECT * FROM Curso WHERE ctarifa<150

SELECT * FROM Curso WHERE combre >="RACIONALISMO"

SELECT ciudad, ventas, objetivo FROM Oficinas WHERE ventas>objetivo

SELECT nombre, ventas, cuota FROM repventas WHERE numempl= 105

9. Eliminar filas duplicadas

SELECT dir FROM oficinas

SELECT DISTINCT dir FROM oficinas

SELECT DISTINCT ctarifa FROM curso

10. Ordenación de resultados

SELECT * FROM personal ORDER BY enombre

SELECT dept, cargo, esueldo FROM personal ORDER BY dept, esueldo **DESC**

SELECT ciudad, region, ventas FROM oficinas ORDER BY region, ciudad



SELECT ciudad, region, ventas-objetivo AS RENDIMIENTO FROM oficinas ORDER BY 3 DESC

SELECT ciudad, region, ventas-objetivo AS RENDIMIENTO FROM oficinas ORDER BY region ASC, 3 DESC

11. Columnas calculadas

SELECT enombre, esueldo*100 AS NUEVO_SUELDO FROM personal

SELECT enombre, esueldo*100 AS NUEVO_SUELDO FROM personal WHERE esueldo-100 <25000

SELECT idfab, idproducto, descripcion, existencias*precio AS INVENTARIO FROM productos

SELECT nombre, cuota, cuota+ 0.003*ventas AS CUOTA_ELEVADA FROM repventas

000000000000000000



12. Condiciones de búsqueda.

COMPARACIÓN

SELECT nombre FROM repventas WHERE contrato < '1988-01-01'

SELECT * FROM oficinas WHERE ventas<0.8*objetivo

CONTRASTE DE RANGO

SELECT cnombre, cno, cdescp FROM curso WHERE cnombre BETWEEN 'ES' AND 'ESZ'

SELECT * FROM pedidos WHERE fechapedido BETWEEN '1989-10-01' AND '1989-12-31'

CONTRASTE DE PERTENENCIA. Claúsula IN

SELECT * FROM curso WHERE ctarifa IN (12, 50, 75, 90, 100, 500)

CONTRASTE DE PATRÓN. Claúsula LIKE

SELECT * FROM curso WHERE cdescp LIKE 'PARA %'

SELECT * FROM clientes WHERE limitecredito NOT LIKE 20000



SELECT * FROM clientes WHERE empresa LIKE '_e%'

CONTRASTE DE VALOR NULO. IS NULL

SELECT FNOMBRE, FNUMDEP, FDEPT FROM CLAUSTRO WHERE fnumdep IS NOT NULL

CONTRASTES COMPUESTOS: AND, OR, NOT

SELECT * FROM curso WHERE (CTARIFA >=100) AND (ctarifa <=500)

13. COMBINACIÓN DE CONSULTAS: UNION

Combina los resultados de 2 o más consultas en una única tabla

SELECT esueldo FROM personal UNION SELECT fsueldo FROM claustro SELECT idfab, idproducto FROM productos WHERE precio >2000 UNION select DISTINCT fab, producto FROM PEDIDOS where IMPORTE >3000

Florida

1.- Sentencia SELECT

SELECT idfab, idproducto FROM productos WHERE precio >2000 UNION ALL SELECT fab, producto FROM PEDIDOS where IMPORTE >3000

SELECT idfab, idproducto FROM productos WHERE precio >2000 UNION ALL SELECT DISTINCT fab, producto FROM PEDIDOS where IMPORTE >3000 ORDER BY 1,2



Consultas multitabla (JOIN)

Las consultas multitabla o JOINS permiten recuperar datos de 2 tablas o más según las relaciones lógicas entre ellas.

Una condición de consulta multitabla (combinación o composición) define la forma de relacionarse las tablas en la consulta:

- Especificar que columna en cada tabla debe usarse para la combinación. P.ej. La clave externa y su clave asociada en la otra tabla.
- Especificar un operador lógico para usarlo en los valores de comparación de las columnas.



EJEMPLOS DE USO

SELECT p.numpedido, p.importe, c.empresa, c.limitecredito FROM pedidos p INNER JOIN clientes c ON p.clie=c.numclie (sintáxis ANSI)

Para evitar ambigüedades se puede calificar las columnas con los nombres de las tablas.

SELECT pedidos.numpedido, pedidos.importe, clientes.empresa, clientes.limitecredito FROM pedidos, clientes WHERE clie=numclie (sintáxis SQL92)

Mejor sintáxis ANSI. Más compatibilidad y separación entre cláusulas ON y WHERE



JOIN DE 2 TABLAS

SELECT * FROM curso INNER JOIN departamento ON curso.cdept=departamento.dept

JOIN DE 2 TABLAS CON CRITERIOS DE SELECCIÓN

SELECT cnombre, ctarifa, dchfno FROM curso c INNER JOIN departamento d ON c.cdept=d.dept WHERE ctarifa>100

SELECT o.ciudad, r.nombre, r.titulo FROM repventas r INNER JOIN oficinas o ON o.dir=r.numempl WHERE o.objetivo<600000

JOIN DE 2 TABLAS CON MÚLTIPLES COLUMNAS DE EMPAREJAMIENTO

SELECT numpedido, importe, descripcion FROM pedidos INNER JOIN productos ON fab=idfab AND producto=idproducto;



JOIN DE 3 o MÁS TABLAS

SELECT numpedido, importe, empresa, nombre FROM pedidos INNER JOIN clientes INNER JOIN repventas ON clie=numclie AND rep=numempl WHERE importe >25000

Para realizar una JOIN no es necesario que las columnas de emparejamiento tengan relación clave primaria/clave foránea

SELECT numpedido, importe, fechapedido, nombre FROM pedidos INNER JOIN repventas ON fechapedido=contrato

Tampoco es necesario que la relación entre las columnas sea la de igualdad SELECT nombre, cuota, ciudad, objetivo FROM repventas INNER JOIN oficinas ON cuota > objetivo



JOIN INTERNO

SELECT campos FROM tabla1 INNER JOIN tabla2 ON tabla1.campo1 operadorComparación tabla2.campo2

SELECT nombre, titulo, objetivo, oficina, ciudad FROM repventas INNER JOIN oficinas ON oficinarep=oficina

Se pueden anidar instrucciones JOIN:

SELECT campos FROM tabla1 INNER JOIN (tabla2 INNER JOIN [tabla3 [INNER JOIN [(] tablax [INNER JOIN....)]

ON tabla3.campo3 *operadorComparación* tablaX.campoX)]

ON tabla2.campo2 operadorComparación tabla3.campo3)

ON tabla1.campo1 operadorComparación tabla2.campo2)



JOIN EXTERNO

SELECT campos FROM tabla1 OUTER JOIN tabla2 ON tabla1.campo1 operadorComparación tabla2.campo2

SELECT nombre, titulo, objetivo, oficina, ciudad FROM repventas OUTER JOIN oficinas ON oficinarep=oficina



JOIN IZQUIERDO Y DERECHO

El LEFT JOIN se utiliza para crear una combinación (JOIN) externa izquierda. Incluirá todos los registros de la primera (parte izquierda) de dos tablas, aunque no haya valores coincidentes para los registros de la segunda tabla(parte derecha)

El RIGHT JOIN se utiliza para crear una combinación (JOIN) externa derecha. Incluirá todos los registros de la segunda (parte derecha) de dos tablas, aunque no haya valores coincidentes para los registros de la primera tabla(parte izquierda)

SELECT campos FROM tabla1 [LEFT | RIGHT] JOIN tabla2 ON tabla1.campo1 operadorComparación tabla2.campo2

SELECT nombre, titulo, objetivo, oficina, ciudad FROM repventas LEFT JOIN oficinas ON numempl=dir



Una operación LEFT JOIN o RIGHT JOIN se puede anidar en una operación INNER JOIN, pero INNER JOIN no se puede anidar en otra operación LEFT JOINN o RIGHT JOIN.

Puede vincularse múltiples claúsulas ON.

SELECT empresa, numpedido, importe FROM clientes INNER JOIN pedidos ON numclie=clie AND numclie='2103'

SELECT empresa, numpedido, importe FROM clientes LEFT JOIN pedidos ON numclie=clie AND numclie='2103'

SELECT empresa, numpedido, importe FROM clientes RIGHT JOIN pedidos ON numclie=clie AND numclie='2103'



Funciones de agregado en la lista de selección

Las funciones de agregado (SUM, AVG, STD, COUNT, COUNT(*), MAX, MIN, etc..) generan valores de resumen en los conjuntos de resultados de las consultas.

Pueden aplicarse a todas las filas de la tabla, a un subconjunto especificado en WHERE o a uno o varios grupos de filas con GROUP BY.

DISTINCT puede usarse para eliminar duplicados antes de realizar la operación de agregado.

EJEMPLOS:

SELECT AVG(ctarifa) AS MEDIA, MAX(ctarifa) AS MAXIMO, MIN(ctarifa) AS MINIMO FROM curso WHERE cdept="CIS" AND ctarifa>0

SELECT AVG(cuota) AS CUOTA_PROMEDIO, AVG(ventas) AS VENTAS_PROMEDIO, AVG(100*(ventas/cuota)) AS RENDIMIENTO_CUOTA_PROMEDIO FROM repventas



SELECT AVG(ctarifa) AS MEDIA, MAX(ctarifa) AS MAXIMO, MIN(ctarifa) AS MINIMO FROM curso WHERE cdept="CIS" AND ctarifa>0

SELECT AVG(cuota) AS CUOTA_PROMEDIO, AVG(ventas) AS VENTAS_PROMEDIO, AVG(100*(ventas/cuota)) AS RENDIMIENTO_CUOTA_PROMEDIO FROM repventas

SELECT SUM(cuota) "TOTAL", SUM(ventas) "VENTAS TOTALES", MIN(cuota) "CUOTA MINIMA", MAX(cuota) "CUOTA MAXIMA" FROM repventas



Función COUNT(*)

La función COUNT() devuelve el número de elementos de un grupo.

COUNT ({ [ALL | DISTINCT] expression] | * })

Con DISTINCT obligamos a valores únicos no NULL

* Especifica que tienen que contar todas las filas

SELECT COUNT(*) AS VENDEDORES FROM repventas WHERE ventas > cuota

SELECT COUNT(*) AS PEDIDOS FROM pedidos WHERE importe > 25000



Valores NULL en funciones de agregado

Tratamiento de filas duplicadas con DISTINCT

Agrupar filas con GROUP BY

La cláusula GROUP BY permite producir valores de agregado para cada conjunto de filas de resultados

SELECT cdept AS DEPARTAMENTO, SUM(cred) AS CREDITOS FROM curso GROUP BY cdept

SELECT rep AS VENDEDOR, clie AS CLIENTE, SUM(importe) AS PEDIDO_TOTAL FROM pedidos GROUP BY rep, clie



La cláusula GROUP BY contiene:

- Una o más expresiones sin agregados (referencias a columnas)
- Opcionalmente, la palabra clave ALL que especifica que se devuelvan todos los grupos generados incluso si no cumplen las condiciones de búsqueda.
- La cláusula HAVING se usa junto a GROUP BY aunque también puede se puede especificar si ella.

Si se usa la cláusula WHERE en una consulta con GROUP BY las filas que no cumplen las condiciones WHERE se eliminan antes de l agrupamiento.

SELECT numempl AS EMPLEADO, nombre, SUM(importe) AS PEDIDO_TOTAL FROM pedidos, repventas WHERE rep=numempl GROUP BY numempl, nombre

La cláusula HAVING permite establecer condiciones de búsqueda igual que WHERE pero éstas se aplican después de producirse el agrupamiento. HAVING tiene una sintaxis similar a WHERE pero permite funciones de agregado y debe hacer referencia sólo a elementos que aparecen en la lista de selección.

SELECT rep AS EMPLEADO, AVG(importe) AS PEDIDO_MEDIO FROM pedidos GROUP BY rep HAVING SUM(importe)>30000

SELECT ciudad, SUM(cuota) AS CUOTA, SUM(repventas.ventas) AS VENTAS FROM oficinas, repventas WHERE oficina=oficinarep GROUP BY ciudad HAVING COUNT(*)>2