

SENTENCIAS AVANZADAS

BASES DE DATOS I


Giuliano Crenna

ORDER BY



El comando **ORDER BY** se utiliza para ordenar el conjunto de resultados de una consulta según una o más columnas. Puedes especificar el orden como ascendente (**ASC**, que es el valor predeterminado) o descendente (**DESC**). Este comando es fundamental cuando necesitas presentar datos de forma estructurada y comprensible.

Mostrar los libros ordenados alfabéticamente por título.



```
SELECT titulo, fecha_publicacion  
FROM libros  
ORDER BY titulo ASC;
```

ORDER BY



Mostrar los libros ordenados por año de publicación en orden descendente:



```
SELECT titulo, fecha_publicacion  
FROM libros  
ORDER BY fecha_publicacion DESC;
```

FUNCIONES DE GRUPO

Las **funciones de grupo** realizan cálculos en un conjunto de filas y devuelven un único resultado por grupo. Estas funciones son esenciales para realizar análisis de datos agregados. Las más comunes son:

- **COUNT()**: cuenta el número de filas.
- **SUM()**: suma los valores de una columna.
- **AVG()**: calcula el promedio de los valores.
- **MAX()**: obtiene el valor máximo.
- **MIN()**: obtiene el valor mínimo.

Contar cuántos préstamos ha realizado cada usuario:

```
SELECT usuario_id, COUNT(*) AS total_prestamos
FROM prestamos
GROUP BY usuario_id;
```

FUNCIONES DE GRUPO

Obtener el promedio de libros prestados por autor:



```
SELECT autor_id, AVG(cantidad_prestamos)
FROM libros
GROUP BY autor_id;
```

HAVING

La cláusula **HAVING** se utiliza para filtrar los resultados de una consulta después de haber realizado una agregación con funciones de grupo. A diferencia de la cláusula **WHERE**, que filtra filas individuales antes de la agregación, **HAVING** actúa sobre los resultados agrupados.

Mostrar autores que tienen más de 5 libros en la biblioteca:

```
SELECT autor_id, COUNT(*) AS total_libros
FROM libros
GROUP BY autor_id
HAVING COUNT(*) > 5;
```

Obtener usuarios que han realizado más de 3 préstamos:

```
SELECT usuario_id, COUNT(*) AS total_prestamos
FROM prestamos
GROUP BY usuario_id
HAVING COUNT(*) > 3;
```

SUBCONSULTAS

Las subconsultas son consultas dentro de otras consultas, permitiendo el uso del resultado de una consulta para completar otra. Son útiles para filtrar datos o realizar cálculos complejos.

Libros que han sido prestados más de 10 veces.

```
SELECT titulo
FROM libros
WHERE id IN (
    SELECT libro_id
    FROM prestamos
    GROUP BY libro_id
    HAVING COUNT(*) > 10
);
```

SUBCONSULTAS

Usuarios que han tomado libros prestados del autor "César Aira".

```
SELECT nombre, apellido
FROM usuario
WHERE id IN (
    SELECT usuario_id
    FROM prestamos
    WHERE libro_id IN (
        SELECT id
        FROM libros
        WHERE autor_id = (
            SELECT id
            FROM autores
            WHERE nombre = 'César' AND apellido = 'Aira'
        )
    )
);
```


INDEXACIÓN Y OPTIMIZACIÓN

La **indexación** mejora el rendimiento de las consultas al crear índices sobre columnas frecuentemente consultadas, como claves primarias o columnas en condiciones **WHERE**. Esto permite que las búsquedas y las operaciones sean más rápidas.

Ejemplo de creación de un índice:

```
CREATE INDEX idx_usuario_id ON prestamos(usuario_id);
```



UTN - FRRO



GIULIANO CRENNNA



giulicrenna@gmail.com

¡MUCHAS GRACIAS!