BASES DE DATOS 2
clase

Sub-consultas
Agrupamiento

Universidad ORT - Uruguay

Agenda

- Continuación subconsultas
- Agrupamiento
- Having

Subconsultas

Una subconsulta es una sentencia **SELECT** que aparece dentro de otra sentencia **SELECT**.

Normalmente se utilizan para filtrar una clausula **WHERE** o **HAVING** con el conjunto de resultados de la subconsulta, aunque también pueden utilizarse en la lista de selección.

Subconsultas

SELECT A1,...,An
FROM R1,...,Rm
WHERE (Aj,...,Ak) <op-comp>
(SELECT Bj,..., Bk
FROM S1,..., Sn
WHERE C)
op-comp puede ser:

- **=**
- **IN**
- **NOT IN**
- **= ANY**
- **> ANY**
- **= ALL**
- **> ALL**

Subconsultas - Ejemplo

Dar el código y nombre de los estudiantes que estén inscriptos a alguno de los cursos que esté inscripto el estudiante con código = 1.

```
SELECT idEstudiante,nombre
FROM ESTUDIANTES E, INSCRIPCIONES I
WHERE E.idEstudiante = I.idEstudiante
        AND E.idEstudiante <> 1 AND I.idCurso IN
        (SELECT idCurso FROM INSCRIPCIONES
                WHERE INSCRIPCIONES.idEstudiante = 1)
```

Subconsultas - Ejemplo

Mostrar los datos de las inscripciones que tengan costo de inscripción al curso 2 mayor a alguno de los costos de inscripción del curso 1.

```
SELECT *
FROM  INSCRIPCIONES I
WHERE I.idCurso = 2 AND I.costo > ANY
        (SELECT costo FROM INSCRIPCIONES I1
                WHERE I1.idCurso = 1)
```

Subconsultas - Ejemplo

Mostrar los datos de las inscripciones que tengan costo de inscripción al curso 2 mayor a todas las inscripciones del curso 1.

```
SELECT *
FROM  INSCRIPCIONES I
WHERE I.idCurso = 2 AND I.costo > ALL
        (SELECT costo FROM INSCRIPCIONES I1
                WHERE I1.idCurso = 1)
```

Subconsultas – Función exists

- Sirve para chequear si el resultado de una consulta anidada es vacío.
- Retorna verdadero cuando una subconsulta retorna al menos una fila.
- Se puede utilizar la negación del EXISTS: NOT EXISTS.
- La función EXISTS puede ser utilizada en:
  - SELECT
  - UPDATE
  - INSERT
  - DELETE

Subconsultas – Ejemplo EXISTS

Dar el código y nombre de los estudiantes que sólo estén inscriptos al curso con código = 1.

```
SELECT E1.idEstudiante,nombre
FROM ESTUDIANTES E1, INSCRIPCIONES I1
WHERE E1.idEstudiante = I1.idEstudiante
        AND I1.idCurso = 1 AND NOT EXISTS
        (SELECT * FROM INSCRIPCIONES I2
                WHERE I2.idEstudiante = E1.idEstudiante
                        AND I2.idCurso <> 1);
```

Agrupamiento de tuplas

- Dar para cada estudiante, la cantidad de cursos a los que está inscripto.
  - Esto implica:
    - Tomar cada grupo de tuplas en INSCRIPCIONES correspondiente a un estudiante.
    - Contar la cantidad de tuplas en el grupo.
- Para realizar esta operación es necesaria la cláusula **GROUP BY**.

Group By

- Es una cláusula más que se agrega al SELECT, FROM, WHERE.
- Ejemplo:

```
SELECT estudianteCod, count(*)
FROM INSCRIPCIONES
GROUP BY estudianteCod
```

- ¿Cómo funciona?
  - Genera un grupo por cada estudianteCod distinto.
  - De cada grupo devuelve el estudianteCod y la cantidad de tuplas **de dicho grupo.**

Ejemplos GROUP BY (1)

Dar el número de estudiante y los promedios de costos de sus inscripciones.

```
SELECT estudianteCod, avg(costo)
FROM INSCRIPCIONES
GROUP BY estudianteCod
```

Ejemplos GROUP BY (2)

Dar el número de estudiante, el nombre y sus totales de costo de inscripciones.

```
SELECT E.estudianteCod, E.nombre, sum(costo)
FROM INSCRIPCIONES I, ESTUDIANTES E
WHERE I.estudianteCod = E.estudianteCod
GROUP BY E.estudianteCod, E.nombre
```

GROUP BY - Sintaxis

```
SELECT columna1, columna2,... columna_n,
    funcion_agregacion(expresion)
FROM <tablas>
WHERE <condiciones>
GROUP BY columna1, columna2,…columna_n;
```

GROUP BY – Regla de sintaxis

- En una sentencia SQL que tiene cláusula GROUP BY, las expresiones en el SELECT pueden ser sólo:
  - Atributos presentes en la cláusula GROUP BY.
  - Funciones de agregación sobre atributos.
  - Expresiones aritméticas que utilicen los anteriores.

- El agrupamiento se realiza **después de** aplicar el WHERE. O sea, sobre las tuplas que cumplen la condición.

HAVING – Condiciones sobre grupos

- Con la cláusula HAVING se pueden especificar condiciones sobre los grupos.
- Por ejemplo:
  - Dar el código de estudiante y sus promedios de costo de inscripciones, **pero para los estudiantes inscriptos a más de un curso**.

```
SELECT estudianteCod, avg(costo)
FROM INSCRIPCIONES
GROUP BY estudianteCod
HAVING count(*) > 1;
```

Ejemplos HAVING

Dar el número de estudiante y su nombre para los inscriptos a más de un curso, sobre cursos con id mayor a 1. Mostrar el costo total de dichas inscripciones.

```
SELECT E.estudianteCod, E.nombre, sum(costo)
FROM INSCRIPCIONES I, ESTUDIANTES E
WHERE I.estudianteCod = E.estudianteCod AND I.cursoCod > 1
GROUP BY E.estudianteCod, E.nombre
HAVING count(*) > 1;
```