BASES DE DATOS

Clase 16 - Lenguaje de consulta formal
Algebra Relacional

Profesora: Erika Gutiérrez Beltrán

Tema 2: Bases de datos relacionales

Objetivo

Transformar los modelos conceptuales a lógicos. Construir bases de datos para datos estructurados

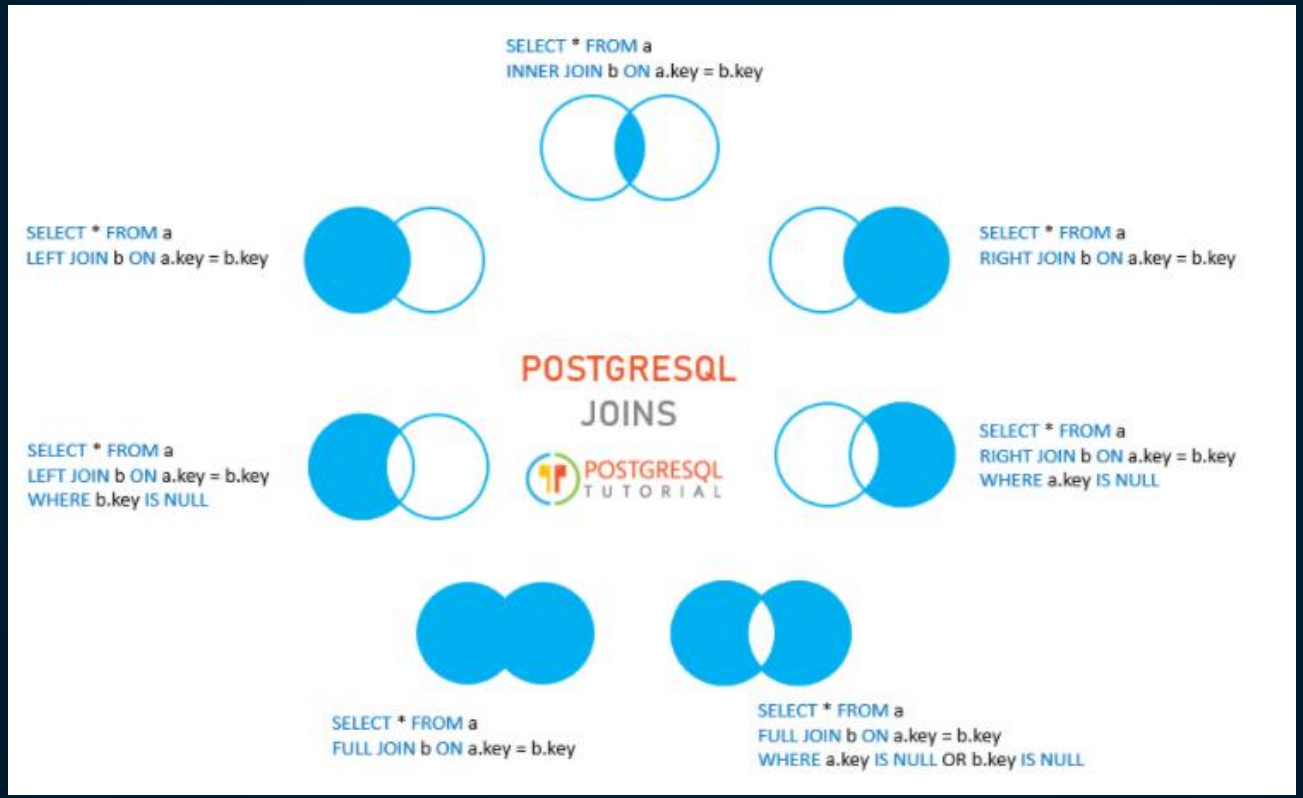
Temario

- Utilización de los modelos en el mundo real ✅
- Transformación del modelado a la base de datos ✅
- Clases de entidades ✅
- Modelo relacional, restricciones de integridad y normalización ✅
- Lenguaje de consulta formal
- Lenguaje de consulta SQL ✅
- Arquitectura de los sistemas gestores de Bases de datos, OLTP, OLAP

Logros

Capturar datos desde una pantalla de usuario real y almacenarlos en una base de datos. Manipulación de estos datos.

JOIN aprendidos previamente

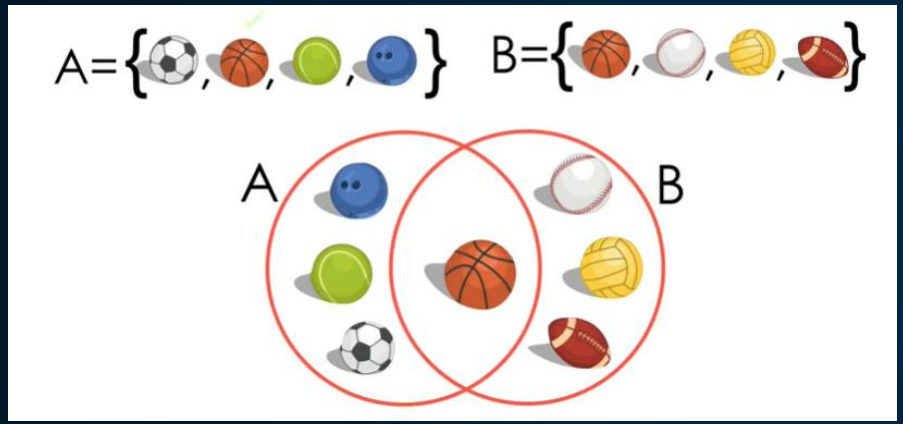


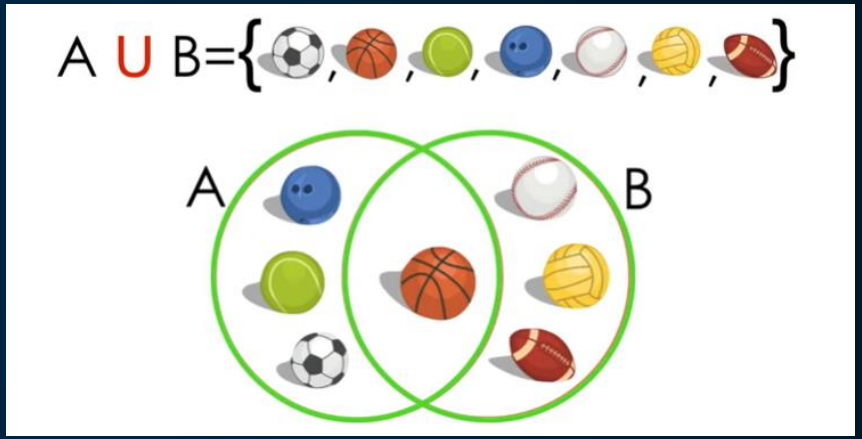
Algebra relacional

Teniendo en cuenta la lectura: Extending the database relational model to capture more meaning, nos enfocaremos en los operadores de conjuntos como UNION, INTERSECCIÓN Y DIFERENCIA. Estas operaciones como el caso de la unión deben realizarse sobre entidades (conjuntos) que estén bajo el mismo dominio. La unión a nivel de sentencia SQL combina el resultado de dos o más instrucciones para entregar como resultado una entidad.

Vamos a revisar el siguiente ejemplo para entender como funciona la UNION en la teoría de conjuntos.

Algebra relacional - UNION





<\>

Algebra relacional - UNION

Para lograr la unión con sentencias SQL es necesario que los tipos de datos sean similares, las entidades tengan la misma cantidad de campos, en la selección deben estar en el mismo orden.

```
select fullname, address from "EjerciciosClase".students
    union
select fullname, address from "EjerciciosClase".professors;
```

La sentencia **unión all** nos trae todos los datos de ambas tablas sin importar que estén repetidos.

<\/>

Algebra relacional - UNION

```sql
-- Obtener todos los datos de las tablas aunque esten repetidos
select fullname, address from "EjerciciosClase".students
    union all
select fullname, address from "EjerciciosClase".professors;
```

```sql
-- Identificar de que tabla es cada registro
select fullname, address, 'student' as conditiontable from "EjerciciosClase".students
    union
select fullname, address, 'professor' from "EjerciciosClase".professors
order by conditiontable;
```

Algebra relacional – PRODUCTO CARTESIANO

En la teoría de conjuntos el producto cartesiano da como resultado un nuevo conjunto que ha sido combinado de manera que todos los datos de un conjunto A se relacionan con cada uno de los datos de la tabla B.

El producto cartesiano en SQL es posible aplicarlo, aunque no se tengan restricciones (constraints), ya que sugiere crear una nueva entidad a partir de la combinación de dos o más entidades, aunque no estén asociadas, pero SÍ pertenecen al mismo dominio.

Algebra relacional – PRODUCTO CARTESIANO

En el juego de vestir la muñeca, podemos tomar como conjuntos cada sección del juego como “Accessories”, “Tops”, “Bottoms”, “shoes”. Donde por cada uno de los elementos del conjunto es posible armar un look creando combinaciones con cada sección

Algebra relacional – PRODUCTO CARTESIANO

Ejemplo de producto cartesiano:

$(AXB) = \{(x,y) \mid x \in A \text{ y } y \in B\}$

A = {4,7}
B = {2,5,9}

AXB = {(4,2), (4,5), (4,9), (7,2), (7,5), (7,9)}

Algebra relacional – PRODUCTO CARTESIANO

```
select *
  from "Grupo 5"."pelicula" , "Grupo 5"."categoria";
```

Algebra relacional – PRODUCTO CARTESIANO

Ejercicios para poner en practica lo aprendido:

- Crea las tablas
MOVIES (id_movie, name_movie, synopsis, premiere)
SERIES (id_serie, name_serie, name_movie, synopsis, premiere, provider)
SERIES (id_serie, name_serie, name_movie, synopsis, premiere, provider)
GENRE (id_genre, name_genre)
GENRE_FILMS (id_movie, id_serie, id_genre)

- Agrega los siguientes registro para cada table, 3 para MOVIES, 2 para SERIES, 5 para GENRE, en la tabla GENRE_FILMS agrega 2 películas y 2 series (recuerda que esta es la relacional)

Algebra relacional – PRODUCTO CARTESIANO

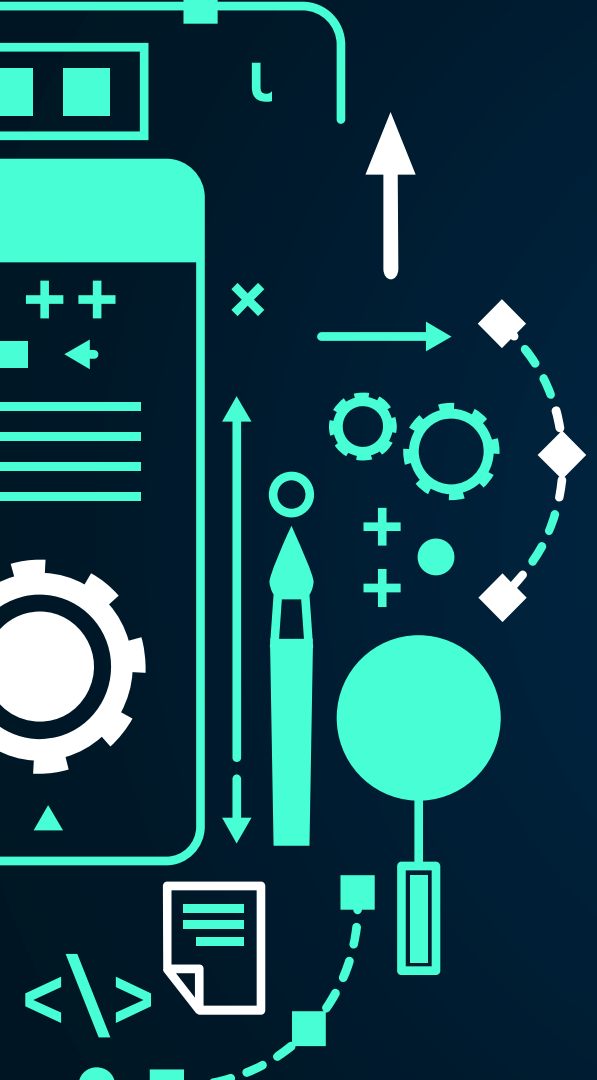
Realiza las consultas JOIN que te permitan:

- Conocer el nombre de una serie, proveedor y género al que pertenece, indicando el nombre del género.
- Mostrar todos los géneros que no han sido asignados a una serie o película.
- Mostrar todas las películas sin género asignado.
- Mostrar todas las series sin un género asignado
- Mostrar todas las series y géneros que no han sido asignados
- Mostrar todas las coincidencias entre dos tablas (series y genre films)
- Muestra todas las películas y series disponibles en la base de datos.

- Ajustar tabla de genre films, para eliminar los valores nulos que quedan al momento de registrar el género de una película o una serie.

REFERENCIAS

- Libro: Fundamentos de base de datos 5 edición, Silberschatz
- Tutorial de PostgreSQL: https://www.postgresqltutorial.com/postgresql-joins/
- Extending the database relational model to capture more meaning, IBM Research Laboratory
- Ejemplo Union: https://www.tutorialesprogramacionya.com/sqlserverya/temarios/descripcion.php?cod=88&punto=82&inicio=



Gracias!