BASES DE DATOS

Clase 11 – Lenguaje de consulta SQL

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.

Clases de entidades

Como hemos visto previamente, recordemos que las entidades están identificadas por clases o tipos que facilitan el entendimiento de estar al momento de hacer diagramas.

Estas son débiles:

o entidades fuertes:

Clases de entidades

Cuando estamos construyendo nuestros modelos deben tener claro que entidades dependen de otras, cómo será al momento de realizar normalización, ¿se crearán nuevas entidades?, ¿desaparecerán algunas de las que ya están?.

Para recordar conceptos de la clase se recomienda realizar la siguiente lectura:

Gestión de bases de datos - introducción

Recondando la normalización de Bases de datos

1NF: Identificar grupos de datos repetidos sobre la misma entidad y registro
2NF: Debe contener 1NF y se debe identificar las dependencias transitivas y funcionales
3NF: Debe contener 2NF y se deben eliminar las dependencias transitivas y todo atributo no principal debe depender de la clave primaria
NFBC: Debe contener 3NF y se cumple si las dependencias funcionales dependen de una clave primaria
4NF: Debe contener NFBC y esta consiste en eliminar todas las dependencias multivaluadas
5NF: Debe contener 4NF y se utiliza para dividir tablas de gran tamaño y mejorar la lectura y manipulación de los datos (proyección – unión)

Recondando la normalización de Bases de datos

nombre_completo	direccion	Películas rentadas	Categoria	Edad
Ana Jones	Calle 31 # 50a-22	Piratas del caribe, Titanic, 30 going on 13, Encanto	Acción, drama, comedia, musical	25
Robert Gonzalez	Transversal 55 # 10a-22	Rapido y furioso, Shrek2, Split	Acción, animada, terror	27
Robert Gonzalez	Calle 30 # 40-20	Infinity war	Acción	27

Aplica a esta tabla la primera, la segunda y la tercera forma normal

Lenguaje de consulta SQL

El lenguaje SQL nos permite manipular datos de nuestra base de datos, creando, eliminando, editando y consultando información. También nos permite crear relaciones y asignar restricciones que harán la base de datos segura y consistente.

Lenguaje de consulta SQL

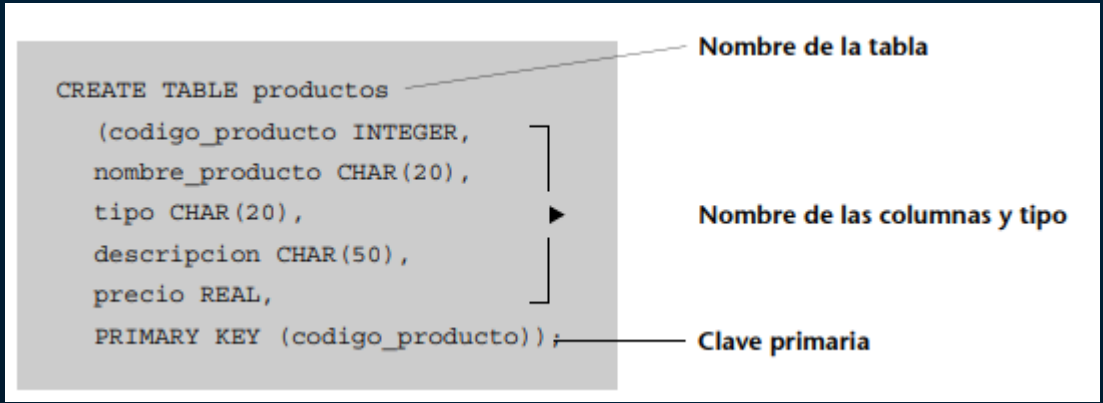
Para tener en cuenta:

- La creación de las tablas la realizaremos con la primera letra en mayúsculas y sin espacios ***UsuarioDireccion***
- Los atributos de cada tabla serán escritos en minúsculas sin caracteres especiales
- Las sentencias propias de SQL irán en mayúsculas

Lenguaje de consulta SQL

Vamos a crear en un script la estructura de creación de tablas basada en el resultado de normalización realizado previamente

Recordemos comandos de interacción con las bases de datos:



Lenguaje de consulta SQL

Estructura para PostgreSQL:

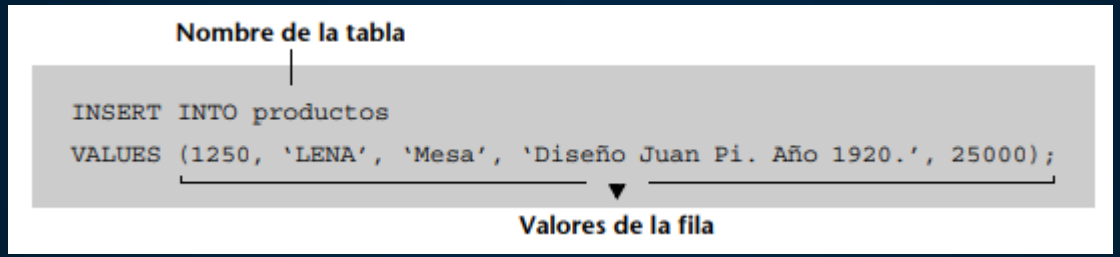
```
CREATE TABLE [IF NOT EXISTS] table_name (
   column1 datatype(length) column_contraint,
   column2 datatype(length) column_contraint,
   column3 datatype(length) column_contraint,
   table_constraints
);
```

```
CREATE TABLE accounts (
        user_id serial PRIMARY KEY,
        username VARCHAR ( 50 ) UNIQUE NOT NULL,
        password VARCHAR ( 50 ) NOT NULL,
        email VARCHAR ( 255 ) UNIQUE NOT NULL,
        created_on TIMESTAMP NOT NULL,
        last_login TIMESTAMP
);
```

El código que aquí se muestra es un ejemplo, reemplaza los valores por los que necesitas para crear la tabla con los valores personalizados
Para más detalle ver el siguiente enlace: https://www.postgresqltutorial.com/postgresql-create-table/

Lenguaje de consulta SQL

Estructura para la inserción de datos en una base de datos



<\>

Lenguaje de consulta SQL

Estructura para PostgreSQL

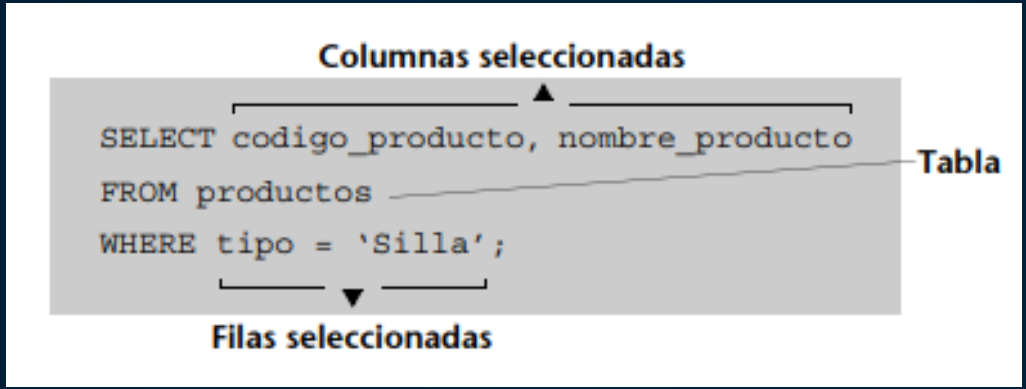
```
INSERT INTO table_name (column_list)
VALUES
    (value_list_1),
    (value_list_2),
    ...
    (value_list_n);
```

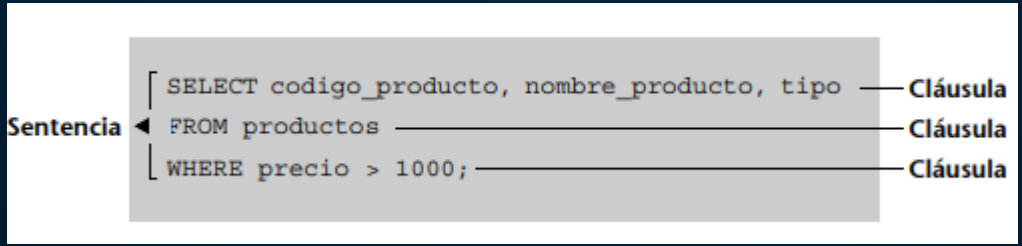
```
INSERT INTO
    links (url, name)
VALUES
    ('https://www.google.com','Google'),
    ('https://www.yahoo.com','Yahoo'),
    ('https://www.bing.com','Bing');
```

El código que aquí se muestra es un ejemplo, reemplaza los valores por los que necesitas insertar datos a la tabla
Para más detalle ver el siguiente enlace: https://www.postgresqltutorial.com/postgresql-insert-multiple-rows/

Lenguaje de consulta SQL

Estructura para la selección de datos de la base de datos (proyección)





Lenguaje de consulta SQL

Estructura para PostgreSQL

```
SELECT * FROM film_r;
```

```
SELECT
    select_list
INTO [ TEMPORARY | TEMP | UNLOGGED ] [ TABLE ] new_table_name
FROM
    table_name
WHERE
    search_condition;
```

El código que aquí se muestra es un ejemplo, reemplaza los valores por los que necesitas para consultar datos de la tabla
Para más detalle ver el siguiente enlace: https://www.postgresqltutorial.com/postgresql-select-into/

Lenguaje de consulta SQL

Estructura para PostgreSQL

```
SELECT * FROM film_r;
```

```
SELECT
    film_id,
    title,
    rental_rate
INTO TABLE film_r
FROM
    film
WHERE
    rating = 'R'
AND rental_duration = 5
ORDER BY
    title;
```

El código que aquí se muestra es un ejemplo, reemplaza los valores por los que necesitas para consultar datos de la tabla
Para más detalle ver el siguiente enlace: https://www.postgresqltutorial.com/postgresql-select-into/

Lenguaje de consulta SQL

Funciones de agregación, el uso de la función COUNT permite mostrar el número de registros que están siendo filtrados, ejemplo:

```
SELECT COUNT(*) AS numero_dep
FROM departamentos
WHERE ciudad_dep = 'Lleida';
```

numero_dep
1

Lenguaje de consulta SQL

Estructura para PostgreSQL

```
SELECT
   COUNT(*)
FROM
   table_name
WHERE
   condition;
```

El código que aquí se muestra es un ejemplo, reemplaza los valores por los que necesitas para consultar datos de la tabla y usar el COUNT
Para más detalle ver el siguiente enlace: https://www.postgresqltutorial.com/postgresql-count-function/

Lenguaje de consulta SQL

Estructura para PostgreSQL

```
SELECT
        COUNT (DISTINCT amount)
FROM
        payment;
```

```
SELECT
    COUNT(*)
FROM
    payment;
```

```
SELECT
        customer_id,
        COUNT (customer_id)
FROM
        payment
GROUP BY
        customer_id;
```

El código que aquí se muestra es un ejemplo, reemplaza los valores por los que necesitas para consultar datos de la tabla y usar el COUNT
Para más detalle ver el siguiente enlace: https://www.postgresqltutorial.com/postgresql-count-function/

Lenguaje de consulta SQL

El GROUP BY devuelve las filas de un SELECT agrupados. Se usan para funciones agregadas como el caso de COUNT, que permite obtener el número de elemento de un grupo.

```
SELECT
   column_1,
   column_2,
   ...,
   aggregate_function(column_3)
FROM
   table_name
GROUP BY
   column_1,
   column_2,
   ...;
```

Lenguaje de consulta SQL

El AS permite agregar un nombre temporal a una columna o tabla para la ejecución de consultas. Es de gran utilidad cuando se tienen consultas compuestas y de gran tamaño.

```sql
SELECT column_name AS alias_name
FROM table_name;
```

```sql
SELECT
   first_name,
   last_name AS surname
FROM customer;
```

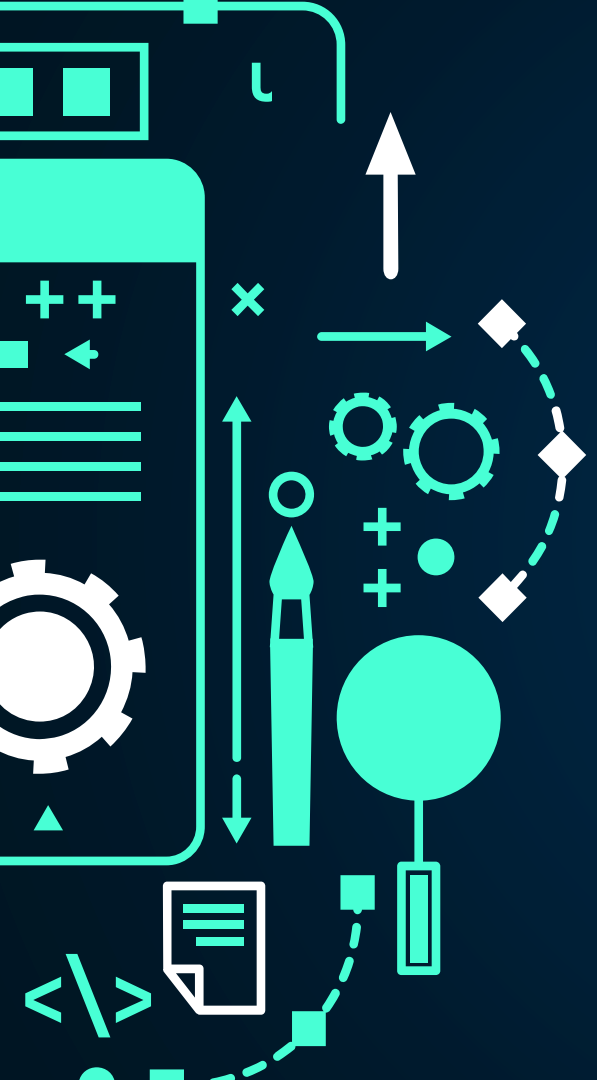
Actividad próxima clase

Realizar la lectura: “**Extending the database relational model to capture more meaning**” la cual se encuentra en el repositorio, en la carpeta de bibliografía.

Leer hasta la página 12 del PDF, sacar preguntas y conclusiones de la lectura, será de gran apoyo para el momento en que hagamos consultas combinadas.

REFERENCIAS

- El lenguaje SQL: https://www.dataprix.com/files/UOC_OpenSource_El_lenguaje_SQL.pdf
- Libro: Fundamentos de base de datos 5 edición, Silberschatz
- Tutorial de PostgreSQL: https://www.postgresqltutorial.com/



Gracias!