Revisión de una Metodología para la Extracción, Limpieza y Organización de Datos SQL con Programación Declarativa en Python para el Desarrollo de Análisis de Datos Descriptivos.

April 22, 2024

1 Procedimientos ETL

Recordemos que los procedimientos ETL constan de Extract, Transform y Load, los cuales nos dan la pauta a seguir para seguir el ciclo de la Ingeniería de Datos. De esta forma seguimos una metodología establecida y evitamos estar trabajando sin una dirección específica.

En este ejemplo, se utilizará la base de datos de Sakila, la cual es una base de datos ejemplo proporcionada por MySQL que es utilizada comúnmente para propósitos educativos, demostraciones y pruebas. Sakila simula una base de datos de una tienda de alquiler de películas, similar a la cadena de alquiler de vídeos Blockbuster que solía existir.

Sakila contiene tablas que representan películas, actores, clientes, tiendas, alquileres, etc.

Para hacer uso de ella, hemos descargado el archivo script SQL a través de la página oficial de MySQL: `https://dev.mysql.com/doc/index-other.html`.

Ahora para instalar la base de datos podremos utilizar los siguientes comandos de MySQL.

```
$ mysql -u root -p sakila < sakila-schema.sql
$ mysql -u root -p sakila < sakila-data.sql
```

Después de haber ejecutado estos comandos, deberíamos ser capaces de ver la base de datos

instalada en nuestro sistema, para comprobarlo podemos ejecutar:

```
mysql> USE sakila;
Database changed
mysql> SHOW TABLES;
+----------------------------+
| Tables_in_sakila           |
+----------------------------+
| actor                      |
| actor_info                 |
| address                    |
| category                   |
| city                       |
| country                    |
| customer                   |
| customer_list              |
| film                       |
| film_actor                 |
| film_category              |
| film_list                  |
| film_text                  |
| inventory                  |
| language                   |
| nicer_but_slower_film_list |
| payment                    |
| rental                     |
| sales_by_film_category     |
| sales_by_store             |
| staff                      |
| staff_list                 |
| store                      |
+----------------------------+
```

Ahora, con ello podremos comenzar a trabajar nuestro procedimiento ETL.

Primeramente, es necesario determinar que incógnita queremos responder sobre los datos disponibles. Esto variará dependiendo de la base de datos y los tipos de datos que ésta almacene. Para este caso, intentaremos responder las preguntas:

- **Patrones de alquier:** ¿Cuáles son las películas más alquiladas?
- **Clientes frecuentes:** ¿Quiénes son los clientes más frecuentes?, ¿Existe alguna relación entre la cantidad de películas alquiladas y la ubicación de las tiendas?

- **Popularidad de las categorías de películas:** ¿Cuáles son las categorías de películas más populares entre los clientes?
- **Ingresos por película:** ¿Cuáles son las películas que generan más ingresos en términos de alquiler?

Una vez planteadas las incógnitas, podemos determinas que tablas serán relevantes para nuestro análisis, las cuales son:

- **Patrones de alquier:**
    1. rental
    2. inventory
    3. film
- **Clientes frecuentes:**
    1. customer
    2. rental
    3. inventory
    4. store
- **Popularidad de las categorías de películas:**
    1. film_category
    2. category
- **Ingresos por película:**
    1. film
    2. rental
    3. payment

1.1 Extract.

Para nuestro ejemplo, utilizaremos Python con SQL Alchemy.

```
# utilizar sqlalchemy
```

1.2 Transform.

Ahora bien, después de haber extraído los datos, es necesario organizarlos de manera que sea más sencillo utilizarlos para el análisis de datos.

```
# utilizar pandas
```

1.3 Load.

Una vez que se haya realizado la Organización de los datos, es una buena idea es crear un acceso a esta información de manera que no sea necesario acceder a la información y organizarla nuevamente, para esto son muy útiles las vistas (views) de SQL.

```
# utilizar sqlalchemy
```