Introducci?n a las bases de datos relacionales con Python

November 8, 2017

1 Creando un motor de bases de datos en Python

A continuación vamos a proceder a activar nuestro primer motor de bases de datos SQL. A continuación crearemos el motor y nos conectaremos con las bases de datos SQLite Chinook.sqlite. Para ello en primer lugar debemos importar la función create_engine de la librería sqlalchemy. Tras esto crearemos la conexión, para ello debemos pasar como parámetro a la función create_engine un string donde indicamos el tipo de base de datos con la que vamos a conectar en este caso será sqlite seguido del string :/// y tras esto el nombre de la base de datos.

```
In [1]: #Importamos create_engine
        from sqlalchemy import create_engine
        #Creamos la conexión
        engine = create_engine('sqlite:///Chinook.sqlite')
```

El siguiente paso es conocer el número de tablas que componen nuestra base de datos, para ello contamos con el método table_names().

```
In [2]: #Mostramos las diferentes tablas que componen nuestra base de datos
        tables = engine.table_names()
        print(tables)
```

```
['Album', 'Artist', 'Customer', 'Employee', 'Genre', 'Invoice', 'InvoiceLine', 'MediaType', 'Pla
```

Primera consulta SQL

Cuando deseamos realizar una consulta en SQL en Python debemos de seguir los siguientes pasos:

```
1) Importar paquetes y funciones necesarias
2) Crear un motor de conexión
3) Conectar con la base de datos
4) Realizar nuestra consulta SQL
5) Salvar los resultados de dicha consulta como un DataFrame
6) Cerrar la conexión
```

La consulta básica, lo que viene a ser el Hello World en SQL es la consulta SELECT * FROM TABLE_NAME, esta consulta lo que hace es importar toda la información de la tabla indicada en TABLE_NAME. Para conectar con la base de datos disponemos del método connect() y para

cerrar nuestra conexión contamos con el método close(). A la hora de realizar la consulta debemos de hacer uso del método execute(), donde le pasamos la consulta que deseamos realizar. Para guardar los datos como un dataframe debemos de hacer uso del método DataFrame de la librería pandas a la cual le pasamos como parámetro respues_consulta-fetchall().

```
In [3]: #Importamos paquetes
        from sqlalchemy import create_engine
        import pandas as pd
        #Creamos el motor de conexión
        engine = create_engine('sqlite:///Chinook.sqlite')
        #Conectamos con la base de datos
        con = engine.connect()
        #Realizamos nuestra primera consulta
        rs = engine.execute('SELECT * FROM Album')
        #Salvamos los resulados como un DataFrame
        df = pd.DataFrame(rs.fetchall())
        #Cerramos la conexión a nuestra base de datos
        con.close()
        #Mostramos las 5 primeras filas de nuestra consulta
        print(df.head())
```

```
   0                                     1  2
0  1  For Those About To Rock We Salute You  1
1  2                      Balls to the Wall  2
2  3                      Restless and Wild  2
3  4                      Let There Be Rock  1
4  5                               Big Ones  3
```

Podemos ver como nuestra consulta se ha realizado de forma correcta, pero no como esperabamos ya que por ejemplo las cabeceras no han sido cargadas. Para ello podemos hacer uso del método keys().

```
In [4]: with engine.connect() as con:
            rs = engine.execute('SELECT Lastname, Title FROM Employee')
            df = pd.DataFrame(rs.fetchmany(size = 3))
            df.columns = rs.keys()

        print(len(df))
        print(df.head())
```

```
3
  LastName                Title
0    Adams      General Manager
1  Edwards        Sales Manager
2  Peacock  Sales Support Agent
```

En este caso podemos ver como haciendo uso del método keys() las cabeceras de nuestro DataFrame se han cargado. Con respecto a la llamada anterior se aprecian una serie de diferecncias: la primera es que en este caso no hemos seleccionado la tabla completa sino que hemos seleccionado aquellas columnas que nos interesa importar, para ello basta con indicarle tras el SELECT los nombres de las columnas que deseamos importar. La segunda diferencia es que hemos hecho uso de fetchmany en lugar de fetchall, ya que en este caso no queremos pasar a DataFrame la tabla completa sino un número determinado de filas para ello fetchmany cuenta con el parámetro size donde le indicamos el número de filas que deseamos importar.

El WHERE dentro de una consulta SQL nos permite filtrar dicha consulta, seleccionando únicamente aquellas filas que cumplen la condición de la columna indicada. Por ejemmplo si de la table Employee deseamos seleccionar aquellos empleados cuyo ID es mayor que 6 debemos realizar la siguiente consulta.

```
In [6]: with engine.connect() as con:
            rs = engine.execute('SELECT * FROM Employee WHERE EmployeeId >= 6')
            df = pd.DataFrame(rs.fetchall())
            df.columns = rs.keys()

        print(df.shape)
        print(df.head())
```

```
(3, 15)
   EmployeeId  LastName FirstName       Title  ReportsTo            BirthDate  \
0           6  Mitchell   Michael  IT Manager          1  1973-07-01 00:00:00
1           7      King    Robert    IT Staff          6  1970-05-29 00:00:00
2           8  Callahan     Laura    IT Staff          6  1968-01-09 00:00:00

              HireDate                     Address        City State Country  \
0  2003-10-17 00:00:00         5827 Bowness Road NW     Calgary    AB  Canada
1  2004-01-02 00:00:00  590 Columbia Boulevard West  Lethbridge    AB  Canada
2  2004-03-04 00:00:00                 923 7 ST NW  Lethbridge    AB  Canada

  PostalCode              Phone                Fax                    Email
0    T3B 0C5  +1 (403) 246-9887  +1 (403) 246-9899  michael@chinookcorp.com
1    T1K 5N8  +1 (403) 456-9986  +1 (403) 456-8485   robert@chinookcorp.com
2    T1H 1Y8  +1 (403) 467-3351  +1 (403) 467-8772    laura@chinookcorp.com
```

El ORDER BY nos permite ordenar los resultados de una consulta, es decir, nos permite ordenar nuestros resultados de forma creciente, decreciente por una columna etc. Por ejemplo si queremos ordenar nuestra consulta a la base de datos Employee en función de la fecha de nacimiento de los empleados debemos de realizar la siguiente consulta.

```
In [7]: with engine.connect() as con:
            rs = engine.execute('SELECT * FROM Employee ORDER BY BirthDate')
            df = pd.DataFrame(rs.fetchall())
            df.columns = rs.keys()

        print(df.head())
```

```
   EmployeeId  LastName FirstName                Title  ReportsTo  \
0           4      Park  Margaret  Sales Support Agent        2.0
1           2   Edwards     Nancy        Sales Manager        1.0
2           1     Adams    Andrew      General Manager        NaN
3           5   Johnson     Steve  Sales Support Agent        2.0
4           8  Callahan     Laura             IT Staff        6.0

             BirthDate             HireDate              Address        City  \
0  1947-09-19 00:00:00  2003-05-03 00:00:00     683 10 Street SW     Calgary
1  1958-12-08 00:00:00  2002-05-01 00:00:00         825 8 Ave SW     Calgary
2  1962-02-18 00:00:00  2002-08-14 00:00:00  11120 Jasper Ave NW    Edmonton
3  1965-03-03 00:00:00  2003-10-17 00:00:00         7727B 41 Ave     Calgary
4  1968-01-09 00:00:00  2004-03-04 00:00:00          923 7 ST NW  Lethbridge

  State Country PostalCode              Phone                Fax  \
0    AB  Canada    T2P 5G3  +1 (403) 263-4423  +1 (403) 263-4289
1    AB  Canada    T2P 2T3  +1 (403) 262-3443  +1 (403) 262-3322
2    AB  Canada    T5K 2N1  +1 (780) 428-9482  +1 (780) 428-3457
3    AB  Canada    T3B 1Y7   1 (780) 836-9987   1 (780) 836-9543
4    AB  Canada    T1H 1Y8  +1 (403) 467-3351  +1 (403) 467-8772

                    Email
0  margaret@chinookcorp.com
1     nancy@chinookcorp.com
2    andrew@chinookcorp.com
3     steve@chinookcorp.com
4     laura@chinookcorp.com
```

2 Realizando consultas SQL directamente con Pandas

Es posible realizar consultas SQL directamente con la librería pandas. Esta librería nos permite hacer uso del método read_sql_query() al cuál cuenta con dos parámetros principales: el código de consulta y el motor de de consulta.

```
In [1]: #Importamos pandas
        from sqlalchemy import create_engine
        import pandas as pd
        #Nos creamos el motor de consulta
        engine = create_engine('sqlite:///Chinook.sqlite')
        #Realizamos la consulta a través de pandas
        df = pd.read_sql_query('SELECT * FROM Album', engine)
        print(df.head())
```

```
   AlbumId                                 Title  ArtistId
0        1  For Those About To Rock We Salute You         1
1        2                      Balls to the Wall         2
2        3                      Restless and Wild         2
```

```
3        4                      Let There Be Rock         1
4        5                                Big Ones         3
```

A continuación vamos a proceder a realizar una consulta un poco más compleja haciendo uso de pandas.

```
In [3]: #Importamos las librerías necesarias
        from sqlalchemy import create_engine
        import pandas as pd
        #Nos creamos el motor
        engine = create_engine('sqlite:///Chinook.sqlite')
        #Realizamos la consulta SQL
        df = pd.read_sql_query('SELECT * FROM Employee WHERE EmployeeId >= 6 ORDER BY Birthdate'
        print(df.head())
```

```
   EmployeeId  LastName FirstName       Title  ReportsTo            BirthDate  \
0           8  Callahan     Laura    IT Staff          6  1968-01-09 00:00:00
1           7      King    Robert    IT Staff          6  1970-05-29 00:00:00
2           6  Mitchell   Michael  IT Manager          1  1973-07-01 00:00:00

              HireDate                     Address        City State Country  \
0  2004-03-04 00:00:00                 923 7 ST NW  Lethbridge    AB  Canada
1  2004-01-02 00:00:00  590 Columbia Boulevard West  Lethbridge    AB  Canada
2  2003-10-17 00:00:00         5827 Bowness Road NW     Calgary    AB  Canada

  PostalCode              Phone                Fax                    Email
0    T1H 1Y8  +1 (403) 467-3351  +1 (403) 467-8772    laura@chinookcorp.com
1    T1K 5N8  +1 (403) 456-9986  +1 (403) 456-8485   robert@chinookcorp.com
2    T3B 0C5  +1 (403) 246-9887  +1 (403) 246-9899  michael@chinookcorp.com
```

3 Inner Join

Supongamos que de la tabla Album deseamos extraer el título junto con el nombre del artista, la información respecto al nombre del artisita lo tenemos en la tabla artist, podemos ver como estas dos tablas comparten las columnas ArtistId por lo que podemos juntar dichas tablas mediante lo que se conoce como un InnerJoin.

```
In [4]: #Importamos las librerías
        import pandas as pd
        from sqlalchemy import create_engine
        #Creamos el motor
        engine = create_engine('sqlite:///Chinook.sqlite')
        #Mostramos las tablas
        print(engine.table_names())
```

```
['Album', 'Artist', 'Customer', 'Employee', 'Genre', 'Invoice', 'InvoiceLine', 'MediaType', 'Pla
```

```
In [6]: #Hacemos el InnerJoin
        df = pd.read_sql_query('SELECT Title, Name FROM Album INNER JOIN Artist on Album.ArtistI
        print(df.head())
```

```
                                     Title       Name
0  For Those About To Rock We Salute You      AC/DC
1                      Balls to the Wall     Accept
2                      Restless and Wild     Accept
3                      Let There Be Rock      AC/DC
4                              Big Ones  Aerosmith
```