

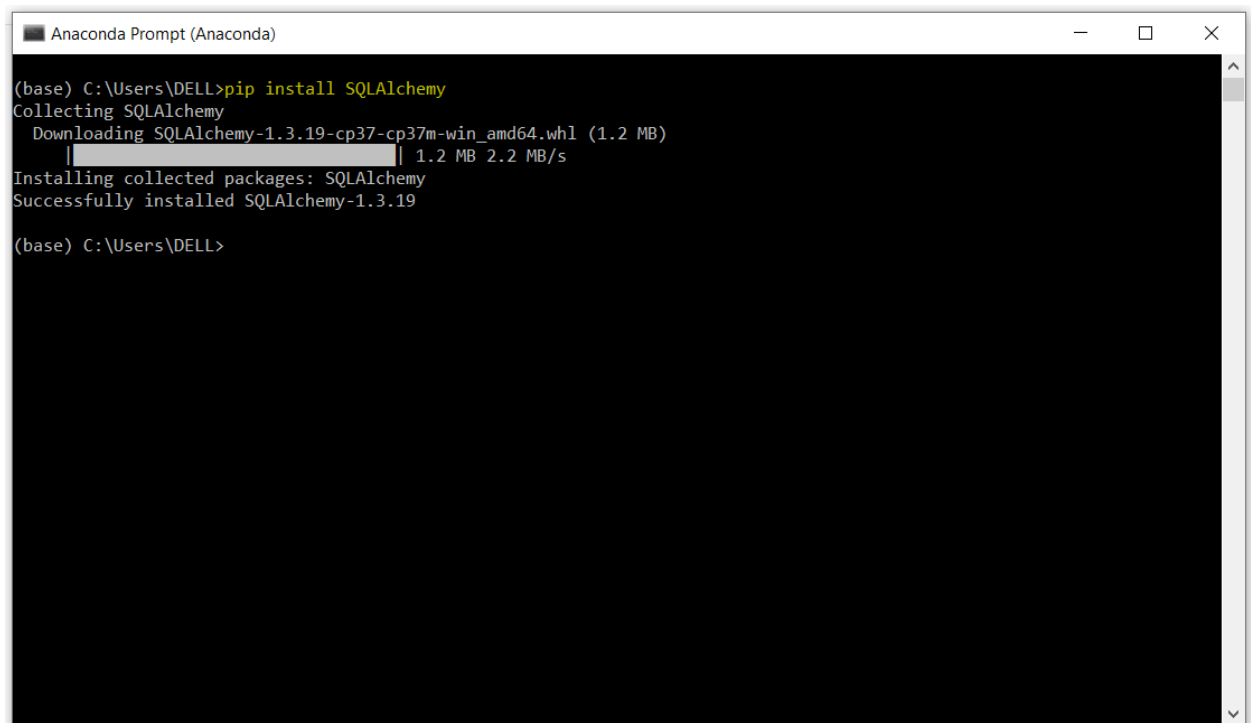
Prepared by Asif Bhat

Access Databases using Jupyter Notebook

Prerequisites

- Install Oracle Database 19c.
- Install MS SQL Server 2019 Express edition.
- Install PostgreSQL
- Install MySQL

Install SQLAlchemy

A screenshot of the Anaconda Prompt (Anaconda) window. The terminal shows the command `(base) C:\Users\DELL>pip install SQLAlchemy` being executed. The output indicates that SQLAlchemy is being collected, the specific wheel file `SQLAlchemy-1.3.19-cp37-cp37m-win_amd64.whl (1.2 MB)` is being downloaded at `1.2 MB 2.2 MB/s`, and it is successfully installed. The prompt then returns to `(base) C:\Users\DELL>`.

```
(base) C:\Users\DELL>pip install SQLAlchemy
Collecting SQLAlchemy
  Downloading SQLAlchemy-1.3.19-cp37-cp37m-win_amd64.whl (1.2 MB)
    | 1.2 MB 2.2 MB/s
Installing collected packages: SQLAlchemy
Successfully installed SQLAlchemy-1.3.19

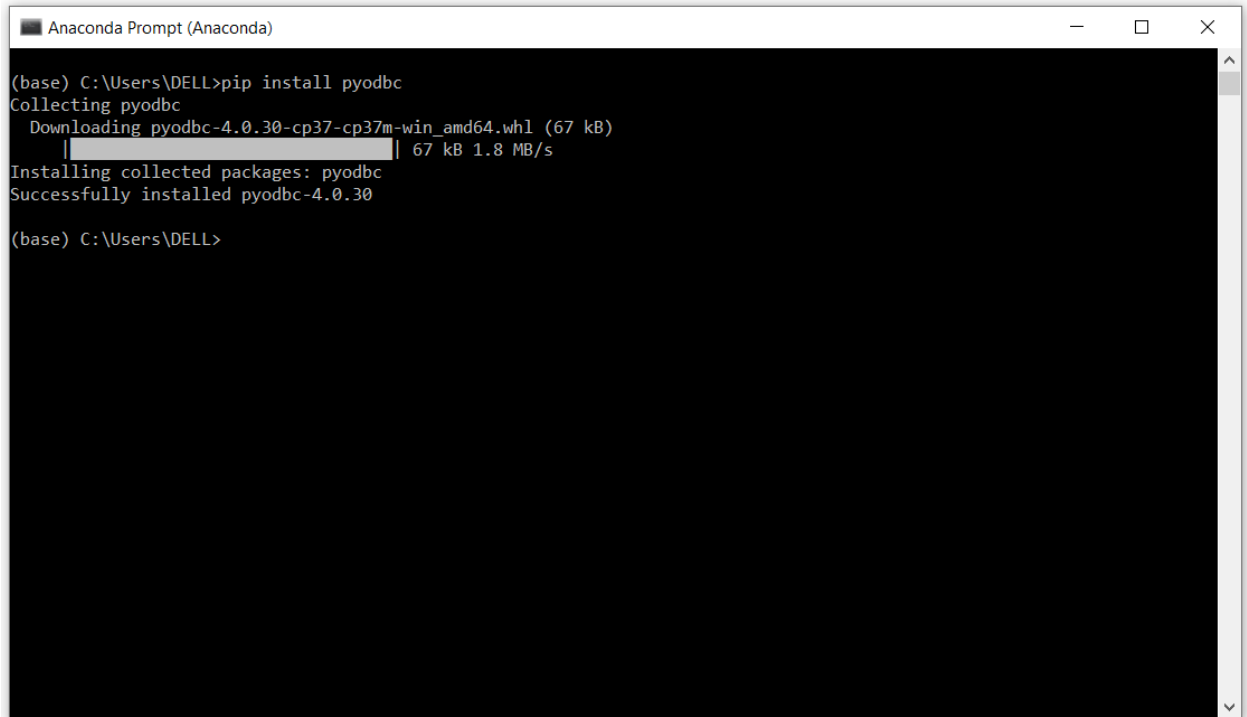
(base) C:\Users\DELL>
```

Install ipython-sql

```
Anaconda Prompt (Anaconda)
(base) C:\Users\DELL>pip install ipython-sql
Collecting ipython-sql
  Using cached ipython_sql-0.4.0-py3-none-any.whl (19 kB)
Requirement already satisfied: sqlalchemy>=0.6.7 in c:\anaconda\lib\site-packages (from ipython-sql) (1.3.19)
Requirement already satisfied: sqlparse in c:\anaconda\lib\site-packages (from ipython-sql) (0.3.1)
Requirement already satisfied: six in c:\anaconda\lib\site-packages (from ipython-sql) (1.14.0)
Requirement already satisfied: prettytable<1 in c:\anaconda\lib\site-packages (from ipython-sql) (0.7.2)
Requirement already satisfied: ipython>=1.0 in c:\anaconda\lib\site-packages (from ipython-sql) (7.12.0)
Requirement already satisfied: ipython-genutils>=0.1.0 in c:\anaconda\lib\site-packages (from ipython-sql) (0.2.0)
Requirement already satisfied: setuptools>=18.5 in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (45.2.0.post20200210)
Requirement already satisfied: decorator in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (4.4.1)
Requirement already satisfied: traitlets>=4.2 in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (4.3.3)
Requirement already satisfied: pickleshare in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (0.7.5)
Requirement already satisfied: backcall in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (0.1.0)
Requirement already satisfied: prompt-toolkit!=3.0.0,!<3.0.1,<3.1.0,>=2.0.0 in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (3.0.3)
Requirement already satisfied: colorama; sys_platform == "win32" in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (0.4.3)
Requirement already satisfied: jedi>=0.10 in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (0.14.1)
Requirement already satisfied: pygments in c:\anaconda\lib\site-packages (from ipython>=1.0->ipython-sql) (2.5.2)
Requirement already satisfied: wcwidth in c:\anaconda\lib\site-packages (from prompt-toolkit!=3.0.0,!<3.0.1,<3.1.0,>=2.0.0->ipython>=1.0->ipython-sql) (0.1.8)
Requirement already satisfied: parso>=0.5.0 in c:\anaconda\lib\site-packages (from jedi>=0.10->ipython>=1.0->ipython-sql) (0.5.2)
Installing collected packages: ipython-sql
Successfully installed ipython-sql-0.4.0

(base) C:\Users\DELL>
```

Install Pyodbc

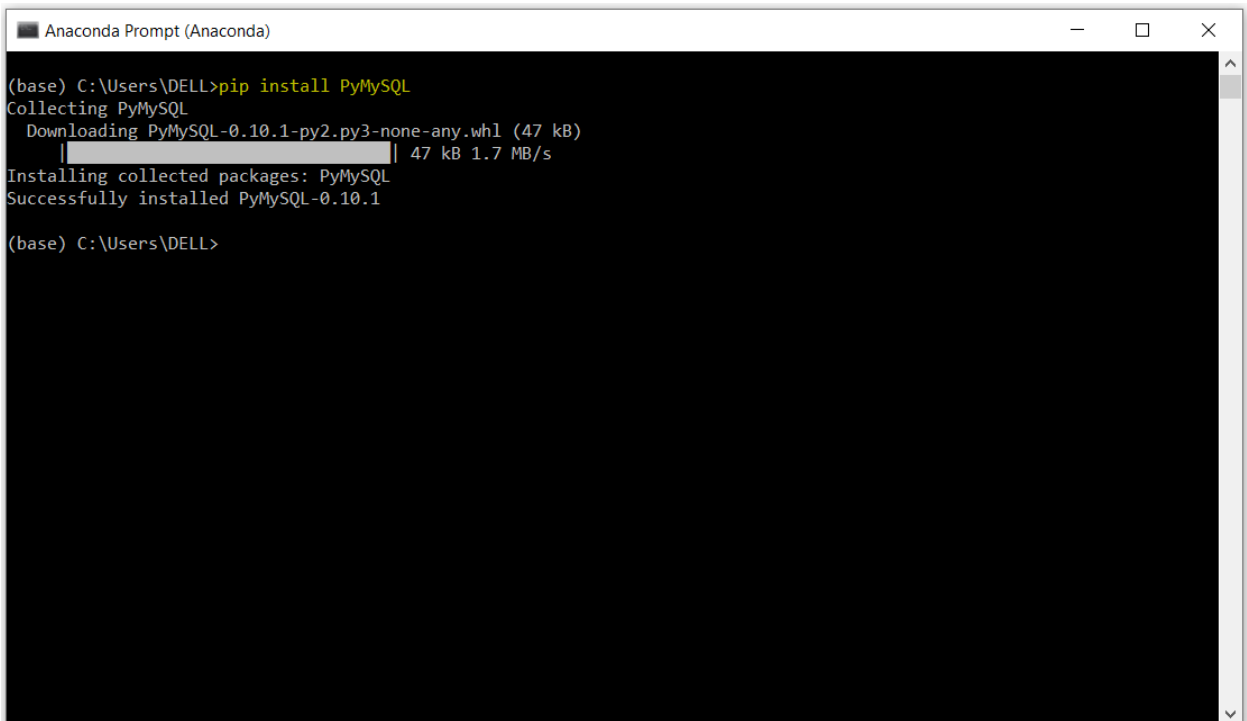


```
Anaconda Prompt (Anaconda)

(base) C:\Users\DELL>pip install pyodbc
Collecting pyodbc
  Downloading pyodbc-4.0.30-cp37-cp37m-win_amd64.whl (67 kB)
    | 67 kB 1.8 MB/s
Installing collected packages: pyodbc
Successfully installed pyodbc-4.0.30

(base) C:\Users\DELL>
```

Install PyMySQL

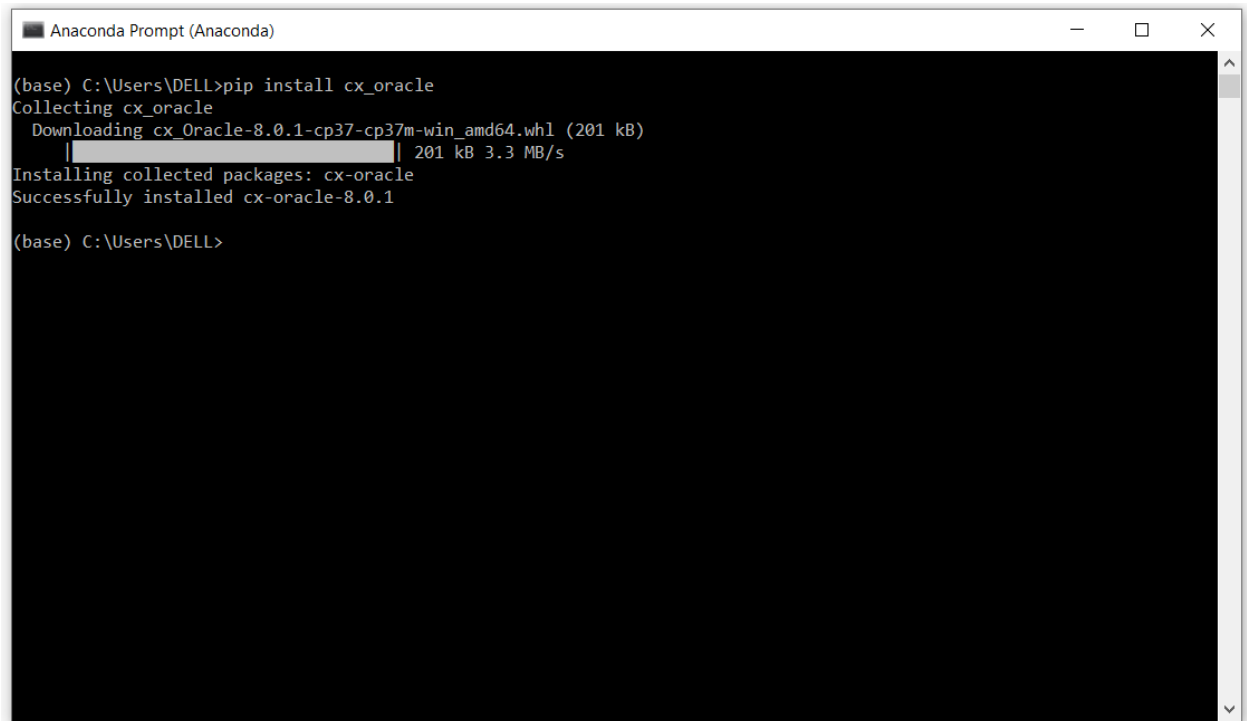


```
Anaconda Prompt (Anaconda)

(base) C:\Users\DELL>pip install PyMySQL
Collecting PyMySQL
  Downloading PyMySQL-0.10.1-py2.py3-none-any.whl (47 kB)
    | 47 kB 1.7 MB/s
Installing collected packages: PyMySQL
Successfully installed PyMySQL-0.10.1

(base) C:\Users\DELL>
```

Install cx_oracle

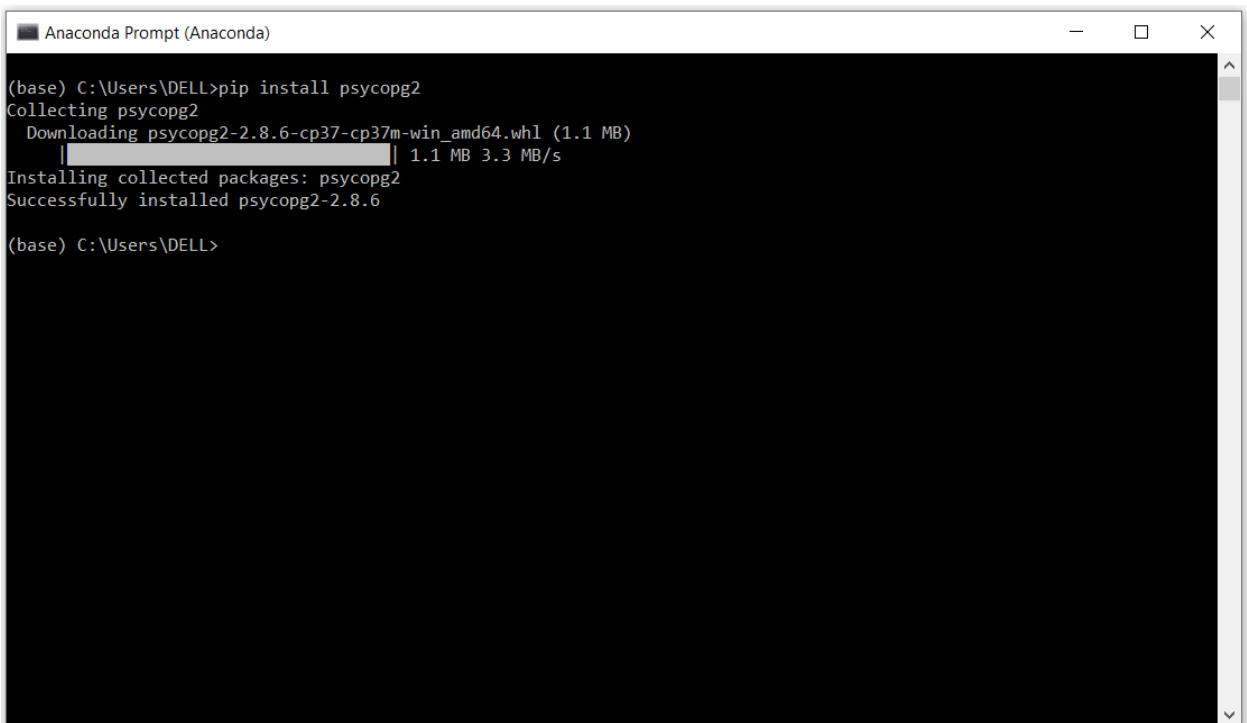


```
Anaconda Prompt (Anaconda)

(base) C:\Users\DELL>pip install cx_oracle
Collecting cx_oracle
  Downloading cx_Oracle-8.0.1-cp37-cp37m-win_amd64.whl (201 kB)
    |-----| 201 kB 3.3 MB/s
Installing collected packages: cx-oracle
Successfully installed cx-oracle-8.0.1

(base) C:\Users\DELL>
```

Install psycopg2



```
Anaconda Prompt (Anaconda)

(base) C:\Users\DELL>pip install psycopg2
Collecting psycopg2
  Downloading psycopg2-2.8.6-cp37-cp37m-win_amd64.whl (1.1 MB)
    |-----| 1.1 MB 3.3 MB/s
Installing collected packages: psycopg2
Successfully installed psycopg2-2.8.6

(base) C:\Users\DELL>
```

```
In [1]: import sqlalchemy
import numpy
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

Connec to SQL Server Database using sqlalchemy

Sample Database Used - <https://www.sqlservertutorial.net/sql-server-sample-database/>
(<https://www.sqlservertutorial.net/sql-server-sample-database/>)

- <https://www.sqlskills.com/resources/conferences/salesdb2014.zip>

```
In [2]: engine_mssql = sqlalchemy.create_engine('mssql+pyodbc://localhost/salesdb?d
```

```
In [3]: print("connecting with engine " + str(engine_mssql))

connecting with engine Engine(mssql+pyodbc://localhost/salesdb?driver=SQL
+Server)
```

```
In [4]: print(engine_mssql.table_names())

['Customers', 'Employees', 'Products', 'Sales']
```

```
In [5]: con_mssql = engine_mssql.connect()
```

```
In [6]: query = "select * from Employees"
df_mssql = pd.read_sql_query(query, con_mssql)
df_mssql
```

```
Out[6]:
```

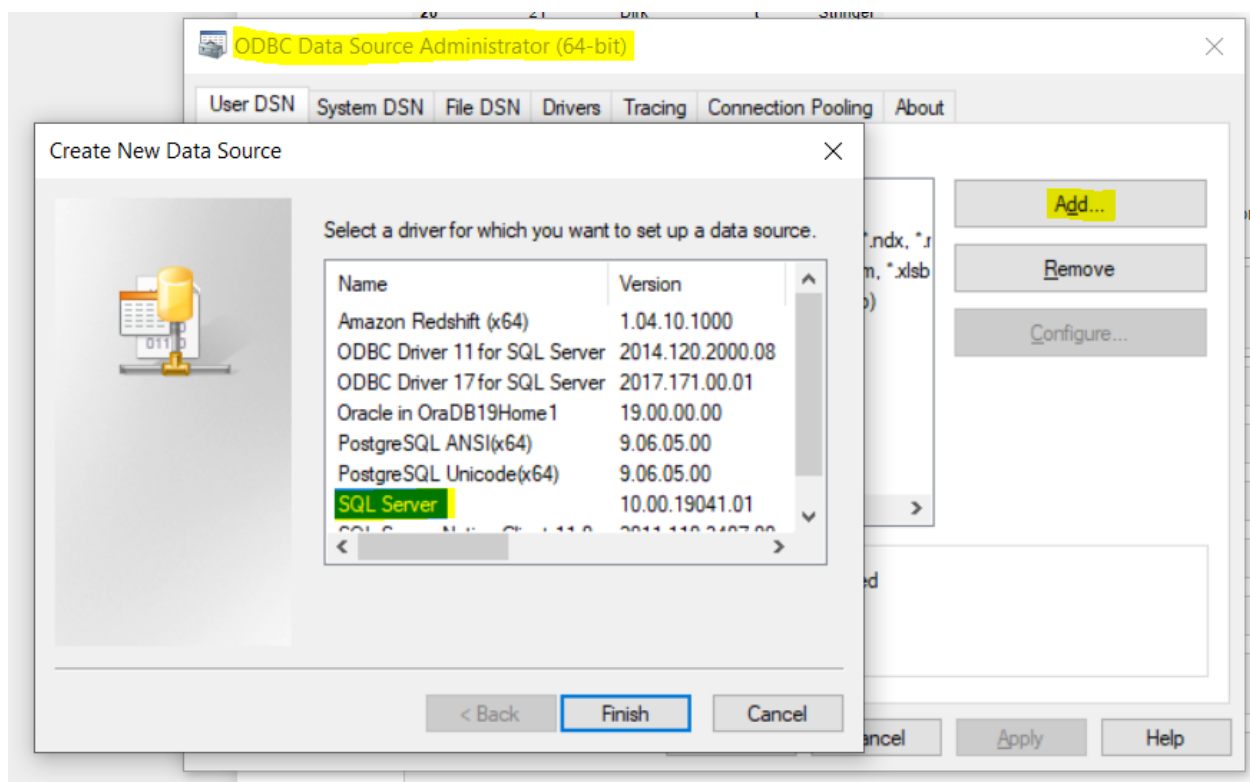
	EmployeeID	FirstName	MiddleInitial	LastName
0	1	Abraham	e	Bennet
1	2	Reginald	l	Blotchet-Halls
2	3	Cheryl	a	Carson
3	4	Michel	e	DeFrance
4	5	Innes	e	del Castillo
5	6	Ann	u	Dull
6	7	Marjorie	r	Green
7	8	Morningstar	r	Greene
8	9	Burt	r	Gringlesby
9	10	Sheryl	u	Hunter
10	11	Livia	a	Karsen
11	12	Charlene	o	Locksley
12	13	Stearns	a	MacFeather
13	14	Heather	c	McBadden
14	15	Michael	'	O'Leary
15	16	Sylvia	a	Panteley
16	17	Albert	i	Ringer
17	18	Anne	i	Ringer
18	19	Meander	m	Smith
19	20	Dean	t	Straight
20	21	Dirk	t	Stringer
21	22	Johnson	h	White
22	23	Akiko	o	Yokomoto

Connec to SQL Server Database using ipython-sql

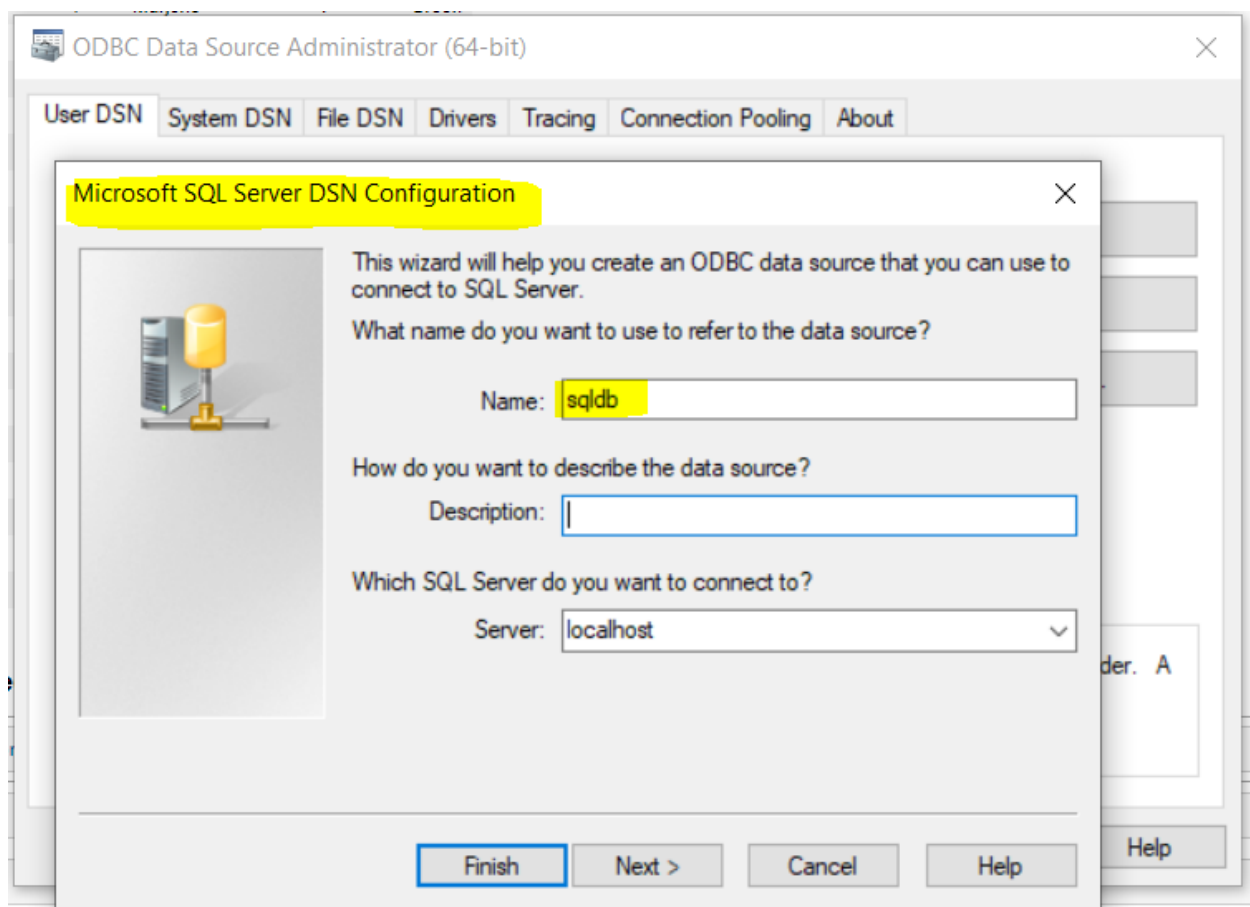
To Connect to SQL Server Database using ipython-sql we need to first create a odbc connection

Steps to create odbc connection

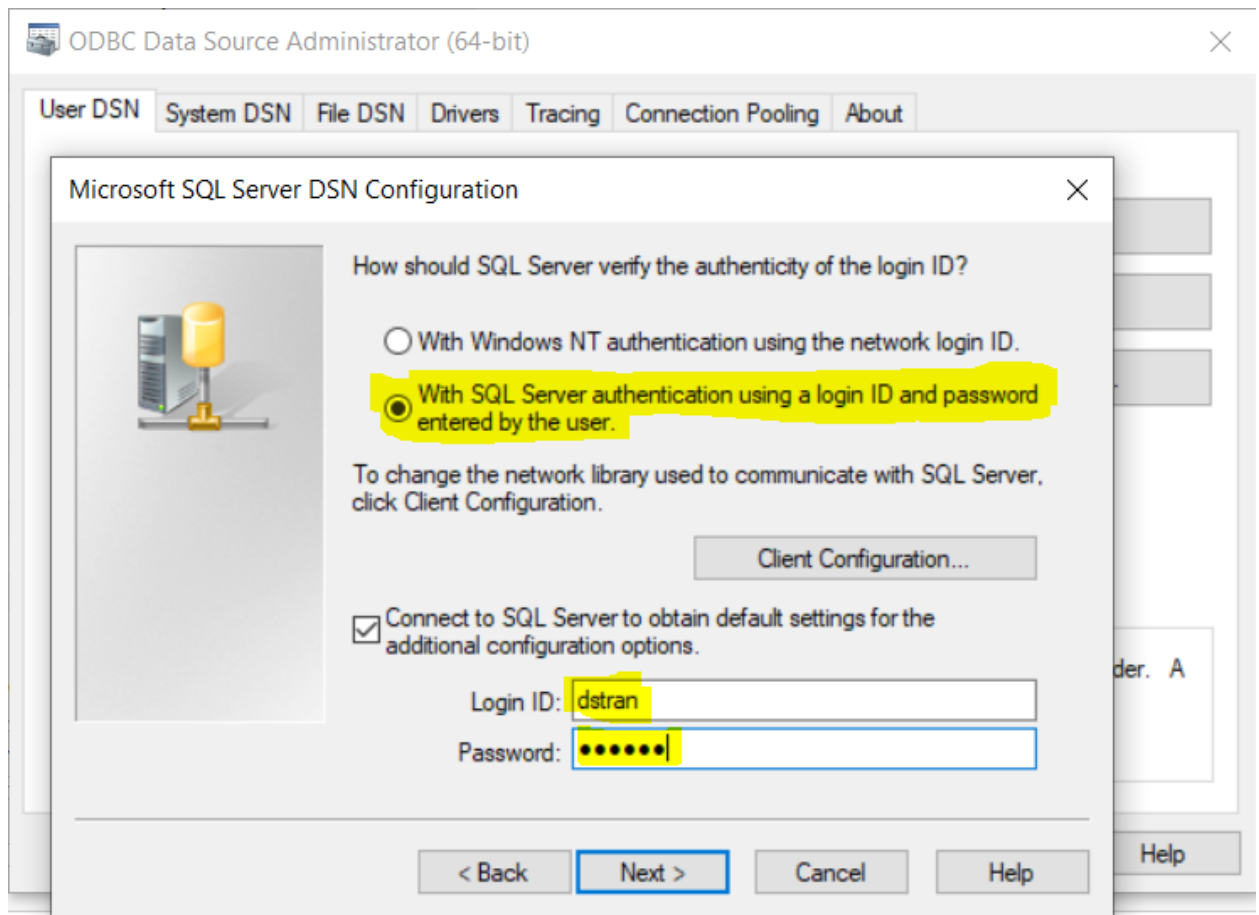
1. Go to ODBC Data Source Administration.
2. Click on Add and select the **SQL Serer** and hit Finish.



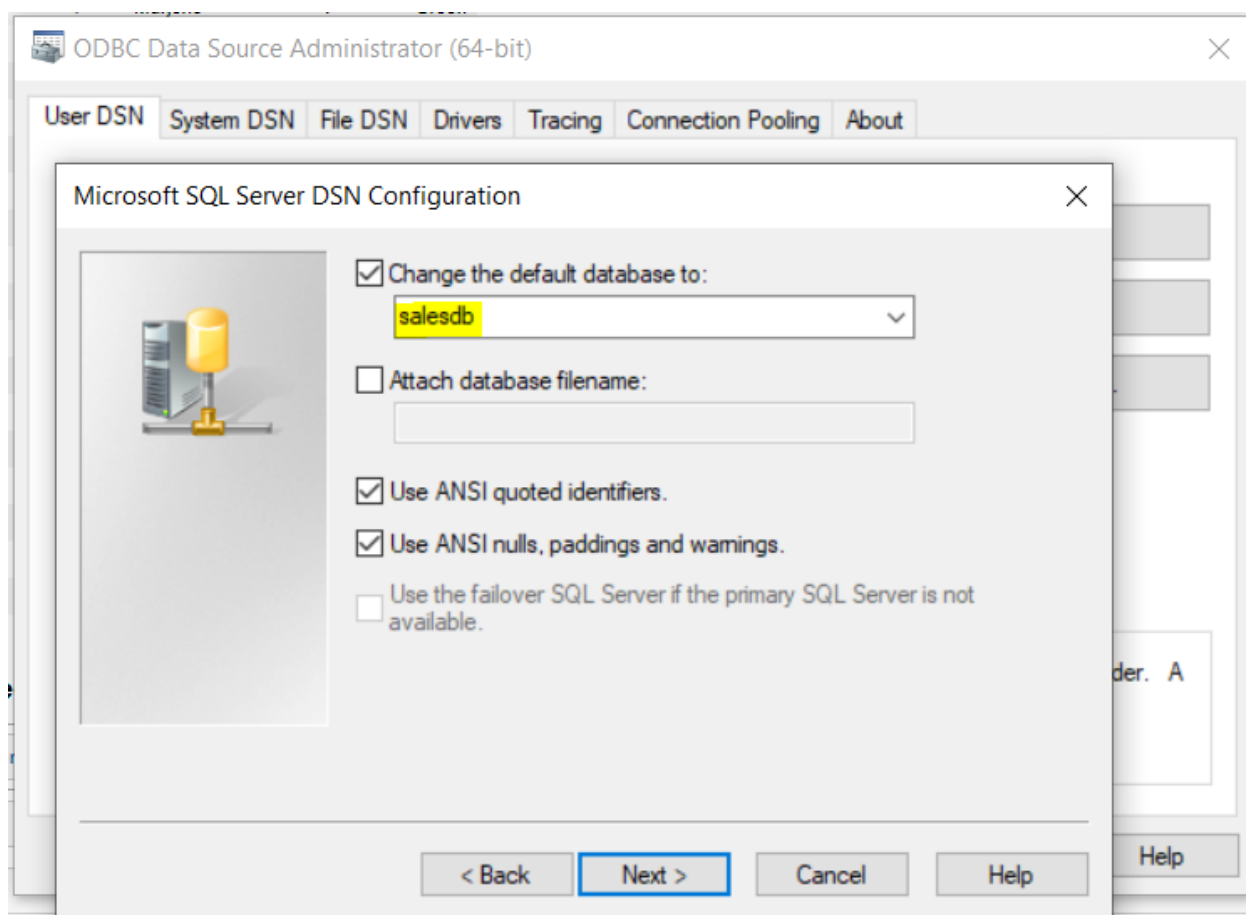
3. Name your ODBC connection (sqldb).
4. Specify SQL Server name and hit finish.



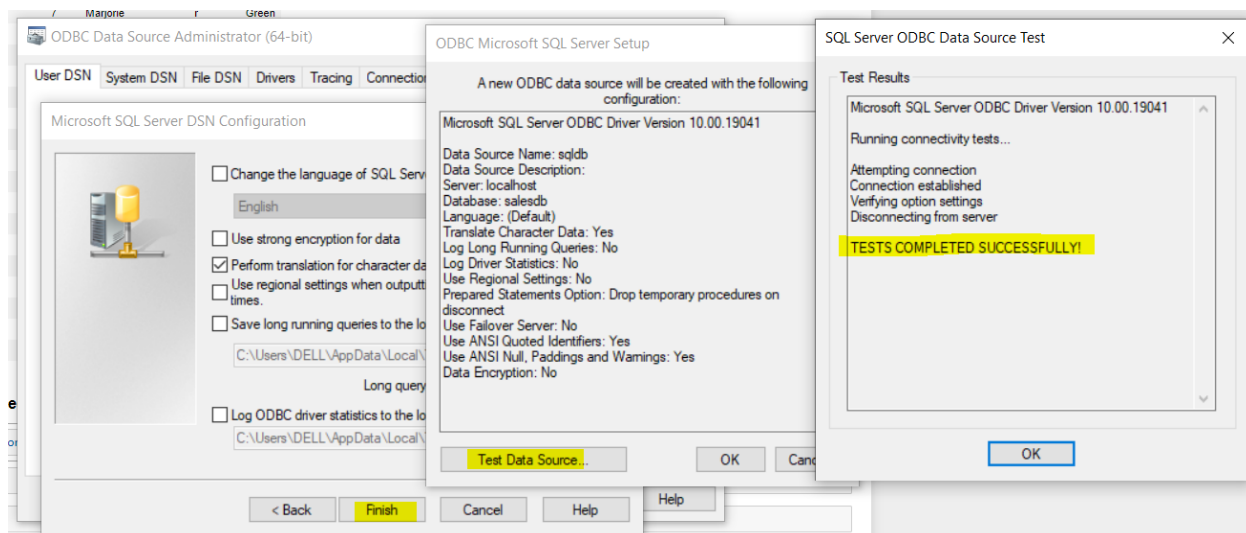
5. Use SQL Server authentication to connect to the database (Salesdb in this case).



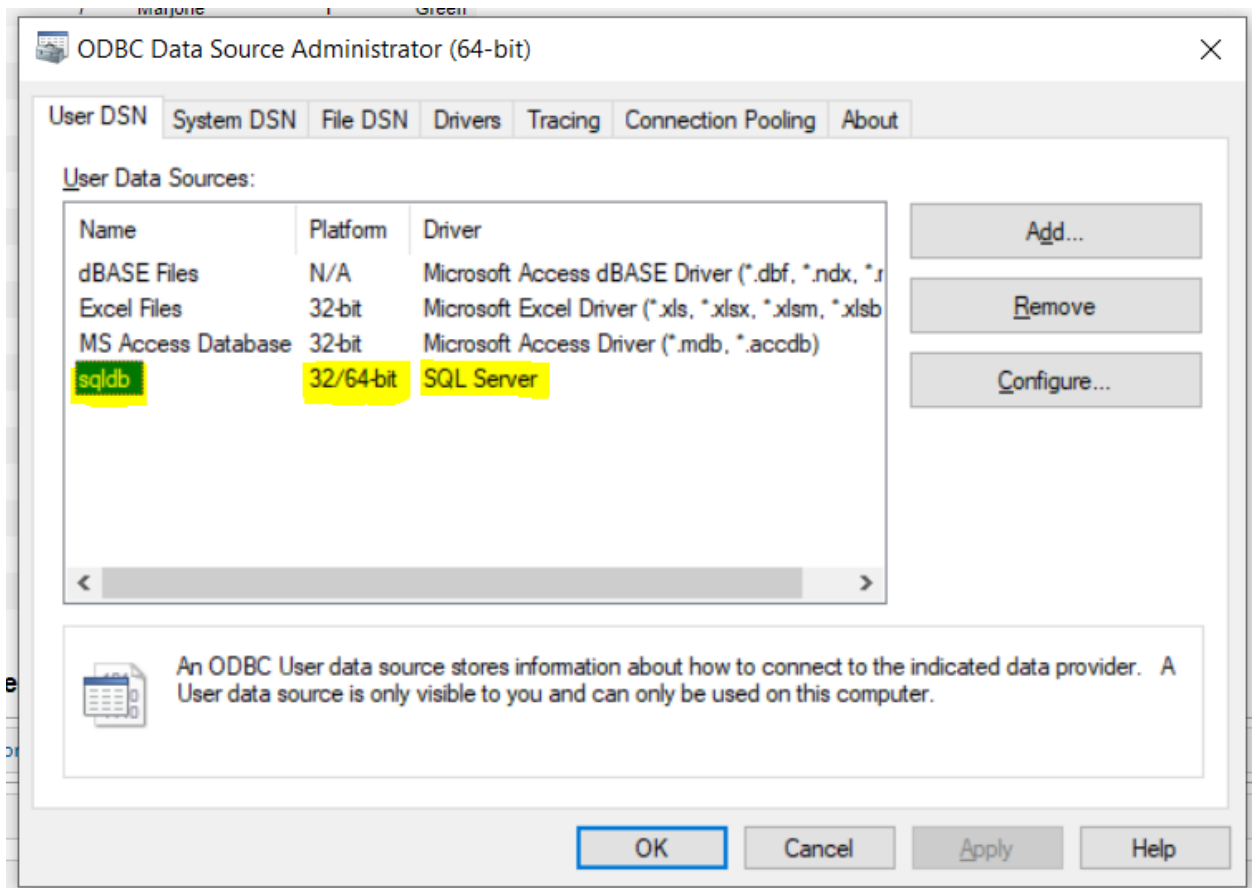
6. Select the database (salesdb) and click next.



7. Test Data Source and hit finish.



8. sqldb data source will be visible in the list now.



```
In [7]: %load_ext sql
        %sql mssql+pyodbc://dstran:dstran@sqlldb
```

```
(pyodbc.ProgrammingError) ('42000', "[42000] [Microsoft][ODBC SQL Server
Driver][SQL Server]Login failed for user 'dstran'. Reason: The password
of the account has expired. (18487) (SQLDriverConnect); [42000] [Microsof
t][ODBC SQL Server Driver][SQL Server]Login failed for user 'dstran'. Re
ason: The password of the account has expired. (18487)")
(Background on this error at: http://sqlalche.me/e/13/f405) (http://sqlalche.me/e/13/f405)
Connection info needed in SQLAlchemy format, example:
    postgresql://username:password@hostname/dbname
or an existing connection: dict_keys([])
```

```
In [8]: %%sql

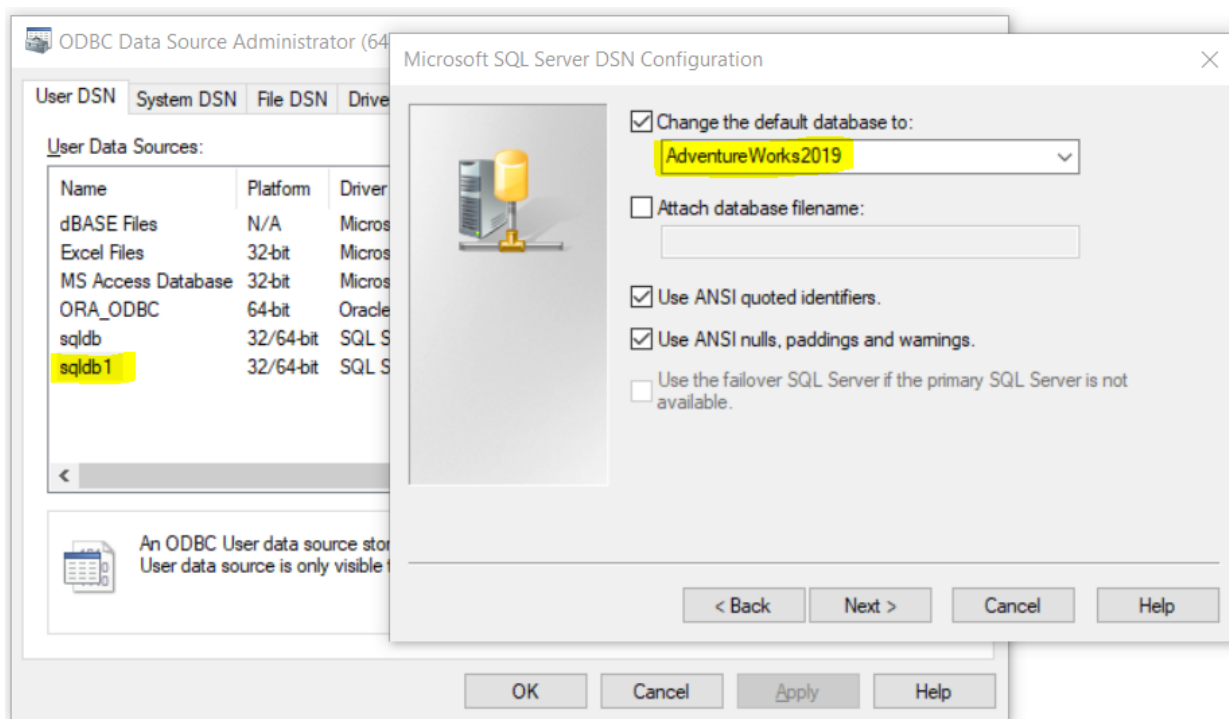
select top 10 * from Employees
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
    postgresql://username:password@hostname/dbname
or an existing connection: dict_keys([])
```

```
In [9]: %reload_ext sql
```

```
In [10]: %load_ext sql
%sql mssql+pyodbc://abhat:abhat@sqldb1
```

The sql extension is already loaded. To reload it, use:
`%reload_ext sql`



In [13]: %%sql

```
select top 10 * from Sales.SpecialOffer
```

```
* mssql+pyodbc://abhat:***@sqlldb1
```

```
mssql+pyodbc://dstran:***@sqlldb
```

Done.

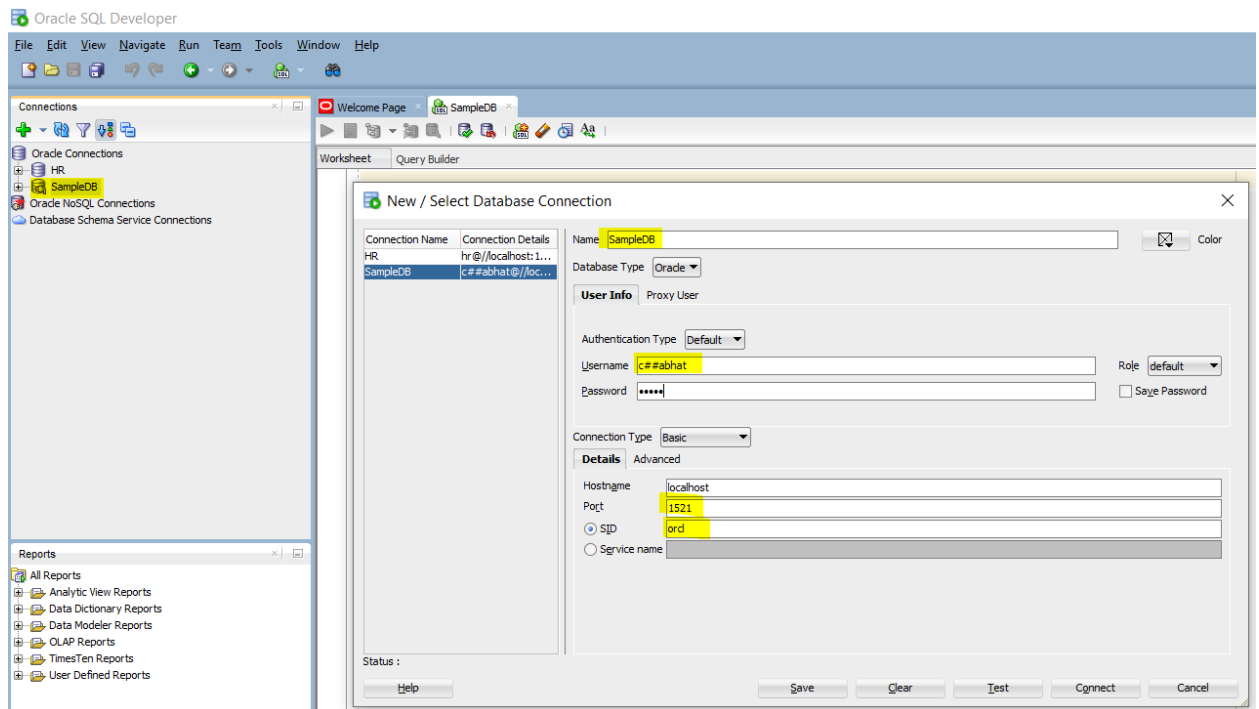
Out[13]:

	SpecialOfferID	Description	DiscountPct	Type	Category	StartDate	EndDate	MinQty	MaxQ
	1	No Discount	0.0000	No Discount	No Discount	2011-05-01 00:00:00	2014-11-30 00:00:00	0	Nor
	2	Volume Discount 11 to 14	0.0200	Volume Discount	Reseller	2011-05-31 00:00:00	2014-05-30 00:00:00	11	1
	3	Volume Discount 15 to 24	0.0500	Volume Discount	Reseller	2011-05-31 00:00:00	2014-05-30 00:00:00	15	2
	4	Volume Discount 25 to 40	0.1000	Volume Discount	Reseller	2011-05-31 00:00:00	2014-05-30 00:00:00	25	4
	5	Volume Discount 41 to 60	0.1500	Volume Discount	Reseller	2011-05-31 00:00:00	2014-05-30 00:00:00	41	6
	6	Volume Discount over 60	0.2000	Volume Discount	Reseller	2011-05-31 00:00:00	2014-05-30 00:00:00	61	Nor
	7	Mountain-100 Clearance Sale	0.3500	Discontinued Product	Reseller	2012-04-13 00:00:00	2012-05-29 00:00:00	0	Nor
	8	Sport Helmet Discount-2002	0.1000	Seasonal Discount	Reseller	2012-05-30 00:00:00	2012-06-29 00:00:00	0	Nor
	9	Road-650 Overstock	0.3000	Excess Inventory	Reseller	2012-05-30 00:00:00	2012-07-30 00:00:00	0	Nor
	10	Mountain Tire Sale	0.5000	Excess Inventory	Customer	2013-05-14 00:00:00	2013-07-29 00:00:00	0	Nor

Connect to ORACLE Database using sqlalchemy

Sample Database Used - <https://www.oracletutorial.com/getting-started/oracle-sample-database/>
(<https://www.oracletutorial.com/getting-started/oracle-sample-database/>)

This is how my connection looks like in SQL developer :-



```
In [14]: engine_oracle = sqlalchemy.create_engine('oracle://c##abhat:abhat@localhost:15
```

```
In [15]: print("connecting with engine " + str(engine_oracle))

connecting with engine Engine(oracle://c##abhat:***@localhost:1521/orcl)
```

```
In [16]: print(engine_oracle.table_names())

['regions', 'countries', 'locations', 'warehouses', 'employees', 'product_categories', 'products', 'customers', 'contacts', 'orders', 'order_items', 'inventories']
```

```
In [17]: con_ORA = engine_oracle.connect()
```

```
In [18]: query = "select * from CONTACTS"
df_ora = pd.read_sql_query(query, con_ORA)
df_ora
```

```
Out[18]:
```

	contact_id	first_name	last_name	email	phone	customer_id
0	208	Stephaine	Booker	stephaine.booker@tsocorp.com	+39 55 012 4559	208
1	209	Emilie	Parsons	emilie.parsons@timewarner.com	+39 10 012 4363	209
2	210	Jaleesa	Bowen	jaleesa.bowen@cstbrands.com	+66 76 012 4633	210
3	211	Jeannie	Poole	jeannie.poole@aboutmcdonalds.com	+91 80 012 4637	211
4	212	Adrienne	Lang	adrienne.lang@qualcomm.com	+39 2 012 4771	212
...
314	203	Vella	Hancock	vella.hancock@pmi.com	+39 49 012 4375	203
315	204	Retta	Martinez	retta.martinez@riteaid.com	+39 49 012 4377	204
316	205	Annelle	Lawrence	annelle.lawrence@techdata.com	+39 10 012 4379	205
317	206	Sherron	Simon	sherron.simon@ally.com	+39 10 012 4381	206
318	207	Carita	Mcintyre	carita.mcintyre@northwesternmutual.com	+86 10 012 4165	207

319 rows × 6 columns

Using cx_oracle

```
In [54]: engine_oral = sqlalchemy.create_engine('oracle+cx_oracle://c##abhat:abhat@1
```

```
In [55]: print("connecting with engine " + str(engine_oral))
```

```
connecting with engine Engine(oracle+cx_oracle://c##abhat:***@localhost:1521/orcl)
```

```
In [56]: print(engine_oral.table_names())
```

```
['regions', 'countries', 'locations', 'warehouses', 'employees', 'product_categories', 'products', 'customers', 'contacts', 'orders', 'order_items', 'inventories']
```

```
In [57]: con_ORA1 = engine_oral.connect()
```

```
In [58]: query = "select * from CONTACTS"
df_oral = pd.read_sql_query(query, con_ORA1)
df_oral
```

```
Out[58]:
```

	contact_id	first_name	last_name	email	phone	customer_id
0	208	Stephaine	Booker	stephaine.booker@tsocorp.com	+39 55 012 4559	208
1	209	Emilie	Parsons	emilie.parsons@timewarner.com	+39 10 012 4363	209
2	210	Jaleesa	Bowen	jaleesa.bowen@cstbrands.com	+66 76 012 4633	210
3	211	Jeannie	Poole	jeannie.poole@aboutmcdonalds.com	+91 80 012 4637	211
4	212	Adrienne	Lang	adrienne.lang@qualcomm.com	+39 2 012 4771	212
...
314	203	Vella	Hancock	vella.hancock@pmi.com	+39 49 012 4375	203
315	204	Retta	Martinez	retta.martinez@riteaid.com	+39 49 012 4377	204
316	205	Annelle	Lawrence	annelle.lawrence@techdata.com	+39 10 012 4379	205
317	206	Sherron	Simon	sherron.simon@ally.com	+39 10 012 4381	206
318	207	Carita	Mcintyre	carita.mcintyre@northwesternmutual.com	+86 10 012 4165	207

319 rows × 6 columns

Connect to ORACLE Database using ipython-sql

```
In [19]: %load_ext sql
          %sql "oracle://c##abhat:abhat@localhost:1521/orcl"
```

The sql extension is already loaded. To reload it, use:

```
%reload_ext sql
```

```
In [20]: %%sql
          select * from contacts where rownum < 11

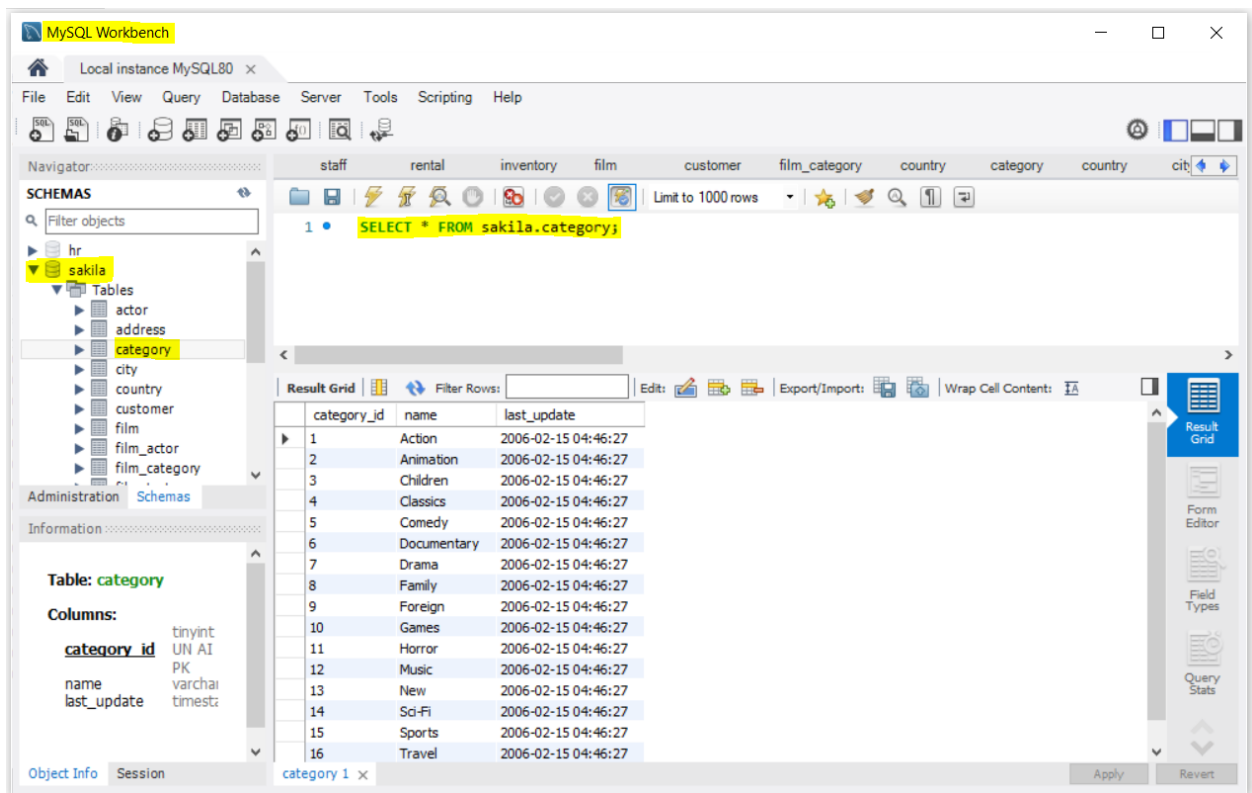
          mssql+pyodbc://abhat:***@sqldb1
          mssql+pyodbc://dstran:***@sqldb
          * oracle://c##abhat:***@localhost:1521/orcl
          0 rows affected.
```

```
Out[20]:
```

contact_id	first_name	last_name	email	phone	customer_id
208	Stephaine	Booker	stephaine.booker@tsocorp.com	+39 55 012 4559	208
209	Emilie	Parsons	emilie.parsons@timewarner.com	+39 10 012 4363	209
210	Jaleesa	Bowen	jaleesa.bowen@cstbrands.com	+66 76 012 4633	210
211	Jeannie	Poole	jeannie.poole@aboutmcdonalds.com	+91 80 012 4637	211
212	Adrienne	Lang	adrienne.lang@qualcomm.com	+39 2 012 4771	212
213	Jess	Nguyen	jess.nguyen@searsholdings.com	+39 2 012 4773	213
214	Tandy	House	tandy.house@ebay.com	+39 2 012 4775	214
215	Herman	Stokes	herman.stokes@capitalone.com	+39 49 012 4777	215
216	Keesha	Lambert	keesha.lambert@emc.com	+39 49 012 4779	216
217	Lauren	Williamson	lauren.williamson@usaa.com	+39 49 012 4781	217

Connect to MySQL Database using sqlalchemy

This is how my Schema looks like in MySQL Workbench :-



```
In [21]: engine_mysql = sqlalchemy.create_engine('mysql+pymysql://root:root@localhost:3306/sakila')
```

```
In [22]: print("connecting with engine " + str(engine_mysql))
```

```
connecting with engine Engine(mysql+pymysql://root:***@localhost:3306/sakila)
```

```
In [23]: print(engine_mysql.table_names())
```

```
['actor', 'address', 'category', 'city', 'country', 'customer', 'film', 'film_actor', 'film_category', 'film_text', 'inventory', 'language', 'payment', 'rental', 'staff', 'store']
```

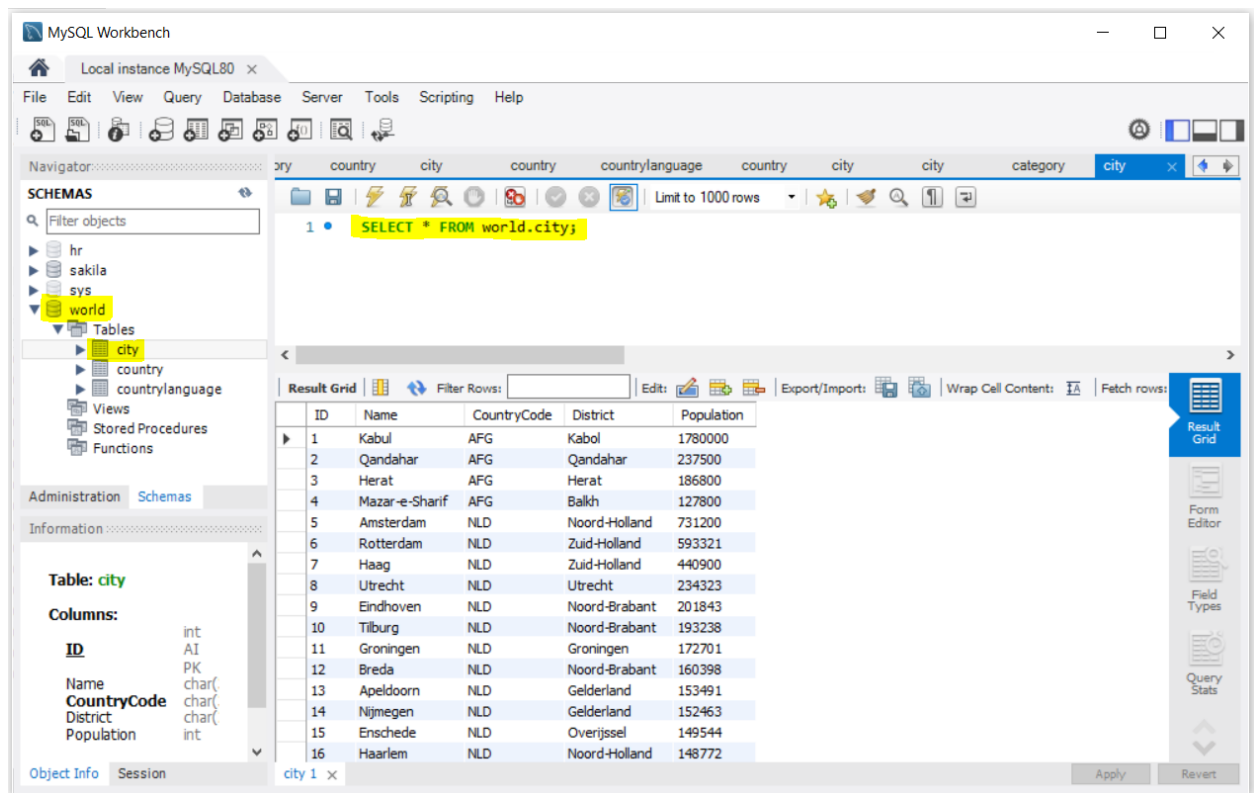
```
In [24]: con_mysql = engine_mysql.connect()
```

```
In [25]: query = "select * from category"
df_mysql = pd.read_sql_query(query, con_mysql)
df_mysql
```

```
Out[25]:
```

	category_id	name	last_update
0	1	Action	2006-02-15 04:46:27
1	2	Animation	2006-02-15 04:46:27
2	3	Children	2006-02-15 04:46:27
3	4	Classics	2006-02-15 04:46:27
4	5	Comedy	2006-02-15 04:46:27
5	6	Documentary	2006-02-15 04:46:27
6	7	Drama	2006-02-15 04:46:27
7	8	Family	2006-02-15 04:46:27
8	9	Foreign	2006-02-15 04:46:27
9	10	Games	2006-02-15 04:46:27
10	11	Horror	2006-02-15 04:46:27
11	12	Music	2006-02-15 04:46:27
12	13	New	2006-02-15 04:46:27
13	14	Sci-Fi	2006-02-15 04:46:27
14	15	Sports	2006-02-15 04:46:27
15	16	Travel	2006-02-15 04:46:27

Connect to MySQL Database using ipython-sql



```
In [1]: %load_ext sql
        %sql "mysql+pymysql://root:root@localhost:3306/world"
```

Connection info needed in SQLAlchemy format, example:

```
postgresql://username:password@hostname/dbname
or an existing connection: dict_keys([])
```

Could not parse rfc1738 URL from string 'mysql+pymysql://root:root@localhost:3306/world'

Connection info needed in SQLAlchemy format, example:

```
postgresql://username:password@hostname/dbname
or an existing connection: dict_keys([])
```

```
In [27]: %%sql
SELECT * FROM city LIMIT 0,20

mssql+pyodbc://abhat:***@sqldb1
mssql+pyodbc://dstran:***@sqldb
* mysql+pymysql://root:***@localhost:3306/world
oracle://c##abhat:***@localhost:1521/orcl
20 rows affected.
```

```
Out[27]:
```

ID	Name	CountryCode	District	Population
1	Kabul	AFG	Kabol	1780000
2	Qandahar	AFG	Qandahar	237500
3	Herat	AFG	Herat	186800
4	Mazar-e-Sharif	AFG	Balkh	127800
5	Amsterdam	NLD	Noord-Holland	731200
6	Rotterdam	NLD	Zuid-Holland	593321
7	Haag	NLD	Zuid-Holland	440900
8	Utrecht	NLD	Utrecht	234323
9	Eindhoven	NLD	Noord-Brabant	201843
10	Tilburg	NLD	Noord-Brabant	193238
11	Groningen	NLD	Groningen	172701
12	Breda	NLD	Noord-Brabant	160398
13	Apeldoorn	NLD	Gelderland	153491
14	Nijmegen	NLD	Gelderland	152463
15	Enschede	NLD	Overijssel	149544
16	Haarlem	NLD	Noord-Holland	148772
17	Almere	NLD	Flevoland	142465
18	Arnhem	NLD	Gelderland	138020
19	Zaanstad	NLD	Noord-Holland	135621
20	Â's-Hertogenbosch	NLD	Noord-Brabant	129170

Connect to PostgreSQL Database using sqlalchemy

PgAdmin Console

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Servers' tree is expanded to show the 'Pagila' database under 'PostgreSQL 12'. The 'Schemas' folder is selected, showing 'postgres' and 'sportsdb'. On the right, the 'Query Editor' is active, showing a SQL query: `SELECT * FROM public.category ORDER BY category_id ASC`. Below the query editor, the 'Data Output' tab displays the results of the query in a table format.

category_id [PK] integer	name character varying (25)	last_update timestamp without time zone
2	Animation	2006-02-15 09:46:27
3	Children	2006-02-15 09:46:27
4	Classics	2006-02-15 09:46:27
5	Comedy	2006-02-15 09:46:27
6	Documentary	2006-02-15 09:46:27
7	Drama	2006-02-15 09:46:27
8	Family	2006-02-15 09:46:27
9	Foreign	2006-02-15 09:46:27
10	Games	2006-02-15 09:46:27
11	Horror	2006-02-15 09:46:27
12	Music	2006-02-15 09:46:27
13	New	2006-02-15 09:46:27
14	Sci-Fi	2006-02-15 09:46:27
15	Sports	2006-02-15 09:46:27
16	Travel	2006-02-15 09:46:27

```
In [29]: engine_postgresql = sqlalchemy.create_engine('postgresql://postgres:postgre
```

```
In [30]: print("connecting with engine " + str(engine_postgresql))
```

```
connecting with engine Engine(postgresql://postgres:***@localhost:5432/Pa
gila)
```

```
In [31]: print(engine_postgresql.table_names())
```

```
['actor', 'film', 'payment_p2007_02', 'payment_p2007_03', 'payment_p2007_
04', 'payment_p2007_05', 'payment_p2007_06', 'payment_p2007_01', 'addres
s', 'category', 'city', 'country', 'customer', 'film_actor', 'film_catego
ry', 'inventory', 'language', 'rental', 'staff', 'store', 'payment']
```

```
In [32]: con_postgresql = engine_postgresql.connect()
```

```
In [33]: query = "select * from category"
df_postgresql = pd.read_sql_query(query, con_postgresql)
df_postgresql
```

```
Out[33]:
```

	category_id	name	last_update
0	1	Action	2006-02-15 09:46:27
1	2	Animation	2006-02-15 09:46:27
2	3	Children	2006-02-15 09:46:27
3	4	Classics	2006-02-15 09:46:27
4	5	Comedy	2006-02-15 09:46:27
5	6	Documentary	2006-02-15 09:46:27
6	7	Drama	2006-02-15 09:46:27
7	8	Family	2006-02-15 09:46:27
8	9	Foreign	2006-02-15 09:46:27
9	10	Games	2006-02-15 09:46:27
10	11	Horror	2006-02-15 09:46:27
11	12	Music	2006-02-15 09:46:27
12	13	New	2006-02-15 09:46:27
13	14	Sci-Fi	2006-02-15 09:46:27
14	15	Sports	2006-02-15 09:46:27
15	16	Travel	2006-02-15 09:46:27

Connect to PostgreSQL Using psycopg2

pgAdmin Console

The screenshot shows the pgAdmin interface. On the left, the 'Servers' tree is expanded to show 'PostgreSQL 12' > 'Databases (4)' > 'NorthWind'. Under 'NorthWind', the 'Schemas (1)' section is expanded to show the 'public' schema, which contains 'Tables (14)'. The 'categories' table is highlighted. On the right, the 'Query Editor' shows a SQL query: `SELECT * FROM public.categories ORDER BY category_id ASC LIMIT 100`. Below the query editor, the 'Data Output' tab displays the results of the query in a table format.

	category_id [PK] smallint	category_name character varying (15)	description text	picture bytea
1	1	Beverages	Soft drinks, coff...	[binary data]
2	2	Condiments	Sweet and savo...	[binary data]
3	3	Confections	Desserts, candi...	[binary data]
4	4	Dairy Products	Cheeses	[binary data]
5	5	Grains/Cereals	Breads, cracker...	[binary data]
6	6	Meat/Poultry	Prepared meats	[binary data]
7	7	Produce	Dried fruit and b...	[binary data]
8	8	Seafood	Seaweed and fish	[binary data]

```
In [39]: engine_postgresql1 = sqlalchemy.create_engine('postgresql+psycopg2://postgres:***@localhost:5432/NorthWind')
```

```
In [40]: print("connecting with engine " + str(engine_postgresql1))
```

```
connecting with engine Engine(postgresql+psycopg2://postgres:***@localhost:5432/NorthWind)
```

```
In [41]: print(engine_postgresql1.table_names())
```

```
['us_states', 'customers', 'orders', 'employees', 'shippers', 'products', 'order_details', 'categories', 'suppliers', 'region', 'territories', 'employee_territories', 'customer_demographics', 'customer_customer_demo']
```

```
In [42]: con_postgresql1 = engine_postgresql1.connect()
```

```
In [45]: query = "select * from categories"
df_postgresql1 = pd.read_sql_query(query, con_postgresql1)
df_postgresql1
```

```
Out[45]:
```

	category_id	category_name	description	picture
0	1	Beverages	Soft drinks, coffees, teas, beers, and ales	
1	2	Condiments	Sweet and savory sauces, relishes, spreads, an...	
2	3	Confections	Desserts, candies, and sweet breads	
3	4	Dairy Products	Cheeses	
4	5	Grains/Cereals	Breads, crackers, pasta, and cereal	
5	6	Meat/Poultry	Prepared meats	
6	7	Produce	Dried fruit and bean curd	
7	8	Seafood	Seaweed and fish	

Connect to PostgreSQL Database using ipython-sql

```
In [48]: %load_ext sql
%sql "postgresql://postgres:postgres@localhost:5432/Pagila"
```

The sql extension is already loaded. To reload it, use:

```
%reload_ext sql
```



```
In [49]: %%sql
SELECT * FROM category
```

```
mssql+pyodbc://abhat:***@sqldb1
mssql+pyodbc://dstran:***@sqldb
mysql+pymysql://root:***@localhost:3306/world
oracle://c##abhat:***@localhost:1521/orcl
* postgresql://postgres:***@localhost:5432/Pagila
16 rows affected.
```

```
Out[49]:
```

	category_id	name	last_update
	1	Action	2006-02-15 09:46:27
	2	Animation	2006-02-15 09:46:27
	3	Children	2006-02-15 09:46:27
	4	Classics	2006-02-15 09:46:27
	5	Comedy	2006-02-15 09:46:27
	6	Documentary	2006-02-15 09:46:27
	7	Drama	2006-02-15 09:46:27
	8	Family	2006-02-15 09:46:27
	9	Foreign	2006-02-15 09:46:27
	10	Games	2006-02-15 09:46:27
	11	Horror	2006-02-15 09:46:27
	12	Music	2006-02-15 09:46:27
	13	New	2006-02-15 09:46:27
	14	Sci-Fi	2006-02-15 09:46:27
	15	Sports	2006-02-15 09:46:27
	16	Travel	2006-02-15 09:46:27

```
In [50]: %load_ext sql
%sql "postgresql+psycpg2://postgres:postgres@localhost:5432/NorthWind"
```

```
The sql extension is already loaded. To reload it, use:
%reload_ext sql
```

```
In [52]: %%sql
SELECT category_name , description FROM categories

mssql+pyodbc://abhat:***@sqldb1
mssql+pyodbc://dstran:***@sqldb
mysql+pymysql://root:***@localhost:3306/world
oracle://c##abhat:***@localhost:1521/orcl
* postgresql+psycpg2://postgres:***@localhost:5432/NorthWind
postgresql://postgres:***@localhost:5432/Pagila
8 rows affected.
```

```
Out[52]:
```

category_name	description
Beverages	Soft drinks, coffees, teas, beers, and ales
Condiments	Sweet and savory sauces, relishes, spreads, and seasonings
Confections	Desserts, candies, and sweet breads
Dairy Products	Cheeses
Grains/Cereals	Breads, crackers, pasta, and cereal
Meat/Poultry	Prepared meats
Produce	Dried fruit and bean curd
Seafood	Seaweed and fish

END