Assignment 13

Q1. Create an sqlalchemy engine using a sample from the data set

Answer:

import sqlalchemy

import numpy as np

from sqlalchemy import Table, Column, Integer, String, MetaData, ForeignKey

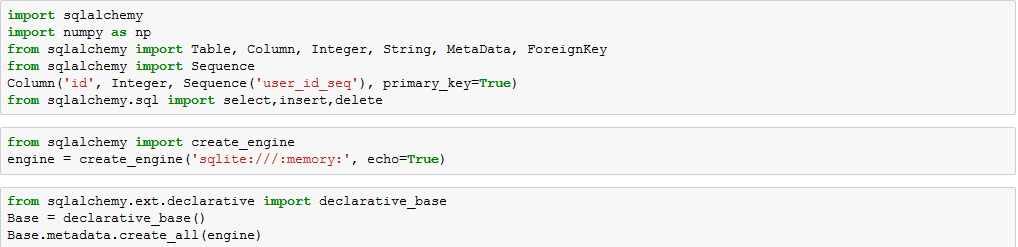
from sqlalchemy import Sequence

Column('id', Integer, Sequence('user\_id\_seq'), primary\_key=True)

from sqlalchemy.sql import select,insert,delete

from sqlalchemy import create\_engine

engine = create\_engine('sqlite:///:memory:', echo=True)



from sqlalchemy.ext.declarative import declarative\_base

Base = declarative\_base()

Base.metadata.create\_all(engine)

class Mydb(Base):

\_\_tablename\_\_ = 'mydb'

id=Column('id',Integer,Sequence('user\_id\_seq'), primary\_key=True)

age=Column(Integer)

workclass=Column(String)

education=Column(String)

marital\_status=Column(String)

occupation=Column(String)

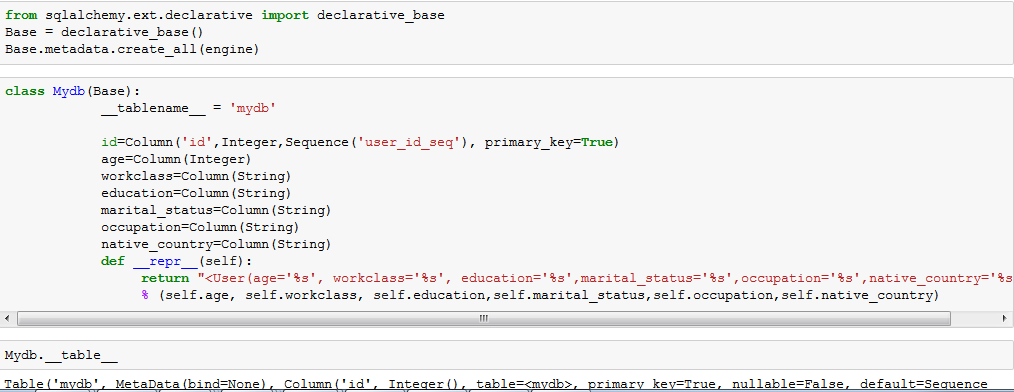
native\_country=Column(String)

def \_\_repr\_\_(self):

return "<User(age='%s', workclass='%s', education='%s',marital\_status='%s',occupation='%s',native\_country='%s')>"

% (self.age, self.workclass, self.education,self.marital\_status,self.occupation,self.native\_country)

Mydb.\_\_table\_\_



Base.metadata.create\_all(engine)

import numpy as np

import pandas as pd

from pandas import DataFrame, Series

import sqlite3 as db

from pandasql import sqldf

pysqldf = lambda q: sqldf(q, globals())

sqladb=pd.read\_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data',names=['Age','workclass','fnlwgt','education','education-num','marital-status','occupation','relationship','race','sex','capital-gain','capital-loss','hours-per-week','native-country','>50K, <=50K'])

conn = engine.connect()

from sqlalchemy.orm import sessionmaker

Session = sessionmaker(bind=engine)

Session = sessionmaker()

Session.configure(bind=engine)

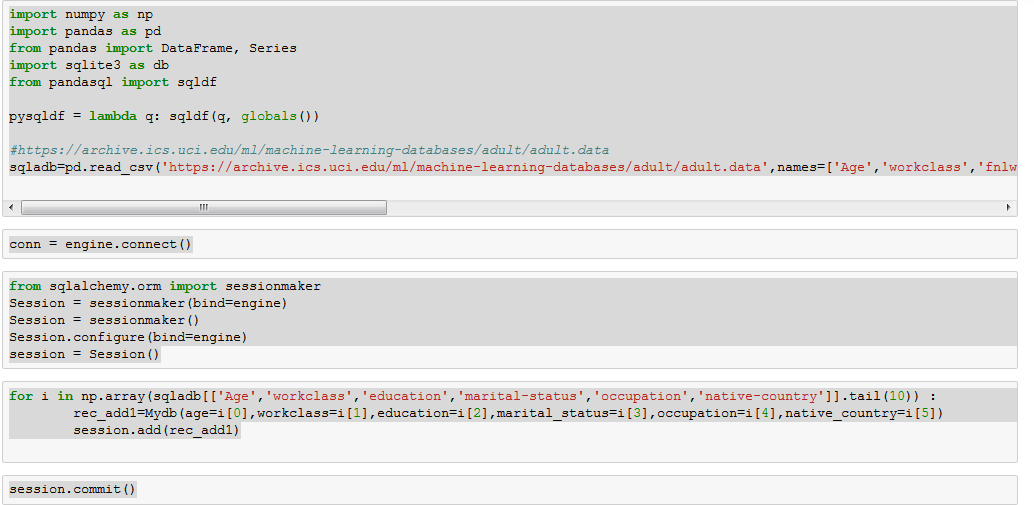
session = Session()

for i in np.array(sqladb[['Age','workclass','education','marital-status','occupation','native-country']].tail(10)) :

rec\_add1=Mydb(age=i[0],workclass=i[1],education=i[2],marital\_status=i[3],occupation=i[4],native\_country=i[5])

session.add(rec\_add1)

session.commit()



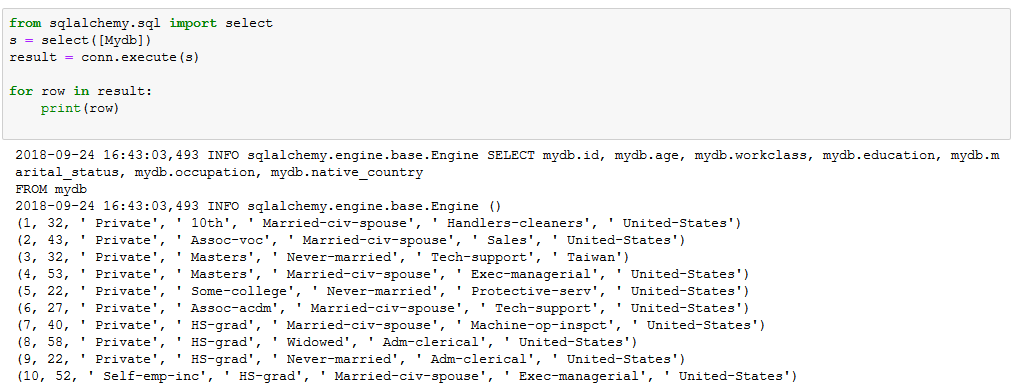
from sqlalchemy.sql import select

s = select([Mydb])

result = conn.execute(s)

for row in result:

print(row)



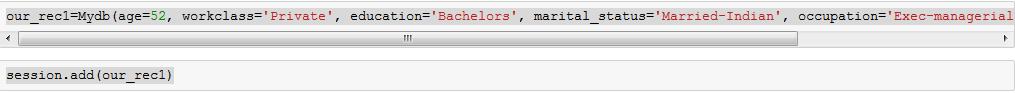
2. Write two basic update queries

Answer :

## Adding Record###

our\_rec1=Mydb(age=52, workclass='Private', education='Bachelors', marital\_status='Married-Indian', occupation='Exec-managerial', native\_country='India')

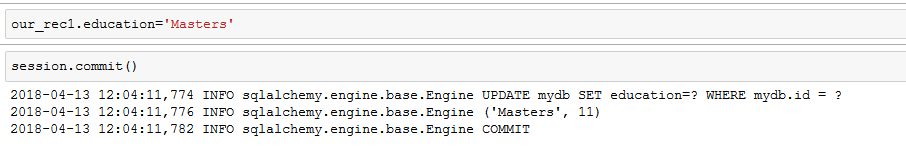
session.add(our\_rec1)



**##1: Updating – column - education###**

our\_rec1.education='Masters'

session.commit()



Or (another way of updating)

from sqlalchemy import update

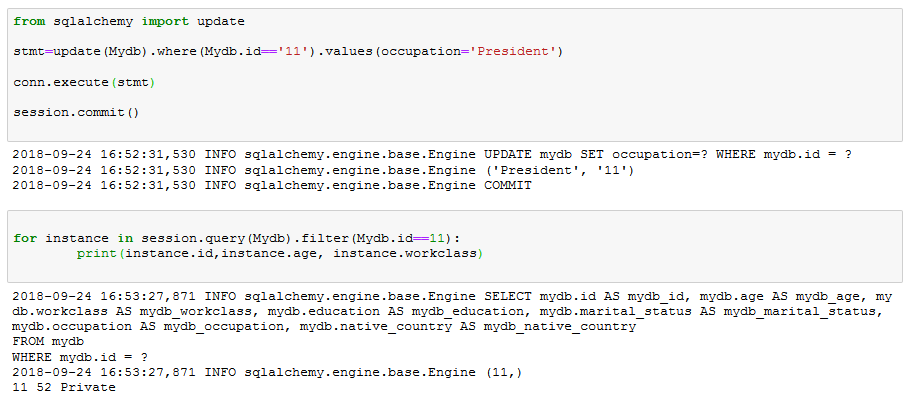
stmt=update(Mydb).where(Mydb.id=='11').values(occupation='President')

conn.execute(stmt)

session.commit()

for instance in session.query(Mydb).filter(Mydb.id==11):

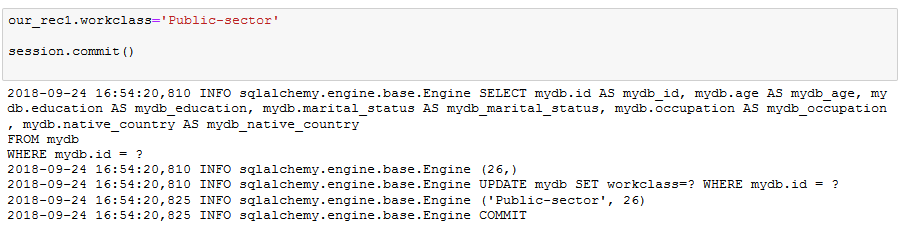
print(instance.id,instance.age, instance.workclass)



**##2 Updating – column – workclass###**

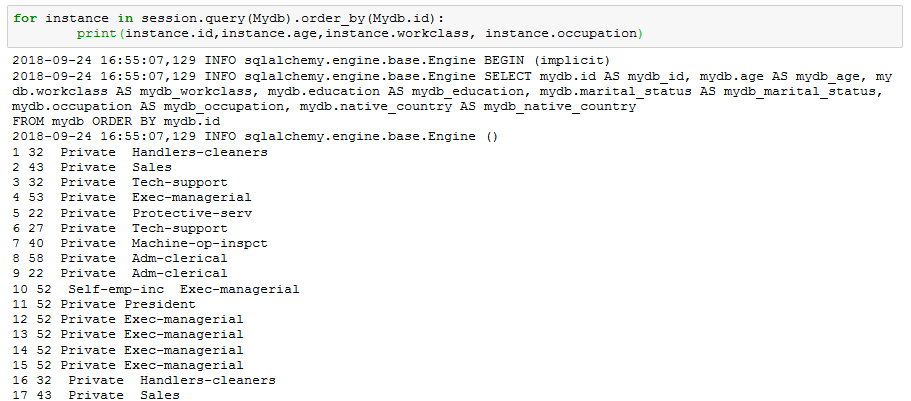
our\_rec1.workclass='Public-sector'

session.commit()



for instance in session.query(Mydb).order\_by(Mydb.id):

print(instance.id,instance.age,instance.workclass, instance.occupation)



Or (another way of updating)

from sqlalchemy import update

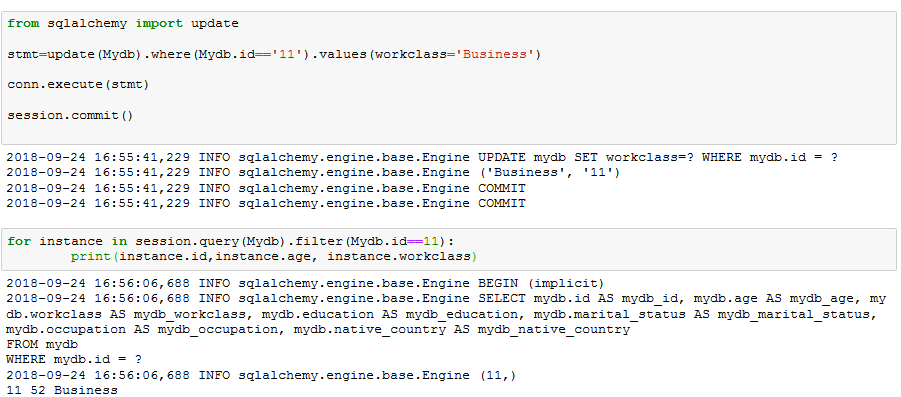
stmt=update(Mydb).where(Mydb.id=='11').values(workclass='Business')

conn.execute(stmt)

session.commit()

for instance in session.query(Mydb).filter(Mydb.id==11):

print(instance.id,instance.age, instance.workclass)



3. Write two delete queries

Answer :

#1###Delete based on id#####

from sqlalchemy import delete

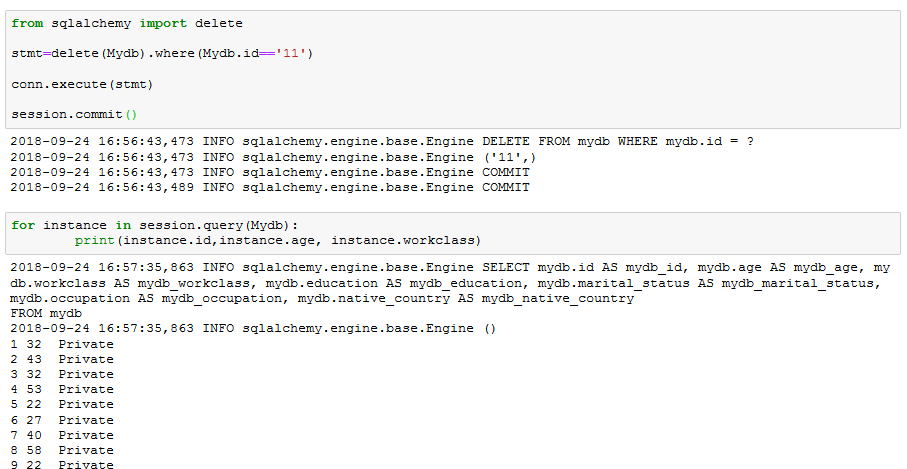
stmt=delete(Mydb).where(Mydb.id=='11')

conn.execute(stmt)

session.commit()

for instance in session.query(Mydb):

print(instance.id,instance.age, instance.workclass)



#2###Delete based on workclass= ‘ Private’#####

from sqlalchemy import delete

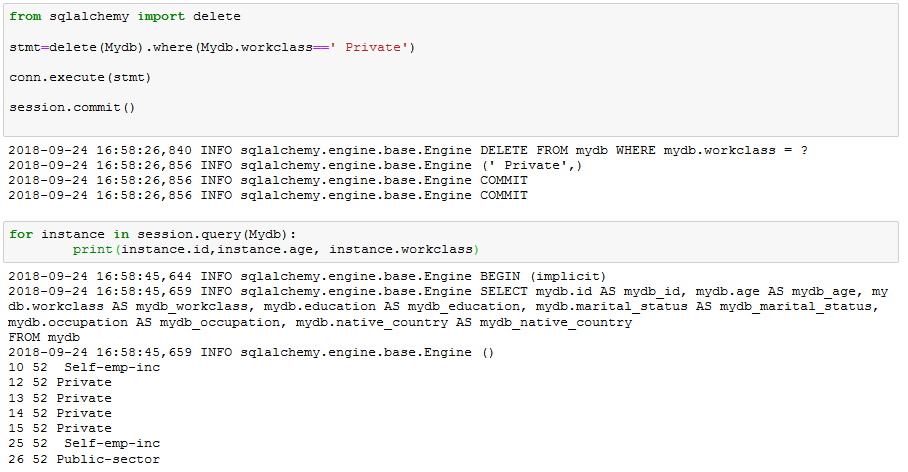
stmt=delete(Mydb).where(Mydb.workclass==' Private')

conn.execute(stmt)

session.commit()

for instance in session.query(Mydb):

print(instance.id,instance.age, instance.workclass)



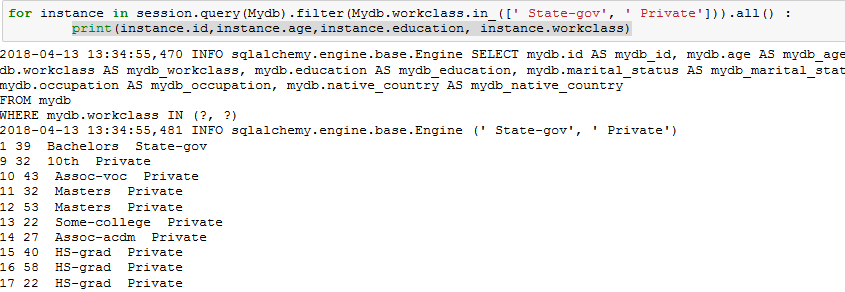
4. Write two filter queries

Answer :

Filter query1 :

for instance in session.query(Mydb).filter(Mydb.workclass.in\_([' State-gov', ' Private'])).all() :

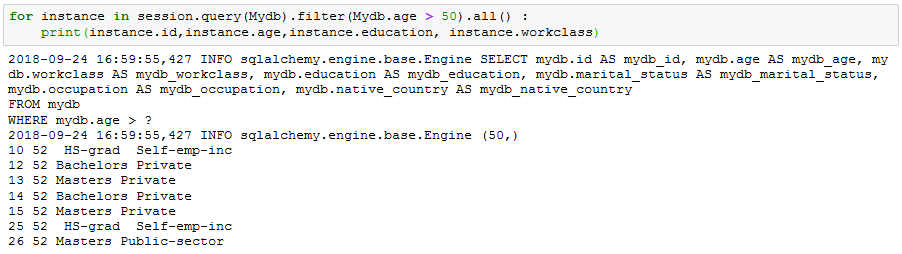
print(instance.id,instance.age,instance.education, instance.workclass)



Filter query2 :

for instance in session.query(Mydb).filter(Mydb.age > 50).all() :

print(instance.id,instance.age,instance.education, instance.workclass)



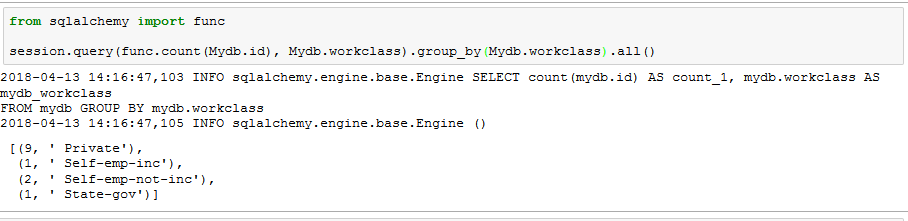
5. Write two function queries

Answer:

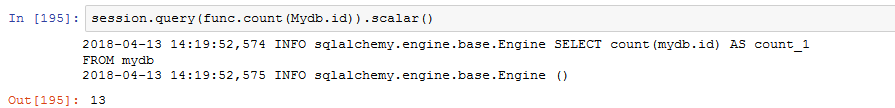
##Function query 1##

from sqlalchemy import func

session.query(func.count(Mydb.id), Mydb.workclass).group\_by(Mydb.workclass).all()



session.query(func.count(Mydb.id)).scalar()



##Function query 2##

from sqlalchemy import func

session.query(func.avg(Mydb.age), Mydb.workclass).group\_by(Mydb.workclass).all()

