Chand Mohammad

Youtube links

https://youtu.be/jaKMm9njcJc

https://youtu.be/dIV8NHzboxU

https://youtu.be/T6Q6bsv05To

https://youtu.be/9xkfCo5StAk

https://youtu.be/RKYiBmA9Mbk

https://youtu.be/mLme6rJqkMI

Sqlalchemy basic questions and answers

https://climbtheladder.com/sqlalchemy-interview-questions/

PROGRAMS

Create table

```
# Create engine
# create session
# create table
# migrate
#session: for manage transaction
from sqlalchemy import create_engine,Column,Integer,String
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base
engine=create_engine('mysql+mysqldb://root:root@localhost:3306/edureka',echo=True)
Session=sessionmaker(bind=engine)
session=Session()
Base=declarative_base()
```

```
class employee(Base):
    __tablename__='employee'
    id=Column(Integer,primary_key=True)
    name=Column(String(50))
    age=Column(Integer)
    salary=Column(Integer)
    grade=Column(String(50))

Base.metadata.create_all(engine)
```

Delete records

```
# Steps:-
# Get record
# Delete record
# Commit Record


from sqlalchemy import create_engine,Column,Integer,String
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base
engine=create_engine('mysql+mysqldb://root:root@localhost:3306/edureka',echo=True)
Session=sessionmaker(bind=engine)
session=Session()
Base=declarative_base()

class employee(Base):
    __tablename__='employee'
    id=Column(Integer,primary_key=True)
    name=Column(String(50))
    age=Column(Integer)
    salary=Column(Integer)
    grade=Column(String(50))

Base.metadata.create_all(engine)


employees=session.query(employee).filter(employee.name=='raza').first()
session.delete(employees)
session.commit()
# print(employees.name)
```

Insert data

```
# create instance of table
# add data to sesssion
# Commit changes to database


from sqlalchemy import create_engine,Column,Integer,String
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base
engine=create_engine('mysql+mysqldb://root:root@localhost:3306/edureka',echo=True)
Session=sessionmaker(bind=engine)
session=Session()
Base=declarative_base()
```

```python
class employee(Base):
    __tablename__='employee'
    id=Column(Integer,primary_key=True)
    name=Column(String(50))
    age=Column(Integer)
    salary=Column(Integer)
    grade=Column(String(50))

Base.metadata.create_all(engine)

# employee1=employee(name='chand',age=24,salary=10000,grade='fifth')
# session.add(employee1)

employee2=employee(name='saif',age=25,salary=11000,grade='fourth')
employee3=employee(name='salman',age=26,salary=12000,grade='sixth')
employee4=employee(name='uviash',age=27,salary=13000,grade='seventh')
session.add_all([employee2,employee3,employee4])
session.commit()
```

Read Data

```python
# Query table:-
# Get all data
# Get data in order
# Get data by filtering
# Count of results

from sqlalchemy import create_engine,Column,Integer,String,or_
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base
engine=create_engine('mysql+mysqldb://root:root@localhost:3306/edureka',echo=True)
Session=sessionmaker(bind=engine)
session=Session()
Base=declarative_base()

class employee(Base):
    __tablename__='employee'
    id=Column(Integer,primary_key=True)
    name=Column(String(50))
    age=Column(Integer)
    salary=Column(Integer)
    grade=Column(String(50))

Base.metadata.create_all(engine)


# Get all data
employees=session.query(employee)
for employee in employees:
    print(employee.id, employee.name,employee.age,employee.salary,employee.grade)


# # Get data in order
# employees=session.query(employee).order_by(employee.age)
# for employee in employees:
#     print(employee.age)
#     # print(employee.age,employee.salary,employee.name)


# # Get data by filtering
# employees=session.query(employee).filter(employee.name=="saif").all()
# print(employees.name,employees.age,employees.salary,employees.grade)

#.......................................................

# employees=session.query(employee).filter(or_ (employee.name=="saif",employee.name=="chand"))
# for employee in employees:
#     print(employee.name,employee.age,employee.salary,employee.grade)



# # Count of results
# employees_cnt=session.query(employee).filter(employee.name=="saif").count()
# print(employees_cnt)
```

Update records

```
# Get record
# Change value
# Commit change

from sqlalchemy import create_engine,Column,Integer,String
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base
engine=create_engine('mysql+mysqldb://root:root@localhost:3306/edureka',echo=True)
Session=sessionmaker(bind=engine)
session=Session()
Base=declarative_base()

class employee(Base):
    __tablename__='employee'
    id=Column(Integer,primary_key=True)
    name=Column(String(50))
    age=Column(Integer)
    salary=Column(Integer)
    grade=Column(String(50))

Base.metadata.create_all(engine)

employees=session.query(employee).filter(employee.name=='raza').first()
employees.name='saif'
session.commit()
# print(employees.name)
```

Using Foreignkey

```
from sqlalchemy import create_engine, ForeignKey, Column, Integer, String
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import relationship

engine = create_engine('mysql+mysqldb://root:root@localhost:3306/edureka', echo=True)
Base = declarative_base()

class Customer(Base):
    __tablename__ = 'customers'
    id = Column(Integer, primary_key=True)
    name = Column(String(255))  # Specify a length for the VARCHAR column
    address = Column(String(100))
    email = Column(String(100))
    invoices = relationship("Invoice", back_populates="customer")

class Invoice(Base):
    __tablename__ = 'invoices'
    id = Column(Integer, primary_key=True)
    custid = Column(Integer, ForeignKey('customers.id'))
    invno = Column(Integer)
    amount = Column(Integer)
    customer = relationship("Customer", back_populates="invoices")

Base.metadata.create_all(engine)


from sqlalchemy.orm import sessionmaker
Session = sessionmaker(bind = engine)
session = Session()


# c1 = Customer(name = "Gopal Krishna", address = "Bank Street Hydarebad", email = "gk@gmail.com")
# c1.invoices = [Invoice(invno = 10, amount = 15000), Invoice(invno = 14, amount = 3850)]

# c2 = [
#    Customer(
#       name = "Govind Pant",
#       address = "Gulmandi Aurangabad",
#       email = "gpant@gmail.com",
#       invoices = [Invoice(invno = 3, amount = 10000),
#       Invoice(invno = 4, amount = 5000)]
#    )
# ]

rows = [
   Customer(
      name = "Govind Kala",
      address = "Gulmandi Aurangabad",
      email = "kala@gmail.com",
      invoices = [Invoice(invno = 7, amount = 12000), Invoice(invno = 8, amount = 18500)]),

   Customer(
      name = "Abdul Rahman",
      address = "Rohtak",
      email = "abdulr@gmail.com",
      invoices = [Invoice(invno = 9, amount = 15000),
      Invoice(invno = 11, amount = 6000)
   ])
]

session.add_all(rows)
session.commit()
```

Insert records_two table

```
from sqlalchemy import Column, Integer, String, ForeignKey, Float
from sqlalchemy import create_engine
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship

# Create a MySQL database engine
engine = create_engine('mysql+mysqldb://root:root@localhost:3306/edureka', echo=True)

Base = declarative_base()

class Customer(Base):
    __tablename__ = 'customers'
    id = Column(Integer, primary_key=True)
    name = Column(String)
    address = Column(String)
    email = Column(String)

    invoices = relationship('Invoice', back_populates='customer')

class Invoice(Base):
    __tablename__ = 'invoices'

    id = Column(Integer, primary_key=True)
    custid = Column(Integer, ForeignKey('customers.id'))  # Corrected indentation here
    invno = Column(Integer)  # Corrected indentation here
    amount = Column(Integer)
    customer = relationship("Customer", back_populates="invoices")

# Create a session to interact with the database
Session = sessionmaker(bind=engine)
session = Session()

# s = session.query(Customer).filter(Customer.invoices.any(Invoice.invno == 1))
# session.add_all(s)
# session.commit()



# Create customer records
customer1 = Customer(name='John Doe', address='123 Main St', email='john@example.com')
customer2 = Customer(name='Jane Smith', address='456 Elm St', email='jane@example.com')

# Add customers to the session
session.add_all([customer1, customer2])
session.commit()

# Create invoice records
invoice1 = Invoice(amount=100.0, customer=customer1)
invoice2 = Invoice(amount=150.0, customer=customer2)

# Add invoices to the session
session.add_all([invoice1, invoice2])
session.commit()
```

Using Joins

```
from sqlalchemy import create_engine, ForeignKey, Column, Integer, String
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import relationship

engine = create_engine('mysql+mysqldb://root:root@localhost:3306/edureka', echo=True)
Base = declarative_base()

class Customer(Base):
    __tablename__ = 'customers'
    id = Column(Integer, primary_key=True)
    name = Column(String(255))  # Specify a length for the VARCHAR column
    address = Column(String(100))
    email = Column(String(100))
    invoices = relationship("Invoice", back_populates="customer")

class Invoice(Base):
    __tablename__ = 'invoices'
    id = Column(Integer, primary_key=True)
    custid = Column(Integer, ForeignKey('customers.id'))
    invno = Column(Integer)
    amount = Column(Integer)
    customer = relationship("Customer", back_populates="invoices")

Base.metadata.create_all(engine)


from sqlalchemy.orm import sessionmaker
Session = sessionmaker(bind = engine)
session = Session()


# result = session.query(Customer).join(Invoice).filter(Invoice.amount == 18500)
# for customer in result:  # Use 'customer' here
#     for inv in customer.invoices:  # Use 'customer.invoices' here
#         print(customer.id, customer.name, inv.invno, inv.amount)

# session.close()
from sqlalchemy.sql import select
from sqlalchemy import func

stmt = select([Invoice.custid, func.count().label('invno')]).group_by(Invoice.custid).subquery()

for u, count in session.query(Customer, stmt.c.invno).outerjoin(stmt, Customer.id ==
stmt.c.custid).order_by(Customer.id):
    print(u.name, count)




# for c, i in session.query(Customer, Invoice).filter(Customer.id == Invoice.custid).all():
#    print ("ID: {} Name: {} Invoice No: {} Amount: {}".format(c.id,c.name, i.invno, i.amount))
```

Using Operator

```
from sqlalchemy import create_engine, ForeignKey, Column, Integer, String
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import relationship

engine = create_engine('mysql+mysqldb://root:root@localhost:3306/edureka', echo=True)
Base = declarative_base()

class Customer(Base):
    __tablename__ = 'customers'
    id = Column(Integer, primary_key=True)
    name = Column(String(255))  # Specify a length for the VARCHAR column
    address = Column(String(100))
    email = Column(String(100))
    invoices = relationship("Invoice", back_populates="customer")
```

```
class Invoice(Base):
    __tablename__ = 'invoices'
    id = Column(Integer, primary_key=True)
    custid = Column(Integer, ForeignKey('customers.id'))
    invno = Column(Integer)
    amount = Column(Integer)
    customer = relationship("Customer", back_populates="invoices")

Base.metadata.create_all(engine)


from sqlalchemy.orm import sessionmaker
Session = sessionmaker(bind = engine)
session = Session()

    #Using Operator
# s = session.query(Customer).filter(Invoice.invno.__eq__(8))
# s = session.query(Customer).filter(Invoice.custid.__ne__(2))
# s = session.query(Customer).filter(Invoice.invno.in_([3, 4, 5]))  # Use .in_() for a list of values
# s = session.query(Customer).filter(Customer.invoices.any(Invoice.invno==11))

s = session.query(Invoice).filter(Invoice.customer.has(name='Govind Pant'))
for invoice in s:
    print(invoice.customer.id, invoice.customer.name, invoice.invno, invoice.amount)




# for customer in s:
#     for invoice in customer.invoices:
#         print(customer.id, customer.name, invoice.invno, invoice.amount)
```

Using filter() function

```
from sqlalchemy import Column, Integer, String
from sqlalchemy import create_engine
engine = create_engine('mysql+mysqldb://root:root@localhost:3306/edureka', echo = True)
from sqlalchemy.ext.declarative import declarative_base
Base = declarative_base()

class Customers(Base):
   __tablename__ = 'customers'

   id = Column(Integer, primary_key = True)
   name = Column(String)

   address = Column(String)
   email = Column(String)

from sqlalchemy.orm import sessionmaker
Session = sessionmaker(bind = engine)

session = Session()
result = session.query(Customers).filter(Customers.id==3)
# from sqlalchemy import text
# for cust in session.query(Customers).filter(text("id<3")):
#    print(cust.name)
# result = session.query(Customers).filter(Customers.name.like('c%'))

for row in result:
   print ("ID:", row.id, "Name: ",row.name, "Address:",row.address, "Email:",row.email)
```

Using Query() Function

```
#Output
# Name:  Chand Address: Gyaspur Email: chandbspr@gmail.com
# Name:  Komal Pande Address: Koti, Hyderabad Email: komal@gmail.com
# Name:  Rajender Nath Address: Sector 40, Gurgaon Email: nath@gmail.com
# Name:  S.M.Krishna Address: Budhwar Peth, Pune Email: smk@gmail.com

from sqlalchemy import Column, Integer, String
from sqlalchemy import create_engine
engine = create_engine('mysql+mysqldb://root:root@localhost:3306/edureka', echo = True)
from sqlalchemy.ext.declarative import declarative_base
Base = declarative_base()

class Customers(Base):
   __tablename__ = 'customers'
   id = Column(Integer, primary_key =  True)
   name = Column(String)

   address = Column(String)
   email = Column(String)


from sqlalchemy.orm import sessionmaker
Session = sessionmaker(bind = engine)
session = Session()
result = session.query(Customers).all()

for row in result:
   print ("Name: ",row.name, "Address:",row.address, "Email:",row.email)
```

Conjuctions

```
from sqlalchemy import create_engine, MetaData, Table, Column, Integer, String, and_
from sqlalchemy.sql import select

# Create the engine
engine = create_engine('mysql+mysqldb://root:root@localhost:3306/edureka', echo=True)

# Define the metadata
meta = MetaData()

# Define the 'students' table
students = Table(
    'students', meta,
    Column('id', Integer, primary_key=True),
    Column('name', String),
    Column('lastname', String),
)

# Create the table in the database
meta.create_all(engine)

# # Insert sample data into the 'students' table
conn = engine.connect()
# conn.execute(students.insert().values(name='chand', lastname='smith'))
# conn.execute(students.insert().values(name='john', lastname='doe'))
# Construct the SELECT statement with a WHERE clause using the select() method
# stmt = select([students]).where(and_(students.c.name == 'chand', students.c.id < 2))


# # Query the database using advanced conjunctions
# stmt = students.select().where(
#     and_(

#             students.c.id <= 10,
#             students.c.name == 'chand'

#       )
# )


# stmt = select([students]).where(or_(students.c.name == 'chand', students.c.id < 3))
from sqlalchemy import asc
from sqlalchemy import between
stmt = select([students]).where(between(students.c.id,2,3))
print (stmt)

# stmt = select([students]).order_by(asc(students.c.name))

# # Execute the query and fetch all results
# result = conn.execute(stmt)
# rows = result.fetchall()

# # Print the fetched rows
# for row in rows:
#     print(row)
```

Multiple table row delete

```
from sqlalchemy import create_engine, Column, Integer, String, MetaData, ForeignKey
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship

# Step 1: Define the models for the tables
Base = declarative_base()

class Student(Base):
    __tablename__ = 'student'
    id = Column(Integer, primary_key=True)
    name = Column(String(100))
    lastname = Column(String(100))
    # Add other columns as needed

class addresses(Base):
    __tablename__ = 'addresses'
    id = Column(Integer, primary_key=True)
    st_id = Column(Integer, ForeignKey('student.id'))
    postal_add = Column(String(100))
    email_add = Column(String(100))
    # Add other columns as needed

# Step 2: Create a session to interact with the database
engine = create_engine('mysql+mysqldb://root:root@localhost:3306/edureka')  # Replace with your database connection
string
Session = sessionmaker(bind=engine)
session = Session()
```

```python
# Step 3: Perform the delete operation on multiple tables using transactions
try:
    # Start a transaction
    session.begin()

    # Step 4: Use the delete statement to delete from the "grades" table based on a condition
    session.query(addresses).filter(addresses.postal_add == 'gyaspur').delete()

    # Step 5: Use the delete statement to delete from the "students" table based on a condition
    session.query(Student).filter(Student.lastname == 'Doe').delete()

    # Step 6: Commit the transaction
    session.commit()
    print("Deletion successful!")
except Exception as e:
    # Rollback the transaction in case of an error
    session.rollback()
    print("Error occurred. Transaction rolled back.")
    print(str(e))

# Close the session after you are done with the deletion
session.close()
```