#%%

from typing import List

from typing import Optional

from sqlalchemy import ForeignKey

from sqlalchemy import String, Integer, Date

from sqlalchemy.orm import DeclarativeBase

from sqlalchemy.orm import Mapped

from sqlalchemy.orm import mapped\_column

from sqlalchemy.orm import relationship

from sqlalchemy import create\_engine

from sqlalchemy.orm import Session

from sqlalchemy import select

#%%

#DB Connection: create\_engine(DBMS\_name+driver://<username>:<password>@<hostname>/<database\_name>)

engine = create\_engine("postgresql+psycopg2://postgres:@localhost/")

#%%

#Define Classes/Tables

class Base(DeclarativeBase):

pass

# Base.metadata.drop\_all(bind=engine, tables=[Patient.\_\_table\_\_, Sample.\_\_table\_\_,Result.\_\_table\_\_,Company.\_\_table\_\_])

class Patient(Base):

\_\_tablename\_\_ = "Patient"

pid: Mapped[str] = mapped\_column(String(50), primary\_key=True)

pFirst: Mapped[str] = mapped\_column(String(50))

pLast: Mapped[str] = mapped\_column(String(50))

pDOB: Mapped[str] = mapped\_column(Date)

pAddress: Mapped[str] = mapped\_column(String(50))

pEmail: Mapped[str] = mapped\_column(String(50))

pPhone: Mapped[str] = mapped\_column(String(50))

Sample: Mapped[List["Sample"]] = relationship(

back\_populates="Patient", cascade="all, delete-orphan"

)

def \_\_repr\_\_(self) -> str: #represents the object as a string

return f"Patient(pid={self.pid!r}, pFirst={self.pFirst!r}, pLast={self.pLast!r}, pDOB={self.pDOB!r}, pAddress={self.pAddress!r}, pEmail={self.pEmail!r}, pPhone={self.pPhone!r})"

class Sample(Base):

\_\_tablename\_\_ = "Sample"

sid: Mapped[str] = mapped\_column(String(50), primary\_key=True)

collection\_date: Mapped[str] = mapped\_column(Date)

sType: Mapped[str] = mapped\_column(String(50))

pid: Mapped[str] = mapped\_column(String(50), ForeignKey("Patient.pid"))

Patient: Mapped["Patient"] = relationship(back\_populates="Sample")

Result: Mapped["Result"] = relationship(

back\_populates="Sample",cascade="all, delete-orphan"

)

def \_\_repr\_\_(self) -> str:

return f"Sample(sid={self.sid!r}, collection\_date={self.collection\_date!r}, sType={self.Stype!r})"

class Result(Base):

\_\_tablename\_\_ = "Result"

rid: Mapped[str] = mapped\_column(String(50), primary\_key=True)

seqs: Mapped[str] = mapped\_column(String(100))

variant: Mapped[str] = mapped\_column(String(50))

drug: Mapped[str] = mapped\_column(String(50))

mlevel: Mapped[str] = mapped\_column(String(50))

sid: Mapped[str] = mapped\_column(String(50), ForeignKey("Sample.sid"))

Sample: Mapped["Sample"] = relationship(back\_populates="Result")

cid: Mapped[Optional[List[str]]] = mapped\_column(String(50), ForeignKey("Company.cid"))

Company: Mapped[Optional['Company']] = relationship(

back\_populates='Result')

def \_\_repr\_\_(self) -> str:

return f"Result(rid={self.rid!r}, seqs={self.seqs!r}, variant={self.variant!r}, drug={self.drug!r}, mlevel={self.mlevel!r})"

class Company(Base):

\_\_tablename\_\_ = "Company"

cid: Mapped[str] = mapped\_column(String(50), primary\_key=True)

cName: Mapped[str] = mapped\_column(String(50))

cEmail: Mapped[str] = mapped\_column(String(50))

cPhone: Mapped[str] = mapped\_column(String(50))

Result: Mapped[List["Result"]] = relationship(

back\_populates='Company', cascade='all, delete-orphan')

def \_\_repr\_\_(self) -> str:

return f"Company(cid={self.cid!r}, cName={self.cName!r}, cEmail={self.cEmail!r}, cPhone={self.cPhone!r})"

#Create Tables

Base.metadata.create\_all(engine)

#%%

# Test Data

with Session(engine) as session:

P001 = Patient(

pid ='P001',

pFirst ='Bradley',

pLast ='Ostberg',

pDOB ='1978-05-23',

pAddress ='1433 Cherry Street, Denver, Colorado',

pEmail ='bostberg@gmail.com',

pPhone ='720-123-4567',

Sample = [

Sample(sid='S001',collection\_date='2016-05-02',sType='oral',

Result=Result(rid='R001',seqs='ATGGTCTTACTTGGTCTTGCAGAAGCAGGGTATGGAACAGTCCCTTTGTCTTCC',variant='CYP2C19\*1/\*17',drug='Clopidogrel',mlevel='rapid',cid='C001')),

Sample(sid='S002',collection\_date='2016-05-02',sType='blood',

Result=Result(rid='R002',seqs='GTCCTGCTCGCGCGCTCGCGCGCGCGCGCGCGCTGCGCGCTGCGCGCGCGCGC',variant='CYP2C19\*1/\*17',drug='Voriconazole',mlevel='rapid',cid='C001'))

])

P002 = Patient(

pid ='P002',

pFirst ='Shirlee',

pLast ='Mould',

pDOB ='1986-01-14',

pAddress ='789 Elmwood Avenue, Austin, Texas',

pEmail ='smould@yahoo.com',

pPhone ='512-789-0123',

Sample = [

Sample(sid='S003',collection\_date='2016-05-18',sType='blood',

Result=Result(rid='R003',seqs='CGGAGTGACACGTCTTGAACTGTGATGTTGTGTCTTCAGTTTCCGAGAAGGGC',variant='CYP2C19\*1/\*1',drug='Voriconazole',mlevel='normal',cid='C003')

)

])

P003 = Patient(

pid ='P003',

pFirst ='Shania',

pLast ='Graves',

pDOB ='1992-08-07',

pAddress ='5279 Maple Drive, Seattle, Washington',

pEmail ='sgraves@gmail.com',

pPhone ='206-555-6789',

Sample = [

Sample(sid='S004',collection\_date='2019-08-21',sType='urine',

Result=Result(rid='R004',seqs='TGCTGCCAACTTGGAGGCGCAGCGCGAGCGCGCGCGCGCGCGCGCGCGCGCGC',variant='CYP2B6\*1/\*4',drug='Efavirenz',mlevel='rapid',cid='C002')

),

Sample(sid='S013',collection\_date='2020-10-12',sType='oral',

Result=Result(rid='R013',seqs='GGAGCTTTGGGAGGAAGCCAGGAAGAGTGCTCAGAGCTGGGAGGTGTTGTGC',variant='CYP2C19\*1/\*17',drug='Clopidogrel',mlevel='rapid',cid='C003'))

])

P004 = Patient(

pid ='P004',

pFirst ='Wisteria',

pLast ='Poole',

pDOB ='1972-10-11',

pAddress ='2218 Oak Street, New Orleans, Louisiana',

pEmail ='wpoole@hotmail.com',

pPhone ='504-234-5678',

Sample = [

Sample(sid='S005',collection\_date='2017-11-20',sType='oral',

Result=Result(rid='R005',seqs='GAGGGGGATGTTGGAGCTGCGGCGTTGCCTCTGGGGTTCTAGGTGTTTTGCTG',variant='CYP2B6\*6/\*6',drug='Efavirenz',mlevel='poor',cid='C002')),

Sample(sid='S006',collection\_date='2017-12-13',sType='blood',

Result=Result(rid='R006',seqs='GGAGCGTGCGCTTGCGCGCGAGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGC',variant='CYP2C19\*17/\*17',drug='Celecoxib',mlevel='ultrarapid',cid='C001'

))

])

P005 = Patient(

pid ='P005',

pFirst ='Sasha',

pLast ='Law',

pDOB ='1999-03-29',

pAddress ='4002 Pine Avenue, Sacramento, California',

pEmail ='slaw@gmail.com',

pPhone ='916-345-6789',

Sample = [

Sample(sid='S007',collection\_date='2020-06-04',sType='oral',

Result=Result(rid='R007',seqs='ACGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCG',variant='CYP2C19\*17/\*17',drug='Voriconazole',mlevel='ultrarapid',cid='C003'))

])

P006 = Patient(

pid ='P006',

pFirst ='Tahnee',

pLast ='Harlan',

pDOB ='1983-06-09',

pAddress ='9026 Cedar Lane, Indianapolis, Indiana',

pEmail ='tharlan@yahoo.com',

pPhone ='317-456-7890',

Sample = [

Sample(sid='S008',collection\_date='2020-03-02',sType='oral',

Result=Result(rid='R008',seqs='GATTTGGTTGGGGAGTTGCTGAGGCAGAAGGCTGGCCAGTGTTCTCTGATTTA',variant='CYP2C19\*2/\*2',drug='Clopidogrel',mlevel='poor',cid='C003'))

])

P007 = Patient(

pid ='P007',

pFirst ='Kae',

pLast ='Andrews',

pDOB ='1979-12-01',

pAddress ='6890 Birchwood Drive, Atlanta, Georgia',

pEmail ='kandrews@yahoo.com',

pPhone ='404-567-8901',

Sample = [

Sample(sid='S009',collection\_date='2018-07-23',sType='urine',

Result=Result(rid='R009',seqs='GCTCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCG',variant='CYP2C19\*1/\*17',drug='Voriconazole',mlevel='rapid',cid='C001')),

Sample(sid='S010',collection\_date='2018-07-23',sType='blood',

Result=Result(rid='R010',seqs='TGGTACGTGTGAGTCCAGGGTCCAGGATAGGCGTCTCCATCCCTGTGATGGG',variant='CYP2C19\*17/\*17',drug='Clopidogrel',mlevel='ultrarapid',cid='C002'

))

])

P008 = Patient(

pid ='P008',

pFirst ='Dora',

pLast ='Peyton',

pDOB ='1996-04-12',

pAddress ='1753 Spruce Street, Portland, Oregon',

pEmail ='dpeyton@gmail.com',

pPhone ='503-678-9012',

Sample = [

Sample(sid='S011',collection\_date='2020-06-08',sType='oral',

Result=Result(rid='R011',seqs='ATCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGC',variant='CYP2B6\*1/\*4',drug='Efavirenz',mlevel='rapid',cid='C001'

))

])

P009 = Patient(

pid ='P009',

pFirst ='Orrell',

pLast ='Scrivener',

pDOB ='1988-09-03',

pAddress ='3387 Aspen Court, Baltimore, Maryland',

pEmail ='oscrivener@gmail.com',

pPhone ='410-789-0123',

Sample = [

Sample(sid='S012',collection\_date='2020-07-19',sType='oral',

Result=Result(rid='R012',seqs='CGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGC',variant='CYP2B6\*1/\*1',drug='Efavirenz',mlevel='normal',cid='C002'))

])

P010 = Patient(

pid ='P010',

pFirst ='Dorinda',

pLast ='Law',

pDOB ='1976-11-25',

pAddress ='620 Poplar Road, Kansas City, Missouri',

pEmail ='dlaw@hotmail.com',

pPhone ='816-234-5678',

Sample = [

Sample(sid='S014',collection\_date='2021-01-06',sType='oral',

Result=Result(rid='R014',seqs='CTCTCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGC',variant='CYP2C19\*1/\*1',drug='Clopidogrel',mlevel='normal',cid='C003')),

Sample(sid='S015',collection\_date='2020-11-12',sType='oral',

Result=Result(rid='R015',seqs='GTGAGTGTGAGTCTGGAGAGGATGAGGACAGGGAAGAGGGACGGGAGGGAGC',variant='CYP2C19\*1/\*1',drug='Voriconazole',mlevel='normal',cid='C002'

))

])

C001 = Company(

cid = 'C001',

cName = 'Medtronic',

cEmail = 'info@medtronic.com',

cPhone = '692-254-0958',

Result = [])

C002 = Company(

cid="C002",

cName='Novartis',

cEmail="contact.center@novartis.com",

cPhone="789-574-8531",

Result = [])

C003 =Company(

cid="C003",

cName="Max Health",

cEmail="info.mh@max.com",

cPhone="727-382-0333",

Result = [])

session.add\_all([P001,P002,P003,P004,P005,P006,P007,P008,P009,P010,C001,C002,C003])

session.commit()

#Queries

session = Session(engine)

print('## BloodSamples')

stmt = (

select(Patient)

.join(Patient.Sample)

.where(Sample.sType == "blood")

)

for p in session.scalars(stmt):

print(p)

print('## PatientClopidogrelPrescription ')

stmt1 = (

select(Patient)

.join(Patient.Sample)

.join(Sample.Result)

.where(Result.drug == "Clopidogrel")

)

for p in session.scalars(stmt1):

print(p)

print("##PatientWithResultFromNovartis")

stmt2 = (

select(Patient,Company)

.join(Patient.Sample)

.join(Sample.Result)

.join(Result.Company)

.where(Company.cName == 'Novartis'))

for p in session.scalars(stmt2):

print(p)

print('#Patients\_with\_Variant\_CYP2C19\*1/\*17#')

stmt4 = (

select(Patient)

.join(Patient.Sample)

.join(Sample.Result)

.where(Result.variant == "CYP2C19\*1/\*17")

)

for p in session.scalars(stmt4):

print(p)

session.commit()