Source Code: AI Generated Report

Generated on: 2026-02-07 13:45:04

```
from sqlalchemy import create_engine, Column, Integer, String, Float, Date, ForeignKey
from sqlalchemy.orm import sessionmaker, declarative_base
from sqlalchemy.exc import IntegrityError
import json
from datetime import datetime

Base = declarative_base()
engine = create_engine('sqlite:///:memory:', connect_args={'check_same_thread': False})
Session = sessionmaker(bind=engine)

class User(Base):
    __tablename__ = 'users'
    id = Column(Integer, primary_key=True)
    username = Column(String, unique=True)
    password = Column(String)

class Transaction(Base):
    __tablename__ = 'transactions'
    id = Column(Integer, primary_key=True)
    user_id = Column(Integer, ForeignKey('users.id'))
    date = Column(Date)
    amount = Column(Float)
    category = Column(String)
```

```
    description = Column(String)
    payment_method = Column(String)

Base.metadata.create_all(bind=engine)

def add_user(username, password):
    session = Session()
    new_user = User(username=username, password=password)
    session.add(new_user)
    try:
        session.commit()
        print(f"User {username} added successfully.")
    except IntegrityError:
        session.rollback()
        print(f"User {username} already exists.")
    finally:
        session.close()

def add_transaction(user_id, date, amount, category, description, payment_method):
    session = Session()
           new_transaction   =   Transaction(user_id=user_id,   date=date,   amount=amount,
category=category, description=description, payment_method=payment_method)
    session.add(new_transaction)
    session.commit()
    session.close()
    print(f"Transaction added: {amount} on {date} in category {category}.")
```

```
def retrieve_transactions(user_id):
    session = Session()
            transactions   =   session.query(Transaction).filter(Transaction.user_id   ==
user_id).all()
    session.close()
    return transactions


def generate_report(transactions):
    total_income = sum(t.amount for t in transactions if t.amount > 0)
    total_expenses = sum(-t.amount for t in transactions if t.amount < 0)
    print(f"Total Income: {total_income}, Total Expenses: {total_expenses}")


# Mock data
add_user("john_doe", "securepassword")
add_transaction(1,  datetime(2023,  10,  1),  1000.0,  "Salary",  "Monthly  salary",  "Bank
Transfer")
add_transaction(1,  datetime(2023,  10,  5),  -200.0,  "Groceries",  "Weekly  groceries",
"Credit Card")
add_transaction(1,  datetime(2023,  10,  10),  -150.0,  "Utilities",  "Electricity  bill",
"Bank Transfer")

transactions = retrieve_transactions(1)
generate_report(transactions)
```