

# Лабораторная работа №10

Косенко Данил, Сабырбекулы Нурдаулет ПЗА

## Тема: Python и БД. ORM

**Цель:** освоение концепции ORM.

### Задача:

Для своей схемы базы данных:

- инициализировать проект с использованием SQLAlchemy.
- определить модели для представления различных сущностей.
- произвести добавление, редактирование и удаление данных.
- выполнить запросы на выборку данных из базы данных.

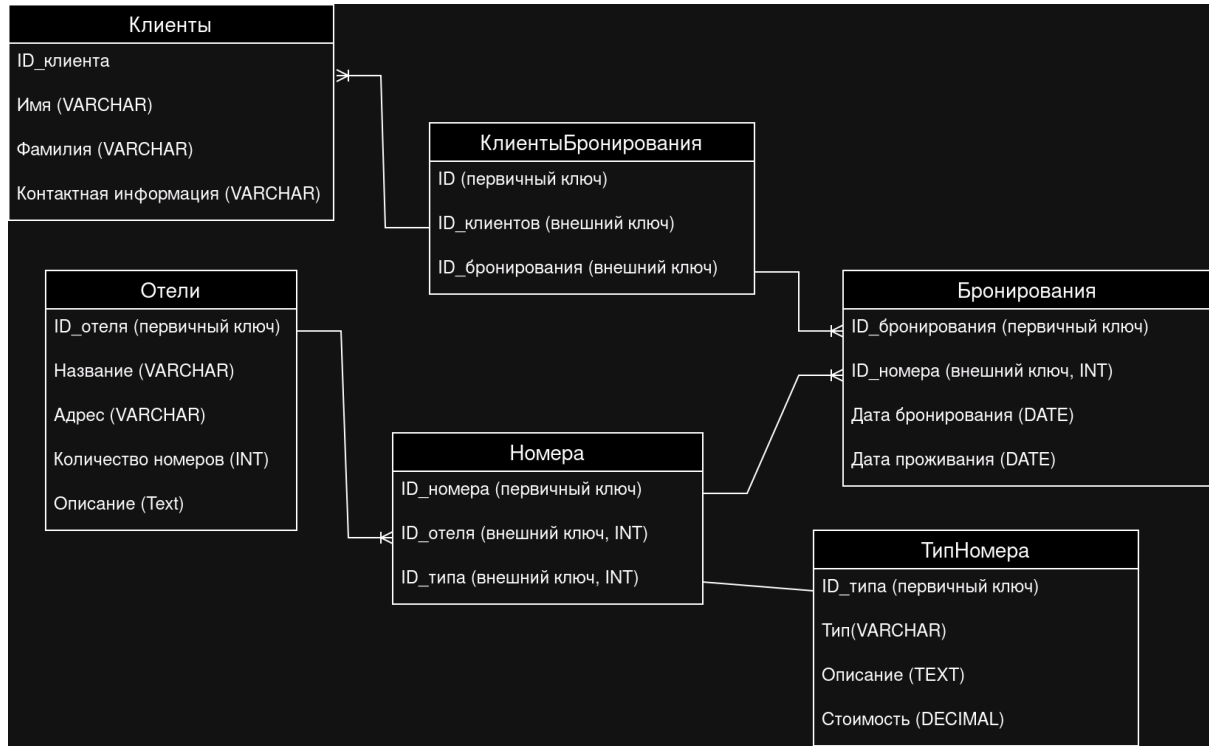
### Результаты:

Отчет должен содержать:

1. Текст задания.
2. Схема базы данных.
3. Листинг файла \*.py.
4. Скриншоты с результатами выполнения запросов.

# Бронирование отелей

## Схема БД:



## Листинг файла \*.py

```
from sqlalchemy import create_engine, Column, Integer, String, Date,
ForeignKey, Text, DECIMAL
from sqlalchemy.orm import declarative_base, sessionmaker, relationship
from datetime import date

Base = declarative_base()

# Модели для представления
class Clients(Base):
    __tablename__ = 'clients'
    client_id = Column(Integer, primary_key=True)
    first_name = Column(String(50))
    last_name = Column(String(50))
    contact_info = Column(String(100))
    reservations = relationship("ClientsReservations",
back_populates="client")
```

```

class Hotels(Base):
    __tablename__ = 'hotels'
    hotel_id = Column(Integer, primary_key=True)
    name = Column(String(100))
    address = Column(String(100))
    room_count = Column(Integer)
    description = Column(Text)
    rooms = relationship("Rooms", back_populates="hotel")

class Rooms(Base):
    __tablename__ = 'rooms'
    room_id = Column(Integer, primary_key=True)
    hotel_id = Column(Integer, ForeignKey('hotels.hotel_id'))
    type_id = Column(Integer, ForeignKey('roomtype.type_id'))
    hotel = relationship("Hotels", back_populates="rooms")
    room_type = relationship("RoomType", back_populates="rooms")
    reservations = relationship("Reservations", back_populates="room")

class RoomType(Base):
    __tablename__ = 'roomtype'
    type_id = Column(Integer, primary_key=True)
    type = Column(String(50))
    description = Column(Text)
    cost = Column(DECIMAL(10, 2))
    rooms = relationship("Rooms", back_populates="room_type")

class Reservations(Base):
    __tablename__ = 'reservations'
    reservation_id = Column(Integer, primary_key=True)
    room_id = Column(Integer, ForeignKey('rooms.room_id'))
    booking_date = Column(Date)
    stay_date = Column(Date)
    clients_reservations = relationship("ClientsReservations",
back_populates="reservation")
    room = relationship("Rooms", back_populates="reservations")

class ClientsReservations(Base):
    __tablename__ = 'clientsreservations'
    id = Column(Integer, primary_key=True)
    client_id = Column(Integer, ForeignKey('clients.client_id'))
    reservation_id = Column(Integer,
ForeignKey('reservations.reservation_id'))

```

```

    client = relationship("Clients", back_populates="reservations")
    reservation = relationship("Reservations",
back_populates="clients_reservations")

engine =
create_engine('postgresql://postgres:root@localhost:5543/hotelreservations1')
Base.metadata.create_all(engine)

Session = sessionmaker(bind=engine)
session = Session()

# Добавление данных
print("До добавления данных:")
clients_before = session.query(Clients).all()
for client in clients_before:
    print("|".join([str(client.client_id), client.first_name,
client.last_name, client.contact_info]))
print('\n')

new_client = Clients(first_name="Новый", last_name="Клиент",
contact_info="new.client@example.com")
session.add(new_client)
session.commit()

print("После добавления данных:")
clients_after = session.query(Clients).all()
for client in clients_after:
    print("|".join([str(client.client_id), client.first_name,
client.last_name, client.contact_info]))
print('\n')

# Редактирование данных
client_to_edit =
session.query(Clients).filter_by(first_name="Новый").first()
if client_to_edit:
    print("До редактирования данных:")
    print("|".join([str(client_to_edit.client_id),
client_to_edit.first_name, client_to_edit.last_name,
client_to_edit.contact_info]))

```

```
print('\n')

client_to_edit.first_name = "Отредактированный"
client_to_edit.last_name = "Клиент"
client_to_edit.contact_info = "edited.client@example.com"
session.commit()

print("После редактирования данных:")
print("|".join([str(client_to_edit.client_id),
client_to_edit.first_name, client_to_edit.last_name,
client_to_edit.contact_info]))
print('\n')
else:
    print("Клиент с именем 'Новый' не найден.")
    print('\n')

# Удаление данных
client_to_delete =
session.query(Clients).filter_by(first_name="Отредактированный").first(
)
if client_to_delete:
    print("До удаления данных:")
    clients_before_deletion = session.query(Clients).all()
    for client in clients_before_deletion:
        print("|".join([str(client.client_id), client.first_name,
client.last_name, client.contact_info]))
    print('\n')

    session.delete(client_to_delete)
    session.commit()

    print("После удаления данных:")
    clients_after_deletion = session.query(Clients).all()
    for client in clients_after_deletion:
        print("|".join([str(client.client_id), client.first_name,
client.last_name, client.contact_info]))
    print('\n')
else:
    print("Клиент с именем 'Отредактированный' не найден.")
    print('\n')
```

```
# Примеры запросов:
print("Примеры запросов:\n")

# Получение всех клиентов
print("Получение всех клиентов:")
clients = session.query(Clients).all()
for client in clients:
    print("|".join([str(client.client_id), client.first_name,
client.last_name, client.contact_info]))
print('\n')

# Получение всех отелей с их описанием
print("Получение всех отелей с их описанием:")
hotels = session.query(Hotels).all()
for hotel in hotels:
    print("|".join([str(hotel.hotel_id), hotel.name,
hotel.description]))
print('\n')

# Получение всех бронирований с датами
print("Получение всех бронирований с датами:")
reservations = session.query(Reservations).all()
for reservation in reservations:
    print("|".join([str(reservation.reservation_id),
str(reservation.booking_date), str(reservation.stay_date)]))
print('\n')

# Получение всех типов комнат
print("Получение всех типов комнат:")
room_types = session.query(RoomType).all()
for room_type in room_types:
    print("|".join([str(room_type.type_id), room_type.type,
room_type.description, str(room_type.cost)]))
print('\n')

# Получение всех клиентов, которые забронировали номера
print("Получение всех клиентов, которые забронировали номера:")
clients_with_reservations =
session.query(Clients).filter(Clients.reservations != None).all()
for client in clients_with_reservations:
    print("|".join([str(client.client_id), client.first_name,
client.last_name, client.contact_info]))
print('\n')
```

## Результаты выполнения запросов:

```
● PS C:\Users\bossd\OneDrive\Desktop\lab10P0> & C:/Users/bossd/OneDrive/Desktop/lab10P0/main.py
До добавления данных:
2|Анна|Петрова|anna.petrova@example.com
3|Сергей|Сидоров|sergey.sidorov@example.com
4|Мария|Смирнова|maria.smirnova@example.com
5|Петр|Кузнецов|petr.kuznetsov@example.com
6|Елена|Новикова|elena.novikova@example.com
7|Наталья|Козлова|natalya.kozlova@example.com
1|Иван|Иванов|updated_contact@example.com

После добавления данных:
2|Анна|Петрова|anna.petrova@example.com
3|Сергей|Сидоров|sergey.sidorov@example.com
4|Мария|Смирнова|maria.smirnova@example.com
5|Петр|Кузнецов|petr.kuznetsov@example.com
6|Елена|Новикова|elena.novikova@example.com
7|Наталья|Козлова|natalya.kozlova@example.com
1|Иван|Иванов|updated_contact@example.com
9|Новый|Клиент|new.client@example.com

До редактирования данных:
9|Новый|Клиент|new.client@example.com

После редактирования данных:
9|Отредактированный|Клиент|edited.client@example.com
```

До удаления данных:

```
2|Анна|Петрова|anna.petrova@example.com
3|Сергей|Сидоров|sergey.sidorov@example.com
4|Мария|Смирнова|maria.smirnova@example.com
5|Петр|Кузнецов|petr.kuznetsov@example.com
6|Елена|Новикова|elena.novikova@example.com
7|Наталья|Козлова|natalya.kozlova@example.com
1|Иван|Иванов|updated_contact@example.com
9|Отредактированный|Клиент|edited.client@example.com
```

После удаления данных:

```
2|Анна|Петрова|anna.petrova@example.com
3|Сергей|Сидоров|sergey.sidorov@example.com
4|Мария|Смирнова|maria.smirnova@example.com
5|Петр|Кузнецов|petr.kuznetsov@example.com
6|Елена|Новикова|elena.novikova@example.com
7|Наталья|Козлова|natalya.kozlova@example.com
1|Иван|Иванов|updated_contact@example.com
```



Примеры запросов:

Получение всех клиентов:

```
2|Анна|Петрова|anna.petrova@example.com
3|Сергей|Сидоров|sergey.sidorov@example.com
4|Мария|Смирнова|maria.smirnova@example.com
5|Петр|Кузнецов|petr.kuznetsov@example.com
6|Елена|Новикова|elena.novikova@example.com
7|Наталья|Козлова|natalya.kozlova@example.com
1|Иван|Иванов|updated_contact@example.com
```

Получение всех отелей с их описанием:

```
1|Отель Астана|Отель в центре столицы Казахстана
2|Горный курорт Алатау|Горный курорт с видом на Алматы
3|Прибрежный отель Бархатный берег|Отель у пляжа на Каспийском море
4|Отель Волжский|Уютный отель на берегу Волги
5|Горнолыжный курорт Снежная вершина|Идеальное место для зимнего отдыха
6|Отели Ривьеры|Лучшие отели Сочи с видом на море
```

Получение всех бронирований с датами:

```
2|2023-11-01|2023-11-03
3|2023-11-10|2023-11-15
4|2023-11-20|2023-11-25
5|2023-11-15|2023-11-18
6|2023-12-01|2023-12-10
```

Получение всех типов комнат:

```
1|Стандартный|Телевизор, Wi-Fi, кондиционер|10000.00
2|Делюкс|Телевизор, Wi-Fi, кондиционер, вид на город|15000.00
3|Горная хижина|Телевизор, камин, панорамные окна в горы|8000.00
4|Люкс|Телевизор, бассейн, ресторан у моря|20000.00
5|Полулюкс|Телевизор, мини-бар, вид на город|18000.00
6|Фешн-люкс|VIP-сервис, джакузи, панорамные окна|25000.00
7|Эконом|Ограниченные удобства по минимальной цене|8000.00
```

Получение всех клиентов, которые забронировали номера:

```
2|Анна|Петрова|anna.petrova@example.com
3|Сергей|Сидоров|sergey.sidorov@example.com
4|Мария|Смирнова|maria.smirnova@example.com
5|Петр|Кузнецов|petr.kuznetsov@example.com
6|Елена|Новикова|elena.novikova@example.com
1|Иван|Иванов|updated_contact@example.com
```

PS C:\Users\bossd\OneDrive\Desktop\lab10PO> █