Содержание

1. Формулировка задания

Спроектировать базу данных для туроператора. База данных должна содержать информацию о турах, пользователях, отелях и перелетах, а также отражать данные о заказах и "избранном" пользователей.

2. Концептуальная модель базы данных

После проведения анализа предметной области была спроектирована следующая концептуальная модель:

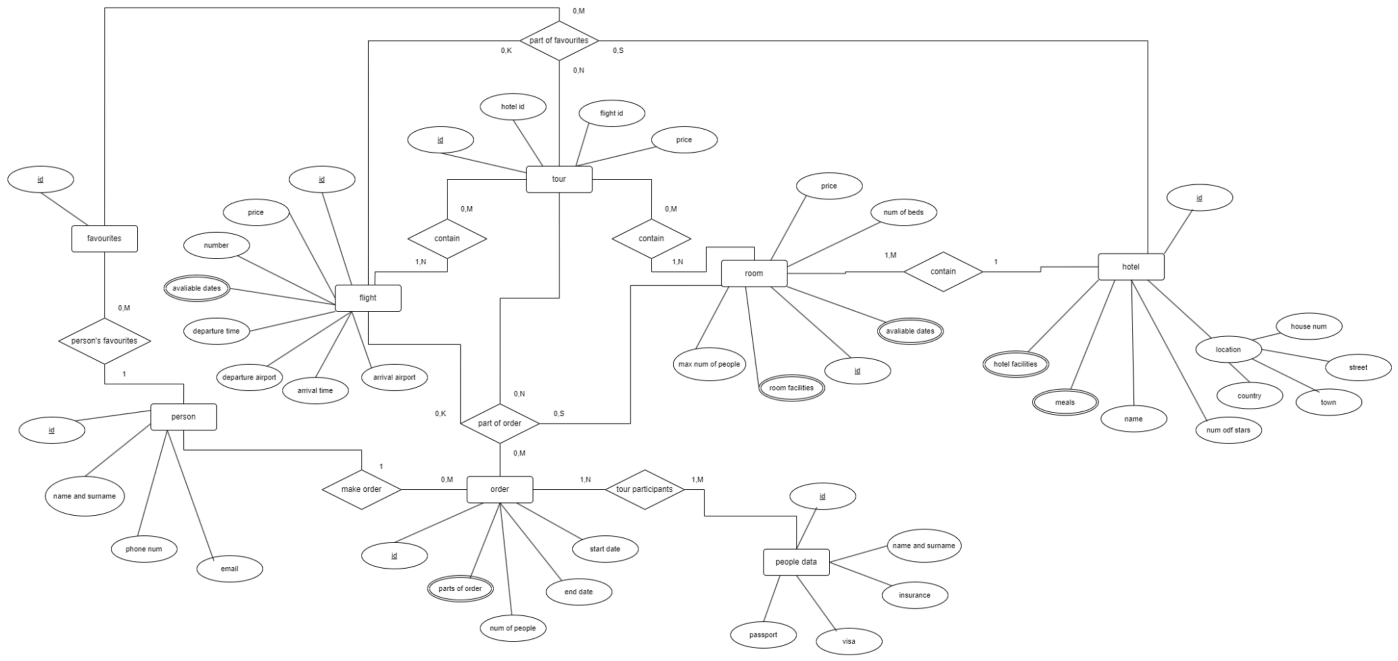


Рис. 1 – Концептуальная модель базы данных

2.1. Конкретизация предметной области

Необходимо создать систему, отражающую информацию о турах, отелях, перелетах и их бронировании. Каждый пользователь может заказать тур, номер в отеле или перелет для группы человек, а также он может добавить любой тур, отель или перелет в избранное

2.2. Описание предметной области

Система рассчитана на работу с зарегистрированными работниками турагентства, обычные пользователи доступа к базе данных не имеют.

Каждому конкретному пользователю доступна следующая информация: список всех туров, отелей и перелетов, история заказов и список избранных позиций.

2.3. Описание атрибутов

В процессе анализа были выделены следующие атрибуты, название и описание которых приведены в таблице ниже:

Имя атрибута	**Расшифровка**
id	Уникальный идентификатор. Есть у каждого объекта.
nearest_airpor_code	Трехбуквенный код аэропорта
country	Страна
name	Имя(пользователя/отеля/города)
street	Улица на которой расположен отель
house	Номер здания, где расположен отель
num_of_stars	Число звезд отеля
swimming_pool, parking, gymnasium, spa_center, free_WiFi, private_beach, restaurant, golf_field, bar	Удобства в отеле
all_inclusive_price, BB_price, HB_price,	Цены в отеле за разные типы питания

FB_price	
max_num_of_people	Вместимость номера
num_of_beds	Число кроватей в номере
price	Цена(номера/тура/перелета)
num_of_rooms_in_hotel	Число номеров такого типа в отеле
air_condition, chimney, balcon, kitchen, private_bathroom, mini_bar, tea_coffee, TV	Удобства в номере
departure_airport_townId	Аэропорт вылета
arrival_airport_townId	Аэропорт прилета
flight_number	Номер перелета
departure_time	Время вылета
arrival_time	Время прилета
surname	Фамилия
phone_num	Номер телефона пользователя
email	Почта пользователя
start_date	Дата начала бронирования
end_date	Дата окончания бронирования
insurance	Номер страховки
visa	Информация о визе
passport	Номер загранпаспорта

3. Логическое проектирование

Следующим шагом на основе КМПО была разработана логическая модель базы данных, представленная ниже:

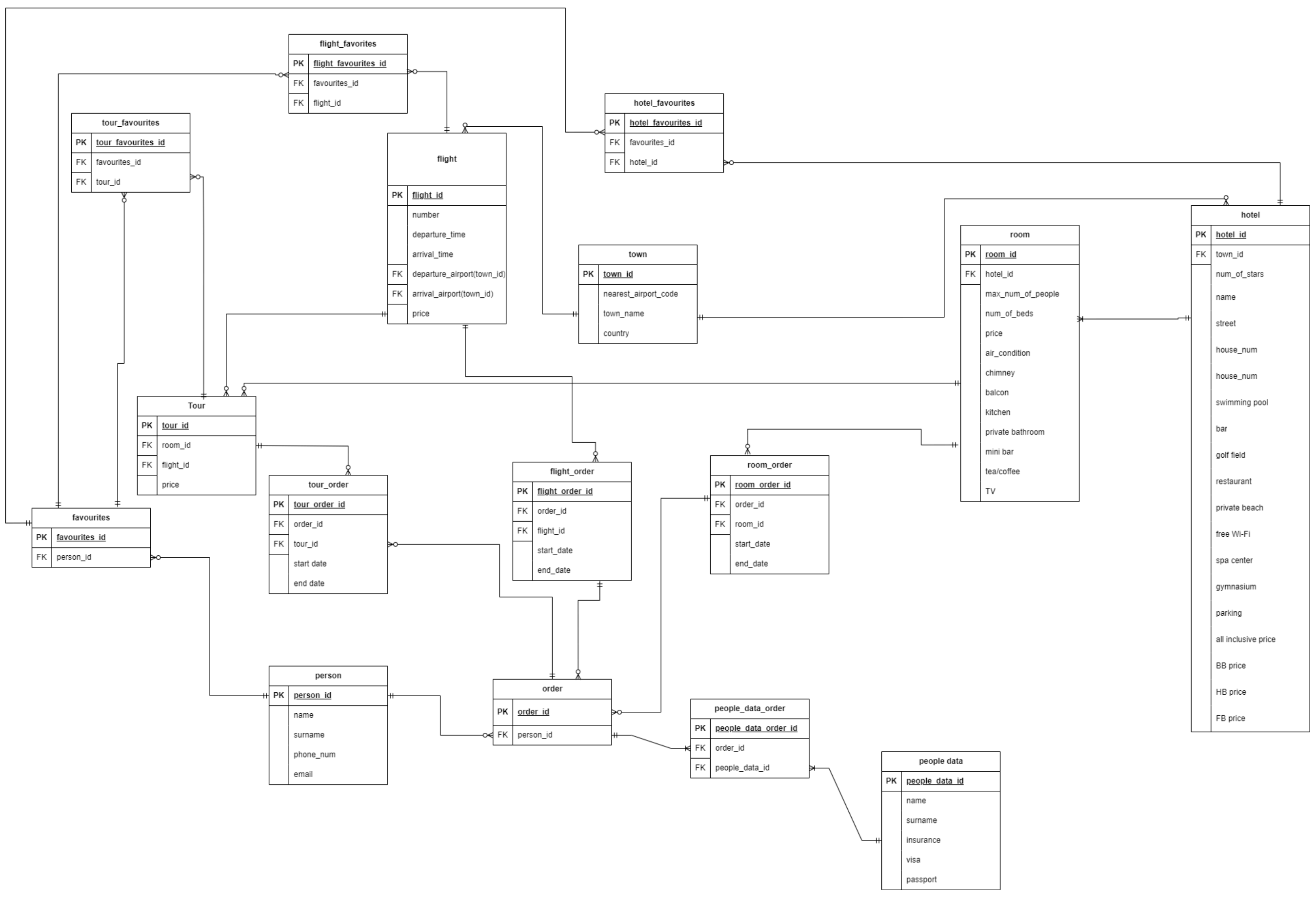


Рис. 2 – Логическая модель базы данных

4. Физическое проектирование

В качестве СУБД для реализации разработанной базы данных была выбрана PostgreSQL. В связи с проведённым анализом предметной области была проработана следующая физическая схема БД. Она представлена на следующем рисунке:

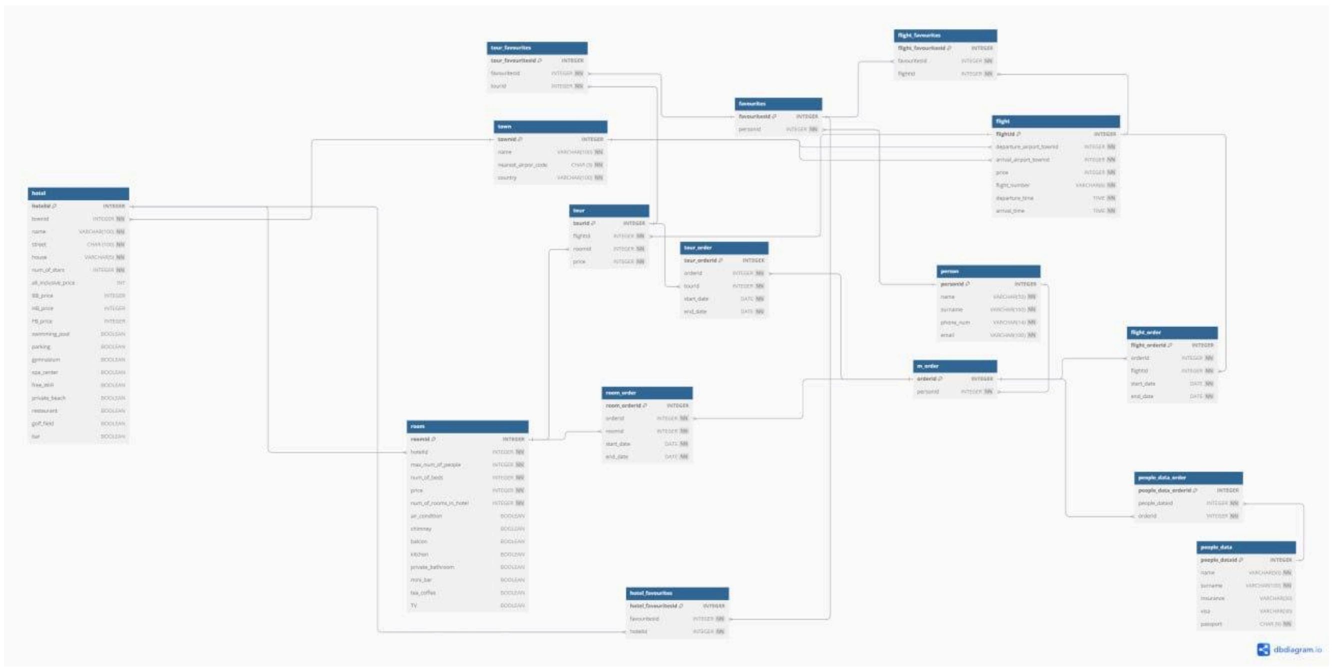
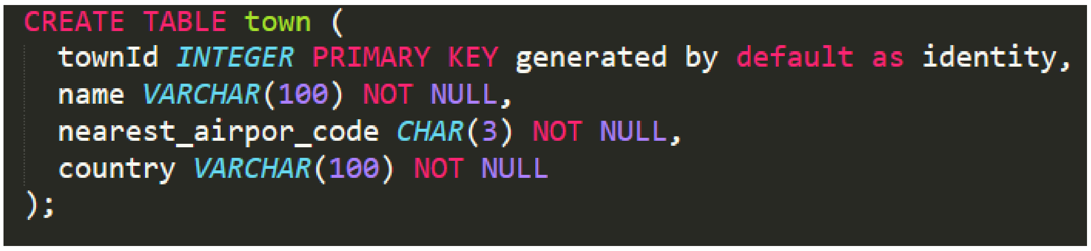


Рис. 3 – Графическое представление базы данных

4.1. Создание таблиц

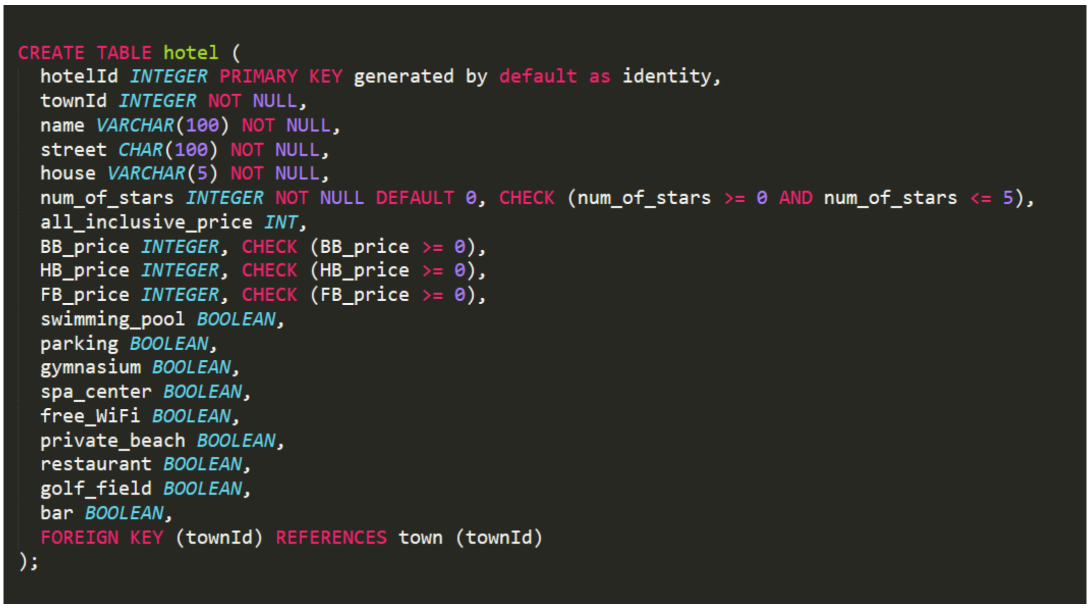
Ниже приведены запросы на языке SQL (диалект PostgreSQL) для создания таблиц, описанных выше.

- Код для создания таблицы “Town”:



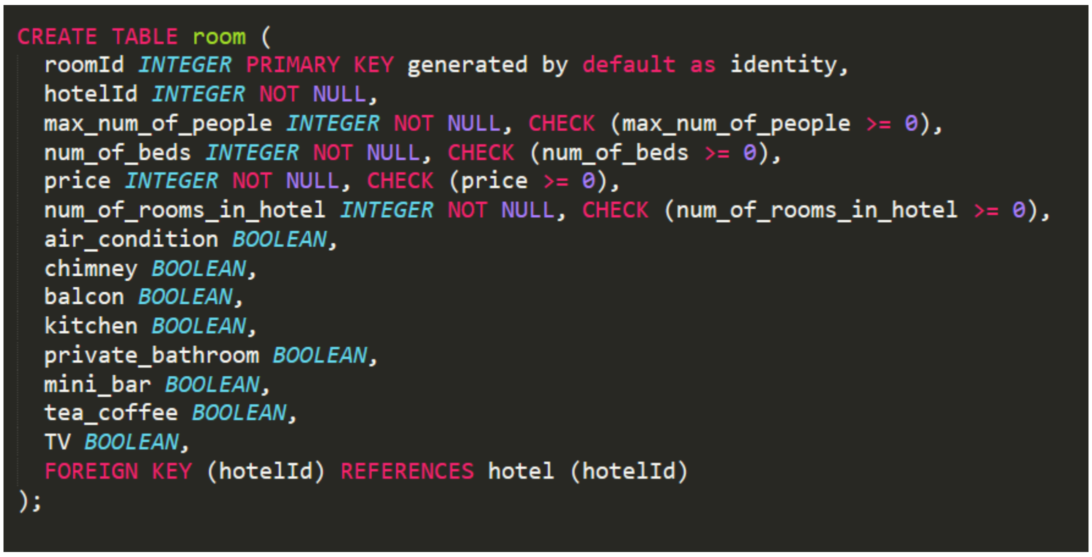
```
CREATE TABLE town (
  townId INTEGER PRIMARY KEY generated by default as identity,
  name VARCHAR(100) NOT NULL,
  nearest_airpor_code CHAR(3) NOT NULL,
  country VARCHAR(100) NOT NULL
);
```

- Код для создания таблицы “Hotel”:



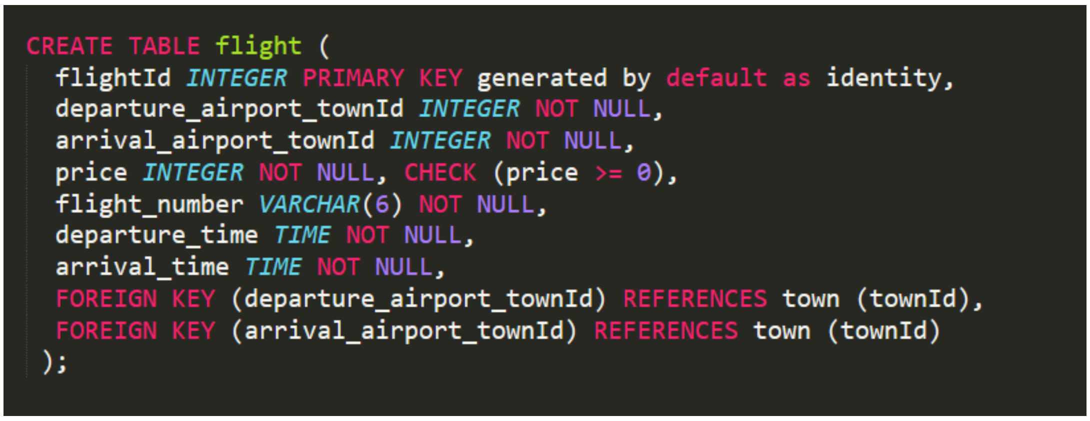
```
CREATE TABLE hotel (
  hotelId INTEGER PRIMARY KEY generated by default as identity,
  townId INTEGER NOT NULL,
  name VARCHAR(100) NOT NULL,
  street CHAR(100) NOT NULL,
  house VARCHAR(5) NOT NULL,
  num_of_stars INTEGER NOT NULL DEFAULT 0, CHECK (num_of_stars >= 0 AND num_of_stars <= 5),
  all_inclusive_price INT,
  BB_price INTEGER, CHECK (BB_price >= 0),
  HB_price INTEGER, CHECK (HB_price >= 0),
  FB_price INTEGER, CHECK (FB_price >= 0),
  swimming_pool BOOLEAN,
  parking BOOLEAN,
  gymnasium BOOLEAN,
  spa_center BOOLEAN,
  free_WiFi BOOLEAN,
  private_beach BOOLEAN,
  restaurant BOOLEAN,
  golf_field BOOLEAN,
  bar BOOLEAN,
  FOREIGN KEY (townId) REFERENCES town (townId)
);
```

- Код для создания таблицы “Room”:



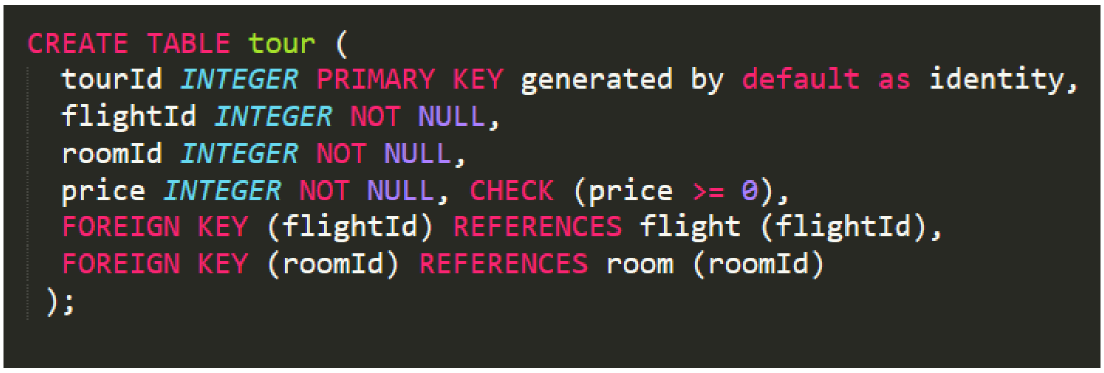
```
CREATE TABLE room (
  roomId INTEGER PRIMARY KEY generated by default as identity,
  hotelId INTEGER NOT NULL,
  max_num_of_people INTEGER NOT NULL, CHECK (max_num_of_people >= 0),
  num_of_beds INTEGER NOT NULL, CHECK (num_of_beds >= 0),
  price INTEGER NOT NULL, CHECK (price >= 0),
  num_of_rooms_in_hotel INTEGER NOT NULL, CHECK (num_of_rooms_in_hotel >= 0),
  air_condition BOOLEAN,
  chimney BOOLEAN,
  balcon BOOLEAN,
  kitchen BOOLEAN,
  private_bathroom BOOLEAN,
  mini_bar BOOLEAN,
  tea_coffee BOOLEAN,
  TV BOOLEAN,
  FOREIGN KEY (hotelId) REFERENCES hotel (hotelId)
);
```

- Код для создания таблицы “Flight”:



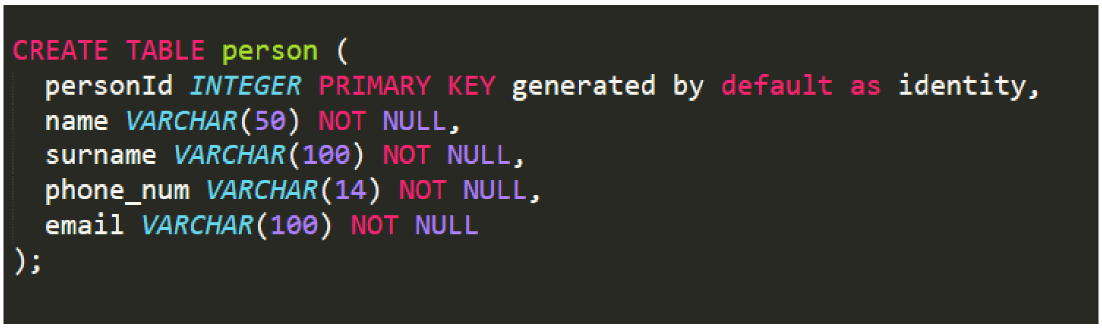
```
CREATE TABLE flight (
  flightId INTEGER PRIMARY KEY generated by default as identity,
  departure_airport_townId INTEGER NOT NULL,
  arrival_airport_townId INTEGER NOT NULL,
  price INTEGER NOT NULL, CHECK (price >= 0),
  flight_number VARCHAR(6) NOT NULL,
  departure_time TIME NOT NULL,
  arrival_time TIME NOT NULL,
  FOREIGN KEY (departure_airport_townId) REFERENCES town (townId),
  FOREIGN KEY (arrival_airport_townId) REFERENCES town (townId)
 );
```

- Код для создания таблицы “Tour”:



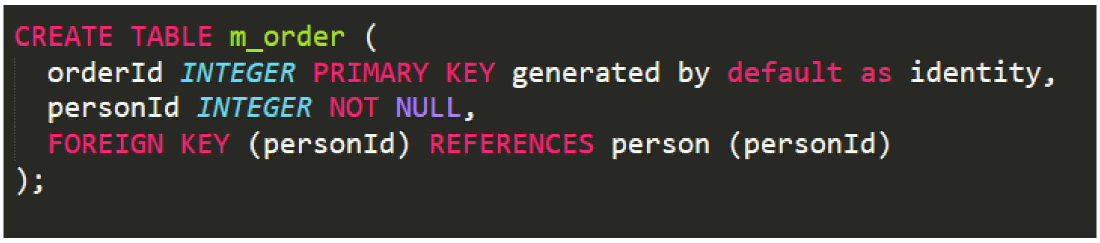
```
CREATE TABLE tour (
  tourId INTEGER PRIMARY KEY generated by default as identity,
  flightId INTEGER NOT NULL,
  roomId INTEGER NOT NULL,
  price INTEGER NOT NULL, CHECK (price >= 0),
  FOREIGN KEY (flightId) REFERENCES flight (flightId),
  FOREIGN KEY (roomId) REFERENCES room (roomId)
 );
```

- Код для создания таблицы “Person”:



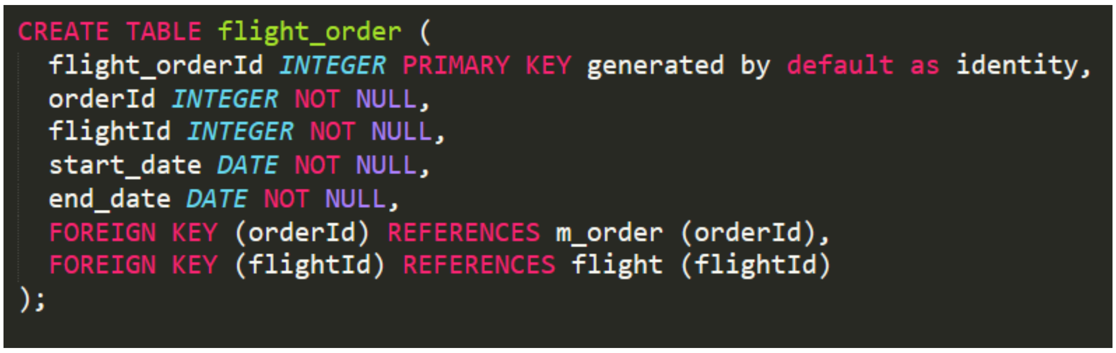
```
CREATE TABLE person (
  personId INTEGER PRIMARY KEY generated by default as identity,
  name VARCHAR(50) NOT NULL,
  surname VARCHAR(100) NOT NULL,
  phone_num VARCHAR(14) NOT NULL,
  email VARCHAR(100) NOT NULL
);
```

- Код для создания таблицы “Order”:



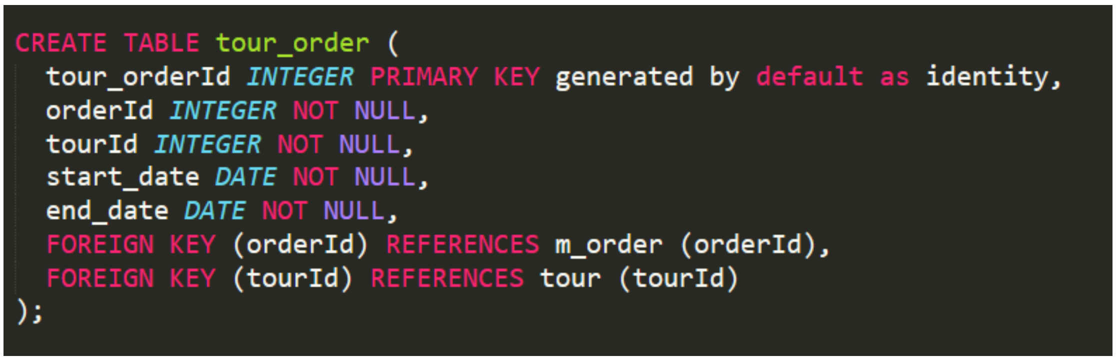
```
CREATE TABLE m_order (
  orderId INTEGER PRIMARY KEY generated by default as identity,
  personId INTEGER NOT NULL,
  FOREIGN KEY (personId) REFERENCES person (personId)
);
```

- Код для создания таблицы “Flight_order”:



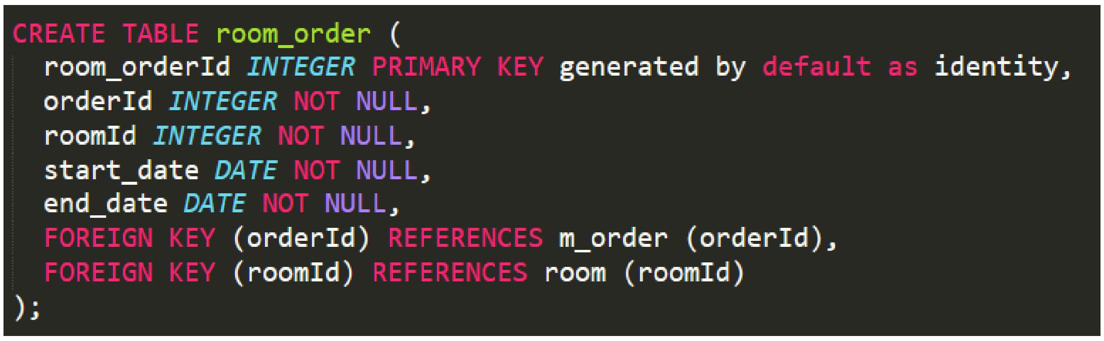
```
CREATE TABLE flight_order (
  flight_orderId INTEGER PRIMARY KEY generated by default as identity,
  orderId INTEGER NOT NULL,
  flightId INTEGER NOT NULL,
  start_date DATE NOT NULL,
  end_date DATE NOT NULL,
  FOREIGN KEY (orderId) REFERENCES m_order (orderId),
  FOREIGN KEY (flightId) REFERENCES flight (flightId)
);
```

- Код для создания таблицы “Tour_order”:



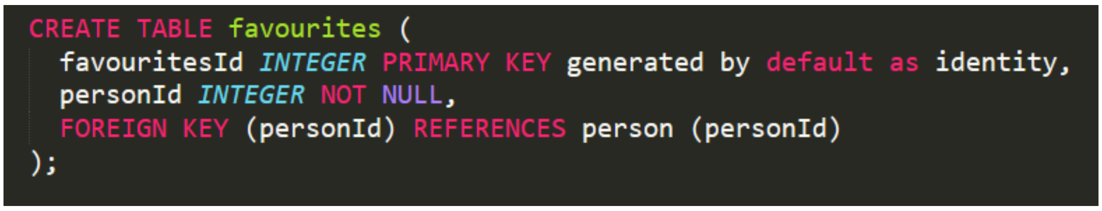
```
CREATE TABLE tour_order (
  tour_orderId INTEGER PRIMARY KEY generated by default as identity,
  orderId INTEGER NOT NULL,
  tourId INTEGER NOT NULL,
  start_date DATE NOT NULL,
  end_date DATE NOT NULL,
  FOREIGN KEY (orderId) REFERENCES m_order (orderId),
  FOREIGN KEY (tourId) REFERENCES tour (tourId)
);
```

- Код для создания таблицы “Room_order”:



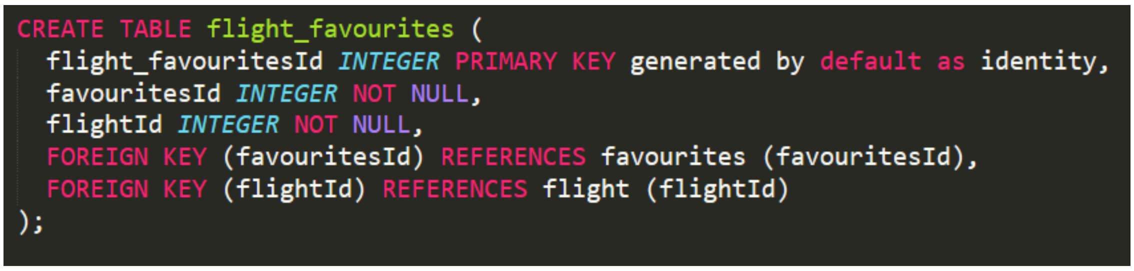
```
CREATE TABLE room_order (
  room_orderId INTEGER PRIMARY KEY generated by default as identity,
  orderId INTEGER NOT NULL,
  roomId INTEGER NOT NULL,
  start_date DATE NOT NULL,
  end_date DATE NOT NULL,
  FOREIGN KEY (orderId) REFERENCES m_order (orderId),
  FOREIGN KEY (roomId) REFERENCES room (roomId)
);
```

- Код для создания таблицы “Favourites”:



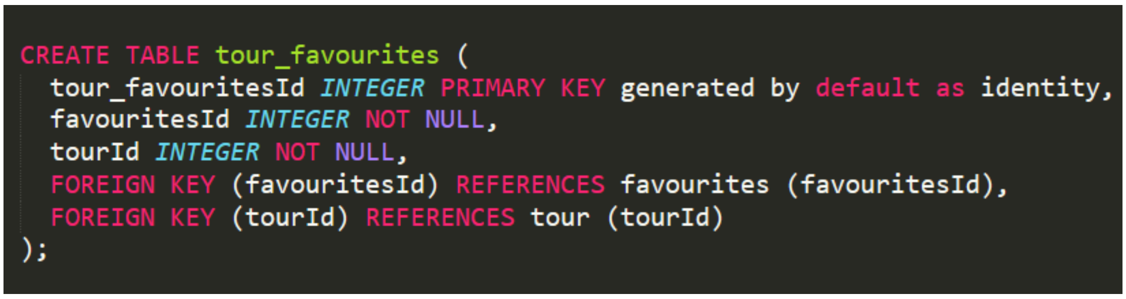
```
CREATE TABLE favourites (
  favouritesId INTEGER PRIMARY KEY generated by default as identity,
  personId INTEGER NOT NULL,
  FOREIGN KEY (personId) REFERENCES person (personId)
);
```

- Код для создания таблицы “Flight_favourites”:



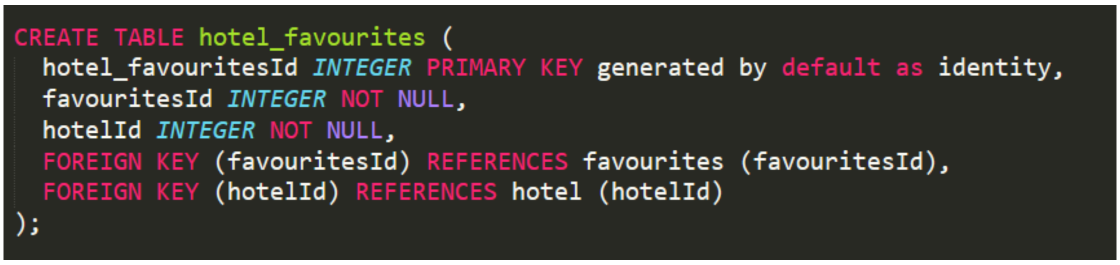
```sql
CREATE TABLE flight_favourites (
  flight_favouritesId INTEGER PRIMARY KEY generated by default as identity,
  favouritesId INTEGER NOT NULL,
  flightId INTEGER NOT NULL,
  FOREIGN KEY (favouritesId) REFERENCES favourites (favouritesId),
  FOREIGN KEY (flightId) REFERENCES flight (flightId)
);
```

- Код для создания таблицы “Tour_favourites”:



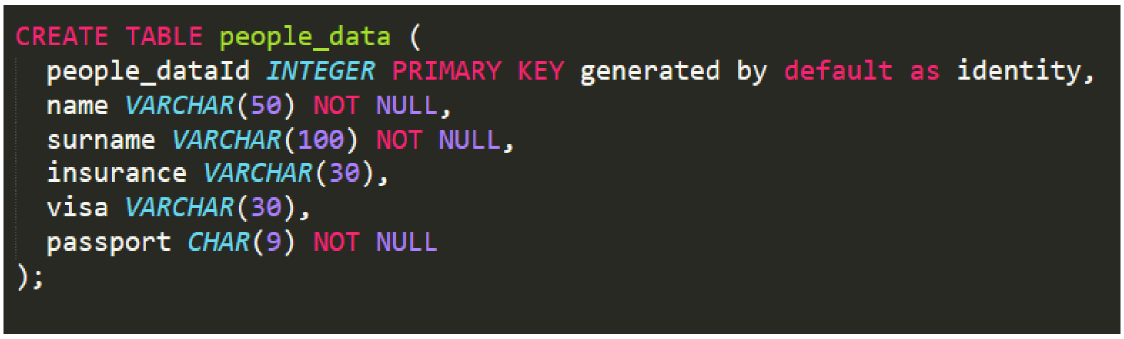
```sql
CREATE TABLE tour_favourites (
  tour_favouritesId INTEGER PRIMARY KEY generated by default as identity,
  favouritesId INTEGER NOT NULL,
  tourId INTEGER NOT NULL,
  FOREIGN KEY (favouritesId) REFERENCES favourites (favouritesId),
  FOREIGN KEY (tourId) REFERENCES tour (tourId)
);
```

- Код для создания таблицы “Hotel_favourites”:



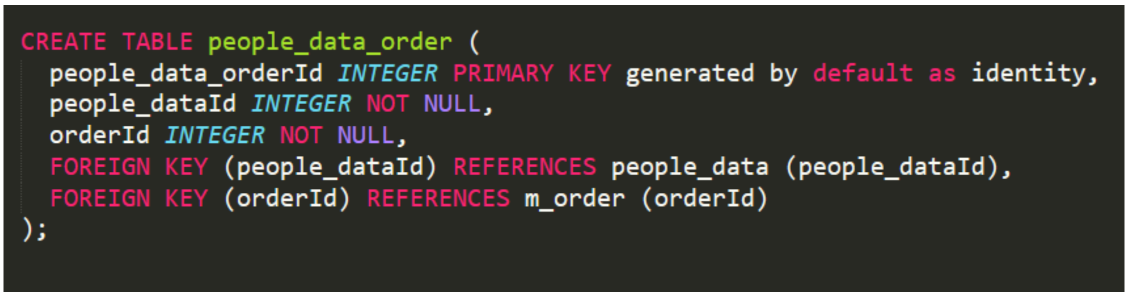
```sql
CREATE TABLE hotel_favourites (
  hotel_favouritesId INTEGER PRIMARY KEY generated by default as identity,
  favouritesId INTEGER NOT NULL,
  hotelId INTEGER NOT NULL,
  FOREIGN KEY (favouritesId) REFERENCES favourites (favouritesId),
  FOREIGN KEY (hotelId) REFERENCES hotel (hotelId)
);
```

- Код для создания таблицы “People_data”:



```sql
CREATE TABLE people_data (
  people_dataId INTEGER PRIMARY KEY generated by default as identity,
  name VARCHAR(50) NOT NULL,
  surname VARCHAR(100) NOT NULL,
  insurance VARCHAR(30),
  visa VARCHAR(30),
  passport CHAR(9) NOT NULL
);
```

- Код для создания таблицы “People_data_order”:



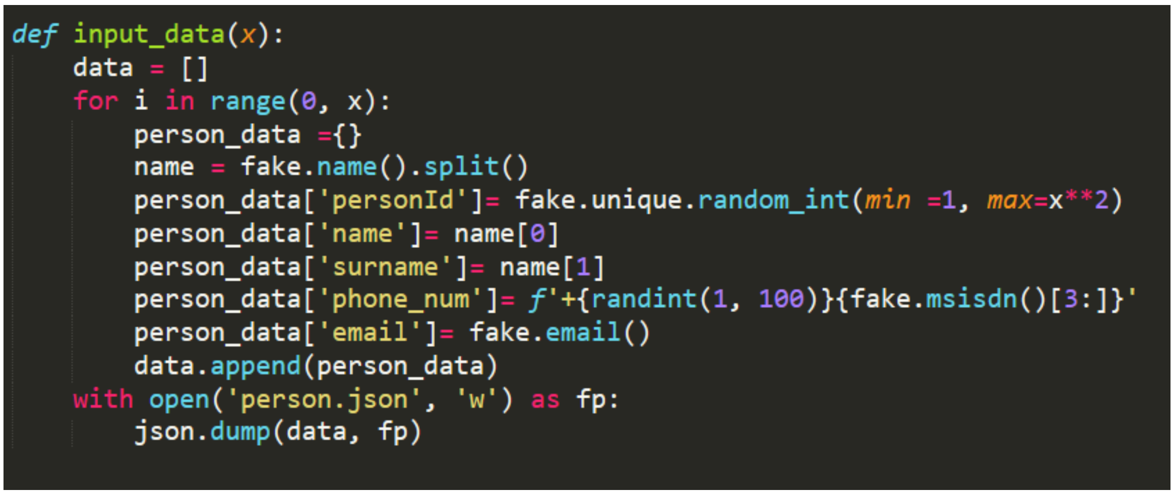
```sql
CREATE TABLE people_data_order (
  people_data_orderId INTEGER PRIMARY KEY generated by default as identity,
  people_dataId INTEGER NOT NULL,
  orderId INTEGER NOT NULL,
  FOREIGN KEY (people_dataId) REFERENCES people_data (people_dataId),
  FOREIGN KEY (orderId) REFERENCES m_order (orderId)
);
```

4.2. Заполнение базы данных

Заполнение базы данных проводилось при помощи библиотеки для работы с PostgreSQL psycopg2 и библиотеки Faker языка программирования Python. С помощью программы были созданы таблицы, которые потом с помощью команды COPY вставлялись в базу данных. Данные для таблицы town были взяты из интернета (https://docs.google.com/spreadsheets/d/1eepIWOHicQsLyZsb0mSXGPTXDp3vlql-aGuy1AWJED0/edit?gid=863195591#gid=863195591)

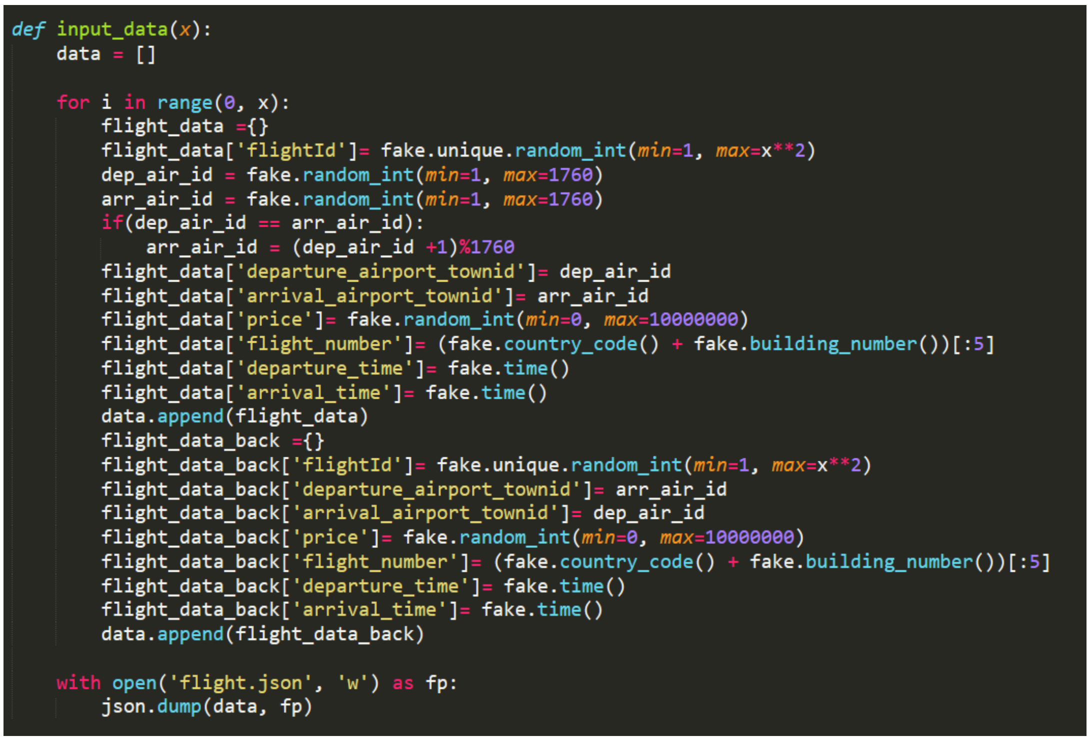
4.2.1 Программа для заполнения базы данных

- Код для заполнения таблицы “Person”:



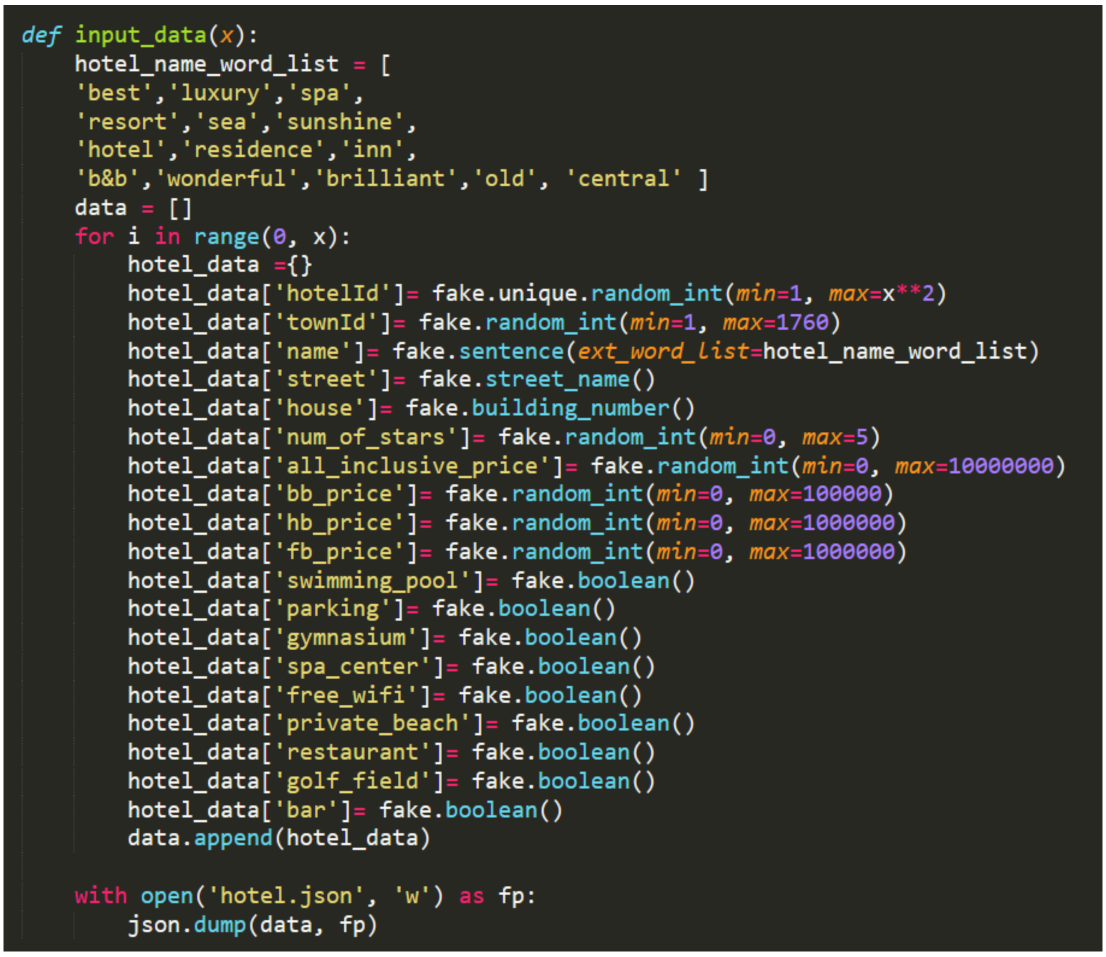
```python
def input_data(x):
    data = []
    for i in range(0, x):
        person_data ={}
        name = fake.name().split()
        person_data['personId']= fake.unique.random_int(min =1, max=x**2)
        person_data['name']= name[0]
        person_data['surname']= name[1]
        person_data['phone_num']= f'+{randint(1, 100)}{fake.msisdn()[3:]}'
        person_data['email']= fake.email()
        data.append(person_data)
    with open('person.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Flight”:



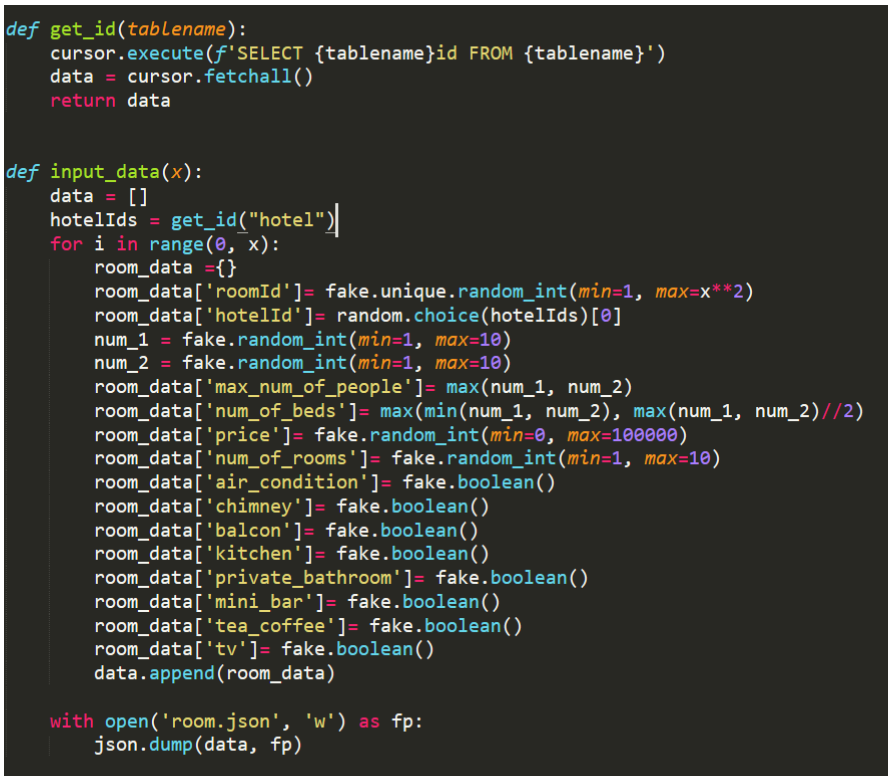
```python
def input_data(x):
    data = []

    for i in range(0, x):
        flight_data ={}
        flight_data['flightId']= fake.unique.random_int(min=1, max=x**2)
        dep_air_id = fake.random_int(min=1, max=1760)
        arr_air_id = fake.random_int(min=1, max=1760)
        if(dep_air_id == arr_air_id):
            arr_air_id = (dep_air_id +1)%1760
        flight_data['departure_airport_townid']= dep_air_id
        flight_data['arrival_airport_townid']= arr_air_id
        flight_data['price']= fake.random_int(min=0, max=10000000)
        flight_data['flight_number']= (fake.country_code() + fake.building_number())[:5]
        flight_data['departure_time']= fake.time()
        flight_data['arrival_time']= fake.time()
        data.append(flight_data)
        flight_data_back ={}
        flight_data_back['flightId']= fake.unique.random_int(min=1, max=x**2)
        flight_data_back['departure_airport_townid']= arr_air_id
        flight_data_back['arrival_airport_townid']= dep_air_id
        flight_data_back['price']= fake.random_int(min=0, max=10000000)
        flight_data_back['flight_number']= (fake.country_code() + fake.building_number())[:5]
        flight_data_back['departure_time']= fake.time()
        flight_data_back['arrival_time']= fake.time()
        data.append(flight_data_back)

    with open('flight.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Hotel”:



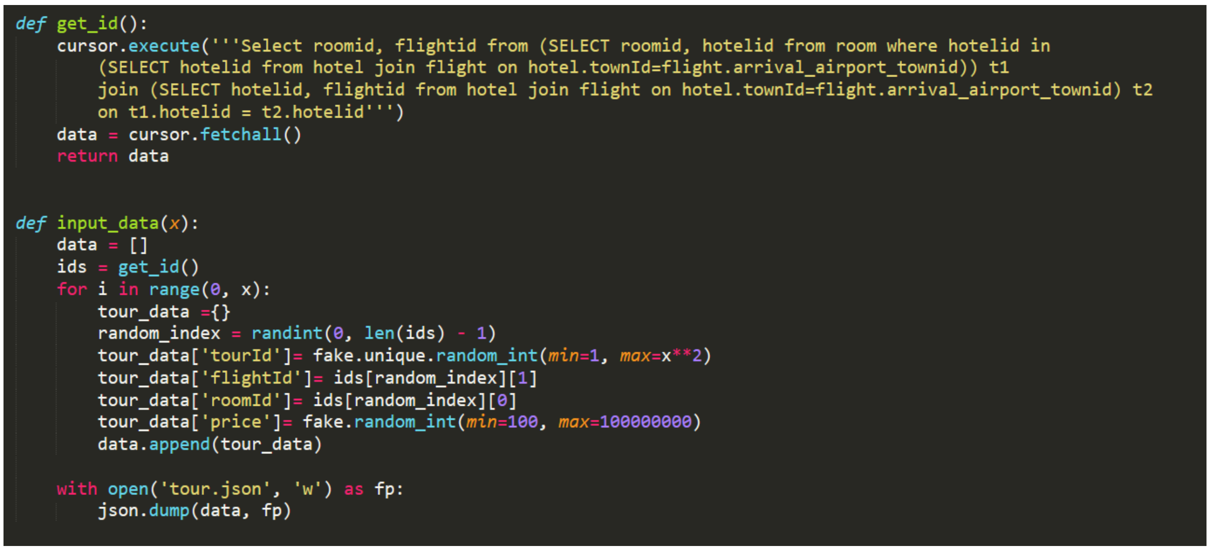
```python
def input_data(x):
    hotel_name_word_list = [
    'best','luxury','spa',
    'resort','sea','sunshine',
    'hotel','residence','inn',
    'b&b','wonderful','brilliant','old', 'central' ]
    data = []
    for i in range(0, x):
        hotel_data ={}
        hotel_data['hotelId']= fake.unique.random_int(min=1, max=x**2)
        hotel_data['townId']= fake.random_int(min=1, max=1760)
        hotel_data['name']= fake.sentence(ext_word_list=hotel_name_word_list)
        hotel_data['street']= fake.street_name()
        hotel_data['house']= fake.building_number()
        hotel_data['num_of_stars']= fake.random_int(min=0, max=5)
        hotel_data['all_inclusive_price']= fake.random_int(min=0, max=10000000)
        hotel_data['bb_price']= fake.random_int(min=0, max=100000)
        hotel_data['hb_price']= fake.random_int(min=0, max=1000000)
        hotel_data['fb_price']= fake.random_int(min=0, max=1000000)
        hotel_data['swimming_pool']= fake.boolean()
        hotel_data['parking']= fake.boolean()
        hotel_data['gymnasium']= fake.boolean()
        hotel_data['spa_center']= fake.boolean()
        hotel_data['free_wifi']= fake.boolean()
        hotel_data['private_beach']= fake.boolean()
        hotel_data['restaurant']= fake.boolean()
        hotel_data['golf_field']= fake.boolean()
        hotel_data['bar']= fake.boolean()
        data.append(hotel_data)

    with open('hotel.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Room”:



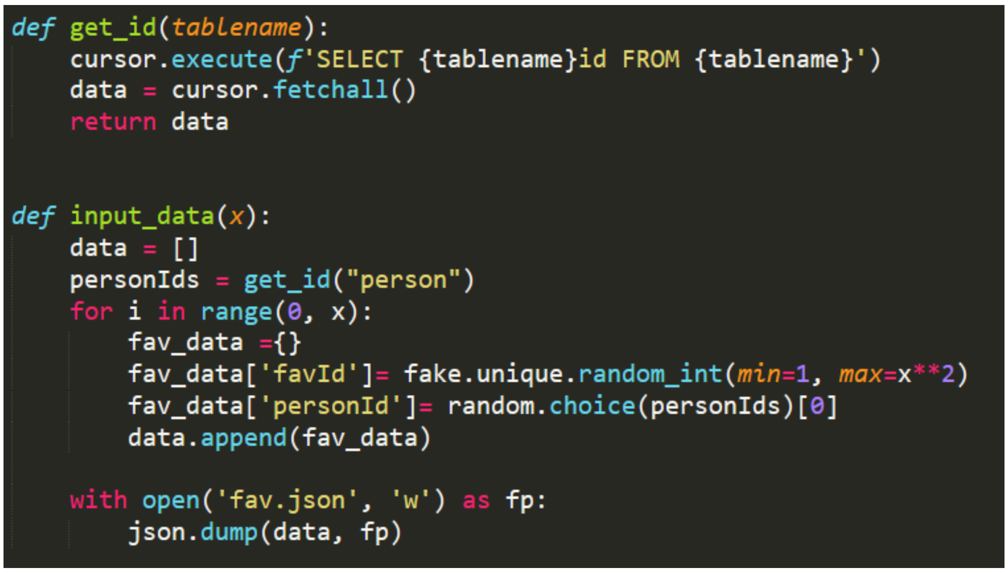
```python
def get_id(tablename):
    cursor.execute(f'SELECT {tablename}id FROM {tablename}')
    data = cursor.fetchall()
    return data


def input_data(x):
    data = []
    hotelIds = get_id("hotel")
    for i in range(0, x):
        room_data ={}
        room_data['roomId']= fake.unique.random_int(min=1, max=x**2)
        room_data['hotelId']= random.choice(hotelIds)[0]
        num_1 = fake.random_int(min=1, max=10)
        num_2 = fake.random_int(min=1, max=10)
        room_data['max_num_of_people']= max(num_1, num_2)
        room_data['num_of_beds']= max(min(num_1, num_2), max(num_1, num_2)//2)
        room_data['price']= fake.random_int(min=0, max=100000)
        room_data['num_of_rooms']= fake.random_int(min=1, max=10)
        room_data['air_condition']= fake.boolean()
        room_data['chimney']= fake.boolean()
        room_data['balcon']= fake.boolean()
        room_data['kitchen']= fake.boolean()
        room_data['private_bathroom']= fake.boolean()
        room_data['mini_bar']= fake.boolean()
        room_data['tea_coffee']= fake.boolean()
        room_data['tv']= fake.boolean()
        data.append(room_data)

    with open('room.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Tour”:



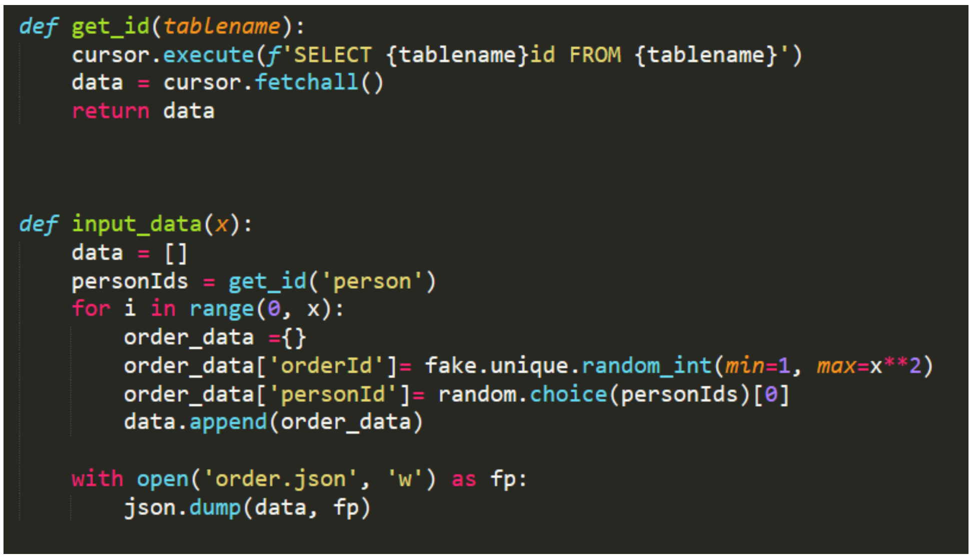
```python
def get_id():
    cursor.execute('''Select roomid, flightid from (SELECT roomid, hotelid from room where hotelid in
        (SELECT hotelid from hotel join flight on hotel.townId=flight.arrival_airport_townid)) t1
        join (SELECT hotelid, flightid from hotel join flight on hotel.townId=flight.arrival_airport_townid) t2
        on t1.hotelid = t2.hotelid''')
    data = cursor.fetchall()
    return data


def input_data(x):
    data = []
    ids = get_id()
    for i in range(0, x):
        tour_data ={}
        random_index = randint(0, len(ids) - 1)
        tour_data['tourId']= fake.unique.random_int(min=1, max=x**2)
        tour_data['flightId']= ids[random_index][1]
        tour_data['roomId']= ids[random_index][0]
        tour_data['price']= fake.random_int(min=100, max=100000000)
        data.append(tour_data)

    with open('tour.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Favourites”:



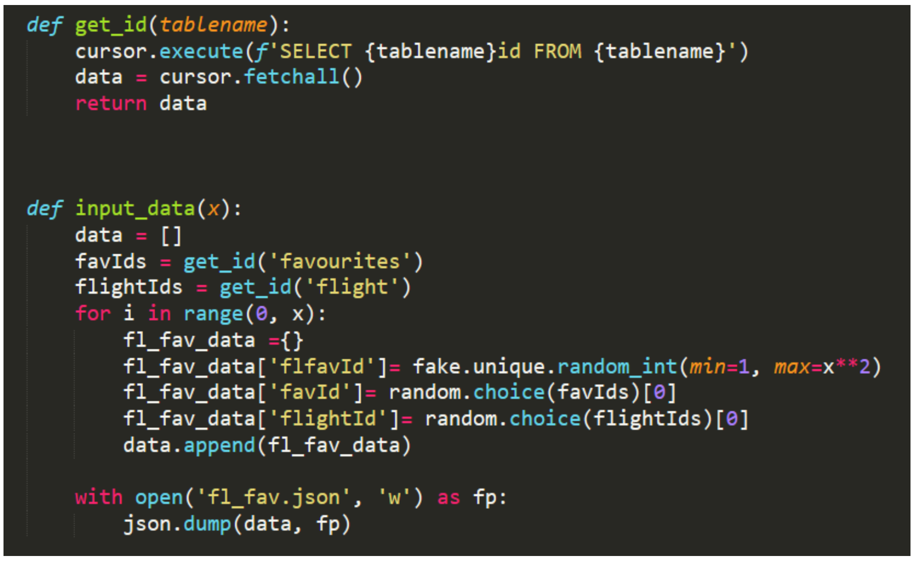
```python
def get_id(tablename):
    cursor.execute(f'SELECT {tablename}id FROM {tablename}')
    data = cursor.fetchall()
    return data


def input_data(x):
    data = []
    personIds = get_id("person")
    for i in range(0, x):
        fav_data ={}
        fav_data['favId']= fake.unique.random_int(min=1, max=x**2)
        fav_data['personId']= random.choice(personIds)[0]
        data.append(fav_data)

    with open('fav.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Order”:



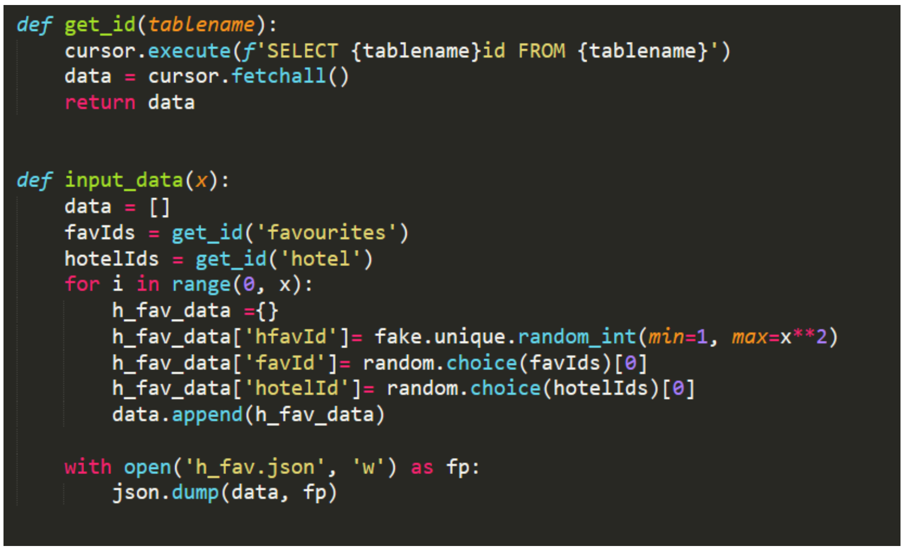
```python
def get_id(tablename):
    cursor.execute(f'SELECT {tablename}id FROM {tablename}')
    data = cursor.fetchall()
    return data



def input_data(x):
    data = []
    personIds = get_id('person')
    for i in range(0, x):
        order_data ={}
        order_data['orderId']= fake.unique.random_int(min=1, max=x**2)
        order_data['personId']= random.choice(personIds)[0]
        data.append(order_data)

    with open('order.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Flight_favourites”:



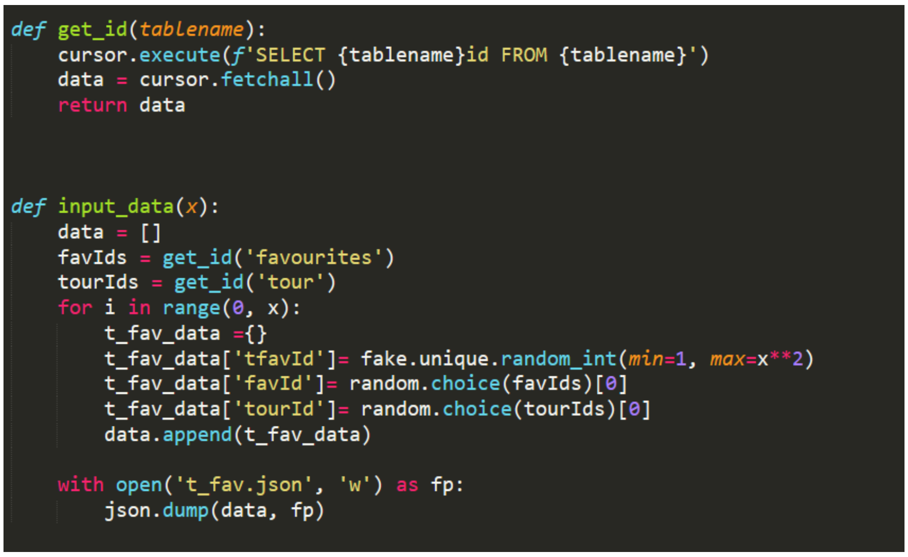
```python
def get_id(tablename):
    cursor.execute(f'SELECT {tablename}id FROM {tablename}')
    data = cursor.fetchall()
    return data



def input_data(x):
    data = []
    favIds = get_id('favourites')
    flightIds = get_id('flight')
    for i in range(0, x):
        fl_fav_data ={}
        fl_fav_data['flfavId']= fake.unique.random_int(min=1, max=x**2)
        fl_fav_data['favId']= random.choice(favIds)[0]
        fl_fav_data['flightId']= random.choice(flightIds)[0]
        data.append(fl_fav_data)

    with open('fl_fav.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Hotel_favourites”:



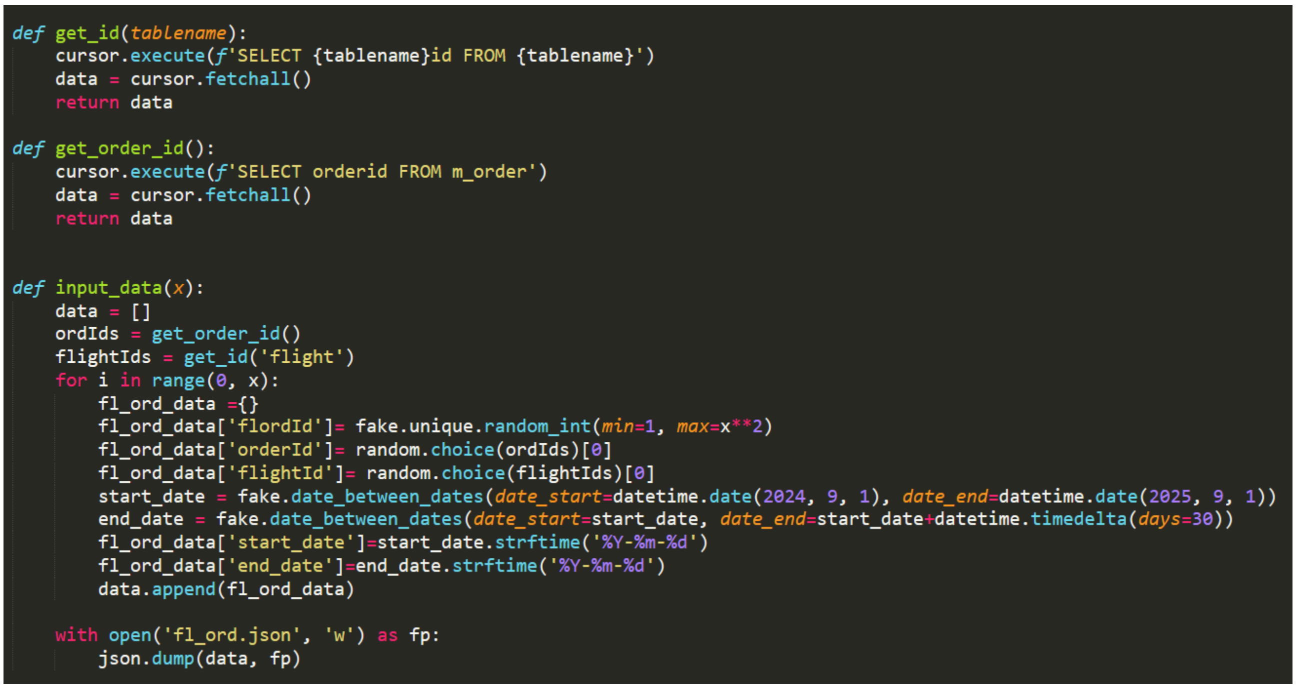
```python
def get_id(tablename):
    cursor.execute(f'SELECT {tablename}id FROM {tablename}')
    data = cursor.fetchall()
    return data


def input_data(x):
    data = []
    favIds = get_id('favourites')
    hotelIds = get_id('hotel')
    for i in range(0, x):
        h_fav_data ={}
        h_fav_data['hfavId']= fake.unique.random_int(min=1, max=x**2)
        h_fav_data['favId']= random.choice(favIds)[0]
        h_fav_data['hotelId']= random.choice(hotelIds)[0]
        data.append(h_fav_data)

    with open('h_fav.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Tour_favourites”:



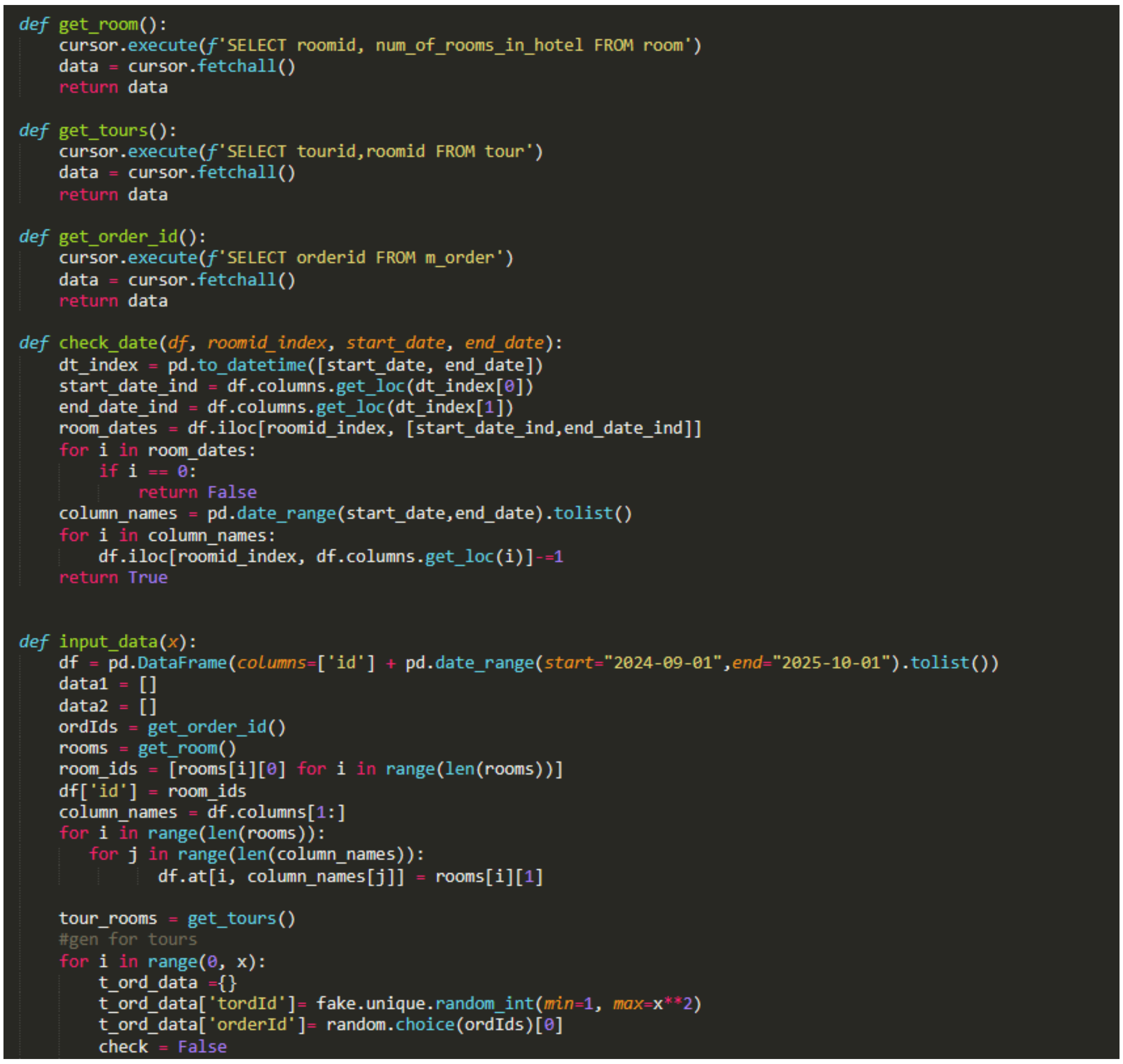
```python
def get_id(tablename):
    cursor.execute(f'SELECT {tablename}id FROM {tablename}')
    data = cursor.fetchall()
    return data



def input_data(x):
    data = []
    favIds = get_id('favourites')
    tourIds = get_id('tour')
    for i in range(0, x):
        t_fav_data ={}
        t_fav_data['tfavId']= fake.unique.random_int(min=1, max=x**2)
        t_fav_data['favId']= random.choice(favIds)[0]
        t_fav_data['tourId']= random.choice(tourIds)[0]
        data.append(t_fav_data)

    with open('t_fav.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Flight_order”:

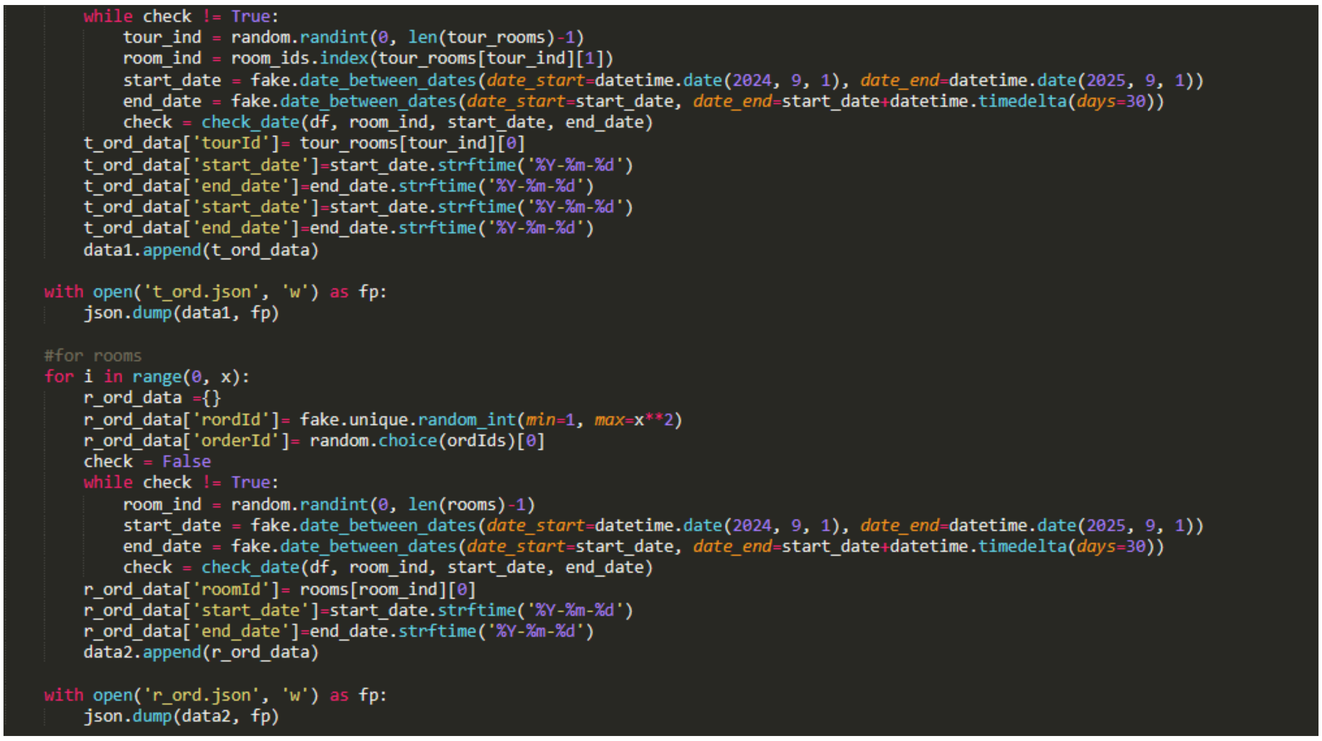


```python
def get_id(tablename):
    cursor.execute(f'SELECT {tablename}id FROM {tablename}')
    data = cursor.fetchall()
    return data

def get_order_id():
    cursor.execute(f'SELECT orderid FROM m_order')
    data = cursor.fetchall()
    return data


def input_data(x):
    data = []
    ordIds = get_order_id()
    flightIds = get_id('flight')
    for i in range(0, x):
        fl_ord_data ={}
        fl_ord_data['flordId']= fake.unique.random_int(min=1, max=x**2)
        fl_ord_data['orderId']= random.choice(ordIds)[0]
        fl_ord_data['flightId']= random.choice(flightIds)[0]
        start_date = fake.date_between_dates(date_start=datetime.date(2024, 9, 1), date_end=datetime.date(2025, 9, 1))
        end_date = fake.date_between_dates(date_start=start_date, date_end=start_date+datetime.timedelta(days=30))
        fl_ord_data['start_date']=start_date.strftime('%Y-%m-%d')
        fl_ord_data['end_date']=end_date.strftime('%Y-%m-%d')
        data.append(fl_ord_data)

    with open('fl_ord.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “Room_order” и “Tour_order”:

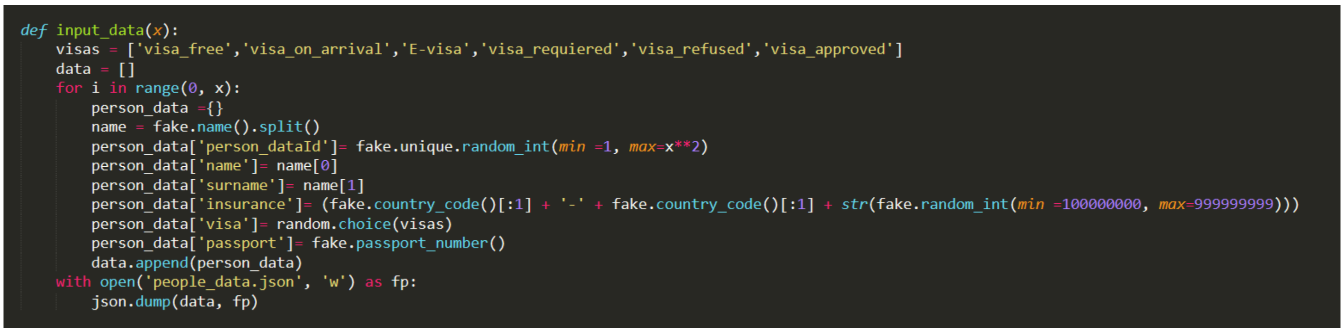


```python
def get_room():
    cursor.execute(f'SELECT roomid, num_of_rooms_in_hotel FROM room')
    data = cursor.fetchall()
    return data

def get_tours():
    cursor.execute(f'SELECT tourid,roomid FROM tour')
    data = cursor.fetchall()
    return data

def get_order_id():
    cursor.execute(f'SELECT orderid FROM m_order')
    data = cursor.fetchall()
    return data

def check_date(df, roomid_index, start_date, end_date):
    dt_index = pd.to_datetime([start_date, end_date])
    start_date_ind = df.columns.get_loc(dt_index[0])
    end_date_ind = df.columns.get_loc(dt_index[1])
    room_dates = df.iloc[roomid_index, [start_date_ind,end_date_ind]]
    for i in room_dates:
        if i == 0:
            return False
    column_names = pd.date_range(start_date,end_date).tolist()
    for i in column_names:
        df.iloc[roomid_index, df.columns.get_loc(i)]-=1
    return True


def input_data(x):
    df = pd.DataFrame(columns=['id'] + pd.date_range(start="2024-09-01",end="2025-10-01").tolist())
    data1 = []
    data2 = []
    ordIds = get_order_id()
    rooms = get_room()
    room_ids = [rooms[i][0] for i in range(len(rooms))]
    df['id'] = room_ids
    column_names = df.columns[1:]
    for i in range(len(rooms)):
       for j in range(len(column_names)):
              df.at[i, column_names[j]] = rooms[i][1]

    tour_rooms = get_tours()
    #gen for tours
    for i in range(0, x):
        t_ord_data ={}
        t_ord_data['tordId']= fake.unique.random_int(min=1, max=x**2)
        t_ord_data['orderId']= random.choice(ordIds)[0]
        check = False
```



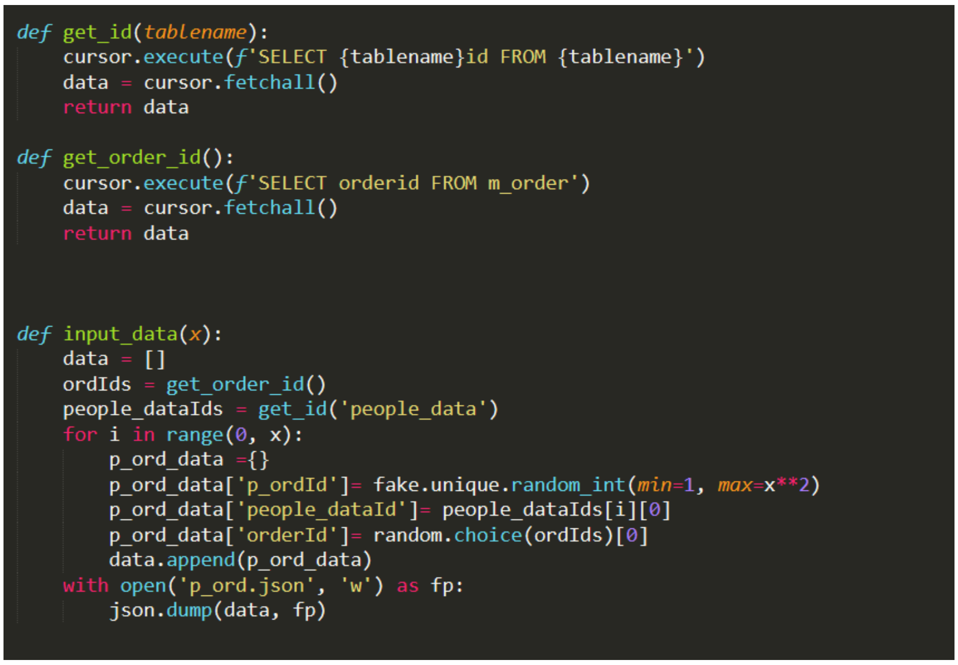
```python
        while check != True:
            tour_ind = random.randint(0, len(tour_rooms)-1)
            room_ind = room_ids.index(tour_rooms[tour_ind][1])
            start_date = fake.date_between_dates(date_start=datetime.date(2024, 9, 1), date_end=datetime.date(2025, 9, 1))
            end_date = fake.date_between_dates(date_start=start_date, date_end=start_date+datetime.timedelta(days=30))
            check = check_date(df, room_ind, start_date, end_date)
        t_ord_data['tourId']= tour_rooms[tour_ind][0]
        t_ord_data['start_date']=start_date.strftime('%Y-%m-%d')
        t_ord_data['end_date']=end_date.strftime('%Y-%m-%d')
        t_ord_data['start_date']=start_date.strftime('%Y-%m-%d')
        t_ord_data['end_date']=end_date.strftime('%Y-%m-%d')
        data1.append(t_ord_data)

    with open('t_ord.json', 'w') as fp:
        json.dump(data1, fp)

    #for rooms
    for i in range(0, x):
        r_ord_data ={}
        r_ord_data['rordId']= fake.unique.random_int(min=1, max=x**2)
        r_ord_data['orderId']= random.choice(ordIds)[0]
        check = False
        while check != True:
            room_ind = random.randint(0, len(rooms)-1)
            start_date = fake.date_between_dates(date_start=datetime.date(2024, 9, 1), date_end=datetime.date(2025, 9, 1))
            end_date = fake.date_between_dates(date_start=start_date, date_end=start_date+datetime.timedelta(days=30))
            check = check_date(df, room_ind, start_date, end_date)
        r_ord_data['roomId']= rooms[room_ind][0]
        r_ord_data['start_date']=start_date.strftime('%Y-%m-%d')
        r_ord_data['end_date']=end_date.strftime('%Y-%m-%d')
        data2.append(r_ord_data)

    with open('r_ord.json', 'w') as fp:
        json.dump(data2, fp)
```

- Код для заполнения таблицы “People_data”:



```python
def input_data(x):
    visas = ['visa_free','visa_on_arrival','E-visa','visa_requiered','visa_refused','visa_approved']
    data = []
    for i in range(0, x):
        person_data ={}
        name = fake.name().split()
        person_data['person_dataId']= fake.unique.random_int(min =1, max=x**2)
        person_data['name']= name[0]
        person_data['surname']= name[1]
        person_data['insurance']= (fake.country_code()[:1] + '-' + fake.country_code()[:1] + str(fake.random_int(min =100000000, max=999999999)))
        person_data['visa']= random.choice(visas)
        person_data['passport']= fake.passport_number()
        data.append(person_data)
    with open('people_data.json', 'w') as fp:
        json.dump(data, fp)
```

- Код для заполнения таблицы “People_data_order”:

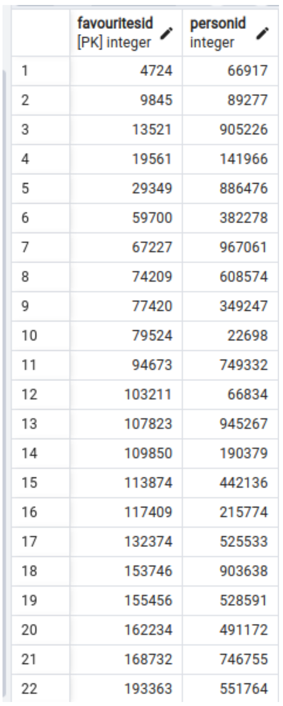


```python
def get_id(tablename):
    cursor.execute(f'SELECT {tablename}id FROM {tablename}')
    data = cursor.fetchall()
    return data

def get_order_id():
    cursor.execute(f'SELECT orderid FROM m_order')
    data = cursor.fetchall()
    return data



def input_data(x):
    data = []
    ordIds = get_order_id()
    people_dataIds = get_id('people_data')
    for i in range(0, x):
        p_ord_data ={}
        p_ord_data['p_ordId']= fake.unique.random_int(min=1, max=x**2)
        p_ord_data['people_dataId']= people_dataIds[i][0]
        p_ord_data['orderId']= random.choice(ordIds)[0]
        data.append(p_ord_data)
    with open('p_ord.json', 'w') as fp:
        json.dump(data, fp)
```

4.2.2 Результаты заполнения

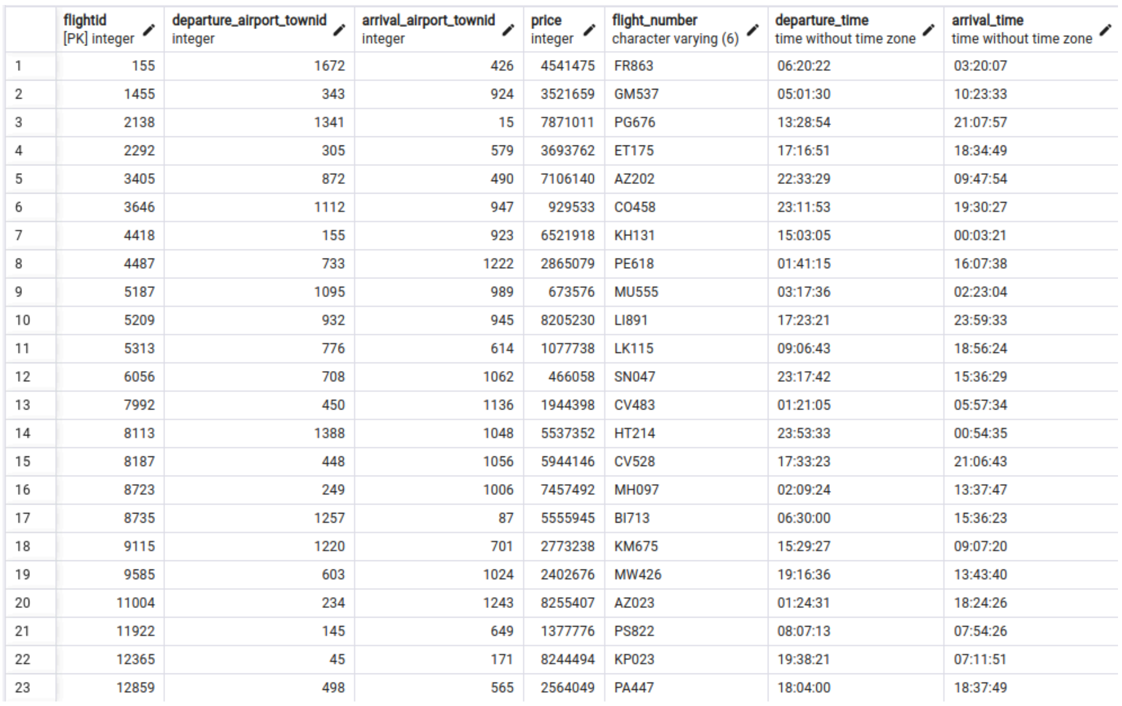
Далее представлены результаты работы программы на примере таблиц, соответствующих функциям, приведенным выше.

- Заполнение таблицы “Favourites”:



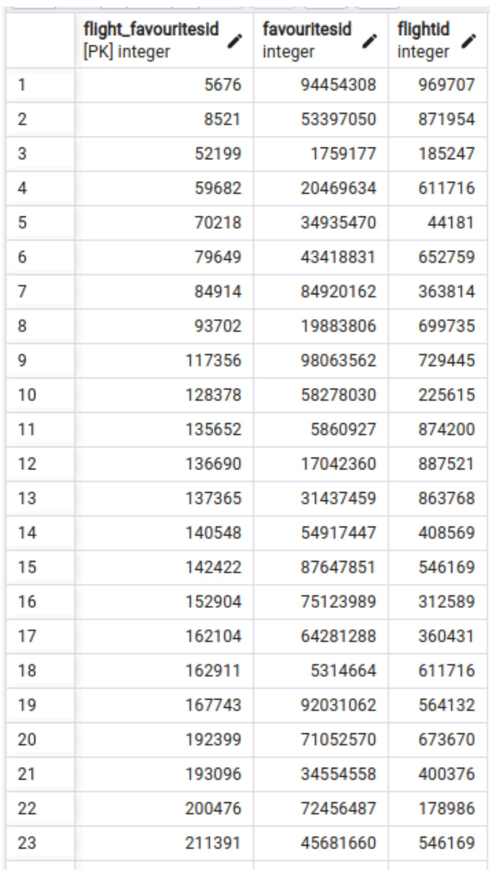
	favouritesid [PK] integer	personid integer
1	4724	66917
2	9845	89277
3	13521	905226
4	19561	141966
5	29349	886476
6	59700	382278
7	67227	967061
8	74209	608574
9	77420	349247
10	79524	22698
11	94673	749332
12	103211	66834
13	107823	945267
14	109850	190379
15	113874	442136
16	117409	215774
17	132374	525533
18	153746	903638
19	155456	528591
20	162234	491172
21	168732	746755
22	193363	551764

- Заполнение таблицы “Flight”:



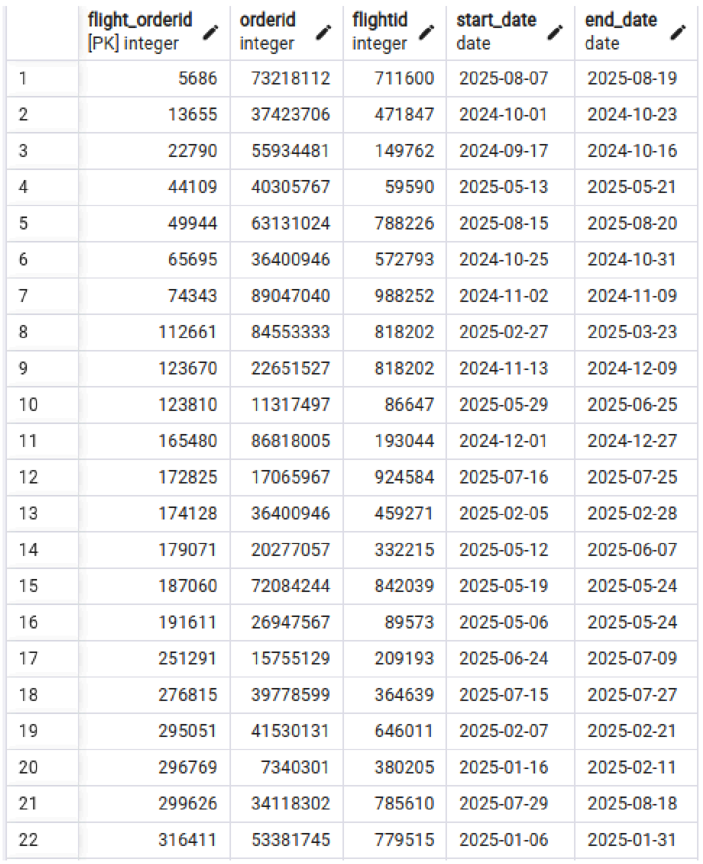
	flightid [PK] integer	departure_airport_townid integer	arrival_airport_townid integer	price integer	flight_number character varying (6)	departure_time time without time zone	arrival_time time without time zone
1	155	1672	426	4541475	FR863	06:20:22	03:20:07
2	1455	343	924	3521659	GM537	05:01:30	10:23:33
3	2138	1341	15	7871011	PG676	13:28:54	21:07:57
4	2292	305	579	3693762	ET175	17:16:51	18:34:49
5	3405	872	490	7106140	AZ202	22:33:29	09:47:54
6	3646	1112	947	929533	CO458	23:11:53	19:30:27
7	4418	155	923	6521918	KH131	15:03:05	00:03:21
8	4487	733	1222	2865079	PE618	01:41:15	16:07:38
9	5187	1095	989	673576	MU555	03:17:36	02:23:04
10	5209	932	945	8205230	LI891	17:23:21	23:59:33
11	5313	776	614	1077738	LK115	09:06:43	18:56:24
12	6056	708	1062	466058	SN047	23:17:42	15:36:29
13	7992	450	1136	1944398	CV483	01:21:05	05:57:34
14	8113	1388	1048	5537352	HT214	23:53:33	00:54:35
15	8187	448	1056	5944146	CV528	17:33:23	21:06:43
16	8723	249	1006	7457492	MH097	02:09:24	13:37:47
17	8735	1257	87	5555945	BI713	06:30:00	15:36:23
18	9115	1220	701	2773238	KM675	15:29:27	09:07:20
19	9585	603	1024	2402676	MW426	19:16:36	13:43:40
20	11004	234	1243	8255407	AZ023	01:24:31	18:24:26
21	11922	145	649	1377776	PS822	08:07:13	07:54:26
22	12365	45	171	8244494	KP023	19:38:21	07:11:51
23	12859	498	565	2564049	PA447	18:04:00	18:37:49

- Заполнение таблицы “Flight_favourites”:



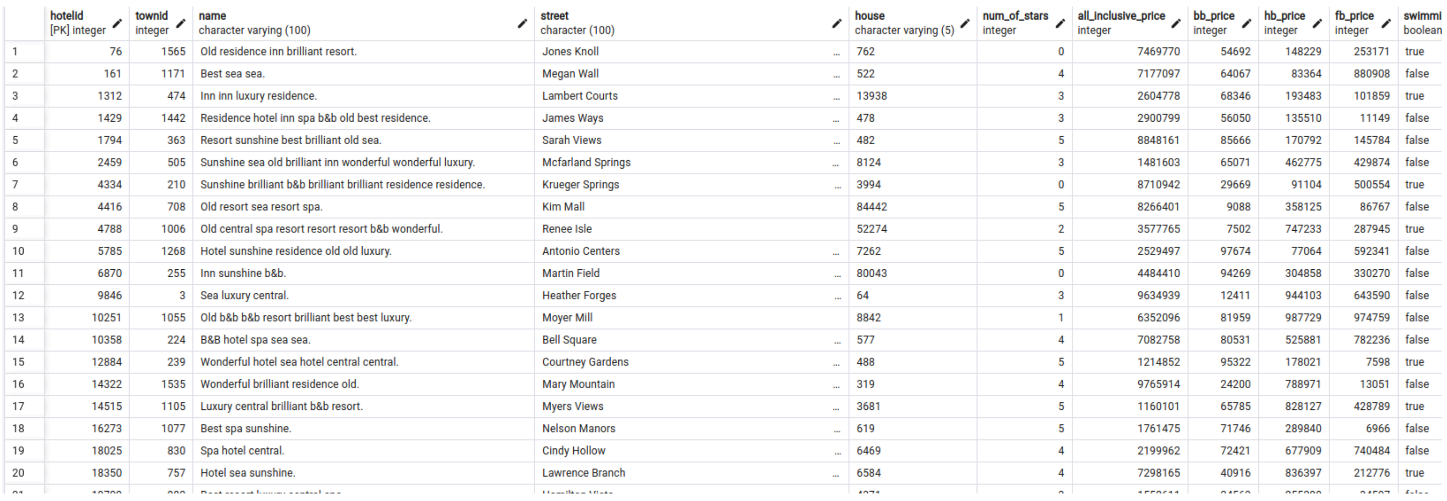
	flight_favouritesid [PK] integer	favouritesid integer	flightid integer
1	5676	94454308	969707
2	8521	53397050	871954
3	52199	1759177	185247
4	59682	20469634	611716
5	70218	34935470	44181
6	79649	43418831	652759
7	84914	84920162	363814
8	93702	19883806	699735
9	117356	98063562	729445
10	128378	58278030	225615
11	135652	5860927	874200
12	136690	17042360	887521
13	137365	31437459	863768
14	140548	54917447	408569
15	142422	87647851	546169
16	152904	75123989	312589
17	162104	64281288	360431
18	162911	5314664	611716
19	167743	92031062	564132
20	192399	71052570	673670
21	193096	34554558	400376
22	200476	72456487	178986
23	211391	45681660	546169

- Заполнение таблицы “Flight_order”:



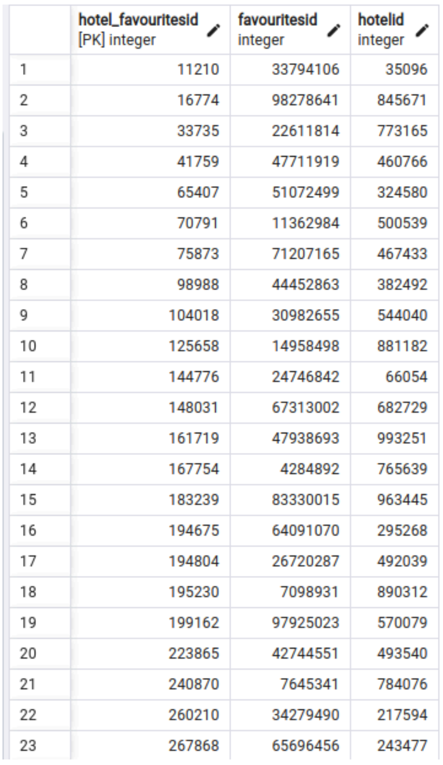
	flight_orderid [PK] integer	orderid integer	flightid integer	start_date date	end_date date
1	5686	73218112	711600	2025-08-07	2025-08-19
2	13655	37423706	471847	2024-10-01	2024-10-23
3	22790	55934481	149762	2024-09-17	2024-10-16
4	44109	40305767	59590	2025-05-13	2025-05-21
5	49944	63131024	788226	2025-08-15	2025-08-20
6	65695	36400946	572793	2024-10-25	2024-10-31
7	74343	89047040	988252	2024-11-02	2024-11-09
8	112661	84553333	818202	2025-02-27	2025-03-23
9	123670	22651527	818202	2024-11-13	2024-12-09
10	123810	11317497	86647	2025-05-29	2025-06-25
11	165480	86818005	193044	2024-12-01	2024-12-27
12	172825	17065967	924584	2025-07-16	2025-07-25
13	174128	36400946	459271	2025-02-05	2025-02-28
14	179071	20277057	332215	2025-05-12	2025-06-07
15	187060	72084244	842039	2025-05-19	2025-05-24
16	191611	26947567	89573	2025-05-06	2025-05-24
17	251291	15755129	209193	2025-06-24	2025-07-09
18	276815	39778599	364639	2025-07-15	2025-07-27
19	295051	41530131	646011	2025-02-07	2025-02-21
20	296769	7340301	380205	2025-01-16	2025-02-11
21	299626	34118302	785610	2025-07-29	2025-08-18
22	316411	53381745	779515	2025-01-06	2025-01-31

- Заполнение таблицы “Hotel”:



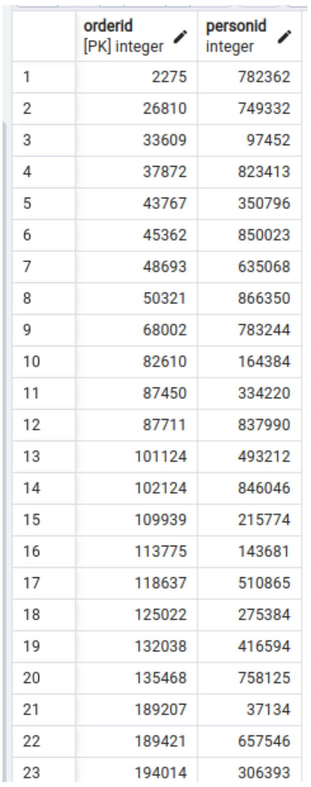
	hotelid [PK] integer	townid integer	name character varying (100)	street character (100)	house character varying (5)	num_of_stars integer	all_inclusive_price integer	bb_price integer	hb_price integer	fb_price integer	swimmi boolean
1	76	1565	Old residence inn brilliant resort.	Jones Knoll ...	762	0	7469770	54692	148229	253171	true
2	161	1171	Best sea sea.	Megan Wall ...	522	4	7177097	64067	83364	880908	false
3	1312	474	Inn inn luxury residence.	Lambert Courts ...	13938	3	2604778	68346	193483	101859	true
4	1429	1442	Residence hotel inn spa b&b old best residence.	James Ways ...	478	3	2900799	56050	135510	11149	false
5	1794	363	Resort sunshine best brilliant old sea.	Sarah Views ...	482	5	8848161	85666	170792	145784	false
6	2459	505	Sunshine sea old brilliant inn wonderful wonderful luxury.	Mcfarland Springs ...	8124	3	1481603	65071	462775	429874	false
7	4334	210	Sunshine brilliant b&b brilliant brilliant residence residence.	Krueger Springs ...	3994	0	8710942	29669	91104	500554	true
8	4416	708	Old resort sea resort spa.	Kim Mall	84442	5	8266401	9088	358125	86767	false
9	4788	1006	Old central spa resort resort resort b&b wonderful.	Renee Isle	52274	2	3577765	7502	747233	287945	true
10	5785	1268	Hotel sunshine residence old old luxury.	Antonio Centers ...	7262	5	2529497	97674	77064	592341	false
11	6870	255	Inn sunshine b&b.	Martin Field ...	80043	0	4484410	94269	304858	330270	false
12	9846	3	Sea luxury central.	Heather Forges ...	64	3	9634939	12411	944103	643590	false
13	10251	1055	Old b&b b&b resort brilliant best best luxury.	Moyer Mill	8842	1	6352096	81959	987729	974759	false
14	10358	224	B&B hotel spa sea sea.	Bell Square ...	577	4	7082758	80531	525881	782236	false
15	12884	239	Wonderful hotel sea hotel central central.	Courtney Gardens ...	488	5	1214852	95322	178021	7598	true
16	14322	1535	Wonderful brilliant residence old.	Mary Mountain ...	319	4	9765914	24200	788971	13051	false
17	14515	1105	Luxury central brilliant b&b resort.	Myers Views ...	3681	5	1160101	65785	828127	428789	true
18	16273	1077	Best spa sunshine.	Nelson Manors ...	619	5	1761475	71746	289840	6966	false
19	18025	830	Spa hotel central.	Cindy Hollow ...	6469	4	2199962	72421	677909	740484	false
20	18350	757	Hotel sea sunshine.	Lawrence Branch ...	6584	4	7298165	40916	836397	212776	true

- Заполнение таблицы “Hotel_favourites”:



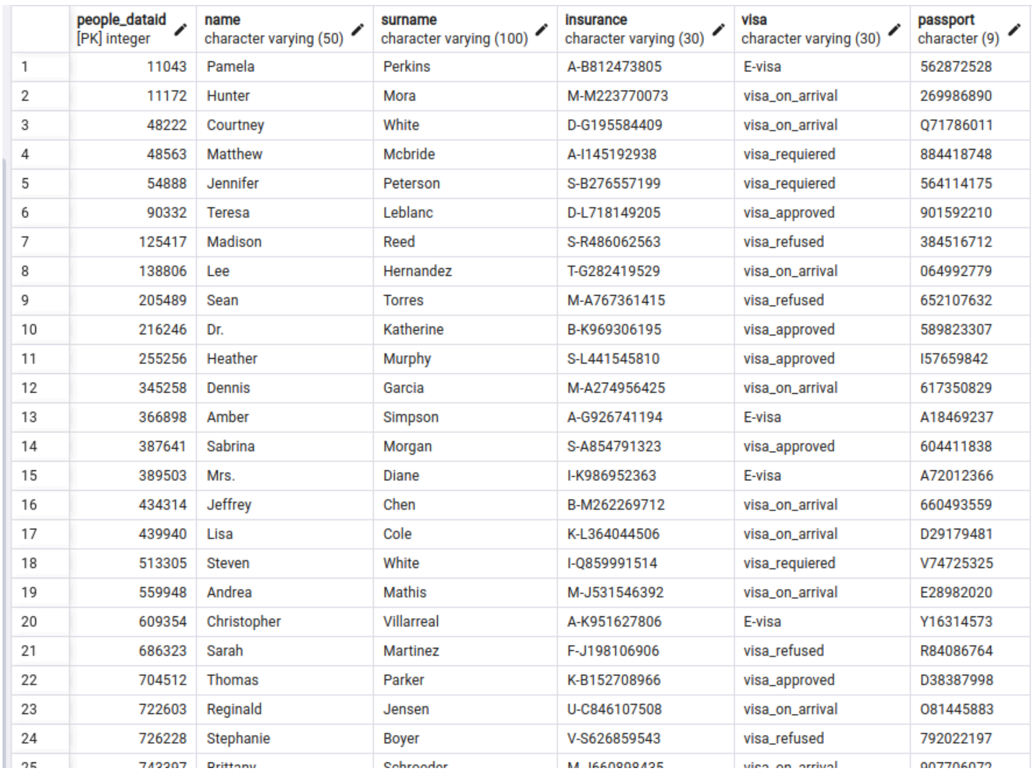
	hotel_favouritesid [PK] integer	favouritesid integer	hotelid integer
1	11210	33794106	35096
2	16774	98278641	845671
3	33735	22611814	773165
4	41759	47711919	460766
5	65407	51072499	324580
6	70791	11362984	500539
7	75873	71207165	467433
8	98988	44452863	382492
9	104018	30982655	544040
10	125658	14958498	881182
11	144776	24746842	66054
12	148031	67313002	682729
13	161719	47938693	993251
14	167754	4284892	765639
15	183239	83330015	963445
16	194675	64091070	295268
17	194804	26720287	492039
18	195230	7098931	890312
19	199162	97925023	570079
20	223865	42744551	493540
21	240870	7645341	784076
22	260210	34279490	217594
23	267868	65696456	243477

- Заполнение таблицы “Order”:



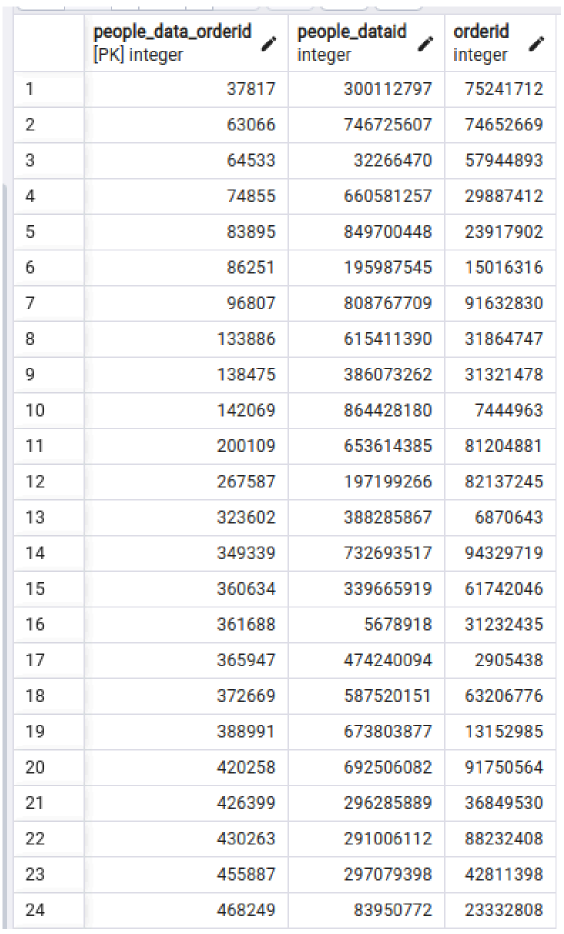
	orderid [PK] integer	personid integer
1	2275	782362
2	26810	749332
3	33609	97452
4	37872	823413
5	43767	350796
6	45362	850023
7	48693	635068
8	50321	866350
9	68002	783244
10	82610	164384
11	87450	334220
12	87711	837990
13	101124	493212
14	102124	846046
15	109939	215774
16	113775	143681
17	118637	510865
18	125022	275384
19	132038	416594
20	135468	758125
21	189207	37134
22	189421	657546
23	194014	306393

- Заполнение таблицы “People_data”:



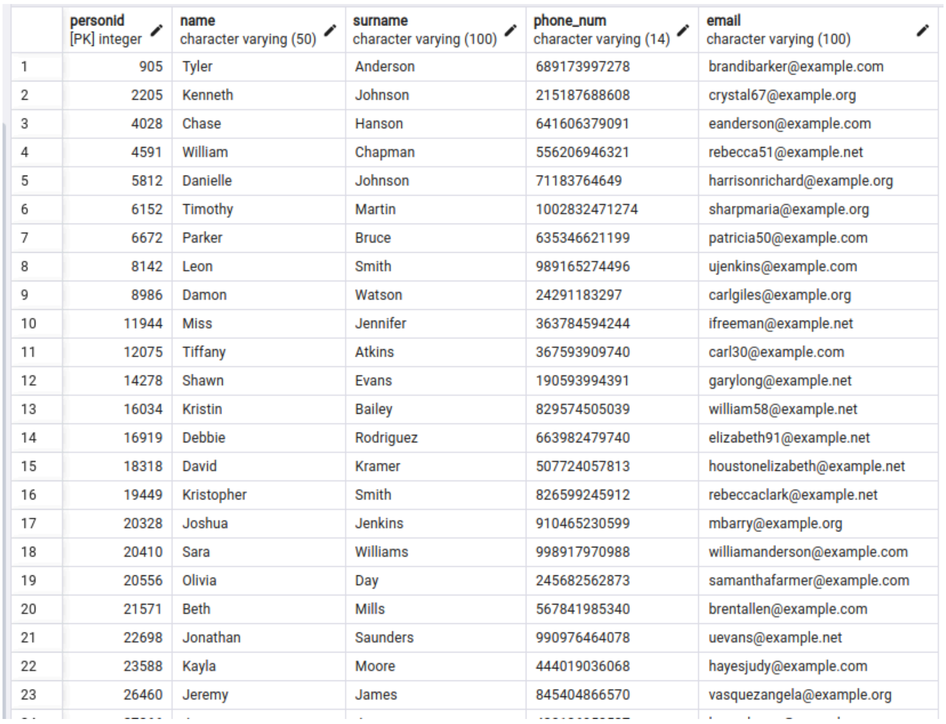
	people_dataid [PK] integer	name character varying (50)	surname character varying (100)	insurance character varying (30)	visa character varying (30)	passport character (9)
1	11043	Pamela	Perkins	A-B812473805	E-visa	562872528
2	11172	Hunter	Mora	M-M223770073	visa_on_arrival	269986890
3	48222	Courtney	White	D-G195584409	visa_on_arrival	Q71786011
4	48563	Matthew	Mcbride	A-I145192938	visa_requiered	884418748
5	54888	Jennifer	Peterson	S-B276557199	visa_requiered	564114175
6	90332	Teresa	Leblanc	D-L718149205	visa_approved	901592210
7	125417	Madison	Reed	S-R486062563	visa_refused	384516712
8	138806	Lee	Hernandez	T-G282419529	visa_on_arrival	064992779
9	205489	Sean	Torres	M-A767361415	visa_refused	652107632
10	216246	Dr.	Katherine	B-K969306195	visa_approved	589823307
11	255256	Heather	Murphy	S-L441545810	visa_approved	I57659842
12	345258	Dennis	Garcia	M-A274956425	visa_on_arrival	617350829
13	366898	Amber	Simpson	A-G926741194	E-visa	A18469237
14	387641	Sabrina	Morgan	S-A854791323	visa_approved	604411838
15	389503	Mrs.	Diane	I-K986952363	E-visa	A72012366
16	434314	Jeffrey	Chen	B-M262269712	visa_on_arrival	660493559
17	439940	Lisa	Cole	K-L364044506	visa_on_arrival	D29179481
18	513305	Steven	White	I-Q859991514	visa_requiered	V74725325
19	559948	Andrea	Mathis	M-J531546392	visa_on_arrival	E28982020
20	609354	Christopher	Villarreal	A-K951627806	E-visa	Y16314573
21	686323	Sarah	Martinez	F-J198106906	visa_refused	R84086764
22	704512	Thomas	Parker	K-B152708966	visa_approved	D38387998
23	722603	Reginald	Jensen	U-C846107508	visa_on_arrival	081445883
24	726228	Stephanie	Boyer	V-S626859543	visa_refused	792022197
25	743397	Brittany	Schroeder	M-J660898435	visa_on_arrival	907706072

- Заполнение таблицы “People_data_order”:



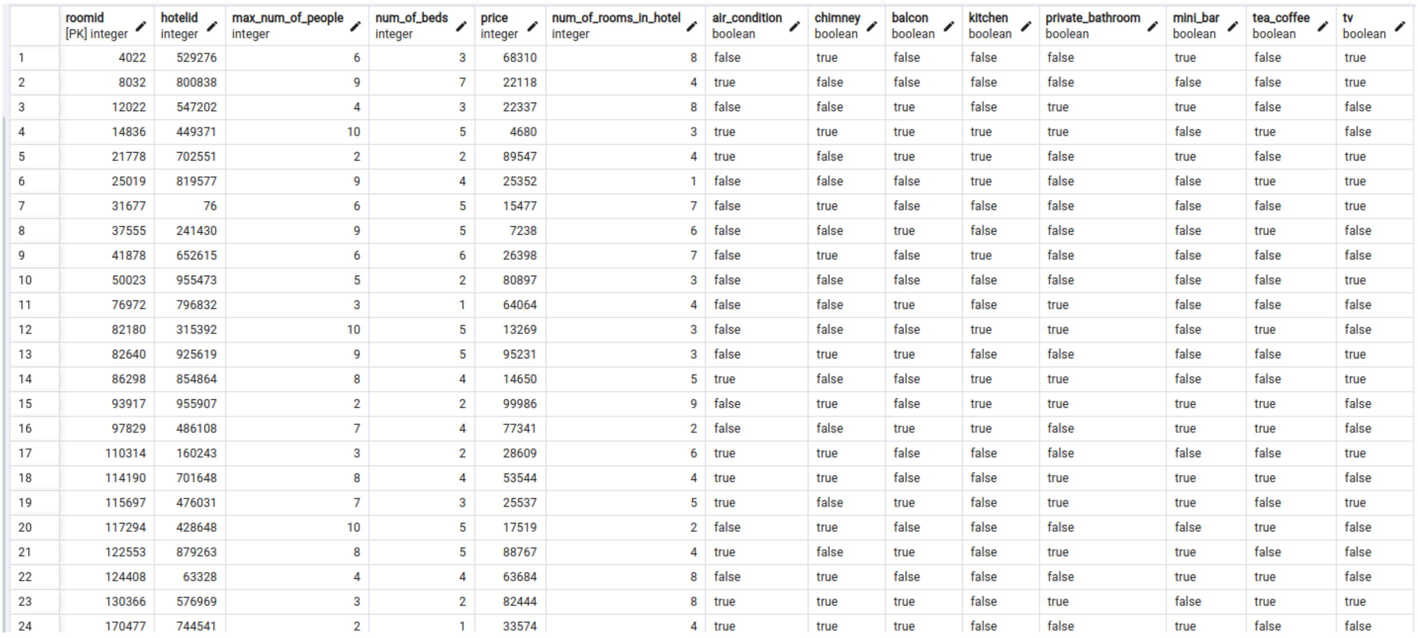
	people_data_orderid [PK] integer	people_dataid integer	orderid integer
1	37817	300112797	75241712
2	63066	746725607	74652669
3	64533	32266470	57944893
4	74855	660581257	29887412
5	83895	849700448	23917902
6	86251	195987545	15016316
7	96807	808767709	91632830
8	133886	615411390	31864747
9	138475	386073262	31321478
10	142069	864428180	7444963
11	200109	653614385	81204881
12	267587	197199266	82137245
13	323602	388285867	6870643
14	349339	732693517	94329719
15	360634	339665919	61742046
16	361688	5678918	31232435
17	365947	474240094	2905438
18	372669	587520151	63206776
19	388991	673803877	13152985
20	420258	692506082	91750564
21	426399	296285889	36849530
22	430263	291006112	88232408
23	455887	297079398	42811398
24	468249	83950772	23332808

- Заполнение таблицы “Person”:



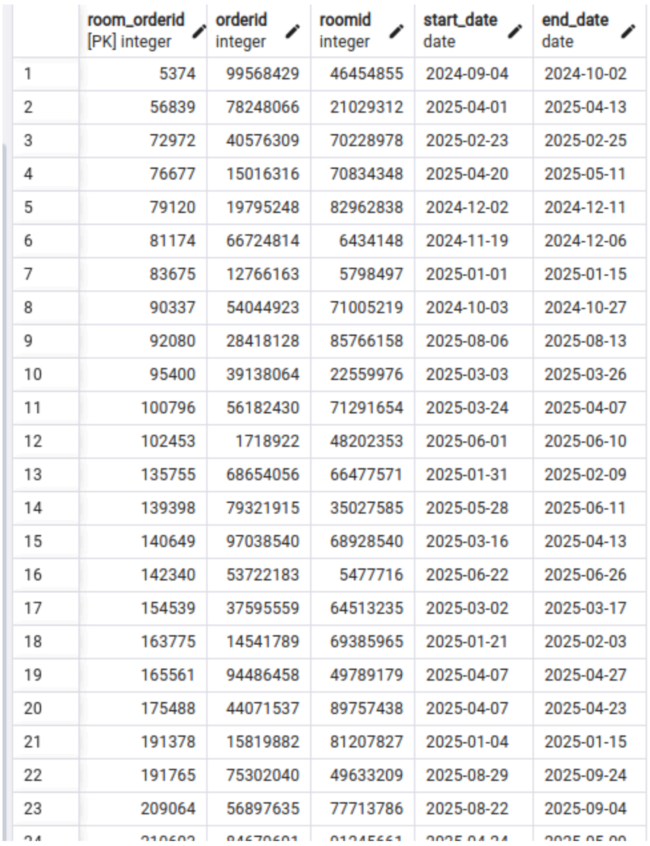
	personid [PK] integer	name character varying (50)	surname character varying (100)	phone_num character varying (14)	email character varying (100)
1	905	Tyler	Anderson	689173997278	brandibarker@example.com
2	2205	Kenneth	Johnson	215187688608	crystal67@example.org
3	4028	Chase	Hanson	641606379091	eanderson@example.com
4	4591	William	Chapman	556206946321	rebecca51@example.net
5	5812	Danielle	Johnson	71183764649	harrisonrichard@example.org
6	6152	Timothy	Martin	1002832471274	sharpmaria@example.org
7	6672	Parker	Bruce	635346621199	patricia50@example.com
8	8142	Leon	Smith	989165274496	ujenkins@example.com
9	8986	Damon	Watson	24291183297	carlgiles@example.org
10	11944	Miss	Jennifer	363784594244	ifreeman@example.net
11	12075	Tiffany	Atkins	367593909740	carl30@example.com
12	14278	Shawn	Evans	190593994391	garylong@example.net
13	16034	Kristin	Bailey	829574505039	william58@example.net
14	16919	Debbie	Rodriguez	663982479740	elizabeth91@example.net
15	18318	David	Kramer	507724057813	houstonelizabeth@example.net
16	19449	Kristopher	Smith	826599245912	rebeccaclark@example.net
17	20328	Joshua	Jenkins	910465230599	mbarry@example.org
18	20410	Sara	Williams	998917970988	williamanderson@example.com
19	20556	Olivia	Day	245682562873	samanthafarmer@example.com
20	21571	Beth	Mills	567841985340	brentallen@example.com
21	22698	Jonathan	Saunders	990976464078	uevans@example.net
22	23588	Kayla	Moore	444019036068	hayesjudy@example.com
23	26460	Jeremy	James	845404866570	vasquezangela@example.org

- Заполнение таблицы “Room”:



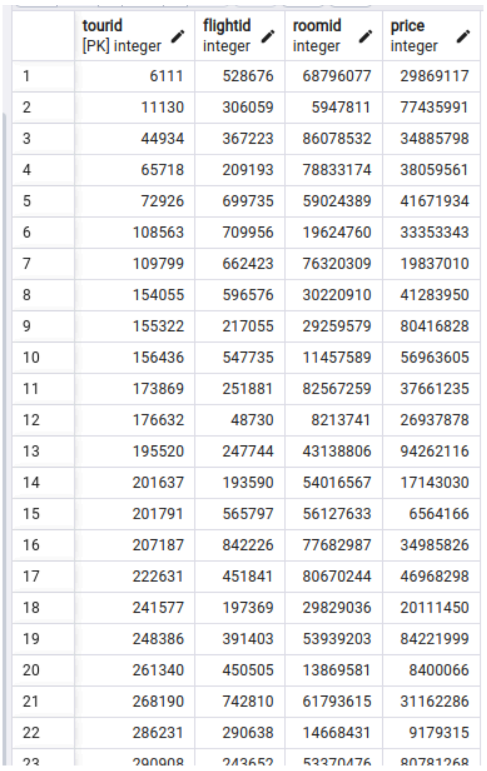
	roomid [PK] integer	hotelid integer	max_num_of_people integer	num_of_beds integer	price integer	num_of_rooms_in_hotel integer	air_condition boolean	chimney boolean	balcon boolean	kitchen boolean	private_bathroom boolean	mini_bar boolean	tea_coffee boolean	tv boolean
1	4022	529276	6	3	68310	8	false	true	false	false	false	true	false	true
2	8032	800838	9	7	22118	4	true	false	false	false	false	false	false	true
3	12022	547202	4	3	22337	8	false	false	true	false	true	true	false	false
4	14836	449371	10	5	4680	3	true	true	true	true	true	false	true	false
5	21778	702551	2	2	89547	4	true	false	true	true	false	true	false	true
6	25019	819577	9	4	25352	1	false	false	false	true	false	false	true	true
7	31677	76	6	5	15477	7	false	true	false	false	false	false	false	true
8	37555	241430	9	5	7238	6	false	false	true	false	false	false	true	false
9	41878	652615	6	6	26398	7	false	true	false	true	false	false	false	false
10	50023	955473	5	2	80897	3	false	false	false	false	false	false	false	true
11	76972	796832	3	1	64064	4	false	false	true	false	true	false	false	false
12	82180	315392	10	5	13269	3	false	false	false	true	true	false	true	false
13	82640	925619	9	5	95231	3	false	true	true	false	false	false	false	true
14	86298	854864	8	4	14650	5	true	false	false	true	true	false	false	true
15	93917	955907	2	2	99986	9	false	true	false	true	true	true	true	false
16	97829	486108	7	4	77341	2	false	false	true	true	false	true	true	false
17	110314	160243	3	2	28609	6	true	true	false	false	false	false	false	true
18	114190	701648	8	4	53544	4	true	true	false	false	true	true	true	false
19	115697	476031	7	3	25537	5	true	false	true	false	true	true	false	true
20	117294	428648	10	5	17519	2	false	true	false	false	false	false	true	false
21	122553	879263	8	5	88767	4	true	false	true	false	true	true	false	false
22	124408	63328	4	4	63684	8	false	true	false	false	false	true	true	false
23	130366	576969	3	2	82444	8	true	true	true	false	true	false	true	true
24	170477	744541	2	1	33574	4	true	true	true	false	false	true	false	false

- Заполнение таблицы “Room_order”:



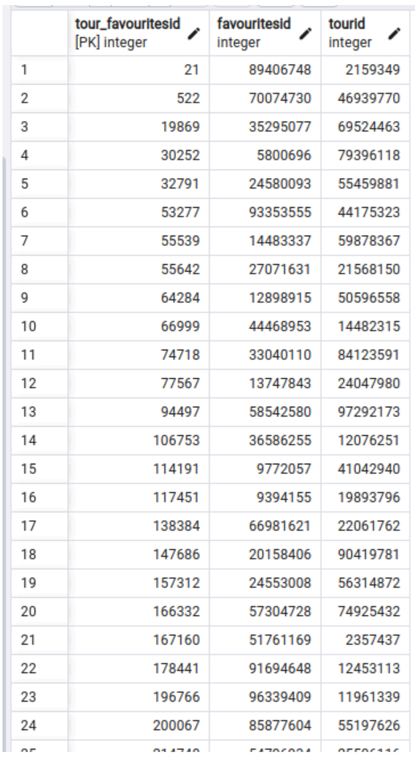
	room_orderid [PK] integer	orderid integer	roomid integer	start_date date	end_date date
1	5374	99568429	46454855	2024-09-04	2024-10-02
2	56839	78248066	21029312	2025-04-01	2025-04-13
3	72972	40576309	70228978	2025-02-23	2025-02-25
4	76677	15016316	70834348	2025-04-20	2025-05-11
5	79120	19795248	82962838	2024-12-02	2024-12-11
6	81174	66724814	6434148	2024-11-19	2024-12-06
7	83675	12766163	5798497	2025-01-01	2025-01-15
8	90337	54044923	71005219	2024-10-03	2024-10-27
9	92080	28418128	85766158	2025-08-06	2025-08-13
10	95400	39138064	22559976	2025-03-03	2025-03-26
11	100796	56182430	71291654	2025-03-24	2025-04-07
12	102453	1718922	48202353	2025-06-01	2025-06-10
13	135755	68654056	66477571	2025-01-31	2025-02-09
14	139398	79321915	35027585	2025-05-28	2025-06-11
15	140649	97038540	68928540	2025-03-16	2025-04-13
16	142340	53722183	5477716	2025-06-22	2025-06-26
17	154539	37595559	64513235	2025-03-02	2025-03-17
18	163775	14541789	69385965	2025-01-21	2025-02-03
19	165561	94486458	49789179	2025-04-07	2025-04-27
20	175488	44071537	89757438	2025-04-07	2025-04-23
21	191378	15819882	81207827	2025-01-04	2025-01-15
22	191765	75302040	49633209	2025-08-29	2025-09-24
23	209064	56897635	77713786	2025-08-22	2025-09-04
24	210602	84670601	91245661	2025-04-24	2025-05-09

- Заполнение таблицы “Tour”:



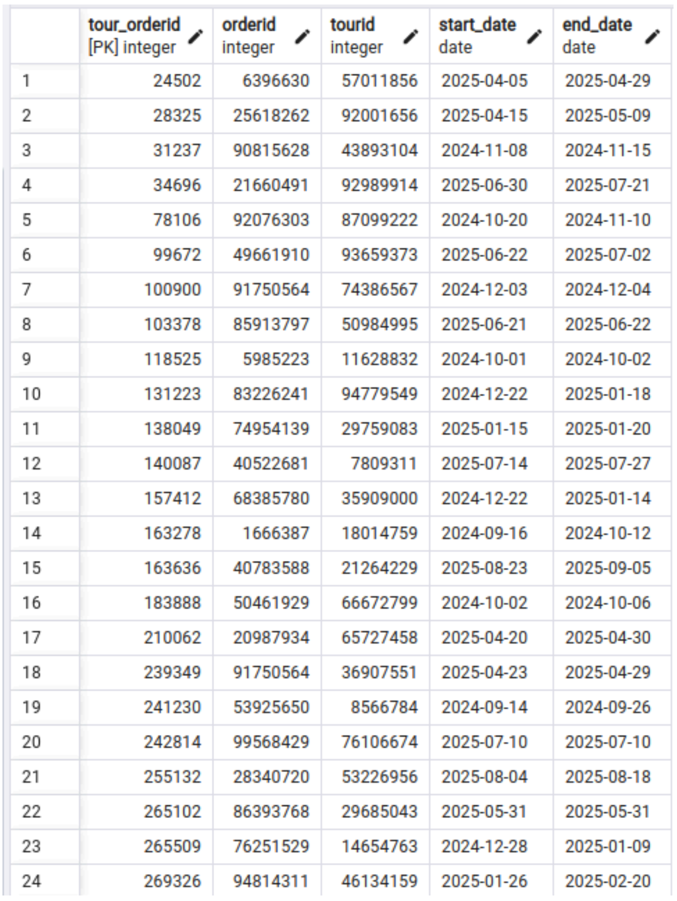
	tourid [PK] integer	flightid integer	roomid integer	price integer
1	6111	528676	68796077	29869117
2	11130	306059	5947811	77435991
3	44934	367223	86078532	34885798
4	65718	209193	78833174	38059561
5	72926	699735	59024389	41671934
6	108563	709956	19624760	33353343
7	109799	662423	76320309	19837010
8	154055	596576	30220910	41283950
9	155322	217055	29259579	80416828
10	156436	547735	11457589	56963605
11	173869	251881	82567259	37661235
12	176632	48730	8213741	26937878
13	195520	247744	43138806	94262116
14	201637	193590	54016567	17143030
15	201791	565797	56127633	6564166
16	207187	842226	77682987	34985826
17	222631	451841	80670244	46968298
18	241577	197369	29829036	20111450
19	248386	391403	53939203	84221999
20	261340	450505	13869581	8400066
21	268190	742810	61793615	31162286
22	286231	290638	14668431	9179315
23	290908	243652	53370476	80781268

● Заполнение таблицы “Tour_favourites”:



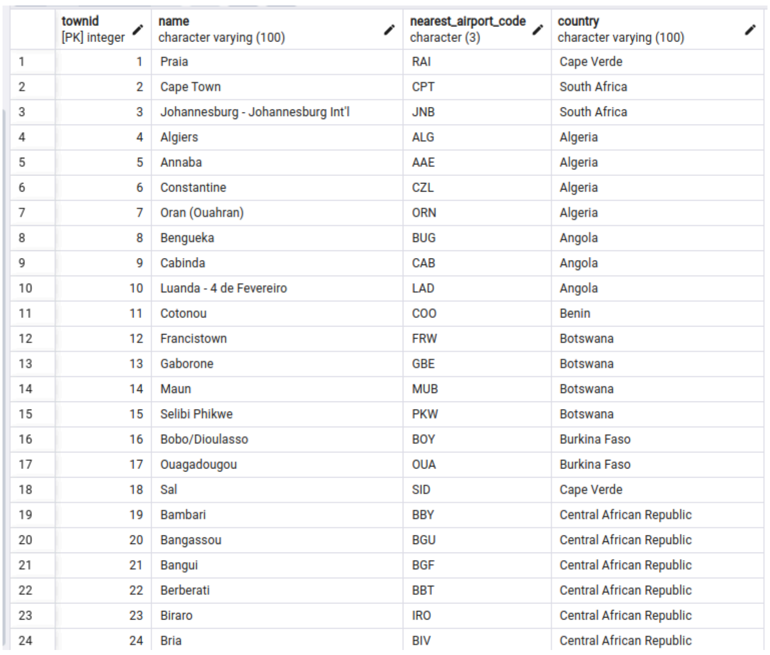
	tour_favouritesid [PK] integer	favouritesid integer	tourid integer
1	21	89406748	2159349
2	522	70074730	46939770
3	19869	35295077	69524463
4	30252	5800696	79396118
5	32791	24580093	55459881
6	53277	93353555	44175323
7	55539	14483337	59878367
8	55642	27071631	21568150
9	64284	12898915	50596558
10	66999	44468953	14482315
11	74718	33040110	84123591
12	77567	13747843	24047980
13	94497	58542580	97292173
14	106753	36586255	12076251
15	114191	9772057	41042940
16	117451	9394155	19893796
17	138384	66981621	22061762
18	147686	20158406	90419781
19	157312	24553008	56314872
20	166332	57304728	74925432
21	167160	51761169	2357437
22	178441	91694648	12453113
23	196766	96339409	11961339
24	200067	85877604	55197626
25	[illegible]	[illegible]	[illegible]

● Заполнение таблицы “Tour_order”:



	tour_orderid [PK] integer	orderid integer	tourid integer	start_date date	end_date date
1	24502	6396630	57011856	2025-04-05	2025-04-29
2	28325	25618262	92001656	2025-04-15	2025-05-09
3	31237	90815628	43893104	2024-11-08	2024-11-15
4	34696	21660491	92989914	2025-06-30	2025-07-21
5	78106	92076303	87099222	2024-10-20	2024-11-10
6	99672	49661910	93659373	2025-06-22	2025-07-02
7	100900	91750564	74386567	2024-12-03	2024-12-04
8	103378	85913797	50984995	2025-06-21	2025-06-22
9	118525	5985223	11628832	2024-10-01	2024-10-02
10	131223	83226241	94779549	2024-12-22	2025-01-18
11	138049	74954139	29759083	2025-01-15	2025-01-20
12	140087	40522681	7809311	2025-07-14	2025-07-27
13	157412	68385780	35909000	2024-12-22	2025-01-14
14	163278	1666387	18014759	2024-09-16	2024-10-12
15	163636	40783588	21264229	2025-08-23	2025-09-05
16	183888	50461929	66672799	2024-10-02	2024-10-06
17	210062	20987934	65727458	2025-04-20	2025-04-30
18	239349	91750564	36907551	2025-04-23	2025-04-29
19	241230	53925650	8566784	2024-09-14	2024-09-26
20	242814	99568429	76106674	2025-07-10	2025-07-10
21	255132	28340720	53226956	2025-08-04	2025-08-18
22	265102	86393768	29685043	2025-05-31	2025-05-31
23	265509	76251529	14654763	2024-12-28	2025-01-09
24	269326	94814311	46134159	2025-01-26	2025-02-20

- Заполнение таблицы “Town”:



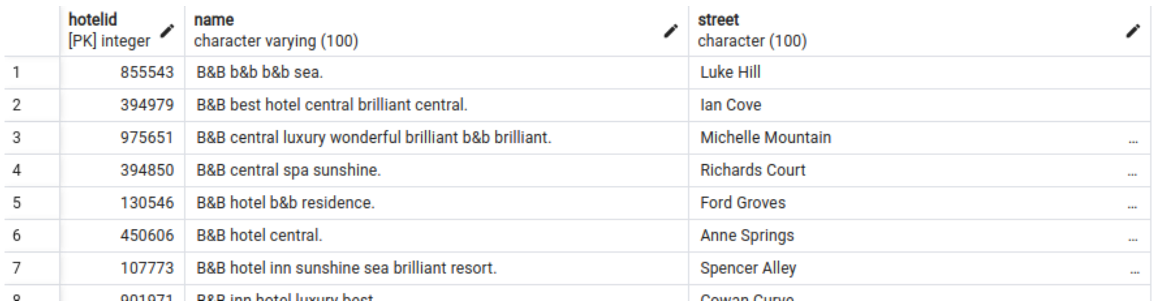
	townid [PK] integer	name character varying (100)	nearest_airport_code character (3)	country character varying (100)
1	1	Praia	RAI	Cape Verde
2	2	Cape Town	CPT	South Africa
3	3	Johannesburg - Johannesburg Int'l	JNB	South Africa
4	4	Algiers	ALG	Algeria
5	5	Annaba	AAE	Algeria
6	6	Constantine	CZL	Algeria
7	7	Oran (Ouahran)	ORN	Algeria
8	8	Bengueka	BUG	Angola
9	9	Cabinda	CAB	Angola
10	10	Luanda - 4 de Fevereiro	LAD	Angola
11	11	Cotonou	COO	Benin
12	12	Francistown	FRW	Botswana
13	13	Gaborone	GBE	Botswana
14	14	Maun	MUB	Botswana
15	15	Selibi Phikwe	PKW	Botswana
16	16	Bobo/Dioulasso	BOY	Burkina Faso
17	17	Ouagadougou	OUA	Burkina Faso
18	18	Sal	SID	Cape Verde
19	19	Bambari	BBY	Central African Republic
20	20	Bangassou	BGU	Central African Republic
21	21	Bangui	BGF	Central African Republic
22	22	Berberati	BBT	Central African Republic
23	23	Biraro	IRO	Central African Republic
24	24	Bria	BIV	Central African Republic

5. Выполнение запросов

В этом разделе приведены различные запросы к реализованной базе данных — их краткие описания, непосредственно запрос на языке SQL и результат выполнения.

1. Найти все отели в США с кондиционером

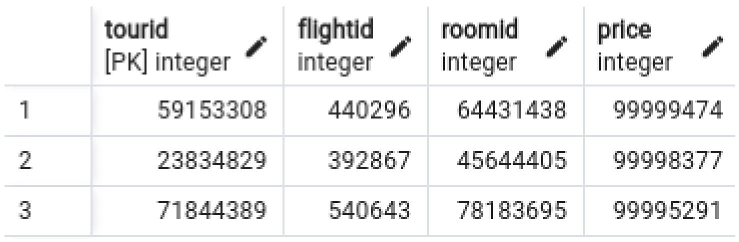
```sql
SELECT DISTINCT
    hotelid,
    name,
    street
FROM
    (SELECT *
    FROM
        hotel
        NATURAL JOIN
            (SELECT townid FROM town WHERE town.country='USA')
    ) t1
    NATURAL JOIN
    room
WHERE air_condition='true'
ORDER by name ASC;
```



	hotelid [PK] integer	name character varying (100)	street character (100)
1	855543	B&B b&b b&b sea.	Luke Hill
2	394979	B&B best hotel central brilliant central.	Ian Cove
3	975651	B&B central luxury wonderful brilliant b&b brilliant.	Michelle Mountain ...
4	394850	B&B central spa sunshine.	Richards Court ...
5	130546	B&B hotel b&b residence.	Ford Groves ...
6	450606	B&B hotel central.	Anne Springs ...
7	107773	B&B hotel inn sunshine sea brilliant resort.	Spencer Alley ...
8	901971	B&B inn hotel luxury host	Cowan Curve

2. Найти три самых дорогих тура

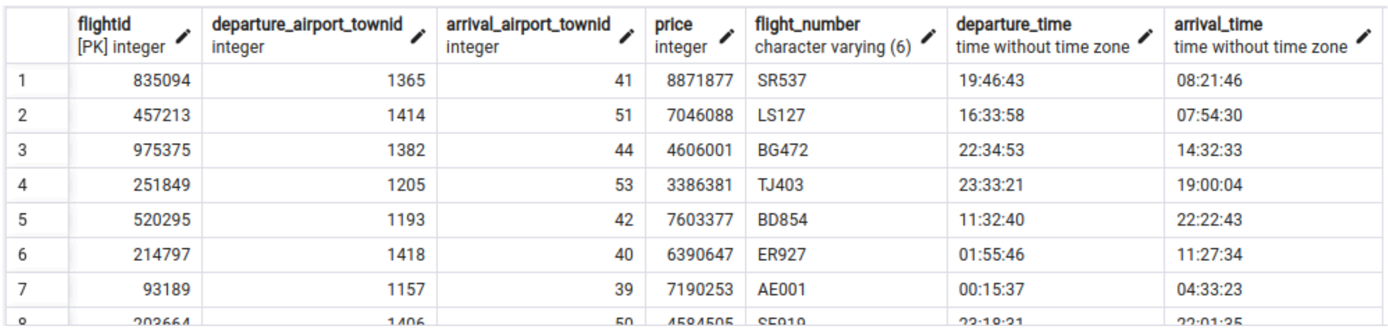
```sql
SELECT *
FROM
    tour
ORDER BY price DESC LIMIT 3;
```



	tourid [PK] integer	flightid integer	roomid integer	price integer
1	59153308	440296	64431438	99999474
2	23834829	392867	45644405	99998377
3	71844389	540643	78183695	99995291

3. Найти все перелеты из США в Египет

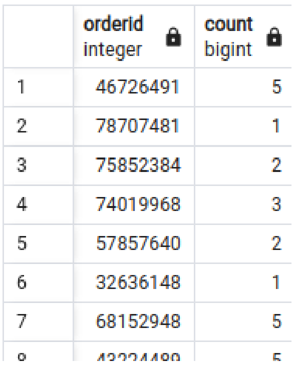
```
SELECT *
FROM
    flight
WHERE
    flight.departure_airport_townid IN (SELECT townid FROM town WHERE
town.country='USA')
    AND flight.arrival_airport_townid IN (SELECT townid FROM town WHERE
town.country='Egypt');
```



	flightid [PK] integer	departure_airport_townid integer	arrival_airport_townid integer	price integer	flight_number character varying (6)	departure_time time without time zone	arrival_time time without time zone
1	835094	1365	41	8871877	SR537	19:46:43	08:21:46
2	457213	1414	51	7046088	LS127	16:33:58	07:54:30
3	975375	1382	44	4606001	BG472	22:34:53	14:32:33
4	251849	1205	53	3386381	TJ403	23:33:21	19:00:04
5	520295	1193	42	7603377	BD854	11:32:40	22:22:43
6	214797	1418	40	6390647	ER927	01:55:46	11:27:34
7	93189	1157	39	7190253	AE001	00:15:37	04:33:23
8	203664	1406	50	4584505	SE919	23:18:31	22:01:35

4. Найти сколько человек связано с заказом

```
SELECT
    orderid,
    COUNT(*)
FROM
    people_data_order GROUP BY orderid;
```

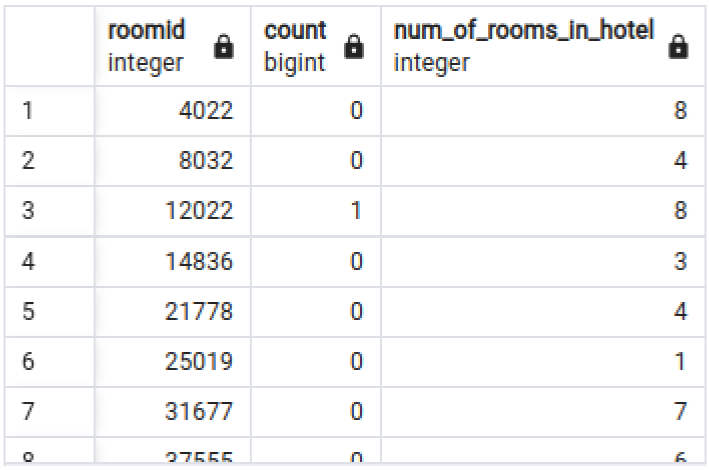


	orderid integer	count bigint
1	46726491	5
2	78707481	1
3	75852384	2
4	74019968	3
5	57857640	2
6	32636148	1
7	68152948	5
8	43224489	5

5. Найти все свободные номера в период между 2025.07.08 и 2025.08.09

```
WITH t1 AS (SELECT
                roomid
            FROM
                room_order
            WHERE (start_date, end_date) OVERLAPS ('2025-07-08'::date,
                                                   '2025-08-09'::date)
                                                  ),
t2 AS (SELECT
             roomid
        FROM
            (SELECT *
            FROM tour_order NATURAL JOIN tour)
        WHERE (start_date, end_date) OVERLAPS ('2025-07-08'::date,
                                               '2025-08-09'::date)
                                              ),
t3 AS (SELECT * FROM t1 UNION ALL SELECT * FROM t2),
t4 AS (SELECT
            roomid,
            COUNT(*) AS count
        FROM t3 GROUP BY roomid),
t5 AS (SELECT coalesce(t4.roomid,room.roomid) AS roomid,
              coalesce(count ,0) AS count,
              num_of_rooms_in_hotel FROM t4 RIGHT JOIN room ON room.roomid=
t4.roomid)
SELECT * FROM t5
```

```
WHERE count < num_of_rooms_in_hotel
ORDER BY roomid;
```

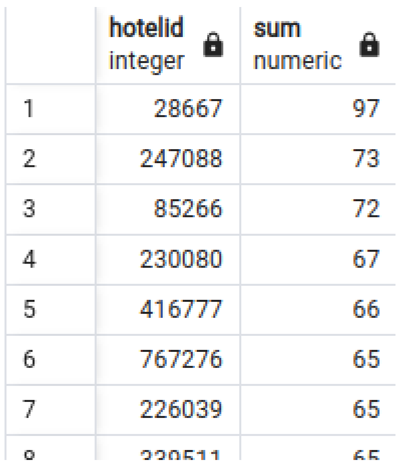


	roomid integer	count bigint	num_of_rooms_in_hotel integer
1	4022	0	8
2	8032	0	4
3	12022	1	8
4	14836	0	3
5	21778	0	4
6	25019	0	1
7	31677	0	7
8	37555	0	6

6. Найти какие отели бронировали наибольшее число раз

```
WITH t1 AS (SELECT
                roomid
            FROM room_order),
t2 AS (SELECT
            roomid
            FROM (SELECT *
                 FROM tour_order NATURAL JOIN tour)
       ),
t3 AS (SELECT * FROM t1 UNION ALL SELECT * FROM t2),
t4 AS (SELECT
            roomid,
            COUNT(*) AS count
        FROM t3 GROUP BY roomid),
t5 AS (SELECT
            hotelid,
            coalesce(count ,0) AS c
        FROM t4 RIGHT JOIN room ON room.roomid= t4.roomid)
SELECT
    hotelid,
    SUM(c) AS sum
```

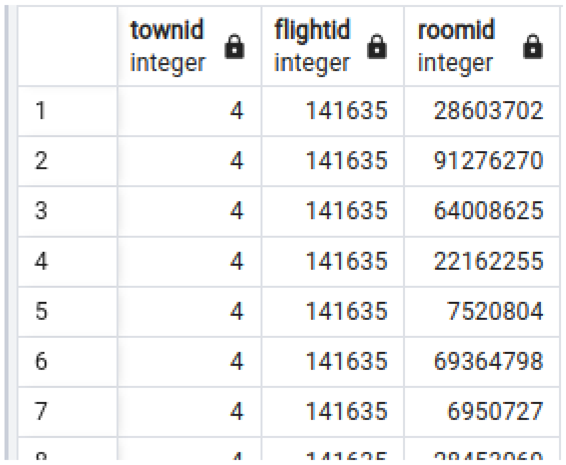
```
FROM t5
GROUP BY hotelid ORDER BY sum DESC;
```



	hotelid integer	sum numeric
1	28667	97
2	247088	73
3	85266	72
4	230080	67
5	416777	66
6	767276	65
7	226039	65
8	229511	65

7. Найти все перелеты в конкретный город и все номера в этом городе

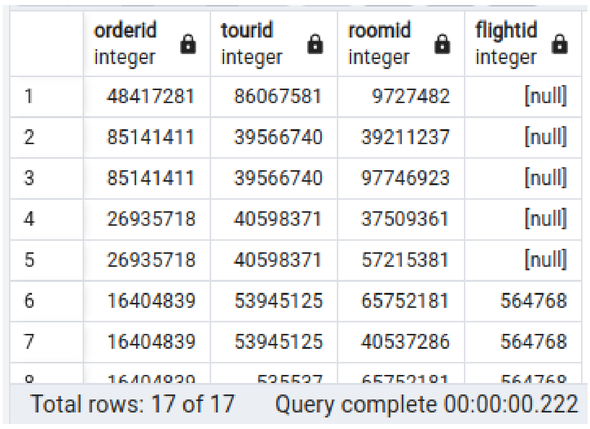
```
SELECT
    townid,
    flightid,
    roomid
FROM (SELECT
            roomid,
            hotelid
      FROM room
      WHERE hotelid IN (SELECT
                                hotelid
                            FROM hotel JOIN flight ON
hotel.townId=flight.arrival_airport_townid)
      )
NATURAL JOIN
(SELECT
    hotelid,
    flightid,
    townid
FROM hotel JOIN flight ON hotel.townId=flight.arrival_airport_townid)
ORDER BY townid;
```



	townid integer	flightid integer	roomid integer
1	4	141635	28603702
2	4	141635	91276270
3	4	141635	64008625
4	4	141635	22162255
5	4	141635	7520804
6	4	141635	69364798
7	4	141635	6950727
8	4	141635	28452060

8. Найти заказы перелетов, отелей и туров конкретного пользователя

```
WITH t0 AS (SELECT *
            FROM m_order WHERE personid='97452'),
t1 AS (SELECT *
       FROM t0 NATURAL JOIN room_order),
t2 AS (SELECT *
       FROM t0 NATURAL JOIN tour_order),
t3 AS (SELECT *
       FROM t0 NATURAL JOIN flight_order),
t4 AS (SELECT
            coalesce(t1.orderid, t2.orderid) AS orderid,
            tourid,
            roomid
        FROM t2 FULL OUTER JOIN t1 ON t1.orderid=t2.orderid)
SELECT
    coalesce(t3.orderid, t4.orderid) AS orderid,
    tourid,
    roomid,
    flightid
FROM t3 FULL OUTER JOIN t4 ON t3.orderid=t4.orderid;
```



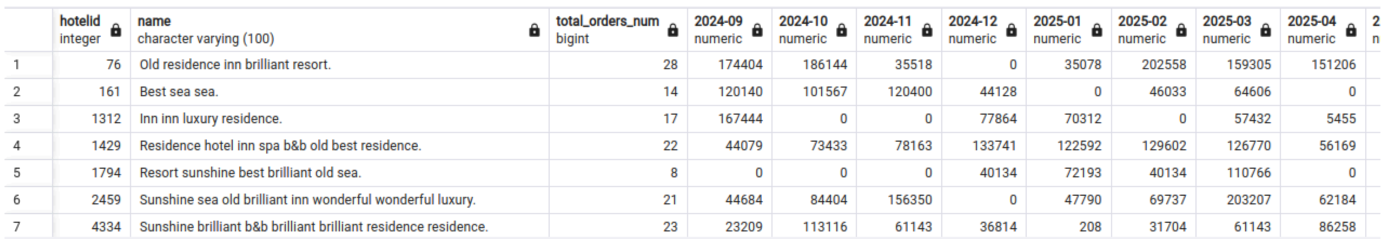
	orderid integer	tourid integer	roomid integer	flightid integer
1	48417281	86067581	9727482	[null]
2	85141411	39566740	39211237	[null]
3	85141411	39566740	97746923	[null]
4	26935718	40598371	37509361	[null]
5	26935718	40598371	57215381	[null]
6	16404839	53945125	65752181	564768
7	16404839	53945125	40537286	564768
8	16404839	535537	65752181	564768

Total rows: 17 of 17 Query complete 00:00:00.222

9. Для каждого отеля посчитать общее число бронирований за год и доход отеля в каждом месяце

```
WITH t1 AS (SELECT
                roomid,
                start_date,
                end_date
            FROM room_order),
t2 AS (SELECT
            roomid,
            start_date,
            end_date
        FROM (SELECT *
            FROM tour_order NATURAL JOIN tour)
        ),
t3 AS (SELECT * FROM t1 UNION ALL SELECT * FROM t2),
--t4-t5 выделение в полученной таблице месяца и года и соединение ее с отелем
t4 AS (SELECT
            roomid,
            DATE_PART('month',start_date) AS month,
            date_part('year', start_date) AS year,
            start_date
        FROM t3),
t5 AS (SELECT
            hotelid,
            roomid,
            month,
            year,
            price
        FROM t4 NATURAL JOIN room),
--t6 подсчет числа заказов за год
t6 AS (SELECT
            hotelid,
            name,
            COUNT(*) AS total_orders_num
        FROM t5 NATURAL JOIN hotel
        GROUP BY hotelid, name),
--t7 расчет выручки по месяцам для каждой комнаты
t7 AS (SELECT
            roomid, hotelid,
            SUM(CASE WHEN month = 9 AND YEAR = 2024 THEN price ELSE 0 END)
                AS "2024-09",
            SUM(CASE WHEN month = 10 AND YEAR = 2024 THEN price ELSE 0 END)
                AS "2024-10",
            SUM(CASE WHEN month = 11 AND YEAR = 2024 THEN price ELSE 0 END)
                AS "2024-11",
            SUM(CASE WHEN month = 12 AND YEAR = 2024 THEN price ELSE 0 END)
                AS "2024-12",
            SUM(CASE WHEN month = 1 AND YEAR = 2025 THEN price ELSE 0 END)
                AS "2025-01",
            SUM(CASE WHEN month = 2 AND YEAR = 2025 THEN price ELSE 0 END)
                AS "2025-02",
            SUM(CASE WHEN month = 3 AND YEAR = 2025 THEN price ELSE 0 END)
                AS "2025-03",
            SUM(CASE WHEN month = 4 AND YEAR = 2025 THEN price ELSE 0 END)
                AS "2025-04",
            SUM(CASE WHEN month = 5 AND YEAR = 2025 THEN price ELSE 0 END)
                AS "2025-05",
            SUM(CASE WHEN month = 6 AND YEAR = 2025 THEN price ELSE 0 END)
```

```
                AS "2025-06",
            SUM(CASE WHEN month = 7 AND YEAR = 2025 THEN price ELSE 0 END)
                AS "2025-07",
            SUM(CASE WHEN month = 8 AND YEAR = 2025 THEN price ELSE 0 END)
                AS "2025-08"
        FROM t5
        GROUP BY roomid, hotelid)
SELECT
            hotelid, name, total_orders_num,
            SUM("2024-09") AS "2024-09",
            SUM("2024-10") AS "2024-10",
            SUM("2024-11") AS "2024-11",
            SUM("2024-12") AS "2024-12",
            SUM("2025-01") AS "2025-01",
            SUM("2025-02") AS "2025-02",
            SUM("2025-03") AS "2025-03",
            SUM("2025-04") AS "2025-04",
            SUM("2025-05") AS "2025-05",
            SUM("2025-06") AS "2025-06",
            SUM("2025-07") AS "2025-07",
            SUM("2025-08") AS "2025-08"
        FROM t7 NATURAL JOIN t6
        GROUP BY hotelid, name, total_orders_num;
```



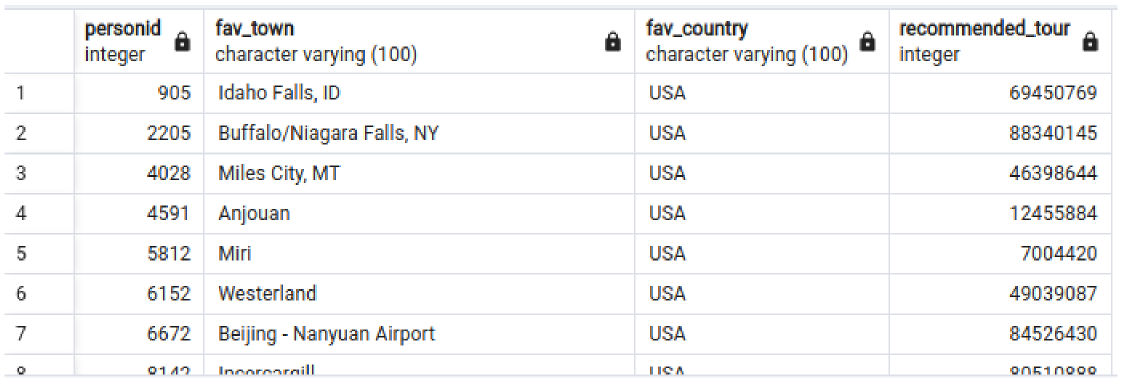
	hotelid integer	name character varying (100)	total_orders_num bigint	2024-09 numeric	2024-10 numeric	2024-11 numeric	2024-12 numeric	2025-01 numeric	2025-02 numeric	2025-03 numeric	2025-04 numeric
1	76	Old residence inn brilliant resort.	28	174404	186144	35518	0	35078	202558	159305	151206
2	161	Best sea sea.	14	120140	101567	120400	44128	0	46033	64606	0
3	1312	Inn inn luxury residence.	17	167444	0	0	77864	70312	0	57432	5455
4	1429	Residence hotel inn spa b&b old best residence.	22	44079	73433	78163	133741	122592	129602	126770	56169
5	1794	Resort sunshine best brilliant old sea.	8	0	0	0	40134	72193	40134	110766	0
6	2459	Sunshine sea old brilliant inn wonderful wonderful luxury.	21	44684	84404	156350	0	47790	69737	203207	62184
7	4334	Sunshine brilliant b&b brilliant brilliant residence residence.	23	23209	113116	61143	36814	208	31704	61143	86258

10. Найти для каждого пользователя те направления в которых он предпочитает путешествовать(предпочтительная страна и предпочтительные города) на основе этих предпочтений предложить пользователю тур, который мог бы ему понравиться

```
--t1-t11 получение таблицы человек-город по всем заказам и всем избранным
WITH tour_town AS (SELECT
                        tourid,
                        arrival_airport_townid AS townid
                    FROM flight JOIN tour ON tour.flightid=flight.flightid),
t1 AS (SELECT
            orderid,
            personid,
            flightid
        FROM m_order NATURAL JOIN flight_order),
t2 AS (SELECT
            orderid,
            personid,
            tourid
        FROM m_order NATURAL JOIN tour_order),
t3 AS (SELECT
            orderid,
            personid,
            roomid
        FROM m_order NATURAL JOIN room_order),
t4 AS (SELECT
            orderid,
            personid,
            townid
        FROM t2 NATURAL JOIN tour_town),
t6 AS (SELECT
            orderid,
            personid,
            arrival_airport_townid
        FROM t1 NATURAL JOIN flight),
t7 AS (SELECT
            orderid,
            personid,
            townid
        FROM t3
        NATURAL JOIN
        (SELECT * FROM hotel NATURAL JOIN room)
        ),
t8 AS (SELECT
            favouritesid,
            personid,
            arrival_airport_townid
        FROM favourites
        NATURAL JOIN
        (SELECT * FROM flight_favourites NATURAL JOIN flight)
        ),
t9 AS (SELECT
            favouritesid,
            personid,
            arrival_airport_townid
        FROM favourites
        NATURAL JOIN
            (SELECT *
            FROM tour_favourites NATURAL JOIN
                                            (SELECT *
                                            FROM flight JOIN tour ON
tour.flightid=flight.flightid)
            )
        ),
t10 AS (SELECT
            favouritesid,
            personid,
            townid
        FROM favourites
        NATURAL JOIN
        (SELECT * FROM hotel_favourites NATURAL JOIN hotel)
```

```sql
        ),
t11 AS (SELECT * FROM t4
        UNION ALL
        SELECT * FROM t6
        UNION ALL
        SELECT * FROM t7
        UNION ALL
        SELECT * FROM t8
        UNION ALL
        SELECT * FROM t9
        UNION ALL
        SELECT * FROM t10),
count_towns AS (SELECT
                    personid,
                    name,
                    townid,
                    ROW_NUMBER() OVER (PARTITION BY personid ORDER BY
COUNT(name) DESC) rn1
                FROM t11 NATURAL JOIN town
                GROUP BY personid, name, townid),
count_countries AS (SELECT
                        personid,
                        country,
                        ROW_NUMBER() OVER (PARTITION BY personid ORDER BY
COUNT(country) DESC) rn2
                    FROM t11 NATURAL JOIN town
                    GROUP BY personid, country),
--t12-t13 получение предпочитаемого города и предпочитаемой страны
t12 as(SELECT
            personid,
            townid,
            name AS fav_town
        FROM count_towns
        WHERE rn1 = 1),
t13 AS (SELECT
            personid,
            country AS fav_country
        FROM count_countries
        WHERE rn2 = 1),
fav AS (SELECT * FROM t12 NATURAL JOIN t13),
rec_by_town AS (SELECT fav.personid,
                       (SELECT tourid
                        FROM tour_town
                        WHERE tour_town.townid = fav.townid
                        EXCEPT
                        SELECT tourid
                        FROM t2
                        WHERE personid = fav.personid LIMIT 1) AS
recommend_by_town_tourid
                FROM fav),
tour_country AS (SELECT * FROM tour_town NATURAL JOIN town),
rec_by_country AS (SELECT fav.personid,
                        (SELECT tourid
                        FROM tour_country
                        WHERE tour_country.country = fav.fav_country
                        EXCEPT
                        SELECT tourid
                        FROM t2
                        WHERE personid = fav.personid LIMIT 1) AS
recommend_by_country_tourid
                    FROM fav)
SELECT
    personid,
    fav_town,
    fav_country,
    COALESCE(recommend_by_town_tourid, recommend_by_country_tourid) AS
```

```
recommended_tour
FROM fav NATURAL JOIN
(rec_by_town NATURAL JOIN rec_by_country);
```



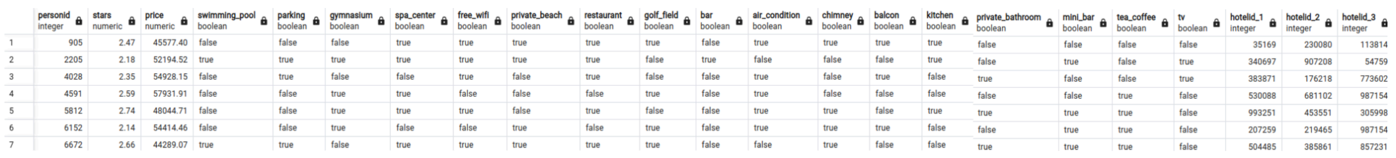
	personid integer	fav_town character varying (100)	fav_country character varying (100)	recommended_tour integer
1	905	Idaho Falls, ID	USA	69450769
2	2205	Buffalo/Niagara Falls, NY	USA	88340145
3	4028	Miles City, MT	USA	46398644
4	4591	Anjouan	USA	12455884
5	5812	Miri	USA	7004420
6	6152	Westerland	USA	49039087
7	6672	Beijing - Nanyuan Airport	USA	84526430
8	8142	Invercargill	USA	80510888

11. Для каждого пользователя определить параметры отеля, которые он обычно выбирает и предложить ему три отеля, которые могли бы ему понравиться

```
--t1-t5 получение единой таблицы человек-отель-комната по всем заказам
WITH t1 AS (SELECT
                personid,
                roomid
            FROM (SELECT * FROM room_order NATURAL JOIN m_order)
            ),
t2 AS  (SELECT
            personid,
            roomid
        FROM (SELECT *
             FROM tour NATURAL JOIN (SELECT * FROM tour_order NATURAL JOIN
m_order)
             )
        ),
t3 AS  (SELECT
            personid,
            roomid
        FROM favourites NATURAL JOIN (SELECT *
                                      FROM tour_favourites
                                                    NATURAL JOIN
                                                    (SELECT
                                                        tour.roomid,
                                                        tourid
                                                    FROM room JOIN tour ON
room.roomid=tour.roomid)
                                      )
        ),
t4 AS (SELECT * FROM t1 UNION ALL SELECT * FROM t2 UNION ALL SELECT * FROM
t3),
t5 AS (SELECT *
        FROM t4
        NATURAL JOIN
        (SELECT * FROM room NATURAL JOIN hotel)
      ),
--t6 общее число заказов для каждого человека
t6 AS (SELECT
            personid,
            COUNT(*) AS total_count
        FROM t5
        GROUP BY personid),
--t7 расчет средней цены, среднего количества звезд и в скольки процентах
случаев человек выбирал то или иное удобство
t7 AS (SELECT personid,
                AVG(num_of_stars) AS stars,
                AVG(price) AS price,
                CAST(SUM(swimming_pool::int)AS real)*100/total_count
                    AS swimming_pool,
                CAST(SUM(parking::int)AS real)*100/total_count
                    AS parking,
                CAST(SUM(gymnasium::int)AS real)*100/total_count
                    AS gymnasium,
                CAST(SUM(spa_center::int)AS real)*100/total_count
                    AS spa_center,
                CAST(SUM(free_WiFi::int)AS real)*100/total_count
                    AS free_WiFi,
                CAST(SUM(private_beach::int)AS real)*100/total_count
                    AS private_beach,
                CAST(SUM(restaurant::int)AS real)*100/total_count
                    AS restaurant,
                CAST(SUM(golf_field::int)AS real)*100/total_count
                    AS golf_field ,
                CAST(SUM(bar::int)AS real)*100/total_count
                    AS bar,
                CAST(SUM(air_condition::int)AS real)*100/total_count
                    AS air_condition,
                CAST(SUM(chimney::int)AS real)*100/total_count
                    AS chimney,
                CAST(SUM(balcon::int)AS real)*100/total_count
                    AS balcon,
                CAST(SUM(kitchen::int)AS real)*100/total_count
                    AS kitchen,
                CAST(SUM(private_bathroom::int)AS real)*100/total_count
                    AS private_bathroom,
                CAST(SUM(mini_bar::int)AS real)*100/total_count
                    AS mini_bar,
                CAST(SUM(tea_coffee::int)AS real)*100/total_count
                    AS tea_coffee,
                CAST(SUM(tv::int)AS real)*100/total_count
                    AS tv
        FROM t5 NATURAL JOIN t6 GROUP BY personid, total_count),
pref AS (SELECT
                personid, round(stars, 2) AS stars, round(price,2)
                    AS price,
                CASE WHEN swimming_pool > 50 THEN true ELSE false END
```

```
                    AS swimming_pool,
                CASE WHEN parking > 50 THEN true ELSE false END
                    AS parking,
                CASE WHEN gymnasium > 50 THEN true ELSE false END
                    AS gymnasium,
                CASE WHEN spa_center > 50 THEN true ELSE false END
                    AS spa_center,
                CASE WHEN free_WiFi > 50 THEN true ELSE false END
                    AS free_WiFi,
                CASE WHEN private_beach > 50 THEN true ELSE false END
                    AS private_beach,
                CASE WHEN restaurant > 50 THEN true ELSE false END
                    S restaurant,
                CASE WHEN golf_field > 50 THEN true ELSE false END
                    AS golf_field,
                CASE WHEN bar > 50 THEN true ELSE false END
                    AS bar,
                CASE WHEN air_condition > 50 THEN true ELSE false END
                    AS air_condition,
                CASE WHEN chimney > 50 THEN true ELSE false END
                    AS chimney,
                CASE WHEN balcon > 50 THEN true ELSE false END
                    AS balcon,
                CASE WHEN kitchen > 50 THEN true ELSE false END
                    AS kitchen,
                CASE WHEN private_bathroom > 50 THEN true ELSE false END
                    AS private_bathroom,
                CASE WHEN mini_bar > 50 THEN true ELSE false END
                    AS mini_bar,
                CASE WHEN tea_coffee > 50 THEN true ELSE false END
                    AS tea_coffee,
                CASE WHEN tv > 50 THEN true ELSE false END
                    AS tv
```

```
            FROM t7),
ranked_hotels AS (SELECT
                    personid, hotel_room.hotelid,
                    ROW_NUMBER() OVER (PARTITION BY pref.personid ORDER BY (
                        (pref.price >= hotel_room.price)::int
                        +
                        (pref.stars <= hotel_room.num_of_stars)::int
                        +
                        (pref.swimming_pool = hotel_room.swimming_pool)::int
                        +
                        (pref.parking = hotel_room.parking)::int
                        +
                        (pref.gymnasium = hotel_room.gymnasium)::int
                        +
                        (pref.spa_center = hotel_room.spa_center)::int
                        +
                        (pref.free_WiFi = hotel_room.free_wifi)::int
                        +
                        (pref.private_beach = hotel_room.private_beach)::int
                        +
                        (pref.restaurant = hotel_room.restaurant)::int
                        +
                        (pref.golf_field = hotel_room.golf_field)::int
                        +
                        (pref.bar = hotel_room.bar)::int
                        +
                        (pref.air_condition = hotel_room.air_condition)::int
                        +
                        (pref.chimney = hotel_room.chimney)::int
                        +
                        (pref.balcon = hotel_room.balcon)::int
                        +
                        (pref.kitchen = hotel_room.kitchen)::int
                        +
                        (pref.private_bathroom =
                            hotel_room.private_bathroom)::int
                        +
                        (pref.mini_bar = hotel_room.mini_bar)::int
                        +
                        (pref.tea_coffee = hotel_room.tea_coffee)::int
                        +
                        (pref.tv = hotel_room.tv)::int) DESC) AS rn
                    FROM pref, (hotel NATURAL JOIN room) AS hotel_room),
recommend AS (SELECT
                  personid,
                  MAX(CASE WHEN rn = 1 THEN hotelId END) AS hotelid_1,
                  MAX(CASE WHEN rn = 2 THEN hotelId END) AS hotelid_2,
                  MAX(CASE WHEN rn = 3 THEN hotelId END) AS hotelid_3
                FROM ranked_hotels
                GROUP BY personid)
SELECT * FROM pref NATURAL JOIN recommend;
```

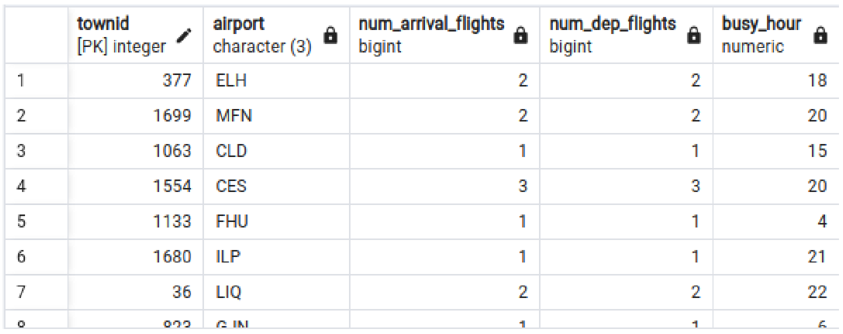


	personid	stars	price	swimming_pool	parking	gymnasium	spa_center	free_wifi	private_beach	restaurant	golf_field	bar	air_condition	chimney	balcon	kitchen	private_bathroom	mini_bar	tea_coffee	tv	hotelid_1	hotelid_2	hotelid_3
	integer	numeric	numeric	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	boolean	integer	integer	integer
1	905	2.47	45577.40	false	false	false	true	true	true	true	true	false	true	true	true	true	false	false	false	false	35169	230080	113814
2	2205	2.18	52194.52	true	true	true	true	true	true	true	false	true	false	false	false	true	false	true	false	true	340697	907208	54759
3	4028	2.35	54928.15	false	true	false	false	true	false	true	false	false	true	false	true	false	true	false	false	true	383871	176218	773602
4	4591	2.59	57931.91	false	false	true	true	false	false	false	true	false	true	false	false	false	false	false	true	false	530088	681102	987154
5	5812	2.74	48044.71	false	false	true	true	true	false	true	false	false	true	true	false	true	true	true	true	false	993251	453551	305998
6	6152	2.14	54414.46	false	false	true	false	false	true	false	true	false	true	true	true	false	false	true	true	false	207259	219465	987154
7	6672	2.66	44289.07	true	true	false	true	true	true	true	true	false	false	true	false	false	true	true	true	false	504485	385861	857231

12. Для каждого аэропорта посчитать сколько самолетов за день вылетает из него и сколько в него прилетает, а также в какие часы аэропорт наиболее загружен

```
WITH t1 AS (SELECT
                townid,
                nearest_airport_code,
                extract(hour FROM arrival_time) AS time
            FROM flight JOIN town ON
                             flight.arrival_airport_townid = town.townid),
t2 AS (SELECT
            townid,
```

```sql
            nearest_airport_code,
            extract(hour FROM departure_time) AS time
        FROM flight JOIN town ON flight.arrival_airport_townid= town.townid),
t3 AS (SELECT * FROM t1 UNION ALL SELECT * FROM t2),
num_flights AS (SELECT *
                FROM
                    (SELECT
                        townid,
                        nearest_airport_code AS airport,
                        COUNT(*) AS num_arrival_flights
                    FROM t1
                    GROUP BY townid, nearest_airport_code)
                    NATURAL JOIN
                    (SELECT
                        townid,
                        nearest_airport_code AS airport,
                        COUNT(*) AS num_dep_flights
                    FROM t2
                    GROUP BY townid, nearest_airport_code)
                ),
t4 AS (SELECT
            townid,
            nearest_airport_code, time,
            COUNT(*) AS c
        FROM t3
        GROUP BY townid, nearest_airport_code, time
        ORDER BY townid),
t5 AS (SELECT
            townid,
            nearest_airport_code,
            time,
            ROW_NUMBER() OVER (PARTITION BY townid ORDER BY time DESC) AS rn
        FROM t4),
busy_hour AS (SELECT
                    townid,
                    nearest_airport_code AS airport,
                    time AS busy_hour
              FROM t5 WHERE rn=1)
SELECT * FROM num_flights NATURAL JOIN busy_hour;
```



	townid [PK] integer	airport character (3)	num_arrival_flights bigint	num_dep_flights bigint	busy_hour numeric
1	377	ELH	2	2	18
2	1699	MFN	2	2	20
3	1063	CLD	1	1	15
4	1554	CES	3	3	20
5	1133	FHU	1	1	4
6	1680	ILP	1	1	21
7	36	LIQ	2	2	22
8	823	GJN	1	1	6