|  |  |
| --- | --- |
| **Российский университет транспорта (МИИТ)**  **Институт транспортной техники и систем управления**  **Кафедра «Управление и защита информации»** | |
| **Отчет**  **по курсовому проекту**  **по теме «Разработка БД для гостиницы»**  **по дисциплине «Основы построения защищенных баз данных»** | |
|  | Выполнил:  Студент группы ТКИ-441  Талатин Д.А.  Проверил:  Доцент кафедры УиЗи, к.т.н.  Васильева М.А. |
| Москва 2024 | |

Оглавление

[ЦЕЛЬ КУРСОВОГО ПРОЕКТА 3](#_Toc167878977)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 3](#_Toc167878978)

[2 АНАЛИЗ ИНФОРМАЦИОННЫХ ЗАДАЧ И КРУГА ПОЛЬЗОВАТЕЛЕЙ СИСТЕМЫ 4](#_Toc167878979)

[2.1 Функциональные возможности 5](#_Toc167878980)

[2.2 Готовые запросы 5](#_Toc167878981)

[3 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БД 5](#_Toc167878982)

[3.1 Преобразование ER-диаграммы в схему базы данных 5](#_Toc167878983)

[3.2 Составление реляционных отношений 7](#_Toc167878984)

[3.3 Определение дополнительных ограничений 8](#_Toc167878985)

[3.4 Нормализация полученных отношений 10](#_Toc167878986)

[4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БД 11](#_Toc167878987)

[4.1 Разработка скриптов для создания базы данных и таблиц 11](#_Toc167878988)

[4.1.1 Ограничения для таблицы «Заказы» 12](#_Toc167878989)

[4.1.2 Ограничения для таблицы «Клиенты» 13](#_Toc167878990)

[4.1.3 Ограничения для таблицы «Транспорт» 14](#_Toc167878991)

[4.2 Диаграмма базы данных 15](#_Toc167878992)

[4.3 Разработка скриптов для добавления данных в таблицы 16](#_Toc167878993)

[4.4 Проверка ограничений 17](#_Toc167878994)

[4.5 Разработка необходимых представлений (view) 20](#_Toc167878995)

[4.6 Разработка необходимых функций и процедур 21](#_Toc167878996)

[4.7 Разработка необходимых триггеров 22](#_Toc167878997)

[ЗАКЛЮЧЕНИЕ 22](#_Toc167878998)

# ЦЕЛЬ КУРСОВОГО ПРОЕКТА

Целью курсового проекта является изучение методов и закрепление знаний в проектировании реляционных баз данных (РБД) в системе управления базами данных (СУБД) PostgreSQL.

# 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В данном курсовом проекте для проектирования реляционной базы данных была выбрана гостиница.

БД создается для информационного обслуживания сотрудников гостиницы и клиентов. В гостинице имеются номера, которые различаются по количеству мест, количеству комнат, удобствам и т. д., от всех этих параметров зависит цена номера (цена за номер будет выше). Посетители могут занимать номера, бронировать их заранее.

Постоянным посетителями и детям до 12 лет предоставляется скидка.

Предметная область данной системы включает в себя следующие особенности:

1. Возможность бронирования номера с учетом пожеланий клиента
2. Подбор подходящего номера в зависимости от необходимого количество спальных мест
3. Предоставление списка доступных для подключения услуг
4. Расчет стоимости номера с учетом предпочитаемых услуг
5. Расчет скидки предоставляемой клиенту.

Выделим базовые сущности предметной области:

1. Сущность «Гость» содержит в себе идентификатор клиента, также ФИО клиента, дату рождения, номер телефона, количество посещений данной гостиницы и информация о том имеется у него скидка или нет;
2. Сущность «Комната» содержит в себе номер комнаты, цену, статус комнаты и количество спальных мест;
3. Сущность «Сервисы» содержит в себе идентификационный номер услуги, название предоставляемой услуги и цену;
4. Сущность «Услуги в номер» содержит в себе идентификационный номер подключения услуги, название , номер в котором подключена данная услуга, цену услуги и идентификационный номер услуг;
5. Общая сущность, связывающая все вышеизложенные, называется   
   «Бронирование номера» содержит в себе идентификационный номер брони, идентификационный номер гостя, проживающего в номере, номер комнаты, даты заезда и выезда из номера
6. ER-диаграмма БД, выполненная в графическом редакторе *MS Visio*, приведена далее (Рисунок 1). Данная БД содержит связи «один-ко-многим».

|  |
| --- |
|  |
| 1. – ER-диаграмма |

# 2 АНАЛИЗ ИНФОРМАЦИОННЫХ ЗАДАЧ И КРУГА ПОЛЬЗОВАТЕЛЕЙ СИСТЕМЫ

Система создается для информационного обслуживания сотрудников гостиницы, где требуется отслеживание бронирования номеров, отслеживание информации о клиентах и услугах предоставляемых в номера.

## 2.2 Готовые запросы

Наша система будет иметь следующие готовые запросы:

1. Бронирование посетителем номера в гостинице в  
   зависимости от потребностей (количество мест и т. д.).
2. Отображение количества свободных комнат для данного типа  
   номера.
3. Определение скидки для посетителя.
4. Отображение информации по данному номеру (количество мест,  
   холодильник, телевизор, бар и т. д.).

# 3 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БД

## 3.1 Преобразование ER-диаграммы в схему базы данных

База данных создаётся на основании схемы базы данных. Преобразование ER-диаграммы в схему БД выполняется путем сопоставления каждой сущности и каждой связи, имеющей атрибуты, отношения   
(таблицы БД).

На схеме, представленной ниже (Рисунок 2), введено используемое обозначение:

|  |
| --- |
|  |
| 1. – Обозначение на схеме базы данных |

Полученная схема реляционной базы данных (далее, РБД) гостиницы приведена ниже (Рисунок 3).

|  |
| --- |
|  |
| 1. – Схема РБД, полученная из ER-диаграммы компании |

В вашем примере создания таблиц базы данных можно определить следующие типы отношений:

1. **Отношение "один ко многим" между таблицами "Guests" и "Reservations"**: В таблице "Reservations " есть поле "GuestID", которое является внешним ключом, ссылающимся на таблицу "Guests". Это означает, что один клиент может иметь несколько броней, но каждая бронь связана только с одним клиентом.
2. **Отношение "один ко многим" между таблицами "RoomService" и "Rooms"**: В таблице "RoomService" есть поле "RoomNumber", которое является внешним ключом, ссылающимся на таблицу "Rooms". Это означает, что в одном номере могут быть подключены несколько услуг, но каждая услуга связан только с одним номером.
3. **Отношение "один ко многим" между таблицами " Reservations " и "Rooms"**: В таблице "Reservarions" есть поле "RoomNumber", которое является внешним ключом, ссылающимся на таблицу "Rooms". Это означает, зарезервировано может быть несколько номеров, но каждый номер связан только с одним бронированием.

Таким образом, в вашей базе данных реализованы отношения "один ко многим" между гостями и списком броней, бронированием номеров и номерами, а также между услугами в номере и номерами .

## 3.2 Составление реляционных отношений

Одно реляционное отношение соответствует одной сущности (объекту предметной области) и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи. В том случае, если базовое отношение не имеет потенциальных ключей, вводится суррогатный первичный ключ, который не несёт смысловой нагрузки и служит только для идентификации записей.

В данной системе сущности «Гости» и «Комнаты» имеют атрибут «GuestID» и «ServiceID» соответственно с целью компактного сбора в общей сущности «Reservations» - то есть можно назвать такие атрибуты суррогатными первичными ключами.

Отношения приведены ниже (см. Таблица 1). Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной.

Таблица 1 –Гости

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Гости  (Guests) | Идентификатор гостя | GuestID | Целое число | Первичный ключ |
| Имя клиента | FirstName | Строка (255) | Обязательное  поле |
| Фамилия клиента | LastNAme | Строка (255) | Обязательное поле |
| Телефон | PhoneNumber | Строка (255) | Обязательное, уникальное поле |
| Количество Посещений | VisitCount | Целое число | Обязательное поле |
| Скидка | Discount | numeric | Обязательное поле |

Таблица 2 –Комнаты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Комнаты  (Rooms) | Номер комнаты | RoomNumber | Целое число | Первичный ключ |
| Цена | Price | Десятичное число | Обязательное  поле |
| Статус кмонаты | IsAvialble | Логическое | Обязательное поле |
| Количество спальных мест | Beds | Целое число | Обязательное поле |

Таблица 3 –Бронирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Бронирование  (Reservations) | Идентификатор брони | ReservationID | Целое число | Первичный ключ |
| Идентификатор клиента | GuestID | Целое число | Обязательное, уникальное поле |
| Номер комнаты | RoomNumber | Целое число | Обязательное поле |
| Дата заезда | CheckInDate | Дата | Обязательное поле |
| Дата выезда | CheckOutDate | Дата | Обязательное поле |

Таблица 4 –Подключенные услуги

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Подключенные Услуги  (RoomServices) | Идентификатор перечня услуг | RoomServiceID | Целое число | Первичный ключ |
| Название услуги | ServiceName | Строка  (255) | Обязательное поле |
| Номер комнаты | RoomNumber | Целое число | Обязательное уникальное поле |
| Цена услуги | ServiceCost | Десятичное число | Обязательное поле |
| Идентификатор услуги | ServiceID | Целое число | Обязательное уникальное поле |

Таблица 3 –Услуги

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Услуги  (Services) | Идентификатор услуги | ServiceID | Целое число | Первичный ключ |
| Название услуги | ServiceName | Строка  (255) | Обязательное поле |
| Цена услуги | ServiceCost | Десятичное число | Обязательное поле |

## 3.3 Определение дополнительных ограничений

Для обеспечения целостности данных и правильности отношений в базе данных, можно добавить следующие ограничения:

1. **Первичные ключи**: Каждая таблица должна иметь первичный ключ, который гарантирует уникальность записей и не допускает NULL значений.
2. **Ограничение на тип данных**: Поля, которые хранят числовые данные, должны иметь соответствующие ограничения на тип данных.
3. **Ограничения ссылочной целостности**: эти ограничения гарантируют, что значения внешних ключей всегда ссылается на существующие значения в связанных таблицах. Например, в таблице "Reservations" ссылаются на существующие значения в таблицах "Guest" и "Rooms".
4. **Ограничение на уникальность**: ограничения, гарантируют, что каждая запись в таблице имеет уникальное значение для определенного поля или комбинации полей.
5. **Ограничения целостности проверки**: они позволяют определить допустимые диапазоны значений для определенных полей. Например, для таблицы "Services" можно добавить ограничение, что стоимость услуги не может быть отрицательной.
6. **Ограничения на обновление и удаление**: можно установить правила, которые определяют, что происходит при обновлении или удалении записи, связанной с другими записями. Например, для таблицы "Reservations" можно установить правило каскадного удаления, чтобы при удалении клиента или вида транспорта все связанные заказы также удалялись:
7. **Ограничения на нулевые значения**: гарантируют, что определенные поля не могут содержать нулевые значения.

## 3.4 Нормализация полученных отношений

Нормализация отношений в БД является одним из ключевых аспектов проектирования и играет важную роль в обеспечении целостности данных, эффективности и удобства их использования.

Для применения нормализации к нашей базе данных, состоящей из таблиц "Guests", "Rooms" и "Reservations","RoomServisces", "Services" мы проверим каждую таблицу на соответствие требованиям первой (1NF), второй (2NF) и третьей (3NF) нормальных форм.

**1. Первая нормальная форма (1NF)**

Таблицы, представленные выше, уже находятся в первой нормальной форме, так как:

* Все атрибуты атомарны.
* Каждая строка уникальна.
* Таблицы не содержат повторяющихся групп.

**2. Вторая нормальная форма (2NF)**

Все три таблицы также находятся во второй нормальной форме, так как:

* Находятся в первой нормальной форме.
* Все неключевые атрибуты зависят от всего первичного ключа.

**3. Третья нормальная форма (3NF)**

Все три таблицы находятся в третьей нормальной форме, так как:

* Находятся во второй нормальной форме.
* Все неключевые атрибуты зависят только от первичного ключа и не имеют транзитивных зависимостей.

Таким образом, все таблицы в вашей базе данных соответствуют требованиям третьей нормальной формы. Это означает, что данные в базе данных нормализованы, что уменьшает избыточность и помогает избежать аномалий при обновлении, вставке или удалении данных.

# 4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БД

## 4.1 Разработка скриптов для создания базы данных и таблиц

Первоначально создаётся база данных «Гостиница»:

create database "Hotel";

Далее, были созданы таблицы «Гости», «Комнаты» ,«Бронирование», «Услуги», «Подключение услуг»:

CREATE TABLE "Guests" (

"GuestID" INT ,

"FirstName" VARCHAR(255),

"LastName" VARCHAR(255),

"DateOfBirth" DATE,

"PhoneNumber" VARCHAR(255),

"VisitCount" INT,

"Discount" Decimal (3,2)

)

CREATE TABLE "Rooms" (

"RoomNumber" INT,

"Price" DECIMAL(10, 2),

"IsAvailable" BOOLEAN,

"Beds" INT

)

CREATE TABLE "Reservations" (

"ReservationID" INT,

"GuestID" INT,

"RoomNumber" INT,

"CheckInDate" DATE,

"CheckOutDate" DATE

)

CREATE TABLE "Services" (

"ServiceID" INT,

"ServiceName" VARCHAR(255),

"ServiceCost" DECIMAL(10, 2)

CREATE TABLE "RoomServices" (

"RoomServiceID" INT ,

"RoomNumber" INT ,

"ServiceCost" Decimal (10,2),

"ServiceID" INT

);

Добавление в созданные таблицы первичных и внешних ключей. Первичные ключи:

ALTER TABLE "Guests"

ADD CONSTRAINT "PK\_Guests"

PRIMARY KEY ("GuestID");

ALTER TABLE "Rooms"

ADD CONSTRAINT "PK\_Rooms"

PRIMARY KEY ("RoomNumber");

ALTER TABLE "Reservations"

ADD CONSTRAINT "PK\_Reservations"

PRIMARY KEY ("ReservationID");

ALTER TABLE "Services"

ADD CONSTRAINT "PK\_Services"

PRIMARY KEY ("ServiceID");

ALTER TABLE "RoomServices"

ADD CONSTRAINT "PK\_RoomServices"

PRIMARY KEY ("RoomServiceID");

Внешние ключи:

ALTER TABLE "Guests"

ADD CONSTRAINT "FK\_Rooms\_Guests"

FOREIGN KEY ("RoomNumber")

REFERENCES "Rooms"("RoomNumber");

ALTER TABLE "Reservations"

ADD CONSTRAINT "FK\_Rooms\_Reservations"

FOREIGN KEY ("RoomNumber")

REFERENCES "Rooms"("RoomNumber");

ALTER TABLE "Reservations"

ADD CONSTRAINT "FK\_Guests\_Reservations"

FOREIGN KEY ("GuestID")

REFERENCES "Guests"("GuestID");

ALTER TABLE "RoomServices"

ADD constraint "FK\_Rooms\_RoomServices"

FOREIGN KEY ("RoomNumber")

REFERENCES "Rooms"("RoomNumber");

ALTER TABLE "RoomServices"

ADD constraint "FK\_Services\_RoomServices"

FOREIGN KEY ("ServiceID")

REFERENCES "Services"("ServiceID");

На следующем этапе вносятся ограничения для таблиц.

### 4.1.1 Ограничения для таблицы «Гости»

ALTER TABLE "Guests"

ALTER COLUMN "FirstName"

SET NOT NULL;

ALTER TABLE "Guests"

ALTER COLUMN "LastName"

SET NOT NULL;

ALTER TABLE "Guests"

ALTER COLUMN "DateOfBirth"

SET NOT NULL;

ALTER TABLE "Guests"

ALTER COLUMN "PhoneNumber"

SET NOT NULL;

ALTER TABLE "Guests"

ADD CONSTRAINT "CH\_Guests\_DateOfBirth"

CHECK ("DateOfBirth" < CURRENT\_DATE);

ALTER TABLE "Guests"

ALTER COLUMN "Discount" SET DEFAULT 0;

### 4.1.2 Ограничения для таблицы «Комнаты»

ALTER TABLE "Rooms"

ALTER COLUMN "Price"

SET NOT NULL;

ALTER TABLE "Rooms"

ALTER COLUMN "IsAvailable"

SET NOT NULL;

ALTER TABLE "Rooms"

ALTER COLUMN "Beds"

SET NOT NULL;

### 4.1.3 Ограничения для таблицы «Бронирование»

ALTER TABLE "Reservations"

ADD CONSTRAINT "PK\_Reservations"

PRIMARY KEY ("ReservationID");

ALTER TABLE "Reservations"

ALTER COLUMN "CheckInDate"

SET NOT NULL;

ALTER TABLE "Reservations"

ALTER COLUMN "CheckOutDate"

SET NOT NULL;

ALTER TABLE "Reservations"

ADD CONSTRAINT "CH\_Reservations\_CheckOutDate"

CHECK ("CheckOutDate" >= "CheckInDate");

### Ограничения для таблицы «Услуги »

ALTER TABLE "Services"

ALTER COLUMN "ServiceName"

SET NOT NULL;

ALTER TABLE "Services"

ALTER COLUMN "ServiceCost"

SET NOT NULL;

ALTER TABLE "Services"

ADD CONSTRAINT "U\_Services\_ServiceName"

UNIQUE ("ServiceName");

## 4.2 Диаграмма базы данных

Результатом разработки скриптов является диаграмма базы данных, представленная ниже (Рисунок 4).

|  |
| --- |
|  |
| 1. – Диаграмма базы данных |

## 4.3 Разработка скриптов для добавления данных в таблицы

Заполнение таблицы «Guests»:

INSERT INTO "Guests" values

(2,'Мария', 'Петрова', '1992-07-23', '+79037654321', 1, 0),

(3,'Игорь', 'Смирнов', '1978-11-01', '+79039876543', 3, 0),

(4,'Елена', 'Кузнецова', '1989-02-17', '+79034567890', 3, 0),

(5,'Дмитрий', 'Соколов', '1995-05-05', '+79033098765', 2, 0),

-- Добавление гостя младше 12 лет

(6,'Иван', 'Морозов', '2013-06-08', '+79032222334', 1, 0),

-- Добавление гостя с количеством посещений равным 10

(7,'Светлана', 'Волкова', '1980-11-11', '+79036664455', 1, 0);

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Guests» |

Заполнение таблицы «Rooms»:

INSERT INTO "Rooms" values

(101, 1500.00, TRUE, 2),

(102, 2000.00, TRUE, 2),

(103, 2500.00, TRUE, 3),

(104, 3000.00, TRUE, 4);

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Rooms» |

Заполнение таблицы «Reservations»:

INSERT INTO "Reservations" values

(1, 1, 101, '2024-06-03', '2024-06-10'),

(2, 4, 102, '2024-06-05', '2024-06-15'),

(3, 3, 103, '2024-06-08', '2024-06-12'),

(4, 7, 104, '2024-06-09', '2024-06-14');

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Reservations» |

Заполнение таблицы «Services»:

INSERT INTO "Services" values

(1, 'Холодильник', 500.00),

(2, 'Телевизор', 300.00),

(3, 'Кондиционер', 200.00),

(4, 'МиниБар', 1500.00);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

1. – Внесение данных в таблицу «Services»

Заполнение таблицы «RoomServices»:

INSERT INTO "RoomServices" values

(7,'Холодильник', 104, 500.00,1)

(1,'Холодильник', 101, 500.00, 1),

(2, 'Телевизор',101, 300.00, 2),

(3, 'Холодильник',102, 500.00, 1),

(4, 'Кондиционер',102, 200.00, 3),

(5,'Телевизор', 104, 300.00, 2),

(6, 'МиниБар',104, 1500.00, 4);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

1. – Внесение данных в таблицу «RoomServices»

## 4.5 Разработка необходимых представлений (view)

К необходимому представлению, относится вывод свободных номеров по заданному критерию, которое реализуется следующим образом:

CREATE OR REPLACE VIEW "AvailableRoomsView" AS

SELECT

r."RoomNumber",

r."Beds",

r."Price",

ROW\_NUMBER() OVER (PARTITION BY r."Beds" ORDER BY r."Price" ASC) as "RoomRank"

FROM

"Rooms" r

WHERE

r."IsAvailable" = TRUE

AND NOT EXISTS (

SELECT 1

FROM "Reservations" res

WHERE res."RoomNumber" = r."RoomNumber"

AND res."CheckInDate" <= CURRENT\_DATE

AND res."CheckOutDate" >= CURRENT\_DATE

);

--Вызов представления view

SELECT "RoomNumber", "Beds", "Price"

FROM "AvailableRoomsView"

WHERE "Beds" >= 1;

|  |
| --- |
|  |
| 1. – Представление и результат |

## 4.6 Разработка необходимых функций

Основная функция, используемая в данной работе – это добавление новой записи о бронировании комнаты и подключении необходимых услуг

**Функция бронирования**:

CREATE OR REPLACE FUNCTION "ReserveRoom"(

p\_GuestID INT,

p\_NumberOfBeds INT,

p\_CheckInDate DATE,

p\_CheckOutDate DATE,

p\_SelectedServices INT[]

) RETURNS TABLE (

GuestID INTEGER,

RoomNumber INTEGER,

CheckInDate DATE,

CheckOutDate DATE,

ConnectedServices TEXT,

FinalPrice NUMERIC

) AS

$$

DECLARE

v\_GuestRecord RECORD;

v\_RoomNumber INT;

v\_ServiceDetails TEXT;

v\_Discount DECIMAL;

v\_TotalPrice DECIMAL;

v\_ServiceID INT;

v\_ServiceName VARCHAR(255);

v\_ServiceCost NUMERIC;

v\_ServicesTotalCost NUMERIC := 0;

BEGIN

-- Получение данных о госте по ID

SELECT \* INTO v\_GuestRecord FROM "Guests" WHERE "GuestID" = p\_GuestID;

-- Выбор одной подходящей комнаты и обновление её статуса

SELECT "RoomNumber" INTO v\_RoomNumber FROM (

SELECT "RoomNumber"

FROM "Rooms"

WHERE "Beds" = p\_NumberOfBeds AND "IsAvailable" = TRUE

ORDER BY "RoomNumber"

LIMIT 1

FOR UPDATE SKIP LOCKED

) AS subquery;

UPDATE "Rooms"

SET "IsAvailable" = FALSE

WHERE "RoomNumber" = v\_RoomNumber;

-- Добавление записи в "Reservations"

INSERT INTO "Reservations" ("ReservationID", "GuestID", "RoomNumber", "CheckInDate", "CheckOutDate")

VALUES (NEXTVAL('reservations\_sequence'), p\_GuestID, v\_RoomNumber, p\_CheckInDate, p\_CheckOutDate);

-- Добавление услуг в "RoomServices" с названием и стоимостью и расчет общей стоимости услуг

FOR v\_ServiceID IN SELECT UNNEST(p\_SelectedServices)

LOOP

SELECT "ServiceName", "ServiceCost" INTO v\_ServiceName, v\_ServiceCost

FROM "Services"

WHERE "ServiceID" = v\_ServiceID;

INSERT INTO "RoomServices" ("RoomNumber", "ServiceID", "ServiceName", "ServiceCost")

VALUES (v\_RoomNumber, v\_ServiceID, v\_ServiceName, v\_ServiceCost);

v\_ServicesTotalCost := v\_ServicesTotalCost + v\_ServiceCost;

END LOOP;

-- Получение подключенных услуг в виде строки

SELECT STRING\_AGG("ServiceName", ', ') INTO v\_ServiceDetails

FROM "Services"

WHERE "ServiceID" = ANY(p\_SelectedServices);

-- Расчет скидки

SELECT "Discount" INTO v\_Discount

FROM "Guests"

WHERE "GuestID" = p\_GuestID;

-- Расчет итоговой цены с учетом скидки

SELECT "Price" INTO v\_TotalPrice

FROM "Rooms"

WHERE "RoomNumber" = v\_RoomNumber;

-- Расчет конечной цены по новой формуле

v\_TotalPrice := (v\_TotalPrice + v\_ServicesTotalCost) \* (1 - (v\_Discount / 100));

-- Возвращение сводной информации

RETURN QUERY

SELECT

p\_GuestID AS "GuestID",

v\_RoomNumber AS "RoomNumber",

p\_CheckInDate AS "CheckInDate",

p\_CheckOutDate AS "CheckOutDate",

v\_ServiceDetails AS "ConnectedServices",

v\_TotalPrice AS "FinalPrice";

END;

$$ LANGUAGE plpgsql;

-- Вызов функции

SELECT \* FROM "ReserveRoom"(

p\_GuestID := 8,

p\_NumberOfBeds := 4,

p\_CheckInDate := '2024-06-10',

p\_CheckOutDate := '2024-06-15',

p\_SelectedServices := ARRAY[1, 2, 3] -- перечень необходимых услуг

Выполнение и результат данной функции представлен   
далее

|  |
| --- |
|  |
| 1. – Выполнение функции бронирования |

**Функция расчета скидки**:

CREATE OR REPLACE FUNCTION increment\_visit\_count()

RETURNS TRIGGER AS $$

BEGIN

-- Проверяем наличие совпадения по номеру телефона и дате рождения

IF EXISTS (SELECT 1 FROM "Guests"

WHERE "PhoneNumber" = NEW."PhoneNumber" AND "DateOfBirth" = NEW."DateOfBirth") THEN

-- Увеличиваем счетчик посещений для существующего клиента

UPDATE "Guests"

SET "VisitCount" = "VisitCount" + 1

WHERE "PhoneNumber" = NEW."PhoneNumber" AND "DateOfBirth" = NEW."DateOfBirth";

-- Отменяем вставку новой записи

RETURN NULL;

ELSE

-- Вставляем новую запись, если совпадений нет

RETURN NEW;

END IF;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger\_increment\_visit\_count

BEFORE INSERT ON "Guests"

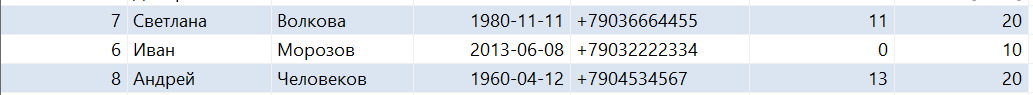
FOR EACH ROW

EXECUTE FUNCTION increment\_visit\_count();

-- Проверка триггера

INSERT INTO "Guests" values

(1, 'Алексей', 'Иванов', '1985-04-12', '+79031234567', 2, 0)



1. – Выполнение функции для подсчета скидки

**Функция вывода информации о номере**:

CREATE OR REPLACE FUNCTION GetFullRoomInformation(p\_RoomNumber INT)

RETURNS TABLE (

"RoomNumber" INT,

"Status" TEXT,

"Services" TEXT,

"TotalPrice" DECIMAL(10, 2)

) AS $$

BEGIN

RETURN QUERY

WITH "ServiceDetails" AS (

SELECT

"RS"."RoomNumber",

STRING\_AGG("S"."ServiceName", ', ') AS "Services",

SUM("S"."ServiceCost") AS "ServiceTotal"

FROM

"RoomServices" "RS"

JOIN "Services" "S" ON "RS"."ServiceID" = "S"."ServiceID"

WHERE

"RS"."RoomNumber" = p\_RoomNumber

GROUP BY

"RS"."RoomNumber"

),

"RoomStatus" AS (

SELECT

"R"."RoomNumber",

CASE

WHEN "R"."IsAvailable" THEN 'Свободна'

ELSE 'Занята'

END AS "Status",

"R"."Price" AS "BasePrice"

FROM

"Rooms" "R"

WHERE

"R"."RoomNumber" = p\_RoomNumber

)

SELECT

"R"."RoomNumber",

"R"."Status",

COALESCE("SD"."Services", 'Нет услуг') AS "Services",

("R"."BasePrice" + COALESCE("SD"."ServiceTotal", 0)) AS "TotalPrice"

FROM

"RoomStatus" "R"

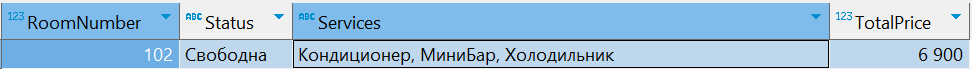
LEFT JOIN "ServiceDetails" "SD" ON "R"."RoomNumber" = "SD"."RoomNumber";

END;

$$ LANGUAGE plpgsql;

-- Вызов функции

SELECT \* FROM GetFullRoomInformation(104);



1. – Выполнение функции для вывода информации

## 4.7 Разработка необходимых триггеров

В данной работе, был создан, необходимый триггер - заключается он в том, чтобы автоматически увеличивать счетчик посещений клиента при повторном внесении его данных в базу.

INSERT INTO "Guests" values

(11,'Светлана', 'Волкова', '1980-11-11', '+79036664455', 0)

(10,'Алексей', 'Иванов', '1985-04-12', '+79031234567', 2)

select \*

from "Services" r r g

select \*

from "RoomServices"

select \*

from "Reservations"

where "GuestID" = 8

select \*

from "Guests"

ALTER TABLE "RoomServices"

add constraint "FK\_Rooms\_RoomServices"

FOREIGN KEY ("RoomNumber")

REFERENCES "Rooms"("RoomNumber");

select \*

from "Rooms"

delete from "Reservations"

where "RoomNumber" is null

UPDATE "Guests"

SET "VisitCount" = 9

WHERE "GuestID" = 7;

alter table "Guests"

rename column "V" numeric

select \*

from "Guests"

SELECT calculate\_discount(104);

UPDATE "Guests"

SET "Discount" = calculate\_discount("VisitCount", "DateOfBirth")

WHERE "GuestID" = 5;

SELECT \* FROM view\_available\_rooms(1);

# ЗАКЛЮЧЕНИЕ

Проект позволил закрепить знания по проектированию реляционных баз данных в системе управления базами данных PostgreSQL. База данных для Гостиницы позволит эффективно получать информацию о номерах и подключенных в них услугах, производить расчет скидки , предоставляемой клиенту, выполнять поиск свободных номеров по заданному критерию , а также позволит производить процедуру бронирования номеров с учетом пожеланий клиента.

P.s. Полученные и закрепленные в результате выполнения данной курсовой работы знания помогут мне сэкономить если я захочу открыть гостиницу 😊 .

Выводы о проделанной работе не моя сильная сторона , поэтому спасибо за внимание)