Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

Факультет цифровой экономики и массовых коммуникаций

Кафедра «Прикладная информатика»

Работа принята на проверку Работа допущена к защите

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_20\_\_ года «\_\_\_» \_\_\_\_\_\_\_\_\_20\_\_ года

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО, подпись руководителя) (ФИО, подпись руководителя)

**КУРСОВАЯ РАБОТА**по дисциплине «Проектирование баз данных»  
Тема: «Разработка базы данных справочной аэропорта»

ВЫПОЛНИЛ

студент 2 курса факультета \_\_\_\_

группы \_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_

(инициалы и фамилия) (подпись)

Работа защищена с оценкой

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2023 года

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
(ФИО, подпись руководителя)

САМАРА 2023 г.

**РЕЦЕНЗИЯ**

**на курсовую работу**

|  |  |
| --- | --- |
| Студента | Морзюкова Максима Андреевича |

|  |  |
| --- | --- |
| Рецензент | Герасимов Вячеслав Васильевич |

В рецензии отражается соответствие курсовой работы индивидуальному заданию, глубина проработки поставленных вопросов. Выявляются положительные стороны проделанной работы, а также ее недостатки.

Результат рецензирования: допущен/не допущен к защите.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (ФИО рецензента)

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 4](#_Toc152893519)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc152893520)

[2. ПОСТАНОВКА ЗАДАЧИ 6](#_Toc152893521)

[3. ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ 12](#_Toc152893522)

[3.1 КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ 12](#_Toc152893523)

[3.2 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ 15](#_Toc152893524)

[3.3 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ 20](#_Toc152893525)

[4. РЕАЛИЗАЦИЯ ПРОЕКТА БД СРЕДСТВАМИ СУБД 22](#_Toc152893526)

[5. ЗАПРОСЫ К БД 31](#_Toc152893527)

[6. ЗАКЛЮЧЕНИЕ 42](#_Toc152893528)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 42](#_Toc152893529)

# **ВВЕДЕНИЕ**

**Тема курсового проекта:** Разработка базы данных справочной аэропорта .

**Актуальность темы:** В справочные аэропортов часто звонят, что бы узнать информацию о расписании рейсов. Гораздо удобнее следить за

рейсами, билетам и пассажирами с помощью базы данных.

**Цель проектирования:** Разработка базы данных.

**Предметом исследования** является создание и проектирование базы данных.

**Гипотеза исследования:** если к созданию и проектированию базы данных применить методы разработки и методы анализирования, то можно значительно уменьшить дополнительные расходы, связанные с созданием БД.

**Теоретическая значимость:** теоретическая часть помогает познакомиться с БД и углубиться в суть темы.

**Практическая значимость** состоит в том, что с помощью расчётов, построения модели, анализа можно спроектировать БД.

**Методы исследования:** подбор, анализ, расчёт, сопоставление и построение БД.

**Структура проекта** соответствует логике и включает в себя введение, этап проектирования, этап построения, выбора аппаратного обеспечения, выводы, заключение, список источников и литературы.

Морзюков М.А. Вариант 15: разработка базы данных справочной аэропорта

# **1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

База данных справочной аэропорта для удобного хранения информации. Данная база предоставляет информацию о рейсах, билетов, пассажиров, gate-ов, авиакомпаний, полосах, самолетов, льгот и бронирований. В соответствии с этой информацией, составляются выводы о том, как пройти на какой-либо рейс, время вылета и прилета, стоимости билета и т.д.

База данных справочной аэропорта работает по следующему алгоритму:

1. Клиент обращается в справочную с каким-то вопросом.
2. Работник ищет нужную информацию в базе данных.
3. Клиенту предоставляется интересующая его информация.

Данная база данных разработана для хранения и обработки данных о рейсах, билетов и т.д. Это облегчит поиск нужной информации для работников, что положительно скажется на времени ответов и удобства для них самих.

Таким образом, в БД справочной аэропорта должны быть таблицы, содержащие следующую информацию:

1. Информация о рейсах;
2. Информация о билетах и бронировании;
3. Информация о полосах и gate-ов;
4. Информация о пассажирах и льготах;
5. Информация о самолетах и авиакомпаниях;

**2. ПОСТАНОВКА ЗАДАЧИ**

Задача курсовой работы состоит в том, чтобы создать систему управления базой данных, которая должна добавлять, редактировать информацию о заказах, клиентах, сотрудниках, изделиях и материалах.

В данной курсовой работе были рассмотрены пять наиболее популярных систем управления БД:

1. Oracle;
2. MySQL;
3. Microsoft SQL Server;
4. PostgreSQL;
5. MongoDB.

Oracle (Oracle Database) – эта система популярна у разработчиков, проста в использовании, у неё понятная документация, поддержка длинных наименований, разработчик Oracle Corporation, написана на Assembly, C, C++.

MySQL работает на Linux, Windows, MacOS, FreeBSD, Solaris и Unix-подобных системах. Можно начать работать с бесплатным сервером, а затем перейти на коммерческую версию. Лицензия GPL с открытым исходным кодом позволяет модифицировать ПО MySQL, разработчик MySQL AB, Sun Microsystems и Oracle, написана на С и С++.

Microsoft SQL Server – это популярная коммерческая СУБД. Она привязана к Windows. Графический интерфейс и программное обеспечение основаны на командах. Поддерживает SQL, непроцедурные, нечувствительные к регистру и общие языки баз данных, разработчик Microsoft Corporation, написана на С, С++.

PostgreSQL – это масштабируемая объектно-реляционная БД, работает на Linux, MacOS, Windows, OpenBSD, FreeBSD и Solaris. Имеет такие функции, как логическая репликация, декларативное разбиение таблиц, улучшенные параллельные запросы, более безопасная аутентификация по паролю. Разработчик PostgreSQL Global Development, написана на С.

MongoDB – это самая популярная NoSQL СУБД. Поддерживает Linux, MacOS и Windows, но размер БД ограничен 2.5ГБ в 32-битных системах, разработана MongoDB Inc. В 2007 году, написана на C++.

Для выбора подходящей СУБД рассмотрим таблицу особенностей СБУД (Таблица 1.)

Особенности СУБД Таблица 1.

|  |  |
| --- | --- |
| **Наименование СУБД** | **Особенности** |
| Oracle | * Обрабатывает большие данные; * Поддерживает SQL, можно получить доступ из реляционных БД Oracle; * Oracle NoSQL DataBase Java/C API для чтения и записи данных. |
| MySQL | * Масштабируемость; * Лёгкость в использовании; * Безопасность; * ПоддержкаNovell Cluster; * Скорость; * Поддержка множества ОС. |
| Microsoft SQL Server | * Высокая производительность; * Зависимость от платформы; * Генерация скриптов для перемещения данных. |
| PostgreSQL | * Поддержка табличных пространств; * Восстановление на момент времени (PITR); * Асинхронная репликация. |
| MongoDB | * Работа на нескольких серверах; * Возможность индексировать все поля в документе; * Поддержка поиска по регулярным выражениям. |

Таким образом, по данным сравнения из таблицы особенностей СУБД (Таблица 1) отличным вариантом для управления БД является MySQL, так как данная система содержит в себе множество функций для управления БД, обладает высокой скоростью работы, безопасна и к тому же легка в использовании.

Для СУБД курсовой работы необходимо выделить входную и выходную информацию.

Входной информацией будут выступать числовые и буквенные данные. К числовым данным относятся цены билетов, контактные данные пассажиров и аэропортов, количество мест в самолетах, время бронирования и время вылета и прилета. К буквенной информации относятся личные данные пассажиров, сведения о рейсах.

Выходной информацией будут выступать числовые и буквенные данные. К числовым данным относятся дата и время вылета и прилета, цены билетов, количество мест в самолетах. К буквенным данным относятся сведения о рейсах, gate-ах и полосах.

Данная база будет отображать следующую информацию:

1. Информацию о рейсе: авиакомпания, самолет, gate, полоса, время вылета и прилета;
2. ФИО пассажиров, документы и их контактные данные;
3. Информацию о билетах: рейс, цена, место, льгота, статус, бронь;

В курсовой работе будут использованы следующие технические термины:

1. Система управления базами данных (СУБД) – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных;
2. Права доступа – совокупность правил, регламентирующих порядок и условия доступа объекта к объектам информационной системы;
3. Разработка программного обеспечения - действия по созданию нового программного продукта (программирование, тестирование, разработка документации и др.).

К курсовой работе представлен перечень запросов (не менее 10 запросов),

состоящий из следующих типов запросов:

1. С использованием внутреннего соединения по одному полю;
2. С использованием косвенно связанных таблиц;
3. С использованием таблиц, связанных более чем по одному полю;
4. С использованием внешнего соединения таблиц;
5. С использованием рекурсивного соединения;
6. С использованием соединений по отношению;
7. С использованием функций агрегирования;
8. Перекрестный запрос;
9. Запрос на изменение;
10. С вычисляемым полем.

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

1. Введение
   1. Наименование программы: «БД справочной аэропорта».
2. Краткая характеристика области применения программы: данная база предоставляет информацию о аэропорте, рейсах, пассажирах и билетах.
3. Основание для разработки: Учебный план по направлению 09.03.04 «Программная инженерия» (бакалавры).
4. Назначение разработки: БД предназначена для хранения информации о аэропорте, которая может быть интересна клиенту.
5. Требования, предъявляемые к программе
   1. Требования к функциональным характеристикам программы.

В программе должны быть реализованы следующие операции:

* + 1. Добавление информации о рейсах;
    2. Добавление информации о пассажирах;
    3. Учёт билетов;
    4. Отображение общей информации о рейсах.
  1. Требования к техническим средствам, используемым при работе программы.

Работа программы должна осуществляться на персональном компьютере.

* 1. Требования к языкам программы и среде разработки программы.

В курсовой работе не предусмотрена отладка и/или запись программного кода.

* 1. Требования к информационным структурам на входе и выходе программы.

Входной информацией будут выступать числовые и буквенные данные. К числовым данным относятся цены билетов, контактные данные пассажиров и аэропортов, количество мест в самолетах, время бронирования и время вылета и прилета. К буквенной информации относятся личные данные пассажиров, сведения о рейсах.

Выходной информацией будут выступать числовые и буквенные данные. К числовым данным относятся дата и время вылета и прилета, цены билетов, количество мест в самолетах. К буквенным данным относятся сведения о рейсах, gate-ах и полосах.

* 1. Специальные требование. База данных данной программы будет разработана СУБД - MySQL.

1. Требования к программной документации. Состав программной документации должен включать в себя:
   * 1. Техническое задание;
     2. Описание программы;
     3. Текст программы.
2. Этапы разработки.
   * 1. Обзор способов организации данных и обоснование выбора СУБД для эффективного выполнения операций 24.11.2023 - 1.12.2023.
     2. Разработка базы данных: 24.11.2023 - 1.12.2023.
     3. Разработка программной документации: 1.12.2023 - 08.12.2023.
     4. Оформление пояснительной 8.12.2023 - 8.12.2023.
     5. Защита курсовой работы: 15.12.2023 - 22.12.2023.

# **3. ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ**

## **3.1 КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

В данном разделе отображены сущности и атрибуты для последующего создания ER-модели.

Таблица «Рейс» Таблица 2.

|  |  |
| --- | --- |
| ID\_Рейса | ID Рейса |
| ID\_Авиакомпании | ID Авиакомпании |
| ID\_Gate | ID Gate |
| ID\_Полосы | ID Полосы |
| ID\_Самолета | ID Самолета |
| Время вылета | Время вылета |
| Время прилета | Время прилета |

Таблица «Билеты» Таблица 3.

|  |  |
| --- | --- |
| ID\_Билета | ID Билета |
| ID\_Рейса | ID Рейса |
| Цена | Цена |
| Место | Место |
| ID\_Пассажира | ID Пассажира |
| Оплачен | Оплачен |
| ID\_Льготы | ID Льготы |

Таблица «Пассажиры» Таблица 4.

|  |  |
| --- | --- |
| ID\_Пассажира | ID Пассажира |
| Фамилия | Фамилия |
| Имя | Имя |
| Отчество | Отчество |
| Телефон | Телефон |
| Номер Паспорта | Номер Паспорта |

Таблица «Gate» Таблица 5.

|  |  |
| --- | --- |
| ID\_Gate | ID Gate |
| Название | Название |

Таблица «Авиакомпании» Таблица 6.

|  |  |
| --- | --- |
| ID\_Авиакомпании | ID Авиакомпании |
| Название | Название |
| Телефон | Телефон |

Таблица «Полосы» Таблица 7.

|  |  |
| --- | --- |
| ID\_Полосы | ID Полосы |
| Название | Название |

Таблица «Самолеты» Таблица 8.

|  |  |
| --- | --- |
| ID\_Самолета | ID Самолета |
| Модель | Модель |
| Вместимость | Вместимость |

Таблица «Бронирование» Таблица 9.

|  |  |
| --- | --- |
| ID\_ Бронирования | ID Бронирования |
| ID\_Билета | ID Билета |
| Время брони | Время брони |

Таблица «Льготы» Таблица 10.

|  |  |
| --- | --- |
| ID\_Льготы | ID Льготы |
| Название | Название |

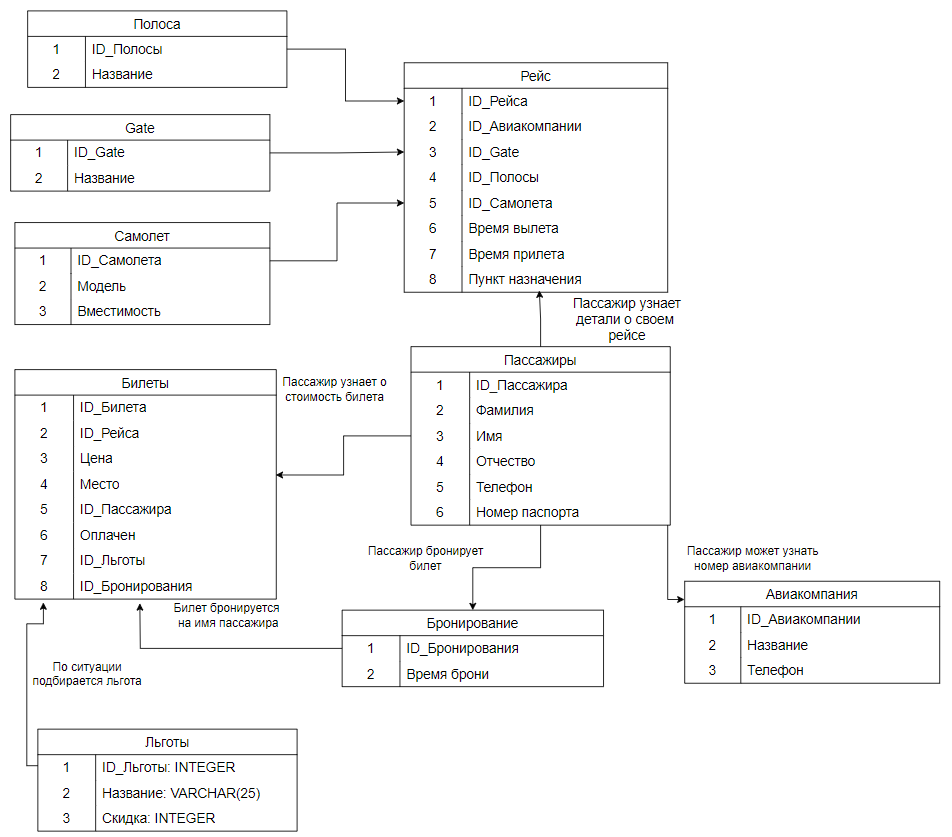
Система, в соответствии с предметной областью, строится с учётом следующих особенностей:

1. У одного пассажира может быть несколько билетов;
2. У одних gate, полосы, авиакомпаний и самолетов может быть много рейсов.
3. Один самолет занимает всю полосу до момента вылета. После прилета ему необходимо вернуться перед тем, как его можно будет использовать в следующем рейсе.

Также определим ключи сущностей:

1. ID\_Рейса – первичный ключ сущности «Рейс»;
2. ID\_Авиакомпании – первичный ключ сущности «Авиакомпания»;
3. ID\_Gate – первичный ключ сущности «Gate»;
4. ID\_Полосы – первичный ключ сущности «Полоса»;
5. ID\_Самолета – первичный ключ сущности «Самолет»;
6. ID\_Билета – первичный ключ сущности «Билеты»;
7. ID\_Пассажира – первичный ключ сущности «Пассажиры».
8. ID\_Льготы – первичный ключ сущности «Льготы».
9. ID\_ Бронирования – первичный ключ сущности «Бронирование».

Построение концептуальной модели БД справочной аэропорта (Рисунок 1).

 Рисунок 1 – Концептуальная модель БД справочной аэропорта

## **3.2 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ**

В данном разделе отображено преобразование ER-диаграммы в схему реляционной базы данных.

Реляционная база данных представлена в третьей нормальной форме. Чтобы проверить верность данного утверждения, проведем анализ реляционной базы данных.

Формирование отношений, для дальнейшей нормализации отношений:

1. Gate -> Рейс (1: N). Формируется одна связь: Рейс[Gate\_IDGate]
2. Авиакомпания -> Рейс (1: N). Формируется одна связь: Рейс [Авиакомпания\_IDАвикомпании]
3. Полоса -> Рейс (1: N). Формируется одна связь: Рейс [Полоса\_IDПолосы]
4. Самолет -> Рейс (1. N). Формируется одна связь: Рейс [Самолет\_IDСамолета]
5. Билеты -> Рейс (1: N). Формируется одна связь: Рейс [Билет\_IDБилета]
6. Бронирование -> Билеты (1:1). Формируется одна связь: Билеты [Бронирование\_IDБронирования]
7. Льготы -> Билеты (1:1). Формируется одна связь: Билеты [Льготы\_IDЛьготы]
8. Пассажиры -> Билеты (1:N). Формируется одна связь: Билеты [Пассажиры\_IDПассажира]

Понятие нормализации. Нормализация — это метод проектирования базы данных, который сводит к минимуму избыточность в базе данных.

Представленные выше связи нормализованы (Рисунок 2).

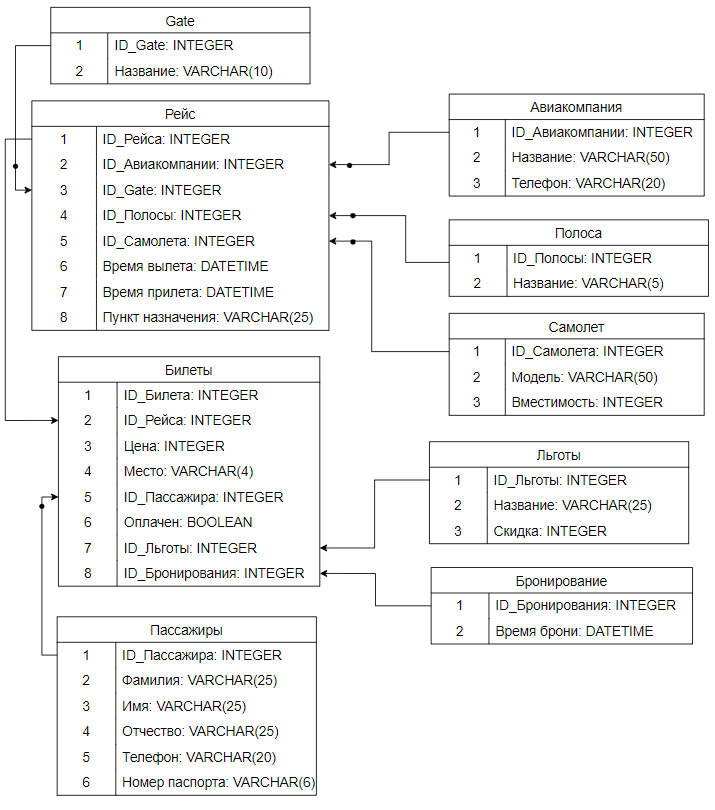


Рисунок 2 – ER-модель БД справочной аэропорта

**ПРИВЕДЕНИЕ К НОРМАЛЬНЫМ ФОРМАМ**

Первая нормальная форма (1НФ): Все атрибуты должны быть атомарными и не повторяющимися. Все таблицы в нашей базе данных уже удовлетворяют этому требованию.

Как пример: Список льгот хранится в отдельной таблице и связан с таблицей билетов, где определенную льготу можно выбрать по ID (Рисунок 3).

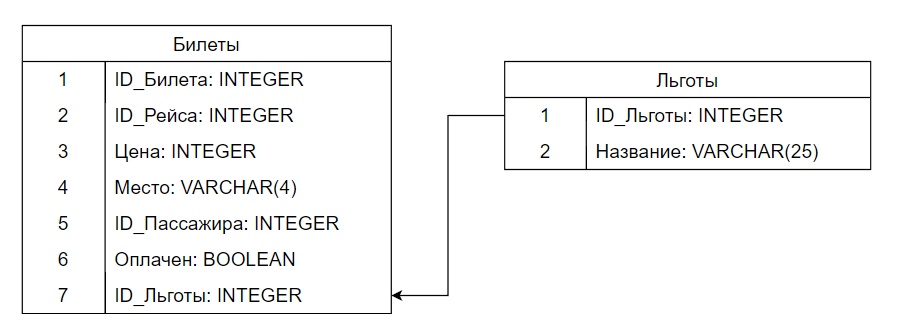


Рисунок 3 – Связь таблиц Льгот и Билетов

Вторая нормальная форма (2НФ): должно соблюдаться условие первой нормальной формы и поля с не первичным ключом не должны быть зависимы от первичного ключа.

В таблице Билеты видна реализация 2НФ. Вся информация о пассажире, с которым связан билет, хранится в отдельной таблице, эти две таблицы соединены связью 1 к м и по ней можно получить информацию о пассажире (Рисунок 4).

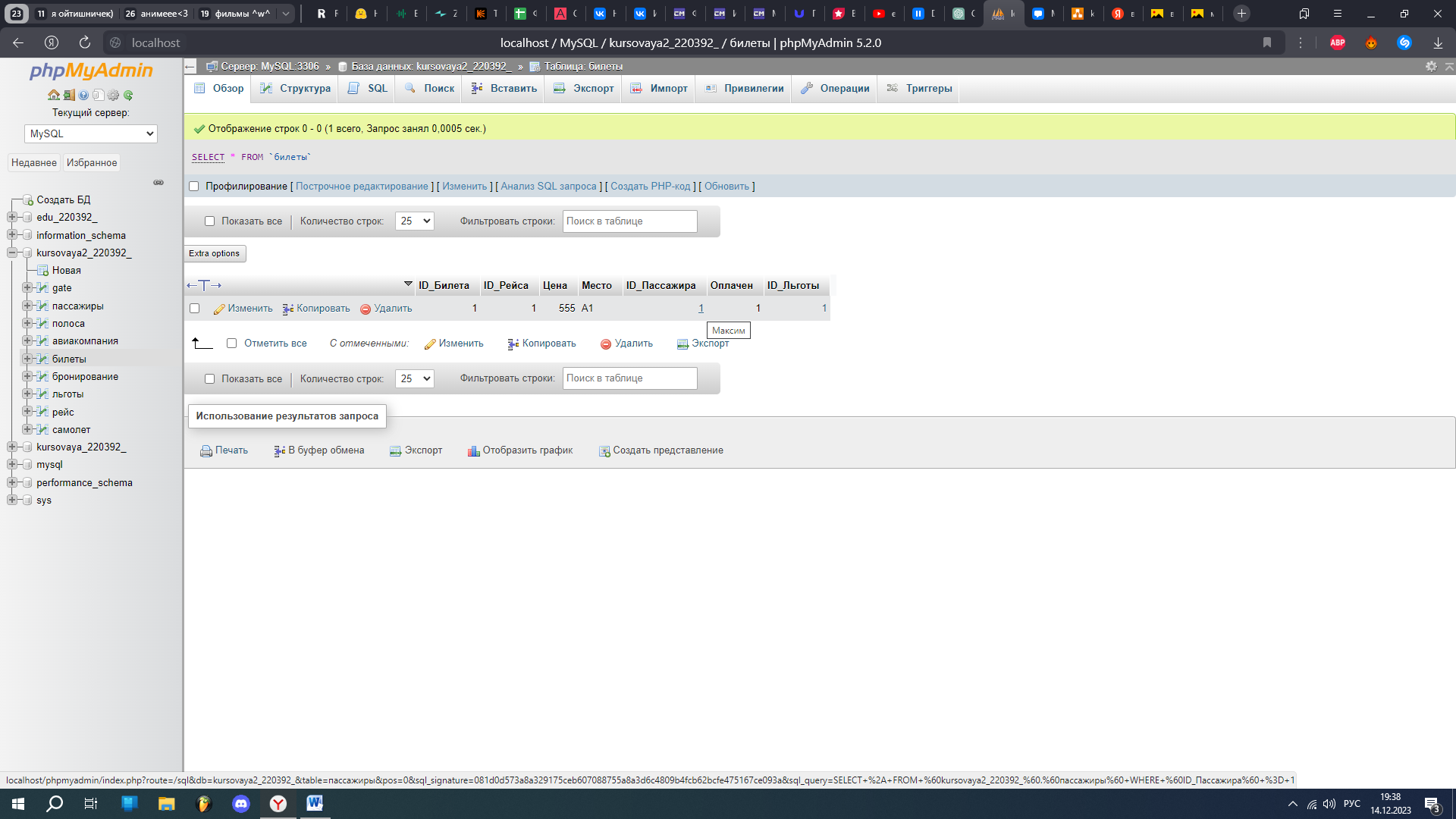


Рисунок 4 – Таблица билеты и ссылка на таблицу Пассажиры

Чтобы нормализовать базу данных до 3НФ, необходимо сделать так, что бы в таблицах отсутствовали не ключевые столбцы, которые зависят от других не ключевых столбцов.

Иными словами, не ключевые столбцы не должны пытаться играть роль ключа в таблице, т.е. они действительно должны быть не ключевыми столбцами, такие столбцы не дают возможности получить данные из других столбов, они дают возможность посмотреть на информацию, которая в них содержится, так как в этом их назначение. (Рисунок 5).

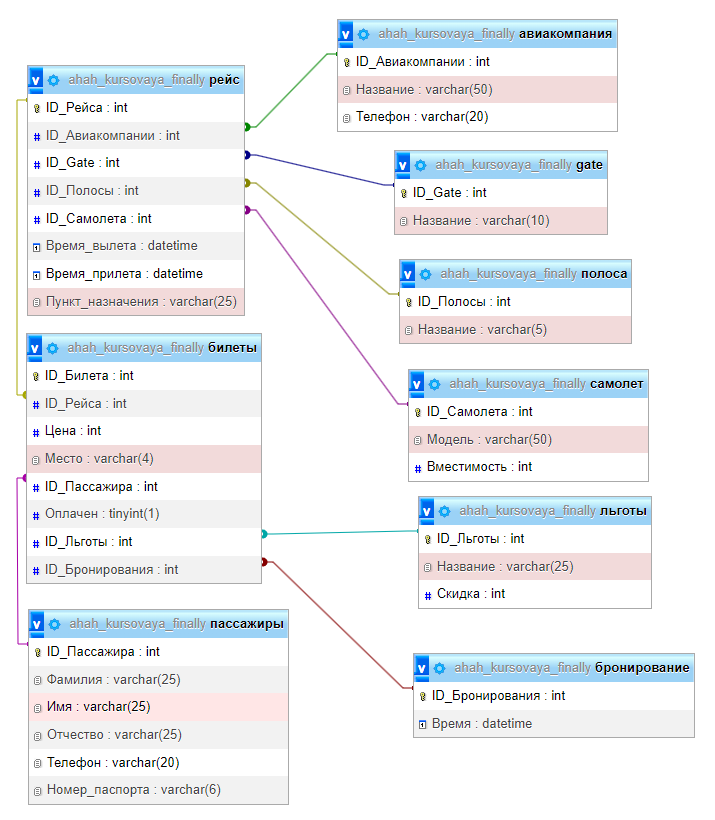


Рисунок 5 – Приведение к 3НФ

## **3.3 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ**

В данном разделе представлены таблицы, соответствующие разрабатываемой базе данных, с указанием количества текстовых полей и символов, а также типы данных (Таблица 9 – Таблица 15).

Таблица «Рейс» Таблица 10.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Рейса | INTEGER |
| ID\_Авиакомпании | INTEGER |
| ID\_Gate | INTEGER |
| ID\_Полосы | INTEGER |
| ID\_Самолета | INTEGER |
| Время вылета | DATETIME |
| Время прилета | DATETIME |
| Пункт назначения | VARCHAR(25) |

Таблица «Билеты» Таблица 11.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Билета | INTEGER |
| ID\_Рейса | INTEGER |
| Цена | INTEGER |
| Место | VARCHAR(4) |
| ID\_Пассажира | INTEGER |
| Оплачен | BOOLEAN |
| ID\_Льготы | INTEGER |
| ID\_Бронирования | INTEGER |

Таблица «Пассажиры» Таблица 12.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Пассажира | INTEGER |
| Фамилия | VARCHAR(25) |
| Имя | VARCHAR(25) |
| Отчество | VARCHAR(25) |
| Телефон | VARCHAR(20) |
| Номер паспорта | VARCHAR(6) |

Таблица «Gate» Таблица 13.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Gate | INTEGER |
| Название | VARCHAR(10) |

Таблица «Авиакомпания» Таблица 14.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Авиакомпании | INTEGER |
| Название | VARCHAR(50) |
| Телефон | VARCHAR(20) |

Таблица «Полоса» Таблица 15.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Полосы | INTEGER |
| Название | VARCHAR(5) |

Таблица «Самолет» Таблица 16.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Самолета | INTEGER |
| Модель | VARCHAR(50) |
| Вместимость | INTEGER |

Таблица «Бронирование» Таблица 17.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Бронирования | INTEGER |
| Время брони | DATETIME |

Таблица «Льготы» Таблица 18.

|  |  |
| --- | --- |
| **Название атрибута** | **Тип, длина** |
| ID\_Льготы | INTEGER |
| Название | VARCHAR(25) |
| Скидка | INTEGER |

Из сведений, указанных в пункте «Проектирование реляционной базы данных», выделим несколько ограничений для БД:

1. Один самолет занимает всю полосу до момента вылета. После прилета ему необходимо вернуться перед тем, как его можно будет использовать в следующем рейсе;
2. Ограничение на типы данных: каждое поле таблицы должно иметь определённый тип данных.

Построим полученную базу данных с учётом всех ограничений и связей (Рисунок 6).

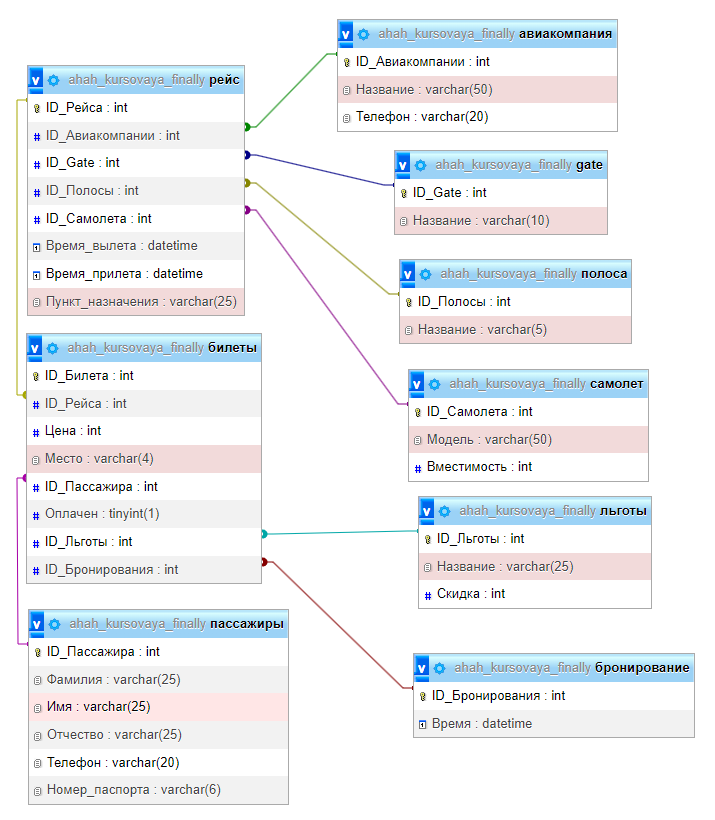


Рисунок 6 – Полученная БД справочной аэропорта

# **4. РЕАЛИЗАЦИЯ ПРОЕКТА БД СРЕДСТВАМИ СУБД**

Создание SQL-запросов, согласно пункту «Постановка задачи»:

1. Описание таблицы авиакомпаний

CREATE TABLE Авиакомпания (

ID\_Авиакомпании INTEGER PRIMARY KEY,

Название VARCHAR(50),

Телефон VARCHAR(20)

) ENGINE=InnoDB;

INSERT INTO Авиакомпания (ID\_Авиакомпании, Название, Телефон)

VALUES

(1, 'Аэрофлот', '74952235555'),

(2, 'S7 Airlines', '74957779999'),

(3, 'Utair', '78002345678'),

(4, 'Россия', '74951234567'),

(5, 'Ural Airlines', '73433456789'),

(6, 'Pobeda', '78001234567'),

(7, 'Yakutia Airlines', '74112223344'),

(8, 'Aurora', '74235556677'),

(9, 'Gazpromavia', '74958887766'),

(10, 'Iraero', '73834445566');

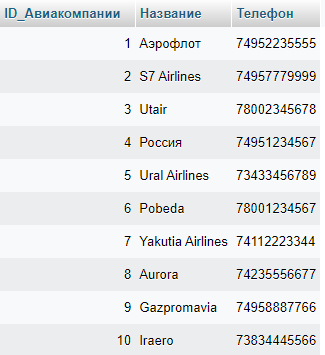


Рисунок 7 – Таблица авиакомпаний

1. Описание таблицы полос

CREATE TABLE Полоса (

ID\_Полосы INT PRIMARY KEY,

Название VARCHAR(5)

)ENGINE=InnoDB;

INSERT INTO Полоса (ID\_Полосы, Название)

VALUES

 (1, 'Пол-1'),

(2, 'Пол-2'),

(3, 'Пол-3'),

(4, 'Пол-4'),

(5, 'Пол-5'),

(6, 'Пол-6'),

(7, 'Пол-7'),

(8, 'Пол-8'),

(9, 'Пол-9'),

(10, 'Пол-10'),

(11, 'Пол-11'),

(12, 'Пол-12'),

(13, 'Пол-13'),

(14, 'Пол-14'),

(15, 'Пол-15'),

(16, 'Пол-16'),

(17, 'Пол-17'),

(18, 'Пол-18'),

(19, 'Пол-19'),

(20, 'Пол-20');

Рисунок 8 – Таблица полос

1. Описание таблицы gate-ов

CREATE TABLE Gate (

ID\_Gate INT PRIMARY KEY,

Название VARCHAR(10)

) ENGINE=InnoDB;

INSERT INTO Gate (ID\_Gate, Название)

VALUES

(1, 'Gate-1'),

(2, 'Gate-2'),

(3, 'Gate-3'),

(4, 'Gate-4'),

(5, 'Gate-5'),

(6, 'Gate-6'),

(7, 'Gate-7'),

(8, 'Gate-8'),

(9, 'Gate-9'),

(10, 'Gate-10');



Рисунок 9 – Таблица gate-ов

1. Описание таблицы самолетов

CREATE TABLE Самолет (

ID\_Самолета INT PRIMARY KEY,

Модель VARCHAR(50),

Вместимость INTEGER

) ENGINE=InnoDB;

INSERT INTO Самолет (ID\_Самолета, Модель, Вместимость)

VALUES

(1, 'Boeing 737', 150),

(2, 'Airbus A320', 180),

(3, 'Boeing 777', 300),

(4, 'Airbus A330', 250),

(5, 'Embraer E190', 100),

(6, 'Bombardier CRJ-700', 70),

(7, 'Boeing 747', 400),

(8, 'Airbus A380', 550),

(9, 'Boeing 757', 200),

(10, 'Boeing 767', 250),

(11, 'Airbus A350', 300),

(12, 'Embraer E175', 80),

(13, 'Bombardier Q400', 90),

(14, 'Boeing 787', 280),

(15, 'Airbus A340', 290),

(16, 'Boeing 717', 120),

(17, 'Embraer E195', 110),

(18, 'Airbus A321', 220),

(19, 'Boeing 727', 170),

(20, 'Bombardier CRJ-900', 90);

Рисунок 10 – Таблица самолетов.

1. Описание таблицы рейсов

CREATE TABLE Рейс (

ID\_Рейса INTEGER PRIMARY KEY,

ID\_Авиакомпании INTEGER,

ID\_Gate INTEGER,

ID\_Полосы INTEGER,

ID\_Самолета INTEGER,

Время\_вылета DATETIME,

Время\_прилета DATETIME,

Пункт\_назначения VARCHAR(25),

FOREIGN KEY (ID\_Авиакомпании) REFERENCES Авиакомпания(ID\_Авиакомпании),

FOREIGN KEY (ID\_Gate) REFERENCES Gate(ID\_Gate),

FOREIGN KEY (ID\_Полосы) REFERENCES Полоса(ID\_Полосы),

FOREIGN KEY (ID\_Самолета) REFERENCES Самолет(ID\_Самолета)

) ENGINE=InnoDB;

INSERT INTO Рейс (ID\_Рейса, ID\_Авиакомпании, ID\_Gate, ID\_Полосы, ID\_Самолета, Время\_вылета, Время\_прилета, Пункт\_назначения)

VALUES

(1, 1, 1, 1, 1, '2023-12-14 12:00:00', '2023-12-14 15:00:00', 'Нью-Йорк'),

(2, 2, 2, 2, 2, '2023-12-15 14:30:00', '2023-12-15 17:30:00', 'Лондон'),

(3, 3, 3, 3, 3, '2023-12-16 10:00:00', '2023-12-16 13:00:00', 'Токио'),

(4, 1, 1, 1, 4, '2023-12-17 08:45:00', '2023-12-17 11:45:00', 'Париж'),

(5, 2, 2, 2, 5, '2023-12-18 16:20:00', '2023-12-18 19:20:00', 'Рим'),

(6, 3, 3, 3, 6, '2023-12-19 11:30:00', '2023-12-19 14:30:00', 'Пекин'),

(7, 1, 1, 1, 7, '2023-12-20 14:00:00', '2023-12-20 17:00:00', 'Москва'),

(8, 2, 2, 2, 8, '2023-12-21 09:15:00', '2023-12-21 12:15:00', 'Берлин'),

(9, 3, 3, 3, 9, '2023-12-22 18:45:00', '2023-12-22 21:45:00', 'Дубай'),

(10, 1, 1, 1, 10, '2023-12-23 07:30:00', '2023-12-23 10:30:00', 'Сан-Франциско'),

(11, 2, 2, 2, 11, '2023-12-24 13:10:00', '2023-12-24 16:10:00', 'Сидней'),

(12, 3, 3, 3, 12, '2023-12-25 15:45:00', '2023-12-25 18:45:00', 'Санкт-Петербург'),

(13, 1, 1, 1, 13, '2023-12-26 20:30:00', '2023-12-26 23:30:00', 'Шанхай'),

(14, 2, 2, 2, 14, '2023-12-27 12:05:00', '2023-12-27 15:05:00', 'Киев'),

(15, 3, 3, 3, 15, '2023-12-28 09:20:00', '2023-12-28 12:20:00', 'Дели'),

(16, 1, 1, 1, 16, '2023-12-29 16:40:00', '2023-12-29 19:40:00', 'Стамбул'),

(17, 2, 2, 2, 17, '2023-12-30 08:55:00', '2023-12-30 11:55:00', 'Торонто'),

(18, 3, 3, 3, 18, '2023-12-31 14:15:00', '2023-12-31 17:15:00', 'Мельбурн'),

(19, 1, 1, 1, 19, '2024-01-01 11:00:00', '2024-01-01 14:00:00', 'Копенгаген'),

(20, 2, 2, 2, 20, '2024-01-02 17:30:00', '2024-01-02 20:30:00', 'Осло');



Рисунок 11 – Таблица рейсов.

1. Описание таблицы пассажиров

CREATE TABLE Пассажиры (

ID\_Пассажира INT PRIMARY KEY,

Фамилия VARCHAR(25),

Имя VARCHAR(25),

Отчество VARCHAR(25),

Телефон VARCHAR(20),

Номер\_паспорта VARCHAR(6)

) ENGINE=InnoDB;

INSERT INTO Пассажиры (ID\_Пассажира, Фамилия, Имя, Отчество, Телефон, Номер\_паспорта)

VALUES

(1, 'Иванов', 'Иван', 'Иванович', '1234567890', '123456'),

(2, 'Петров', 'Петр', 'Петрович', '9876543210', '567890'),

(3, 'Сидоров', 'Сидор', 'Сидорович', '5551112233', '910111'),

(4, 'Алексеев', 'Алексей', 'Алексеевич', '9998887777', '112233'),

(5, 'Козлов', 'Козел', 'Козлович', '7778889999', '334455'),

(6, 'Николаев', 'Николай', 'Николаевич', '1112223334', '556677'),

(7, 'Григорьев', 'Григорий', 'Григорьевич', '4445556667', '778899'),

(8, 'Дмитриев', 'Дмитрий', 'Дмитриевич', '6667778889', '990011'),

(9, 'Андреев', 'Андрей', 'Андреевич', '2223334445', '112233'),

(10, 'Васнецов', 'Василий', 'Васнецович', '8889990000', '334455'),

(11, 'Павлов', 'Павел', 'Павлович', '1230004567', '556677'),

(12, 'Федоров', 'Федор', 'Федорович', '9871110000', '778899'),

(13, 'Герасимов', 'Герасим', 'Герасимович', '1115556666', '990011'),

(14, 'Артемьев', 'Артем', 'Артемьевич', '2223334444', '112233'),

(15, 'Михайлов', 'Михаил', 'Михайлович', '3334445555', '334455'),

(16, 'Евгеньев', 'Евгений', 'Евгеньевич', '4445556666', '556677'),

(17, 'Сергеев', 'Сергей', 'Сергеевич', '5556667777', '778899'),

(18, 'Александров', 'Александр', 'Александрович', '6667778888', '990011'),

(19, 'Кириллов', 'Кирилл', 'Кириллович', '7778889999', '112233'),

(20, 'Денисов', 'Денис', 'Денисович', '8889990000', '334455');

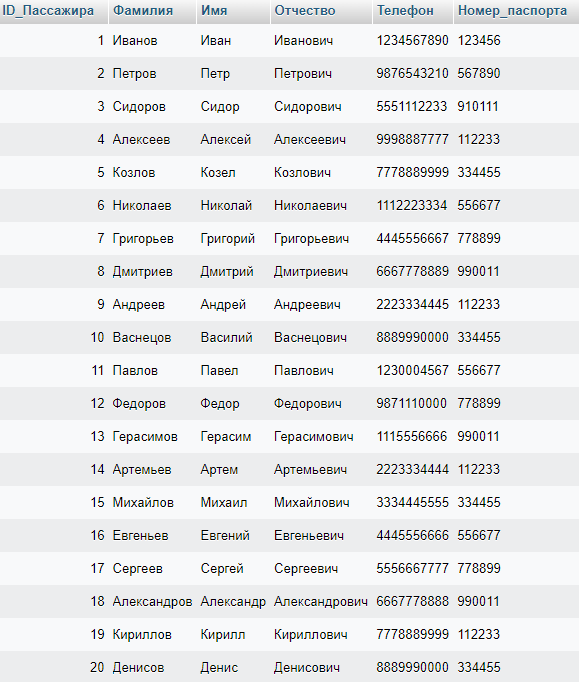


Рисунок 12 – Таблица пассажиров

1. Описание таблицы Льгот

CREATE TABLE Льготы (

ID\_Льготы INT PRIMARY KEY,  
Название VARCHAR(25),

Скидка INT

) ENGINE=InnoDB;

INSERT INTO Льготы (ID\_Льготы, Название, Скидка)

VALUES

(1, 'Студенческая', 20),

(2, 'Пенсионерская', 15),

(3, 'Детская', 30),

(4, 'Инвалиды', 50),

(5, 'Ветераны', 25),

(6, 'Семейная', 10),

(7, 'Групповая', 15),

(8, 'Специальная', 40),

(9, 'Постоянный клиент', 5),

(10, 'Праздничная', 20);

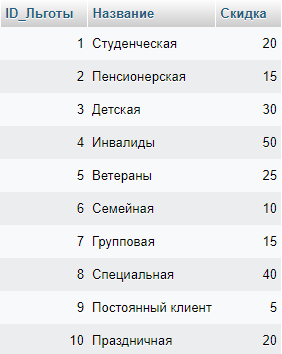
****

Рисунок 13 – Таблица льгот.

1. Описание таблицы бронирования

CREATE TABLE Бронирование (

ID\_Бронирования INT PRIMARY KEY,  
Время DATETIME

) ENGINE=InnoDB;

INSERT INTO Бронирование (ID\_Бронирования, Время)

VALUES

(1, '2023-11-14 14:30:00'),

(2, '2023-11-15 16:45:00'),

(3, '2023-11-16 10:00:00'),

(4, '2023-11-17 12:15:00'),

(5, '2023-11-18 18:30:00');

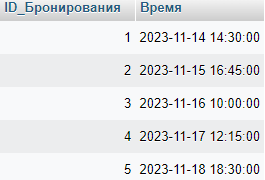


Рисунок 14 – Таблица бронирования.

1. Описание таблицы билетов

CREATE TABLE Билеты (

ID\_Билета INT PRIMARY KEY,

ID\_Рейса INT,  
Цена INT,

Место VARCHAR(4),

ID\_Пассажира INT,

Оплачен BOOLEAN,

ID\_Льготы INT NULL,

ID\_Бронирования INT NULL,

FOREIGN KEY (ID\_Рейса) REFERENCES Рейс(ID\_Рейса),

FOREIGN KEY (ID\_Пассажира) REFERENCES Пассажиры(ID\_Пассажира),

FOREIGN KEY (ID\_Льготы) REFERENCES Льготы(ID\_Льготы),

FOREIGN KEY (ID\_Бронирования) REFERENCES Бронирование(ID\_Бронирования)

) ENGINE=InnoDB;

INSERT INTO Билеты (ID\_Билета, ID\_Рейса, Цена, Место, ID\_Пассажира, Оплачен, ID\_Льготы, ID\_Бронирования)

VALUES

(1, 1, 1500, 'A01', 1, TRUE, 2, 1),

(2, 1, 1500, 'A02', 2, TRUE, NULL, 2),

(3, 2, 1200, 'B01', 3, TRUE, 3, NULL),

(4, 2, 1200, 'B02', 4, FALSE, NULL, 3),

(5, 3, 1800, 'C01', 5, TRUE, 1, NULL),

(6, 3, 1800, 'C02', 6, TRUE, 2, 4),

(7, 4, 2000, 'D01', 7, FALSE, 3, NULL),

(8, 4, 2000, 'D02', 8, TRUE, NULL, 5),

(9, 5, 1600, 'E01', 9, TRUE, 5, NULL),

(10, 5, 1600, 'E02', 10, FALSE, 7, NULL),

(11, 6, 1200, 'F01', 11, TRUE, 4, NULL),

(12, 6, 1200, 'F02', 12, FALSE, NULL, NULL),

(13, 7, 1800, 'G01', 13, TRUE, 8, NULL),

(14, 7, 1800, 'G02', 14, FALSE, NULL, NULL),

(15, 8, 2000, 'H01', 15, TRUE, 1, NULL),

(16, 8, 2000, 'H02', 16, TRUE, NULL, NULL),

(17, 9, 1600, 'I01', 17, FALSE, NULL, NULL),

(18, 9, 1600, 'I02', 18, TRUE, NULL, NULL),

(19, 10, 1400, 'J01', 19, TRUE, 7, NULL),

(20, 10, 1400, 'J02', 20, FALSE, NULL, NULL);



Рисунок 15 – Таблица билетов

**5. ЗАПРОСЫ К БД**

1. Необходимо проверить документы пассажира при заходе на рейс (С использованием внутреннего соединения по одному полю)

SELECT `пассажиры`.`ID\_Пассажира`, `пассажиры`.`Фамилия`, `пассажиры`.`Номер\_паспорта`, `билеты`.`ID\_Рейса`

From `пассажиры`

INNER JOIN `билеты` ON `пассажиры`.`ID\_Пассажира` = `билеты`.`ID\_Пассажира`;



Рисунок 16.

1. Необходимо узнать место для пассажира и его пункт назначения (С использованием косвенно связанных таблиц)

SELECT `пассажиры`.`ID\_Пассажира`, `пассажиры`.`Фамилия`, `билеты`.`Место`, `рейс`.`Пункт\_назначения`

FROM `билеты`

JOIN `пассажиры` ON `билеты`.`ID\_Пассажира` = `пассажиры`.`ID\_Пассажира`

JOIN `рейс` ON `билеты`.`ID\_Рейса` = `рейс`.`ID\_Рейса`;



Рисунок 17.

1. (С использованием таблиц, связанных более чем по одному полю)

WITH `Время` AS (

SELECT `билеты`.`ID\_Билета`, `билеты`.`ID\_Рейса`, `рейс`.`Время\_вылета`, `рейс`.`Время\_прилета`

FROM `билеты`

JOIN `рейс` ON `билеты`.`ID\_Рейса` = `рейс`.`ID\_Рейса`

)

SELECT DISTINCT `билеты`.`ID\_Билета`, `Время`.`Время\_вылета`, `Время`.`Время\_прилета`

FROM `Время`, `билеты`;

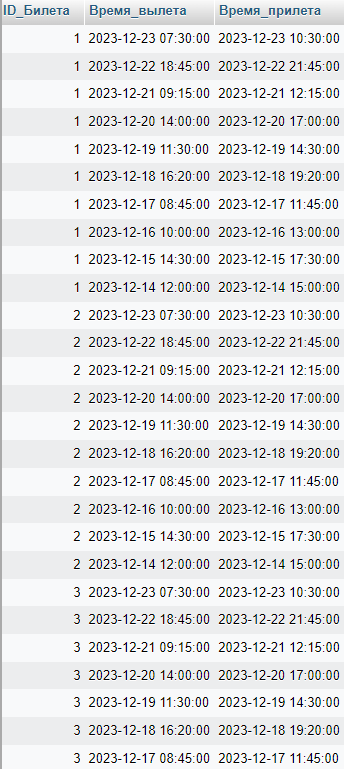


Рисунок 18.

1. (С использованием внешнего соединения таблиц)

SELECT \* FROM `билеты`

LEFT JOIN `рейс` ON `билеты`.`ID\_Рейса` = `рейс`.`ID\_Рейса`



Рисунок 19.

1. Необходимо получить краткую информацию о том, как найти нужный рейс (С использованием рекурсивного соединения)

SELECT `рейс`.`ID\_Рейса`, COUNT(`билеты`.`ID\_Билета`) AS 'Кол-во билетов на рейс', `gate`.`Название`, `полоса`.`Название`, `самолет`.`Модель`

FROM `рейс`

JOIN `билеты` ON `рейс`.`ID\_Рейса` = `билеты`.`ID\_Билета`

JOIN `gate` ON `рейс`.`ID\_Gate` = `gate`.`ID\_Gate`

JOIN `полоса` ON `рейс`.`ID\_Полосы` = `полоса`.`ID\_Полосы`

JOIN `самолет` ON `рейс`.`ID\_Самолета` = `самолет`.`ID\_Самолета`

GROUP BY `рейс`.`ID\_Рейса`;



Рисунок 20.

1. С использованием соединений по отношению

SELECT \*

FROM `рейс`

JOIN `билеты` ON `рейс`.`ID\_Рейса` = `билеты`.`ID\_Билета`

JOIN `gate` ON `рейс`.`ID\_Gate` = `gate`.`ID\_Gate`

JOIN `полоса` ON `рейс`.`ID\_Полосы` = `полоса`.`ID\_Полосы`

JOIN `самолет` ON `рейс`.`ID\_Самолета` = `самолет`.`ID\_Самолета`

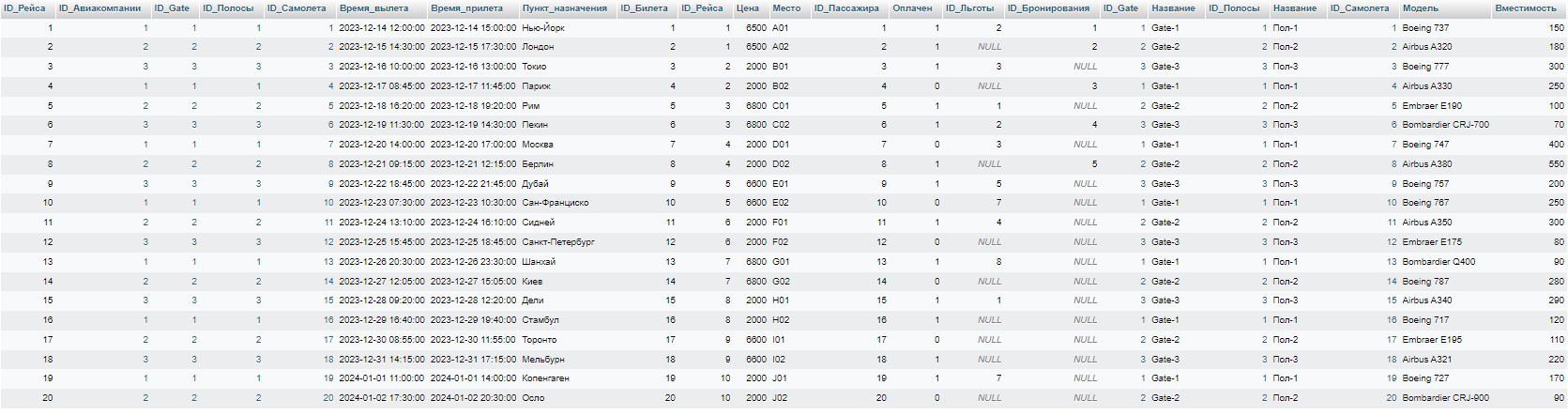


Рисунок 21.

1. Необходимо быстро узнать количество рейсов (С использованием функций агрегирования)

SELECT COUNT(`рейс`.`ID\_Рейса`) AS 'Количество рейсов' FROM `рейс` 

Рисунок 22.

1. Перекрестный запрос

SELECT \* FROM `билеты` CROSS JOIN `пассажиры`;

Рисунок 23.

1. Цены, дешевле 2000 рублей, резко подорожали на 5000. Необходимо занести изменения в базу данных (Запрос на изменение)

UPDATE `билеты` SET `Цена`= `Цена` + 5000

WHERE `Цена` < 2000

  
Рисунок 24.

1. Необходимо узнать, сколько заработали с продажи билетов (С вычисляемым полем)

SELECT SUM(`билеты`.`Цена`) AS 'Весь заработок с билетов' FROM `билеты`



Рисунок 25.

# **6. ЗАКЛЮЧЕНИЕ**

Цель курсовой работы – разработка базы данных справочной аэропорта. В ходе работы были рассмотрены основные требования и задачи, которые стоят перед базой данных справочной аэропорта, а также проведен анализ существующих систем и программ, которые могут быть использованы для реализации проекта.

Была разработана структура базы данных, которая включает в себя таблицы для хранения информации о рейсах, билетах, пассажирах, и других сущностях, необходимых для работы справочной аэропорта. Также были определены связи между таблицами и созданы запросы для получения нужной информации.

При написании курсовой работы были выполнены следующие задачи:

1. Анализ предметной области структуры работы справочной аэропорта;
2. Постановка задачи;
3. Составление технического задания;
4. Проектирование реляционной базы данных;
5. Логическое проектирование реляционной базы данных;
6. Физическое проектирование базы данных;
7. Реализация проекта базы данных.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

* + 1. Дейт, К. Введение в системы баз данных [Текст] / К. Дейт. – 8-е издание: Пер. с англ. – М.: Издательский дом «Вильямс», 2005. – 1328 с.: ил. – Парал. тит. англ.
    2. Зеленков, Ю.А. Введение в базы данных: [Электронный ресурс] / Ю.А. Зеленков. – Режим доступа: http://www.mstu.edu.ru/study/materials/zelenkov/toc.html, свободный. – Загл. с экрана.
    3. Кузнецов, С.Д. Введение в реляционные базы данных [Электронный ресурс] / С.Д. Кузнецов. – Сайт: Интуит. Национальный открытый университет – Режим доступа: https://intuit.ru/studies/courses/74/74/info, свободный. – Загл. С экрана.
    4. Сергеева, Т.И. Базы данных: модели данных, проектирование, язык SQL: учебное пособие [Текст] / Т.И. Сергеева, М.Ю. Сергеев. – Воронеж: ФГБОУ ВПО «Воронежский гос. техн. ун-т», 2012. – 233 с.
    5. Туманов, В.Е. Основы проектирования реляционных баз данных [Текст]: учебное пособие / В.Е. Туманов. – Москва: Интернет-Университет Информационных Технологий (ИНТУИТ): Бином. Лаборатория знаний, 2014. – 424 с.
    6. Бершадская Е. Г. Базы данных: теория, разработка и использование [Текст]: учеб. пособие/ Е. Г. Бершадская, Н. А. Филиппова. ФГБОУ ВПО ПГТА. - Пенза, ПГТА, 2012. -107 с., ил.
    7. Нормализация отношений. Шесть нормальных форм. «Хабр» – интернет источник: [Электронный ресурс] – Режим доступа: https://habr.com/ru/articles/254773/
    8. Введение в базы данных. «Хабр» – интернет источник: [Электронный ресурс] – Режим доступа: https://habr.com/ru/articles/686816/
    9. Национальный исследовательский университет «Высшая школа экономики» Учебные курсы. Теория баз данных [Электронный ресурс] – Режим доступа: <https://www.hse.ru/edu/courses/292667189>
    10. Анисимов, В.В. Лекции по дисциплине «Проектирование информационных систем» [Электронный ресурс] / В.В. Анисимов. – Режим доступа:  
        https://www.sites.google.com/site/anisimovkhv/learning/pris/lecture, свободный. – Загл. с экрана