

МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Казанский национальный исследовательский технологический университет»

(ФГБОУ ВО «КНИТУ»)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кафедра | | Интеллектуальных систем и управления информационными | | |
| ресурсами | | | | |
| Направление | | | | 02.03.03 «Математическое обеспечение и администрирование |
| информационных систем» | | | | |
| Профиль | | | Информационные системы и базы данных | |
| Группа | 4301-21 | | | |

**КУРСОВОЙ ПРОЕКТ**

по дисциплине: «Базы данных и системы управления базами данных»

на тему: «Проектирование базы данных аэропорта»

|  |  |  |  |
| --- | --- | --- | --- |
| Исполнитель |  |  | Алексеева Т.А |
|  | (дата, подпись) |  | (Ф.И.О.) |
| Руководитель |  |  |  |
|  | (дата, подпись) |  | (Ф.И.О.) |
| Нормоконтролер |  |  |  |
|  | (дата, подпись) |  | (Ф.И.О.) |

Работа защищена с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата, подпись)

КАЗАНЬ, *2022*

ЗАДАНИЕ

На курсовой проект студенту кафедры: Алексеевой Татьяне Александровне, 4301-21.

Тема проекта: Проектирование базы данных аэропорта.

Исходные данные к работе: Требуется разработать базу данных для хранения и обработки информации согласно предметной области, указанной в теме курсовой работы. Для реализации применяется реляционная СУБД MySQL. Проект должен включать не менее четырех таблиц, характерные запросы к базе данных, функции для автоматизации вычислений, процедуры для автоматизации рутинных действий пользователей базы данных, а также триггеры, позволяющие поддерживать целостность базы данных.

Содержание расчетно-пояснительной записки (включая перечень подлежащих разработке вопросов, включая вопросы стандартизации и контроля качества)

1. Изучение предметной области, постановка задачи.
2. Концептуальное, логическое и физическое моделирование базы данных.
3. Описание объектов базы данных: запросов, пользовательских функций, хранимых процедур и триггеров.

Перечень графического материала (схемной документации)

1. Модель «сущность-связь» базы данных в нотации П.Чена (1 рис.).

2. EER-модель базы данных в нотации IDEF1X (для объектов) и Crow’s Foot (для связей) (1 рис.).

Консультанты по проекту (с указанием относящихся к ним разделов)

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

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

Дата выдачи задания «19» сентября 2022 г.

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (фамилия, инициалы) 

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc123042865)

[ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc123042866)

[КОНЦЕПТУАЛЬНОЕ МОДЕЛИРОВАНИЕ БАЗЫ ДАННЫХ 6](#_Toc123042867)

[ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 14](#_Toc123042868)

[ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 18](#_Toc123042869)

[ОПИСАНИЕ ЗАПРОСОВ К БАЗЕ ДАННЫХ 23](#_Toc123042870)

[Функции 26](#_Toc123042871)

[Процедуры 28](#_Toc123042872)

[ЗАКЛЮЧЕНИЕ 34](#_Toc123042873)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 35](#_Toc123042874)

# **ВВЕДЕНИЕ**

В современном мире довольно сложно представить работу того или иного предприятия без использования информационных ресурсов или систем. Одной из основных информационных систем является база данных. База данных необходима для автоматизации, обработки, сбора и хранения тех или иных данных. [1]

База данных – это совокупность структур и таблиц, которые содержат в себе наборы определенных данных, которыми можно манипулировать или обрабатывать различными инструментами. Потребность в базе данных появилась, когда люди задумались о том, как образом организованно и структурированно хранить и оперировать большим объемом однотипных данных. [1]

Для того, чтобы пользоваться базой данных на компьютере, существуют специальные программы – системы управления базами данных(СУБД). На данный момент в современном мире существует около пятидесяти различных СУБД. Наиболее популярными являются следующие СУБД: MS SQL Server, Oracle Database, Informix, Sybase, IBM DB2, FoxPro, MS Access, MySQL, Postgresql, FireBird и др. [1].

Целью выполнения данной курсовой работы является формирование теоретических знаний и практических навыков в реализации, разработке и создании базы данных. [2]

В процессе выполнения данной курсовой работы будет:

* изучена предметная область
* произведено логическое, концептуальное и физическое моделирование базы данных согласно предметной области
* описаны объекты данной БД: запросов, пользовательских функций, хранимых процедур и триггеров.

# **ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ**

Существует огромное количество аэропортов по всему миру, которые между собой взаимодействуют. В сфере авиаперевозок задействовано очень много данных, пример, о пассажирах, авиакомпаниях, аэропортах и т.д. И структурированную организацию работы того или иного аэропорта можно представить в виде базы данных. [3]

База данных позволит автоматизировать и ускорить общий процесс авиаперевозок: продажу билетов, регистрацию рейсов, информирование пассажиров о том или ином рейсе и т.д.

Объекты, которые будут представлены для реализации, являются ключевыми для построения базы данных любого аэропорта. В авиаперевозках задействованы рейсы, которые организовывают авиакомпании, предоставляя самолеты. Данными услугами пользуется пассажир, которому нужно откуда-то прилететь или куда-то прилететь. Билеты пассажирам продают сотрудники аэропорта. Точками отправления и прибытия являются аэропорты. [4]

Например, один сотрудник может продать множество билетов на различные рейсы, или одна авиакомпания может осуществлять, организовывать рейсы на определенном самолете.

# **КОНЦЕПТУАЛЬНОЕ МОДЕЛИРОВАНИЕ БАЗЫ ДАННЫХ**

Проектируемая БД аэропорта, позволит отслеживать работу аэропорта, количество пассажиров, купивших билет на тот или иной рейс, а также в целом структурно организовывать авиаперевозки.

В процессе исследования предметной области были определены ключевые сущности для целостной реализации базы данных аэропорта. На данном этапе будут описаны сущности, их атрибуты, связи между сущностями, а также будет построена модель «сущность-связь» в нотации П.Чена.[4]

В процессе анализа предметной области были выделены следующие основные сущности:

1. Авиакомпания(airline)
2. Самолет(airplane)
3. Аэропорты(airport)
4. Сотрудники(employees)
5. Рейс(flight)
6. Пассажир(passenger)
7. Паспорта(passports)
8. Билеты(tickets)

Сущность *Авиакомпания*(airline) содержит данные о так называемых перевозчиках: ID авиакомпании, её наименование, описание, адрес и контактные данные. *Авиакомпания* связана с сущностью *Рейс.* Одна авиакомпания может осуществлять много рейсов. Более подробная характеристика сущности *Авиакомпания*(airline) представлена в *таблице 1*.

*Таблица 2. Сущность Авиакомпания(airline)*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Ограничение | | Описание | Тип  Данных | Ограничение  На тип данных | Домен | Значение по умолчанию |
| ID | Первичный ключ | Идентификатор | | Натуральное число | Not Null  Auto\_Increment  Primary key | Натуральные числа |  |
| Name |  | Наименование компании | | Строка переменной длины | Not Null | Только буквы и символы |  |
| Description |  | Описание авиакомпании | | Строка переменной длины | Not Null | Только буквы и символы | NULL |
| Address |  | Адрес | | Строка переменной длины | Not Null | Только буквы и символы |  |
| Contacts |  | Телефон | | Числа с плавающей точкой | Not Null | Числа |  |

Сущность *Рейс*(flight) содержит данные о рейсах: ID рейса, номер рейса, стоимость билета на рейс, дата начала рейса, дата окончания рейса, ID аэропорта вылета, ID аэропорта прилета, ID авиакомпании, ID самолета. *Рейс*(flight) связан с *билетами*(tickets), *аэропортами*(airport) и *самолетом*(airplane). На один рейс может быть продано много билетов, и один самолет совершает много рейсов. Более подробное описание атрибутов сущности представлено в *таблице 2.*

*Таблица 2. Сущность Рейс(flight)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Название | Ограничение | Описание | Тип  Данных | Ограничение  На тип данных | Домен | Значение по умолчанию |
| ID | Первичный ключ | Идентификатор | Целое число | Not Null  Auto\_Increment  Primary key | Натуральные числа |  |
| Number\_of\_flight |  | Номер рейса | Строка переменной длины | Not Null | Только буквы и символы |  |
| Number\_of\_tickets\_sold |  | Количество проданных билетов | Целое число | Not Null | Только буквы и символы | NULL |
| Date\_of\_beggining |  | Начало | Дата и время | Not Null | Только буквы и символы |  |
| End\_date |  | Конец | Дата и время | Not Null | Только буквы и символы |  |
| Airport\_number1ID | Внешний ключ для airport | ID аэропорта депортации | Целое число | Not Null | Только буквы и символы |  |
| Airport\_number2ID | Внешний ключ для airport | ID аэропорта прилета | Целое число | Not Null | Только буквы и символы |  |
| Airplane\_ID | Внешний ключ для airplane | ID самолета | Целое число | Not Null | Только буквы и символы |  |
| Airline\_ID | Внешний ключ для airline | ID авиакомпании | Целое число | Not Null | Только буквы и символы |  |

Сущность *Билеты*(tickets) содержит данные о билетах: ID билета, номер билета, время покупки, наличие багажа, цена билета на рейс, дата отбытия, дата прибытия, ID пассажира, ID сотрудника, ID рейса. *Билеты*(tickets) связаны с *пассажиром*(passenger), *рейсом*(flight) и *сотрудниками*(employees). На один рейс может быть продано много билетов, один пассажир может купить много билетов, и один сотрудник может продать много билетов. Более подробное описание атрибутов сущности представлено в *таблице 3.* [5]

*Таблица 3. Сущность Билеты(tickets)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Название | Ограничение | Описание | Тип  Данных | Ограничение  На тип данных | Домен | Значение по умолчанию |
| ID | Первичный ключ | Идентификатор | Целое число | Not Null  Auto\_Increment  Primary key | Натуральные числа |  |
| Number\_of\_ticket |  | Номер билета | Целое число | Not Null | Натуральные числа |  |
| Purchase\_time |  | Время покупки билета | Дата и время | Not Null | Натуральные числа | NULL |
| Baggage |  | Наличие багажа | 1 или о |  | 1 или 0 | 0 |
| Price |  | Цена билета | Целое число | Not Null | Натуральные числа | '1800.00' |
| Departure\_date |  | Дата депортации | Дата и время | Not Null | Натуральные числа |  |
| Arrival\_date |  | Дата прибытия | Дата | Not Null | Натуральные числа |  |
| Passenger\_ID | Внешний ключ для passenger | ID пассажира | Целое число | Not Null | Натуральные числа |  |
| Employees\_ID | Внешний ключ для employees | ID сотрудника | Целое число | Not Null | Натуральные числа |  |
| Flight\_ID | Внешний ключ для flight | ID рейса | Целое число | Not Null | Натуральные числа |  |

Сущность *Самолет*(airplane) содержит данные о самолетах: ID самолета, бортовой номер, модель, вместимость, дата начала ввода в эксплуатацию. *Самолет*(airplane) связан с *Рейсом*(flight). Один самолет может совершать много рейсов. Более детальное описание атрибутов данной сущности приведено в *таблице 4*.

*Таблица 4. Сущность Самолет(airplane)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Название | Ограничение | Описание | Тип  Данных | Ограничение  На тип данных | Домен | Значение по умолчанию |
| ID | Первичный ключ | Идентификатор | Целое число | Not Null  Auto\_Increment  Primary key | Натуральные числа |  |
| Bort\_number |  | Бортовой номер | Строка переменной длины | Not Null | Только буквы и символы |  |
| Model |  | Модель самолета | Строка переменной длины | Not Null | Только буквы и символы |  |
| Capacity |  | Вместимость | Целое число | Not Null | Только буквы и символы |  |
| Entry\_date |  | Дата ввода в эксплуатацию | Дата | Not Null | Натуральные Числа |  |

Сущность *Пассажир*(passenger) содержит данные о пассажирах: ID пассажира, имя пассажира, отчество пассажира, фамилия, адрес, контактный телефон, электронная почта, дата рождения. *Пассажир*(passenger) связан с *паспортами*(passports) и *билетами*(tickets). Один пассажир может иметь только один паспорт и один пассажир может купить много билетов. Более подробно атрибуты сущности описаны в *таблице 5*.

*Таблица 5. Сущность Пассажир(passenger)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Название | Ограничение | Описание | Тип  Данных | Ограничение  На тип данных | Домен | Значение по умолчанию |
| ID | Первичный ключ | Идентификатор | Целое число | Not Null  Auto\_Increment  Primary key | Натуральные числа |  |
| Name |  | Имя | Строка переменной длины | Not Null | Только буквы и символы |  |
| Second\_name |  | Отчество | Строка переменной длины | Not Null | Только буквы и символы |  |
| Surname |  | Фамилия | Строка переменной длины | Not Null | Только буквы и символы |  |
| Address |  | Адрес | Строка переменной длины | Not Null | Только буквы и символы |  |
| Phone\_number |  | Контактный телефон | Целое число | Not Null | Натуральные числа |  |
| e\_mail |  | Электронная почта | Строка переменной длины | Not Null | Только буквы и символы |  |
| date\_birth |  | Дата рождения | Целое число | Not Null | Натуральные числа |  |

Сущность *Сотрудники*(employees) содержит данные о сотрудниках: ID сотрудника, имя сотрудника, отчество сотрудника, фамилия, должность, контактный телефон, адрес, электронная почта. *Сотрудники*(employees) связаны с *билетами*(tickets). Один сотрудник может продать много билетов. Более подробное описание атрибутов сущности находится в *таблице 6.*

*Таблица 6. Сущность Сотрудники(employees)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Название | Ограничение | Описание | Тип  Данных | Ограничение  На тип данных | Домен | Значение по умолчанию |
| ID | Первичный ключ | Идентификатор | Целое число | Not Null  Auto\_Increment  Primary key | Натуральные числа |  |
| Name |  | Имя | Строка переменной длины | Not Null | Только буквы и символы |  |
| Second\_name |  | Отчество | Строка переменной длины | Not Null | Только буквы и символы |  |
| Surname |  | Фамилия | Строка переменной длины | Not Null | Только буквы и символы |  |
| Job\_tittle |  | Должность | Строка переменной длины | Not Null | Только буквы и символы |  |
| Contacts |  | Контактный телефон | Целое число | Not Null | Натуральные числа |  |
| Address |  | Адрес | Строка переменной длины | Not Null | Только буквы и символы |  |
| e\_mail |  | Электронная почта | Строка переменной длины | Not Null | Только буквы и символы |  |

Сущность *Паспорта*(passports) содержит данные о паспортах: ID паспорта, имя пассажира (по загранпаспорту), отчество пассажира(по загранпаспорту), фамилия(по загранпаспорту), имя, отчество, фамилия, серия паспорта, номер паспорта. *Паспорта*(passports) связаны с *пассажиром*(passenger). Один пассажир может иметь только один паспорт. Более подробное описание сущности представлено в *таблице 7.*

*Таблица 7. Сущность Паспорта(passports)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Название | Ограничение | Описание | Тип  Данных | Ограничение  На тип данных | Домен | Значение по умолчанию |
| ID | Первичный ключ | Идентификатор | Целое число | Not Null  Auto\_Increment  Primary key | Натуральные числа |  |
| Name |  | Имя(загранпаспорт) | Строка переменной длины | Not Null | Только буквы и символы |  |
| Second\_name |  | Отчество(загранпаспорт) | Строка переменной длины | Not Null | Только буквы и символы |  |
| Surname |  | Фамилия(загранпаспорт) | Строка переменной длины | Not Null | Только буквы и символы |  |
| Имя |  | Имя | Строка переменной длины | Not Null | Только буквы и символы |  |
| Фамилия |  | Фамилия | Строка переменной длины | Not Null | Только буквы и символы |  |
| Отчество |  | Отчество | Строка переменной длины | Not Null | Только буквы и символы |  |
| Passport\_series |  | Серия паспорта | Строка переменной длины | Not Null | Только буквы и символы |  |
| Number\_of\_passports |  | Номер паспорта | Строка переменной длины | Not Null | Только буквы и символы |  |

Сущность *Аэропорты*(airport) содержит данные об аэропортах: ID аэропорта, его полное наименование и короткое наименование, адрес. *Аэропорты*(airport) связаны с *рейсом*(flight). В один аэропорт могут прилетать самолеты по различным рейсам и улетать – аналогично. Более подробное описание сущности приведено в *таблице 8.*

*Таблица 8. Сущность Аэропорты(airport)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Название | Ограничение | Описание | Тип  Данных | Ограничение  На тип данных | Домен | Значение по умолчанию |
| ID | Первичный ключ | Идентификатор | Целое число | Not Null  Auto\_Increment  Primary key | Натуральные числа |  |
| Full\_name |  | Полное наименование аэропорта | Строка переменной длины | Not Null | Только буквы и символы |  |
| Short\_name |  | Короткое наименование | Строка переменной длины | Not Null | Только буквы и символы |  |
| Address |  | Адрес | Строка переменной длины | Not Null | Только буквы и символы |  |

Вследствие анализа предметной области нами была построена модель «сущность – связь» в нотации П.Чена, которая представлена на *рисунке 1*[10]

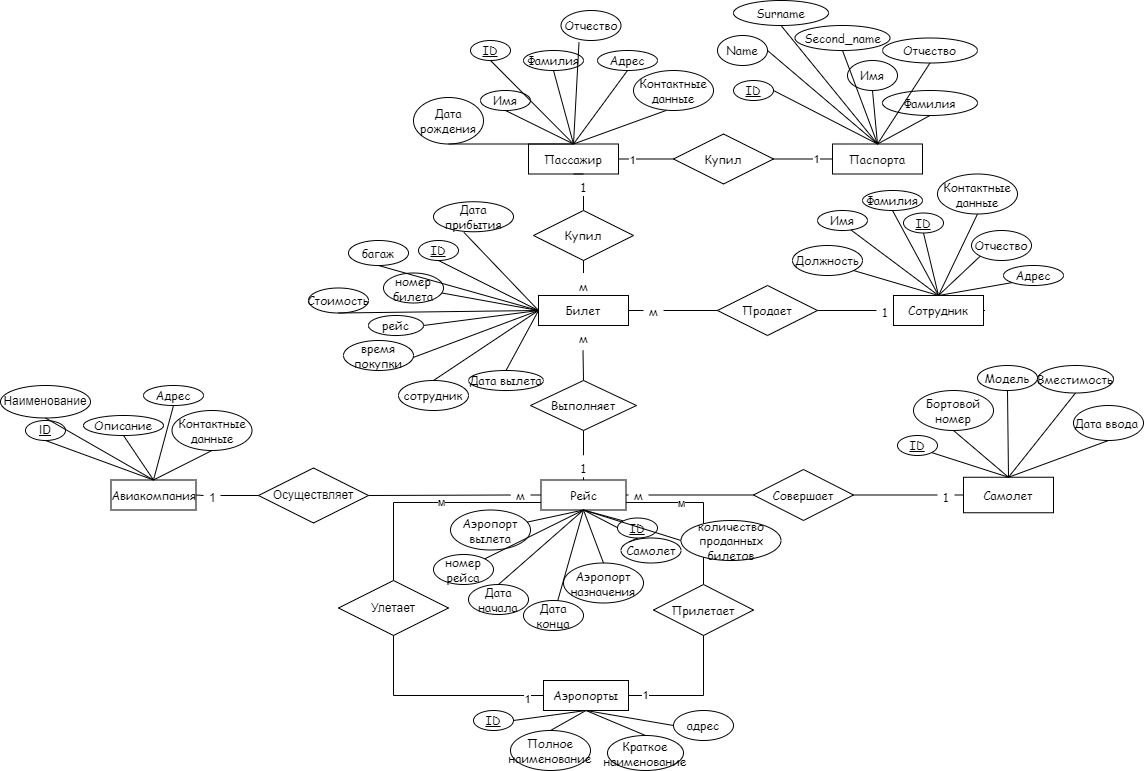


Рисунок 1 – Модель БД в нотации П.Чена

# **ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ**

Таблицы реляционной БД должны быть нормализованы. Таблица соответствует третьей нормальной форме (3НФ), если для нее определен первичный ключ, все значения полей неделимы, все неключевые атрибуты полностью функционально зависят от первичного ключа, но взаимонезависимые атрибуты между собой.[8]

На *рисунке 2* продемонстрированы функциональные зависимости сущности *Авиакомпании*(airline). Так как не ключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица *Авиакомпании*(airline) находится в 3НФ. [7]

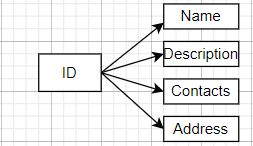


Рисунок 2 – Функциональные зависимости сущности Авиакомпания

На *рисунке 3* представлены функциональные зависимости сущности *Рейс*(flight). Так как не ключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица *Рейс*(flight) находится в 3НФ. [7]

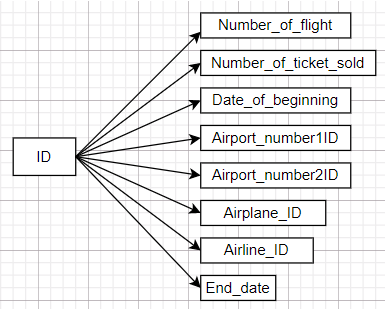


Рисунок 3 – Функциональные зависимости сущности Рейс

На *рисунке 4* представлены функциональные зависимости сущности *Билеты*(tickets). Так как не ключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица *Билеты*(tickets) находится в 3НФ. [7]

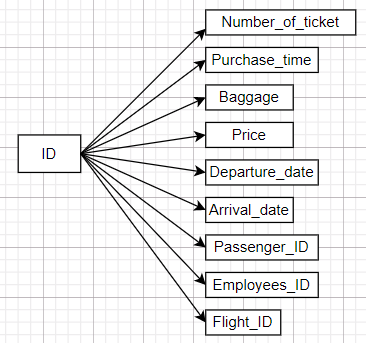


Рисунок 4 – Функциональные зависимости сущности Билеты

На *рисунке 5* представлены функциональные зависимости сущности *Самолет*(airplane). Так как не ключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица *Самолет*(airplane) находится в 3НФ. [7]

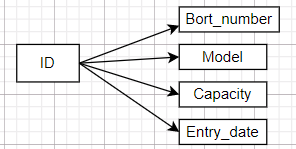


Рисунок 5 – Функциональные зависимости сущности Самолет

На *рисунке 6* представлены функциональные зависимости сущности *Пассажир*(passenger). Так как не ключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица *Пассажир*(passenger) находится в 3НФ. [7]

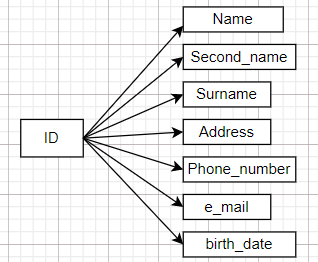


Рисунок 6 – функциональные зависимости сущности Пассажир

На *рисунке 7* представлены функциональные зависимости сущности *Сотрудники*(employees). Так как не ключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица *Сотрудники*(employees) находится в 3НФ.[7]

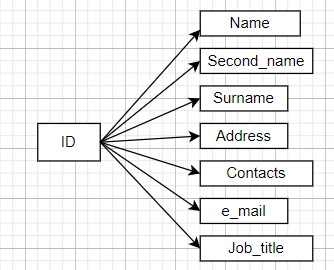


Рисунок 7 – Функциональные зависимости сущности Сотрудники

На *рисунке 8* представлены функциональные зависимости сущности *Паспорта*(passports). Так как не ключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица *Паспорта*(passports) находится в 3НФ.[7]

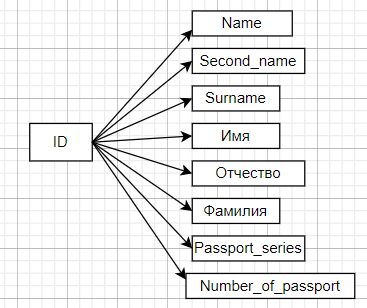


Рисунок 8 – Функциональные зависимости сущности Паспорта

На *рисунке 9* представлены функциональные зависимости сущности *Аэропорты*(airport). Так как не ключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица *Аэропорты*(airport) находится в 3НФ.[7]

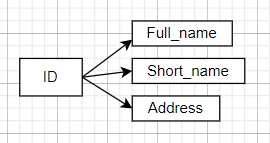


Рисунок 9 – Функциональные зависимости сущности Аэропорты

Вследствие анализа функциональных зависимостей атрибутов всех сущностей можно сделать вывод о том, что сущности не нуждаются в приведении к 3НФ, изменения таблиц не осуществлялись, поэтому корректировка модели «сущность – связь» в нотации П.Чена не требуется. [8]

# **ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ**

После того, как был проведен анализ предметной области, рассмотрены все этапы проектирования базы данных и анализ ее функциональных зависимостей, необходимо выбрать, какой системой управления базами данных (СУБД) будет рациональнее и лучше всего воспользоваться. В данной работе будет использована СУБД MySQL. Ниже приведены преимущества данной системы управления базами данных, которые и послужили причиной для выбора именно её.[9]

Преимущества MySQL:

* Данная СУБД довольно проста в использовании. Множество плагинов и вспомогательных приложений упрощают работу с базами данных.
* MySQL обладает множественным функционалом, который может пригодиться в реализации любого проекта.
* Масштабируемость. MySQL может использоваться для работы как с большими объемами данных, так и с маленькими, что явно говорит о ее универсальности.
* СУБД обладает высокой скоростью, которая реализуется за счет упрощения некоторых стандартов.

Ниже в таблицах 9 – 17 будут приведены примеры данных раннее описанных таблиц (табл. 1 - 8).

В таблице 9 приведен пример данных для сущности *Авиакомпания*(airline).

*Таблица 9. Пример данных для сущности Авиакомпания(airline).*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *ID* | *Name* | *Description* | *Address* | *Contacts* |
| 1 | Победа | Российская бюджетная авиакомпания | Москва, п.Моск-ий, Киевское шоссе, 22-й к-р | 88095054777 |
| 2 | Аэрофлот | Российская авиакомпания | Москва, ул. Арбат, д. 1. | 88004445555 |
| 3 | S7 Airlines | Внутренние и международные авиаперевозки | Россия, Московская область, г. Домодедово | 88007000707 |

В таблице 10 приведен пример данных для сущности *Рейс*(flight).

*Таблица 10. Пример данных для сущности Рейс(flight).*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *ID* | *Number\_of\_flight* | *Number\_of\_ticket\_sold* | *Date\_of\_beginning* | *End\_date* | *Airoport\_number1ID* | *Airoport\_number2ID* | *Airplane\_ID* | *Airline\_ID* |
| 1 | SU 46 | 2 | 2022-12-22 00:00:00 | 2022-12-23 00:00:00 | 1 | 2 | 1 | 2 |
| 2 | 7W 445 | 2 | 2023-01-03 00:00:00 | 2023-01-04 00:00:00 | 2 | 3 | 1 | 2 |

В таблице 11 приведен пример данных для сущности *Билеты*(tickets).

*Таблица 11. Пример данных для сущности Билеты(tickets).*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *ID* | *Number\_of\_ticket* | *Purchase\_time* | *Baggage* | *Price* | *Arrivale\_date* | *Departure\_date* | *Passenger\_ID* | *Employees\_ID* | *Flight\_ID* |
| 1 | 123098 | 2022-10-22 01:30:00.000000 | 1 | 10000.00 | 2022-12-23 | 2022-12-22 00:00:00 | 1 | 2 | 1 |
| 2 | 555555 | 2022-12-12 16:00:00.000000 | 0 | 13000.00 | 2023-01-14 | 2023-01-12 00:00:00 | 2 | 3 | 2 |

В таблице 12 приведен пример данных для сущности *Самолет*(airplane).

*Таблица 12. Пример данных для сущности Самолет(airplane).*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *ID* | *Bort\_number* | *Model* | *Capacity* | *Entry\_date* |
| 1 | VQ-BAU | Airbus A319neo | 158 | 2015-01-01 |
| 2 | RA-67685 | ATR EVO | 140 | 1999-03-01 |
| 3 | RF-36782 | Sukhoi Superjet 100 | 108 | 2010-12-23 |

В таблице 13 приведен пример данных для сущности *Пассажир*(passenger).

*Таблица 13. пример данных для сущности Пассажир(passenger).*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *ID* | *Name* | *Second\_name* | *Surname* | *Address* | *Phone\_number* | *e\_mail* | *Date\_birth* |
| 1 | Татьяна | Александровна | Алексеева | Россия, р-ка Татарстан, г.Зеленодольск ул. Комарова, д.23, кв.47 | 89534044009 | tanya\_alekseeva\_02@bk.ru | 2002-01-02 |
| 2 | Валерия | Дмитриевна | Минеева | Россия, р-ка Татарстан, г. Казань, ул. Маршала Чуйкова, д.29В, кв.73 | 89456782341 | l\_mineeva\_23@mail.ru | 1998-12-21 |

В таблице 14 приведен пример данных для сущности *Сотрудники*(employees).

*Таблица 14. Пример данных для сущности Сотрудники(employees).*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *ID* | *Name* | *Second\_name* | *Surname* | *Job\_title* | *Contacts* | *Address* | *e\_mail* |
| 1 | Лариса | Олеговна | Худякова | Кассир | 898765436542 | г.Москва, ул. Варшавского, д.23, кв.4 | larisa\_anatolyevna\_54@gmail.com |
| 2 | Ольга | Вячеславовна | Петрова | Кассир | 893965743423 | г.Москва, ул.Толстого, д.4, кв.3 | petrova\_1999@gmail.com |

В таблице 15 приведен пример данных для сущности *Паспорта*(passports).

*Таблица 15. Пример данных для сущности Паспорта(passports)*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *ID* | *Name* | *Second\_name* | *Surname* | *Имя* | *Отчество* | *Фамилия* | *Passport\_series* | *Number\_of\_passport* |
| 1 | Alekcandr | Sergeevich | |  | | --- | |  |   Tolmazanov | Александр | Сергеевич | Толмазанов | 9232 | 209976 |
| 2 | Tatyana | Alexandrovna | Usanova | Татьяна | Александровна | Усанова | 9234 | 210133 |

В таблице 16 приведен пример данных для сущности *Аэропорты*(Airport).

*Таблица 16. Пример данных для сущности Аэропорты(Airport).*

|  |  |  |  |
| --- | --- | --- | --- |
| *ID* | *Full\_name* | *Short\_name* | *Address* |
| 1 | Внуково | VKO | Россия, г.Москва, ул.1-я Рейсовая, 12 |
| 2 | Кольцово | SVX | Россия, г.Екатеринбург, пл. Бахчиванджи, 1 |
| 3 | Казань | KZN | Россия, г.Казань, лаишевский район |

База данных была спроектирована в MySQL Models в процессе прямого инжиниринга (forward engineering). MySQL позволяет облегчить создание новых физических моделей данных и модификацию существующих баз данных MySQL с обратным / прямым инжинирингом. Задача MySQL Workbench состоит в том, чтобы наглядно представить пользователю модель базы данных в графическом виде и упростить пользование ею. Для визуального представления были выбраны нотации для объектов – IDEF1X и для связей – Crow’s Foot. На рисунке 10 представлена модель базы данных в виде EER-диаграммы.

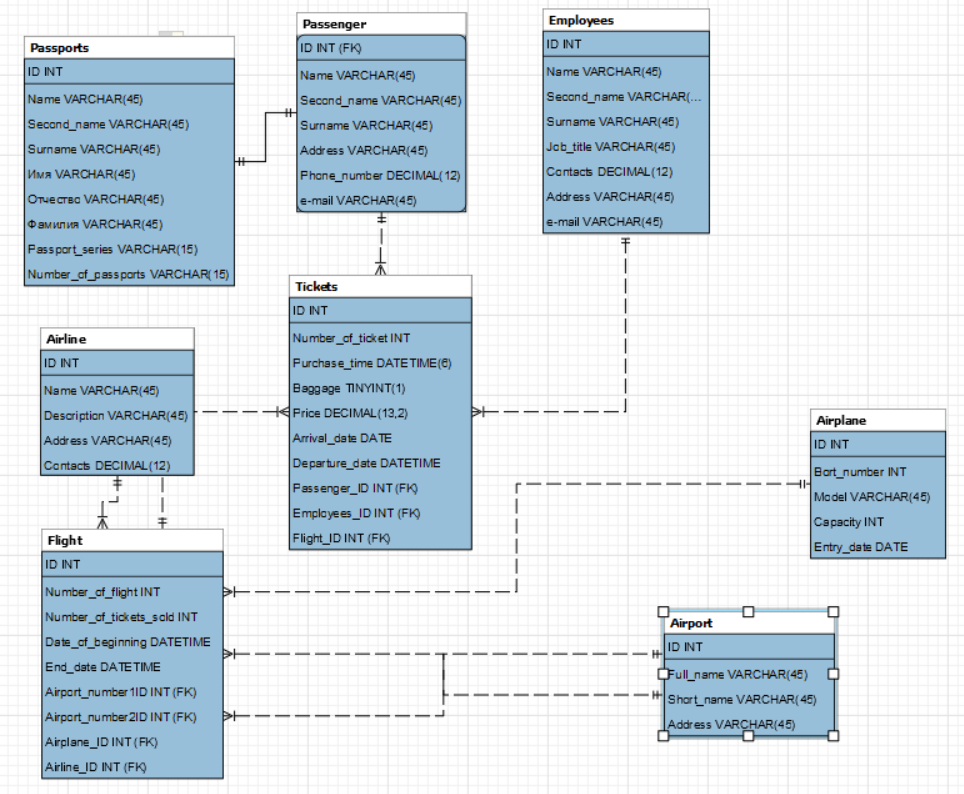


Рисунок 10 – Модель базы данных в виде EЕR – диаграммы

# **ОПИСАНИЕ ЗАПРОСОВ К БАЗЕ ДАННЫХ**

1) Представление – общая информация о билете пассажира

Представление выводит информацию из сущностей: flight, tickets, airport.

Код запроса:

CREATE OR REPLACE VIEW DATA\_PASSENGER (Name\_passenger,From\_, To\_,Carrier,Number\_flight, Date\_, Date\_departure) as

select concat(passenger.Name,' ',passenger.Second\_name,' ',passenger.Surname) as "ФИО пассажира",

a1.Short\_name as "FROM / ОТ",

a2.Short\_name as "TO / КУДА",

airline.Name as "ПЕРЕВОЗЧИК",

flight.Number\_of\_flight as "НОМЕР РЕЙСА",

tickets.Arrival\_date as "ДАТА",

tickets.Departure\_date as "ДАТА вылета"

from passenger

join tickets on tickets.Passenger\_ID=passenger.ID

join tickets ticket on tickets.Passenger\_ID = passenger.ID

join flight on flight.ID = tickets.Flight\_ID

join airport a1 on flight.Airport\_number1ID = a1.ID

join airport a2 on flight.Airport\_number2ID = a2.ID

join airline on `airline`.ID = flight.Airline\_ID;

Пример представления представлен на *рисунке* *10*

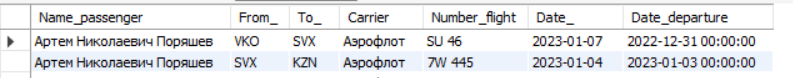


Рисунок 10 – пример представления, которое выводит информацию о билете

2) Запрос, выводящих информацию о пассажирах, имеющих багаж

Запрос включает в себя сущности: tickets, passenger. Запрос выводит ID пассажира, его ФИО и контактные данные(телефон и электронную почту).

Код запроса:

select ID,concat(Name,' ',Second\_name,' ',Surname) as "ФИО пассажира",

concat(Phone\_number,' ',e\_mail) as "Контактный номер телефона и e-mail пассажира" from passenger

where ID in (select Passenger\_ID

from tickets

where Baggage =1);

Пример выполнения данного запроса представлен на *рисунке* *11*

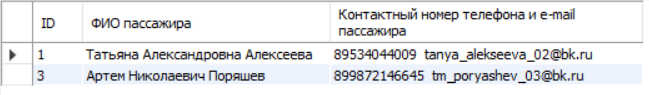


Рисунок 11- вывод данных о пассажирах, имеющих багаж

3) Запрос для вывода данных о пассажирах, летящих по рейсу с ID=3

Запрос включает в себя сущности: tickets, passenger. Запрос выводит ID пассажира, его ФИО и контактные данные (телефон и электронную почту) и адрес, а также сортируется по ФИО пассажира по возрастанию.

Код запроса:

select ID,concat(Name,' ',Second\_name,' ',Surname) as "ФИО пассажира",

concat(Phone\_number,' ',e\_mail) as "Контактный номер телефона и e-mail пассажира",

Address as "Адрес" from passenger

where ID in

( select Passenger\_ID

from tickets

where Flight\_ID=3

)

order by `ФИО пассажира` asc;

Пример выполнения запроса представлен на *рисунке 12*

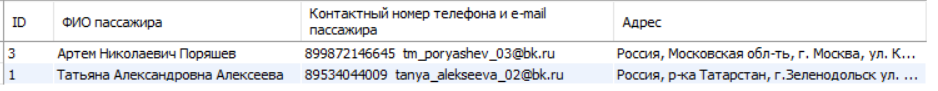


Рисунок 12 – пример выполнения запроса, который выводит данные о пассажирах на третьем рейсе

4) Запрос, выводящий данные пассажиров, которых обслуживал сотрудник с ID = 2.

Запрос включает в себя сущности: tickets, passenger. Запрос выводит ID пассажира, его ФИО и контактные данные (телефон и электронную почту) и адрес, а также сортируется по ФИО пассажира по возрастанию.

Код запроса:

select ID,concat(Name,' ',Second\_name,' ',Surname) as "ФИО пассажира",

Address as "Адрес",

Phone\_number as "Контактный номер телефона",

e\_mail as "Электронная почта" from passenger

where ID in

( select Passenger\_ID

from tickets

where Employees\_ID=2

)

order by "ФИО пассажира" asc;

Пример выполнения запроса представлен на *рисунке 13*

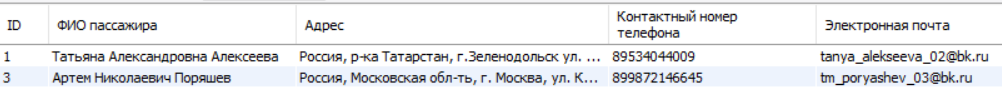


Рисунок 13 – пример выполнения запроса, который выводит данные о пассажирах на третьем рейсе

5) Запрос для вывода ФИО, контактных данных, номера рейса, номера билета, аэропорта улета и прилета пассажиров, которые летят по рейсам с ID = 2, ID = 3

Запрос включает в себя сущности: tickets, passenger, airport, flight. Запрос выводит ФИО пассажира, номер рейса и билета, полное и короткое наименование аэропортов.

Код запроса:

select concat(Name,' ',Second\_name,' ',Surname) as "ФИО пассажира",

flight.Number\_of\_flight as "Номер рейса",

tickets.Number\_of\_ticket as "Номер билета",

a1.Full\_name as "Полное наименование аэропорта улёта",

a2.Full\_name as "Полное наименование аэропорта прилета",

a1.Short\_name as "Короткое наименование аэропорта улета",

a2.Short\_name as "Короткое наименование аэропорта прилёта"

from passenger

join tickets on passenger.ID=tickets.Passenger\_ID

join flight on tickets.Passenger\_ID=flight.ID

join airport a1 on flight.Airport\_number1ID = a1.ID

join airport a2 on flight.Airport\_number2ID = a2.ID

where flight.ID=2 or flight.ID=3;

Пример выполнения запроса представлен на *рисунке 14*

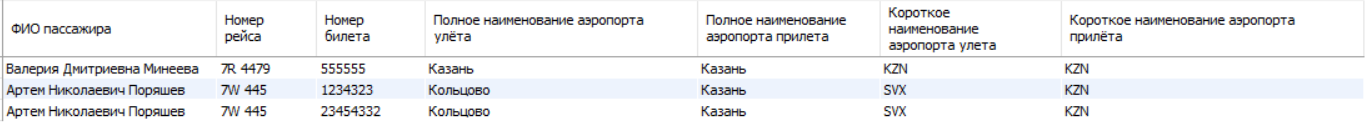


Рисунок 14 – пример выполнения запроса, выводящего данные о пассажирах на третьем и втором рейсе

6) Запрос, выводящий количество перелётов каждого из пассажиров

Запрос включает в себя сущности: tickets, passenger. Запрос выводит ФИО пассажира и его количество перелетов, а также сортируется по ФИО пассажира по убыванию.

Код запроса:

select concat(Name,' ',Second\_name,' ',Surname) as "ФИО пассажира", count(\*) as "Количество перелётов" from passenger

join tickets on tickets.Passenger\_ID=passenger.ID

group by passenger.ID

having count(\*)>=1

order by `Количество перелётов` desc;

Пример выполнения запроса представлен на *рисунке 15*

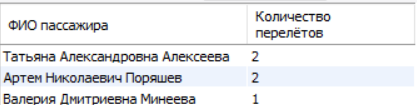


Рисунок 15 – пример выполнения запроса, выводящего количество перелетов

## **Функции**

1) Функция для вычисления свободных мест на рейс по ID

*Синтаксис функции:*

free\_pl\_flight (idfl int)

*Аргументы функции:*

idfl – ID рейса, тип данных – int

*Результат:*

Свободные места на рейс, тип данных – int

*Алгоритм функции:*

Происходит объявление переменной kuplen и mesto. Далее присваивается переменной kuplen значения из столбца Number\_of\_tickets\_sold таблицы flight, а переменной mesto значение разницы между значением столбца Capacity таблицы airplane и переменной kuplen. Функция возвращает значение переменной mesto.

Код функции:

CREATE DEFINER=`root`@`localhost` FUNCTION `free\_pl\_flight`(idfl int) RETURNS int

DETERMINISTIC

BEGIN

declare mesto int;

declare kuplen int;

set kuplen = (select Number\_of\_tickets\_sold from flight join airplane on flight.Airline\_ID = airplane.ID where flight.ID = idfl);

set mesto = (select Capacity from airplane join flight on flight.Airplane\_ID = airplane.ID where flight.ID = idfl) - kuplen;

RETURN mesto;

END

Пример результатов выполнения представлен на *рисунке 16*

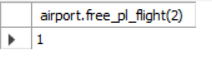


Рисунок 16 - пример выполнения функции для вычисления свободных мест

2) Функция для вычисления возраста пассажира по его ID

*Синтаксис функции:*

vozrast\_passenger (idpas int)

*Аргументы функции:*

idpas – ID пассажира, тип данных – int

*Результат:*

Возраст пассажира, тип данных – int

*Алгоритм функции:*

Происходит объявление переменной vozrast. Далее ей присваивается значение, рассчитываемое по дате рождения пассажира из столбца date\_birth таблицы passenger. Функция возвращает значение данной переменной

Код функции:

CREATE DEFINER=`root`@`localhost` FUNCTION `vozrast\_passenger`(idpas int) RETURNS int

DETERMINISTIC

BEGIN

declare vozrast int;

set vozrast = (select (DATE\_FORMAT(FROM\_DAYS(TO\_DAYS(now()) - TO\_DAYS(date\_birth)), '%Y') + 0) from passenger where passenger.ID = idpas);

if vozrast is null then set vozrast = 0; end if;

return vozrast;

END

Пример результатов выполнения представлен на *рисунке 17*



Рисунок 17 - пример выполнения функции для вычисления возраста

## **Процедуры**

1) Процедура вычисления стоимости билета на рейс при наличии багажа и без него для пассажира по билету с ID

*Синтаксис процедуры:*

cost\_ticket( idbilet int)

*Входные данные процедуры:*

idbilet – ID билета, тип данных – int

*Алгоритм процедуры:*

Происходит объявление переменных nalB, costticket, costticket. Если значение столбца Baggage в таблице tickets при равенстве ID билета и выходных данных будет равно 1, то переменной costticket присваивается значение суммы столбца price из таблицы tickets и переменной constB, иначе данной данной переменной присваивается просто значение столбца price из таблицы tickets. Далее происходит выборка столбцов таблиц для вывода.

Код процедуры:

CREATE DEFINER=`root`@`localhost` PROCEDURE `cost\_ticket`( idbilet int)

DETERMINISTIC

BEGIN

declare nalB varchar(20);

declare costticket int;

declare costticket int;

set constB=3000;

if (select Baggage from tickets where tickets.ID=idbilet)=1

then set nalB = "Багаж есть";

set costticket = (select price from tickets where tickets.ID=idbilet) + constB;

else set nalB = "Багажа нет";

set costticket = (select price from tickets where tickets.ID=idbilet);

END if;

select concat(passenger.Name,' ',passenger.Second\_name,' ',passenger.Surname) as "ФИО пассижира",

flight.Number\_of\_flight as "Номер рейса",

tickets.Number\_of\_ticket as "Номер билета",

nalB as "Наличие багажа",

costticket as "Стоимость билета на рейс"

from passenger

join tickets on tickets.Passenger\_ID = passenger.ID

join flight on flight.ID = tickets.Flight\_ID

where tickets.ID=idbilet;

END

Пример выполнения процедуры показан на *рисунке 18*

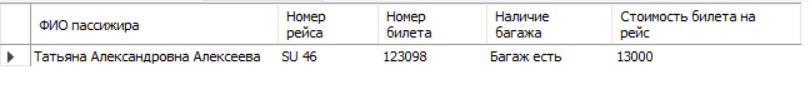


Рисунок 18 – пример выполнения процедуры для вычисления стоимости билета на рейс с багажом или без для пассажира по билету с ID

2) Процедура для вычисления стоимости на рейс с багажом или без по ID рейса

*Синтаксис процедуры:*

data\_about\_cost (reis int)

*Входные данные процедуры:*

reis – ID рейса, тип данных – int

*Алгоритм процедуры:*

Происходит объявление переменных nalB, cost. Если значение столбца Baggage в таблице tickets при равенстве ID рейса в таблице tickets и flight и выходных данных будет равно 1, то переменной costticket присваивается значение суммы столбца price из таблицы tickets и переменной constB, иначе данной данной переменной присваивается просто значение столбца price из таблицы tickets. Далее происходит выборка столбцов таблиц для вывода.

Код процедуры:

CREATE DEFINER=`root`@`localhost` PROCEDURE `data\_about\_cost`(reis int)

DETERMINISTIC

BEGIN

declare nalB varchar(50);

declare cost int;

if(select Baggage from tickets join flight on flight.ID = tickets.Flight\_ID where tickets.Flight\_ID = reis )=1

then set nalB = "Багаж есть";

set cost = (select price from tickets join flight on flight.ID = tickets.Flight\_ID where tickets.Flight\_ID = reis ) + `3000`;

else set nalB = "Багажа нет";

set cost = (select price from tickets join flight on flight.ID = tickets.Flight\_ID where tickets.Flight\_ID = reis);

END if;

select ID as "Идентификатор рейса",

Number\_of\_flight as "Номер рейса",

nalB as "Наличие багажа",

cost as "Стоимость билета на рейс"

from flight

where flight.ID = reis;

END

Пример выполнения процедуры показан на *рисунке 19*

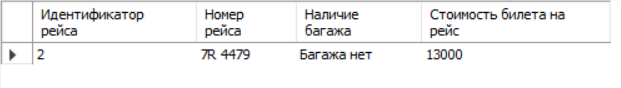


Рисунок 19– пример выполнения процедуры для вычисления стоимости билета по ID рейса

3) Процедура для вывода информации о пассажирах, летящих по рейсу с определенным ID

*Синтаксис процедуры:*

data\_about\_tick (IDflight int)

*Входные данные процедуры:*

IDflight – ID рейса, тип данных – int

*Алгоритм процедуры:*

Пользователь вводит ID рейса. И происходит выборка следующих данных соответственно входным данным: куда и откуда летит пассажир, кто является перевозчиком, номер рейса, даты из таблиц passenger, airport, flight, tickets.

Код процедуры:

CREATE DEFINER=`root`@`localhost` PROCEDURE `data\_about\_tick`(IDflight int)

DETERMINISTIC

begin

select concat(passenger.Name,' ',passenger.Second\_name,' ',passenger.Surname) as "ФИО пассижира",

a1.Short\_name as "FROM / ОТ",

a2.Short\_name as "TO / КУДА",

airline.Name as "ПЕРЕВОЗЧИК",

flight.Number\_of\_flight as "НОМЕР РЕЙСА",

tickets.Arrival\_date as "ДАТА",

tickets.Departure\_date as "ДАТА вылета"

from passenger

join tickets on tickets.Passenger\_ID=passenger.ID

join tickets ticket on tickets.Passenger\_ID = passenger.ID

join flight on flight.ID = tickets.Flight\_ID

join airport a1 on flight.Airport\_number1ID = a1.ID

join airport a2 on flight.Airport\_number2ID = a2.ID

join airline on `airline`.ID = flight.Airline\_ID

where flight.ID = IDflight

limit 100;

end

Пример выполнения процедуры представлен на *рисунке 20*

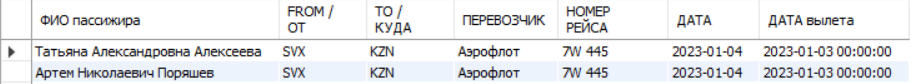


Рисунок 20 – пример выполнения процедуры для вывода информации о пассажирах, летящих по рейсу с определенным ID

4) Процедура для вычисления свободных мест по определенному рейсу

*Синтаксис процедуры:*

Kol\_pas\_flight (idflight int)

*Входные данные процедуры:*

idflight – ID рейса, тип данных – int

*Алгоритм процедуры:*

Пользователь вводит ID рейса. Происходит объявление переменных kolO, kolichestvo, ostatok. Переменной kolO присваивается значение столбца Capacity таблицы airplane, а переменной kolichestvo количество пассажиров на данном рейсе. Переменная ostatok равна разнице двух предыдущих переменных, что будет являться количеством свободных мест на рейс.

Код процедуры:

CREATE DEFINER=`root`@`localhost` PROCEDURE `Kol\_pas\_flight`(idflight int)

DETERMINISTIC

BEGIN

declare kolichestvo int;

declare ostatok int;

declare kolO int;

set kolO = (select Capacity from airplane join flight on flight.Airplane\_ID = airplane.ID where flight.ID = idflight);

set kolichestvo = (select count(Passenger\_ID) from tickets join passenger on passenger.ID = tickets.Passenger\_ID where tickets.Flight\_ID = idflight);

set ostatok = kolO - kolichestvo;

select kolichestvo as "Количество пассажиров на данном рейсе",

ostatok as "Количество свободных мест на данный рейс",

Number\_of\_flight as "Номер рейса"

from flight

where flight.ID = idflight;

END

Пример выполнения процедуры представлен на *рисунке 21*

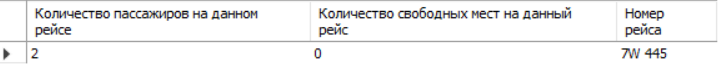


Рисунок 21 - пример выполнения процедуры для вычисления свободных мест на рейс по его ID

5) Процедура для вычисления возраста самолета и его пригодности для эксплуатации

*Синтаксис процедуры:*

vozrast\_airplane (idp int)

*Входные данные процедуры:*

idp – ID самолета, тип данных – int

*Алгоритм процедуры:*

Пользователь вводит ID самолета. Происходит объявление переменной vozrast. Если его возраст, рассчитываемый по date\_birth таблицы airplane меньше 20, то в переменную vozrast присваивается значение "Пригоден для перелётов", иначе - "Не пригоден для перелётов". Далее происходит выборка данных для вывода.

Код процедуры:

CREATE DEFINER=`root`@`localhost` PROCEDURE `vozrast\_airplane`(idp int)

DETERMINISTIC

BEGIN

declare vozrast varchar(30);

if (select (DATE\_FORMAT(FROM\_DAYS(TO\_DAYS(now()) - TO\_DAYS(Entry\_date)), '%Y') + 0) from airplane where airplane.ID=idp )<20

then set vozrast = "Пригоден для перелётов";

else set vozrast = "Не пригоден для перелётов";

END if;

select Model as "Модель самолета",

vozrast as "Экспуатация",

Entry\_date as "Дата ввода в эксплуатацию" from airplane

where airplane.ID = idp;

END

Пример выполнения процедуры представлен на *рисунке 22*

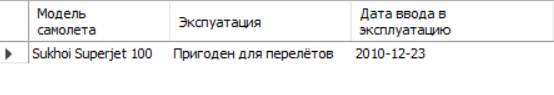


Рисунок 22 – пример выполнения процедуры для вычисления пригодности и возраста самолета по его ID

**Триггеры**

1) Триггер для проверки наличия билетов на рейс

Название: tickets\_BEFORE\_INSERT- предназначен для проверки наличия билетов на рейс.

Алгоритм: добавляется новый билет на определенный рейс, и триггер проверяет по количеству мест в самолете, есть ли еще свободные билеты на данный рейс.

Код триггера:

CREATE DEFINER=`root`@`localhost` TRIGGER `tickets\_BEFORE\_INSERT` BEFORE INSERT ON `tickets` FOR EACH ROW BEGIN

declare capacit int;

set capacit = (select capacity from airplane where ID= (select Airplane\_ID from flight where ID = new.Flight\_ID));

if capacit = (select flight.`Number\_of\_tickets\_sold` from flight where ID = new.Flight\_ID )

then signal sqlstate '45000'set message\_text = "Все билеты на данный рейс проданы";

end if;

END

2) Триггер для прибавления значения по столбцу проданных на рейсы билетов

Название: tickets\_AFTER\_INSERT- предназначен для прибавления значения по столбцу проданных на рейсы билетов

Алгоритм: когда добавляется новый билет, то происходит увеличение на единицу в столбце Number\_of\_ticket\_sold.

Код триггера:

CREATE DEFINER=`root`@`localhost` TRIGGER `tickets\_AFTER\_INSERT` AFTER INSERT ON `tickets` FOR EACH ROW BEGIN

update flight set `Number\_of\_tickets\_sold` = `Number\_of\_tickets\_sold` + 1 where ID = new.Flight\_ID;

END

3) Триггер для уменьшения значения по столбцу проданных на рейсы билетов

Название: tickets\_AFTER\_DELETE- предназначен для уменьшения значения по столбцу проданных на рейсы билетов

Алгоритм: когда удаляется билет, то происходит уменьшение на единицу в столбце Number\_of\_ticket\_sold.

Код триггера:

CREATE DEFINER=`root`@`localhost` TRIGGER `tickets\_AFTER\_DELETE` AFTER DELETE ON `tickets` FOR EACH ROW BEGIN

update flight set `Number\_of\_tickets\_sold` = `Number\_of\_tickets\_sold` - 1 where ID = old.Flight\_ID;

END

# **ЗАКЛЮЧЕНИЕ**

В рамках выполнения данного курсового проекта была создана функционирующая база данных аэропорта. В данном проекте реализован частичный функционал подобной системы, было создано пять процедур, две функции, два триггера и одно представление. Реализованы пользовательские функции для:

• для вычисления свободных мест на рейс по ID – free\_pl\_flight (idfl int)

• для вычисления возраста пассажира по его ID – vozrast\_passenger (idpas int)

Созданы процедуры для:

• вычисления стоимости билета на рейс при наличии багажа и без него для пассажира по билету с ID – cost\_ticket( idbilet int)

• вычисления стоимости на рейс с багажом или без по ID рейса – data\_about\_cost (reis int)

• вывода информации о пассажирах, летящих по рейсу с определенным ID – data\_about\_tick (IDflight int)

• вычисления свободных мест по определенному рейсу – Kol\_pas\_flight (idflight int)

• вычисления возраста самолета и его пригодности для эксплуатации –vozrast\_airplane (idp int)

Разработанные триггеры позволяют в автоматическом режиме обновлять информацию о проданных и возвращенных билетах, фиксировать в БД, когда билеты на определенный рейс заканчиваются.

Для всех таблиц БД были выявлены и проанализированы функциональные зависимости атрибутов, таблицы БД были приведены к 1НФ,

2НФ и 3НФ, чтобы избежать возможных аномалий и для поддержания целостности базы данных. В процессе выполнения были закреплены навыки работы в MySQL Workbench и MySQL Models, а также навыки разработки реляционных баз данных – создания функций, процедур, триггеров, запросов.

# **СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Стружкин, Н. П. Базы данных: проектирование. Практикум: пособие для вузов / Н. П. Стружкин, В. В. Годин. — Москва: Издательство Юрайт, 2020 — 291 с. — Текст: электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/451246 (дата обращения: 04.06.2021).
2. Ульман, Л. MySQL / Л. Ульман. — Москва: ДМК Пресс, 2008 — 352 с. — ISBN 5-94074-229-7. — Текст: электронный // Лань: ЭБС. — URL: https://e.lanbook.com/book/1241обращения: 04.06.2021). — Режим доступа: для авториз. пользователей.
3. Мартишин, С. А. Проектирование и реализация баз данных в СУБД MySQL с использованием MySQL Workbench: учеб. пособие / С.А. Мартишин, В.Л. Симонов, М.В. Храпченко. — М.: ИД «ФОРУМ»: ИНФРА-М, 2019 – 160 с. – Текст: электронный. – URL: <https://znanium.com/catalog/product/1007949> (дата обращения: 04.06.2021). – Режим доступа: по подписке.
4. Мартишин, С. А. Базы данных. Практическое применение СУБД SQL- и NoSQL-типа для проектирования информационных систем: учебное пособие / С.А. Мартишин, В.Л. Симонов, М.В. Храпченко. — Москва: ФОРУМ: ИНФРА-М, 2021 — 368 с. — Текст: электронный. – URL: https://znanium.com/catalog/product/1243192 (дата обращения: 04.06.2021). – Режим доступа: по подписке.
5. Ikirov.ru [сайт] - URL: https://www.ikirov.ru/news/4583-chem-otlichaetsya-turfirma-turagentstvo-turoperator (дата обращения: 04.06.2021).
6. Шнырев С.Л. Базы данных: Учебное пособие. М.: НИЯУ МИФИ, 2011. - 224 с. - Текст: электронный // Лань: электронно-библиотечная система. - URL: https://e.lanbook.com/book/75809 (дата обращения: 04.06.2021). - Режим доступа: для авторизированных пользователей.
7. Осипов Д. Л. Технологии проектирования баз данных. - М.: ДМК Пресс, 2019. - 498 с.: ил. - URL: https://e.lanbook.com/book/131692 (дата обращения: 04.06.2021). - Режим доступа: для авторизированных пользователей.
8. Голицына, О. Л. Базы данных / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - М.: Форум, 2015. - 400 c
9. Кириллов, В.В. Введение в реляционные базы данных (+ CD-ROM) / В.В. Кириллов. - М.: БХВ-Петербург, 2016. - 318 c.
10. Костин, А. Е. Организация и обработка структур данных в вычислительных системах. Учебное пособие / А.Е. Костин, В.Ф. Шаньгин. - М.: Высшая школа, **2014.** - 248 c.