**Государственное бюджетное профессиональное образовательное учреждение**

**Астраханской области «Астраханский колледж вычислительной техники»**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **К защите допустить:** | | | | | | | |
| **Старший методист** | | | | | | | |
|  |  | | | **Л.О. Земцова** | | | |
| **«** |  | **»** |  | | **20** |  | **г.** |

**курсовая работа**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| по МДК | 02.02 | Технология разработки и защиты баз данных | | | | | |
|  | | | | | | | |
|  | | 6 семестр | | | | | |
| Тема курсовой работы | | Разработка базы данных | | | | | |
|  | | для подбора авиабилетов | | | | | |
|  | |  | | | | | |
| Аквт.09.02.07.Кр00.24ПЗ | | | | | | | |
| Листов: | | | 24 |  | | | |
|  | |  | | | | | |
|  | |  | | | | | |
| Разработчик | |  | | | Студент гр. | | ПБ – 32 |
|  | |  | | | Д.А Тюлегенов | | |
|  | |  | | |  |  | |
| Руководитель | |  | | | Преподаватель | | |
|  | |  | | | И.А. Храмцовский | | |

**2023**

|  |
| --- |
| УТВЕРЖДЕНО |
| Цикловой комиссией |
| специальности 09.02.07 |
| ПЦК \_\_\_\_\_\_\_\_\_ Ю.С. Андрианова |
| «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. |
| Специальность | *09.02.07 Информационные системы и программирование* | | |
| МДК 04.01 | *Технология разработки и защиты баз данных* | | |
| Группа | *ПБ– 32* | | |
| **ЗАДАНИЕ** | | | |
| **на курсовое проектирование** | | | |
| ***Тюлегенов Даниял Асхарович*** | | | |
| (фамилия, имя, отчество студента) | | | |
| Тема курсового проекта | | ***Разработка Базы данных*** | |
| ***для подбора авиабилетов*** | | | |

Содержание пояснительной записки курсового проекта

Содержание

Введение

1. Проектирование базы данных
2. Анализ и описание предметной области информационной системы
3. Проектирование концептуальной модели
4. Построение логической модели, схемы базы данных
5. Выбор СУБД
6. Реализация базы данных
7. Физическое проектирование
8. Написание исходного кода БД
9. Написание запросов, функций и хранимых процедур

Заключение

Список использованных источников

Приложения

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Дата выдачи задания | « |  | » |  | |  | 20 |  | г. |
| Руководитель |  | | | | И.А. Храмцовский | | | | |
| Студент |  | | | | Д.А Тюлегенов | | | | |

**Аннотация**

В современном мире авиационная индустрия является одной из самых динамично развивающихся и востребованных отраслей. Сотни тысяч людей ежедневно совершают авиаперелеты, будь то для деловых целей, туризма или личных поездок. При выборе авиабилетов клиентам необходимо учесть множество факторов, таких как цена, доступность, расписание, авиакомпания и многое другое. Чтобы упростить процесс выбора и подбора наиболее подходящего рейса, разработка базы данных подбора авиабилетов является актуальной и значимой задачей.

Целью данной курсовой работы является разработка базы данных для сервиса подбора авиабилетов, который предоставит клиентам удобный и эффективный инструмент для выбора оптимальных рейсов.

**Содержание**

[Введение 5](#_Toc135224632)

[1 Проектирование базы данных 6](#_Toc135224633)

[1.1 Анализ и описание предметной области информационной системы 7](#_Toc135224634)

[1.2 Проектирование концептуальной модели 8](#_Toc135224635)

[1.3 Построение логической модели, схемы базы данных 9](#_Toc135224636)

[1.4 Выбор СУБД 10](#_Toc135224637)

[2 Реализация базы данных 11](#_Toc135224638)

[2.1 Физическое проектирование 12](#_Toc135224639)

[2.2 Написание исходного кода БД 13](#_Toc135224640)

[2.3 Написание запросов, функций и хранимых процедур 14](#_Toc135224641)

[Заключение 15](#_Toc135224642)

[Список использованных источников 16](#_Toc135224643)

[Приложение 17](#_Toc135224644)

**Введение**

Данная курсовая работа посвящена разработке базы данных для подбора авиабилетов с целью автоматизации процесса и предоставления клиентам выгодных цен на авиабилеты. В современном мире авиаперелеты становятся все более популярными, и вместе с ростом спроса возникает необходимость в эффективных инструментах подбора и бронирования билетов.

Цель данного проекта заключается в создании базы данных, которая позволит клиентам узнавать всю необходимую информацию о доступных рейсах, включая уникальные номера рейсов, выбранные направления, стоимость билетов, а также данные об аэропортах отправления и прибытия, времени отправления и прилета.

Основными задачами проекта являются экономия времени и денег клиентов, а также оптимизация расходов авиакомпаний. Благодаря базе данных, пользователи смогут легко находить и выбирать оптимальные варианты перелетов, сравнивать цены и получать полную информацию о каждом рейсе. Это поможет клиентам сэкономить время, необходимое для поиска и сравнения различных предложений, а также снизить расходы на покупку билетов.

Для реализации проекта предлагается использовать таблицы, включающие информацию об аэропортах, пассажирах, рейсах, билетах и местах в салоне. Вся информация будет храниться в структурированном виде, что обеспечит быстрый и удобный доступ к необходимым данным.

Результаты данного проекта будут полезны как для клиентов, так и для авиакомпаний. Клиенты получат удобный и надежный инструмент для поиска и бронирования авиабилетов, а авиакомпании смогут оптимизировать процессы продажи билетов, управлять загруженностью рейсов и снизить затраты на маркетинг и рекламу.

В заключение, разработка базы данных для подбора авиабилетов позволит автоматизировать процесс поиска и бронирования билетов, сделает его более эффективным и удобным для клиентов, а также способствует оптимизации работы авиакомпаний и снижению их расходов.

# **Проектирование базы данных**

## **Анализ и описание предметной области информационной системы**

Анализ и описание предметной области информационной системы авиаперелетов. Перед разработкой базы данных необходимо провести анализ и описание предметной области. В данном случае предметная область - это система подбора авиабилетов.

Целью данной курсовой работы является разработка базы данных для сервиса подбора авиабилетов, который предоставит клиентам удобный и эффективный инструмент для выбора оптимальных рейсов.

Описание предметной области:

Существуют различные аэропорты, представленные в таблице "airport". Каждый аэропорт имеет уникальный идентификатор, наименование, а также связан с определенным городом и страной.

Пользователи приложения представлены в таблице "users". Каждый пользователь имеет уникальный идентификатор, логин, пароль, номер телефона и адрес электронной почты.

Рейсы представлены в таблице "flight". Каждый рейс имеет уникальный идентификатор, указание авиакомпании, номер рейса, а также аэропорт отправления и прибытия, время отправления и прибытия.

Информация о билетах хранится в таблице "Ticket". Каждый билет имеет уникальный идентификатор, связь с пассажиром и рейсом, а также цену.

Пассажиры представлены в таблице "Passenger". Каждый пассажир имеет уникальный идентификатор, имя и связь с пользователем.

Места в салоне самолета хранятся в таблице "Seat". Каждое место имеет уникальный идентификатор, ряд и номер места.

Информация о странах представлена в таблице "Country". Каждая страна имеет уникальный идентификатор и название.

Города представлены в таблице "City". Каждый город имеет уникальный идентификатор, название, временную зону по Гринвичу и связь с определенной страной.

Это описание предметной области предоставляет общее представление о структуре и связях между сущностями, необходимыми для разработки базы данных по подбору авиабилетов.

## **Проектирование концептуальной модели**

На этапе концептуального проектирования использовалась модель «сущонсть-связь». Как видно из рисунка 1, в предметной области выделено 8 сущностей: пользователи (users), пассажиры(Passenger), страны(Country), аэропорты(airport), города(City), билеты(ticket), перелет(flight), место(Seat). Связи между сущностями также изображены на рисунке. Атрибуты сущностей и их типы более подробно будут рассмотрены на этапе логического проектирования базы данных.

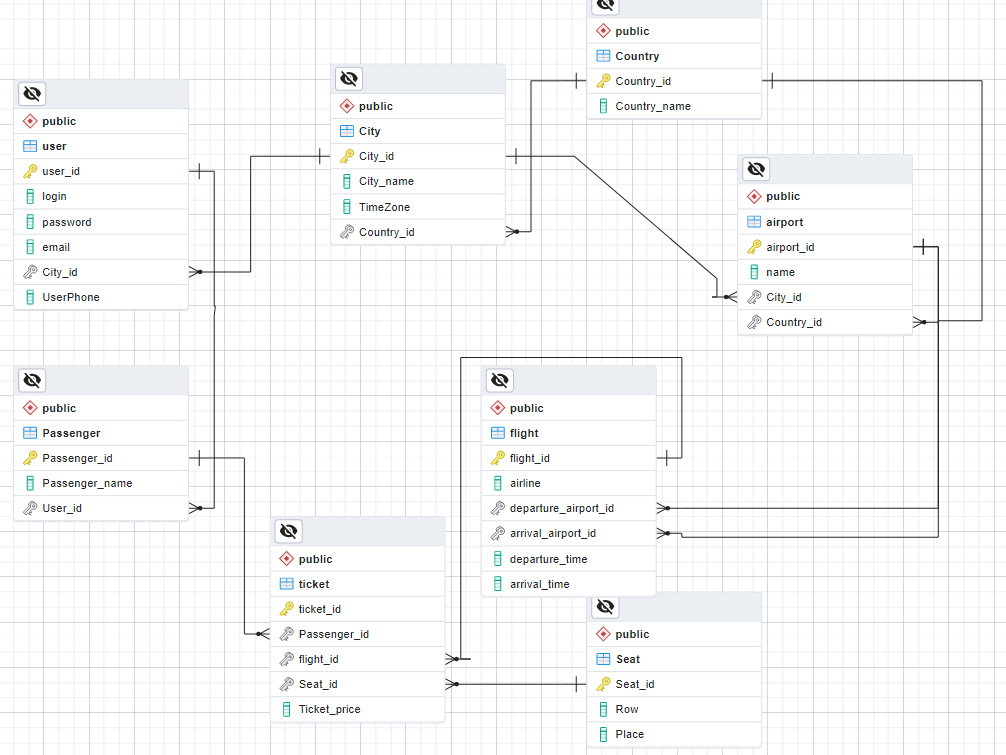


Рисунок 1 – ER-модель разработанной базы данных

Концептуальная модель базы данных представляет собой абстрактное описание предметной области и основных сущностей, связей и атрибутов, которые будут участвовать в базе данных. Она помогает лучше понять и структурировать информацию, а также определить основные требования к базе данных.

## **Построение логической модели, схемы базы данных**

На этом этапе работы производится создание таблиц базы данных, которые отражают основные сущности и их связи. Каждая таблица имеет уникальный идентификатор (PK - Primary Key), который используется для однозначной идентификации записей в таблице. Также устанавливаются внешние ключи (FK - Foreign Key), которые связывают записи в разных таблицах.

Например, таблица "airport" содержит информацию об аэропортах и имеет столбец "airport\_id" в качестве первичного ключа. Также в этой таблице есть столбцы "city\_id" и "country\_id", которые являются внешними ключами и связаны с соответствующими таблицами "city" и "Country".

Аналогичным образом, в других таблицах также создаются первичные ключи для уникальной идентификации записей, а также внешние ключи для связи с другими таблицами. Например, таблица "Tickets" содержит столбец "ticket\_id" в качестве первичного ключа и столбцы "passenger\_id" и "flight\_id" как внешние ключи, связывающие записи с таблицами "Passenger" и "Flights".

Таким образом, создание логической модели и схемы базы данных включает определение таблиц, их структуры и связей, что позволяет эффективно организовать хранение и доступ к данным.

При построении логической модели и схемы базы данных для представленной базы данных, мы учитываем структуру данных, связи между сущностями и их атрибуты. Вот пример построения логической модели:

Логическая модель базы данных:

Таблица "airport":

* airport\_id (PK): идентификатор аэропорта
* name: наименование аэропорта
* city\_id (FK): идентификатор города
* country\_id (FK): идентификатор страны

Таблица "user":

* user\_id (PK): идентификатор пользователя
* login: логин пользователя
* password: пароль пользователя
* email: электронная почта пользователя
* city\_id (FK): идентификатор города пользователя
* UserPhone: номер пользователя

Таблица "flight":

* flight\_id (PK): идентификатор перелета
* airline: авиакомпания
* flight\_number: номер перелета
* departure\_airport\_id (FK): идентификатор аэропорта отправления
* arrival\_airport\_id (FK): идентификатор аэропорта прибытия
* departure\_time: время отправления
* arrival\_time: время прибытия

Таблица "Tickets":

* ticket\_id (PK): идентификатор билета
* passenger\_id (FK): идентификатор пассажира
* flight\_id (FK): идентификатор перелета
* price: цена билета
* seat\_id (FK): идентификатор места в салоне

Таблица "Passenger":

* passenger\_id (PK): идентификатор пассажира
* passenger\_name: имя пассажира
* user\_id (FK): идентификатор пользователя

Таблица "Seat":

* seat\_id (PK): идентификатор места
* row: ряд
* place: место

Таблица "Country":

* country\_id (PK): идентификатор страны
* country\_name: название страны

Таблица "City":

* city\_id (PK): идентификатор города
* city\_name: название города
* time\_zone: зона времени по Гринвичу
* country\_id (FK): идентификатор страны, в которой расположен город

## **Выбор СУБД**

При разработке базы данных для сервиса подбора авиабилетов, выбор подходящей системы управления базами данных (СУБД) является важным этапом. Необходимо учитывать различные факторы, такие как функциональность, производительность, масштабируемость, надежность, доступность, стоимость и требования к проекту.

Одной из популярных и мощных СУБД в этой области является PostgreSQL, также известный как Postgres. PostgreSQL является открытой СУБД с широким набором функциональных возможностей, высокой производительностью и надежностью. Она поддерживает множество расширений и предоставляет дополнительные возможности для работы с геоданными, полнотекстовым поиском, транзакциями, многопоточностью и другими функциями, которые могут быть полезны при разработке базы данных для сервиса подбора авиабилетов.

Основные преимущества PostgreSQL включают открытость и свободную лицензию, что дает возможность свободного использования, изменения и распространения системы. Также PostgreSQL предлагает расширяемость с помощью расширений и плагинов, которые позволяют добавлять дополнительные функции и возможности к базе данных. Благодаря эффективному механизму запросов, оптимизатору запросов, поддержке индексов и параллельной обработке запросов, PostgreSQL обеспечивает высокую производительность даже при работе с большим объемом данных. Она также гарантирует надежность с помощью обеспечения целостности данных, поддержки транзакций и механизмов восстановления после сбоев. Кроме того, PostgreSQL имеет активное сообщество разработчиков и пользователей, что обеспечивает поддержку, документацию и помощь в решении проблем.

Исходя из всех этих факторов, я выбрал PostgreSQL в качестве СУБД для разработки базы данных подбора авиабилетов. Ее функциональность, производительность и открытая лицензия позволяют свободно использовать и настраивать систему в соответствии с требованиями проекта.

# **Реализация базы данных**

## **Физическое проектирование**

Физическое проектирование базы данных включает создание таблиц, определение их полей и атрибутов, а также установление связей между таблицами. Основываясь на предоставленных таблицах, проведем физическое проектирование базы данных для сервиса подбора авиабилетов:

1. Таблица "airport": Содержит информацию об аэропортах, используемых в системе, используется для связи аэропортов с городами и странами.

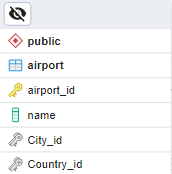


Рисунок 2 – Таблица airport

1. Таблица "user": Содержит информацию о зарегистрированных пользователях сервиса, используется для хранения данных о пользователях и связи их с соответствующими перелетами и городами.

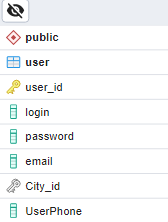


Рисунок 3 – Таблица user

1. Таблица "flight": Содержит информацию о перелетах, доступных в системе, используется для хранения данных о перелетах и связи их с соответствующими аэропортами.

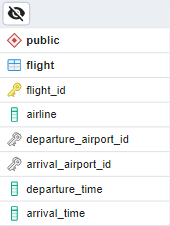


Рисунок 4 – Таблица flight

1. Таблица "ticket": Содержит информацию о билетах на перелеты, используется для связи билетов с пассажирами, перелетами и местами в салоне

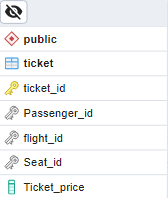


Рисунок 5 -Таблица ticket

1. Таблица "Passenger": Содержит информацию о пассажирах, используется для связи пассажиров с пользователями.

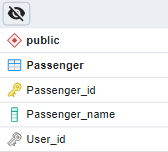


Рисунок 6 – Таблица Passenger

1. Таблица "Seat": Содержит информацию о местах в салоне самолета, используется для хранения данных о доступных местах в салоне.

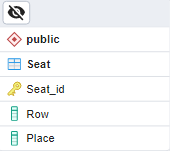


Рисунок 7 – Таблица Seat

1. Таблица "Country": Содержит информацию о странах, используется для хранения данных о странах.



Рисунок 8 – Таблица Country

Таблица "City":

Содержит информацию о городах, используется для хранения данных о городах и их связи с соответствующими странами.

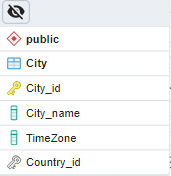


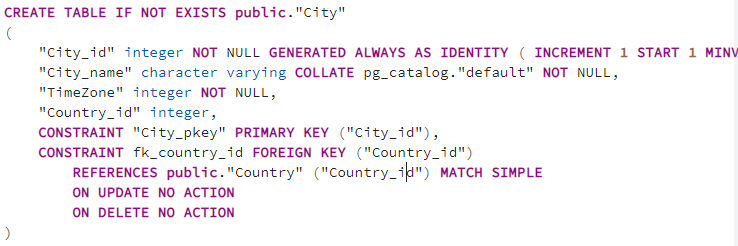
Рисунок 9 – Таблица City

Эти таблицы вместе образуют структуру базы данных, которая позволяет хранить и связывать информацию о аэропортах, пользователях, перелетах, билетах, пассажирах, местах в салоне, странах и городах, необходимую для функционирования сервиса подбора авиабилетов.

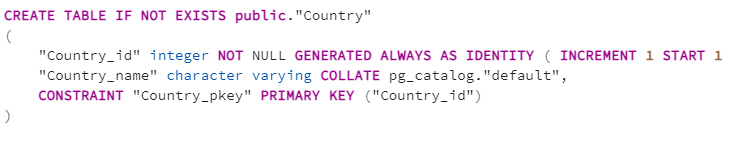
## **Написание исходного кода БД**

Для создания таблиц в базе был написан код, описанный ниже.

* Код для таблицы City (Рисунок 10):

  
Рисунок 10 – Код для таблицы City

* Код для таблицы Country (Рисунок 11):

  
Рисунок 11 – код для таблицы Country

* Код для таблицы Passenger (Рисунок 12):

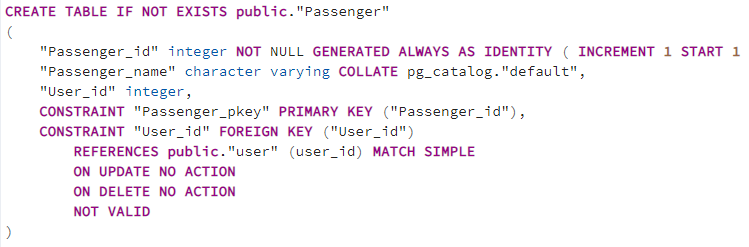


Рисунок 12 – код для таблицы Passenger

* Код для таблицы Seat (Рисунок 13):

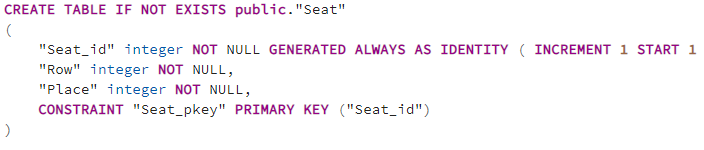


Рисунок 13 – код для таблицы Seat

* Код для таблицы airport (Рисунок 14):

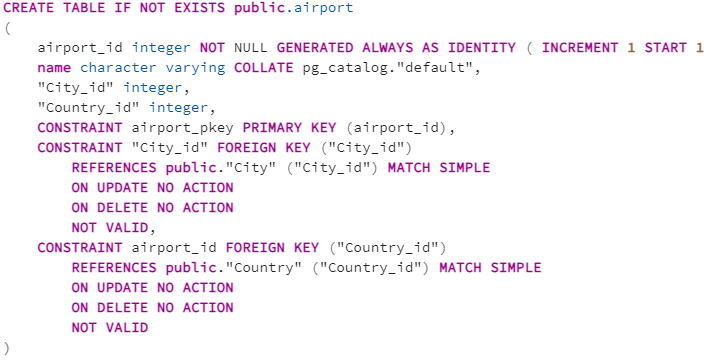


Рисунок 14 – код для таблицы airport

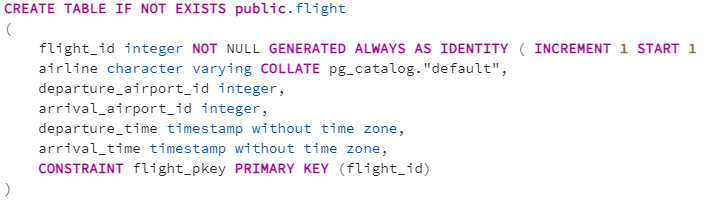
* Код для таблицы flight (Рисунок 15):  
  

Рисунок 15 – код для таблицы flight

* Код для таблицы ticket (Рисунок 16):

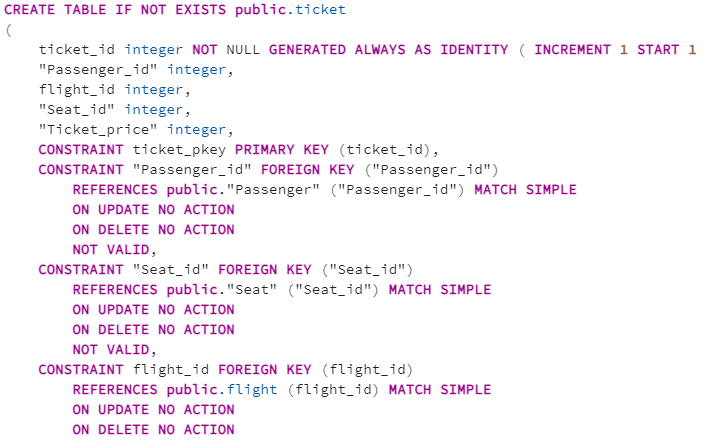


Рисунок 16 – код для таблицы ticket

* Код для таблицы user (Рисунок 17):

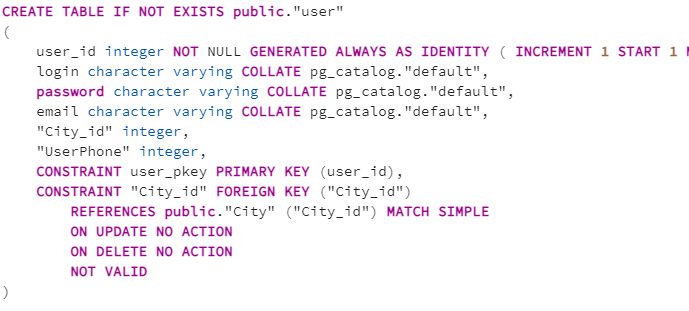


Рисунок 17 – код для таблицы user

## **Написание запросов, функций и хранимых процедур**

Для работы с базой данных написаны следующие запросы:

1. SELECT "сity"."City\_name", COUNT("user"."user\_id") AS passenger\_count

FROM public."сity" LEFT JOIN "user" ON public."сity"."City\_id" = "user"."City\_id" GROUP BY "сity"."City\_name" ORDER BY passenger\_count desc

Этот запрос выбирает информацию с колонки "City\_name" из таблицы "city" и использует функцию count, чтобы подсчитать количество значений столбца "user\_id" из таблицы "user". Затем запрос выполняет LEFT JOIN, чтобы объединить таблицу "city" с таблицей "user" на основе значений столбца "City\_id", далее запрос группирует записи. В итоге было выведено 249 записей (Рисунок 18)



Рисунок 18 – Результат запроса по пассажирам

Итак, этот запрос выполняет следующие действия:

* Выбирает название города и подсчитывает количество пассажиров для каждого города.
* Объединяет таблицы "сity" и "user" по значению столбца "City\_id".
* Группирует результаты по названию города.
* Сортирует результаты по убыванию количества пассажиров.

Таким образом, этот запрос поможет вам получить список городов, отсортированных по количеству пассажиров, проживающих в каждом городе.

1. SELECT flight.airline,

COUNT(DISTINCT CASE WHEN EXTRACT(YEAR FROM flight.departure\_time) = 2019 THEN "user"."user\_id" END) AS "2019",

COUNT(DISTINCT CASE WHEN EXTRACT(YEAR FROM flight.departure\_time) = 2020 THEN "user"."user\_id" END) AS "2020",

COUNT(DISTINCT CASE WHEN EXTRACT(YEAR FROM flight.departure\_time) = 2021 THEN "user"."user\_id" END) AS "2021",

COUNT(DISTINCT CASE WHEN EXTRACT(YEAR FROM flight.departure\_time) = 2022 THEN "user"."user\_id" END) AS "2022",

COUNT(DISTINCT CASE WHEN EXTRACT(YEAR FROM flight.departure\_time) = 2023 THEN "user"."user\_id" END) AS "2023"

FROM flight LEFT JOIN ticket ON flight.flight\_id = ticket.flight\_id LEFT JOIN "user" ON ticket."Passenger\_id" = "user"."user\_id" WHERE flight.departure\_time >= '2019-01-01' AND flight.departure\_time <= '2023-12-31' GROUP BY flight.airline;

Запрос делает следующее:

Связывает таблицы flight, ticket и user, чтобы получить информацию о пассажирах, билетах и перелетах.

Использует функцию COUNT (DISTINCT ...) в сочетании с условными выражениями (CASE WHEN ...) для подсчета уникальных пользователей для каждой авиакомпании в каждом году отдельно.

Группирует результаты по авиакомпаниям с помощью GROUP BY flight.airline.

Ограничивает результаты только перелетами, которые произошли в указанном периоде с помощью WHERE flight.departure\_time >= '2019-01-01' AND flight.departure\_time <= '2023-12-31'. В итоге было выведено 79 записей (Рисунок 19)

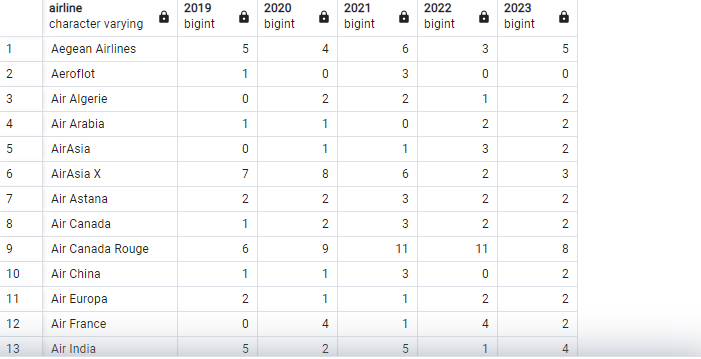


Рисунок 19 – Результат запроса по авиакомпаниям

Таким образом, этот запрос позволяет анализировать популярность и использование различных авиакомпаний в течение заданного периода, основываясь на количестве уникальных пользователей, связанных с каждой компанией.

1. SELECT ticket."Ticket\_price", "user"."login", flight."airline", airport."name" AS departure\_airport, destination\_airport."name" AS arrival\_airport FROM ticket JOIN "user" ON ticket."Passenger\_id" = "user"."user\_id" JOIN flight ON ticket."flight\_id" = flight."flight\_id" JOIN airport ON flight.departure\_airport\_id = airport.airport\_id JOIN airport AS destination\_airport ON flight.arrival\_airport\_id = destination\_airport.airport\_id WHERE ticket."Ticket\_price" > ( SELECT AVG("Ticket\_price") FROM ticket )

Запрос делает следующее:

SELECT: Выбирает нужные столбцы для вывода информации, включая цену билета (ticket."Ticket\_price"), логин пользователя ("user"."login"), название авиакомпании (flight."airline"), имя отправного аэропорта (airport."name") и имя прибытия аэропорта (destination\_airport."name").

FROM: Указывает источник данных для запроса, который состоит из таблицы ticket.

JOIN: Объединяет таблицы ticket, "user", flight и airport по соответствующим идентификаторам, чтобы получить информацию о билетах, пассажирах, перелетах и аэропортах.

WHERE: Устанавливает условие, что цена билета (ticket."Ticket\_price") должна быть выше средней стоимости билетов из таблицы ticket. Для этого используется подзапрос, который вычисляет среднюю стоимость билетов.

ORDER BY: Сортирует результаты по цене билета (ticket."Ticket\_price") в порядке убывания. В итоге было выведено 495 записей (Рисунок 20)

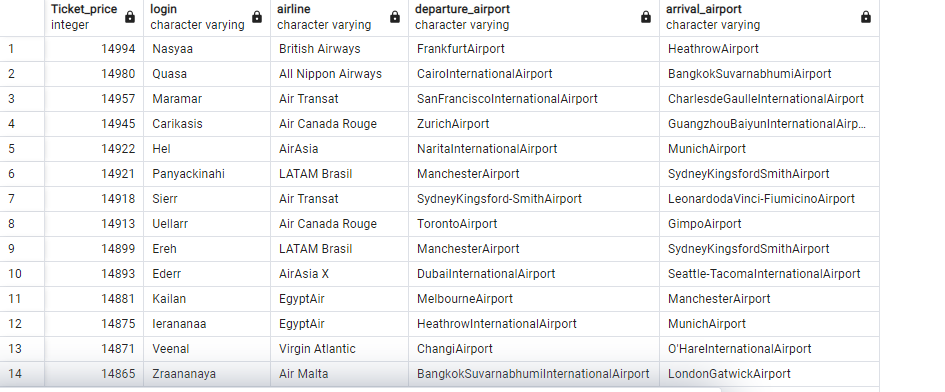


Рисунок 20 – Результат запроса по билетам

Таким образом, запрос позволяет получить список билетов с высокой стоимостью, включая информацию о цене, логине пользователя, названии авиакомпании и именах отправного и аэропорта прибытия. Результаты сортируются по цене билета в порядке убывания. Это может быть полезным для идентификации и анализа дорогих билетов, связанных с конкретными пользователями, авиакомпаниями и аэропортами отправления и прибытия.

1. SELECT flight."airline", COUNT(DISTINCT ticket."Passenger\_id") AS passenger\_count FROM flight JOIN ticket ON flight."flight\_id" = ticket."flight\_id" GROUP BY flight."airline" HAVING COUNT(DISTINCT ticket."Passenger\_id") > 5 ORDER BY passenger\_count DESC;

Запрос делает следующее:

SELECT: Выбирает столбцы для вывода, включая название авиакомпании (flight."airline") и количество уникальных пассажиров (COUNT(DISTINCT ticket."Passenger\_id")) для каждой авиакомпании.

FROM: Указывает источник данных для запроса, который состоит из таблицы flight.

JOIN: Объединяет таблицы flight и ticket по идентификатору перелета ("flight\_id"), чтобы получить информацию о билетах на рейсы.

GROUP BY: Группирует результаты по названию авиакомпании (flight."airline").

HAVING: Устанавливает условие, что количество уникальных пассажиров (COUNT(DISTINCT ticket."Passenger\_id")) должно быть больше 100. Таким образом, в результатах будут только те авиакомпании, у которых количество пассажиров превышает 5.

ORDER BY: Сортирует результаты по количеству пассажиров (passenger\_count) в порядке убывания. В итоге было выведено 75 записей (Рисунок 21)

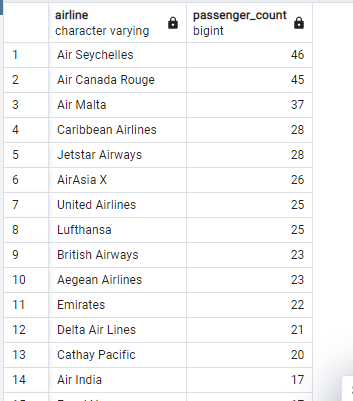


Рисунок 21- результат запроса по заказам авиалиний

**Заключение**

В данной курсовой работе была разработана база данных для подбора авиабилетов с целью автоматизации и улучшения процесса подбора выгодных цен на авиабилеты для клиентов. Главная цель автоматизации заключалась в экономии времени и денег клиентов, а также оптимизации расходов авиакомпаний.

База данных охватывает основные аспекты предметной области, включая информацию о рейсах, авиакомпаниях, билетах, пассажирах, аэропортах, городах и странах. Она позволяет хранить и обрабатывать данные о различных рейсах, включая уникальные номера рейсов, выбранные направления и стоимость билетов.

Благодаря базе данных пользователи могут получить всю необходимую информацию о перелетах, доступные цены на билеты, а также осуществить поиск по различным критериям, таким как дата и время отправления, авиакомпания, место прибытия и другие. Это значительно облегчает процесс выбора и бронирования авиабилетов, а также позволяет сэкономить время и ресурсы.

Основная функциональность базы данных включает возможность хранения, обновления, добавления и удаления данных о рейсах, авиакомпаниях, билетах и пассажирах. Также предусмотрены различные запросы для анализа данных, такие как подсчет количества пассажиров по авиакомпаниям, определение средней стоимости билетов, выявление популярных направлений и другие.

Разработанная база данных имеет потенциал для дальнейшего развития и расширения функциональности. Возможные улучшения могут включать добавление дополнительных таблиц и связей, реализацию функций авторизации и регистрации пользователей, а также интеграцию с системами оплаты и бронирования.

В целом результатом курсовой работы является установка базы данных авиаперелетов, которая отвечает всем требованиям подбора авиабилетов. Она способствует повышению эффективности и удобства процесса выбора и покупки авиабилетов, а также помогает оптимизировать затраты авиакомпаний и обеспечить лучший опыт путешествий для пассажиров.

**Список использованных источников**

**<https://www.postgresql.org/docs/current/datatype-datetime.html>**

[**https://www.postgresql.org/docs/current/sql-insert.html**](https://www.postgresql.org/docs/current/sql-insert.html)

[**https://metanit.com/sql/postgresql/**](https://metanit.com/sql/postgresql/)

[**https://metanit.com/sql/postgresql/4.1.php**](https://metanit.com/sql/postgresql/4.1.php)

[**https://metanit.com/sql/postgresql/4.7.php**](https://metanit.com/sql/postgresql/4.7.php)

[**https://metanit.com/sql/postgresql/6.2.php**](https://metanit.com/sql/postgresql/6.2.php)

**Приложение**