МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»
(Самарский университет)

Институт информатики, математики и электроники
Факультет информатики
Кафедра технической кибернетики

Отчет по лабораторной работе № 1

Дисциплина: “Databases in Enterprise Systems”

(«Корпоративные базы данных»)

Выполнила: Дубман Л. Б.

Группа: 6133-010402D

Самара 2024

Содержание

Задание на лабораторную работу №1

1. Выбрать предметную область

2. Разработать ER модель, включающую минимум 5-6 сущностей и типы связей: 1-N, N:M, 1-1.

3. Создать базу данных по модели в СУБД PostgreSQL.

4. Определить индексы, уникальные индексы.

5. Разработать типовые запросы к СУБД на языке SQL. Получение списков данных. Агрегация. Поиск.

6. Разработайте хранимые процедуры на языке PL/pgSQL для генерации случайных данных для базы данных.

7. Сгенерируйте тестовые данные при помощи разработанных процедур.

8. Протестируйте работу запросов на больших объёмах данных (Порядка 1 миллиона записей в основных таблицах).

9. Измените конфигурацию сервера PostgreSQL для достижения лучшей производительности на самых медленных запросах. Оптимизируйте схему БД и запросы для достижения лучшей производительности.

Пункты 6-7 допустимо реализовывать другими способами без PL/pgSQL

1. Выбор и описание ER-модели

Для работы с базой данных была придумана ER-модель на основе бизнес-модели авиационных перевозок. Данная модель состоит из двух видов сущностей и трех типов связей. Схема модели представлена на рисунке 1.

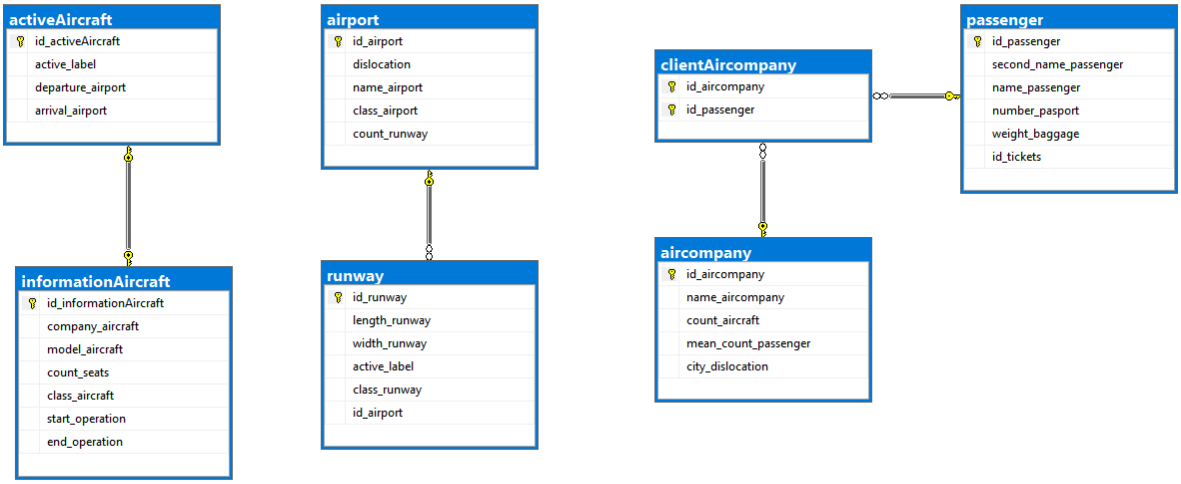

Рисунок 1 – Схема ER-модели бизнес-процесса

2. Разработать ER модель, включающую минимум 5-6 сущностей и типы связей: 1-N, N:M, 1-1.

В моей модели реализовано 7 сущностей, которые описывают те или иные объекты/субъекты авиационных перевозок. Рассмотрим их по подробнее.

2.1 Описание сущностей.

В качестве сущностей ER-модели выбраны реально существующие объекты модели авиационных перевозок. Описываемая модель включает следующие сущности:

- ActiveAircraft
- InformationAircraft
- Airport
- Runway
- Aircompany
- Passenger
- ClientAircompany

Далее рассмотрим каждую более подробно.

2.1.1 Сущность ActiveAircraft

Данная сущность необходима для описания действующего самолета. Сущность модели представлена на рисунке 2.

Рисунок 2 – Сущность ActiveAircraft

Рассмотрим их описание и характеристику.

Таблица 1 – Описание полей сущности ActiveAircraft

Название поля	Тип данных	Описание
id_activeAircraft	int	Является уникальным идентификатором, который используется для однозначной маркировки записей в таблице.
active_label	bool	Является маркером действительности самолета. Если выбрано значение true, то самолет действителен может летать, вследствие чего должен иметь место вылета и место прилета. Если выбрано значение false самолет считается не действительным и летать не может, а в места вылета и прилета имеют значения NULL.
departure_airport	varchar() string	Является полем, содержащим информацию о месте вылета (город) самолета, в случае если он действителен.
arrival_airport	varchar() string	Является полем, содержащим информацию о месте прилета (город) самолета, в случае если он действителен.

2.1.2 Сущность InformationAircraft

Данная сущность необходима для более полного описания действующего самолета и представлена на рисунке 3.

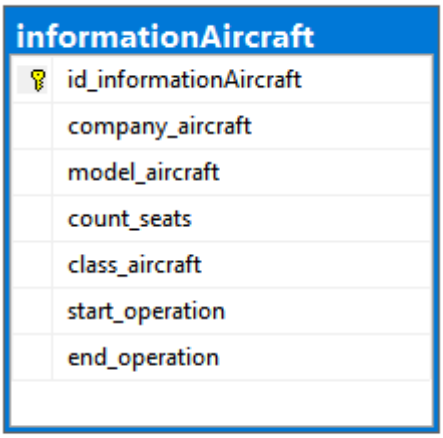

Рисунок 3 – Сущность InformationAircraft

Рассмотрим их описание и характеристику.

Таблица 2 – Описание полей сущности InformationAircraft

Название поля	Тип данных	Описание поля	Ключ
id_informationAircraft	int	Является уникальным идентификатором, который используется для однозначной маркировки записей в таблице.	Primary Key/Foreign Key
company_aircraft	VARCHAR() String	Является полем, содержащим информацию о производителе самолета.	—

Название поля	Тип данных	Описание поля	Ключ
model_aircraft	VARCHAR() String	Является полем, содержащим информацию о модели самолета.	——
count_seats	int	Является полем, содержащим информацию о количестве мест в самолете.	——
class_aircraft	VARCHAR() String	Является полем, содержащим информацию о классе самолета.	——
start_operation	Date	Является полем, содержащим информацию о дате начала эксплуатации самолета.	——
end_operation	Date	Является полем, содержащим информацию о дате окончания эксплуатации самолета.	——

2.1.3 Сущность Airport

Данная сущность необходима для описания аэропорта и представлена на рисунке 4.

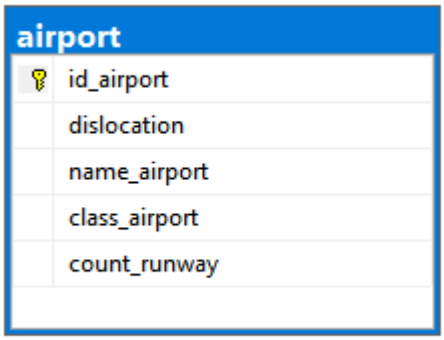

Рисунок 4 – Сущность Airport

Рассмотрим их описание и характеристику.

Таблица 3 – Описание полей сущности Airport

Название поля	Тип данных	Описание	Ключ
id_airport	int	Является уникальным идентификатором, который используется для однозначной маркировки записей в таблице.	Primary Key
dislocation	VARCHAR() string	Является полем, содержащим информацию о городе дислокации аэропорта.	——
name_airport	VARCHAR() string	Является полем, содержащим информацию о имени аэропорта.	——
class_airport	VARCHAR() string	Является полем, содержащим информацию	——

Название поля	Тип данных	Описание	Ключ
		о типе/классе аэропорта.	
count_runway	int	Является полем, содержащим информацию о количестве взлетных полос в аэропорту.	___

2.1.4 Сущность Runway

Данная сущность необходима для описания летной полосы аэропорта и представлена на рисунке 5.

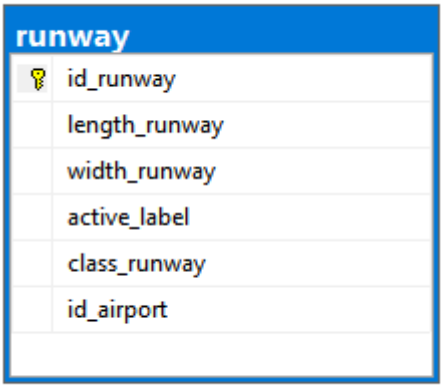

Рисунок 5 – Сущность Runway

Рассмотрим их описание и характеристику.

Таблица 4 – Описание полей сущности Runway

Название поля	Тип данных	Описание	Ключ
id_runway	int	Является уникальным идентификатором, который используется для однозначной маркировки записей в таблице.	Primary Key
lenght_runway	int	Является полем, содержащим информацию о длине взлетной полосы.	___
width_runway	int	Является полем, содержащим информацию о ширине взлетной полосы.	___
active_label	bool	Является маркером действительности взлетной полосы. Если выбрано значение true, то полоса доступна для работы, иначе не доступна.	___
class_runway	int	Является полем, содержащим информацию о классе взлетной полосы.	___
id_airport	int	Является полем, содержащим информацию об аэропорте, в котором находится взлетная полоса.	Foreign Key

2.1.5 Сущность Aircompany

Данная сущность необходима для описания авиакомпаний, осуществляющие авиаперевозки и представлена на рисунке 6.

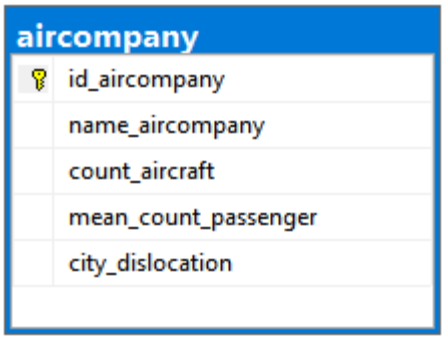

Рисунок 6 – Сущность Aircompany

Рассмотрим их описание и характеристику.

Таблица 5 – Описание полей сущности Aircompany

Название поля	Тип данных	Описание	Ключ
id_aircompany	int	Является уникальным идентификатором, который используется для однозначной маркировки записей в таблице.	Primary Key
name_aircompany	VARCHAR() string	Является полем, содержащим информацию о имени авиакомпании.	——
count_aircraft	int	Является полем, содержащим информацию о количестве самолетов в компании	——
mean_count_passenger	int	Является полем, содержащим информацию о среднем количестве пассажиров, перевозимых компанией.	——
city_dislocation	VARCHAR() string	Является полем, содержащим информацию о городе дислокации авиакомпании.	——

2.1.6 Сущность Passenger

Данная сущность необходима для описания клиентов авиакомпаний – пассажиров, и представлена на рисунке 7.

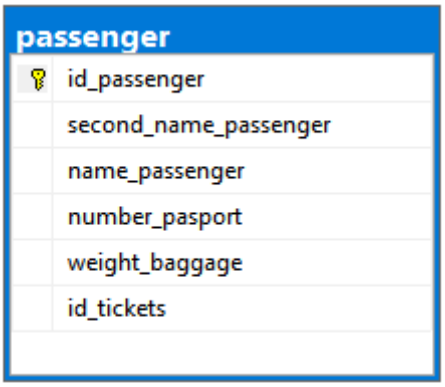

Рисунок 7 – Сущность Passenger

Рассмотрим их описание и характеристику.

Таблица 6 – Описание полей сущности Passenger

Название поля	Тип данных	Описание	Ключ
id_passenger	int	Является уникальным идентификатором, который используется для однозначной маркировки записей в таблице.	Primary Key
second_name_passenger	VARCHAR() String	Является полем, содержащим информацию о фамилии пассажира.	___
name_passenger	VARCHAR() String	Является полем, содержащим информацию о имени пассажира.	___
number_passport	int	Является полем, содержащим информацию о номере паспорта пассажира.	___
weight_baggage	int	Является полем, содержащим информацию о весе багажа пассажира.	___
id_tickets	int	Является полем, содержащим информацию о номере билета пассажира.	___

2.1.7 Сущность ClientAircompany

Данная сущность необходима для описания связи между пассажиром и авиакомпанией, и представлена на рисунке 8.

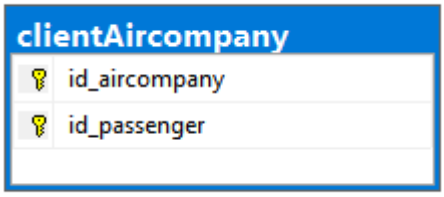

Рисунок 8 – Сущность clientAircompany

Рассмотрим их описание и характеристику.

Таблица 7 – Описание полей сущности Passenger

Название поля	Тип данных	Описание	Ключ
id_aircompany	int	Является уникальным идентификатором, который используется для однозначной маркировки записей в таблице.	Primary Key Foreign Key
id_passenger	int	Является уникальным идентификатором, который используется для однозначной маркировки записей в таблице.	Primary Key Foreign Key

2.2 Описание связей между сущностями

В моей ER-модели реализовано 3 типа связей:

- *1-1 – связь один к одному,*
- *1-N – связь один ко многим,*
- *N:M – связь многие ко многим.*

Далее рассмотрим каждый тип связи, и посмотрим какие сущности каким типом связи связаны.

2.2.1 Связь типа «Один к одному»

Тип связи	Описание
Один к одному	Данным типом связи, связаны 2 сущности: ***"действующий" самолет*** и ***информация об этом самолете,*** поскольку не может быть 2-х абсолютно одинаковых самолетов.

2.2.2 Связь типа «Один ко многим»

Тип связи	Описание
Один к многим	Данным типом связи, связаны 2 сущности: ***аэропорт*** и ***взлетные полосы*** поскольку аэропорт может иметь как одну, так и несколько взлетных полос.

2.2.3 Связь типа «Многие ко многим»

Тип связи	Описание
Многие ко многим	Данным типом связи, связаны 2 сущности, через промежуточную третью: ***авиакомпания и пассажир*** связаны через ***клиентов авиакомпаний.*** Поскольку, пассажир может быть зарегистрирован не в одной авиакомпании, а авиакомпании явно имеют более одного клиента(пассажира).

3 - 4. Создать базу данных по модели в СУБД PostgreSQL. Определить индексы, уникальные индексы.

Для последующего выполнения лабораторной работы нам потребуются следующие инструменты:

MS SQL Server 2019 Developer – SQL сервер, на котором мы и будем работать.

SQL Server Management Studio (SSMS) – утилита из Microsoft SQL Server 2019 для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server.

Далее будет показана, работа с базой данных. Скрипты, упоминаемые в данном отчете, расположены в следующем репозитории на сайте GitHub по ссылке:
https://github.com/WonMin13/EnterpriseDataBase/tree/main/Lab%20Work%20%231

3.1 Создание базы данных

Для создания базы данных в утилите SSMS выполним скрипт со следующей командой, как показано на рисунке 9.

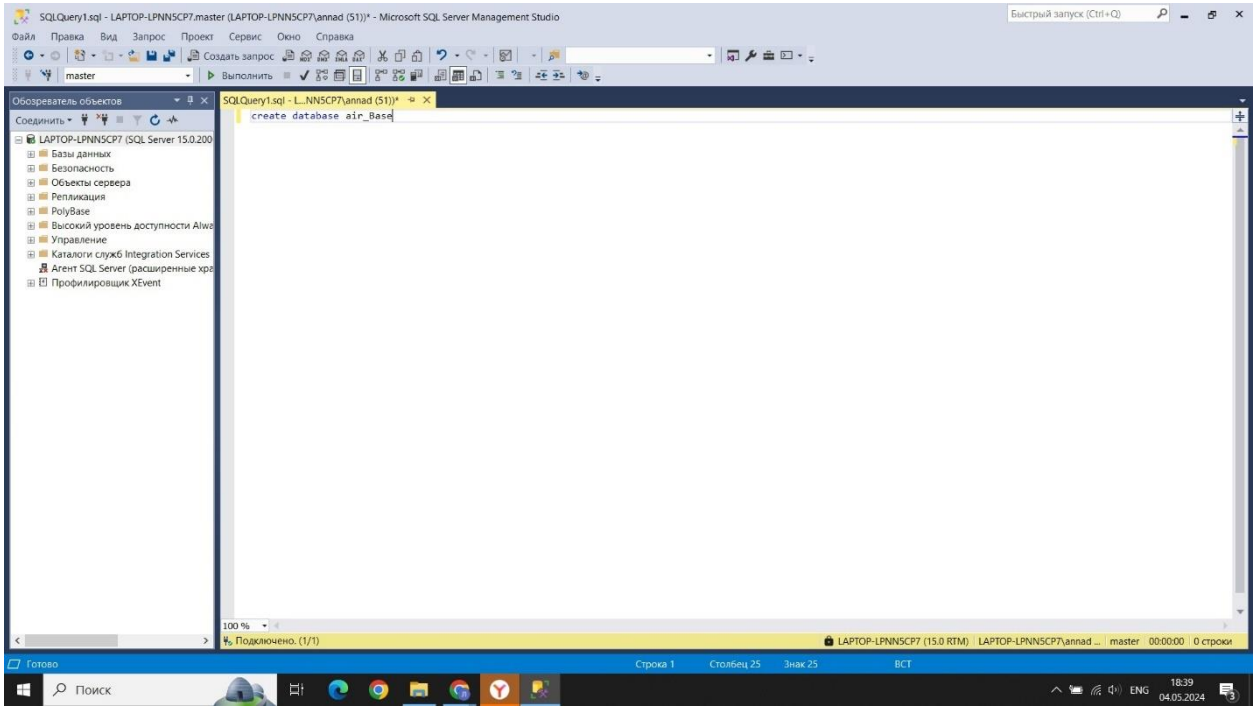

Рисунок 9 – Создание базы данных

3.2 Создание схемы ER-модели данных

Для создания схемы ER-модели данных в утилите SSMS выполним скрипт со следующими командами, как показано на рисунке 10.

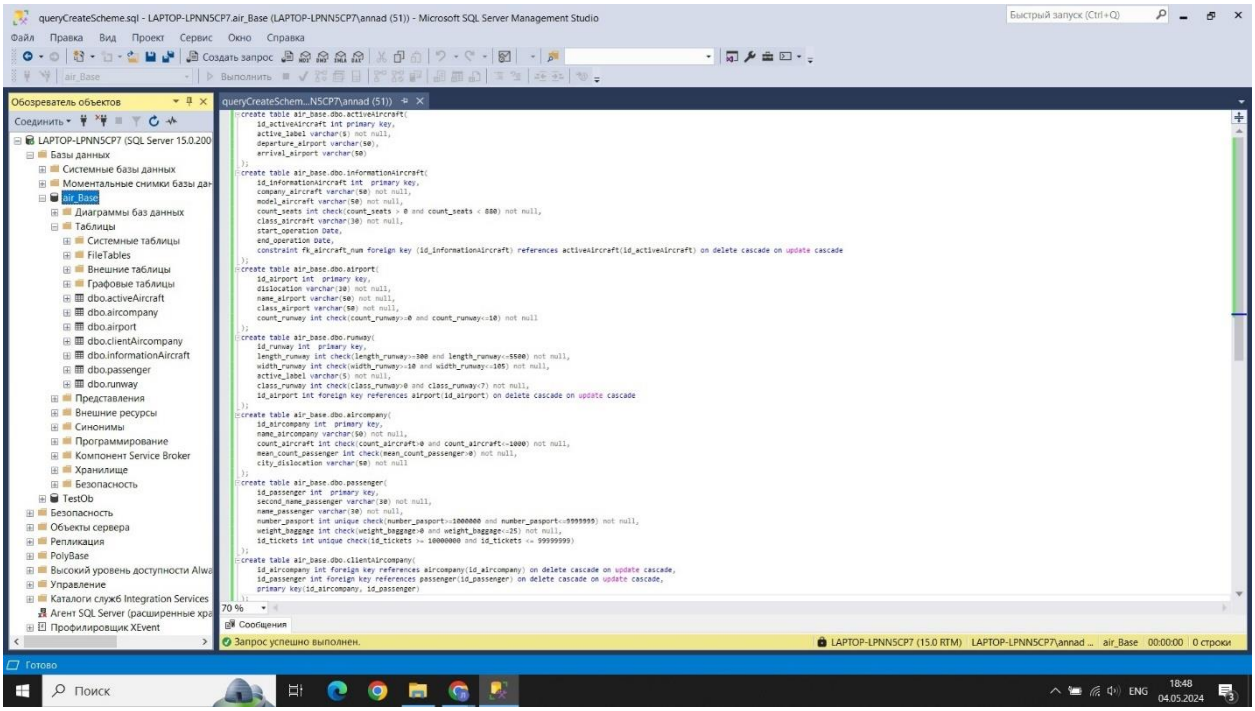

Рисунок 10 – Создание схемы ER-модели данных

4.1 Установка индексов

В качестве уникального индекса для каждой сущности был выбран его уникальный идентификационный номер – id.

5. Разработать типовые запросы к СУБД на языке SQL. Получение списков данных. Агрегация. Поиск.

5.1 Типовые запросы к базе данных

К типовым запросам относят следующие запросы:

- insert

Данный тип запроса производит добавление данных в таблицу посредством добавления или вставки новой строки в конец таблицы. Например, запрос вставки новых данных в таблицу activeAircraft insert into activeAircraft values(@id, 'True', ‘Москва’, 'Самара'). Однако в своей работе я использовала модифицированный запрос insert: bulk insert. Данный запрос производит массовую вставку данных из файла, например, из текстового файла, формата txt. Однако в этом случае данные должны быть форматированы, например, разделителем слов является пробел, а разделителем строк является символ переноса строки, и файл указанного формата сохранен с использованием кодировки utf-16.

Выполнение данного запроса представлено на рисунке 11.

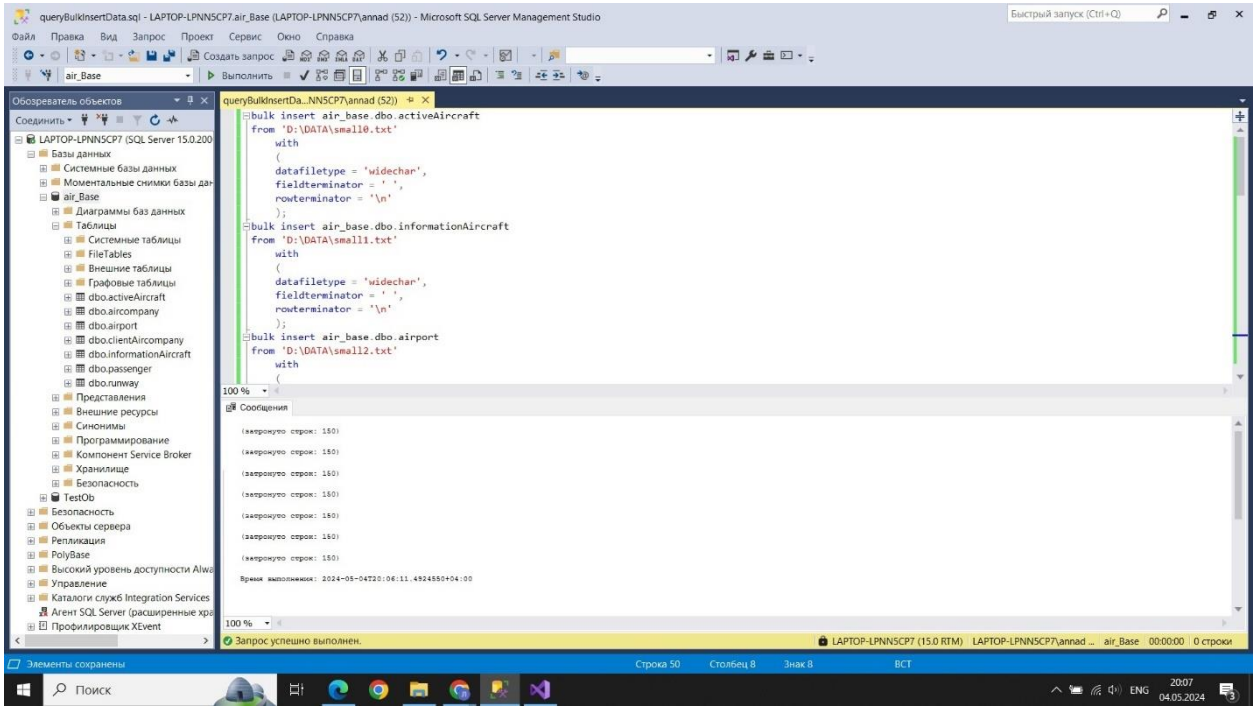

Рисунок 11 – Выполнение запроса bulk insert.

- update

Данный тип запроса производит обновление записей в таблице, посредством изменения значений в троке или столбце таблицы. Например, запрос обновления данных в таблице airport

update airport set airport.class_airport = @class_airport where airport.dislocation like @dislocation and airport.name_airport like @name

Выполнение данного запроса представлено на рисунке 12.

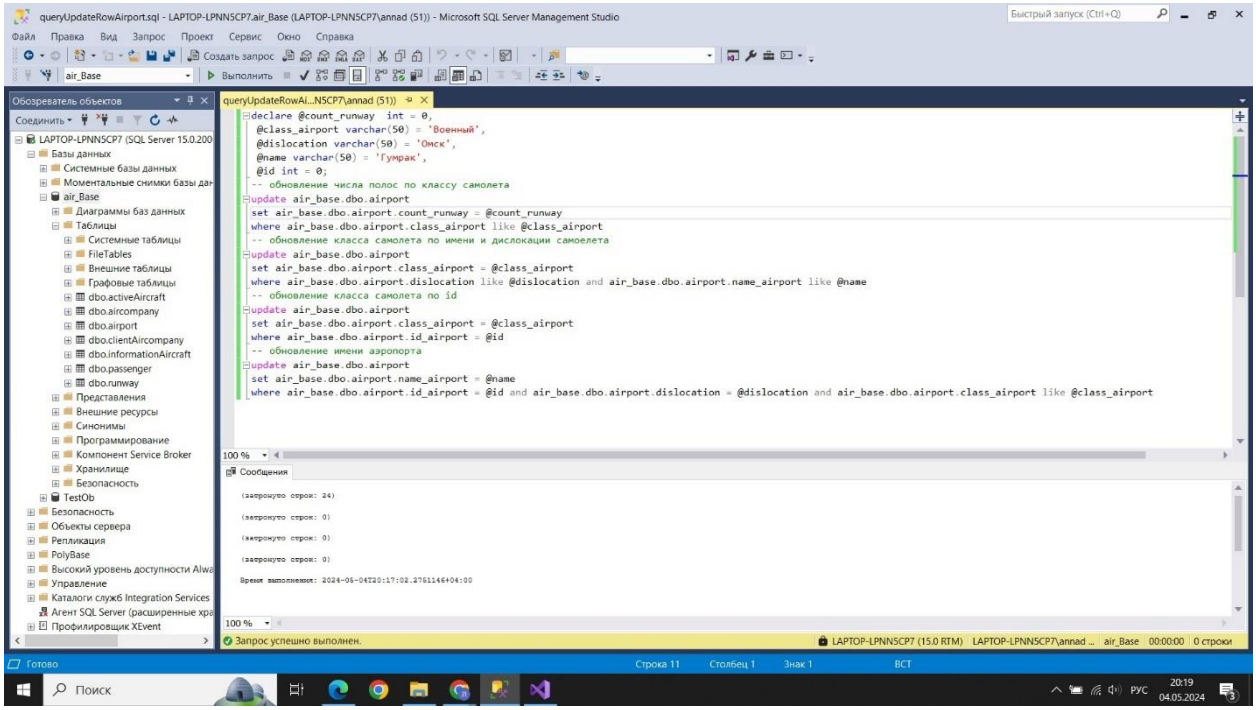

Рисунок 12 – Выполнение типового запроса update.

- delete

Данный тип запроса производит удаление данных из таблицы. Удаление может производиться несколькими путями: удаление конкретной строки, нескольких строк, объединенных определенным условием, так и полное удаление всех данных из таблицы. Например, delete from informationAircraft where informationAircraft.id_informationAircraft = @id;

Выполнение данного запроса представлено на рисунке 13.

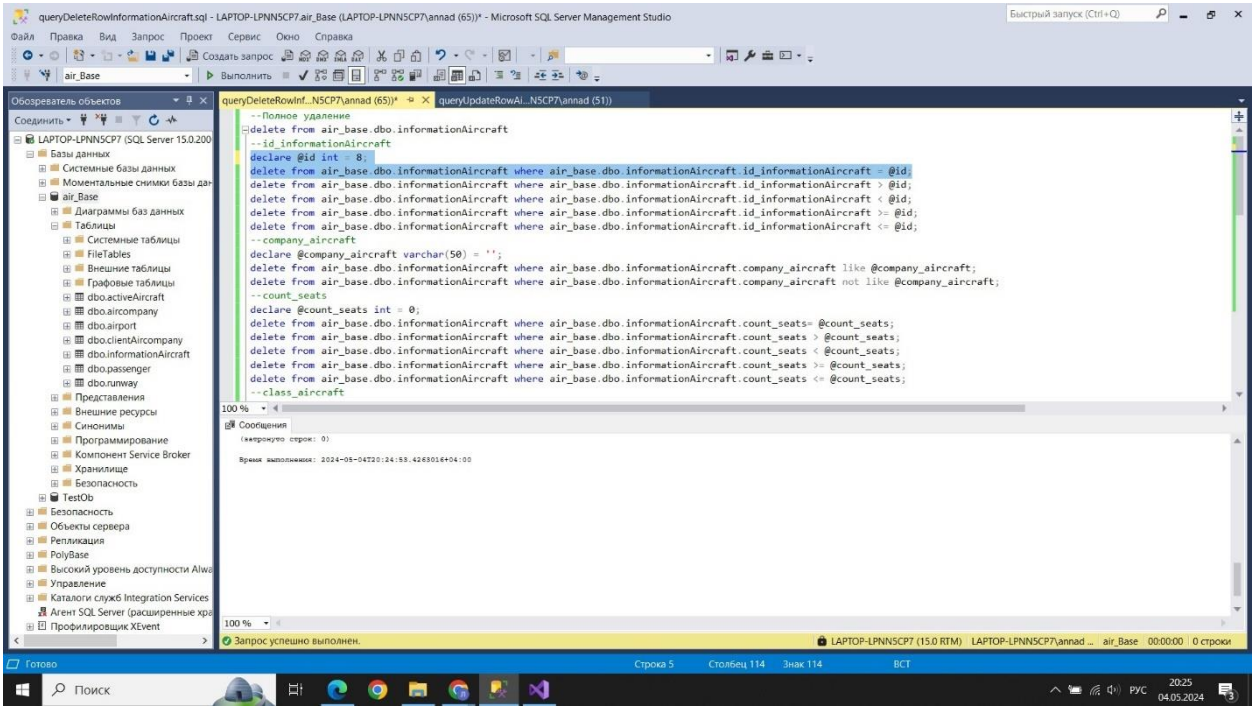

Рисунок 13 – Выполнение типового запроса delete.

5.2 Запросы получения списков данных и поиска

Данные запросы имеют единый тип – запросы поиска, только различие в накладываемых ограничениях. Для получения списка данных происходит наложение менее строгих ограничений, потому что необходимо получить несколько строк данных. Для поиска конкретного элемента (строки), необходимо производить наложение более строгих ограничений. Приведем примеры подобных запросов.

Например, для получения списка данных мы можем использовать следующий запрос, который осуществляет поиск пассажиров, у которых вес багажа соответствует указанному:

select * from passenger where passenger.weight_baggage = @weight;

в результате выполнения данного запроса, явно будет выведена не одна строка, а некоторый список.

Выполнение данного запроса представлено на рисунке 14.

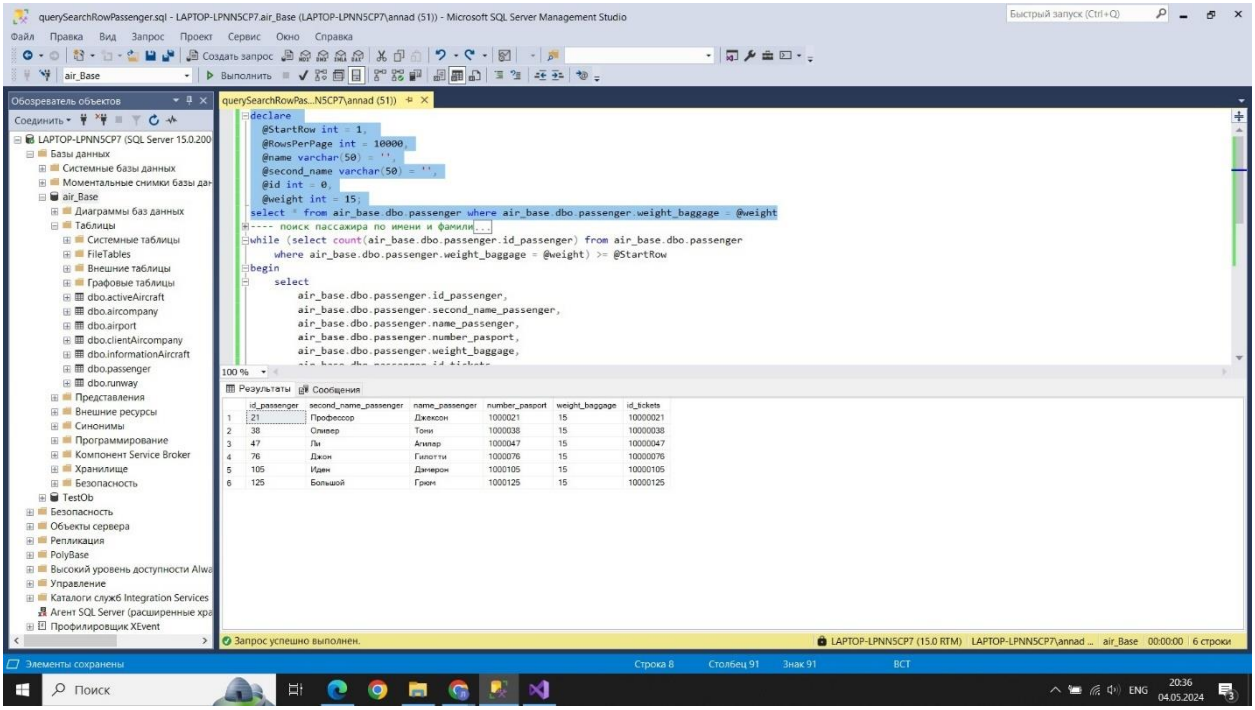

Рисунок 14 – Выполнение запроса получения списка данных.

В это же время для более конкретного и детального поиска элемента таблицы (строки), необходимо накладывать более серьезные ограничения, например, производить поиск по уникальному индексу, который даст однозначный ответ:

select * from airport where airport. id_airport = @id_airport.

Выполнение данного запроса представлено на рисунке 15.

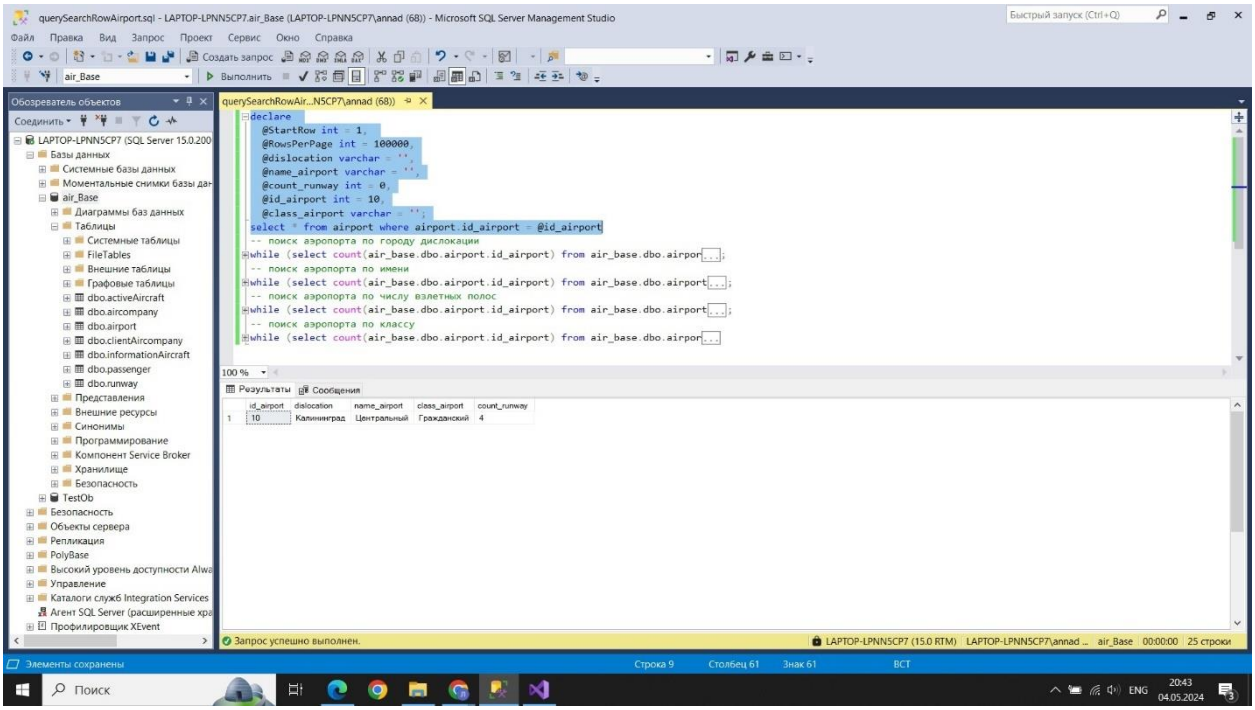

Рисунок 15 – Выполнение запроса поиска.

5.3 Запрос агрегации

Данный тип запроса позволяет производить агрегирование данных, посредством поисков минимального, среднего, максимального числовых значений, а также суммирования числовых данных, в рамках заданного условия.

```
select COUNT(distinct(informationAircraft.model_aircraft))
from informationAircraft
where informationAircraft.company_aircraft like @company
```

Выполнение данного запроса представлено на рисунке 16.

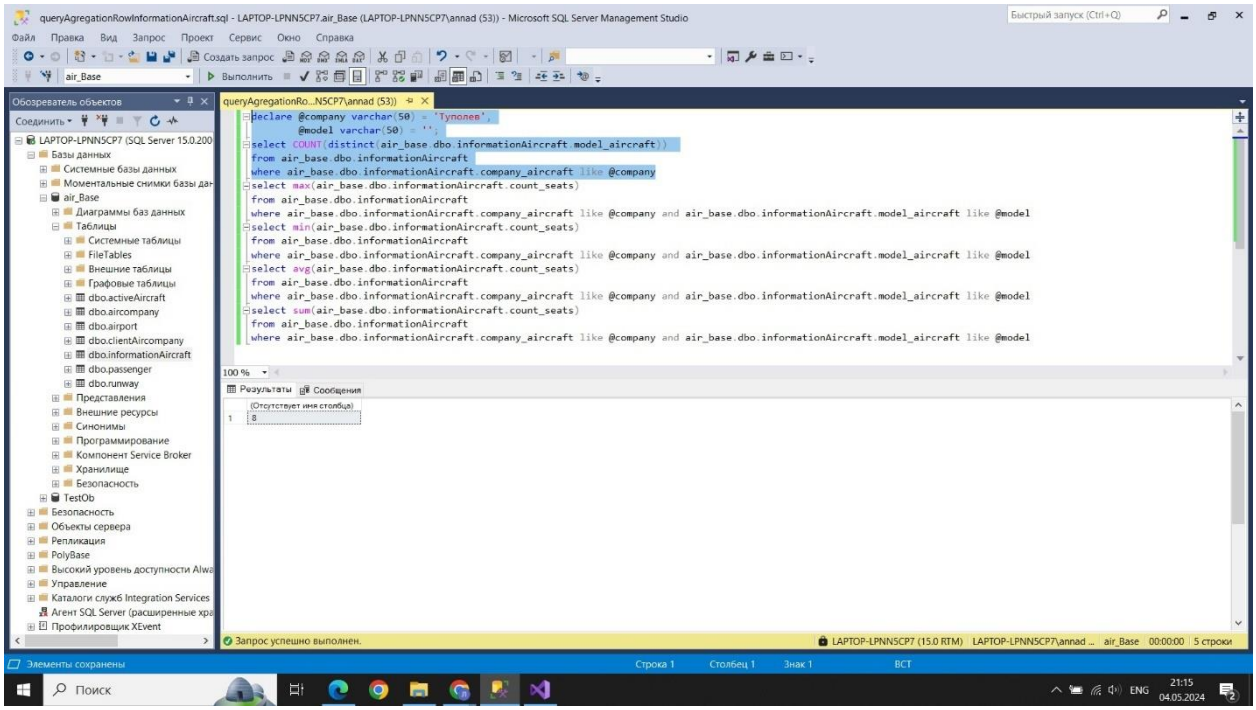

Рисунок 16 – Выполнение запроса поиска.

6-7. Разработайте хранимые процедуры на языке PL/pgSQL для генерации случайных данных для базы данных. Сгенерируйте тестовые данные при помощи разработанных процедур.

Поскольку в задание сказано, что выполнение данного задания допустимо производить без использования PL/pgSQL, то мы так и поступим. Для генерации большого числа данных было разработано консольное приложение на высокоуровневом языке программирования C#.

Для разработки указанного приложения использовалась среда разработки Visual Studio 2019 с языком C#.

В результате разработки приложения было описано несколько классов:

- классы, описывающие сущности ER-модели;
- классы, содержащие необходимые данные для случайного генерирования объектов;
- класс, содержащий точку входа в программу.

6.1 Характеристика классов, описывающих сущности ER-модели

Каждый класс, описывающий сущность, имеет в своей структуре следующие элементы:

- поля;
- конструктор и методы доступа;
- метод случайной генерации объекта;
- метод вывода объекта, согласно заданному формату.

Поля классов, совпадают со столбцами сущностей, описанными в схеме базы данных, и дают некоторые характеристики объекту, которые они описывают, как на рисунке 17.

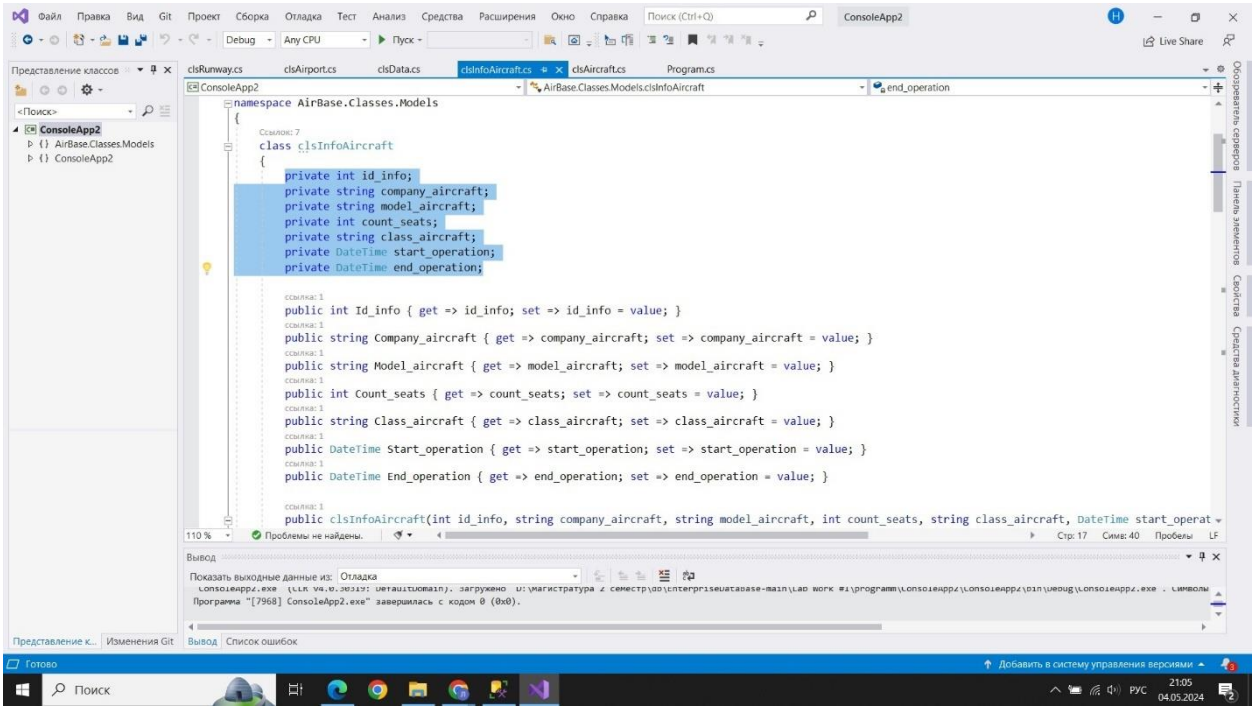

Рисунок 17 – Поля класса clsInfoAircraft

Конструктор, как и методы доступа необходимы, для работы с объектами по канонам объектно-ориентированного программирования, как это показано на рисунке 18.

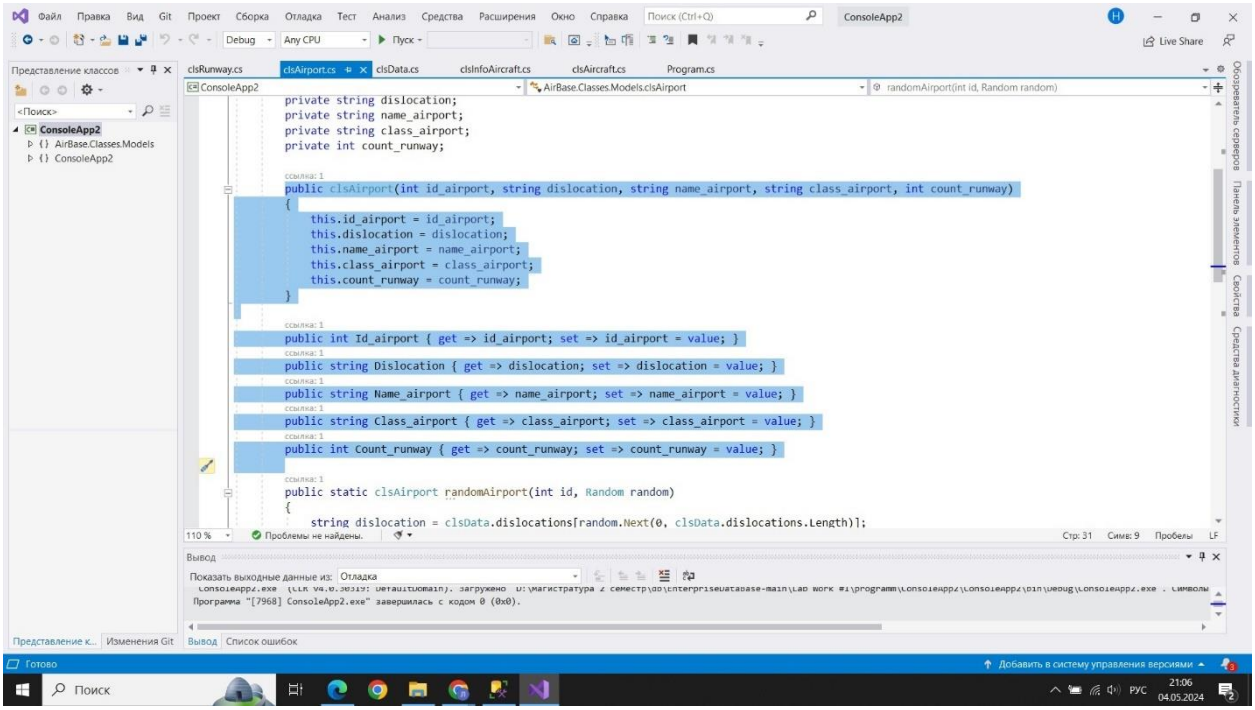

Рисунок 18 – Конструктор и методы доступа класса clsAirport

Метод случайной генерации объекта производит генерацию объекта при каждом ее вызове, для того чтобы не производить генерацию объектов в

ручном режиме. Данные для генерации случайного объекта берутся из специально описанного класса, который содержит в себе наборы данных, из которых и производится выборка данных для генерации.

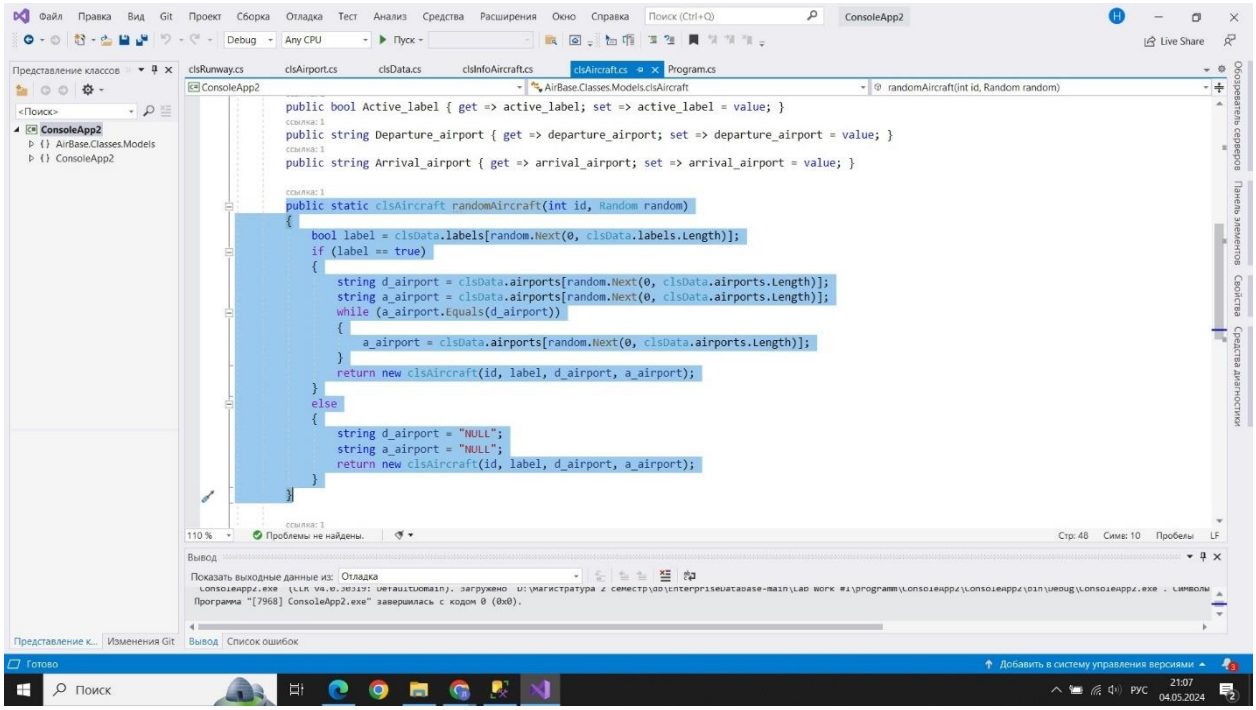

Рисунок 19 – Метод случайной генерации объекта clsAircraft

Метод вывода объекта, согласно заданного формата, производит сохранение объекта в формате строки заданного формата, который необходим для последующей массовой вставки в таблицу базы данных.

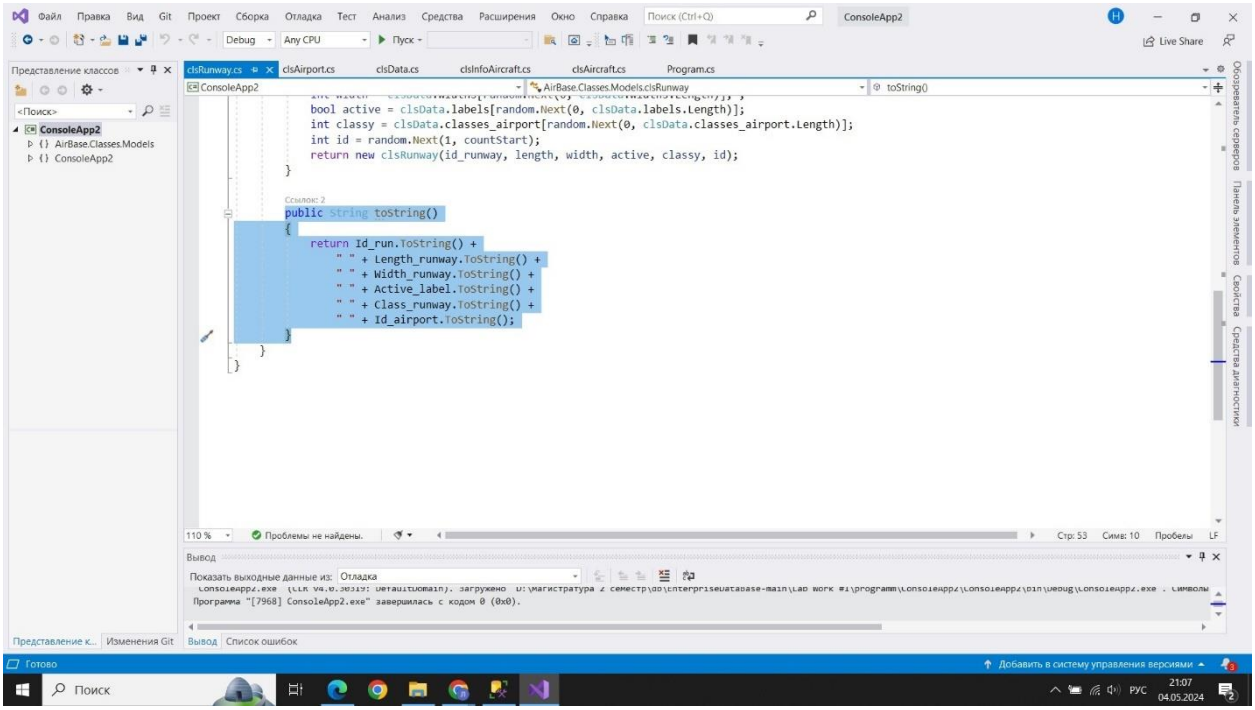

Рисунок 20 – Метод вывода объекта, согласно заданного формата класса clsRunway

6.2 Характеристика класса, содержащего данные для случайной генерации объектов

Данный класс является статическим и содержит в себе массивы различных типов данных, которые используются методом случайной генерации объекта. Часть массивов данного класс показана на рисунке 21.

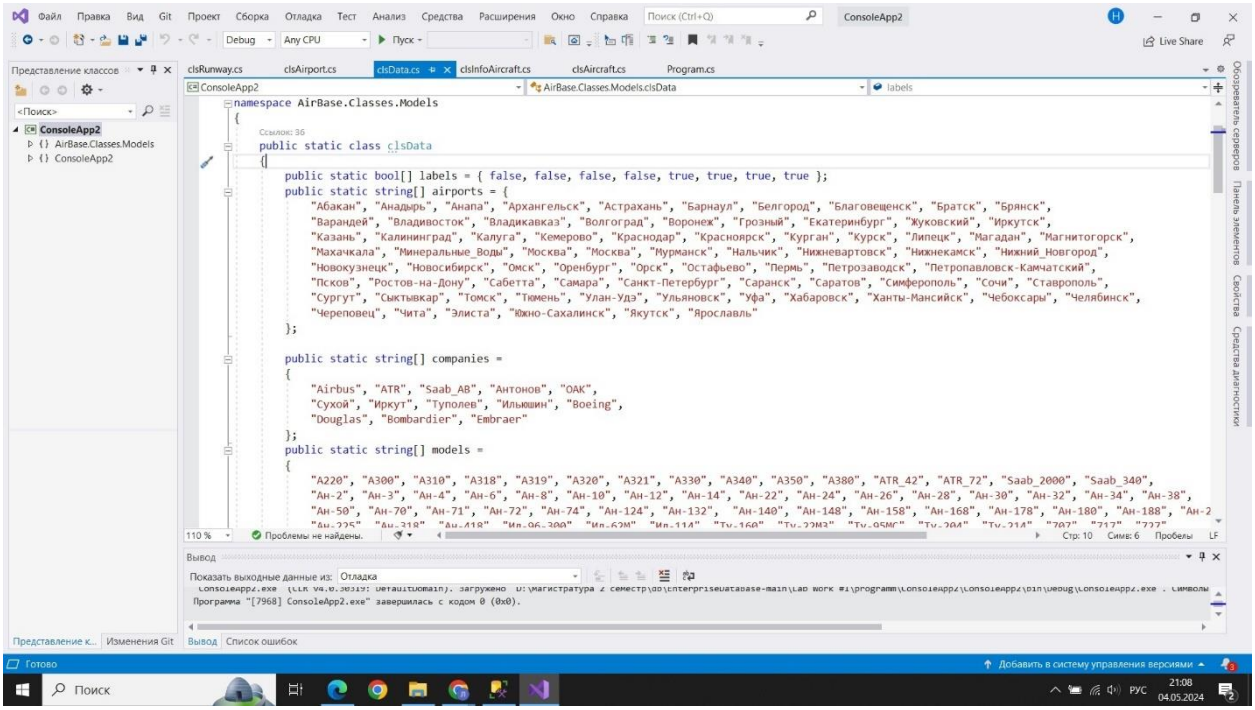

Рисунок 21 – Массивы данных, необходимые для генерации объектов

6.3 Характеристика класса, содержащего точку входа в программу

Данный класс содержит в себе точку входа в программу, а также метод, который и выполняет всю работу приложения, по генерации объектов.

7.1 Генерация тестовых данных.

Генерация тестовых данных производится в результате работы программы. Во время запуска Пользователю предоставляется два варианта работы программы:

- режим малой генерации данных (по 150 элементов каждого класса)
- режим большой генерации данных (по 1000000 элементов каждого класса)

Независимо от режима работы приложения, программа выполняет циклическую последовательность действий:

- Генерация объекта;
- Сохранение его в строковом формате в список.
- По завершении работы цикла, полученный список объектов сохраняется в файл.

Пример работы программы представлен на рисунке 22.

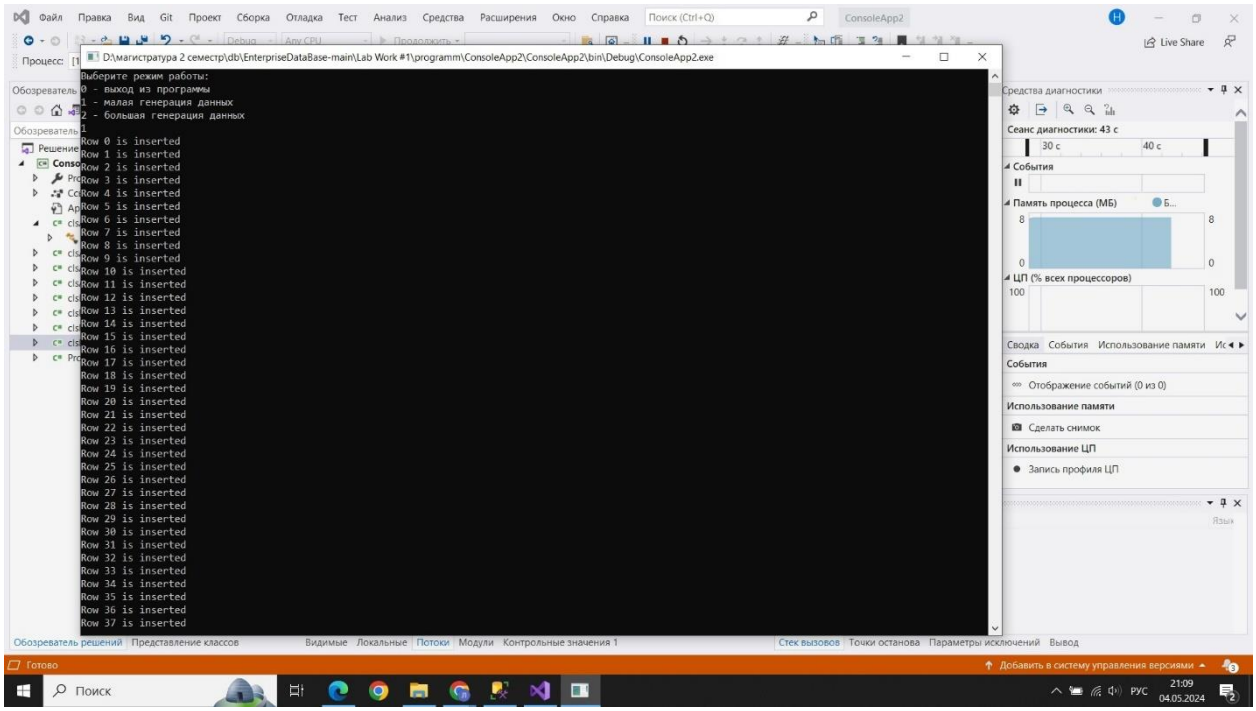

Рисунок 22 – Генерация тестовых данных

8–9. Протестируйте работу запросов на больших объёмах данных (Порядка 1 миллиона записей в основных таблицах). Измените конфигурацию сервера PostgreSQL для достижения лучшей производительности на самых медленных запросах. Оптимизируйте схему БД и запросы для достижения лучшей производительности.

Для выполнения данного задания нам потребуется база данных с большим объемом таблиц. При помощи, описанной выше программы произведем генерацию данных, необходимого объема. После чего подготовленные данные перенесем в таблицы баз данных, с использованием команды массовой вставки, поскольку данные сохранены в необходимом формате.

Совершенно очевидным становится тот факт, что наиболее медленными запросами будут запросы выбора. Поскольку именно эти запросы производят наибольшую обработку строк таблиц базы данных, соответственно и времени потребуется больше. Именно с этим типом запросов мы и будем работать.

Для контроля скорости выполнения запросов воспользуемся штатными средствами SQL Server Management Studio (SSMS):

- План выполнения
- Статистика активных запросов
- Статистика клиента

План выполнения запроса, как и статистика активных запросов помогает нам увидеть слабые или уязвимые места в наших запросах, которые отнимают не мало времени, и очень сильно тормозят процесс получения информации из базы данных, поскольку скорость получения информации в современном мире имеет огромное значение для человека. Работа данных средствах показана на рисунках 23 – 25.

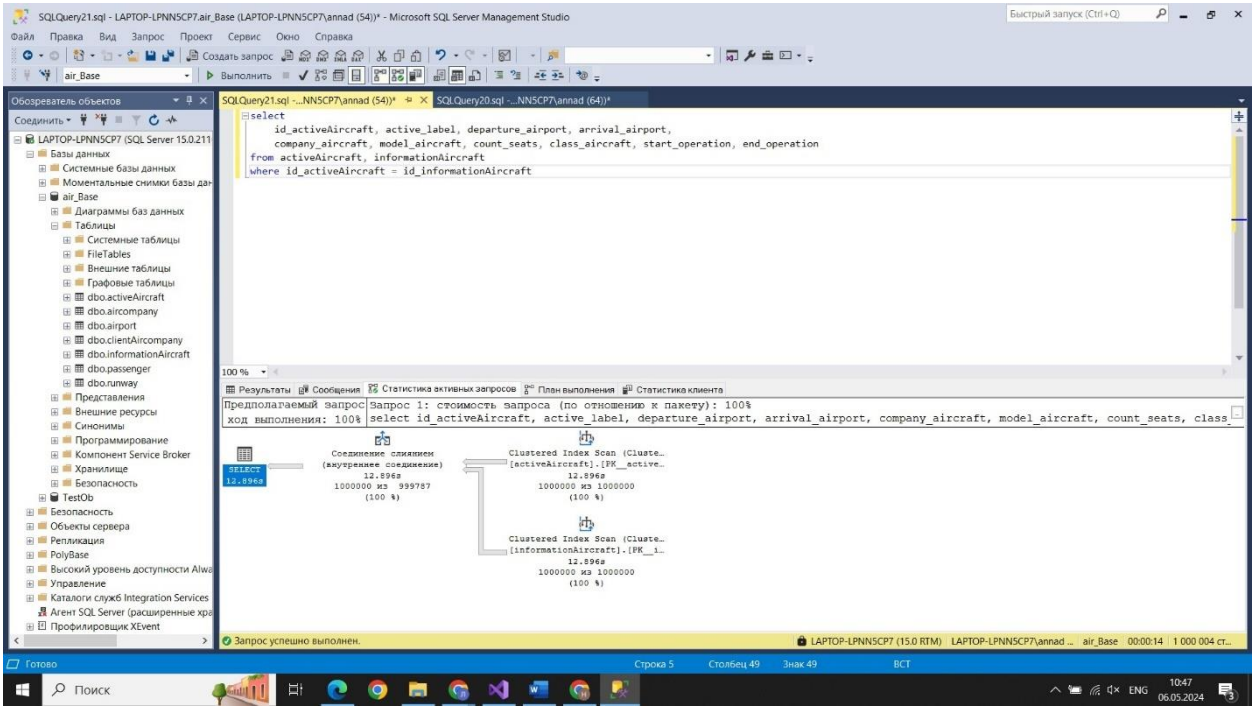

Рисунок 23 – Статистика активных запросов

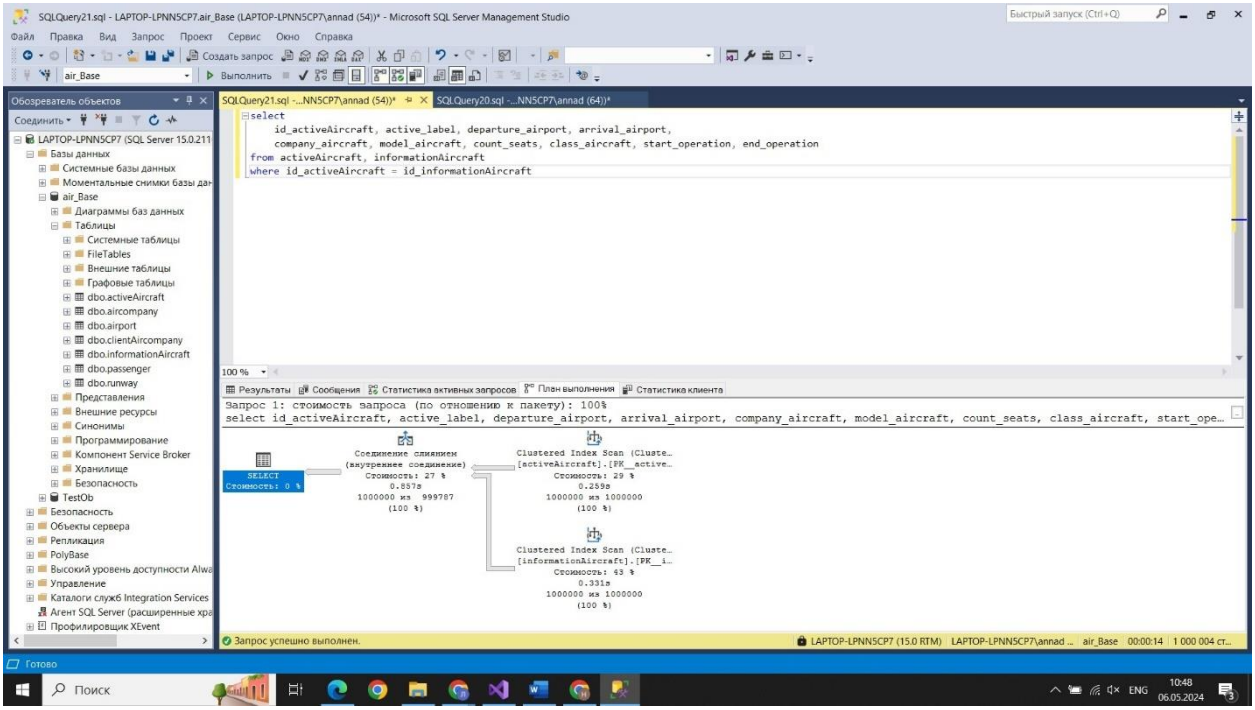

Рисунок 24 – План выполнения запроса

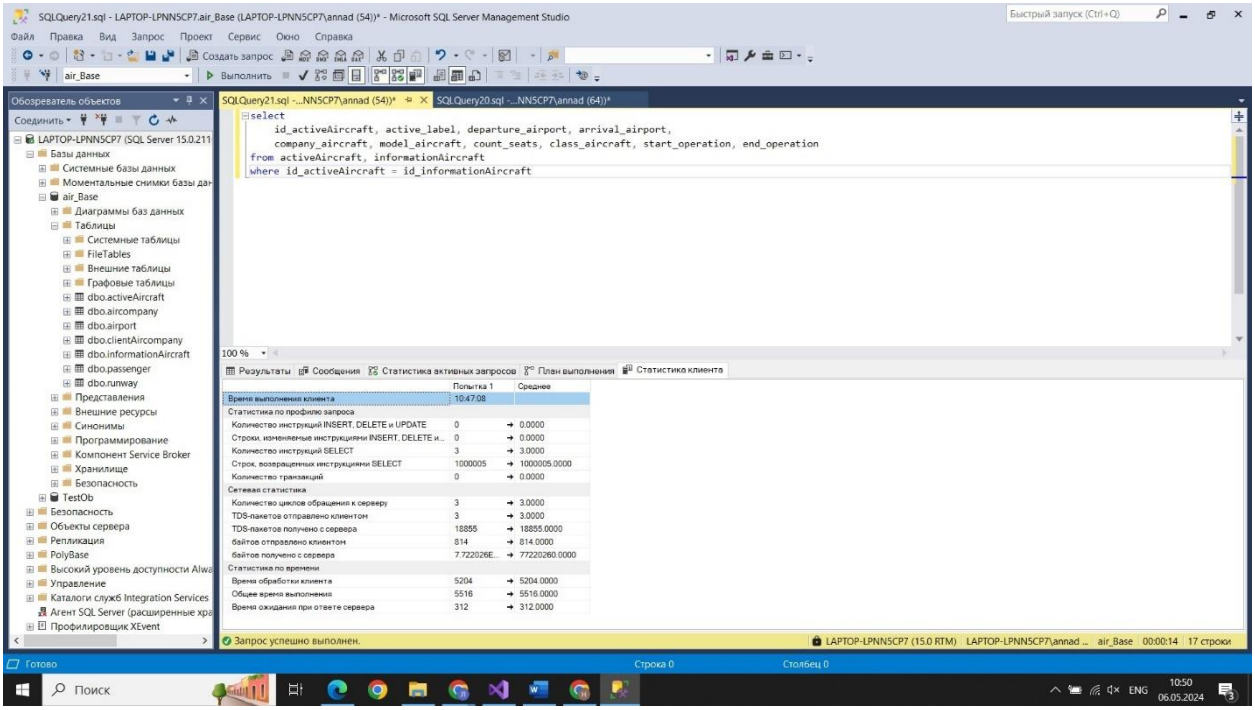

Рисунок 25 – Статистика клиента

Из статистики клиента не трудно заметить, что на выполнение данного запроса требуется около 5,5 секунд. Очевидным становится тот факт, что это долго, однако стоит учитывать объем данных. Как можно сократить время выполнения данного запроса? Это сделать практически невозможно. Однако в выборках такого объема данных нет необходимости, поскольку они становятся не читаемыми и не воспринимаемыми. Отсюда можно сделать вывод, что первой возможностью повысить скорость выполнения запросов – правильно формировать запросы, использовать условия, знать, что конкретно необходимо найти.

Рассмотрим тот же запрос вывода списка данных, однако он будет ограничен по одному условию:

```
select *
from activeAircraft, informationAircraft
where id_activeAircraft = id_informationAircraft and class_aircraft = 'Дальнемагистральный'
```

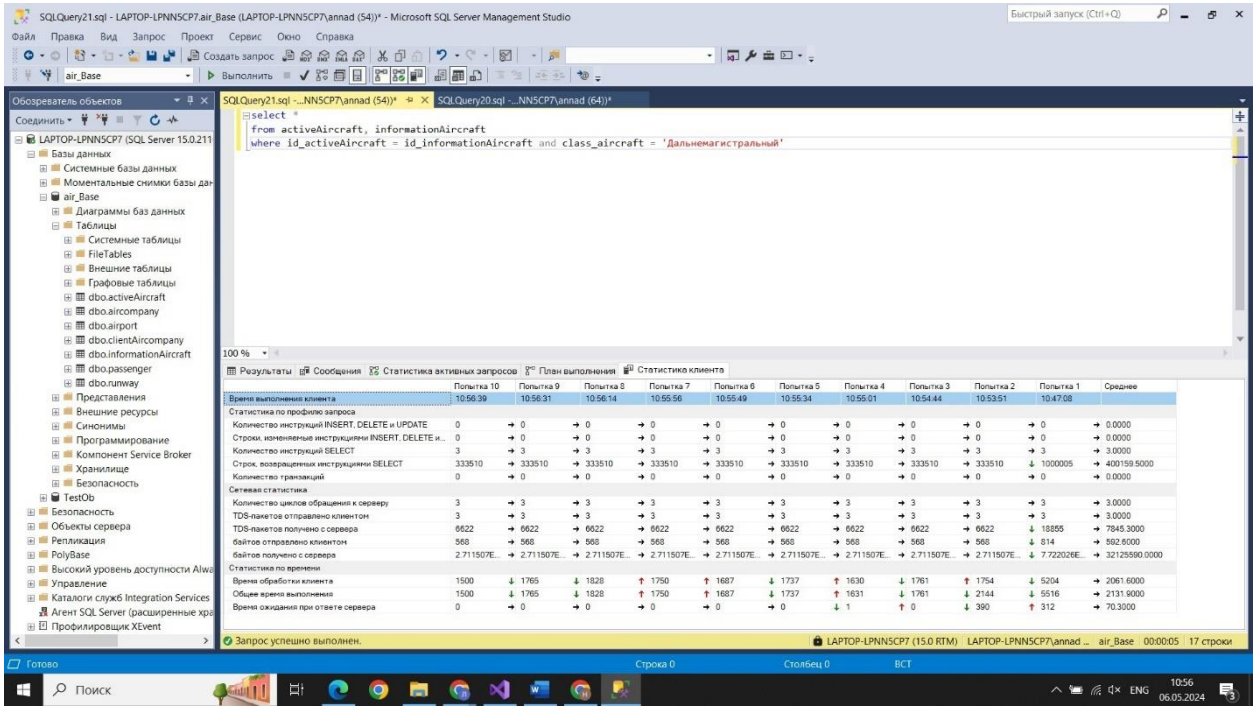

Рисунок 26 – Временная статистика выполнения запроса

Как можно заметить из статистики клиента, на выполнение запроса требуется в среднем 2,1 секунды. Попробуем улучшить данный временной показатель. Модифицируем запрос с применением join.

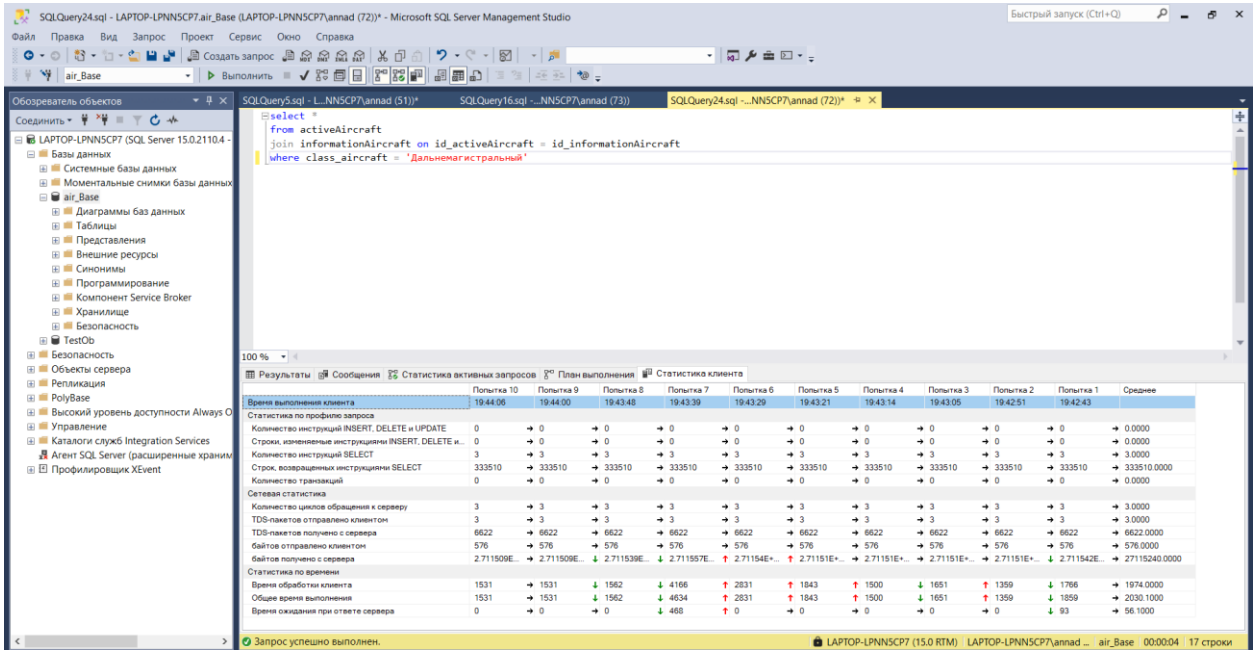

Рисунок 27 – Временная статистика выполнения запроса

Нетрудно увидеть, что модификация запроса не принесла особого успеха и на выполнение запроса все так же требуется в среднем 2 секунды.

Попробуем улучшить данный временной показатель, посредством изменения структуры и данных таблицы: произведем замену строковых

значений в столбце class_aircraft таблицы informationAircraft на числовые. Например, классу ближнемагистральных самолётов, будет соответствовать класс самолетов – 1, для класса среднемагистральных соответствует класс 2, и т.д. С учетом данной корректировки попробуем выполнить вышеуказанный запрос.

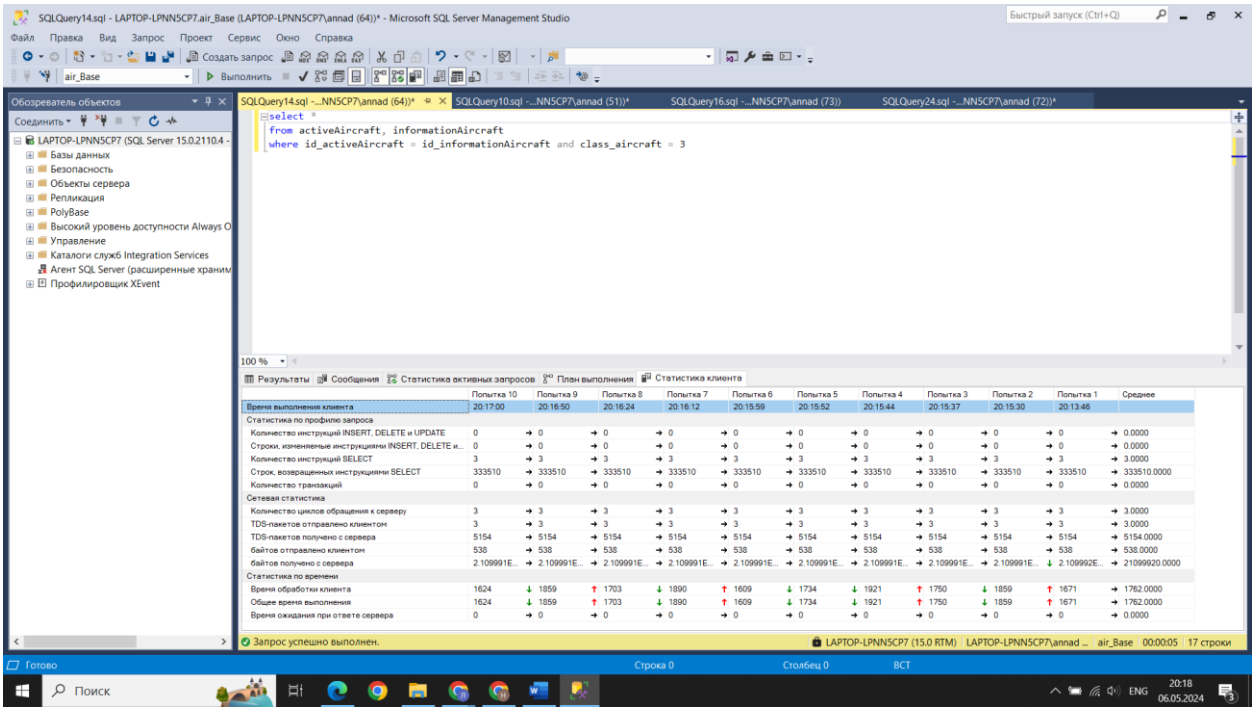

Рисунок 28 – Временная статистика выполнения запроса

После осуществления реорганизации таблицы базы данных, выполнение запроса заняло меньшее время. Попробуем вдобавок к реорганизации базы данных изменить запрос, и повторим его выполнение.

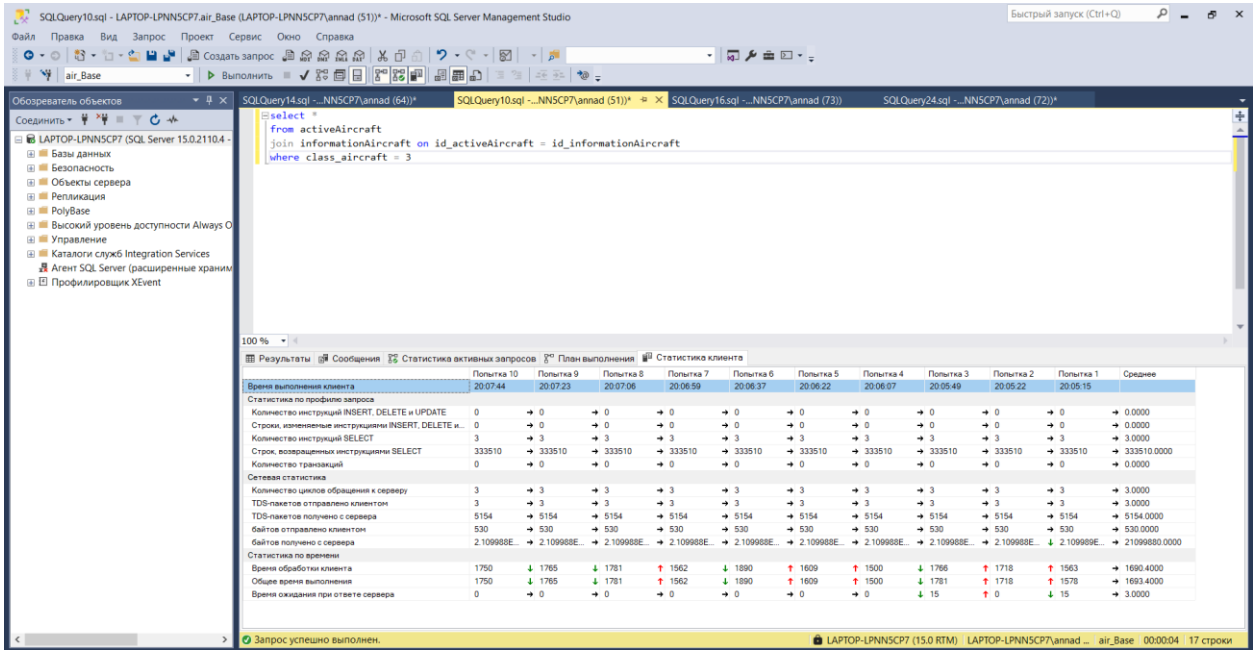

Рисунок 29 – Временная статистика выполнения запроса

После выполнения описанных манипуляций можно заметить сокращение времени выполнения запроса по сравнению с предыдущей модификацией. Давайте сравним время выполнения данного запроса до начала всех модификаций (2130 мс) с временем выполнения запроса после всех модификаций (1690 мс). Мы видим уменьшение времени выполнения на 440 мс, что составляет около 18-20%. Сокращение на 20% — это неплохое улучшение, которое может сказываться на общей эффективности работы с базой данных.

После проведения оптимизации удалось достичь заметного ускорения работы запросов, несмотря на несколько ограничивающих факторов:

- База данных содержит значительное количество данных, порядка 1 миллиона строк, что вносит определенные сложности в процесс обработки запросов.
- Ограниченные возможности в изменении конфигурации СУБД (MSSQL), так как это проприетарный продукт, могут затруднить максимальную оптимизацию системы.
- Ноутбук оснащен процессором AMD Ryzen 5 3500U с тактовой частотой 2.10 ГГц, но который, показывает неплохие результаты в оптимизации работы запросов.

ЗАКЛЮЧЕНИЕ

В результате выполнения лабораторной работы были получены и отточены навыки по работе с СУБД, рассмотрены особенности корпоративных систем, а так особенности работы с базами данных больших объемов, порядка 1 миллиона строк.

Все скрипты, для работы с СУБД MS SQL 2019 Developer, а также код программы для генерации данных расположены на сайте GitHub по ссылке: https://github.com/WonMin13/EnterpriseDataBase/tree/main/Lab%20Work%20%231