## Федеральное государственное автономное образовательное учреждение высшего образования

## НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

## «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

**Высшая школа бизнеса**

**КОНТРОЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ**

## Проектирование и разработка базы данных для продажи авиабилетов

по направлению подготовки 38.03.05 образовательная программа «Бизнес-информатика»

### Проект выполнили:

Патаев Арслан Зольванович

Озтемел Теоман Сердарович

Мухлаев Алан Арслангович

Горяев Даян Мергенович

Москва 2024

**Оглавление**

[**Введение 3**](#_gjdgxs)

[**Общие сведения о предметной области 3**](#_30j0zll)

[**Цель создания базы данных 3**](#_1fob9te)

[**Возможные пользователи базы данных 3**](#_3znysh7)

[**Возможные сценарии взаимодействия пользователей и БД 4**](#_2et92p0)

[**Бизнес-процесс 4**](#_pitng78qhoil)

[**Список основных сущностей, атрибуты с указанием PK 6**](#_tyjcwt)

[**Типы и описание связей между сущностями 13**](#_3dy6vkm)

[**Алгоритмы обработки данных, используемые в сценариях 18**](#_1t3h5sf)

[**Бизнес-функция 18**](#_4d34og8)

[**Планируемый перечень отчетных форм 18**](#_2s8eyo1)

[**Возможная архитектура программного продукта на основе нашей базы данных 18**](#_17dp8vu)

[**Инфологическая модель 19**](#_3rdcrjn)

[**Даталогическая модель 19**](#_26in1rg)

[**Типы данных в физической модели 20**](#_lnxbz9)

[**Соответствие таблиц 3НФ 21**](#_35nkun2)

[**Генерация и заполнение БД данными 22**](#_1ksv4uv)

[**Настройка индексов 28**](#_44sinio)

[**Разработка запросов 30**](#_2jxsxqh)

[**Разработка триггера 46**](#_z337ya)

[**Разработка функций 46**](#_3j2qqm3)

[**Разработка представлений 48**](#_1y810tw)

[**Разработка процедур 49**](#_4i7ojhp)

[**Информационная панель в Excel 54**](#_2xcytpi)

[**Отчеты, визуализация 57**](#_1ci93xb)

[**Описание роли каждого участника проекта 60**](#_3whwml4)

# Введение

В последнее время в области теории баз данных было проведено несколько крупных исследований. Полученные результаты смело можно считать наиболее важным достижением информатики за последние 30 лет. Современные базы данных стали основой информационных систем и значительно изменили алгоритм работы большинства организаций. Развитие технологии баз данных способствовало созданию удобных и быстрых систем, что сделало их популярными среди широкого круга пользователей. В наше время большинство компаний сильно заинтересованы в создании информационных баз данных, так как они не только структурируют необходимую информацию, но и обеспечивают её быструю обработку. Наша команда задумала создать базу данных для удобной группировки и хранения информации о продаже билетов авиакомпаниями.

# Общие сведения о предметной области

Компания N, наш заказчик, обратилась к нам с задачей разработки базы данных для удобной группировки и хранения информации о продаже билетов авиакомпаниями. До недавних изменений в законодательстве и экономической ситуации в мире ежедневно самолеты совершали около 120 тысяч полетов и перевозили примерно 12 миллионов пассажиров. В условиях изменившихся правил и требований к перелетам людям по-прежнему необходимо совершать различные перелеты между городами и странами по разным причинам, несмотря на сокращение количества рейсов. Это делает создание базы данных для компании N актуальным, так как она сможет оптимизировать процессы многих авиакомпаний. Предметная область включает в себя не только учёт продажи авиабилетов различными авиакомпаниями, но и хранение, анализ этой информации для развития компании в этой области в целом. Для этого необходимо хранить и обрабатывать информацию об аэропортах, рейсах, билетах, команде и персонале самолета, операциях и пассажирах. Из-за этой потребности компания N инициировала создание базы данных, которая быстро предоставит доступ к необходимой информации и поможет в принятии управленческих решений.

# Цель создания базы данных

Главной целью создания базы данных для компании N является хранение информации о расписании авиаперелетов, о компаниях, которые их осуществляют, о самолетах этих компаний, о команде, работающей на каждом рейсе, о пассажирах, совершающих перелеты и покупающих билеты, и, конечно, о совершенных покупателями операциях. Сбор этой информации с последующим хранением и анализом не только оптимизирует процесс продажи билетов, но и позволяет компании N более точно рассчитывать показатели эффективности, прогнозировать спрос, улучшать логистические и операционные процессы, а также значительно уменьшать количество возможных ошибок как в продаже авиабилетов, так и в процессах самой компании.

# Возможные пользователи базы данных

Возможными пользователями нашей базы данных могут стать любые компании, организации, их сайты или, например, сайты турагентств, ориентированные на продажу авиабилетов. В этот список входят и сами авиакомпании, нуждающиеся в анализе, отслеживании и хранении информации о проданных билетах. Благодаря нашей базе данных у них появится возможность иметь более структурированное и детальное представление о всех операциях, связанных с продажей авиабилетов, а также использовать данные для совершенствования внутренних процессов и расчета различных показателей эффективности.

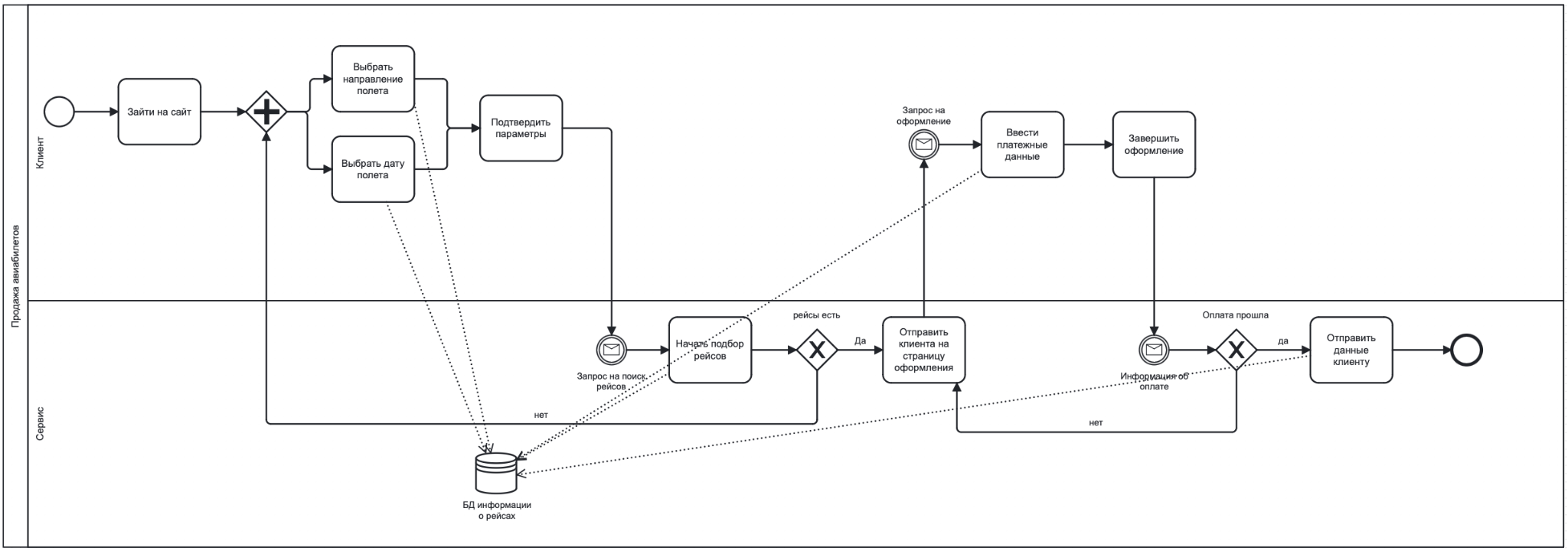
# Возможные сценарии взаимодействия пользователей и БД

Одним из возможных сценариев взаимодействия может быть использование пользователем любого сайта по поиску авиабилетов. В данном процессе он выбирает конкретное направление полета, после чего ему предлагается выбрать дату и время вылета и прилета. Пользователь выбирает необходимый вариант, определяется с местом на борту рейса, вводит свои персональные данные и производит покупку. После этого происходит процесс обращения к базе данных и её работы. В неё заносится вся необходимая информация о пользователе, и происходит его добавление. Параллельно с этим выполняется процесс, который зависит от самого рейса, а именно, определяется самолет, состав персонала, экипажа и т. д. Дополнительно база данных может использоваться для анализа текущих и прошлых данных, что помогает компании N в расчете показателей эффективности, оптимизации маршрутов, управления запасами и планировании новых рейсов, а также в совершенствовании маркетинговых и логистических стратегий.

# Бизнес-процесс

Один из примеров решения бизнес-задач с помощью внедрения базы данных в процесс продажи авиабилетов. Этот пример демонстрирует, как база данных может значительно улучшить управление бизнес-процессами и принятие управленческих решений в авиакомпании.

Процесс 1 – продажа авиабилетов BPMN :



Процесс начинается с того, что клиент заходит на сайт авиакомпании или агрегатора авиабилетов. На первой стадии клиент выбирает направление полета и дату поездки. В этот момент сайт отправляет запрос в базу данных для получения информации о доступных рейсах и свободных местах. База данных возвращает соответствующую информацию, и клиенту отображаются доступные рейсы, включая время вылета, прилета, стоимость билетов и доступные места. После этого клиент выбирает конкретный рейс и место в самолете. Выбранный рейс и место резервируются путем отправки запроса в базу данных. Затем клиент вводит свои персональные данные, которые также сохраняются в базе данных. Следующий этап — это оплата билета. Клиент переходит к оплате и, после успешного завершения транзакции, информация о ней сохраняется в базе данных. После этого база данных обновляет информацию о занятых местах, и клиент получает подтверждение бронирования, а также электронный билет.

Теперь рассмотрим, как база данных помогает в принятии управленческих решений. Во-первых, база данных собирает информацию о выбранных направлениях, датах и времени полетов. Это позволяет анализировать спрос на конкретные рейсы и оптимизировать расписание, что в свою очередь способствует более эффективному использованию ресурсов. Кроме того, с помощью данных о занятых местах можно управлять ценовой политикой, предлагая скидки на менее популярные рейсы или места. Это помогает повысить загрузку самолетов и, соответственно, увеличить доходы компании.

Информация о рейсах и пассажирах, собранная в базе данных, также помогает эффективно распределять ресурсы, такие как персонал на рейсах, обслуживание самолетов и наземные службы. Это позволяет улучшить операционную эффективность и качество обслуживания клиентов. Сохраненные данные о клиентах позволяют персонализировать предложения и улучшать качество обслуживания, что может повысить лояльность клиентов и привлечь новых. Анализ исторических данных дает возможность делать точные прогнозы и планировать будущие рейсы, что способствует стратегическому развитию компании. Наконец, сохраненные данные о транзакциях помогают контролировать финансовые потоки и принимать решения о финансовых стратегиях, что улучшает управление финансовыми ресурсами компании.

Внедрение базы данных не только автоматизирует процесс продажи билетов, но и предоставляет богатую информацию для принятия обоснованных управленческих решений. Это способствует улучшению операционной эффективности и конкурентоспособности компании, делая её более гибкой и готовой к изменениям на рынке.

# Список основных сущностей, атрибуты с указанием PK

Таблица 1 – информация об аэропорте (описание атрибутов сущности AIRPORT) :

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_airport | int | PK | ID аэропорта |
| name\_airport | varchar(50) |  | Название аэропорта |
| duty\_free | varchar(3) |  | Наличие duty free в аэропорту |

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_type\_airport | int | PK | ID типа аэропорта |
| type\_airport | varchar(40) |  | Тип аэропорта |

Таблица 2 – тип аэропорта (описание атрибутов сущности TYPE\_AIRPORT)

Таблица 3 – информация о стране (описание атрибутов сущности COUNTRY):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_country | int | PK | ID страны |
| name\_country | varchar(40) |  | Название страны |

Таблица 4 – информация о городе (описание атрибутов сущности TOWN):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_town | int | PK | ID города |
| name\_town | varchar(50) |  | Название города |

Таблица 5 – информация о самолете (описание атрибутов сущности AIRPLANE):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_plane | int | PK | ID самолета |
| name\_plane | varchar(100) |  | Название самолёта |

|  |  |  |  |
| --- | --- | --- | --- |
| seats\_plane | varchar(8) |  | Количество посадочных мест |

Таблица 6 – информация о модели самолета (описание атрибутов сущности MODEL\_PLANE):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_model\_plane | int | PK | ID модели самолёта |
| model\_plane | varchar(150) |  | Название модели самолёта |

Таблица 7 – информация об экипаже (описание атрибутов сущности TEAM):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_team | int | PK | ID команды |
| num\_team | varchar(8) |  | Номер экипажа |

Таблица 8 – информация о сотрудниках (описание атрибутов сущности EMPLOYEE):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_emp | int | PK | ID сотрудника |
| l\_name | varchar(30) |  | Фамилия сотрудника |
| f\_name | varchar(30) |  | Имя сотрудника |
| birthday\_emp | date |  | Дата рождения сотрудника |
| tel\_emp | nvarchar(100) |  | Номер телефона сотрудника |

|  |  |  |  |
| --- | --- | --- | --- |
| position | varchar(100) |  | Должность сотрудника |

Таблица 9 – информация об авиакомпании (описание атрибутов сущности AIRLINE):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_airline | int | PK | ID авиакомпании |
| airline | varchar(100) |  | Название авиакомпании |
| address\_airline | varchar(300) |  | Адрес головного офиса авиакомпании |

Таблица 10 – информация о рейсах (описание атрибутов сущности FLIGHT):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_flight | int | PK | ID рейса |
| departure\_id\_airport | int |  | ID аэропорта вылета |
| arrival\_id\_airport | int |  | ID аэропорта прибытия |
| departure\_date | datetime |  | Дата вылета |
| num\_flight | varchar(9) |  | Номер рейса |
| arrival\_date | datetime |  | Дата прилета |
| status\_flight | varchar(100) |  | Статус рейса |

Таблица 11 – информация о билетах (описание атрибутов сущности TICKET):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_ticket | int | PK | ID билета |
| cost\_ticket | int |  | Стоимость билета |
| is\_sold | varchar(5) |  | Продажа билета (продан/ не продан |
| num\_ticket | int |  | Номер билета |

Таблица 12 – информация о пассажирах (описание атрибутов сущности PASSENGER):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_pas | int | PK | ID пассажира |
| last\_name | varchar(100) |  | Фамилия пассажира |
| first\_name | varchar(100) |  | Имя пассажира |
| email\_pass | varchar(100) |  | email пассажира |
| birthday\_pass | date |  | Дата рождения |
| sex | varchar(6) |  | Пол/статус пассажира |

Таблица 13 – информация об операции, проведенной с билетом (описание атрибутов сущности OPERATION):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_operation | int | PK | ID покупки |
| date\_operation | date |  | Дата оплаты |

Таблица 14 – информация о классе места (описание атрибутов сущности CLASS\_PLACE):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_class | int | PK | ID класса |
| name\_class | varchar(50) |  | Название класса |

Таблица 15 – Информация о месте в самолете (описание атрибутов сущности PLACES):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_place | int | PK | ID места |
| num\_place | varchar(8) |  | Номер места |
| num\_row | varchar(8) |  | Номер ряда |

Таблица 16 – информация о выполненной операции (описание атрибутов сущности TYPE\_OPERATION):

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_type\_operation | int | PK | ID операции |

|  |  |  |  |
| --- | --- | --- | --- |
| name\_operation | varchar(100) |  | Название операции |

Таблица 17 – информация о распределении самолетов на каждый рейс (описание атрибутов сущности flight\_air) :

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_flight | int | PK | ID рейса |
| id\_plane | int | PK | ID самолета |

Таблица 18 – информация о распределении сотрудников по командам (описание атрибутов сущности team\_emp) :

|  |  |  |  |
| --- | --- | --- | --- |
| Имя атрибута | Тип данных | Комментарий | Значение |
| id\_team | int | PK | ID команды |
| id\_emp | int | PK | ID сотрудника |

# Типы и описание связей между сущностями

Таблица 19 – информация о типах связей между сущностями:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название связи | Главная таблица | Дочерняя таблица | Тип связи | Идентифицирующая |
| flight\_air | FLIGHT | AIRPLANE | (M:N) | см. ниже |
| team\_emp | TEAM | EMPLOYEE | (M:N) | см.ниже |
| which model | MODEL\_PLANE | AIRPLANE | (1:N) | нет |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| seat in plane | MODEL\_PLANE | PLACES | (1:N) | нет |
| fly | AIRPORT | FLIGHT | (1:N) | нет |
| locate | TOWN | AIRPORT | (1:N) | нет |
| belong to airline | AIRLINE | AIRPLANE | (1:N) | нет |
| domestic | COUNTRY | TOWN | (1:N) | нет |
| ticket operation | TICKET | OPERATION | (1:N) | нет |
| operation type | TYPE\_OPERATION | OPERATION | (1:N) | нет |
| place class | CLASS\_PLACE | PLACES | (1:N) | нет |
| ticket for the flight | FLIGHT | TICKET | (1:N) | нет |
| pas operation | PASSENGER | OPERATION | (1:N) | нет |
| has a type | TYPE\_AIRPORT | AIRPORT | (1:N) | нет |
| team flight | FLIGHT | TEAM | (1:N) | нет |

1. Связь "самолет в рейсе" между сущностями FLIGHT и AIRPLANE - (M:N), так как каждый самолет может участвовать в нескольких рейсах, а в каждом рейсе могут участвовать несколько самолетов. Участвующие в связи сущности равноправны, и такая связь не может идентифицировать экземпляры этих сущностей. Связь обязательна с обеих сторон, так как рейс невозможен без самолета, а самолет без рейса. Поэтому определение характера связи (M:N) не имеет смысла.

2. Связь "входит в команду" между сущностями TEAM и EMPLOYEE - (M:N), поскольку каждый сотрудник может быть частью нескольких команд, а каждая команда состоит из нескольких сотрудников. Связь равноправная и не позволяет идентифицировать экземпляры сущностей. Связь обязательна с обеих сторон, так как команда невозможна без сотрудников, а сотрудники не могут существовать без команды. Определение характера связи (M:N) не имеет смысла.

3. Связь "имеет модель" между сущностями MODEL\_PLANE и AIRPLANE - (1:N), так как каждый самолет имеет определенную модель, а моделей самолета может быть несколько. Связь не идентифицирующая, так как сущность AIRPLANE имеет первичный ключ, достаточный для идентификации. Идентифицирующая связь может привести к избыточности ключа. Участие сущности MODEL\_PLANE в связи обязательно, а сущности AIRPLANE - нет.

4. Связь "места в самолете" между сущностями MODEL\_PLANE и PLACES - (1:N), так как определенный набор мест закреплен за моделью самолета, но может принадлежать нескольким моделям. Связь не идентифицирующая (аналогично пункту 3). Участие сущности MODEL\_PLANE в связи обязательно, а сущности PLACES - нет.

5. Связь "вылетают/прилетают" между сущностями AIRPORT и FLIGHT - (1:N), так как из одного аэропорта могут вылетать несколько рейсов, но каждый рейс вылетает только из одного аэропорта. Связь не идентифицирующая (аналогично пункту 3). Участие сущности AIRPORT в связи обязательно (рейс невозможен без аэропорта), а сущности FLIGHT - нет (аэропорт может существовать без рейсов).

6. Связь "находится в городе" между сущностями TOWN и AIRPORT - (1:N), так как каждый аэропорт находится в определенном городе, но в каждом городе может быть несколько аэропортов. Связь не идентифицирующая (аналогично пункту 3). Участие сущности TOWN в связи обязательно, а сущности AIRPORT - нет.

7. Связь "принадлежит авиакомпании" между сущностями AIRLINE и AIRPLANE - (1:N), так как каждый самолет принадлежит определенной авиакомпании, а у одной авиакомпании может быть несколько самолетов. Связь не идентифицирующая (аналогично пункту 3). Участие сущности AIRLINE в связи обязательно, а сущности AIRPLANE - нет.

8. Связь "находится в стране" между сущностями COUNTRY и TOWN - (1:N), так как каждый город находится в определенной стране, но в каждой стране может быть несколько городов. Связь не идентифицирующая (аналогично пункту 3). Участие сущности COUNTRY в связи обязательно, а сущности TOWN - нет.

9. Связь "операция с билетом" между сущностями TICKET и OPERATION - (1:N), так как каждой операции соответствует один билет, а по одному билету может пройти несколько операций. Связь не идентифицирующая (аналогично пункту 3). Участие сущности TICKET в связи обязательно, а сущности OPERATION - нет.

10. Связь "тип операции" между сущностями TYPE\_OPERATION и OPERATION - (1:N), так как каждой операции соответствует один тип, но одному типу может принадлежать несколько операций. Связь не идентифицирующая (аналогично пункту 3). Участие сущности TYPE\_OPERATION в связи обязательно, а сущности OPERATION - нет.

11. Связь между сущностями CLASS\_PLACE и PLACES - (1:N), так как у каждого места есть определенный класс, а каждому классу принадлежит несколько мест. Связь не идентифицирующая (аналогично пункту 3). Участие сущности CLASS\_PLACE в связи обязательно, а сущности PLACES - нет.

12. Связь "билет на полет" между сущностями FLIGHT и TICKET - (1:N), так как каждому билету соответствует один полет, но на один полет может быть продано много билетов. Связь не идентифицирующая (аналогично пункту 3). Участие сущности FLIGHT в связи обязательно, а сущности TICKET - нет.

13. Связь "операции пассажиров" между сущностями PASSANGER и OPERATION - (1:N), так как каждой операции соответствует один пассажир, а один пассажир может выполнить несколько операций. Связь не идентифицирующая (аналогично пункту 3). Участие сущности PASSANGER в связи обязательно, а сущности OPERATION - нет.

14. Связь "аэропорт имеет тип" между сущностями TYPE\_AIRPORT и AIRPORT - (1:N), так как каждый аэропорт имеет тип, но каждый тип может принадлежать нескольким аэропортам. Связь не идентифицирующая (аналогично пункту 3). Участие сущности TYPE\_AIRPORT в связи обязательно, а сущности AIRPORT - нет.

15. Связь "команда полета" между сущностями FLIGHT и TEAM - (1:N), так как в каждом перелете участвует определенная команда, но одна команда может участвовать в нескольких перелетах. Связь не идентифицирующая (аналогично пункту 3). Участие сущности FLIGHT в связи обязательно, а сущности TEAM - нет.

# Алгоритмы обработки данных, используемые в сценариях

В процессе выполнения данного проекта для компании N мы используем стандартные для большинства СУБД SQL-запросы в качестве основного инструмента обработки данных. Кроме того, мы задействуем хранимые процедуры, индексы, функции и триггеры, чтобы обеспечить более качественную и эффективную обработку данных.

# Бизнес-функция

Главная бизнес-функция заключается в учете продаж авиабилетов и получении различной статистики, охватывающей все этапы процесса покупки авиабилетов для различных авиакомпаний.

# Планируемый перечень отчетных форм

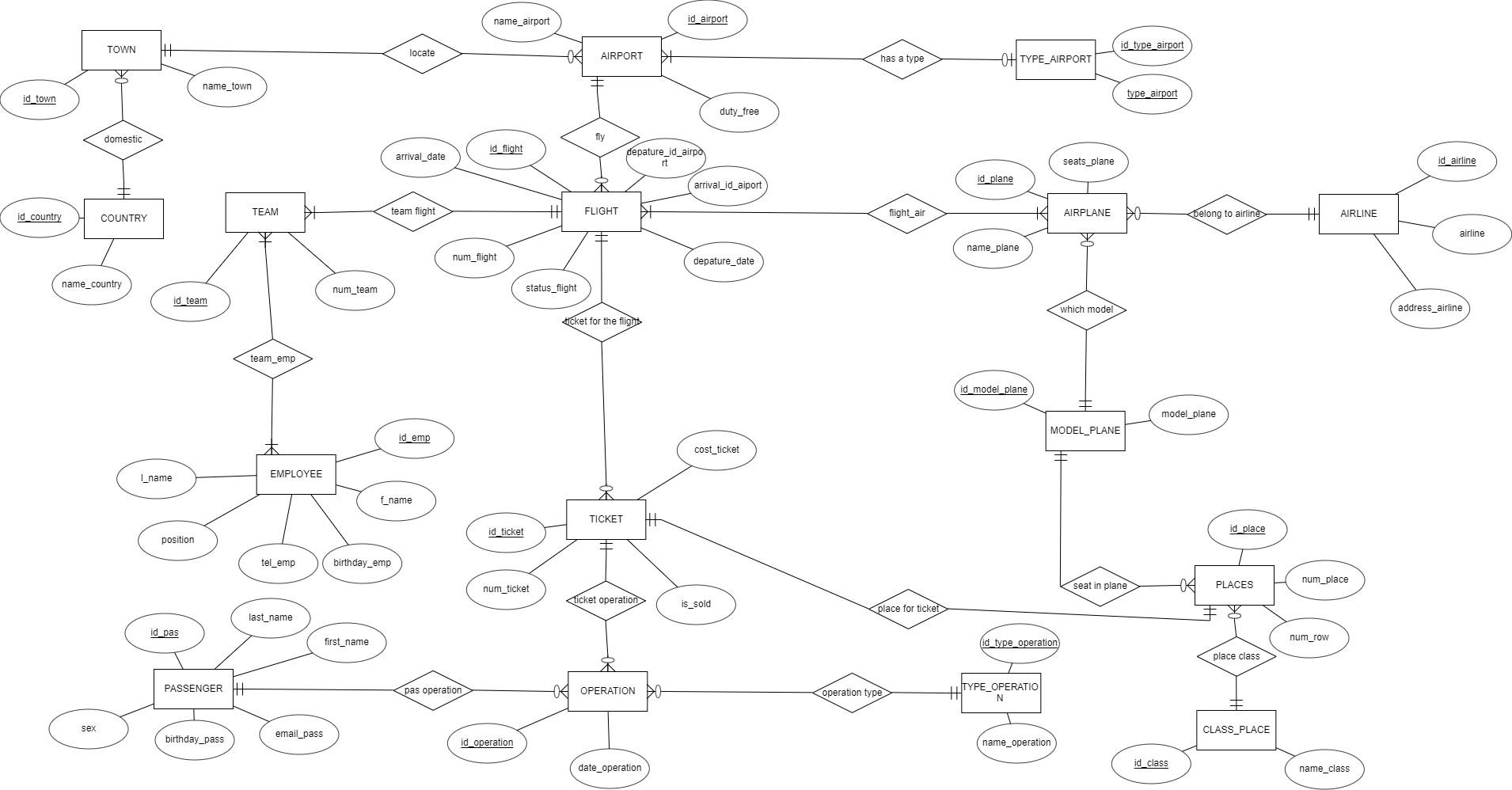
В результате нашей работы компания N сможет получать разнообразную статистику по различным параметрам. Например, можно будет узнать, какой город является наиболее популярным для перелетов среди клиентов авиакомпаний, какова средняя цена билета, сколько в среднем людей покупают авиабилеты в конкретный день, по какой цене чаще всего приобретаются билеты, а также какой класс перелета имеет больший спрос среди клиентов. На основе этой информации можно будет корректировать цены на определенные направления перелетов, а также в определенные дни месяца. Статистика будет извлекаться и предоставляться пользователям нашей базы данных в конкретной форме, например, в виде диаграммы, графика или таблицы, которые будут созданы с помощью инструмента визуализации данных.

# Возможная архитектура программного продукта на основе нашей базы данных

Архитектура программного продукта на основе нашей базы данных будет ориентирована на клиентов авиакомпаний, которые обращаются к организации с целью приобретения авиабилета. Потребитель будет выбирать город назначения, класс перелета, место в самолете, а также вводить свои персональные данные, необходимые для покупки авиабилета. Далее, приложение будет использовать разработанную нами базу данных для предоставления всех возможных вариантов выбора и передавать эту информацию клиенту.

# Инфологическая модель

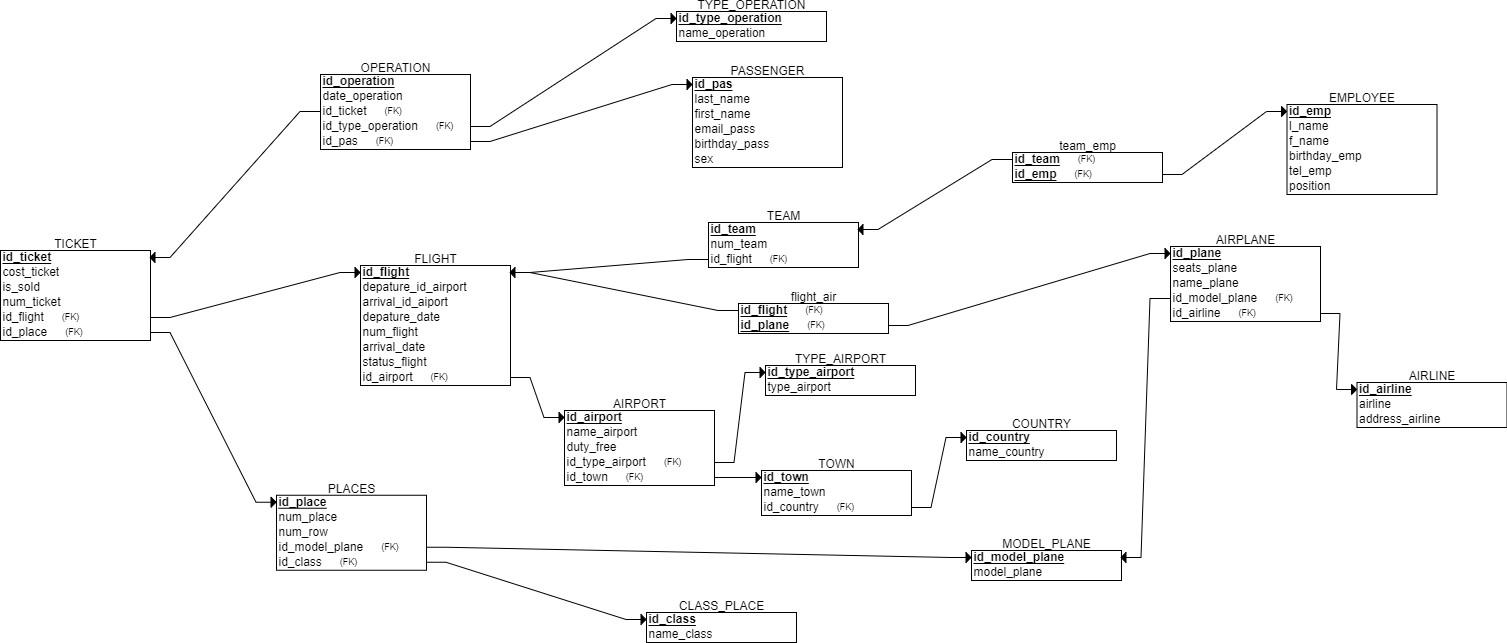
С помощью инструмента ERDPlus мы разработали инфологическую модель, которая дает возможность наглядно увидеть описание структуры всей задействованной области:



Инфологическая модель

# Даталогическая модель

В контексте СУБД данная модель представляет собой инфологическую, показывает данные и связи между ними:



Даталогическая модель

В таблице с названием FLIGHT для нашей БД, следующие FK: id\_departure\_airport (int), id\_arrival\_airport (int). В таблицах team\_emp и flight\_air составной первичный ключ.

# Типы данных в физической модели

Типы данных показаны с помощью диаграммы:

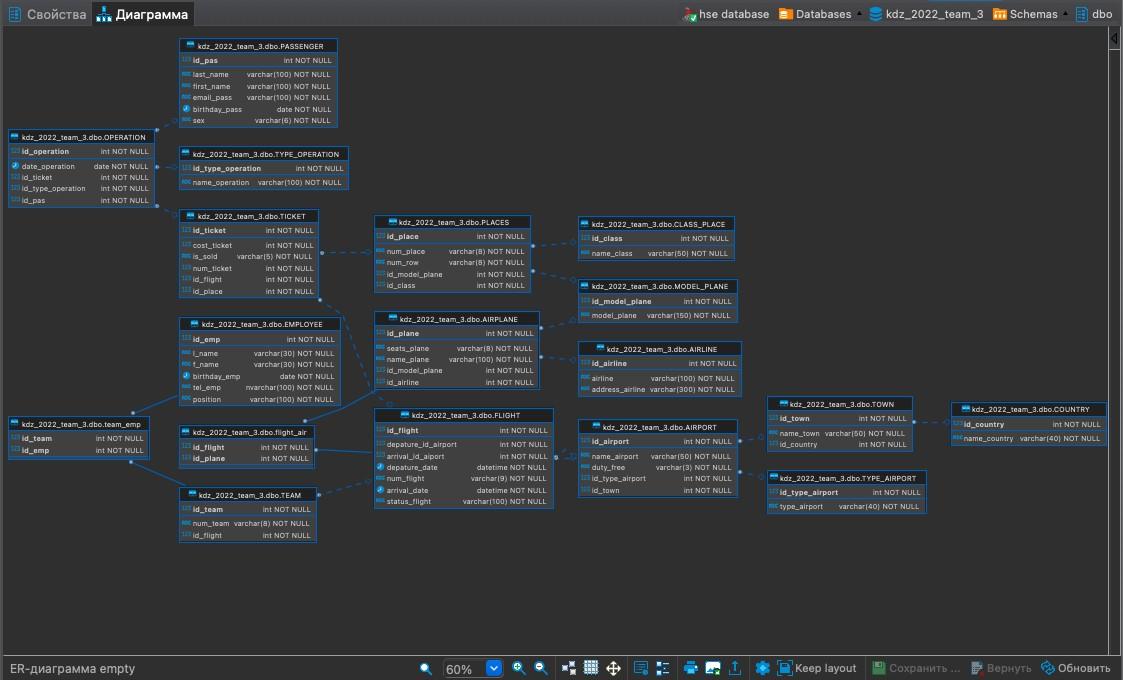
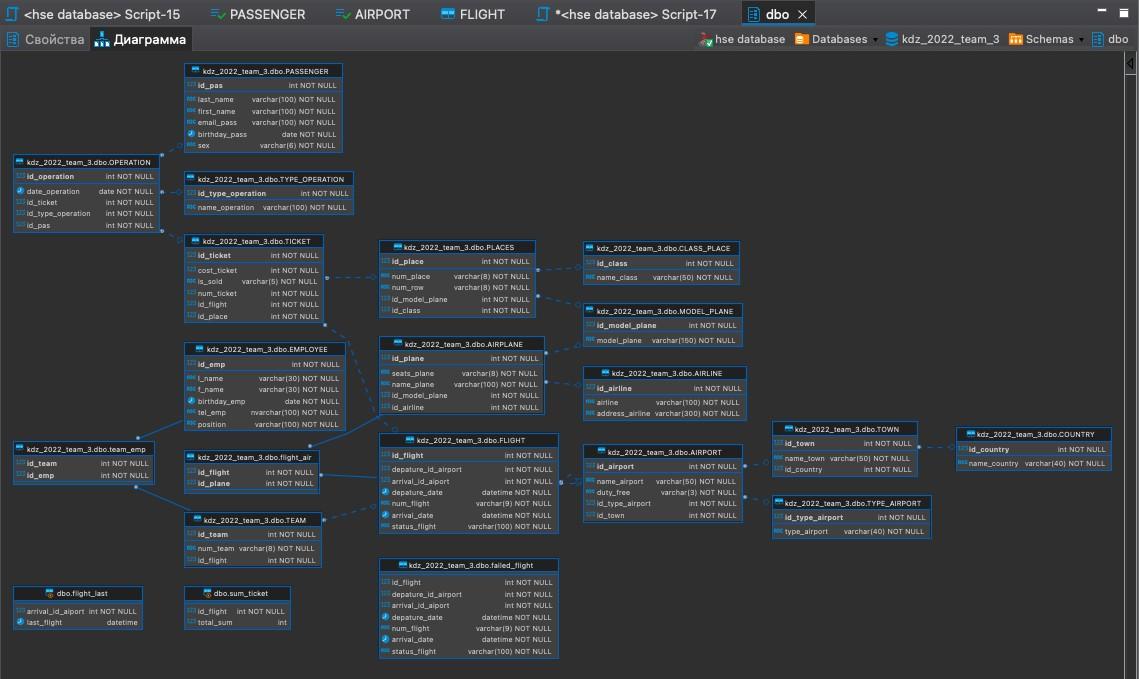
Первоначальная диаграмма

Диаграмма ниже представляет собой конечную версию, после проведенной работы (а именно, написания процедур, запросов, и.т.д.):



Итоговая диаграмма

# Соответствие таблиц 3НФ

Критерии для таблиц 3НФ:

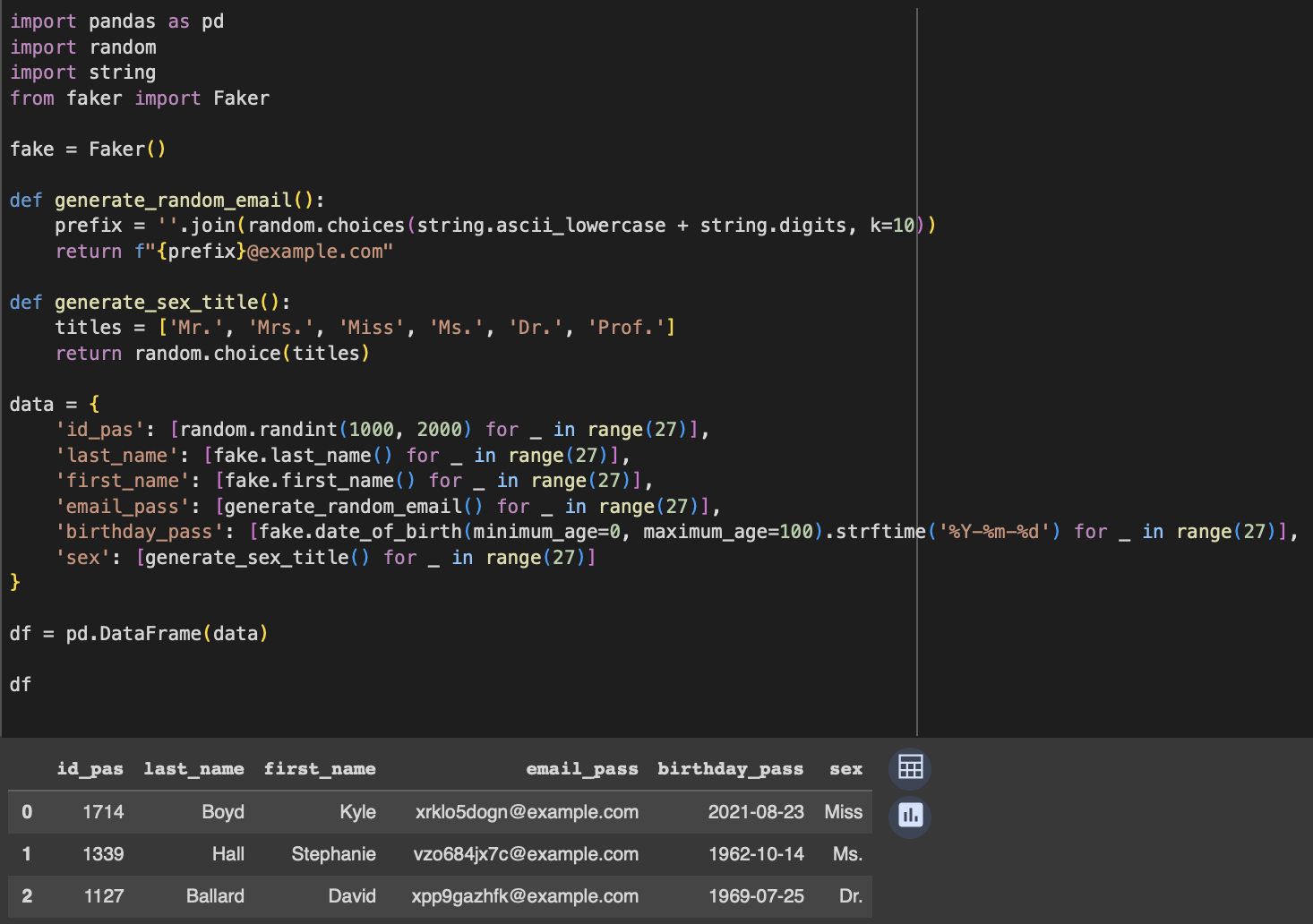
1. Таблица должна быть 2НФ. Для выполнения этого критерия таблица должна быть в 1НФ, должен быть первичный ключ, все атрибуты должны зависеть целиком от первичного ключа. При этом в 1НФ не может быть повторяющихся строк, а в каждой клетке таблицы должно быть ровно одно значение.
2. Каждый неключевой атрибут нетранзитивно функционально зависит от ключевого.

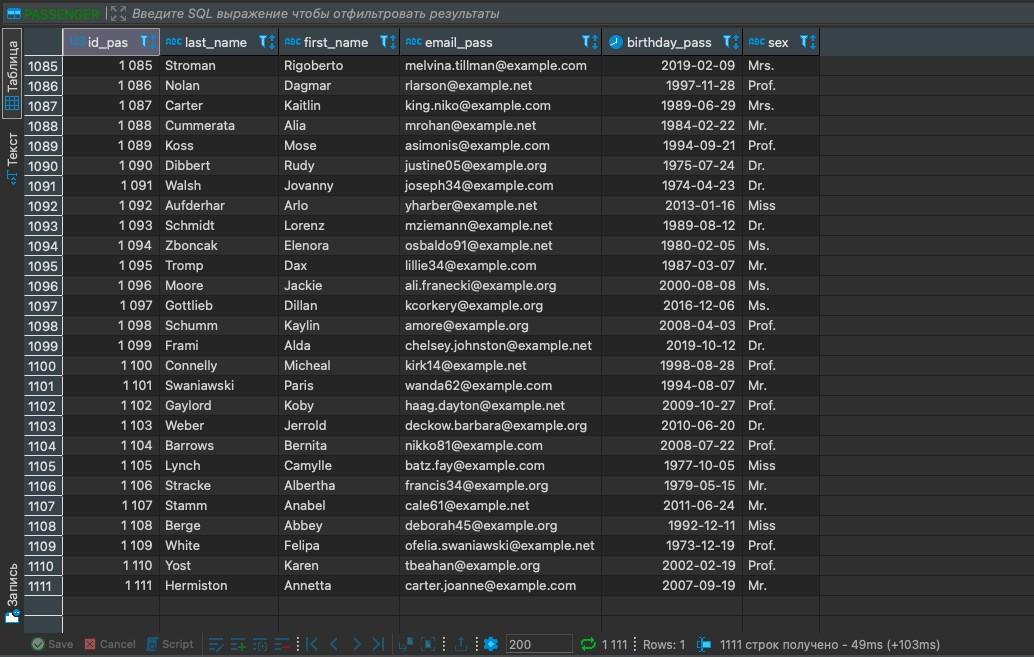
Таблица 20 - подтверждение нахождения таблиц в 3НФ:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Таблица | Повторяющие ся строки | Первичны й ключ | Ровно одно значение в клетке | Транзитивные функциональные зависимости | Зависимость атрибутов от первичного ключа |
| TOWN | - | + | + | - | + |
| COUNTRY | - | + | + | - | + |
| TEAM | - | + | + | - | + |
| EMPLOYEE | - | + | + | - | + |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| PASSENGER | - | + | + | - | + |
| AIRPORT | - | + | + | - | + |
| FLIGHT | - | + | + | - | + |
| OPERATION | - | + | + | - | + |
| TYPE\_AIRPORT | - | + | + | - | + |
| AIRPLANE | - | + | + | - | + |
| MODEL\_PLANE | - | + | + | - | + |
| TYPE\_OPERATION | - | + | + | - | + |
| CLASS\_PLACE | - | + | + | - | + |
| PLACES | - | + | + | - | + |
| AIRLINE | - | + | + | - | + |
| TICKET | - | + | + | - | + |
| FLIGHT\_AIR | - | + | + | - | + |
| TEAM\_EMP | - | + | + | - | + |

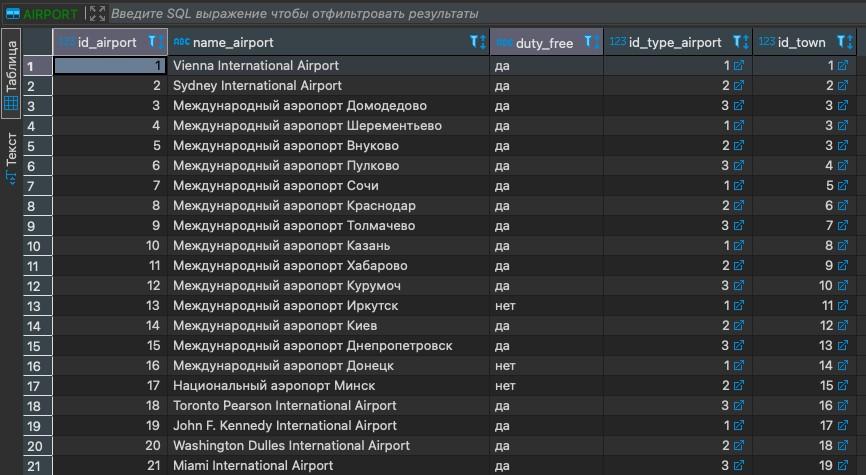
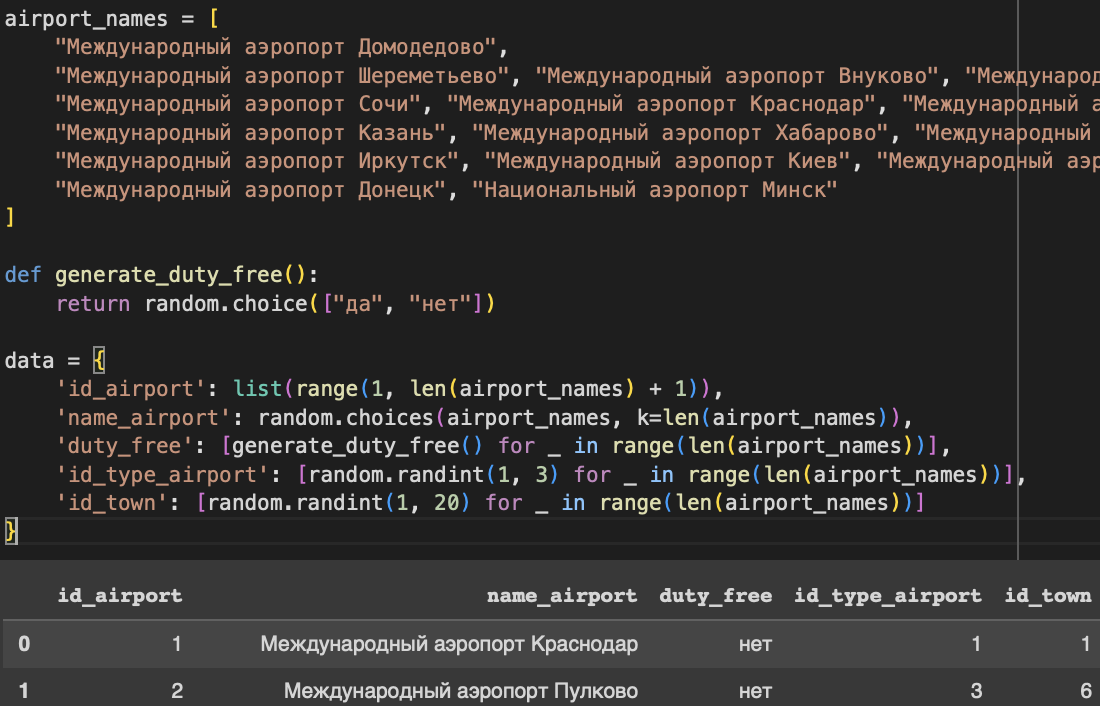
# Генерация и заполнение БД данными

В соответствии с критериями команда заполнила таблицы базы данными. Так, таблица PASSENGER содержит больше 1000 строк (1111). Каждая из таблиц заполнена случайно сгенерированными данными. Для этого мы использовали библиотеки Python. Примеры заполнения таблиц и фрагменты кода:

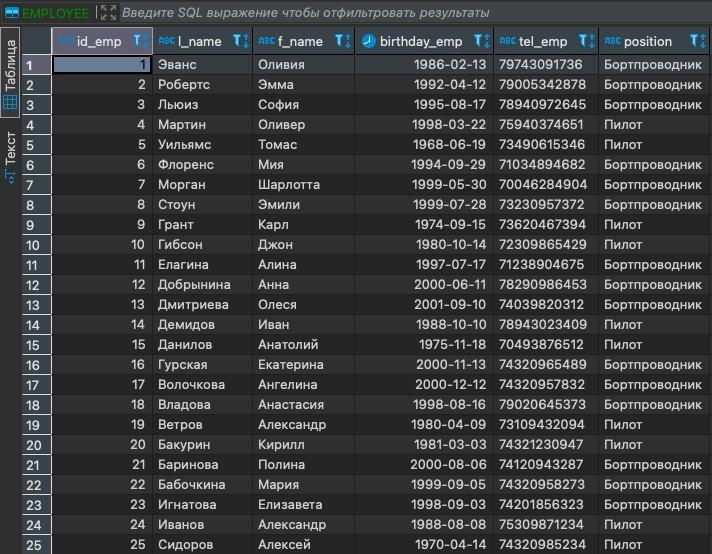


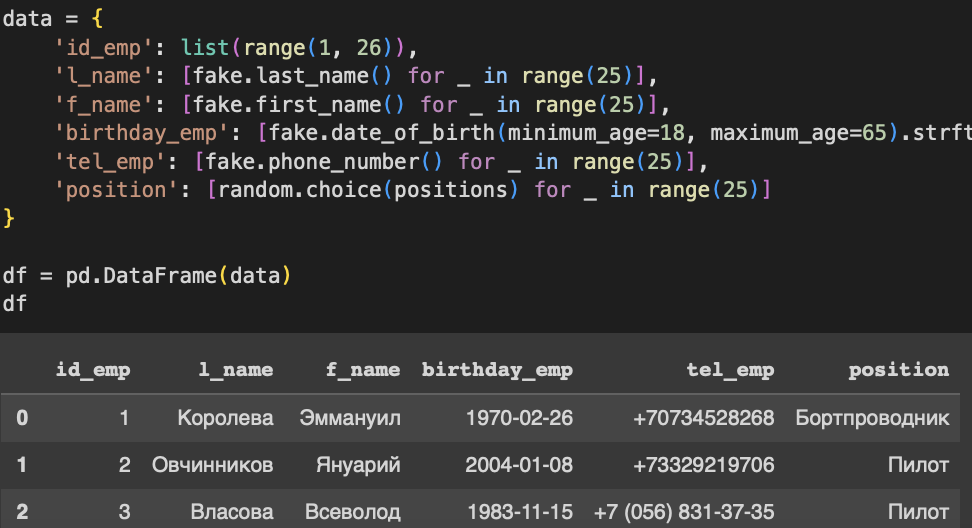
Пример заполнения таблицы PASSENGER

Ниже представлены примеры заполнения еще нескольких таблиц:

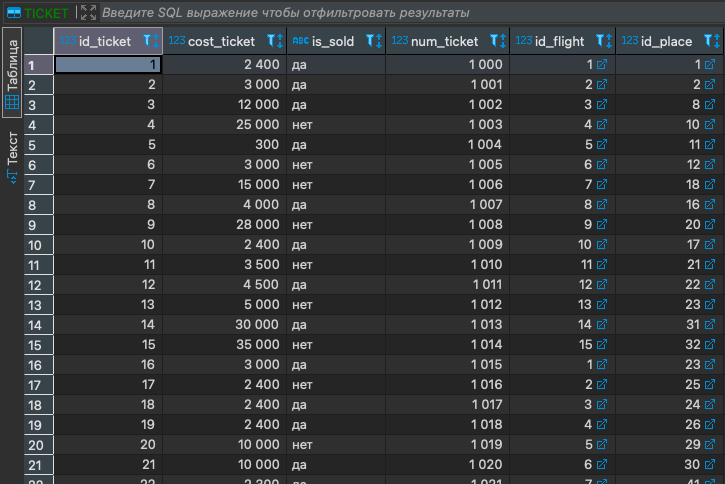


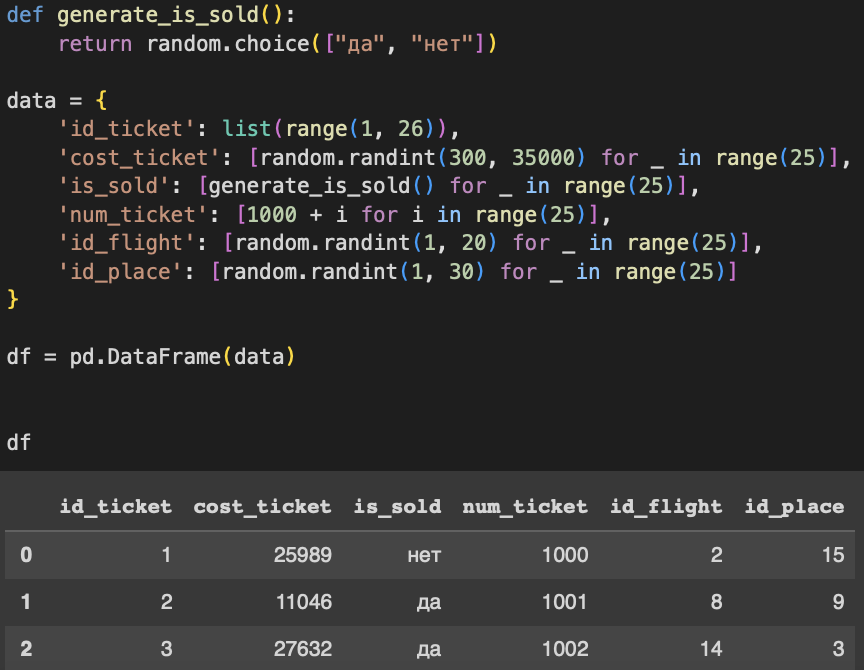
Пример заполнения таблицы AIRPORT





Пример заполнения таблицы EMPLOYEE





Пример заполнения таблицы TICKET

# Настройка индексов

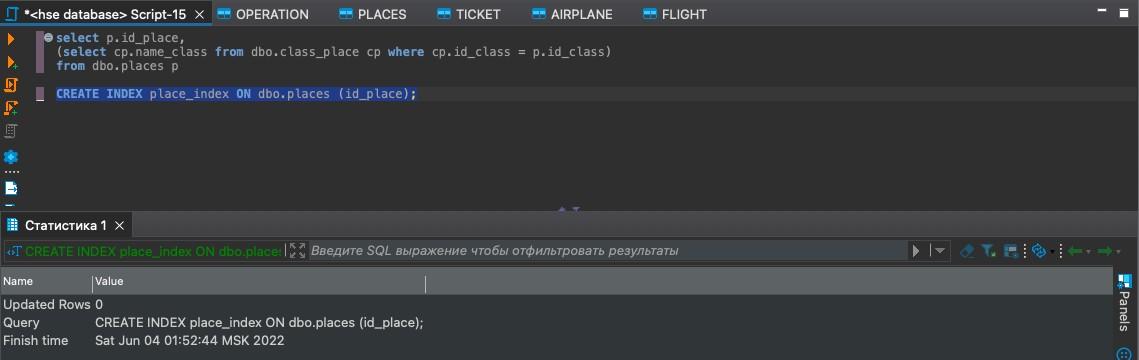
Для настройки индекса используем запрос №3 (см. ниже). Индекс будет значительно сокращать время выполнения запроса, тем самым, оптимизируя процесс.

Первоначальное время:

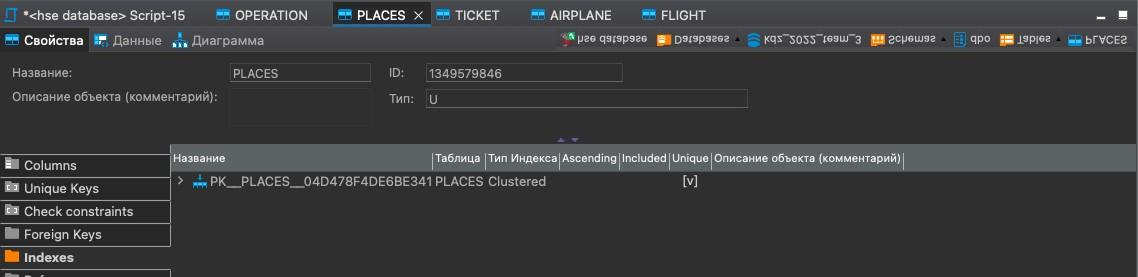
Первоначальное время обработки запроса

Создание индекса:

CREATE INDEX place\_index ON dbo.places (id\_place);

Создание индекса

Индекс:

Наличие индекса в таблице PLACES Время обработки после создания индекса:

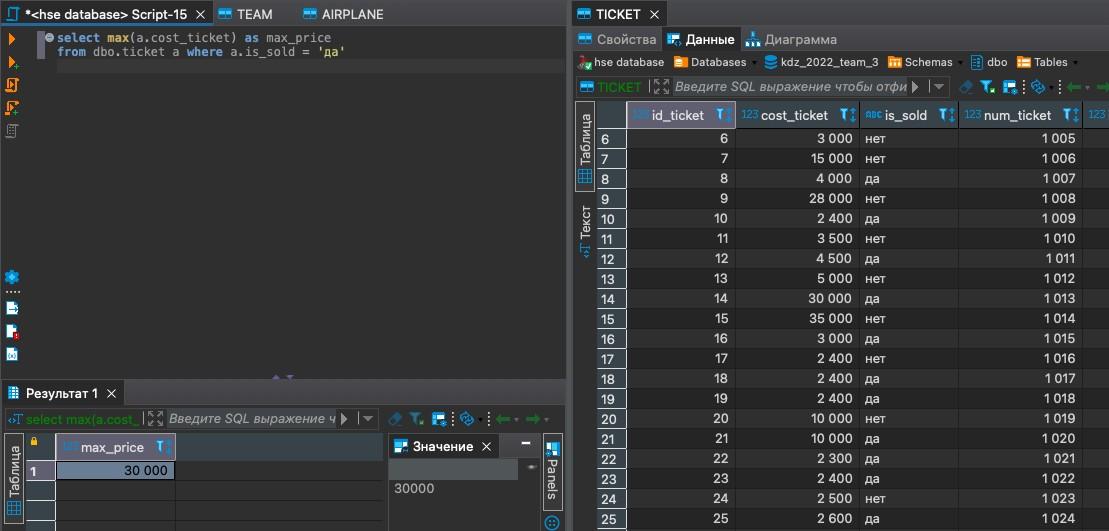


Конечное время обработки запроса

# Разработка запросов

1. Вывод максимальной цены приобретенного билета (простой запрос с условием и формулами в SELECT):

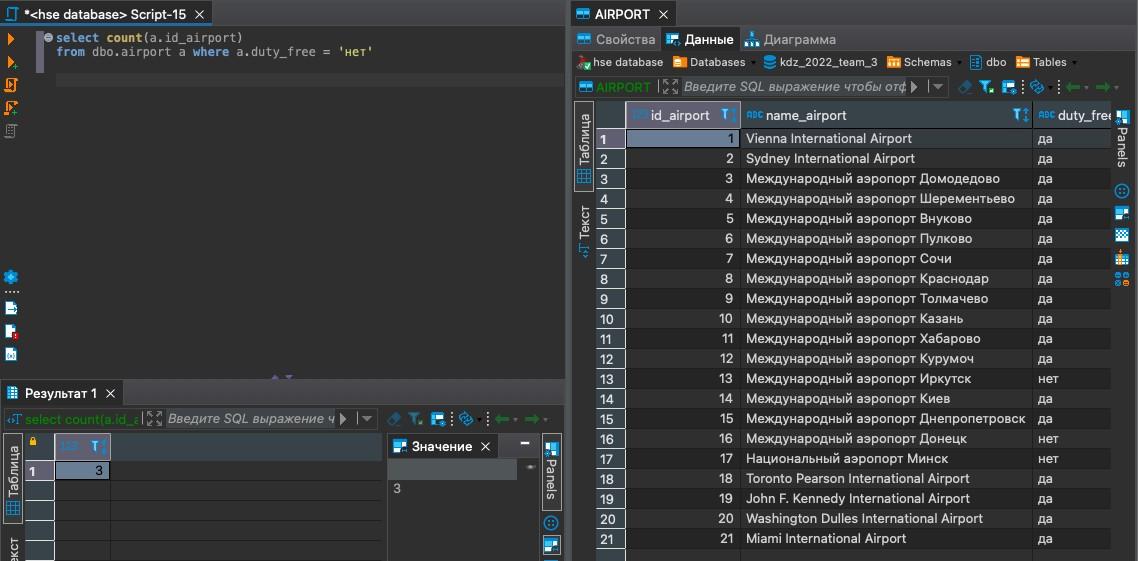
select max(a.cost\_ticket) as max\_price from dbo.ticket a where a.is\_sold ='да'

Запрос №1

1. Вывод количества аэропортов, где нет duty free (простой запрос с условием и формулами в SELECT):

select count(a.id\_airport)

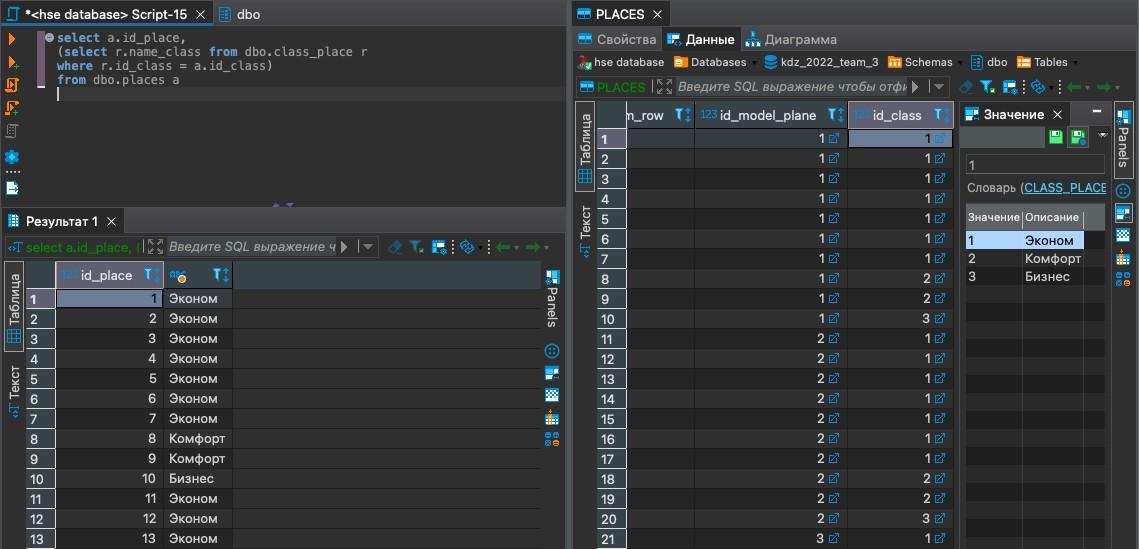
from dbo.airport a where a.duty\_free = 'нет'

Запрос №2

1. Вывод класса для каждого места (запрос с коррелированным подзапросом в SELECT):

select a.id\_place,

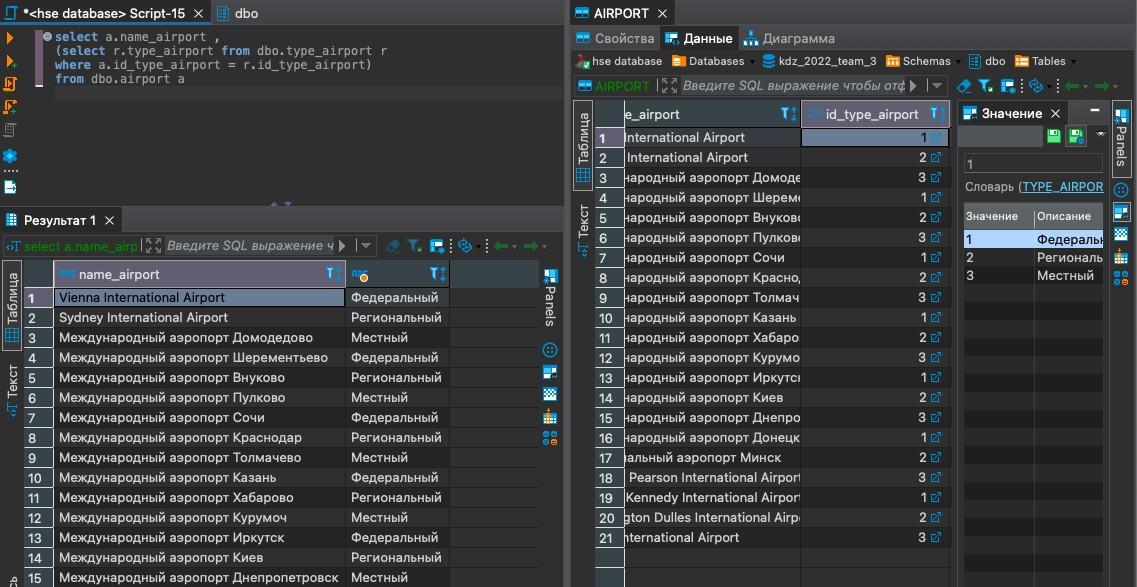
(select r.name\_class from dbo.class\_place r where r.id\_class = a.id\_class) from dbo.places a

Запрос №3

1. Вывод типа для каждого аэропорта (запрос с коррелированным подзапросом в SELECT):

select a.name\_airport ,

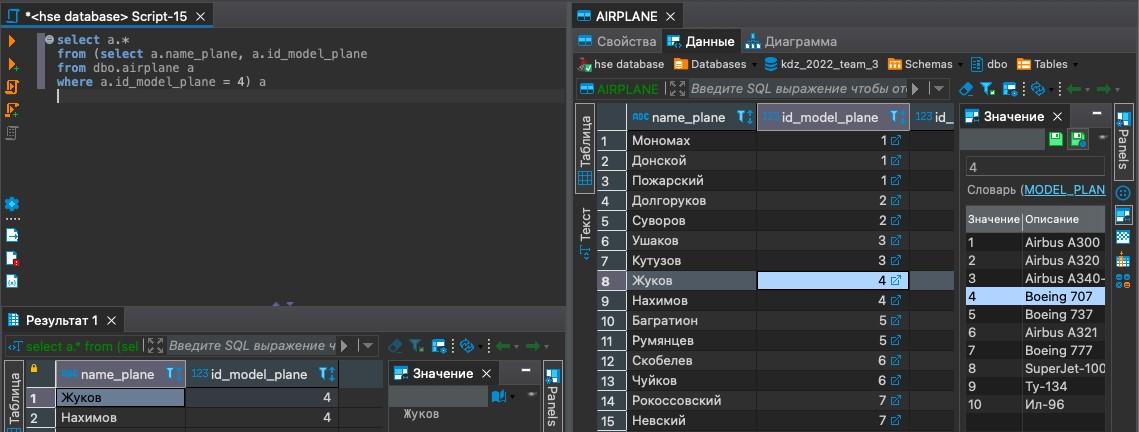
(select r.type\_airport from dbo.type\_airport r where a.id\_type\_airport = r.id\_type\_airport) from dbo.airport a



Запрос №4

1. Вывод именных названий самолетов модели Boeing 707 (запрос с подзапросом в FROM):

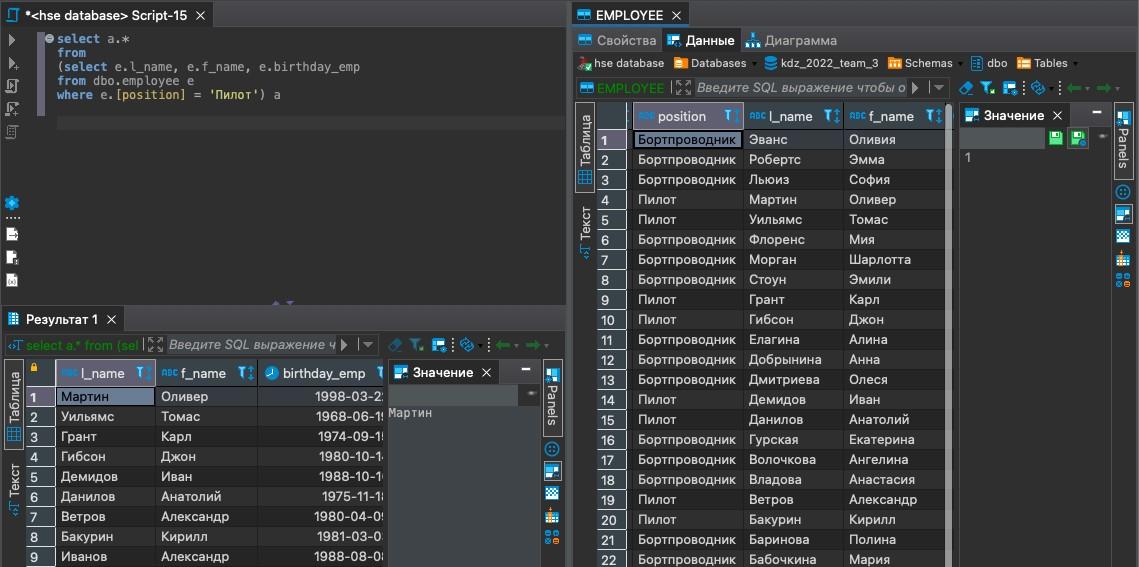
select a.\*

from (select a.name\_plane, a.id\_model\_plane from dbo.airplane a where a.id\_model\_plane = 4) a

Запрос №5

1. Вывод имен, фамилий, дат рождений всех пилотов (запрос с подзапросом в FROM): select a.\*

from (select e.l\_name, e.f\_name, e.birthday\_emp from dbo.employee e where e.[position] = 'Пилот') a



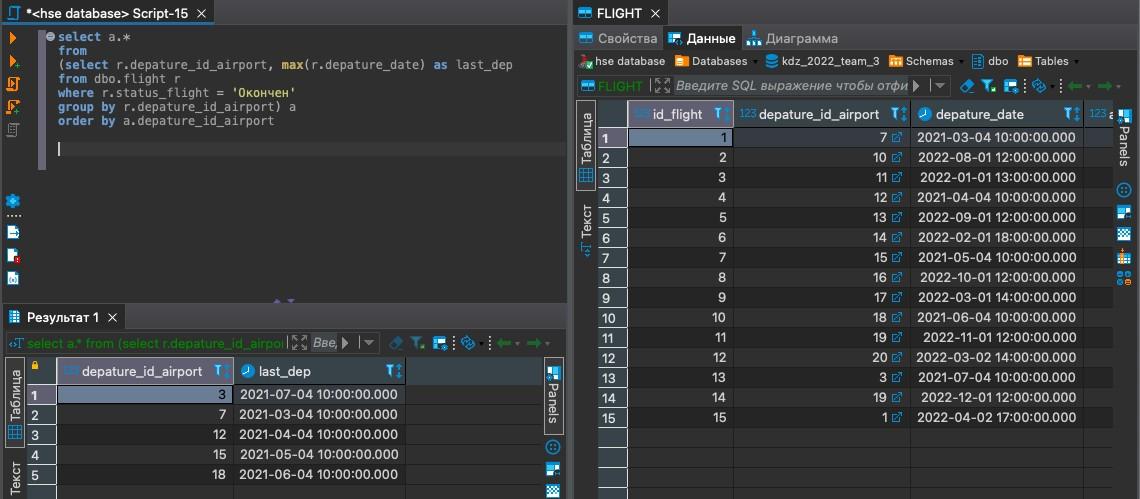
Запрос №6

1. Вывод аэропорта и времени крайнего вылета из этого аэропорта для завершенных полетов (запрос с подзапросом в FROM, агрегированием, группировкой и сортировкой):

select a.\*

from (select r.depature\_id\_airport, max(r.depature\_date) as last\_dep

from dbo.flight r where r.status\_flight = 'Окончен' group by r.depature\_id\_airport) a order by a.depature\_id\_airport



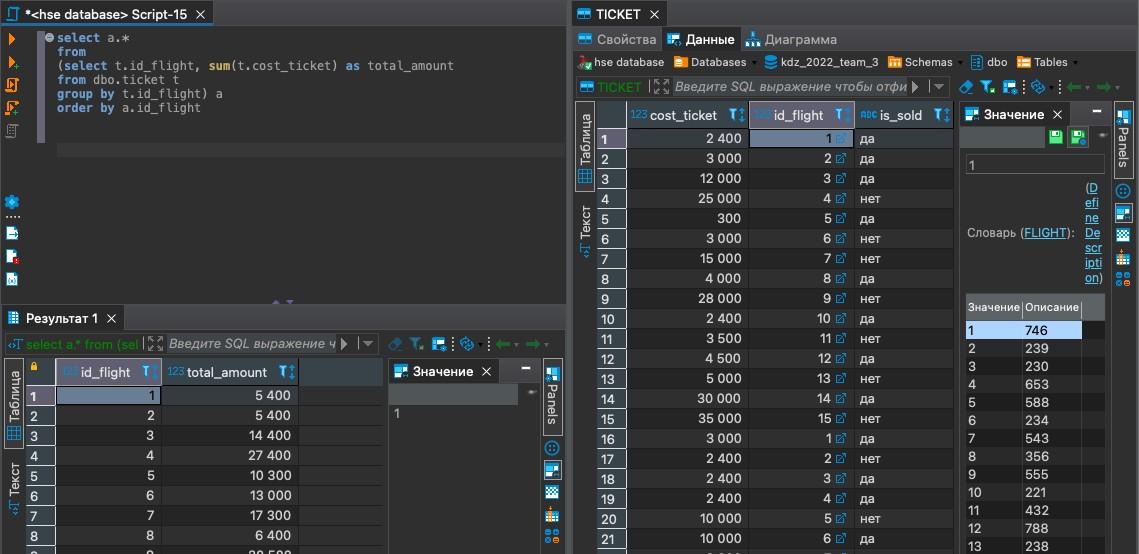
Запрос №7

1. Итоговая сумма билетов, которые купили, в разрезе полетов (запрос с подзапросом в FROM, агрегированием, группировкой и сортировкой):

select a.\*

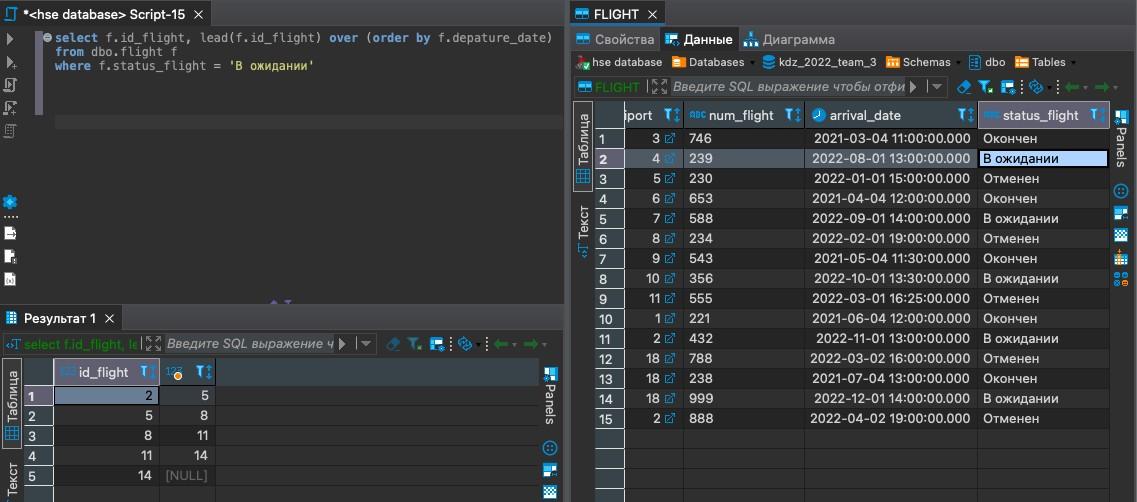
from (select t.id\_flight, sum(t.cost\_ticket) as total\_amount from dbo.ticket t group by t.id\_flight) a

order by a.id\_flight

Запрос №8

1. Вывод полета, находящегося в ожидании, а также следующего за ним (запрос, использующий оконную функцию LEAD):

select f.id\_flight, lead(f.id\_flight) over (order by f.depature\_date) from dbo.flight f where f.status\_flight = 'В ожидании'

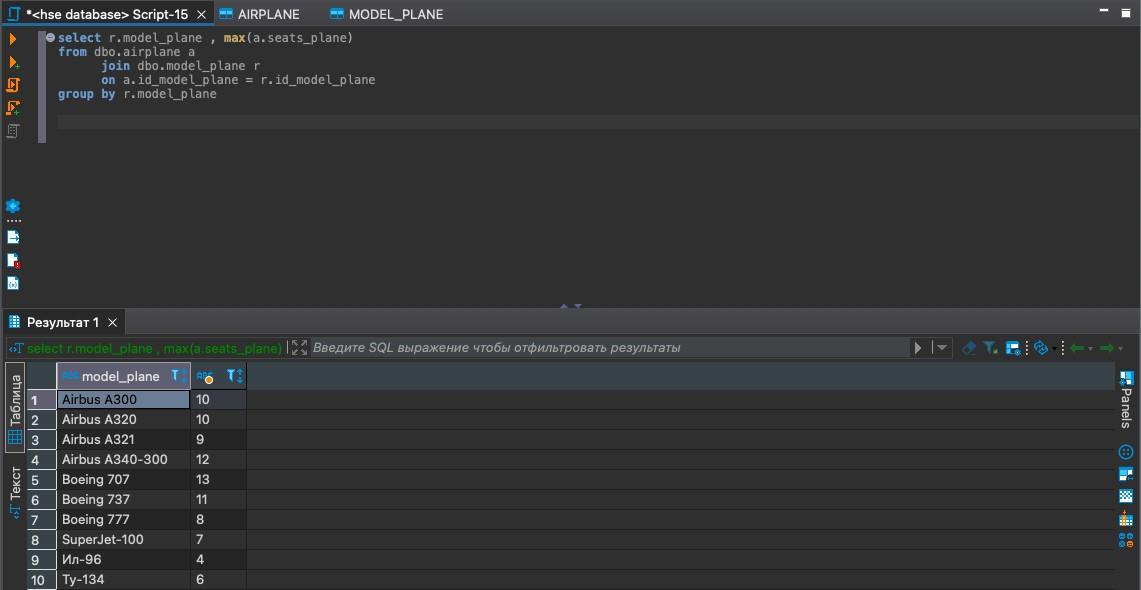


Запрос №9

1. Вывод количества мест для каждой модели самолета (запрос с агрегированием и выражением JOIN, включающим не менее 2 таблиц):

select r.model\_plane , max(a.seats\_plane) from dbo.airplane a

join dbo.model\_plane r

on a.id\_model\_plane = r.id\_model\_plane group by r.model\_plane

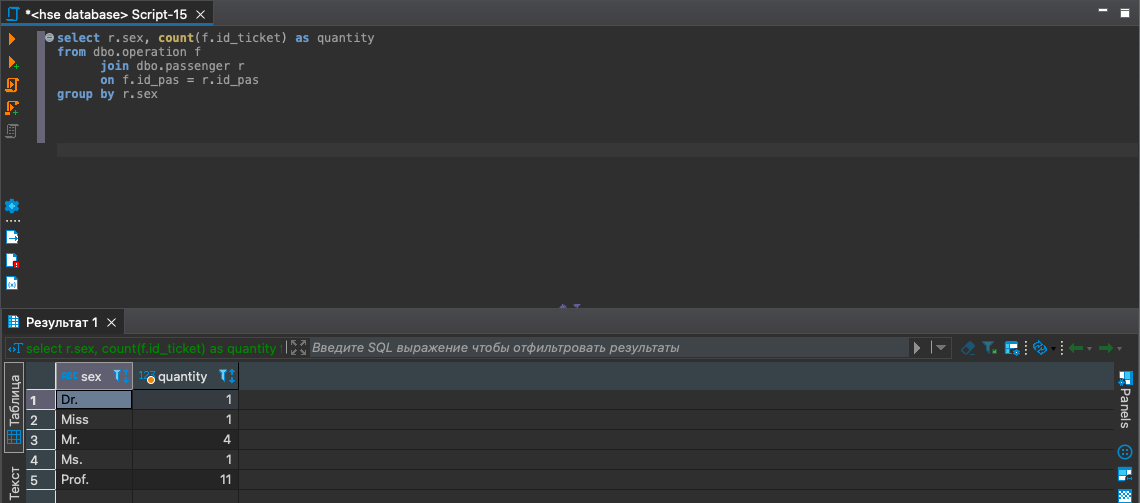
Запрос №10

1. Вывод количества купленных билетов по полу/статусу человека (запрос с агрегированием и выражением JOIN, включающим не менее 2 таблиц):

select r.sex, count(f.id\_ticket) as quantity from dbo.operation f

join dbo.passenger r on f.id\_pas = r.id\_pas

group by r.sex

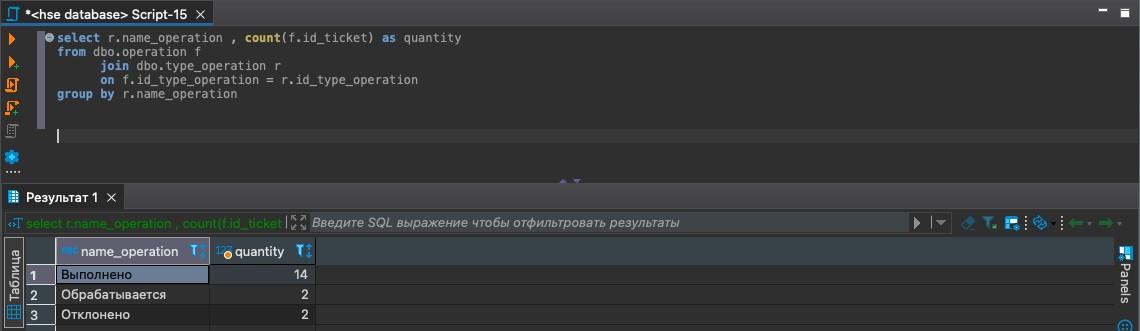
Запрос №11

1. Вывод количества билетов на каждой стадии обработки (запрос с агрегированием и выражением JOIN, включающим не менее 2 таблиц):

select r.name\_operation , count(f.id\_ticket) as quantity from dbo.operation f

join dbo.type\_operation r

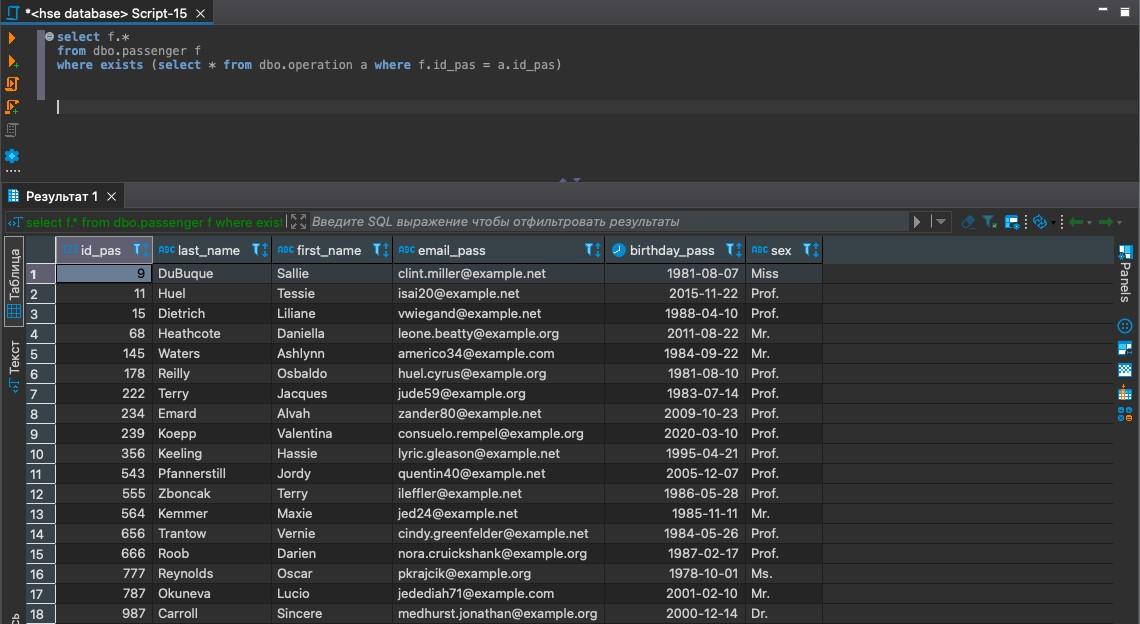
on f.id\_type\_operation = r.id\_type\_operation group by r.name\_operation

Запрос №12

1. Вывод данных о пассажирах, которые совершили любую операцию (запрос с EXISTS):

select f.\* from dbo.passenger f

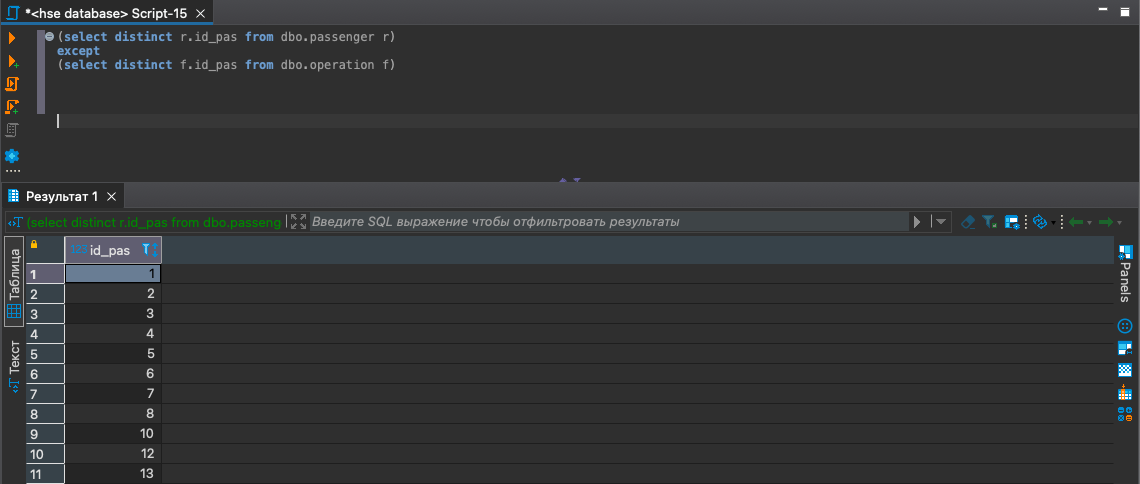
where exists (select \* from dbo.operation a where f.id\_pas = a.id\_pas)

Запрос №13

1. Вывод ID клиентов, которые не совершали никаких операций (запрос, использующий манипуляции с множествами):

(select distinct r.id\_pas from dbo.passenger r) except

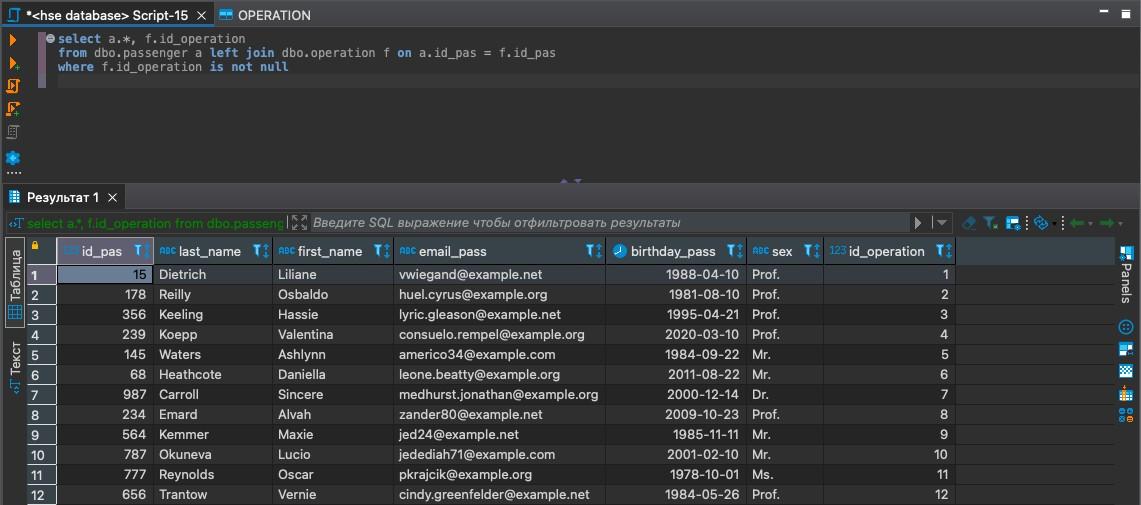
(select distinct f.id\_pas from dbo.operation f)

Запрос №14

1. Вывод полной информации по клиенту, который совершил хотя бы одну операцию, с присоединением ID совершенной операции (запрос с внешним соединением и проверкой на наличие NULL):

select a.\*, f.id\_operation

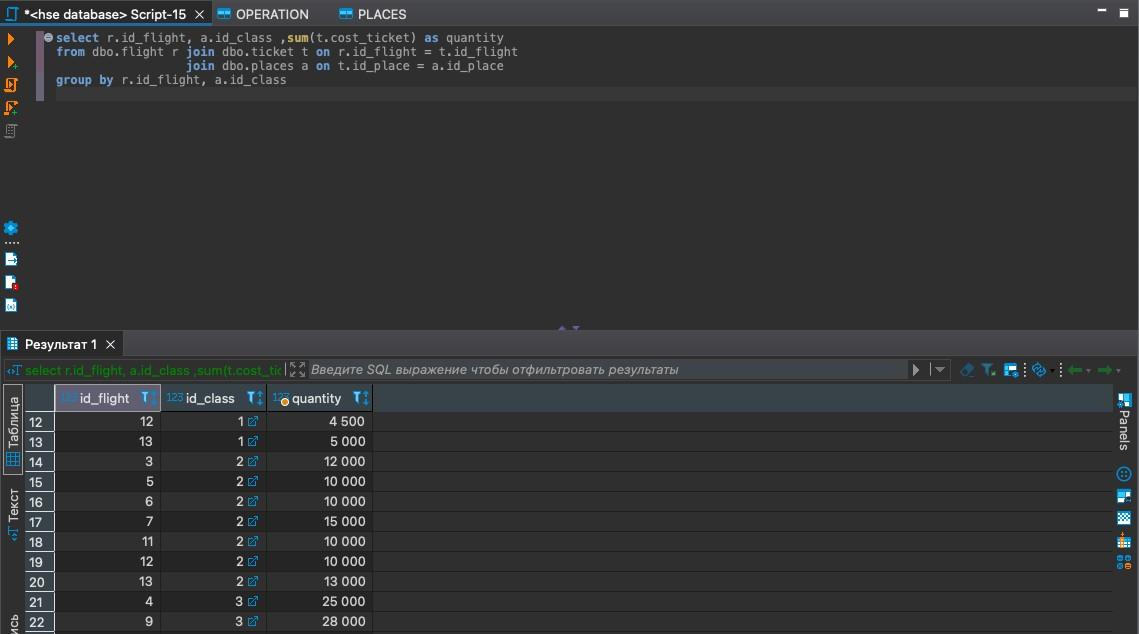
from dbo.passenger a left join dbo.operation f on a.id\_pas = f.id\_pas where f.id\_operation is not null

Запрос №15

1. Вывод суммарной стоимости билетов каждого класса на каждый рейс (запрос с агрегированием и выражением JOIN, включающим не менее 3 таблиц/выражений):

select r.id\_flight, a.id\_class ,sum(t.cost\_ticket) as quantity from dbo.flight r join dbo.ticket t on r.id\_flight = t.id\_flight

join dbo.places a on t.id\_place = a.id\_place group by r.id\_flight, a.id\_class

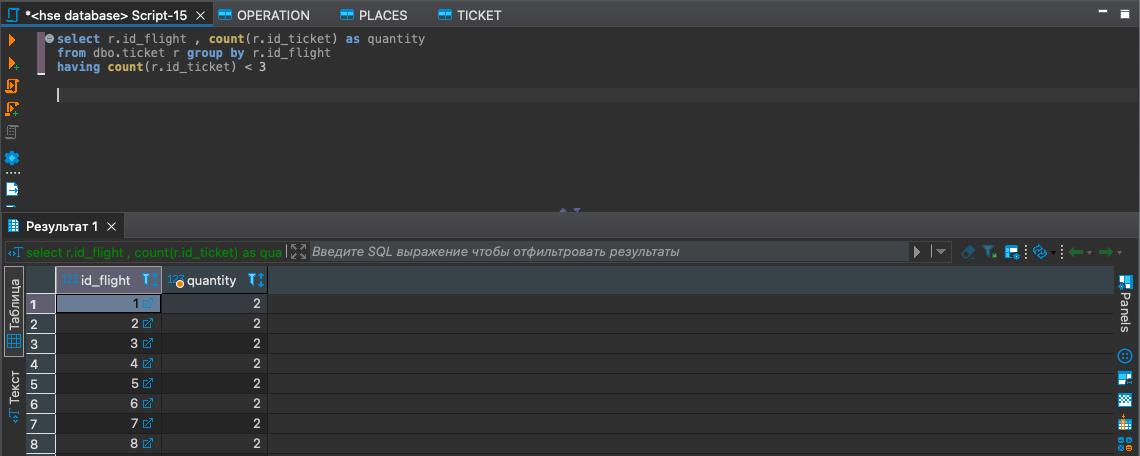


Запрос №16

1. Вывод ID рейса, если на него куплено меньше 3 билетов с указанием количества купленных билетов (запрос с HAVING и агрегированием):

select r.id\_flight , count(r.id\_ticket) as quantity from dbo.ticket r group by r.id\_flight

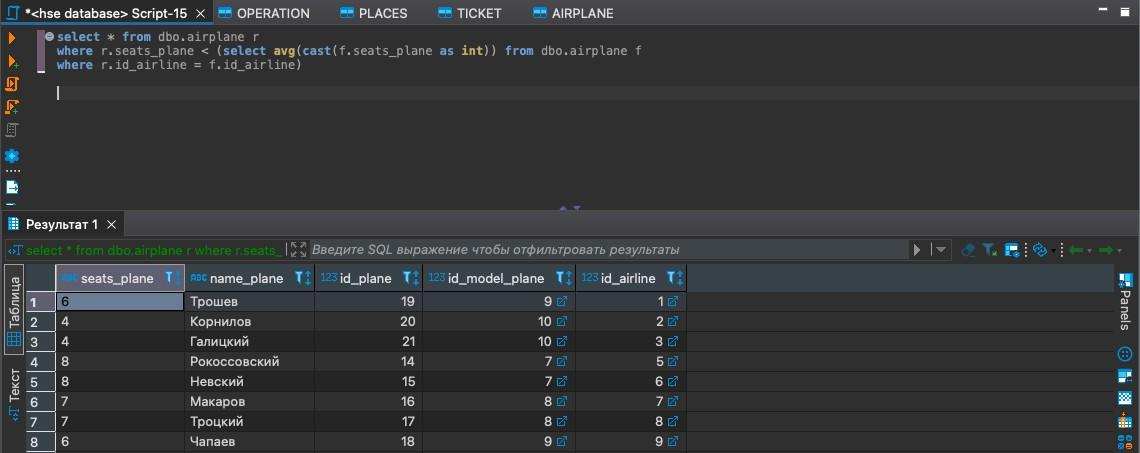
having count(r.id\_ticket) < 3

Запрос №17

1. Вывод информации о самолетах, у которых количество мест меньше среднего среди всех моделей с преобразованием типов данных (запрос с коррелированным подзапросом в WHERE):

select \* from dbo.airplane r

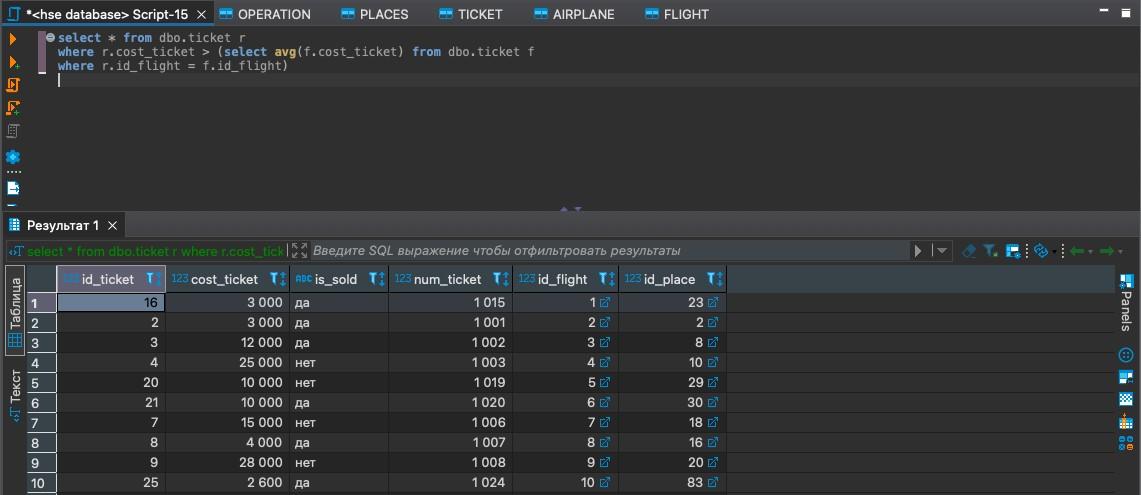
where r.seats\_plane < (select avg(cast(f.seats\_plane as int)) from dbo.airplane f where r.id\_airline = f.id\_airline)

Запрос №18

1. Вывод информации о билетах, у которых цена выше средней на конкретный рейс (запрос с коррелированным подзапросом в WHERE):

select \* from dbo.ticket r

where r.cost\_ticket > (select avg(f.cost\_ticket) from dbo.ticket f where r.id\_flight = f.id\_flight)

Запрос №19

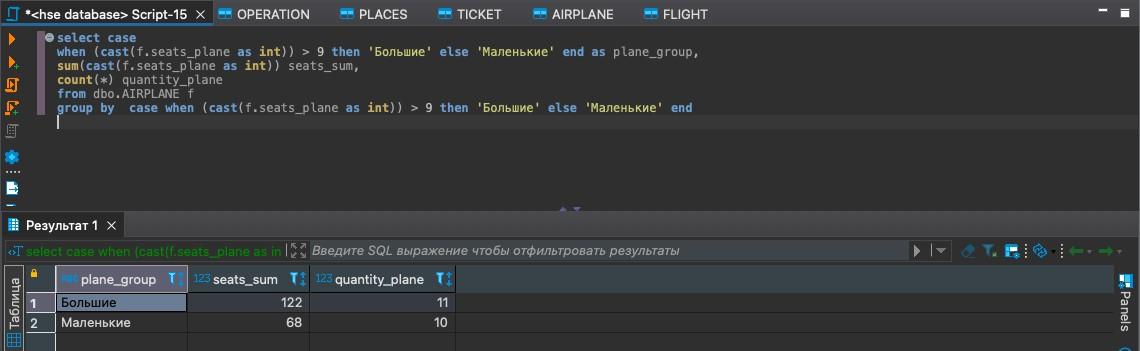
1. Запрос разделяет самолеты на 2 группы: те, у кого больше 9 мест в салоне - большие, а остальные маленькие, выводит информацию о количестве самолетов в этих группах, а также сумму мест в каждой группе (запрос с CASE (IIF) и агрегированием):

select case

when (cast(f.seats\_plane as int)) > 9 then 'Большие' else 'Маленькие' end as plane\_group, sum(cast(f.seats\_plane as int)) seats\_sum,

count(\*) quantity\_plane from dbo.AIRPLANE f

group by case when (cast(f.seats\_plane as int)) > 9 then 'Большие' else 'Маленькие' end

Запрос №20

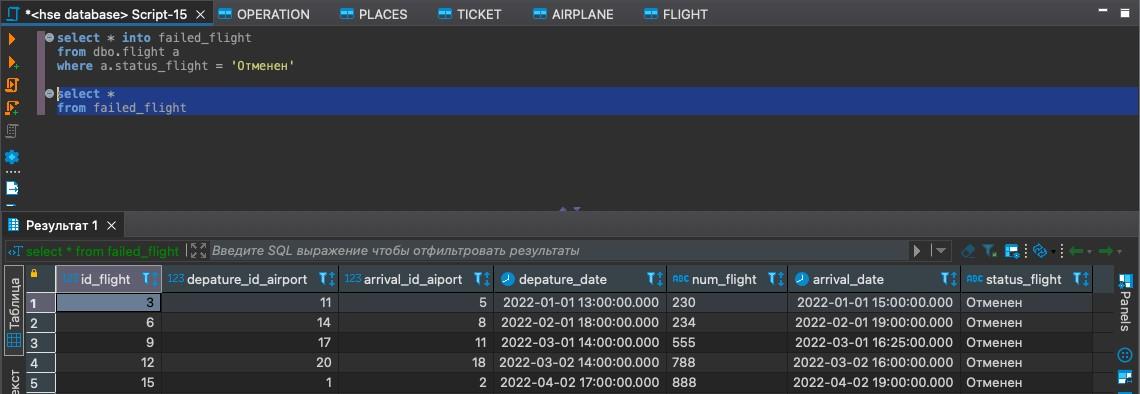
1. Запрос создает таблицу failed\_flight и вносит в нее отмененные рейсы из таблицы FLIGHT, сохраняя колонки и их типы данных (запрос SELECT INTO для подготовки выгрузки):

select \* into failed\_flight from dbo.flight a

where a.status\_flight = 'Отменен'

select \*

from failed\_flight

Запрос №21

# Разработка триггера

Для оптимизации процесса хранения данных был создан триггер, который при удалении полета в таблице FLIGHT удаляет все билеты на данный полет в таблице TICKET:

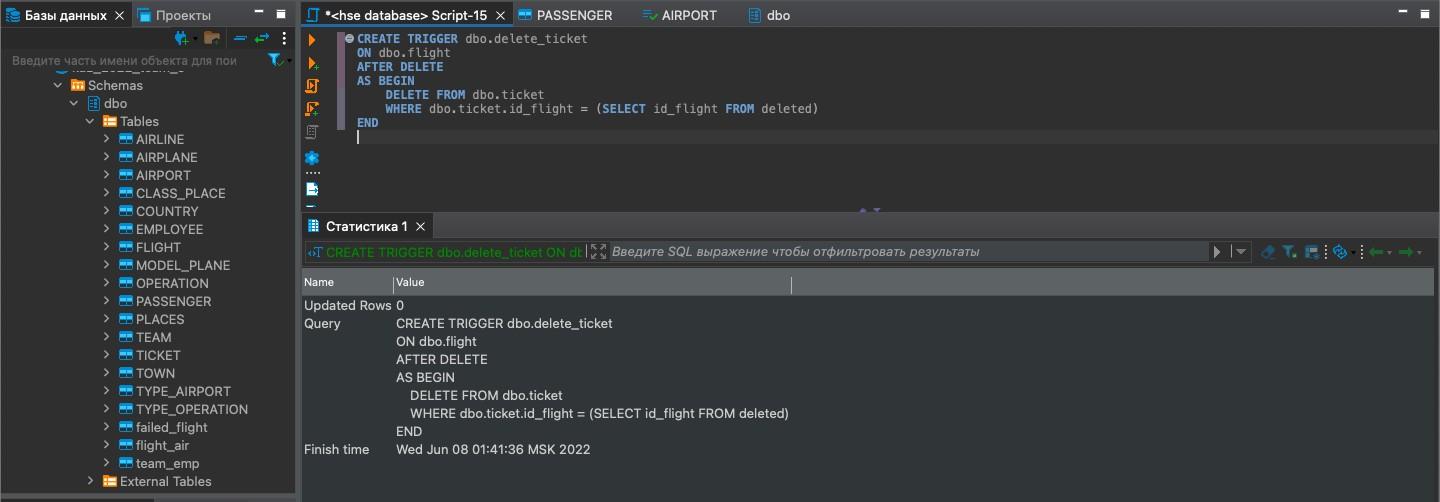
CREATE TRIGGER dbo.delete\_ticket ON dbo.flight

AFTER DELETE AS BEGIN

DELETE FROM dbo.ticket

WHERE dbo.ticket.id\_flight = (SELECT id\_flight FROM deleted)

END

Демонстрация триггера

# Разработка функций

1. Функция рассчитывает прибыль, полученную за продажу всех билетов:

CREATE FUNCTION dbo.profit() RETURNS money

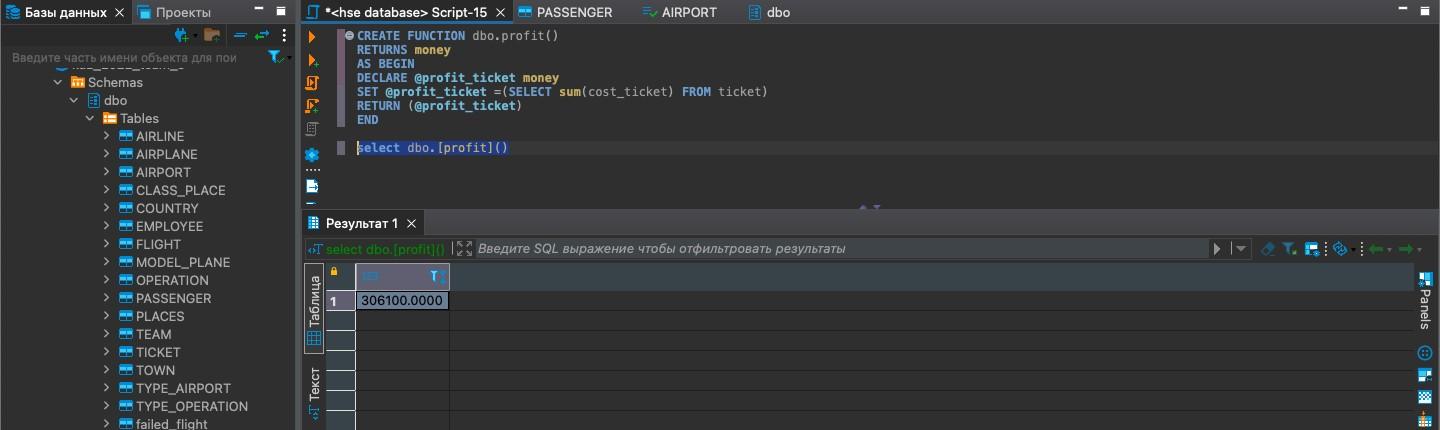
AS BEGIN

DECLARE @profit\_ticket money

SET @profit\_ticket =(SELECT sum(cost\_ticket) FROM ticket) RETURN (@profit\_ticket)

END

select dbo.[profit]()



Демонстрация функции №1

1. Функция определяет количество полетов, которые имеют статус “В ожидании”: CREATE FUNCTION dbo.amount\_flight()

RETURNS int AS BEGIN

DECLARE @amt\_flight int

SET @amt\_flight =(SELECT COUNT(\*) FROM flight Where status\_flight='В ожидании')

RETURN @amt\_flight END

select dbo.[amount\_flight]()

Демонстрация функции №2

# Разработка представлений

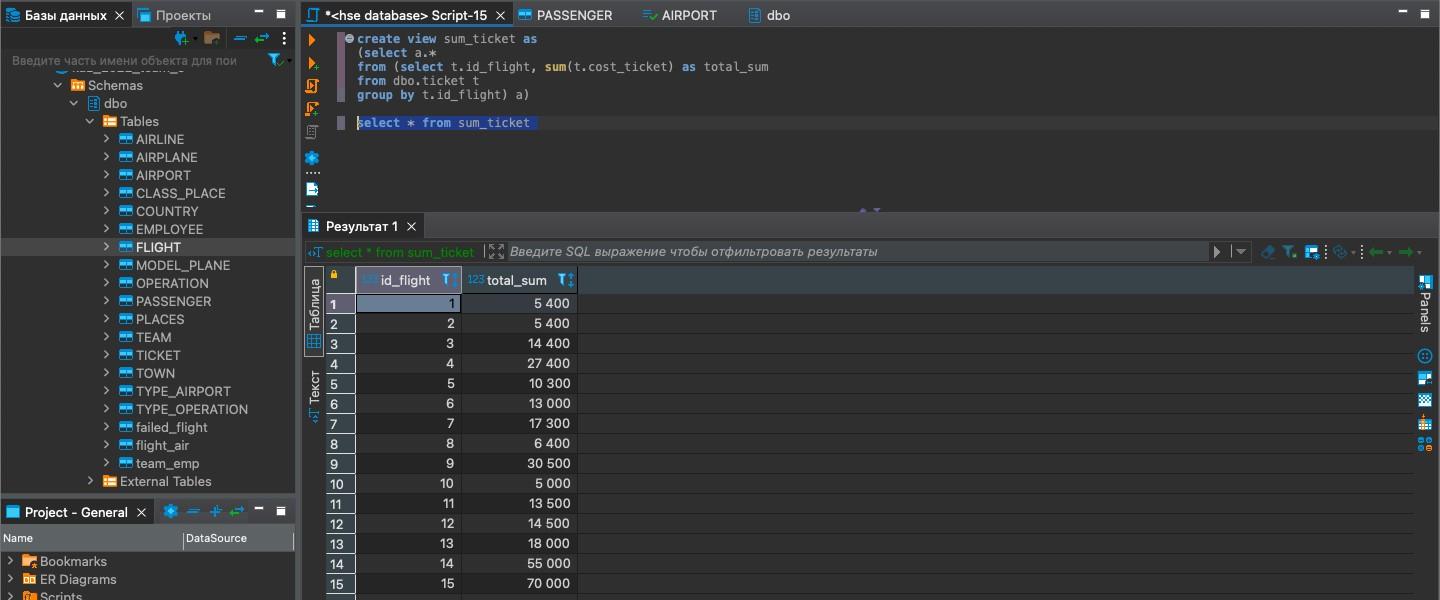
1. Представление, показывающее сумму проданных билетов по каждому из полетов:

create view sum\_ticket as (select a.\*

from (select t.id\_flight, sum(t.cost\_ticket) as total\_sum from dbo.ticket t

group by t.id\_flight) a)

select \* from sum\_ticket

Демонстрация представления №1

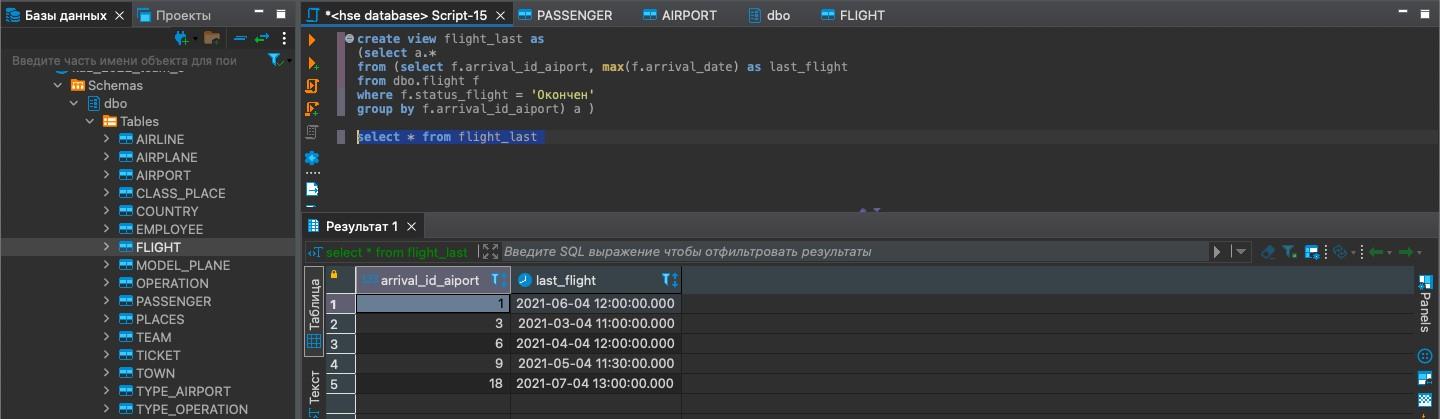
1. Представление, показывающее последний прилёт в каждый аэропорт на данный момент:

create view flight\_last as (select a.\*

from (select f.arrival\_id\_aiport, max(f.arrival\_date) as last\_flight from dbo.flight f

where f.status\_flight = 'Окончен' group by f.arrival\_id\_aiport) a )

select \* from flight\_last



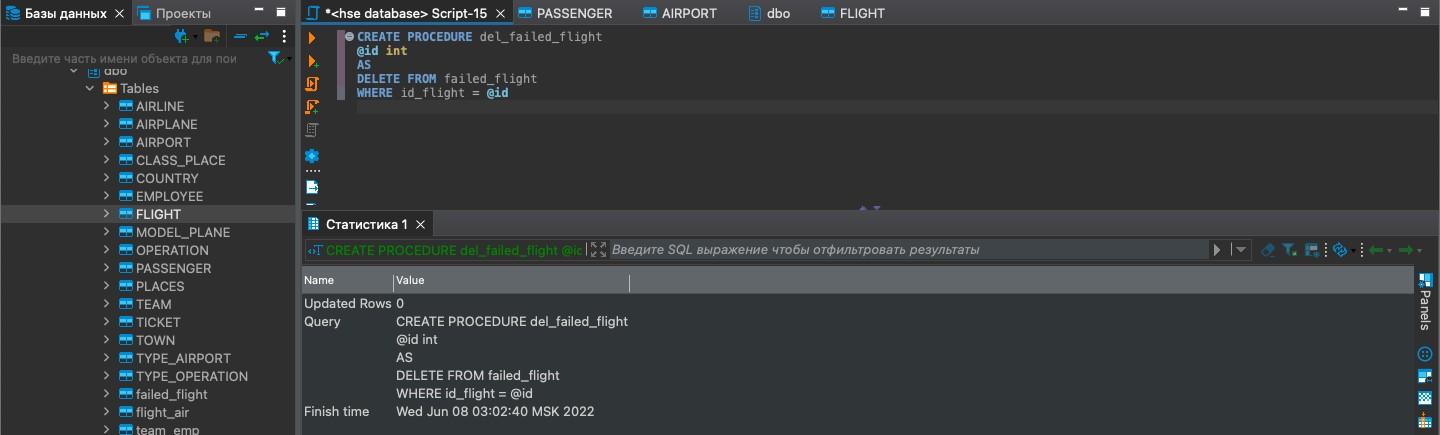
Демонстрация представления №2

# Разработка процедур

1)Процедура, удаляющая данные об отмененном полете:

CREATE PROCEDURE del\_failed\_flight @id int

AS

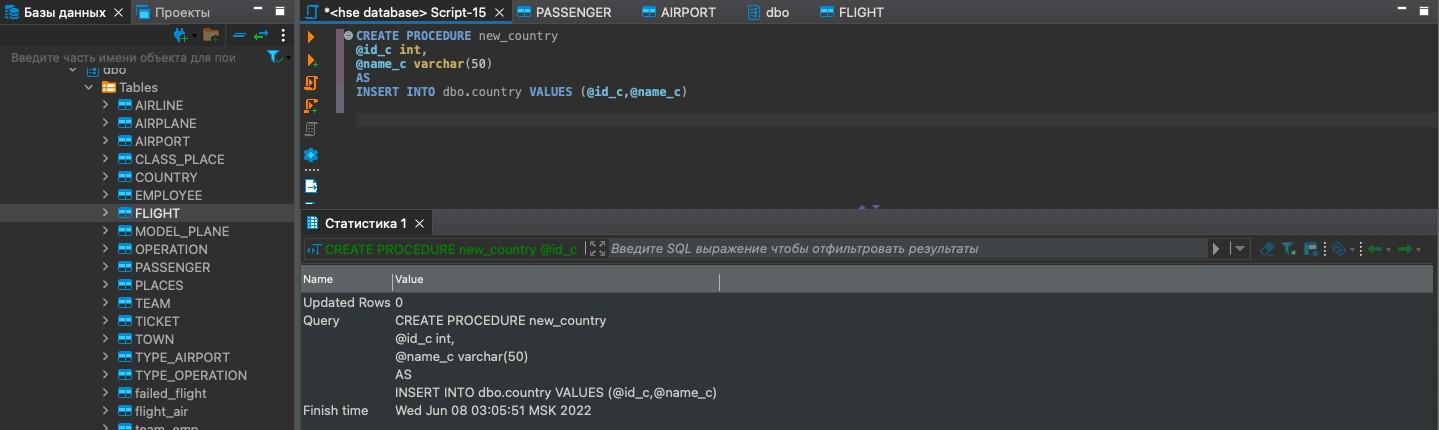
DELETE FROM failed\_flight WHERE id\_flight = @id

Демонстрация процедуры №1 2)Процедура, добавляющая новую страну в таблицу COUNTRY:

CREATE PROCEDURE new\_country @id\_c int,

@name\_c varchar(50) AS

INSERT INTO dbo.country VALUES (@id\_c,@name\_c)



Демонстрация процедуры №2

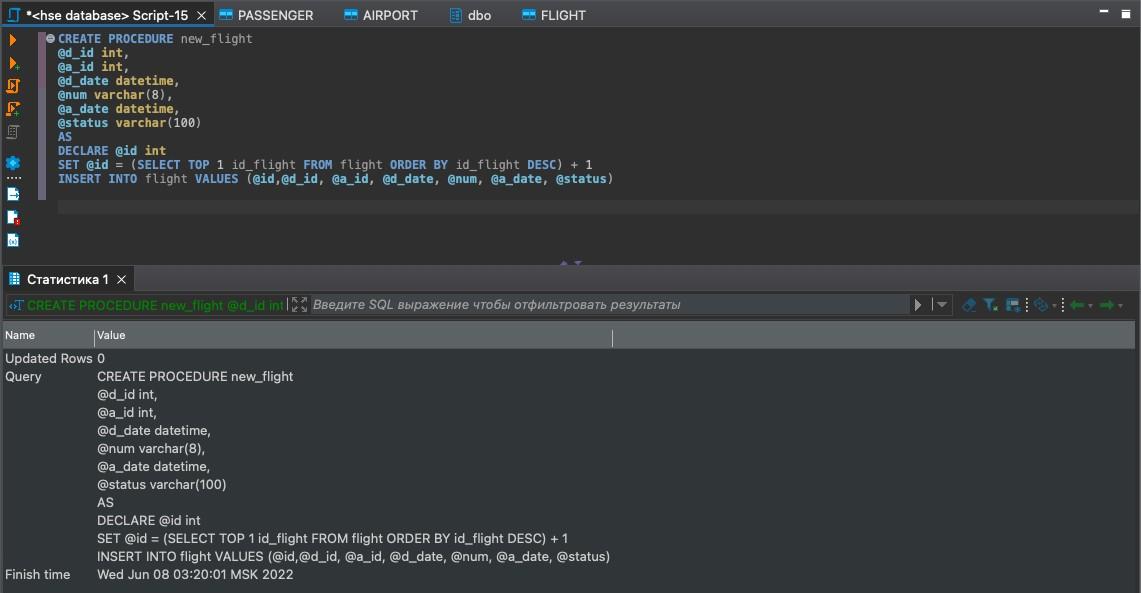
1. Процедура, добавляющая новый полет в таблицу FLIGHT:

CREATE PROCEDURE new\_flight @d\_id int,

@a\_id int, @d\_date datetime, @num varchar(8), @a\_date datetime,

@status varchar(100) AS

DECLARE @id int

SET @id = (SELECT TOP 1 id\_flight FROM flight ORDER BY id\_flight DESC) + 1 INSERT INTO flight VALUES (@id,@d\_id, @a\_id, @d\_date, @num, @a\_date, @status)

Демонстрация процедуры №3

1. Процедура, обновляющая тип аэропорта:

CREATE PROCEDURE upd\_type\_airport @air\_id int,

@t\_id int AS

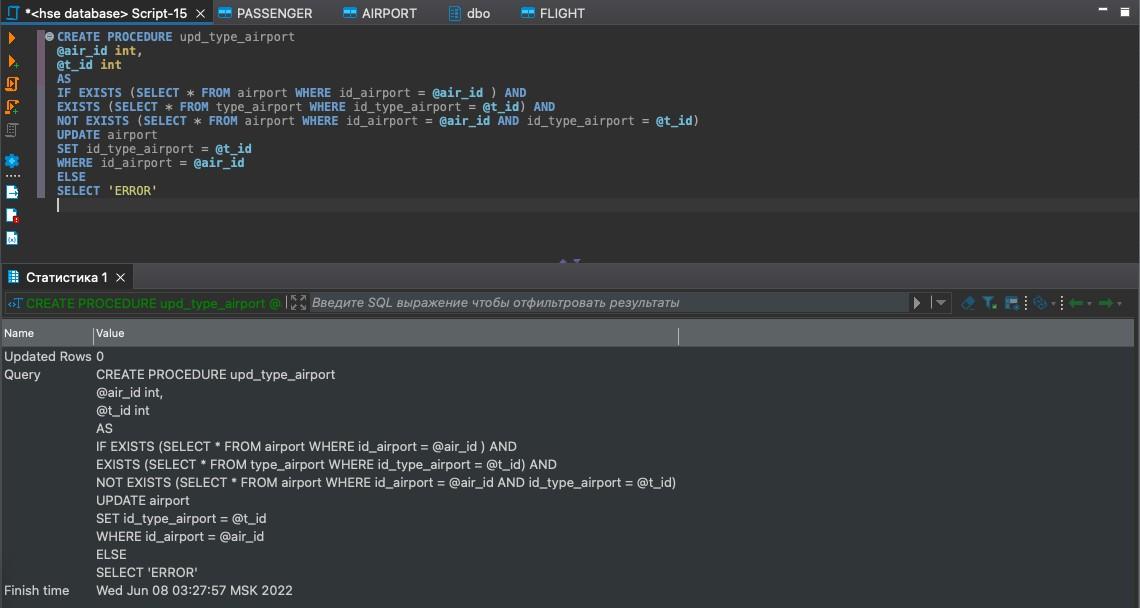
IF EXISTS (SELECT \* FROM airport WHERE id\_airport = @air\_id ) AND EXISTS (SELECT \* FROM type\_airport WHERE id\_type\_airport = @t\_id) AND

NOT EXISTS (SELECT \* FROM airport WHERE id\_airport = @air\_id AND id\_type\_airport = @t\_id)

UPDATE airport

SET id\_type\_airport = @t\_id WHERE id\_airport = @air\_id ELSE

SELECT 'ERROR'

Демонстрация процедуры №4

1. Процедура, добавляющая пассажира и его данные в таблицу PASSENGER:

CREATE PROCEDURE add\_pas @l varchar(100),

@f varchar(100), @email varchar(100), @birth date,

@sex varchar(8) AS

BEGIN BEGIN TRY

BEGIN TRAN

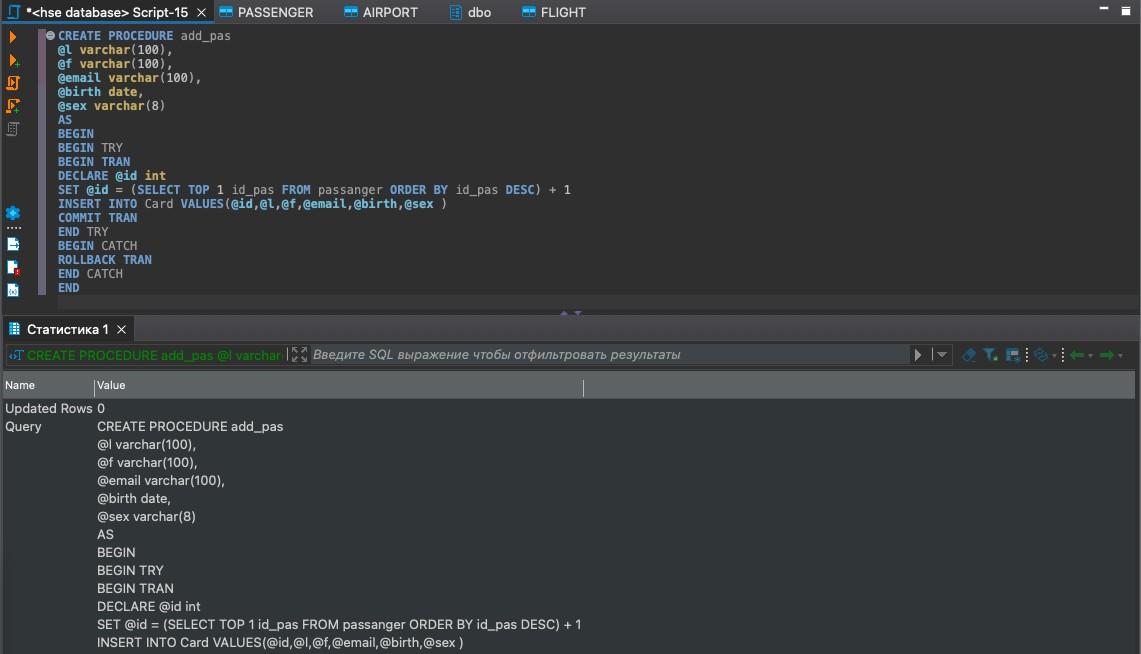
DECLARE @id int

SET @id = (SELECT TOP 1 id\_pas FROM passanger ORDER BY id\_pas DESC) + 1 INSERT INTO Card VALUES(@id,@l,@f,@email,@birth,@sex )

COMMIT TRAN END TRY BEGIN CATCH

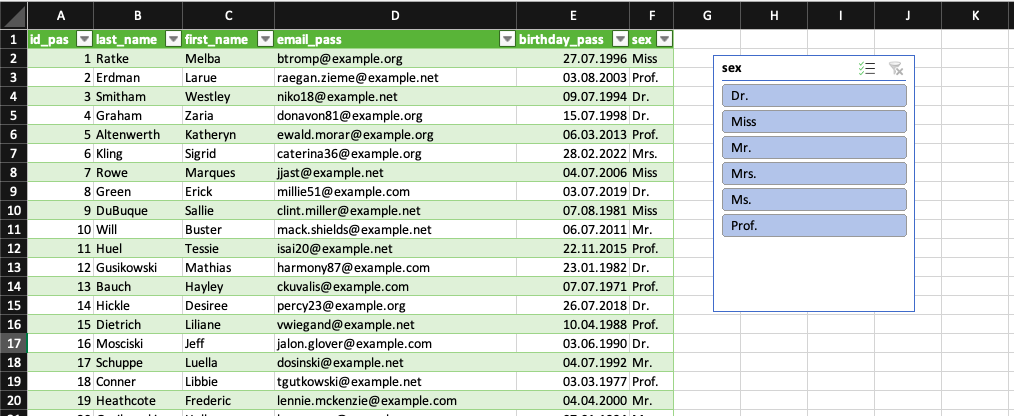
ROLLBACK TRAN END CATCH

END

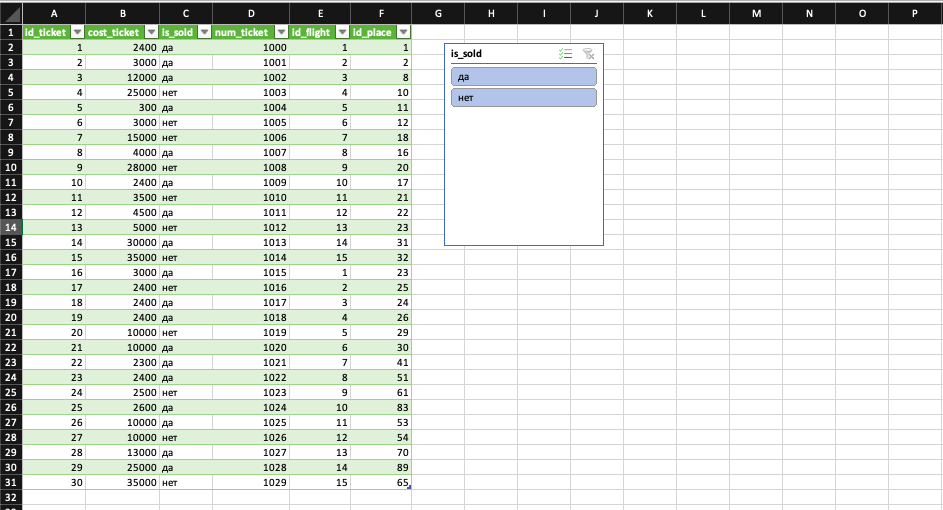
Демонстрация процедуры №5

# Информационная панель в Excel

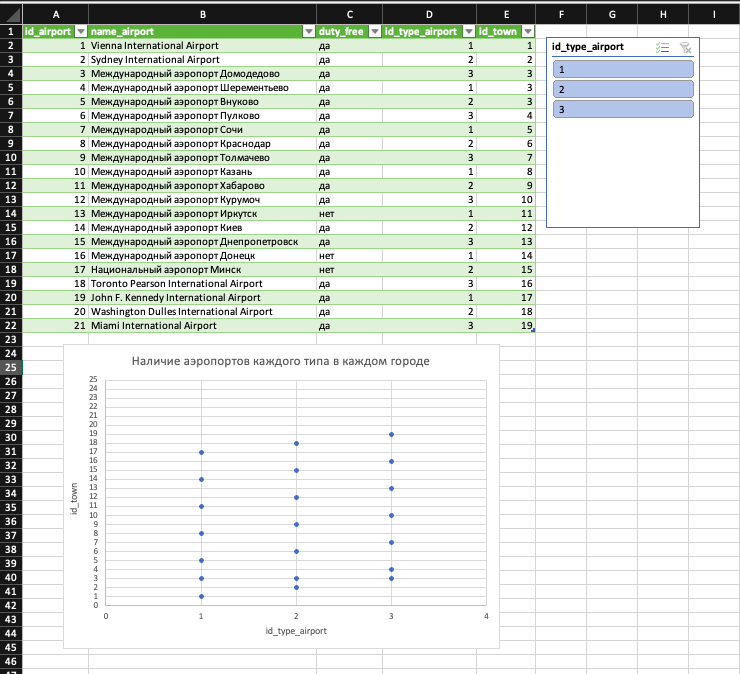
Для создания информационной панели было решено выгрузить несколько таблиц с заполненными данными в Excel. Мы построили срез для таблицы PASSENGER, который может показать по полу/статусу информацию о пассажирах:



Срез для таблицы PASSENGER

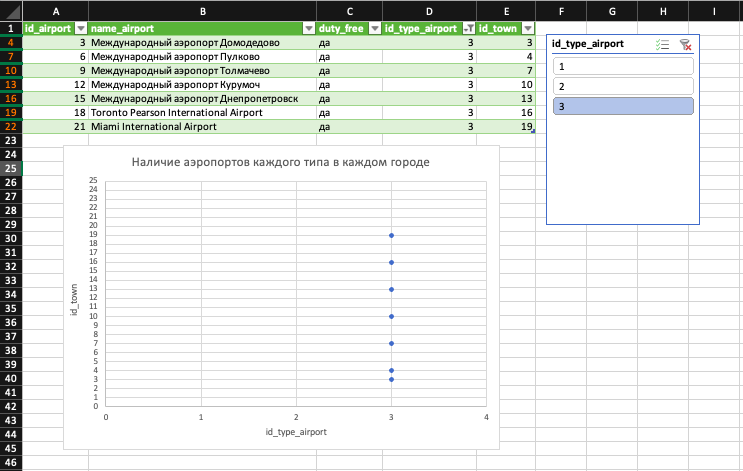
Также построили срез для таблицы TICKET, который быстро может показать информацию о проданных и непроданных билетах:

Срез для таблицы TICKET

После мы построили срез для таблицы AIRPORT, который выбирает аэропорты каждого типа, и построили точечный график, который показывает наличие аэропортов каждого типа в каждом из городов:

Срез и точечный график для таблицы AIRPORT

Также, при выборе определенного типа аэропорта в срезе, график обновляется:



Пример обновления графика

# Отчеты, визуализация

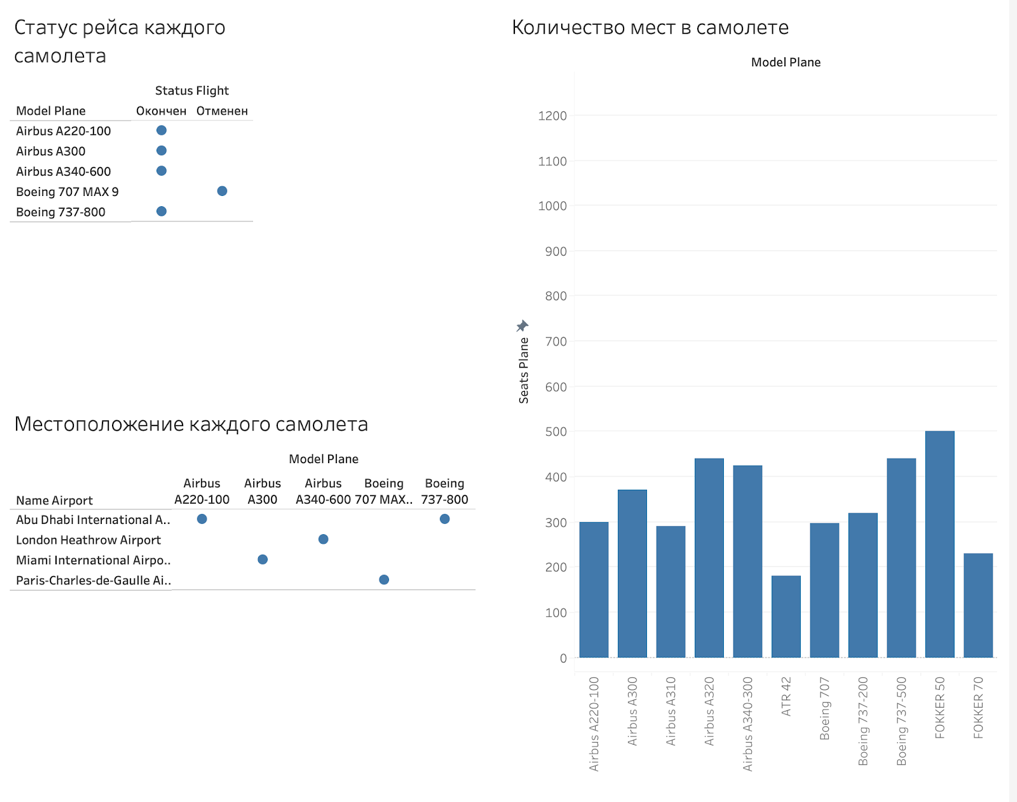
Первый отчет:

В первом графике отображаются модели самолетов и максимальное количество мест в нем.

Во втором графике мы видим модель самолета и статус рейса. Был ли он выполнен или отменен. То есть простаивает ли он в определенном аэропорту или меняет свое местоположение

В последнем графике наглядно показано, в каком аэропорту какой самолет находится. Для нас это очень важно с той точки зрения, чтобы выстраивать последующие рейсы с этими моделями самолетов.

В данном отчете мы хотели демонстрируем, как связаны самолеты, статус рейса этих самолетов (отменен или выполнен), а также где он находится в данный момент.



Отчет 1

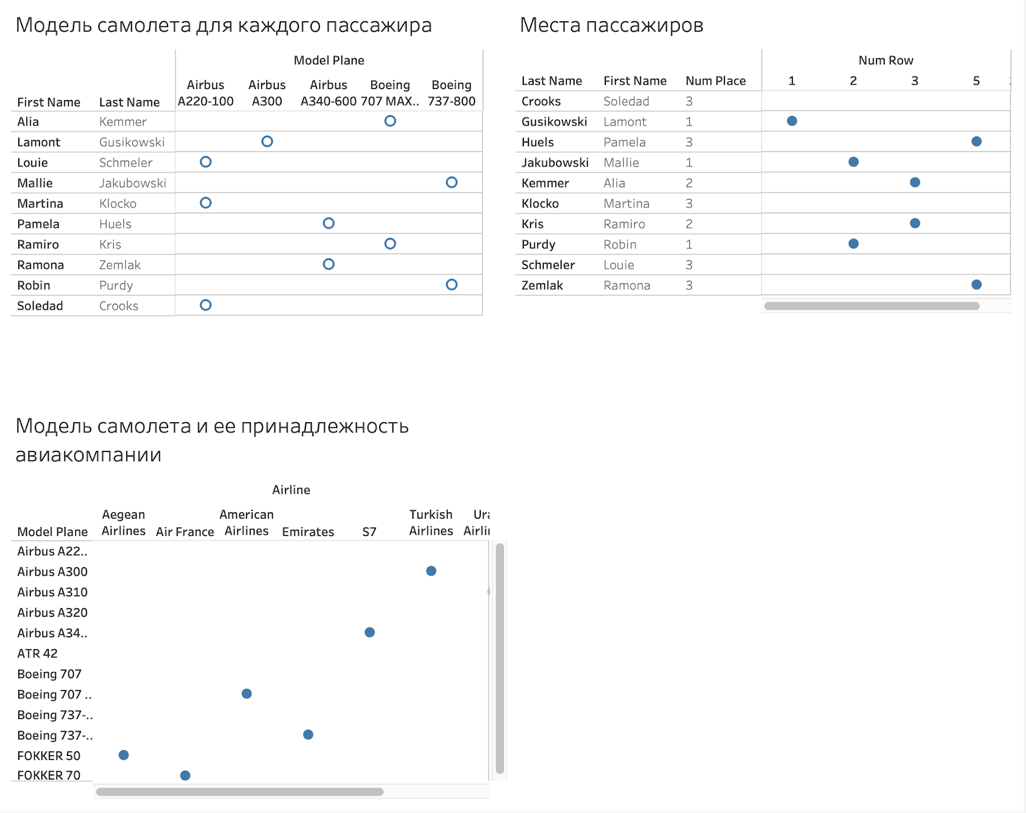
Второй отчет:

Во втором отчете мы можем наблюдать пассажиров, а конкретнее, имя и фамилию, купленные места в самолете и выбранную авиакомпанию.

В первой таблице выведены имя и фамилия пассажиров, купленное место, а точнее ряд и место (сказать, что в ряду 3 места).

На второй таблице можно увидеть, в какой модели самолета было приобретено место.

На третьей таблице изображено, какой авиакомпанией летит пассажир и соответственно какой авиакомпании принадлежит той или иной самолет.



Отчет 2

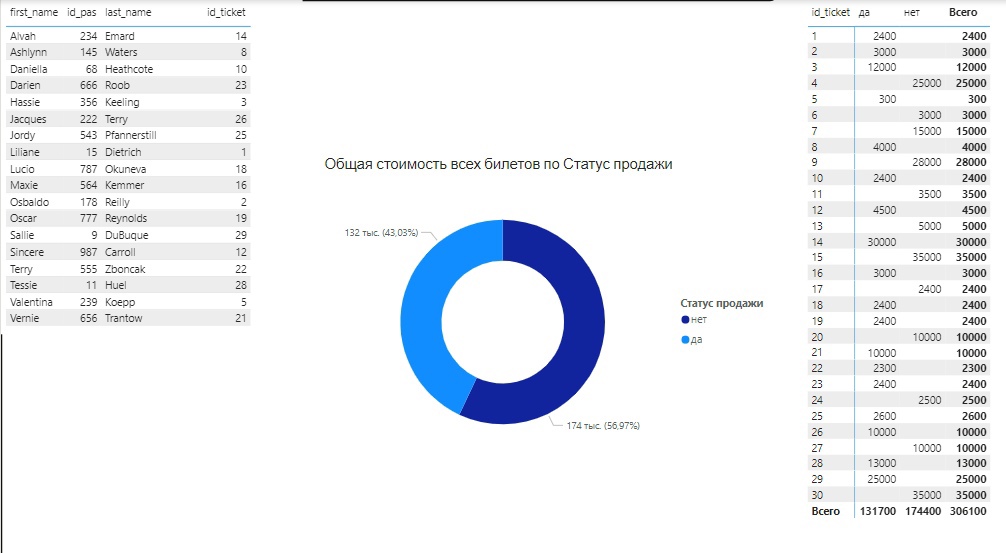
Третий отчет:

В данном отчете представлены данные о билетах и их статусе продаж.

В первой таблице показаны имена и фамилии пассажиров, а также идентификаторы билетов, которые они приобрели. Это позволяет нам отслеживать, какие билеты были куплены каждым пассажиром.

На круговой диаграмме отображена общая стоимость всех билетов, разделенная по статусу продажи. Мы можем видеть, какая часть билетов была продана (статус "да") и какая часть осталась непроданной (статус "нет"). Диаграмма демонстрирует, что 56.97% билетов были проданы, а 43.03% остались непроданными, что позволяет нам оценить эффективность продаж.

В правой таблице представлена детализированная информация по каждому билету, включая их идентификаторы и стоимость. Таблица также разделена на две колонки, указывающие, был ли билет продан или нет, и общую стоимость билетов в каждой категории.

****

Отчет 3

# Описание роли каждого участника проекта

Описание предметной области:

1)Сведения о предметной области - Мухлаев Алан Арслангович

2)Цель создания БД - Мухлаев Алан Арслангович

3)Пользователи БД - Горяев Даян Мергенович

4)Список основных сущностей - Озтемел Теоман Сердарович

5)Описание таблиц - Озтемел Теоман Сердарович

6)Алгоритмы обработки данных – Горяев Даян Мергенович

7)Перечень отчетных форм - Патаев Арслан Зольванович

8)Архитектура программного продукта - Озтемел Теоман Сердарович

Проектирование:

1)Инфологическая модель - Горяев Даян Мергенович

2)Даталогическая модель - Горяев Даян Мергенович

3)Размещение БД в СУБД - Патаев Арслан Зольванович

4)Заполнение БД данными - Мухлаев Алан Арслангович

Разработка:

1)Настройка индексов – Мухлаев Алан Арслангович

2)Разработка запросов - Патаев Арслан Зольванович

3)Разработка триггера - Патаев Арслан Зольванович

4)Разработка функций - Озтемел Теоман Сердарович

5)Разработка представлений- Горяев Даян Мергенович

6)Разработка процедур - Мухлаев Алан Арслангович

Отчеты:

1)Отчет Excel – Мухлаев Алан Арслангович

2)Отчет Power BI - Горяев Даян Мергенович

3)Создание видеоролика - Патаев Арслан Зольванович, Озтемел Теоман Сердарович, Горяев Даян Мергенович, Мухлаев Алан Арслангович

В ходе выполнения проекта вся команда активно участвовала в процессе в равном соотношении. Несмотря на указание ответственного за конкретный этап работы, участие принимал полный состав команды, корректируя и совершенствуя проект в процессе. В результате все поставленные нашей командой задачи были выполнены.