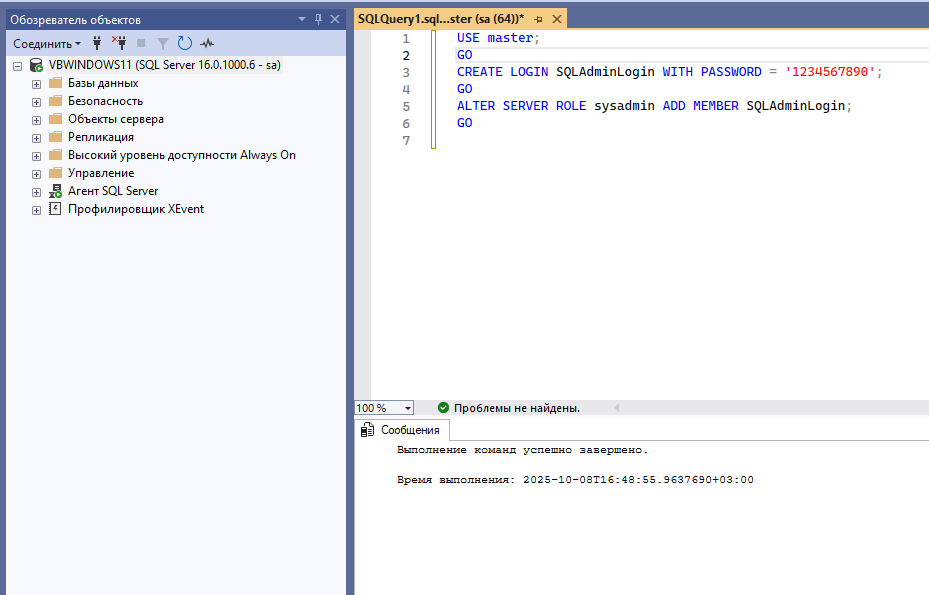
**Цель:** Создать основные учетные записи для доступа к SQL Server и определить их базовые привилегии на уровне сервера.

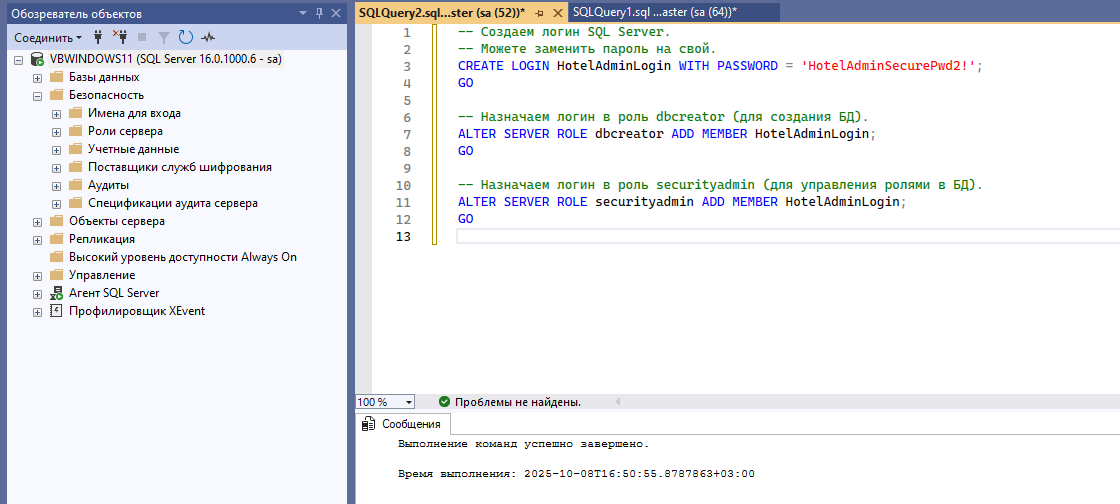
**Ход выполнения работы:**

Запустите SQL Server Management Studio (SSMS) и подключитесь к вашему экземпляру SQL Server под учетной записью **sa** (или другим административным логином).

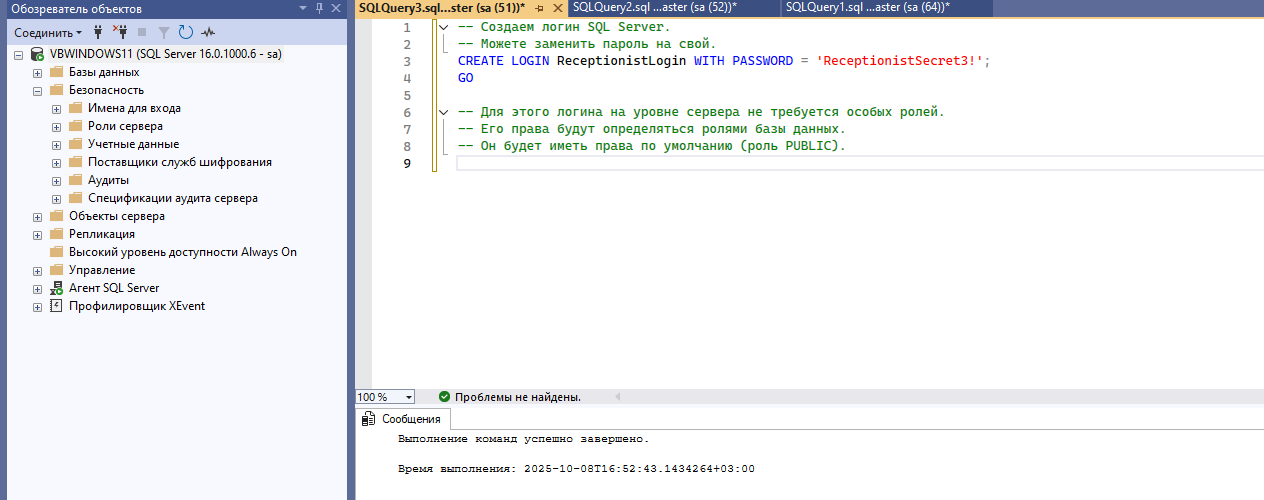
1. **Создание логина для администратора SQL Server.** Этот логин будет иметь полный контроль над сервером.



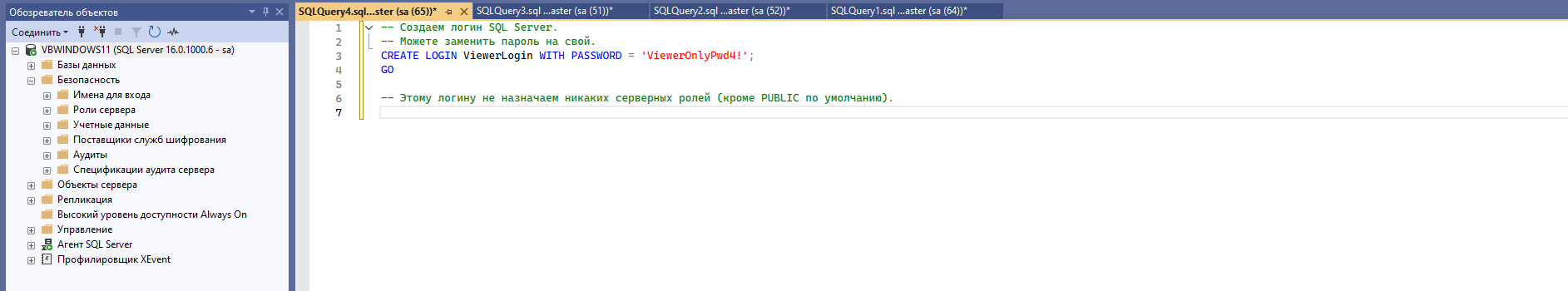
1. **Создание логина для администратора отеля.** Этот логин сможет создавать базы данных и управлять ролями, но не будет иметь полного контроля над всем сервером.



1. **Создание логина для сотрудника ресепшн.** Этот логин будет управлять основными операциями в базе данных отеля (бронирование, заезд/выезд).



1. **Создание логина для внешнего пользователя**, которому нужен только доступ к чтению. Этот логин имеет только минимальные права для входа в SQL Server.

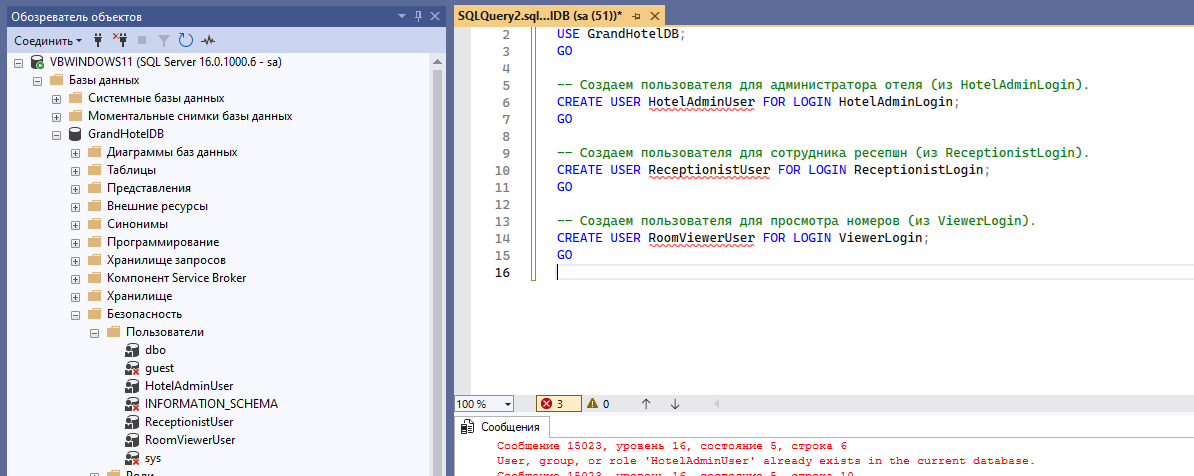


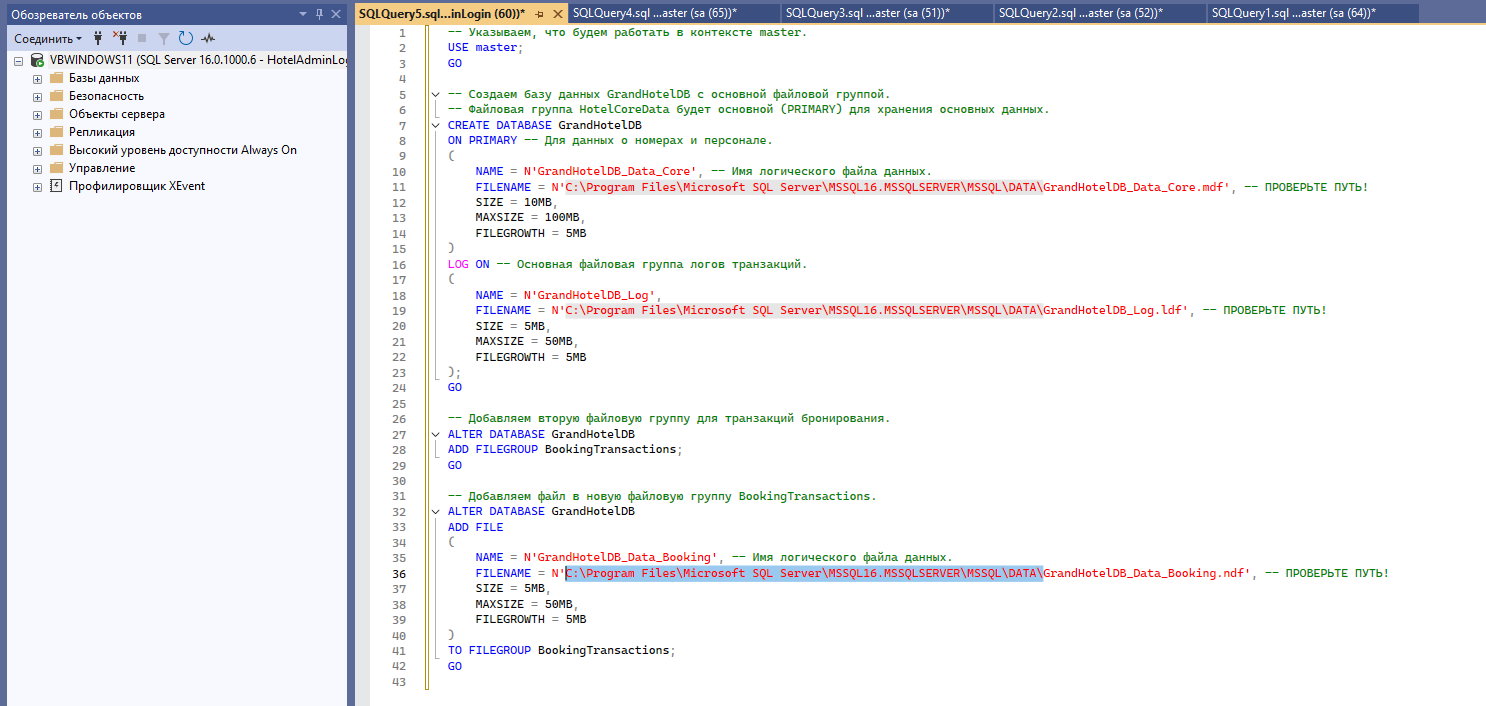
**Цель:** Создать базу данных GrandHotelDB с разделением данных для повышения производительности.

**Ход выполнения работы:**

Запустите SQL Server Management Studio (SSMS) и подключитесь к вашему экземпляру SQL Server под учетной записью **HotelAdminLogin**.

**Создание базы данных GrandHotelDB** с двумя файловыми группами главное от SqlSErverLogin



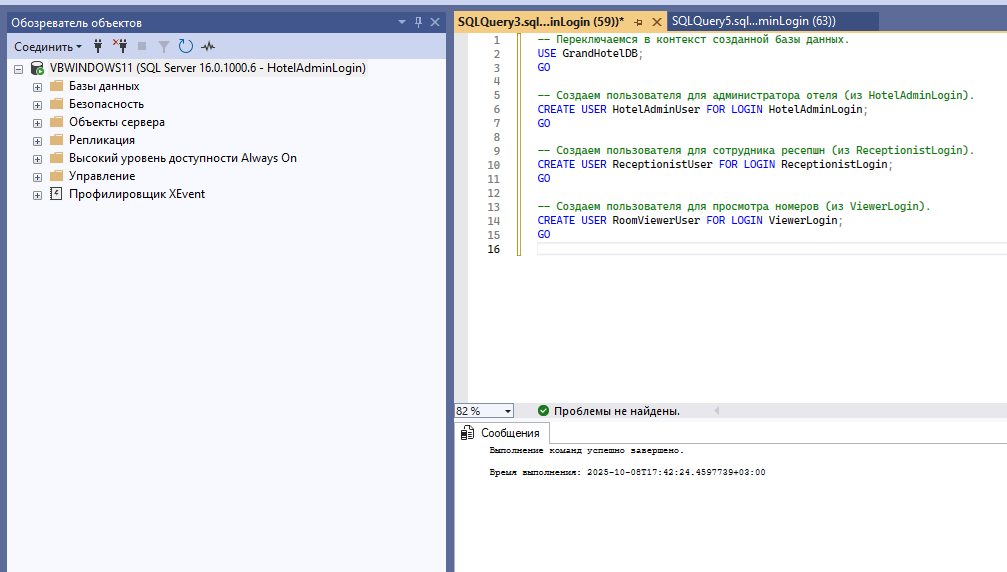


**Цель:** Логически организовать базу данных, определить структуру таблиц и их связи, назначить владельцев.

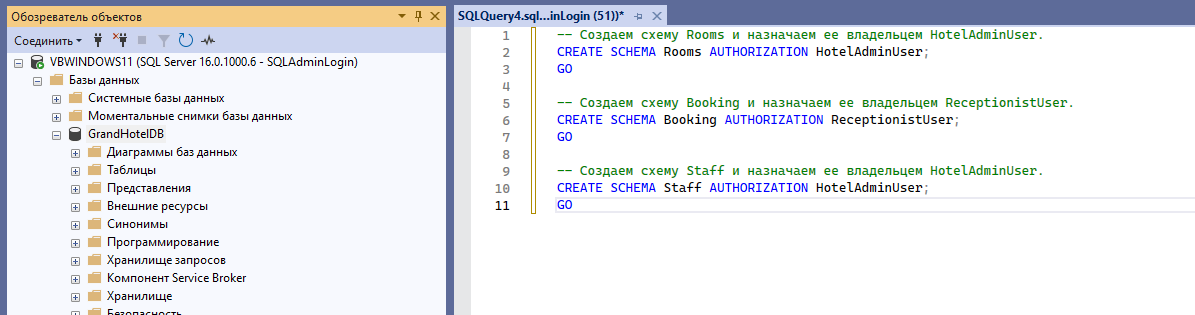
**Ход выполнения работы:**

Запустите SQL Server Management Studio (SSMS) и подключитесь к вашему экземпляру SQL Server под учетной записью **HotelAdminLogin**.

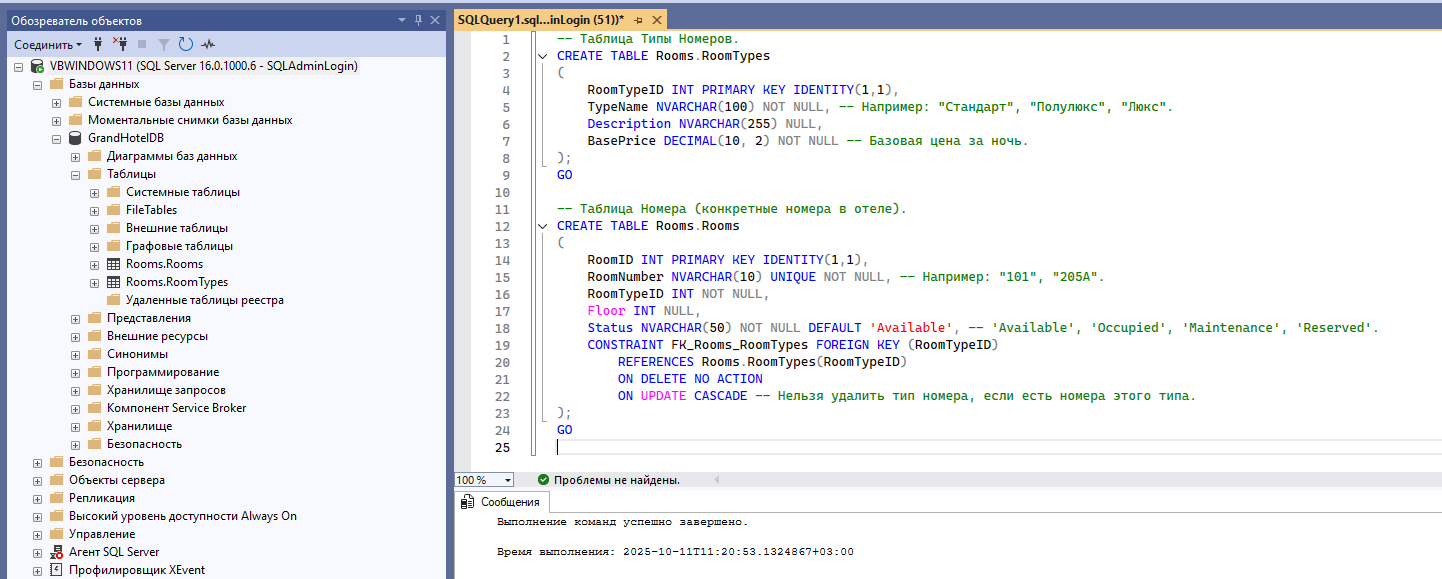
1. **Создание пользователей БД** (учетных записей внутри базы данных GrandHotelDB), которые будут иметь роли и владеть схемами.



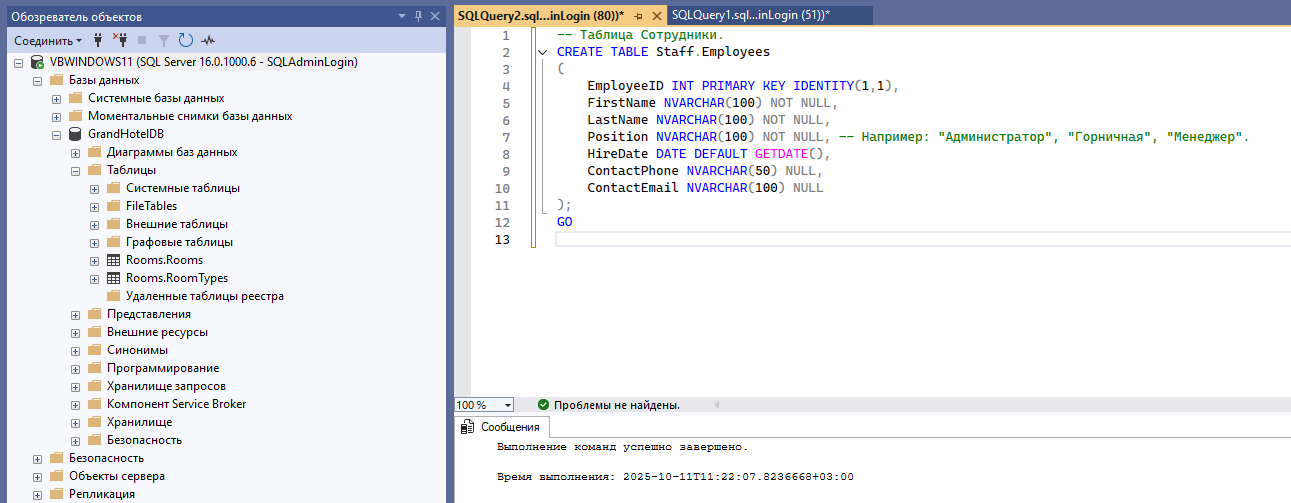
1. **Создание схем и назначение владельцев.** Схемы логически разделяют объекты базы данных. Владелец схемы имеет полный контроль над объектами в ней.



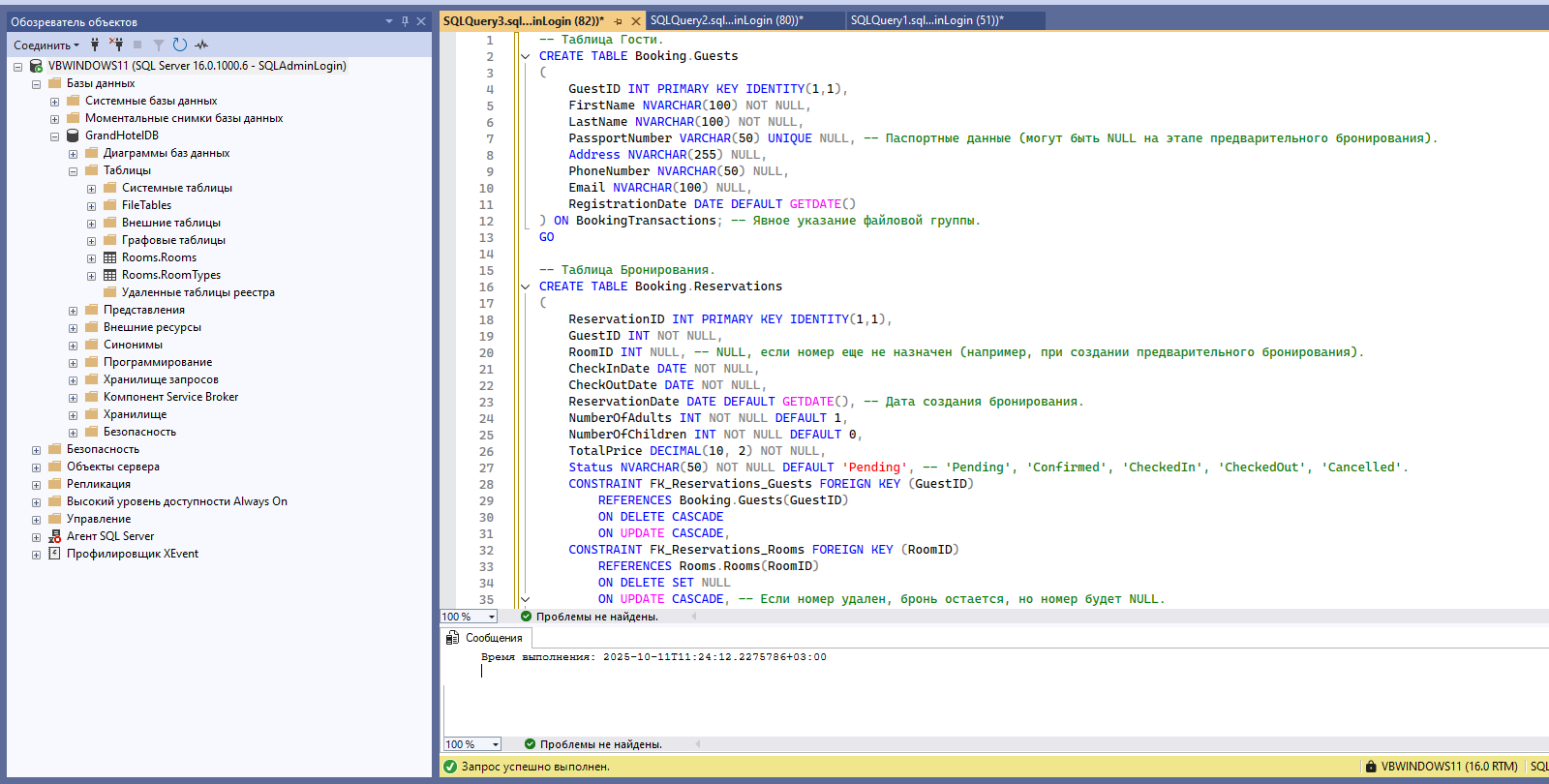
1. **Создание таблиц в схеме Rooms.** Информация о типах номеров, самих номерах и их текущем статусе.



1. **Создание таблиц в схеме Staff.** Информация о сотрудниках отеля.



1. **Создание таблиц в схеме Booking.** Информация о гостях, их бронированиях и истории проживания.

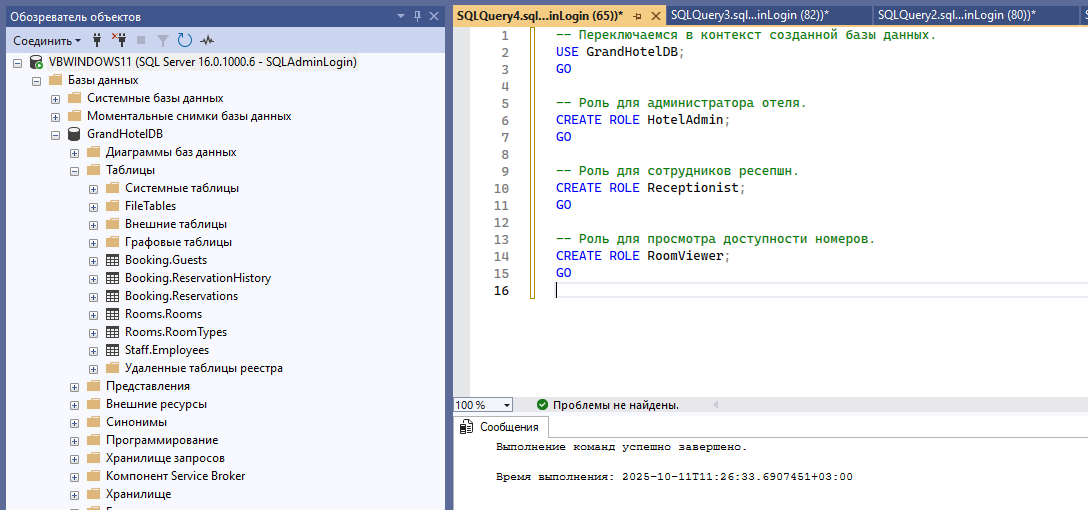


**Цель:** Определить роли внутри GrandHotelDB для разных категорий сотрудников.

**Ход выполнения работы:**

Запустите SQL Server Management Studio (SSMS) и подключитесь к вашему экземпляру SQL Server под учетной записью **HotelAdminLogin**.

1. **Создание ролей**, которым будут присваиваться пользователи.

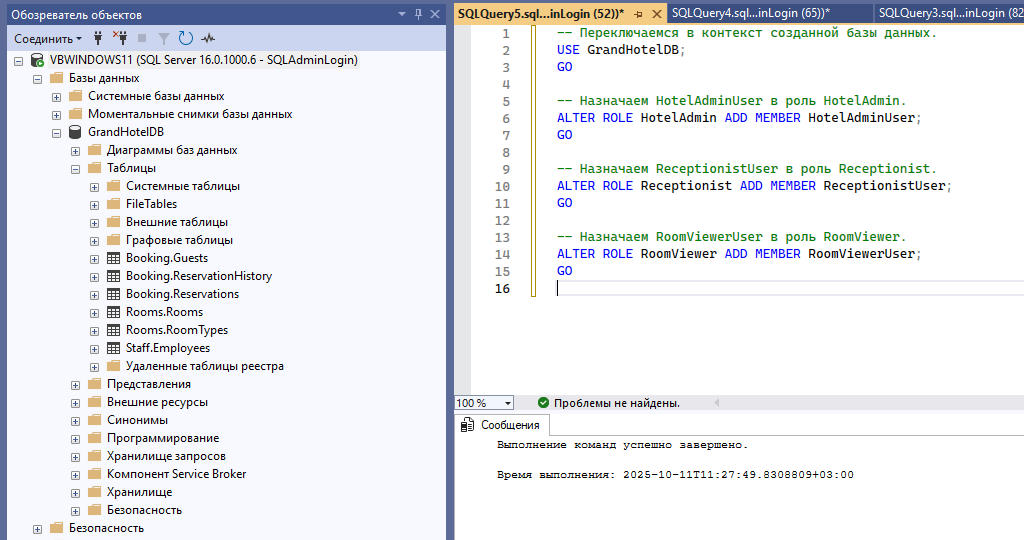


**Цель:** Привязать учетные записи пользователей к ролям БД и назначить пользователей в эти роли.

**Ход выполнения работы:**

Запустите SQL Server Management Studio (SSMS) и подключитесь к вашему экземпляру SQL Server под учетной записью **HotelAdminLogin**.

1. **Назначение ролей пользователям.** Присваиваем созданные роли соответствующим пользователям.

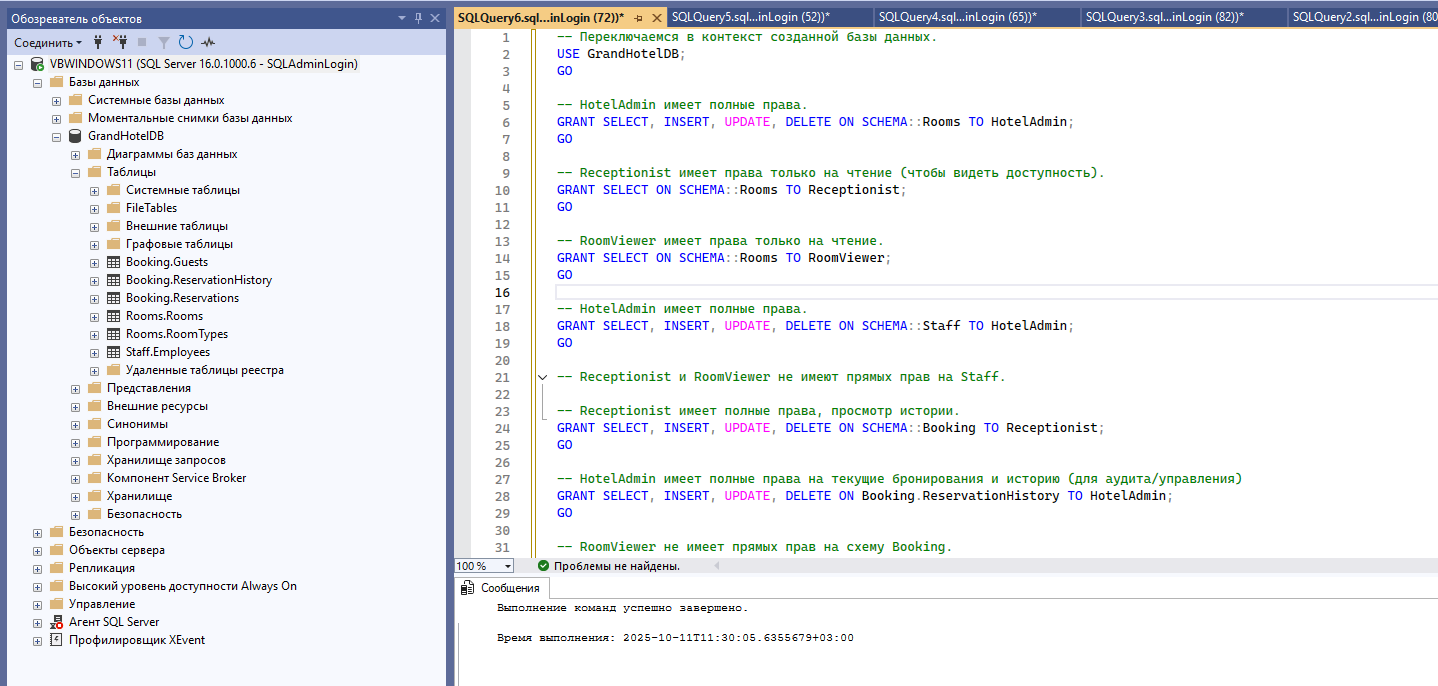


**Цель:** Детализировать права доступа для ролей, создать представление для RoomViewer и наполнить базу данных начальными данными.

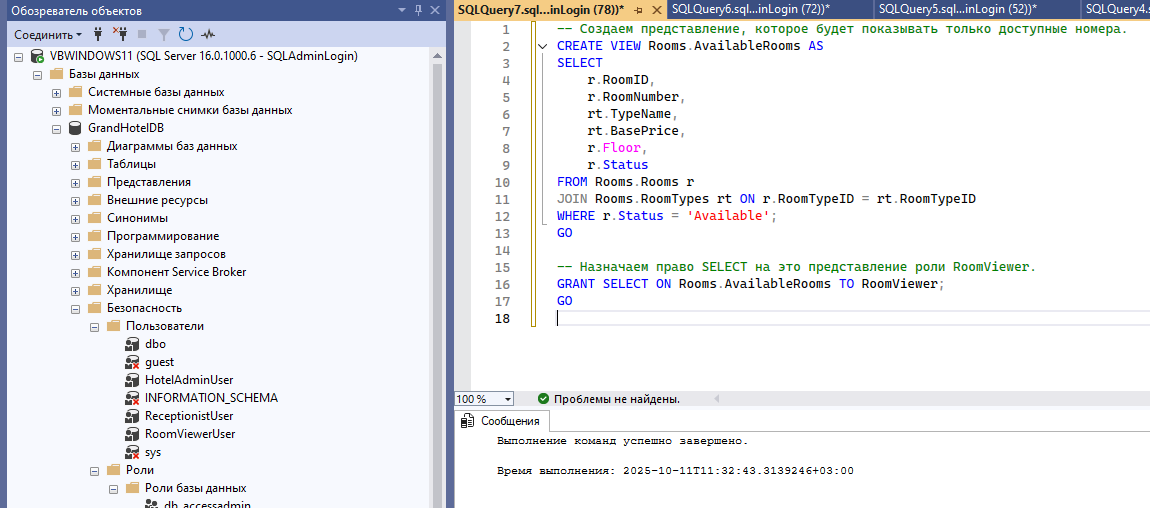
**Ход выполнения работы:**

Запустите SQL Server Management Studio (SSMS) и подключитесь к вашему экземпляру SQL Server под учетной записью **HotelAdminLogin**.

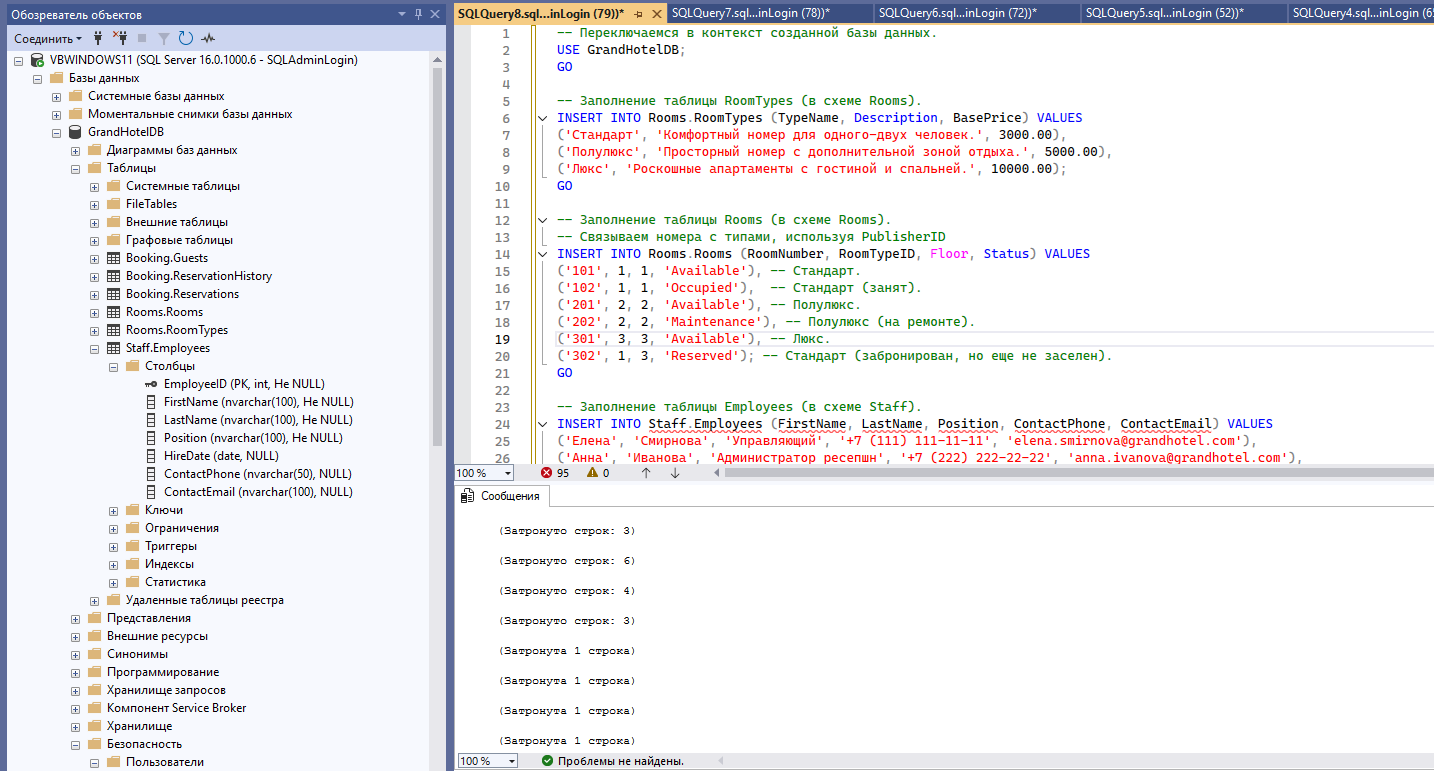
1. **Назначение привилегий.** Предоставляем права ролям на объекты в схемах.



1. **Создание представления для роли RoomViewer.** Это представление будет показывать доступные номера, чтобы RoomViewer мог их просматривать.



1. **Заполнение таблиц данными.** Добавляем тестовые данные для проверки работы системы.



**Полученная структура:**

**SQL Server Logins (серверный уровень):**

- SQLAdminLogin (sysadmin);

- HotelAdminLogin;

- ReceptionistLogin;

- ViewerLogin.

**Database (уровень БД):** GrandHotelDB.

**Filegroups (физическое хранение):**

- HotelCoreData (PRIMARY) - содержит файлы .mdf для Rooms, Staff;

- BookingTransactions - содержит файлы .ndf для Booking.

**Schemas (логическая организация):**

- Rooms (владелец: HotelAdminUser) - содержит объекты, которые будут размещены на файловой группе HotelCoreData;

- Staff (владелец: HotelAdminUser) - содержит объекты, которые будут размещены на файловой группе HotelCoreData;

- Booking (владелец: ReceptionistUser) - содержит объекты, которые будут размещены на файловой группе BookingTransactions.

**Database Users:**

- HotelAdminUser (связан с HotelAdminLogin);

- ReceptionistUser (связан с ReceptionistLogin);

- RoomViewerUser (связан с ViewerLogin).

**Database Roles:**

- HotelAdmin (в него входит HotelAdminUser);

- Receptionist (в него входит ReceptionistUser);

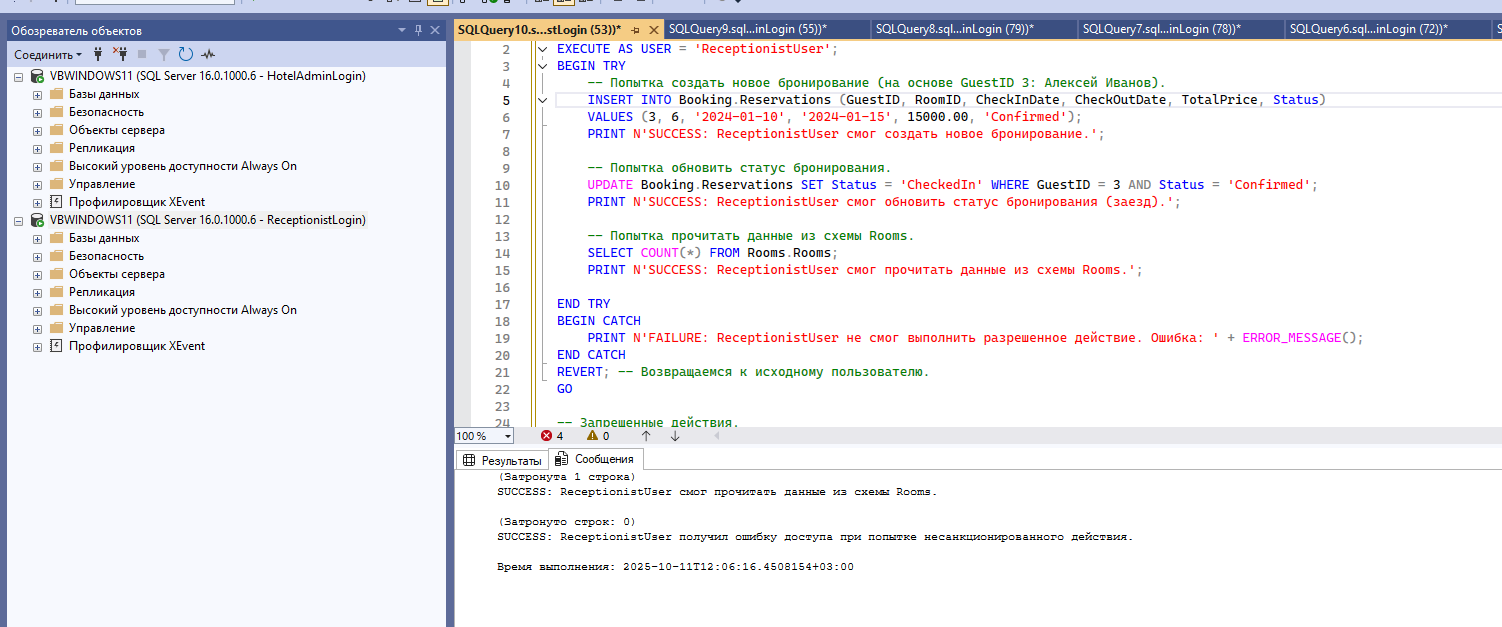
- RoomViewer (в него входит RoomViewerUser).

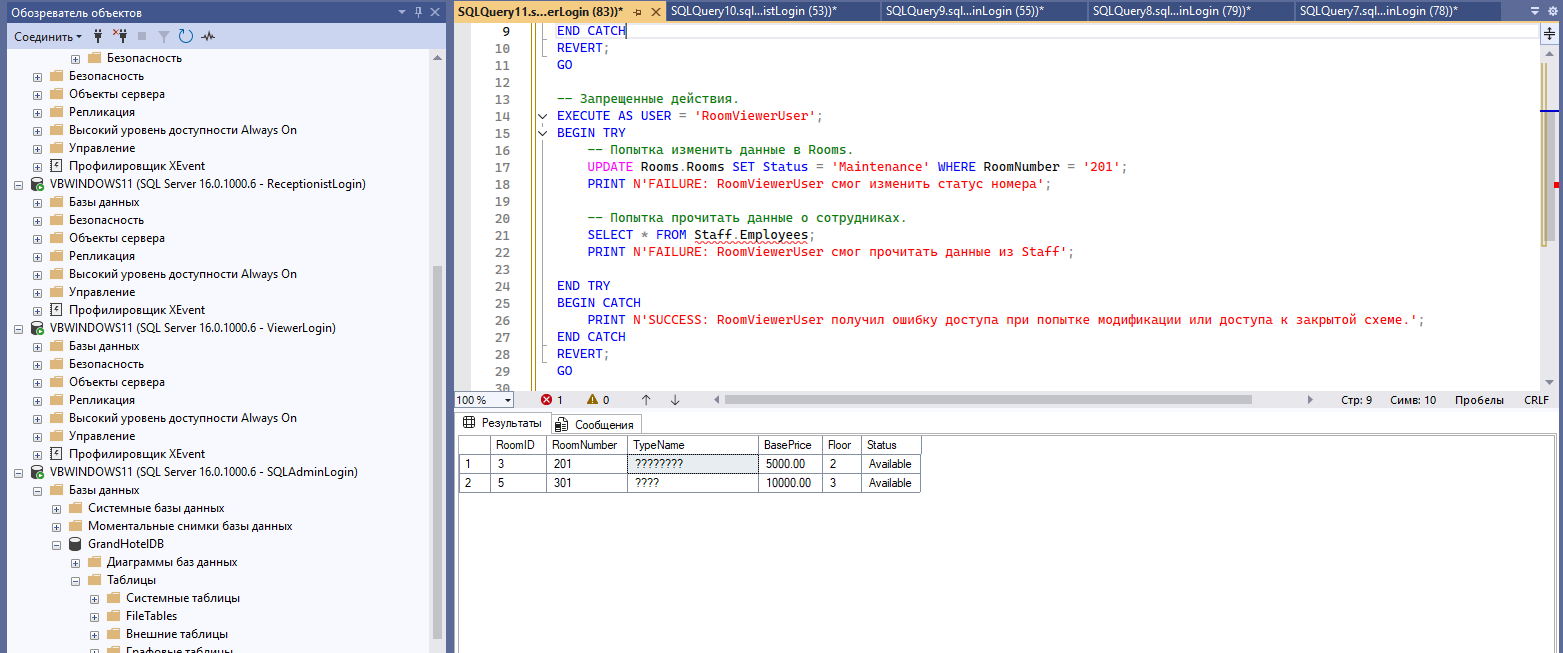
**Тестирование ролей:**

1. Подключитесь к SSMS под HotelAdminUser и попробуйте добавить нового сотрудника, изменить статус номера, удалить запись из истории бронирования.



1. Подключитесь под RoomViewerUser и выполните SELECT \* FROM Rooms.AvailableRooms; – вы должны увидеть только доступные номера. Попробуйте изменить данные о номерах, прочитать информацию о сотрудниках.





**Цель:** Научиться загружать данные в существующую структуру базы данных с использованием различных методов (BULK INSERT, OPENROWSET, а также через мастер импорта SSMS 21).

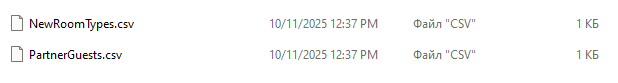
Отель «GrandHotel» получил два файла данных:

1. **Новые типы номеров:** список новых, эксклюзивных люксов, которые пока не внесены в основную таблицу Rooms.RoomTypes.
2. **Списки гостей из партнерских систем:** файл с гостями, которые бронировали через внешних агентов. Эти данные нужно загрузить в Booking.Guests.

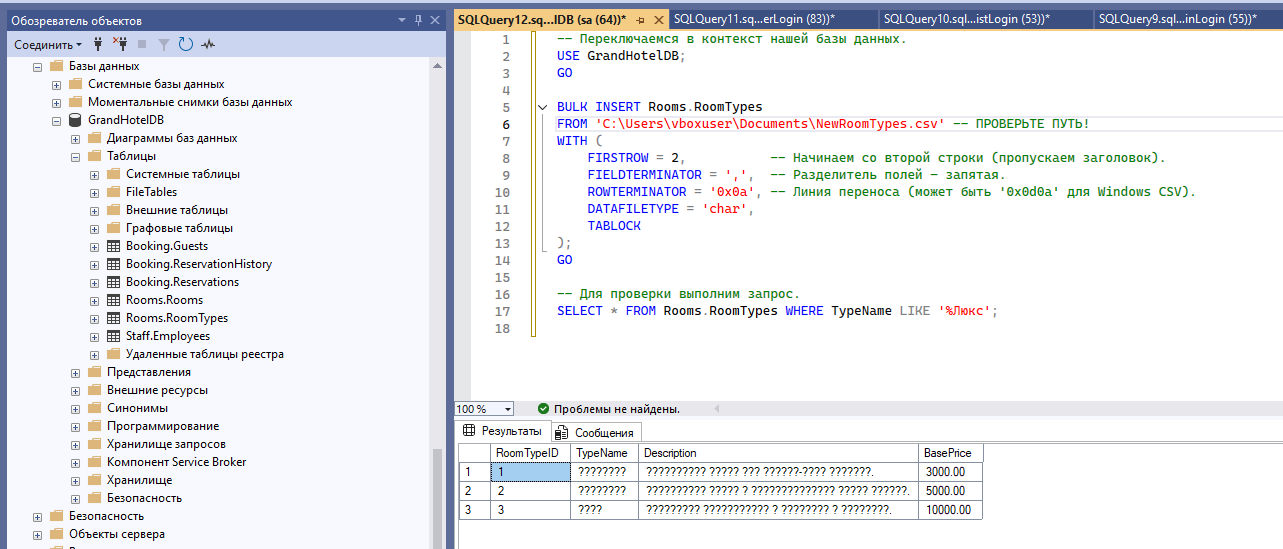
**Ход выполнения работы:**

Запустите SQL Server Management Studio (SSMS) и подключитесь к вашему экземпляру SQL Server под учетной записью **sa** (или другим административным логином).

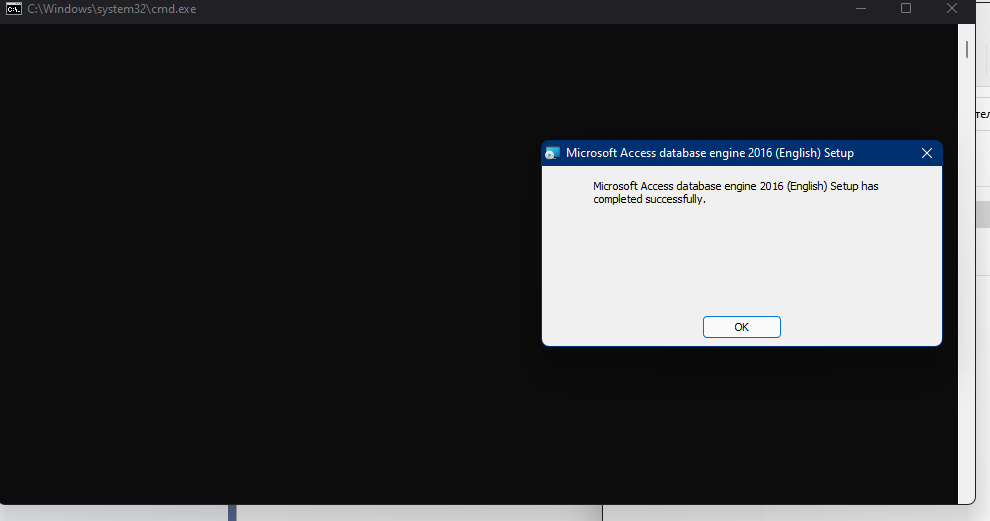
1. **Подготовка данных (Симуляция):**

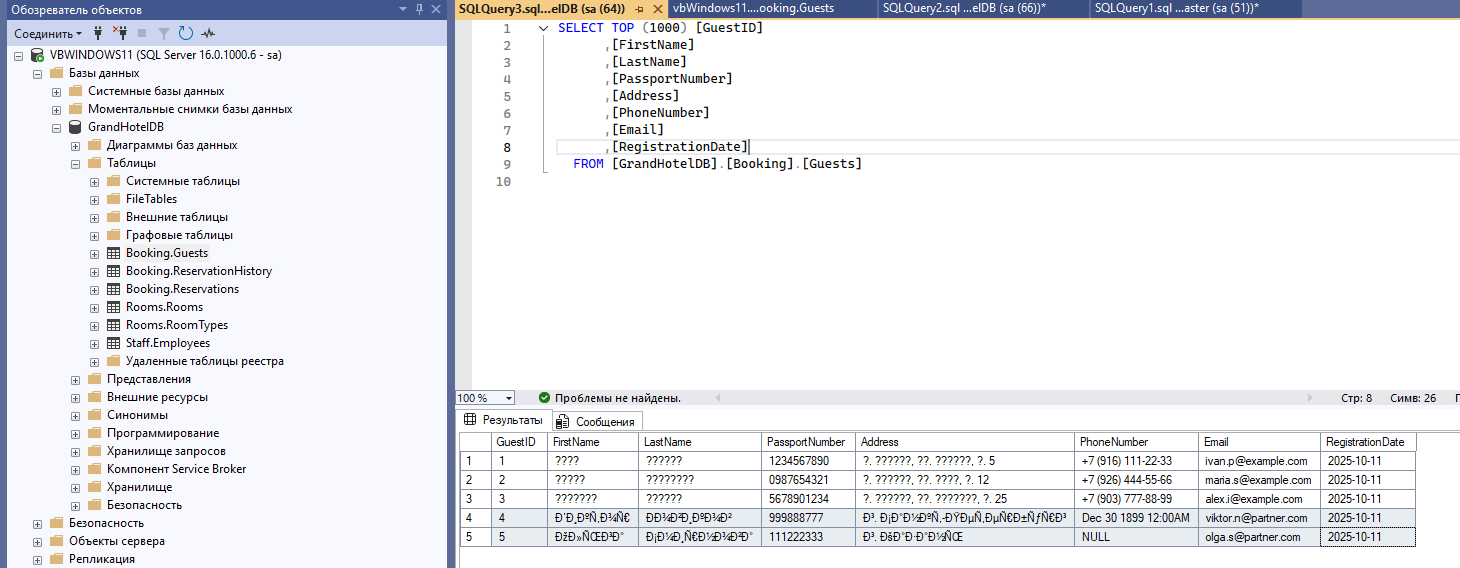


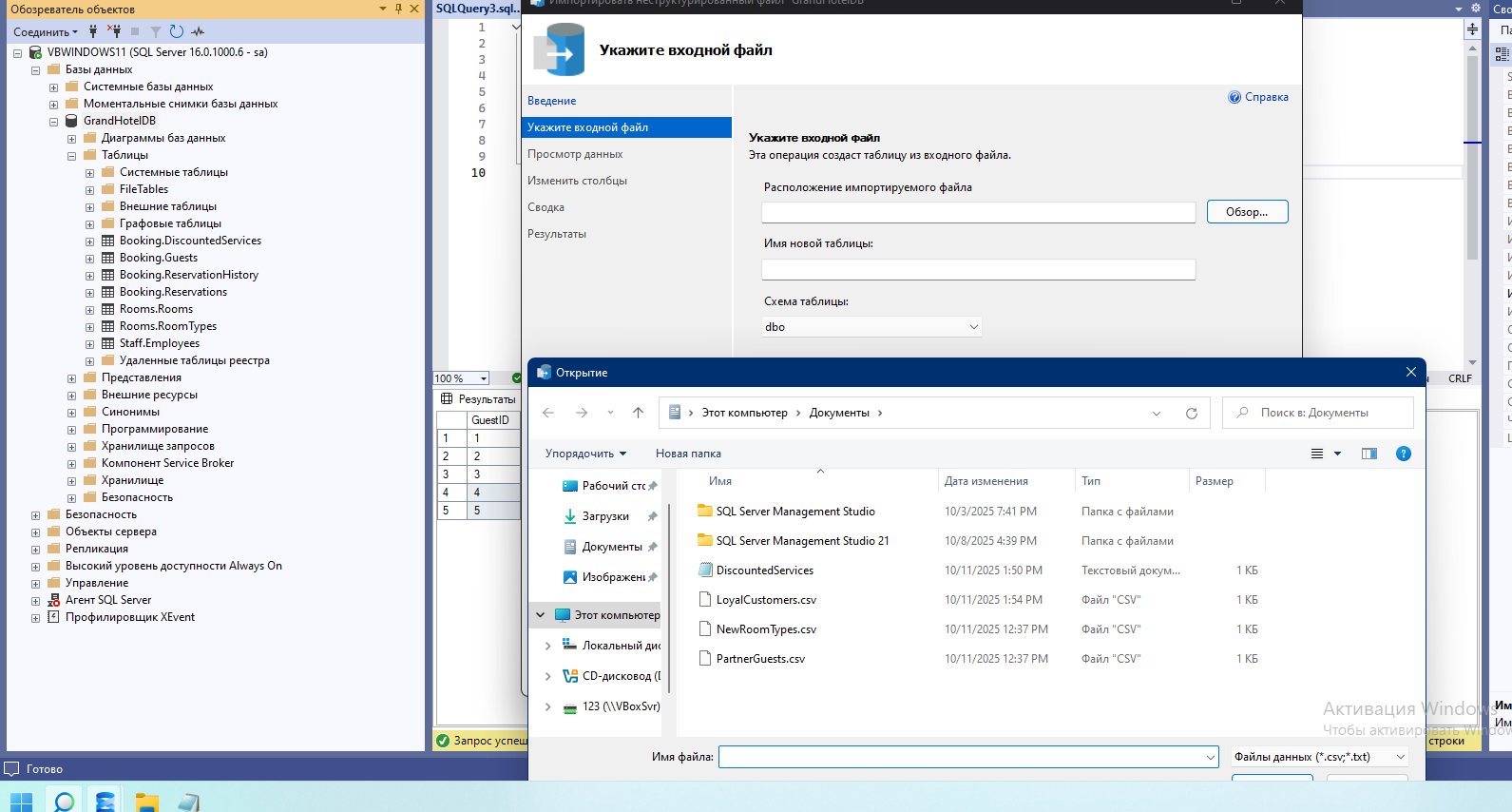
1. **Импорт данных с помощью BULK INSERT (для NewRoomTypes.csv).** Самый простой и быстрый способ импорта из локального файла. Требует прав ADMINISTER BULK OPERATIONS или членства в sysadmin.

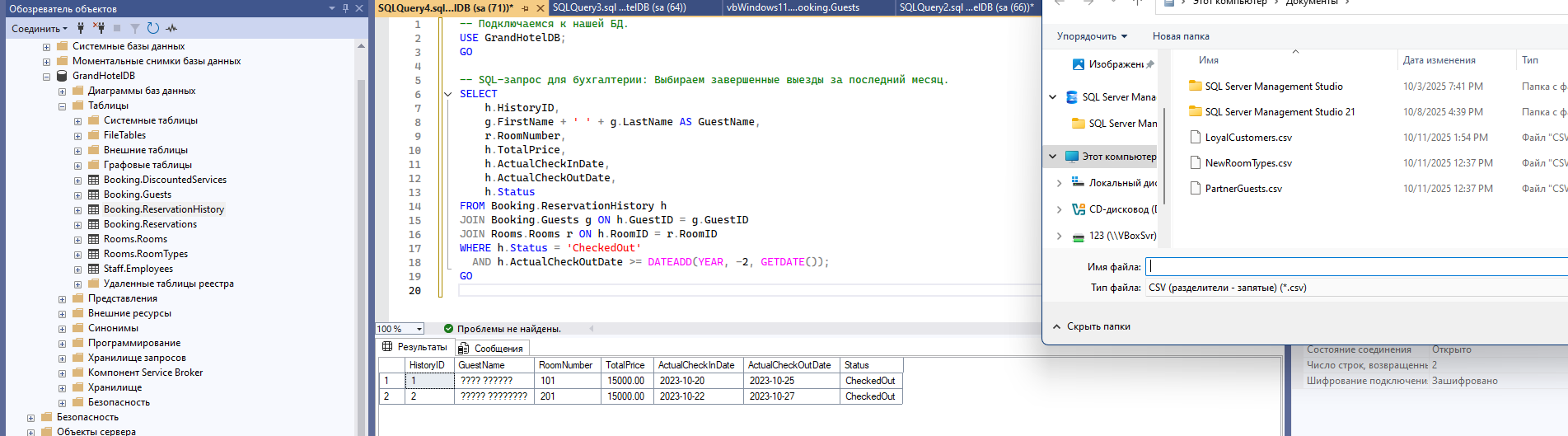


1. **Импорт данных с помощью OPENROWSET (для PartnerGuests.csv).** Позволяет читать данные из внешних источников, часто используя провайдеры OLE DB (например, Microsoft ACE OLEDB Provider для Excel/CSV). Требует, чтобы на сервере был установлен соответствующий провайдер и включена опция Ad Hoc Distributed Queries.













**Цель:** Научиться использовать курсоры T-SQL для построчной обработки данных, когда операции, основанные на множествах (SET-based operations), не подходят.

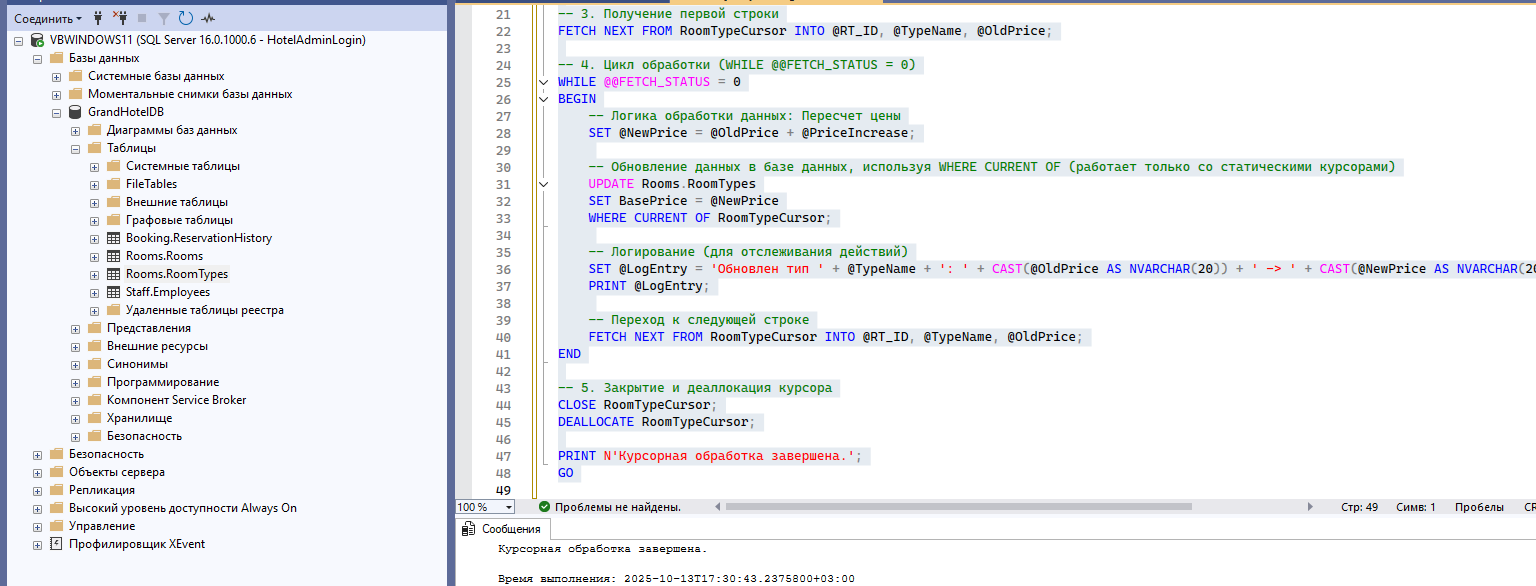
Необходимо пересчитать и обновить цены на все типы номеров (Rooms.RoomTypes). Цена должна увеличиваться на **фиксированную надбавку**, но только для тех типов, которые были созданы до определенной даты, и мы хотим видеть лог каждого изменения.

**Ход выполнения работы:**

1. **Подготовительные действия:**

Подключитесь под HotelAdminUser. Убедитесь, что в Rooms.RoomTypes есть записи.

1. **Создание Курсора.** Мы будем использовать **статический** курсор, чтобы гарантировать, что набор данных не изменится в процессе обработки.



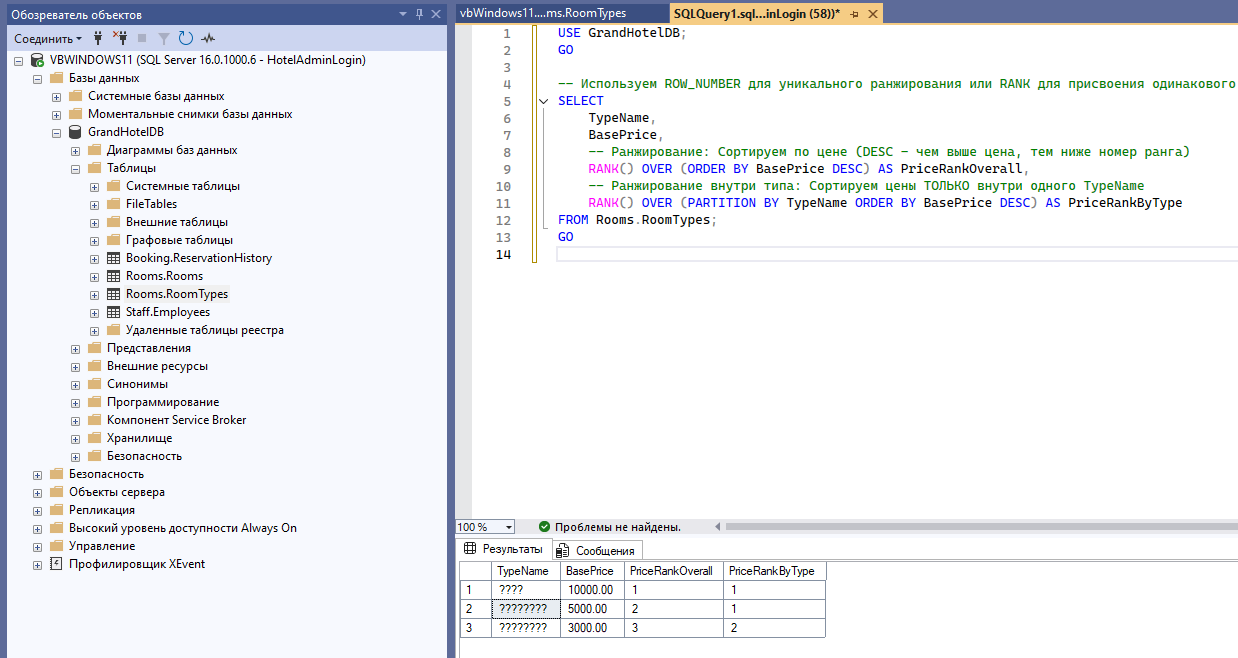
**Цель:** Научиться использовать оконные функции (ROW\_NUMBER, RANK, LAG, агрегатные функции с OVER (PARTITION BY...)) для выполнения сложной аналитики без самосоединений (self-joins).

Администрации отеля нужны два аналитических отчета:

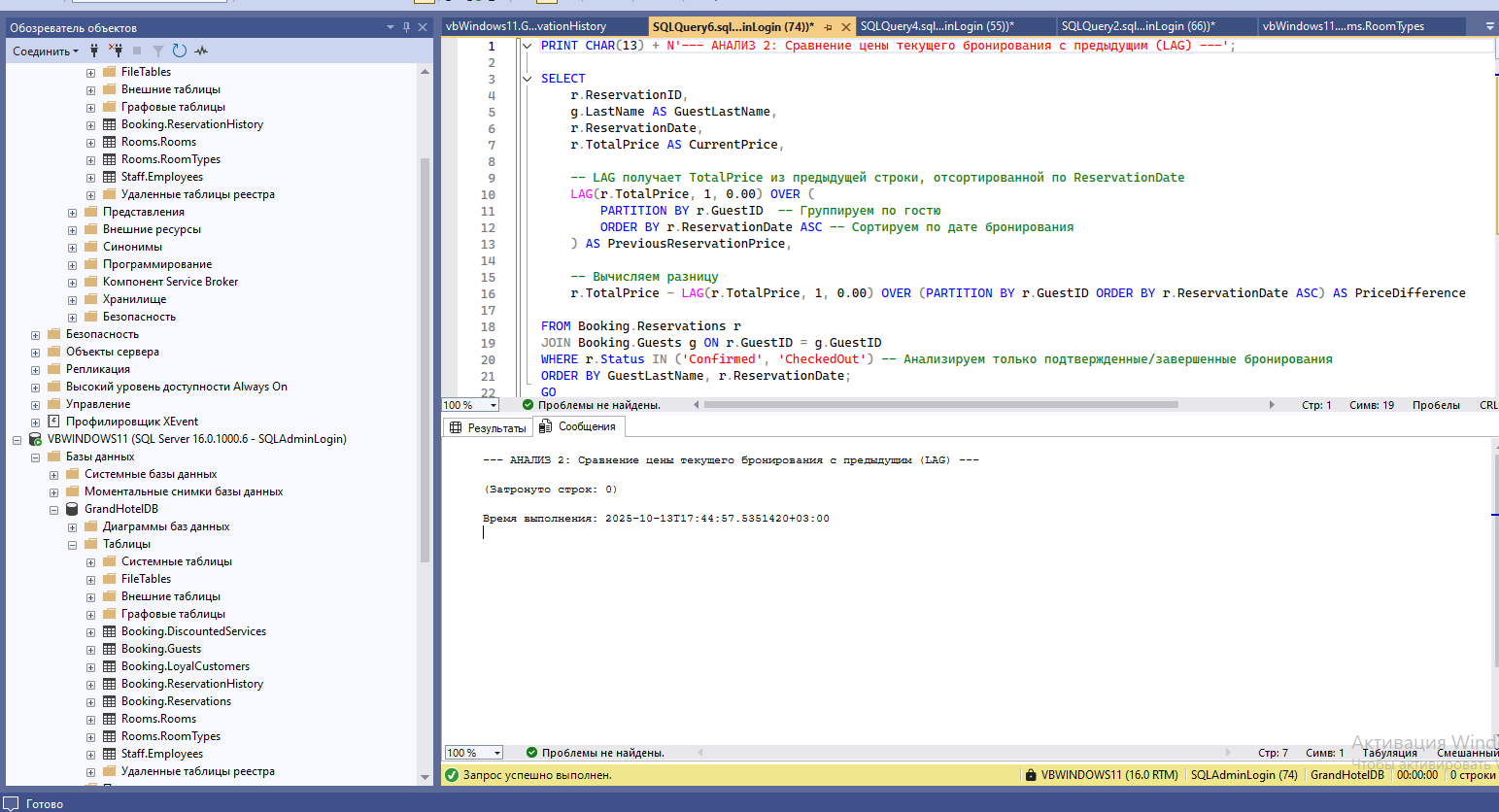
1. **Ранжирование:** Посчитать ранг номеров внутри каждого типа по их базовой цене (чем дороже, тем выше ранг).
2. **Сравнение с предыдущим:** Посмотреть, как изменилась цена бронирования (TotalPrice) по сравнению с предыдущим бронированием того же гостя.

**Ход выполнения работы:**

1. **Подготовительные действия:**
   * Подключитесь под HotelAdminUser или SQLAdminLogin.
   * Убедитесь, что у вас есть данные в Rooms.RoomTypes (включая новые типы из Занятия 7) и в Booking.Reservations.
2. **Анализ 1: Ранжирование цен по типам номеров (ROW\_NUMBER, RANK):**



1. **Анализ 2: Сравнение цен бронирований (LAG).** Функция LAG позволяет получить значение из предыдущей строки в определенном наборе (partition).



**Цель:** Научиться создавать резервные копии базы данных (полные, дифференциальные, журналов транзакций) и восстанавливать базу данных из этих копий.

В отеле «GrandHotel» необходимо разработать план резервного копирования, чтобы защитить данные от потери из-за сбоев оборудования, ошибок пользователей или других непредвиденных обстоятельств.

**Ход выполнения работы:**

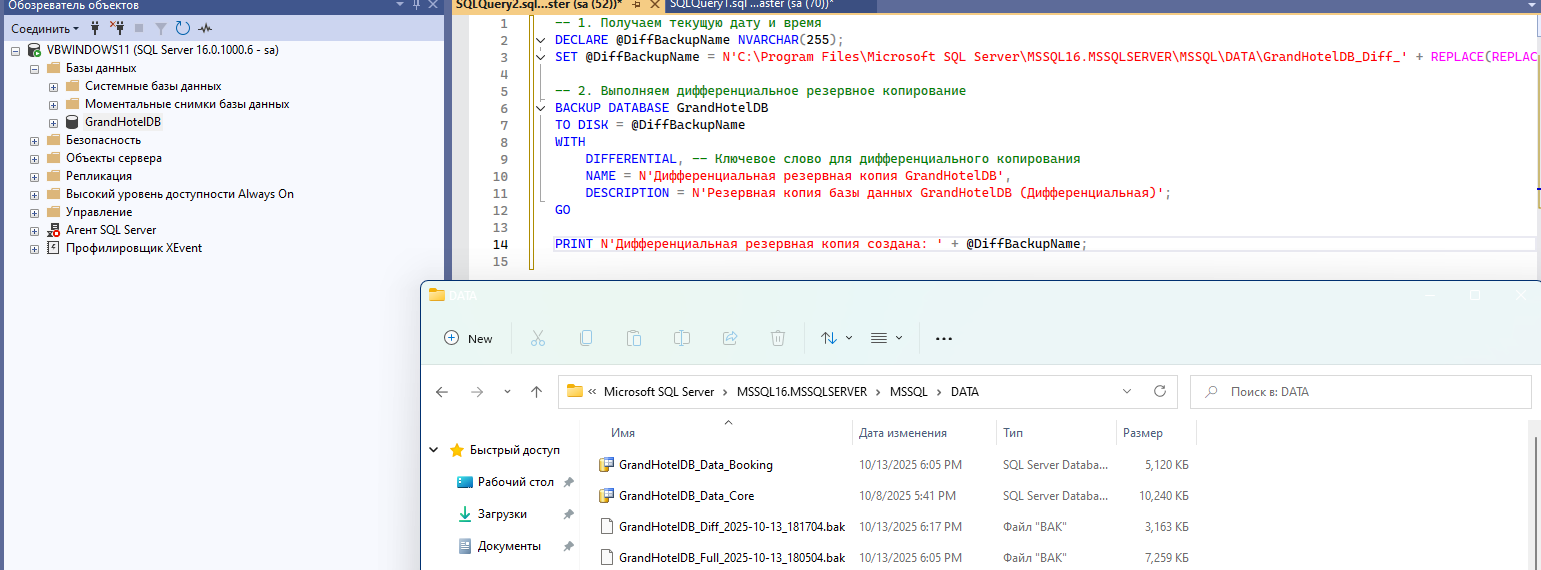
1. **Подготовительные действия:**

Подключитесь под SQLAdminLogin.

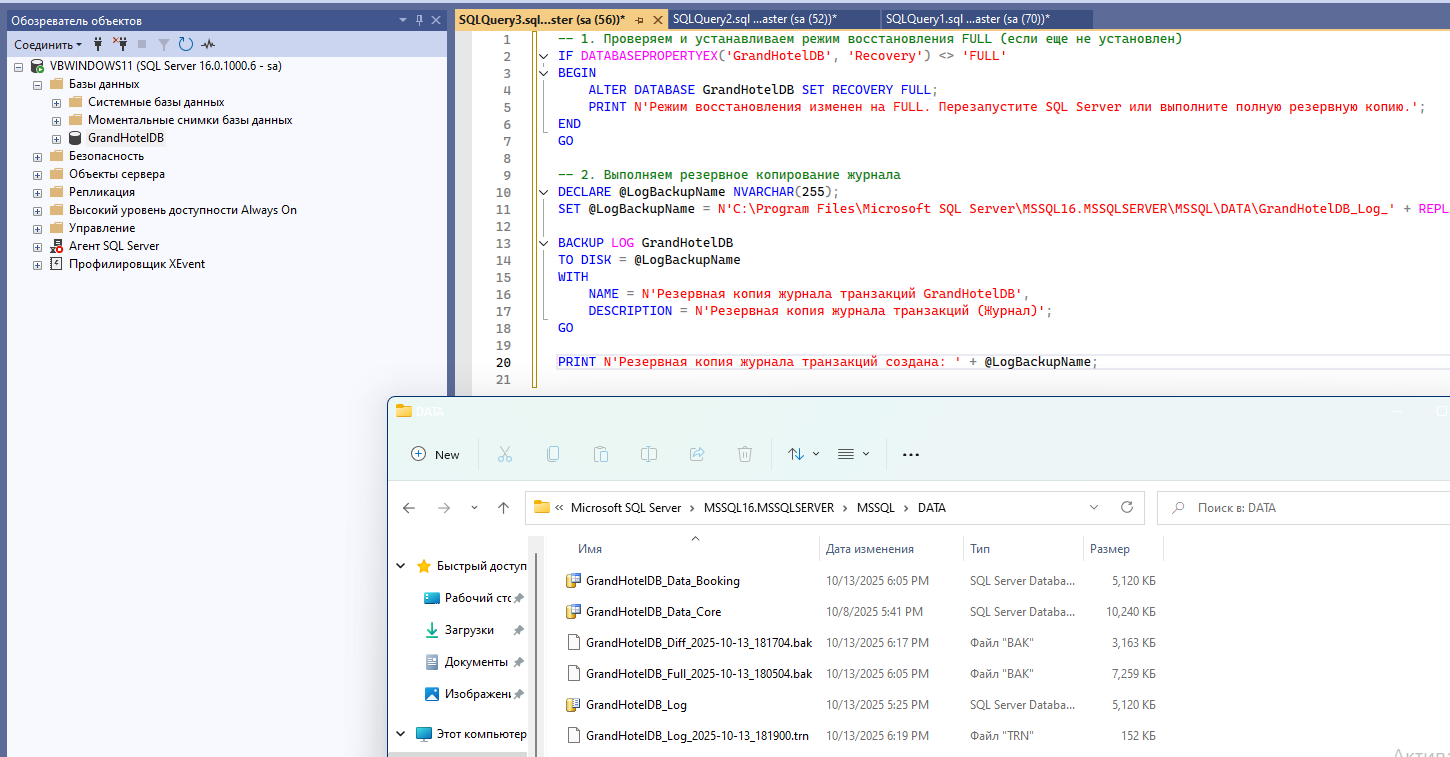
1. **Создание Полной Резервной Копии (Full Backup).** Полная копия содержит все данные и структуру базы данных на момент создания копии.



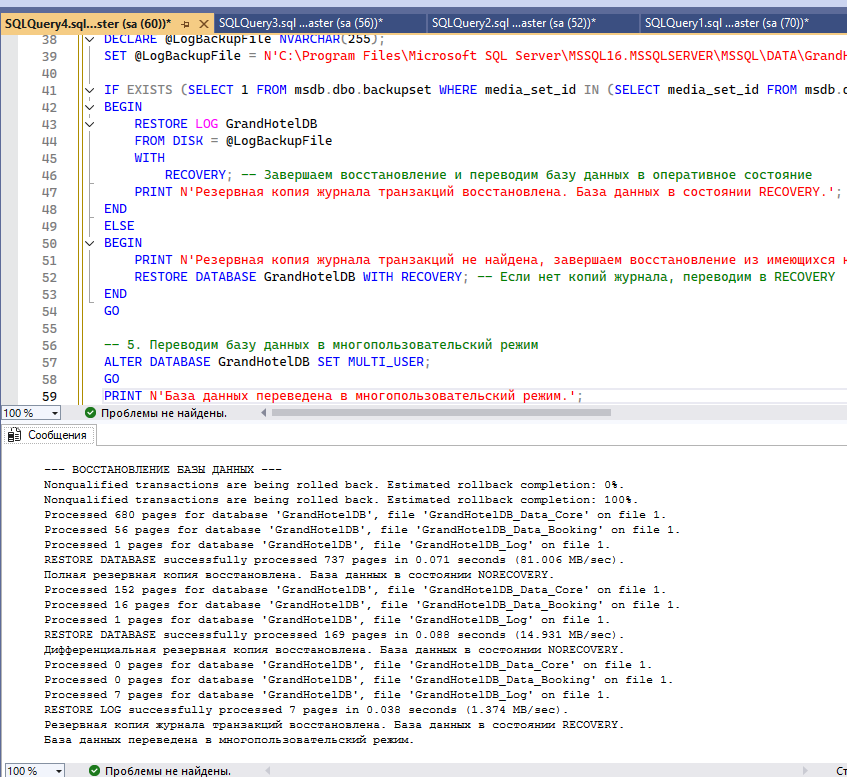
1. **Создание Дифференциальной Резервной Копии (Differential Backup).** Дифференциальная копия содержит только изменения, внесенные с момента последней *полной* резервной копии. Восстановление требует сначала восстановить полную, а затем дифференциальную копию.



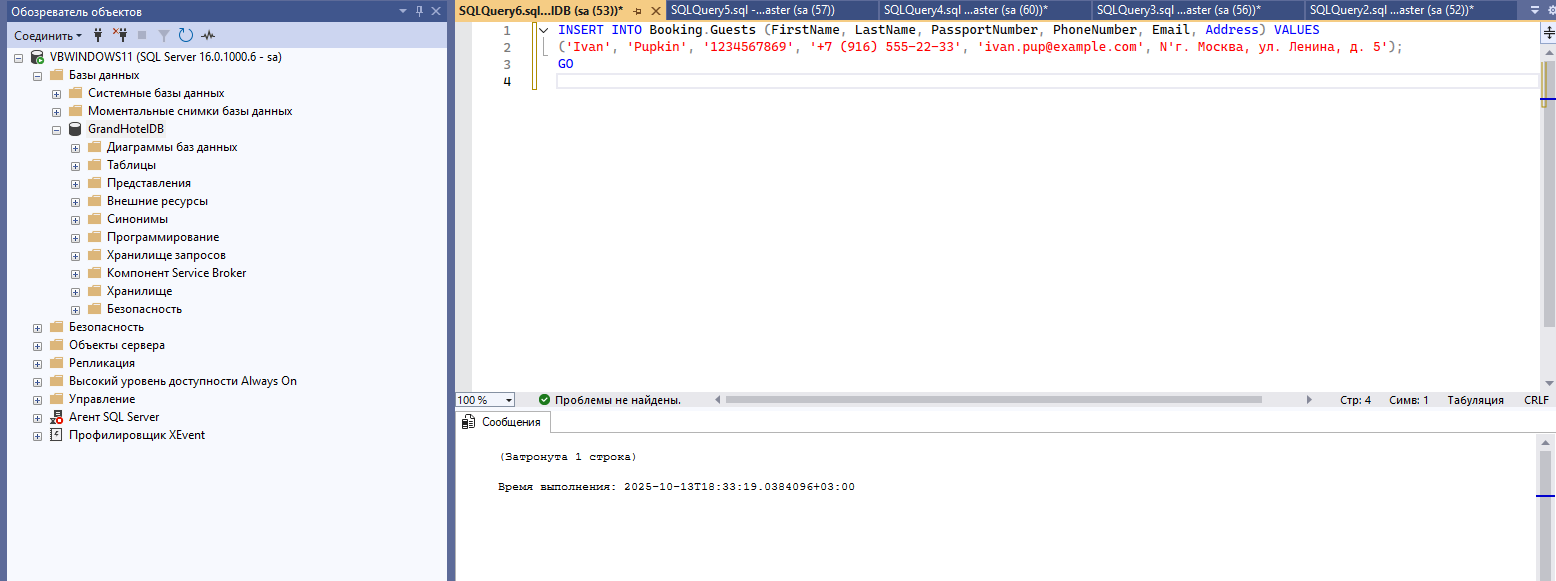
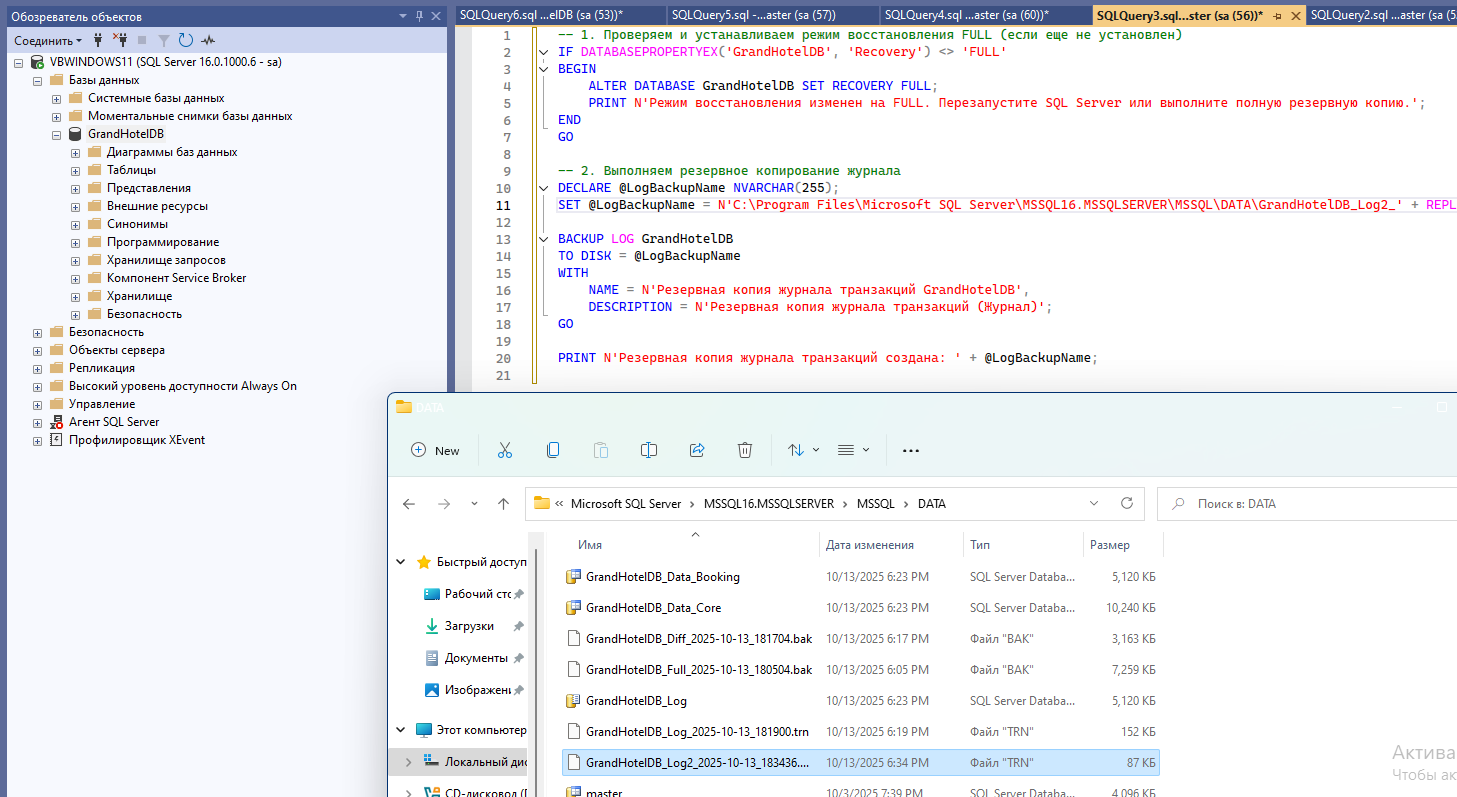
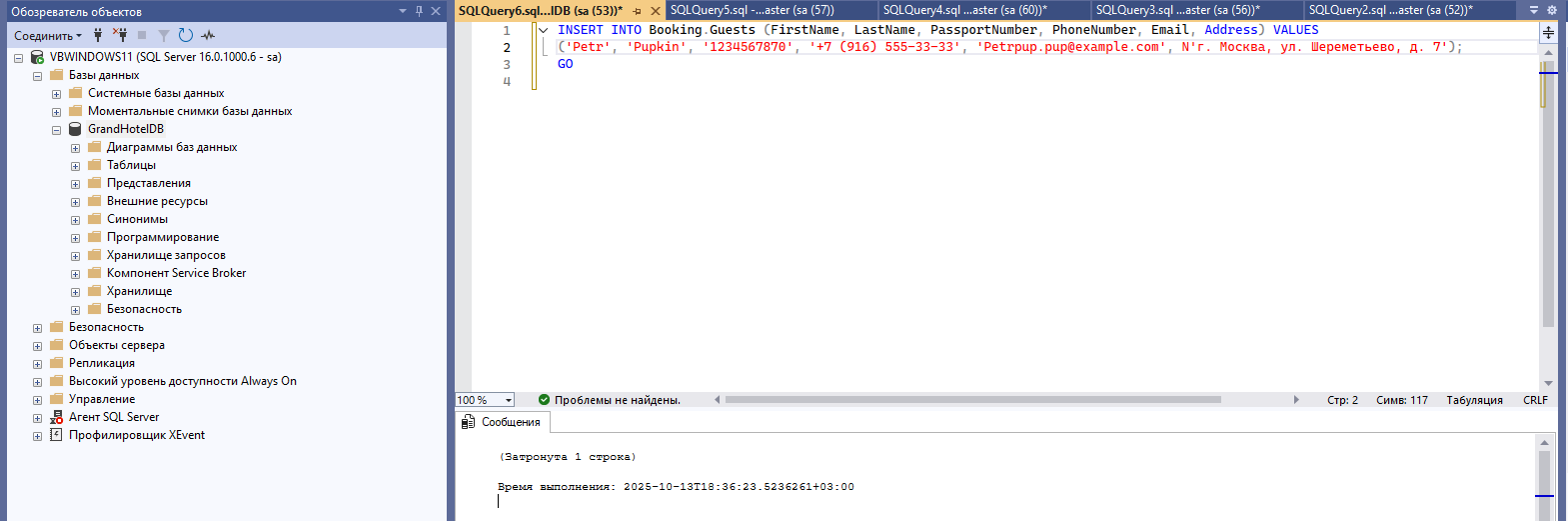
1. **Создание Резервной Копии Журнала Транзакций (Transaction Log Backup).** Журнал транзакций содержит записи обо всех изменениях данных. Резервное копирование журнала позволяет восстановить базу данных на определенный момент времени. Для резервного копирования журналов, база данных должна находиться в режиме полного восстановления (FULL recovery model).



1. **Восстановление Базы Данных (RESTORE).** Имитируем потерю данных и восстанавливаем базу данных из резервных копий. Перед восстановлением необходимо перевести базу данных в состояние SINGLE\_USER (монопольный режим). Рекомендуется использовать опцию RECOVERY при восстановлении *последней* резервной копии.



**Дополнительные задания:**

1. **Восстановление на определенный момент времени (Point-in-Time Recovery):**
   * Включите режим FULL для GrandHotelDB.
   * Сделайте полную резервную копию.
   * Внесите изменения в базу данных (например, добавьте нового гостя, забронируйте номер).
   * Сделайте резервную копию журнала транзакций.
   * Внесите еще изменения.
   * Определите момент времени, на который нужно восстановить базу данных (например, до внесения последних изменений).
   * Восстановите базу данных, указав этот момент времени:
   * 
   * Внесите изменения в базу данных (например, добавьте нового гостя, забронируйте номер).
   * 
   * Сделайте резервную копию журнала транзакций.
   * 
   * Внесите еще изменения.
   * 
   * Определите момент времени, на который нужно восстановить базу данных (например, до внесения последних изменений).
   * Восстановите базу данных, указав этот момент времени:

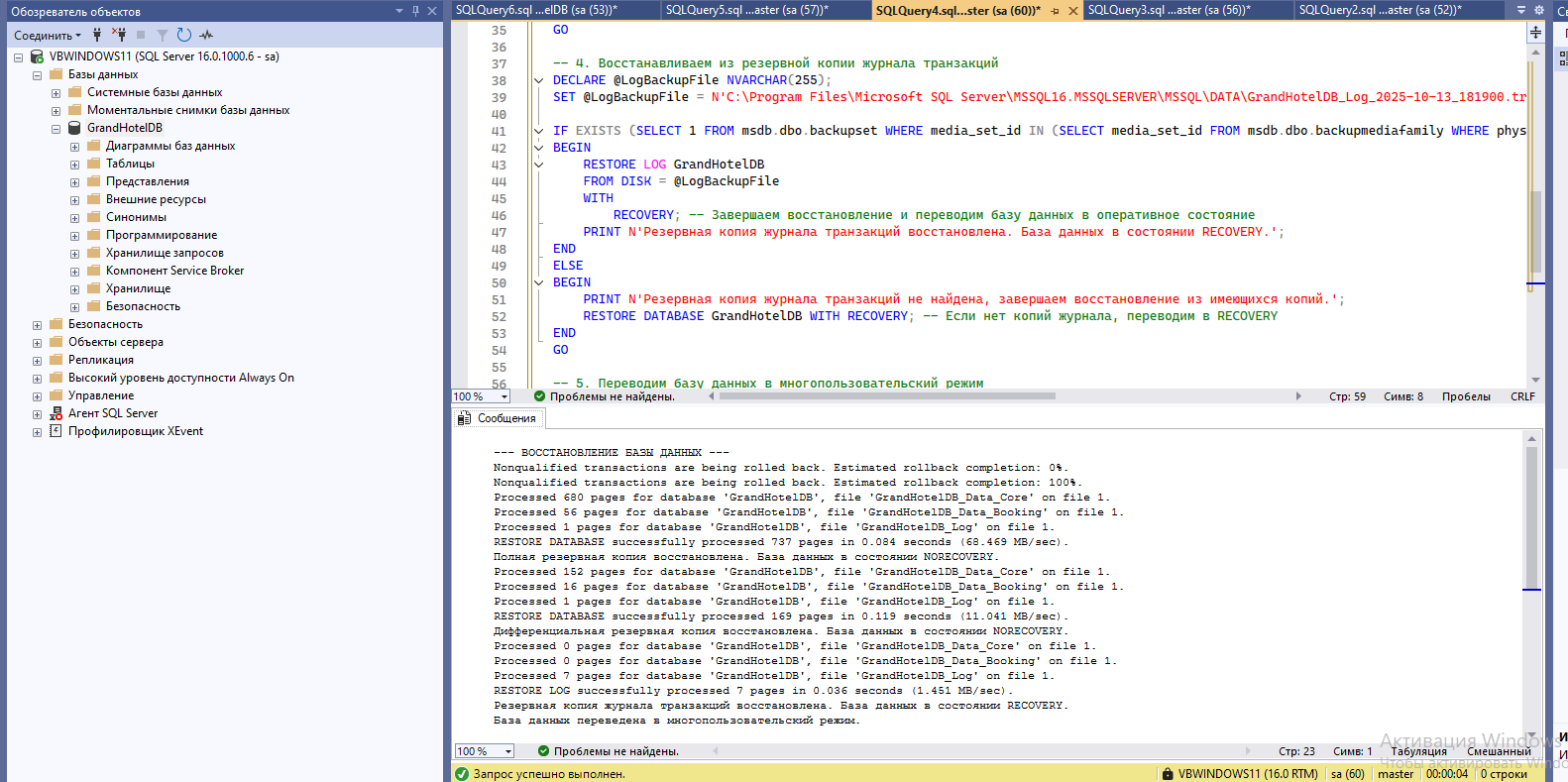
RESTORE DATABASE GrandHotelDB

**FROM** DISK = 'C:\FullBackup.bak' **WITH** NORECOVERY

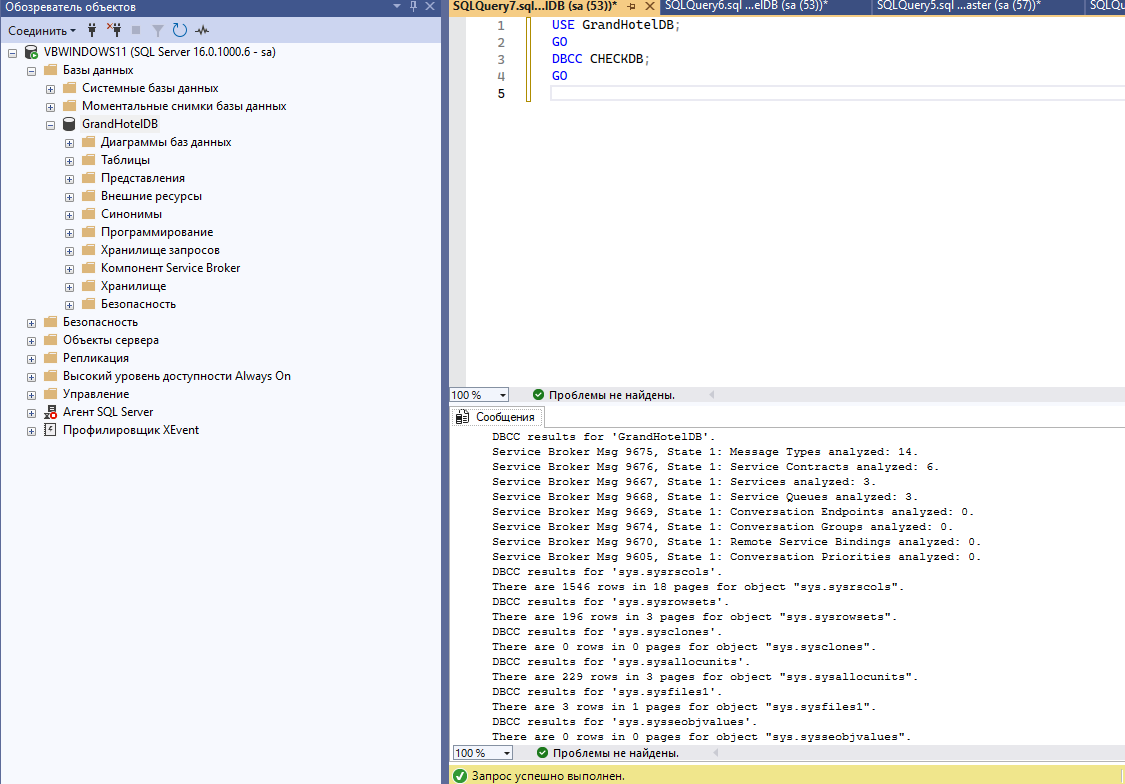
RESTORE LOG GrandHotelDB

**FROM** DISK = 'C:\LogBackup.trn' **WITH** STOPAT = 'Feb 29, 2024 10:35 AM' *-- Укажите дату и время*

RESTORE DATABASE GrandHotelDB **WITH** RECOVERY



1. **Проверка целостности базы данных (DBCC CHECKDB):**
   * После восстановления выполните DBCC CHECKDB для проверки целостности базы данных:



**Цель:** Научиться создавать, настраивать и управлять заданиями (Jobs) в SQL Server Agent для автоматизации рутинных административных задач, таких как очистка старых данных или создание отчетов.

Администрации отеля необходимо:

1. Ежедневно в 01:00 ночи удалять записи о бронированиях, которые были отменены или завершены более 6 месяцев назад из таблицы истории (Booking.ReservationHistory).
2. Еженедельно (по воскресеньям в 03:00) генерировать полный отчет о доступных номерах и сохранять его во временную таблицу в БД.

**Предварительное условие:**

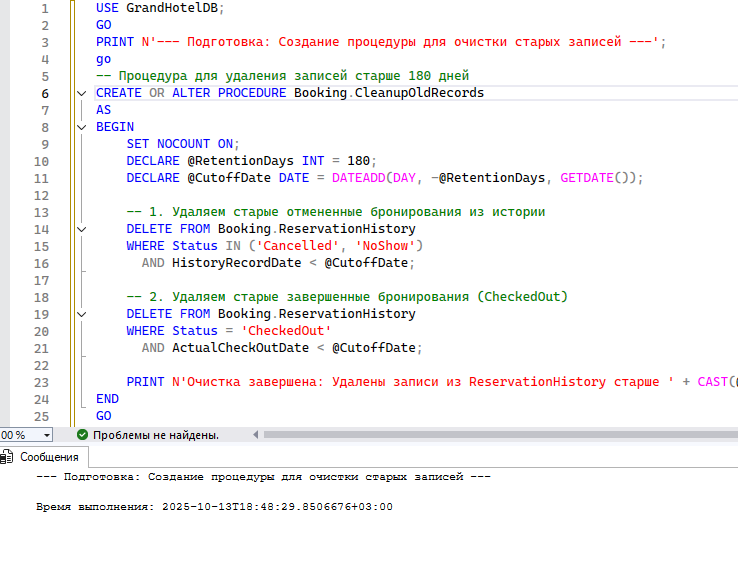
* Убедитесь, что **SQL Server Agent** запущен на вашем экземпляре SQL Server.
* Убедитесь, что у пользователя, от имени которого будет работать задание (Proxy Account или Account associated with the Agent service), есть права DELETE в схеме Booking и права CREATE TABLE в GrandHotelDB.

**Ход выполнения работы:**

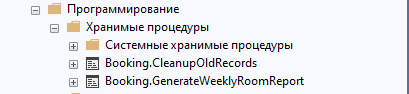
1. **Подготовка: Создание хранимой процедуры для очистки (Step 1)**

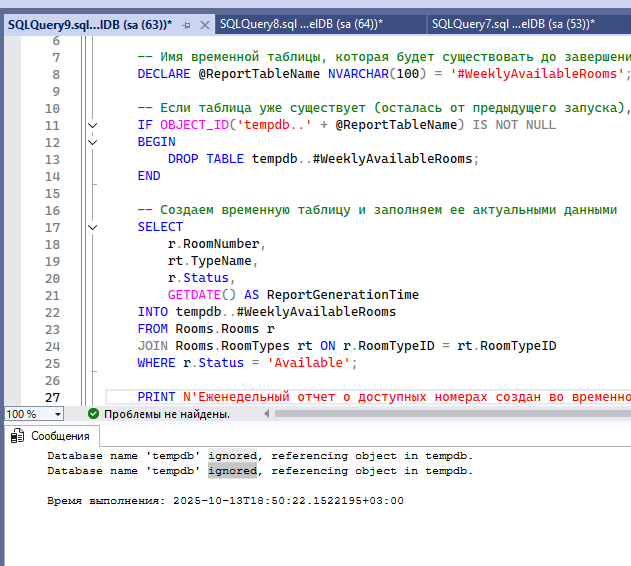
Создадим хранимую процедуру, которая будет выполнять логику удаления старых записей.





1. **Подготовка: Создание хранимой процедуры для еженедельного отчета (Step 2)**

Создадим процедуру, которая генерирует отчет и сохраняет его во временную таблицу



1. **Настройка Задания SQL Server Agent (Job):**
   * *Выполняется в SSMS, через Object Explorer -> SQL Server Agent -> Jobs.*

**A. Создание нового Задания (Job):**

* + Правой кнопкой по **Jobs** -> **New Job…**
  + **General Page:**
    - Name: GH\_Maintenance\_And\_Reporting
    - Owner: Укажите пользователя с нужными правами (например, SQLAdminLogin или учетную запись службы Агента).

**B. Настройка Шагов (Steps):**

* + Перейдите на страницу **Steps** -> **New…**

**Шаг 1: Очистка старых данных** \* Name: Step 1: Cleanup Old Reservations \* Type: T-SQL script (Transact-SQL) \* Database: GrandHotelDB \* Command: sql EXEC Booking.CleanupOldRecords;  \* Нажмите OK.

* + **Шаг 2: Генерация еженедельного отчета**
    - Нажмите **New…**
    - Name: Step 2: Generate Weekly Room Availability Report
    - Type: T-SQL script (Transact-Transact-SQL)
    - Database: GrandHotelDB
    - Command:

**EXEC** Booking.GenerateWeeklyRoomReport;

* + - Нажмите OK.

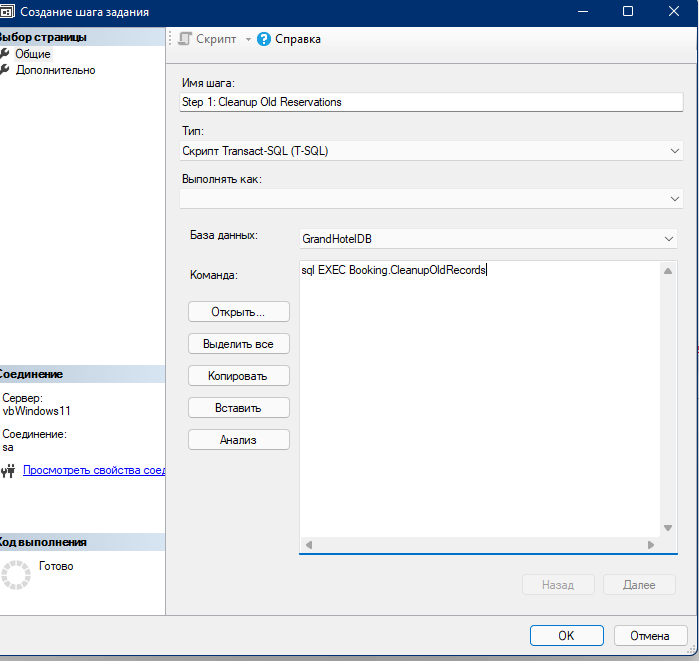
**C. Настройка Расписания (Schedules):**

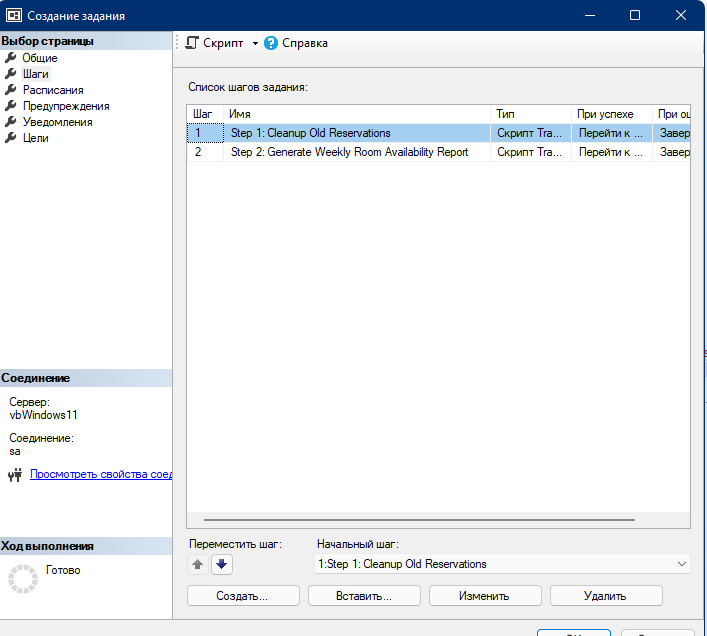
* + Перейдите на страницу **Schedules** -> **New…**
  + **Daily Cleanup Schedule (Для Шага 1):**
    - Name: Daily\_0100\_Cleanup
    - Frequency: Daily
    - Occurs every: 1 Day
    - Time: 01:00:00 AM
    - **Enabled:** Checked
  + Нажмите OK.
  + **Weekly Report Schedule (Для Шага 2):**
    - Нажмите **New…**
    - Name: Weekly\_Sunday\_Report\_0300
    - Frequency: Weekly
    - Occurs every: 1 Week
    - On: **Sunday**
    - Time: 03:00:00 AM
    - **Enabled:** Checked
  + Нажмите OK.

**D. Настройка Уведомлений (Notifications - Опционально):**

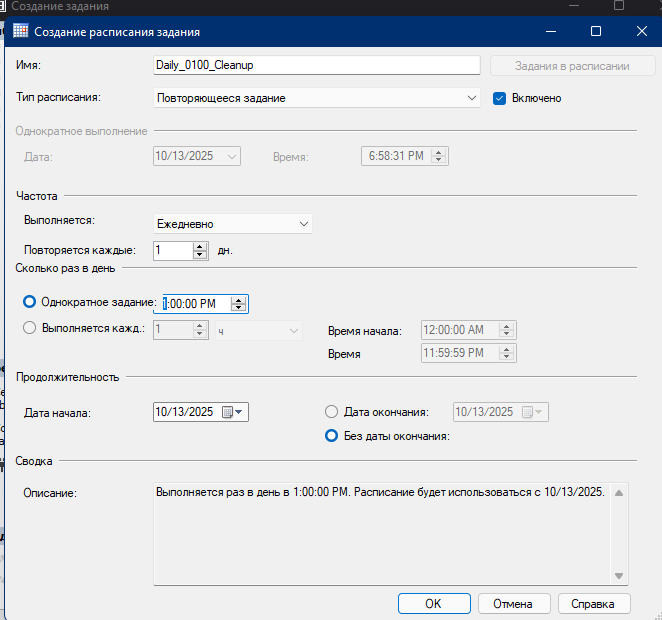
* + Перейдите на страницу **Alerts** и **Notifications**, чтобы настроить уведомления по электронной почте или пейджеру в случае сбоя задания.

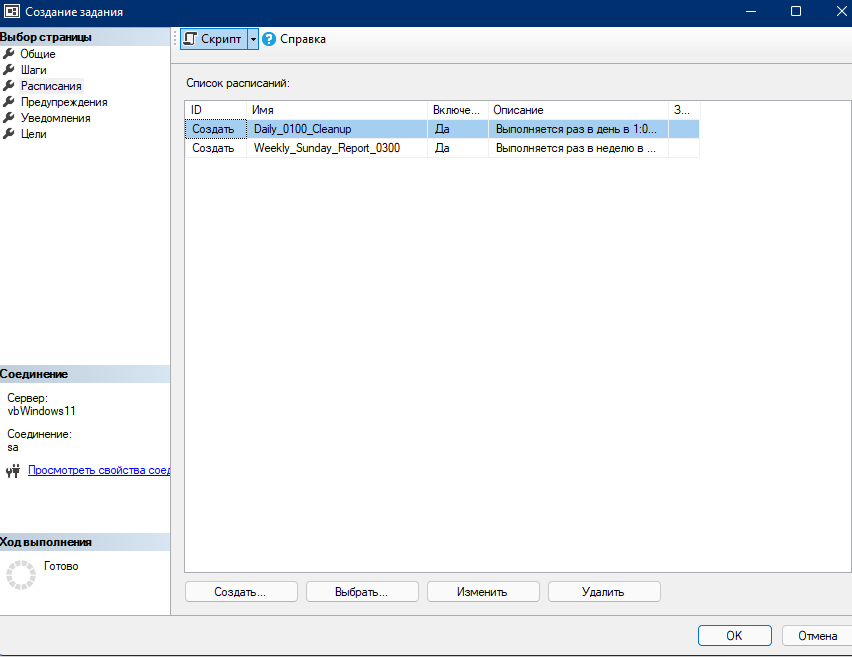
1. **Тестирование Задания:**
   * После сохранения задания, вернитесь в **Jobs**.
   * Правой кнопкой на GH\_Maintenance\_And\_Reporting -> **Start Job at Step…**
   * Проверьте статус выполнения в **Job Activity Monitor**.
   * После успешного выполнения (зеленая галочка), проверьте, что старые записи удалились (Шаг 1) и, если вы запустили Шаг 2, вы можете временно подключиться под ReceptionistUser и проверить, существует ли таблица #WeeklyAvailableRooms в сессии, запущенной после процедуры (или проверить логи Агента).



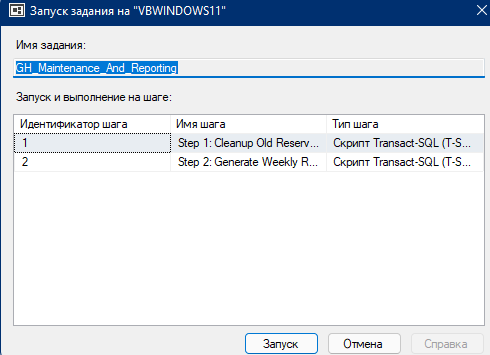


**Настройка Расписания (Schedules):**





**Тестирование Задания**



**Цель:** Настроить SQL Server Audit для логирования критических действий (например, удаление данных или попытки доступа пользователей с низкими привилегиями).

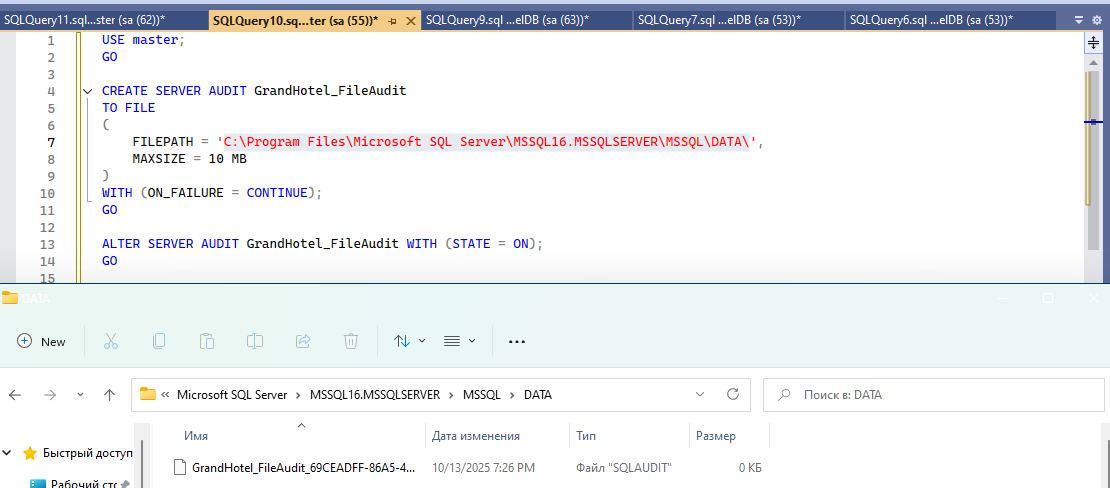
Необходимо отслеживать все попытки пользователей:

1. Удалять записи из таблицы истории бронирований (Booking.ReservationHistory).
2. Пытаться получить доступ к конфиденциальной таблице сотрудников (Staff.Employees).

**Ход выполнения работы:**

1. **Подготовительные действия:**
   * Подключитесь под SQLAdminLogin (требуются права ALTER ANY SERVER AUDIT и ALTER ANY DATABASE AUDIT SPECIFICATION).
   * Убедитесь, что папка для логов аудита доступна SQL Server.
2. **Шаг 1: Создание Спецификации Серверного Аудита (Server Audit)**

Серверный аудит определяет, **где** и **как** будут храниться логи.



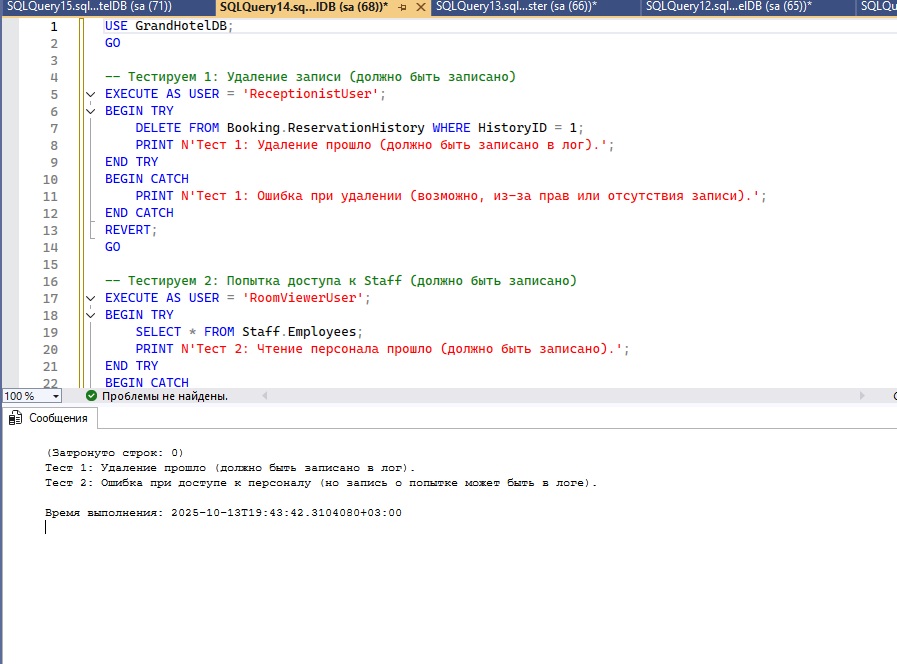
1. **Шаг 2: Создание Спецификации Аудита Базы Данных (Database Audit Specification)**

Спецификация аудита определяет, **что именно** мы хотим отслеживать внутри базы данных.



1. **Шаг 3: Тестирование Аудита**

Теперь выполним действия, которые должны быть записаны в лог.



1. **Шаг 4: Просмотр Логов Аудита**

После выполнения тестовых действий нужно посмотреть, что записалось в файл.

