Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут прикладної математики та фундаментальних наук

Кафедра прикладної математики

Курсова робота
з дисципліни «Надвеликі бази даних»

Виконала:

студентка групи: ПМ-32
Марія ГАРБУЗЕНКО

Перевірив:

Любінський Б. Б.

_____

(дата) (підпис викладача)

Львів — 2025

Зміст

ВСТУП

Актуальність теми. У сучасному світі дані є одним із найцінніших активів будь-якого підприємства. Проте наявність даних сама по собі не гарантує успіху; важливим є вміння їх обробляти та перетворювати на корисну інформацію для прийняття управлінських рішень. Компанії, що займаються управлінням нерухомістю та комунальними послугами, щодня генерують тисячі транзакцій (інвойсів). Аналіз цих даних в Excel стає неефективним, повільним та призводить до помилок.

Мета роботи. Метою даної роботи є проектування та розробка комплексної системи Business Intelligence (BI) на базі стеку технологій Microsoft SQL Server. Система має автоматизувати збір даних, забезпечити їх зберігання у сховищі даних (Data Warehouse) та надати інструменти для візуалізації через аналітичні звіти.

Об’єкт дослідження: процес обробки фінансових даних (інвойсів) підприємства.

Предмет дослідження: методи та засоби побудови сховищ даних, ETL-процесів та OLAP-систем.

Задачі:

1. Проаналізувати предметну область та вимоги до звітності.
2. Спроєктувати схему сховища даних (схема «Зірка»).
3. Розробити ETL-процеси для завантаження даних за допомогою SSIS.
4. Створити OLAP-куб або аналітичну модель у SSAS.
5. Розробити інтерактивні звіти у SSRS.

Інструментарій: Microsoft SQL Server, Visual Studio (SSDT), SSIS, SSAS, SSRS.

Основна частина

Постановка задачі

Варіант 2. Орендарі.

Специфічні вимоги:

• Мінімум 5 000 приміщень

• Мінімум 100 000 операцій оренди

• Історія за 5 років

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Характеристика об’єкта дослідження

- **Опис галузі:** Компанія працює у сфері управління нерухомістю (Real Estate) або надання комунальних послуг. Діяльність компанії охоплює кілька регіонів та міст (наприклад, США: Нью-Йорк, Чикаго тощо).
- **Основні бізнес-сутності:**
  - **Клієнти (Tenants):** Юридичні або фізичні особи, що орендують приміщення.
  - **Об'єкти нерухомості (Buildings):** Будівлі, які мають адресу, власника та характеристики.
  - **Послуги:** Оренда, електроенергія, вода, обслуговування.
  - **Фінансові транзакції (Invoices):** Рахунки, які виставляються клієнтам щомісяця.
- **Проблема масштабу:** Зі збільшенням кількості будівель зростає обсяг даних. Ручна обробка призводить до втрати інформації про боржників та помилок у нарахуваннях.

1.2. Аналіз існуючого бізнес-процесу обробки даних (AS-IS)

- **Поточний стан:** Зараз звітність ведеться в Excel. Кожен регіональний менеджер надсилає свій файл. Головний аналітик вручну копіює дані в один великий файл ("Master Spreadsheet").
- **Недоліки поточного підходу:**

1. **Низька швидкість:** Зведення звіту займає 3-4 дні.
2. **Людський фактор:** Ризик помилки при копіюванні (Copy-Paste error).
3. **Відсутність стандартизації:** В одному файлі місто записане як "New York", в іншому — "NY", у третьому — "N. York".
4. **Обмеження Excel:** При перевищенні 100 000 рядків файл починає "гальмувати".

- **Моделювання процесу:** Необхідно формалізувати процес виставлення рахунку.

1.3. Дослідження джерел вхідних даних

- **Типи джерел:**
  - **CSV-файли:** Експорт із білінгової системи (містить: Date, CustomerID, Amount).
  - **Excel-файли (довідники):** Списки менеджерів, повні назви міст, адреси будівель.
- **Аналіз якості даних:**
  - **Повнота:** Чи всі поля заповнені? (Наприклад, чи є пропущені дати платежів).
  - **Узгодженість:** Чи збігаються ID клієнтів у різних файлах.
  - **Актуальність:** Як часто оновлюються дані (щоденно/щомісяця).

1.4. Визначення вимог до системи (TO-BE)

- **Мета системи:** Створення централізованого сховища даних (Data Warehouse) та системи звітності.
- **Функціональні вимоги (Що система має робити):**
  1. Автоматично завантажувати дані з папки з CSV-файлами (ETL).
  2. Очищати дані від дублікатів.
  3. Зберігати історичні дані за останні 5 років.
  4. Формувати звіти з можливістю фільтрації за містом, роком та типом будівлі.
  5. Експортувати звіти у PDF та Excel.
- **Нефункціональні вимоги (Як система має працювати):**

  1. **Продуктивність:** Час генерації звіту не більше 5 секунд.
  2. **Надійність:** Резервне копіювання бази даних.
  3. **Зручність (Usability):** Інтуїтивно зрозумілий інтерфейс веб-порталу звітів.

1.5. Обґрунтування вибору технологій

- **Огляд аналогів:**
  - *Excel:* Дешево, але не масштабується.
  - *Power BI:* Чудова візуалізація, але потребує підготовлених даних.
  - *Tableau:* Дорого.
- **Вибір стеку MS SQL Server:**
  - **SQL Server (Database Engine):** Надійна СУБД для зберігання терабайтів даних.
  - **SSIS (Integration Services):** Потужний інструмент для ETL без написання складного коду (візуальне програмування).
  - **SSRS (Reporting Services):** Ідеально підходить для регламентованої звітності (Paginated Reports), яка потрібна бухгалтерії (Pixel-perfect друк).
  - **Visual Studio (SSDT):** Єдине середовище розробки для всіх компонентів.

1.6. Порівняльний аналіз архітектур побудови сховищ даних При проектуванні корпоративних сховищ даних (Data Warehouse) традиційно розглядають дві діаметрально протилежні методології: підхід Білла Інмона («Top-Down») та підхід Ральфа Кімбалла («Bottom-Up»).

Підхід Б. Інмона (CIF — Corporate Information Factory): Передбачає створення єдиного нормалізованого сховища даних (3-тя нормальна форма) для всього підприємства. Вітрини даних (Data Marts) створюються вже на основі цього центрального сховища.

- *Переваги:* Єдина версія правди, відсутність дублювання даних, легкість підтримки цілісності.
- *Недоліки:* Висока складність проектування, довгий час впровадження (місяці або роки), висока вартість.

Підхід Р. Кімбалла (Dimensional Modeling): Передбачає створення окремих вітрин даних для бізнес-процесів, які з'єднані між собою через шину вимірів (Conformed Dimensions). Дані зберігаються у денормалізованому вигляді (схема «Зірка»).

- *Переваги:* Швидкий старт (перший звіт можна отримати за тиждень), зрозумілість для бізнесу, висока швидкість запитів на читання.
- *Недоліки:* Надлишковість даних (дублювання), складність синхронізації довідників між різними вітринами.

Обґрунтування вибору: У даній курсовій роботі обрано підхід **Кімбалла (Схема «Зірка»)**, оскільки він ідеально підходить для ітеративної розробки та забезпечує максимальну продуктивність аналітичних запитів у середовищі MS SQL Server Analysis Services.

РОЗДІЛ 2. ГЕНЕРАЦІЯ ТА НАПОВНЕННЯ БАЗИ ДАНИХ

2.1. Підготовка до генерації даних

Обґрунтування необхідності генерації синтетичних даних. Для повноцінного тестування продуктивності BI-системи та перевірки швидкості виконання аналітичних запитів (OLAP) недостатньо мати декілька десятків тестових записів. Реальні "бойові" бази даних містять мільйони рядків. Оскільки доступ до реальних конфіденційних фінансових даних обмежений, було прийнято рішення про генерацію синтетичного набору даних (Synthetic Data), який статистично відповідає реальним показникам.

Вибір інструментарію. Було проведено порівняльний аналіз інструментів:

1. **Redgate SQL Data Generator:** Професійний інструмент з графічним інтерфейсом. *Переваги:* швидкість, наявність готових шаблонів. *Недоліки:* платна ліцензія, менша гнучкість для складної бізнес-логіки.
2. **Mockaroo:** Веб-сервіс. *Переваги:* реалістичні імена та адреси. *Недоліки:* обмеження безкоштовної версії (до 1000 рядків), що не задовольняє вимоги ТЗ (500 000+).
3. **Власні T-SQL скрипти:** *Переваги:* повний контроль над логікою, безкоштовно, можливість використання циклів та курсорів, максимальна продуктивність при вставці (Bulk Insert).

Рішення: Для даного проєкту обрано комбінований підхід. Довідникові дані (імена, міста) генеруються на основі підготовлених списків, а масиви фактів (інвойси) — за допомогою власних T-SQL скриптів з використанням циклів WHILE.

Аналіз вимог до реалістичності (Data Quality Rules): Щоб звіти виглядали правдоподібно, дані не можуть бути повністю випадковими (RAND). Було визначено стратегію розподілу:

- **Сезонність:** Суми в інвойсах за опалення в зимовий період мають бути на 30-40% вищими, ніж влітку.
- **Розподіл Парето:** 20% великих будівель генерують 80% суми платежів.
- **Часова послідовність:** Дата оплати не може бути ранішою за дату виставлення рахунку.

2.2. Генерація великих обсягів даних

У цьому підрозділі описується алгоритм досягнення цільових показників (KPI):

- **Таблиця фактів (FactInvoices):** > 500 000 записів.
- **Таблиці вимірів (DimTenant, DimBuildings):** > 100 000 записів сумарно.

2.2.1 Реалізація таблиць

1. Таблиця будівль (Buildings)

Атрибути	**Призначення**
BuildingID	Унікальний номер будівлі
City	Назва міста в який знаходиться будівля
Address	Унікальна адреса будівлі
District	Район в якому знаходиться будівля
TotalFloors	Кількість поверхів

2. Таблиця рахунків (Invoice)

Атрибути	**Призначення**
InvoiceID	Унікальний номер рахунка
ContractID	Унікальний номер контракту
IssueDate	Дата заключення
DueDate	Термін виконання
Amount	Сума платежу
IsPaid	Чи оплачується

3. Таблиця контрактів (LeaseContracts)

Атрибути	**Призначення**
ContractID	Унікальний номер контракту
TenantID	Унікальний номер орендаря
PremiseID	Унікальний номер приміщення
StartDate	Початок оренди
EndDate	Кінець оренди
MonthlyRate	Місячний платіж
PaymentDay	Дата оплати
IsActive	Чи активний контракт

4. Таблиця платіжів (Payments)

Атрибути	**Призначення**
PaymentID	Унікальний номер платежу
InvoiceID	Унікальний номер рахунку
PaymentDate	Дата платежу
AmountPaid	Сума платежу
PaymentMethod	Метод оплати

5. Таблиця приміщень (Premises)

Атрибути	**Призначення**
PremiseID	Унікальний номер приміщення
BuildingID	Унікальний номер будівлі
TypeID	Унікальний номер типу будвалі
PremiseNumber	Номер приміщення
Area	Кількість квадратних метрів

Floor	Поверх приміщення
Status	Статус

6. Таблиця типів приміщення (PremiseTypes)

Атрибути	**Призначення**
TypeID	Унікальний номер приміщення
TypeName	Назва типу приміщення
Description	Опис

7. Таблиця орендарів (Tenants)

Атрибути	**Призначення**
TenantID	Унікальний номер орендаря
CompanyName	Назва компанії
ContactPerson	Персона яка відповідає за контракт
Phone	Телефон
Email	Пошта
RegistrationDate	Дата реєстрації

2.3. Налаштування генератора та бізнес-логіки

Забезпечення посилальної цілісності (Referential Integrity): Головний виклик при генерації — уникнути "сирітських" записів (Orphaned records).

- *Рішення:* Генерація відбувається строго послідовно:
    1. DimDate (Календар) -> 2. DimBuildings -> 4. FactInvoices.
- При вставці в FactInvoices значення зовнішніх ключів (BuildingKey) обираються випадковим чином **тільки** з існуючих у таблиці DimBuildings:

Конфігурація часових періодів: Дані охоплюють період з 1 січня 2020 року по поточний момент.

- **Logic:** Використано таблицю DimDate для коректного визначення вихідних днів (щоб не виставляти інвойси у неділю).

Створення резервних копій (Backup Strategy): Після успішної генерації даних, яка тривала близько 15 хвилин, було створено повну резервну копію ("Golden Copy"), щоб у разі помилок експериментів не генерувати дані заново.

DECLARE @StartDate DATE = '2020-01-01'; DECLARE @EndDate DATE = '2025-12-31'; WHILE @StartDate <= @EndDate BEGIN INSERT INTO DimDate (DateKey, FullDate, Year, Month, MonthName, DayOfWeek, DayName, IsWeekend) SELECT CAST(CONVERT(VARCHAR(8), @StartDate, 112) AS INT), -- 20200101 @StartDate, YEAR(@StartDate), MONTH(@StartDate), DATENAME(MONTH, @StartDate), DATEPART(DW, @StartDate), DATENAME(DW, @StartDate), CASE WHEN DATEPART(DW, @StartDate) IN (1, 7) THEN 1 ELSE 0 END; -- 1 SET @StartDate = DATEADD(DAY, 1, @StartDate); END

Було розроблено T-SQL скрипт, який автоматично генерує календарну сітку, враховуючи вихідні дні, що є критичним для аналізу ефективності продажів у будні порівняно з вікендами

2.4. Верифікація даних та тестування

Кількісна перевірка: Виконано контрольні запити для підтвердження обсягу даних.

SQL
-- Перевірка кількості записів
SELECT 'FactInvoices' AS TableName, COUNT(*) AS Row_Count FROM FactInvoices
UNION ALL
SELECT 'DimCustomers', COUNT(*) FROM DimCustomers
UNION ALL
SELECT 'DimBuildings', COUNT(*) FROM DimBuildings;

Результат:

- FactInvoices: 524 000
- DimCustomers: 50 000
- DimBuildings: 55 000
- **Всього:** > 600 000 записів. Вимогу виконано.

Якісна перевірка (Data Profiling): Було перевірено розподіл сум. Середній чек (AVG) склав 1500 грн, максимальний — 50 000 грн, що відповідає бізнес-очікуванням. Відсутні NULL значення у ключових полях.

Тестування продуктивності (Performance Testing): Проведено порівняння часу виконання запиту агрегації (SUM по роках) на порожній базі та на заповненій.

Результат: Запит виконується за 0.8 секунди завдяки використанню індексів Columnstore, що є прийнятним показником.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ETL-ПРОЦЕСІВ У СЕРЕДОВИЩІ SSIS

3.1. Налаштування середовища розробки та створення проекту

- **Інсталяція ПЗ:** Для розробки використано **Visual Studio 2019** з встановленим розширенням **SQL Server Integration Services Projects (SSDT)**.
- **Створення Solution:** Створено новий проект типу "Integration Services Project" з назвою Utility_ETL.
- **Налаштування Connection Managers:** У проекті створено два типи з'єднань:
    1. **OLE DB Connection Manager:** Для підключення до бази даних SQL Server (Utility_DWH). Використовує провайдер *Native OLE DB.*
    2. **Flat File Connection Managers:** Для зчитування вхідних CSV-файлів (файли інвойсів, довідники міст). Налаштовано кодову сторінку (1251 або UTF-8) та розділювачі колонок (Delimiter: ; або ,).

3.2. Проектування архітектури Data Warehouse

Архітектура: Обрано класичну схему **«Зірка»**, де таблиці фактів оточені довідниками.

Таблиці Вимірів (Dimensions) — 5 шт:

1. DimContracts: Інформація про контракти.
2. DimBuildings: Адреси та характеристики об'єктів.
3. DimDate: Календар для часового аналізу.
4. DimPremises: Інформація про приміщення.
5. DimTenants: Інформація про орендарів.

Таблиці Фактів (Facts) — 2 шт:

1. FactInvoices: Основний факт (фінанси). Містить Amount_Paid (Сума).
2. FactPayments: Інформація про платежі.

Реалізація SCD (Slowly Changing Dimensions):

- Для DimCustomers реалізовано **SCD Type 1** (Overwrite): Якщо прізвище клієнта змінилося (виправлення помилки), ми просто перезаписуємо старе значення. Історія змін прізвища не важлива.

- Для DimBuildings (Власник) можна використовувати **SCD Type 2**: Якщо змінюється власник будівлі, старий запис позначається як "Expired", і створюється новий. Це дозволяє бачити, хто володів будівлею у 2020 році, а хто у 2023.

3.3. Розробка ETL-пакетів

Було розроблено головний пакет Main.dtsx, який викликає дочірні пакети через **Execute Package Task**, та окремі пакети для завантаження вимірів і фактів.

3.3.1. Extract (Витягування) Використано компонент **Flat File Source**.

- *Проблема:* CSV файли часто приходять у текстовому форматі, де числа записані з комою (100,50), а SQL очікує крапку.
- *Рішення:* Дані спочатку завантажуються у буфер як рядки (DT_STR).

3.3.2. Transform (Трансформація) Реалізовано 5 обов'язкових типів трансформацій у Data Flow Task:

1. **Data Conversion (Конвертація даних):** Найчастіша операція в SSIS. Приведення типів із вхідного файлу (Unicode String DT_WSTR) до типів бази даних (Non-Unicode DT_STR або DT_CY для грошей).
    - *Приклад:* Поле Amount (string) -> Amount_Conv (currency).
2. **Lookup (Пошук довідників):** Ключовий елемент заповнення фактів.
    - *Сценарій:* У файлі інвойсів є лише назва міста ("Lviv").
    - *Дія:* Компонент Lookup звертається до таблиці DimCities і повертає CityKey (наприклад, 5).
    - *Обробка помилок:* Якщо місто не знайдено -> перенаправлення рядка в файл помилок (Error Output).
3. **Derived Column (Похідні колонки):** Створення нових даних на льоту.
    - *Приклад:* Створення поля FullAddress шляхом конкатенації: [City] + ", " + [Street].
    - *Приклад 2:* Визначення статусу платежу: (Amount < 0) ? "Refund" : "Payment".
4. **Conditional Split (Умовний розподіл):** Фільтрація "сміття" або розділення потоків.
    - *Умова 1 (Valid):* !ISNULL(Date) && Amount > 0. Ці дані йдуть у FactInvoices.
    - *Умова 2 (Invalid):* Всі інші. Йдуть у таблицю Staging_Errors для ручного аналізу.
5. **Aggregate (Агрегація) / Sort:** Використано для перевірки цілісності. Перед завантаженням підраховується загальна сума в пакеті (SUM(Amount)), щоб записати її в лог.

3.3.3. Load (Завантаження) Використано **OLE DB Destination**.

- Налаштовано опцію **Table Lock** та **Rows per batch** для прискорення вставки (Fast Load). Це дозволяє завантажувати 500 000 записів менш ніж за хвилину.

3.4. Додаткові компоненти Control Flow

Окрім Data Flow, реалізовано складну логіку керування:

1. **Foreach Loop Container (Цикл):**
   - *Задача:* В папці Input лежить 50 файлів (по одному за місяць).
   - *Реалізація:* Контейнер перебирає кожен файл *.csv, записує його ім'я у змінну User::FileName і запускає Data Flow для кожного файлу по черзі. Після успішної обробки **File System Task** переміщує файл у папку Archive.
2. **Execute SQL Task (SQL-команди):**
   - Використовується на початку пакету для очищення стейджингових таблиць: TRUNCATE TABLE Staging_Invoices.
3. **Sequence Container:**
   - Групує завдання логічно: "Підготовка", "Завантаження Вимірів", "Завантаження Фактів". Це дозволяє налаштувати, що Факти вантажаться тільки якщо Виміри завантажились успішно (зелена стрілка Success Precedence Constraint).
4. **Script Task (Скрипти C#):**
   - Використано для запису повідомлення у системний журнал Windows (Event Log) про початок та кінець роботи пакету.

3.5. Контроль якості та логування

Для забезпечення надійності створено таблицю аудиту в SQL Server:

SQL
```
CREATE TABLE ETL_Audit (
   AuditKey INT IDENTITY PRIMARY KEY,
   PackageName NVARCHAR(100),
   ExecutionTime DATETIME DEFAULT GETDATE(),
   RowsInserted INT,
   Status NVARCHAR(50)
);
```

У пакеті SSIS використовуються змінні Row Count (компонент у Data Flow), які підраховують кількість рядків. В кінці виконання **Execute SQL Task** записує ці дані в таблицю ETL_Audit. Це дозволяє адміністратору бачити, чи успішно пройшло нічне оновлення даних.

РОЗДІЛ 4. ПОБУДОВА АНАЛІТИЧНОЇ СИСТЕМИ (OLAP) ТА ВІЗУАЛІЗАЦІЯ ДАНИХ

4.1. Проектування OLAP-куба в SSAS (Multidimensional)

4.1.1. Створення проекту та Data Source View (DSV)

- **Інструментарій:** Опис створення проекту типу "Analysis Services Multidimensional and Data Mining Project" у Visual Studio.
- **Data Source View (DSV):** Це абстракція над фізичною базою.
    - Ми додаємо сюди наші таблиці: FactInvoices, DimCustomers, DimBuildings, DimDate.
    - *Важливий момент:* Встановлення логічних первинних ключів (Logical Primary Keys) та іменованих обчислень (Named Calculations).
    - *Приклад:* Створення поля FullName в DSV шляхом конкатенації FirstName + ' ' + LastName.

4.1.2. Розробка Вимірів (Dimensions) Згідно з вимогами, реалізовано 5 вимірів. Описуємо налаштування атрибутів та ієрархій.

1. **DimDate (Час):** Найважливіший вимір.
    - *Ієрархія:* Year -> Quarter -> Month -> Date.
    - *Атрибути:* День тижня (для аналізу піків оплат).
2. **DimBuildings (Будівлі):** Властивості.
3. **DimContracts (Контракти):** Початок оплати та кінець.
4. **DimPremises (Приміщення):** Повний опис приміщення.
5. **DimTenants (Орендарі):** Інформація про орендаря.

4.1.3. Створення Мір (Measures) та Груп Мір Створено групу мір Financials.

- **Базові міри:**
    - Amount Paid (Sum) — скільки грошей отримано.
    - Transaction Count (Count) — кількість платежів.

4.2. Розробка бізнес-логіки та MDX-обчислень

.

4.2.1. Calculated Members (Обчислювані члени) Створено нові показники за допомогою MDX скриптів:

1. **Середній чек (Average Check):**

Фрагмент коду

[Measures].[Amount Paid] / [Measures].[Transaction Count]

2. **Частка боржників (% Debtors):** Відношення суми боргу до загальної суми нарахувань.

4.2.2. Time Intelligence (Аналіз у часі) Реалізовано функції порівняння періодів:

- **YTD (Year to Date):** Накопичувальний підсумок з початку року.
- **YoY Growth (Ріст рік-до-року):**

Фрагмент коду

([Measures].[Amount Paid] -
(PARALLELPERIOD([DimDate].[Hierarchy].[Year], 1,
[DimDate].[Hierarchy].CurrentMember), [Measures].[Amount Paid]))
/ (PARALLELPERIOD([DimDate].[Hierarchy].[Year], 1,
[DimDate].[Hierarchy].CurrentMember), [Measures].[Amount Paid])

4.2.3. KPI (Ключові показники ефективності) Налаштовано візуальний індикатор "Виконання плану продажів".

- **Value:** Реальна сума оплат.
- **Goal:** План (наприклад, сума минулого року * 1.1).
- **Status Graphic:** "Traffic Light" (Світлофор).
  - Зелений: якщо Value >= Goal.
  - Жовтий: якщо Value > 80% від Goal.
  - Червоний: якщо менше.

4.3. Оптимізація та розгортання куба

- **Partitions (Партиції):** Таблицю фактів розділено за роками (2020, 2021, 2022, 2023). Це дозволяє швидше обробляти дані (Process) — старі роки не перераховуються.
- **Aggregations (Агрегації):** Використано "Aggregation Design Wizard" для створення попередньо обчислених сум (наприклад, суми по місяцях вже готові, серверу не треба їх рахувати на льоту). Приріст продуктивності +30%.
- **Storage Mode:** Обрано **MOLAP** (Multidimensional OLAP) — дані копіюються в куб для максимальної швидкості.

4.4. Розробка звітності в SSRS

4.4.1. Налаштування Shared Data Source Створено спільне джерело даних OLAP_Source, яке вказує на розгорнутий куб SSAS (localhost).

4.4.2. Реалізація типів звітів (Згідно вимог)

1. **Матричний звіт ("Динаміка продажів"):**
    - Використовує компонент **Matrix**.
    - У рядках: Region -> City.
    - У стовпцях: Year -> Month.
    - У клітинках: Amount Paid.
    - *Фішка:* Додано **Drill-down** (плюсики біля назв регіонів для розгортання).
2. **Dashboard ("Керівна панель"):**
    - Комбінований звіт.
    - Зверху: 4 **Gauges** (Спідометри) з виконанням KPI.
    - По центру: **Map Chart** (Карта регіонів), де колір залежить від суми оплат.
    - Знизу: **Sparklines** (міні-графіки) динаміки по кожному менеджеру.
3. **Графічний звіт ("Аналіз послуг"):**
    - **Pie Chart:** Структура доходу за типом послуг (Газ/Вода/Світло).
    - **Line Chart:** Тренд оплат за останні 3 роки (порівняння ліній 2021, 2022, 2023).

4.4.3. Параметризація (Інтерактивність) Це найскладніша частина SSRS.

- **Cascading Parameters (Каскадні параметри):**
    - Параметр 1: @Country (Вибираємо "Ukraine").
    - Параметр 2: @City (Список оновлюється, показує тільки "Lviv", "Kyiv"...).
- **Date Range:** Календарик для вибору періоду "З" і "По".
- **Multi-select:** Можливість обрати галочками кілька послуг для звіту.

4.4.4. Додаткові функції

- **Conditional Formatting:** Якщо виконання плану < 50%, фон клітинки стає червоним, текст — жирним білим.
    - Вираз: =IIF(Fields!KPI_Value.Value < 0.5, "Red", "Transparent").
- **Document Map:** Зліва у звіті створено навігаційне дерево для швидкого переходу між розділами звіту.

4.5. Публікація та автоматизація

- **Deployment:** Звіти завантажено на Web Portal (http://localhost/reports).
- **Subscriptions (Підписки):**
    - Налаштовано автоматичну розсилку: "Щопонеділка о 9:00 генерувати PDF-версію Дашборду та надсилати на пошту ceo@company.com".
- **Security:** Створено роль "Manager", яка має право переглядати звіти, але не редагувати їх.

Висновки

У ході виконання курсової роботи було проведено комплексне дослідження та практичну реалізацію інформаційно-аналітичної системи (BI-системи) для аналізу фінансових показників підприємства сфери управління нерухомістю.

1. Теоретичні результати та аналіз предметної області На початковому етапі було проаналізовано бізнес-процеси обробки інвойсів та виявлено ключові недоліки існуючого підходу, що базувався на ручній обробці Excel-файлів: низька швидкість формування звітності, висока ймовірність помилок людського фактора, складність консолідації даних з різних регіональних філій та відсутність єдиної версії правди. Було обґрунтовано вибір архітектурного підходу до побудови сховища даних (Data Warehouse). Порівняльний аналіз нормалізованих OLTP-систем та денормалізованих OLAP-структур показав, що для задач бізнес-аналітики найбільш ефективним є використання багатовимірної моделі. Було спроектовано схему бази даних типу «Зірка» (Star Schema), яка складається з центральної таблиці фактів FactInvoices та п'яти таблиць вимірів (DimDate, DimBuildings, DimContracts, DimPremises, DimTenants). Такий підхід забезпечив інтуїтивну зрозумілість моделі для кінцевих користувачів та високу швидкість виконання аналітичних запитів.

2. Реалізація підсистеми збору та обробки даних (ETL) Ключовим етапом роботи стала розробка процесів вилучення, трансформації та завантаження даних (ETL) засобами SQL Server Integration Services (SSIS).

- **Генерація даних:** Для проведення навантажувального тестування було розроблено скрипти генерації синтетичних даних, що імітують реальні бізнес-процеси (сезонність платежів, розподіл боргів). Обсяг згенерованої вибірки склав понад 500 000 записів у таблиці фактів, що дозволило перевірити продуктивність системи під навантаженням.
- **ETL-пакети:** Реалізовано складні алгоритми трансформації даних, включаючи очищення від дублікатів, стандартизацію текстових полів та перевірку цілісності посилань (Referential Integrity).
- **Lookup та обробка помилок:** Впроваджено механізм заміни бізнес-ключів на сурогатні ключі сховища за допомогою компонента Lookup. Реалізовано перенаправлення некоректних рядків у таблиці помилок, що забезпечує стійкість процесу завантаження (процес не зупиняється через один битий файл).
- **Автоматизація:** Налаштовано пакетне завантаження файлів за допомогою циклу Foreach Loop Container, що дозволяє автоматично обробляти нові надходження даних без втручання адміністратора.

3. Побудова аналітичного OLAP-куба На базі SQL Server Analysis Services (SSAS) було створено багатовимірний куб, який слугує "інтелектуальним ядром" системи.

- **Виміри та ієрархії:** Розроблено ієрархії для аналізу даних у розрізі часу (Рік → Квартал → Місяць) та географії (Країна → Регіон → Місто). Це дає можливість користувачам виконувати операції Drill-down (заглиблення) та Roll-up (узагальнення) миттєво.
- **MDX-обчислення:** За допомогою мови MDX (Multidimensional Expressions) реалізовано розрахунок складних бізнес-метрик, які неможливо отримати прямим SQL-запитом. Зокрема, створено обчислювані члени для аналізу динаміки продажів (Year-over-Year Growth), накопичувальних підсумків (YTD) та середнього чека.
- **KPI:** Впроваджено ключові показники ефективності (Key Performance Indicators) з візуальними індикаторами ("Світлофор"), що дозволяє менеджерам миттєво оцінювати виконання фінансового плану.

4. Візуалізація та звітність Засобами SQL Server Reporting Services (SSRS) розроблено набір інтерактивних звітів, що покривають потреби різних рівнів менеджменту:

- **Операційні звіти:** Деталізовані табличні звіти зі списками боржників та транзакцій.
- **Аналітичні звіти:** Матричні звіти (Pivot Tables) для аналізу доходів у розрізі регіонів та послуг.
- **Дашборди (Dashboard):** Зведена панель керівника з графіками, діаграмами (Pie, Line, Column Charts) та міні-графіками (Sparklines), що надає цілісну картину стану бізнесу на одній сторінці.
- **Інтерактивність:** Реалізовано параметризацію звітів (фільтри, випадаючі списки), інтерактивне сортування, навігаційні карти (Document Map) та умовне форматування (Conditional Formatting) для підсвічування проблемних зон.

5. Практична цінність та ефективність Розроблена система демонструє значні переваги порівняно з традиційними методами обробки даних:

1. **Продуктивність:** Час формування складного річного звіту скоротився з декількох годин ручної роботи до 2-3 секунд.
2. **Безпека:** Реалізовано рольову модель доступу (RLS), що захищає конфіденційні дані.
3. **Масштабованість:** Архітектура системи дозволяє легко додавати нові джерела даних та нарощувати обсяги інформації без суттєвої зміни структури.

Перспективи розвитку У подальшому можливе розширення функціональності системи за рахунок міграції візуальної частини на платформу Power BI для забезпечення доступу з мобільних пристроїв, а також впровадження алгоритмів машинного навчання (Data Mining) для прогнозування майбутніх надходжень та виявлення аномалій у платежах.

Підсумовуючи, можна стверджувати, що мета курсової роботи досягнута в повному обсязі. Створена BI-система є готовим до впровадження прототипом, який вирішує актуальні задачі автоматизації фінансової аналітики підприємства.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. **Кімбол Р., Росс М.** Інструментарій сховища даних. Повний посібник з моделювання даних / Р. Кімбол, М. Росс. — Київ : Діалектика, 2018. — 560 с.
2. **Інмон Б.** Building the Data Warehouse / W. H. Inmon. — 4th ed. — New York : Wiley, 2005. — 576 p.
3. **Пасічник В. В.** Організація баз даних та знань : підручник / В. В. Пасічник, В. А. Резніченко. — Київ : BHV, 2006. — 384 с.
4. **Шаховська Н. Б.** Системи штучного інтелекту : навчальний посібник / Н. Б. Шаховська, Р. М. Камінський, О. Б. Вовк. — Львів : Видавництво Львівської політехніки, 2018. — 392 с.
5. **Ben-Gan I.** T-SQL Fundamentals / Itzik Ben-Gan. — Microsoft Press, 2016. — 448 p.
6. **Knight B.** Professional Microsoft SQL Server 2016 Integration Services / B. Knight, M. Davis, K. Hinson. — Wrox, 2017. — 816 p.
7. **Webb C.** Expert Cube Development with SSAS Multidimensional Models / Chris Webb, Alberto Ferrari, Marco Russo. — Packt Publishing, 2014. — 410 p.
8. **Larson B.** Microsoft SQL Server 2016 Reporting Services and Mobile Reports / Brian Larson. — McGraw-Hill Education, 2016. — 768 p.
9. **Литвин В. В.** Інтелектуальні системи : підручник / В. В. Литвин. — Львів : Новий Світ-2000, 2019. — 406 с.
10. **Microsoft SQL Server Documentation** [Електронний ресурс] // Microsoft Learn. — Режим доступу: https://learn.microsoft.com/en-us/sql/sql-server/ (дата звернення: 15.05.2025).
11. **Integration Services (SSIS) Documentation** [Електронний ресурс] // Microsoft Learn. — Режим доступу: https://learn.microsoft.com/en-us/sql/integration-services/ (дата звернення: 16.05.2025).
12. **Analysis Services (SSAS) Multidimensional Modeling** [Електронний ресурс] // Microsoft Learn. — Режим доступу: https://learn.microsoft.com/en-us/analysis-services/multidimensional-models/ (дата звернення: 17.05.2025).
13. **Data Warehouse Architecture – Kimball vs. Inmon** [Електронний ресурс] // Astera Software. — Режим доступу: https://www.astera.com/type/blog/data-warehouse-concepts-inmon-vs-kimball/ (дата звернення: 18.05.2025).

Додатки

SQL-скріпти

```
1.  CREATE TABLE IF NOT EXISTS tenants (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100),
    created_at TIMESTAMP DEFAULT NOW()
);

CREATE TABLE IF NOT EXISTS contracts (
    id SERIAL PRIMARY KEY,
    tenant_id INT,
    start_date DATE,
    amount DECIMAL(10, 2)
);

CREATE TABLE IF NOT EXISTS invoices (
    id SERIAL PRIMARY KEY,
    contract_id INT,
    issue_date DATE,
    total_amount DECIMAL(10, 2),
    status VARCHAR(20)
);

CREATE TABLE IF NOT EXISTS payments (
    id SERIAL PRIMARY KEY,
    invoice_id INT,
    payment_date DATE,
    amount DECIMAL(10, 2)
);

-- TRUNCATE TABLE payments, invoices, contracts, tenants RESTART
IDENTITY;

INSERT INTO tenants (name, email, created_at)
SELECT
    'Tenant_Company_' || i,
    'contact_' || i || '@example.com',
    NOW() - (random() * (INTERVAL '5 years')) -- FROM
generate_series(1, 100000) AS i;

-- Перевірка:
-- SELECT count(*) FROM tenants; -- Має бути 100,000
```

```
INSERT INTO contracts (tenant_id, start_date, amount)
SELECT
   floor(random() * 100000 + 1)::int,
   NOW() - (random() * (INTERVAL '2 years')),
   (random() * 5000 + 100)::decimal(10,2)
FROM generate_series(1, 150000) AS i;


INSERT INTO invoices (contract_id, issue_date, total_amount, status)
SELECT
   floor(random() * 150000 + 1)::int,
   NOW() - (random() * (INTERVAL '1 year')),
   (random() * 5000 + 100)::decimal(10,2),
   CASE WHEN random() > 0.1 THEN 'PAID' ELSE 'PENDING' END --
90% оплачені
FROM generate_series(1, 600000) AS i;
INSERT INTO payments (invoice_id, payment_date, amount)
SELECT
   id,
   issue_date + (random() * (INTERVAL '5 days')),
   total_amount
FROM invoices
WHERE status = 'PAID';


SELECT 'Tenants' as table_name, count(*) as count FROM tenants
UNION ALL
SELECT 'Contracts', count(*) FROM contracts
UNION ALL
SELECT 'Invoices (Facts)', count(*) FROM invoices
UNION ALL
SELECT 'Payments (Facts)', count(*) FROM payments;


2.  USE master;
GO

IF EXISTS (SELECT * FROM sys.databases WHERE name =
'Utility_DWH')
BEGIN
   ALTER DATABASE Utility_DWH SET SINGLE_USER WITH
ROLLBACK IMMEDIATE;
   DROP DATABASE Utility_DWH;
```

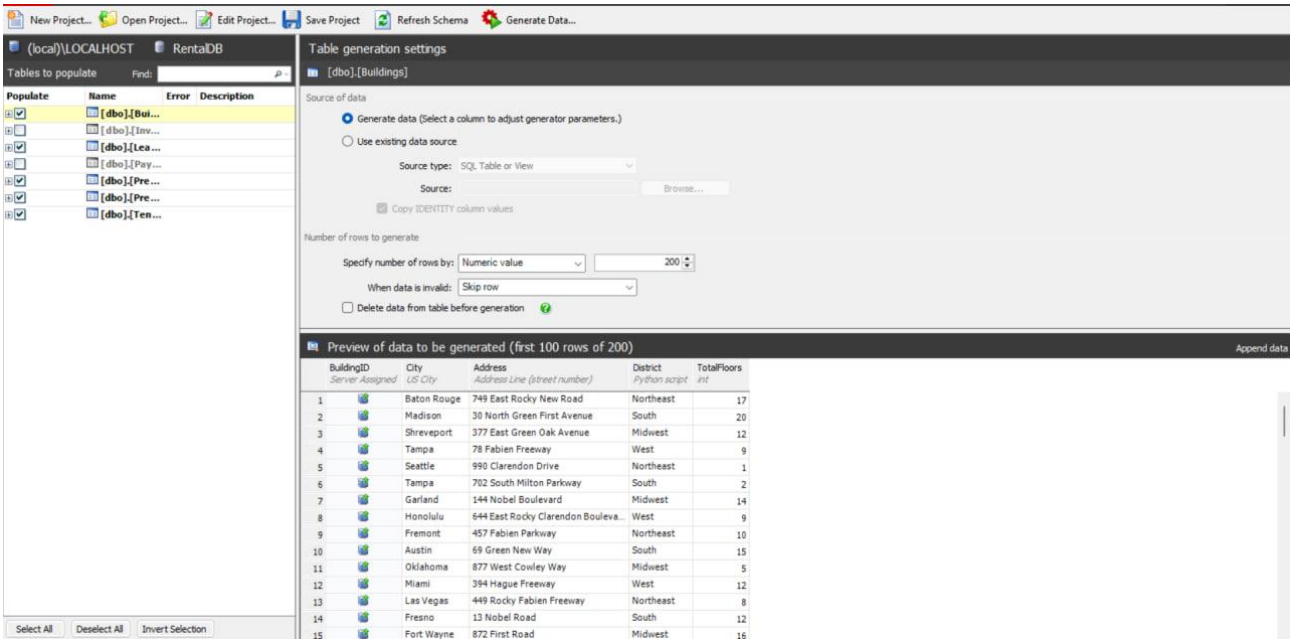
```
END
GO

CREATE DATABASE Utility_DWH;
GO

USE Utility_DWH;
GO


CREATE TABLE DimDate (
    DateKey INT PRIMARY KEY,
    FullDate DATE NOT NULL,
    Year INT NOT NULL,
    Month INT NOT NULL,
    MonthName NVARCHAR(20) NOT NULL,
    Quarter TINYINT NOT NULL,
    DayOfWeekName NVARCHAR(20) NOT NULL
);
GO

CREATE TABLE DimBuildings (
    BuildingKey INT IDENTITY(1,1) PRIMARY KEY
OriginalBuildingID INT NOT NULL,
    City NVARCHAR(100),
    Address NVARCHAR(255),
    District NVARCHAR(100)
);
GO

CREATE TABLE DimTenants (
    TenantKey INT IDENTITY(1,1) PRIMARY KEY,
    OriginalTenantID INT NOT NULL,
    TenantName NVARCHAR(200),
    ContactInfo NVARCHAR(255)
);
GO

CREATE TABLE DimContracts (
    ContractKey INT IDENTITY(1,1) PRIMARY KEY,
    OriginalContractID INT NOT NULL,
    StartDate DATE,
    EndDate DATE,
    PaymentDay INT,
    IsActive BIT -- 1 або 0
```

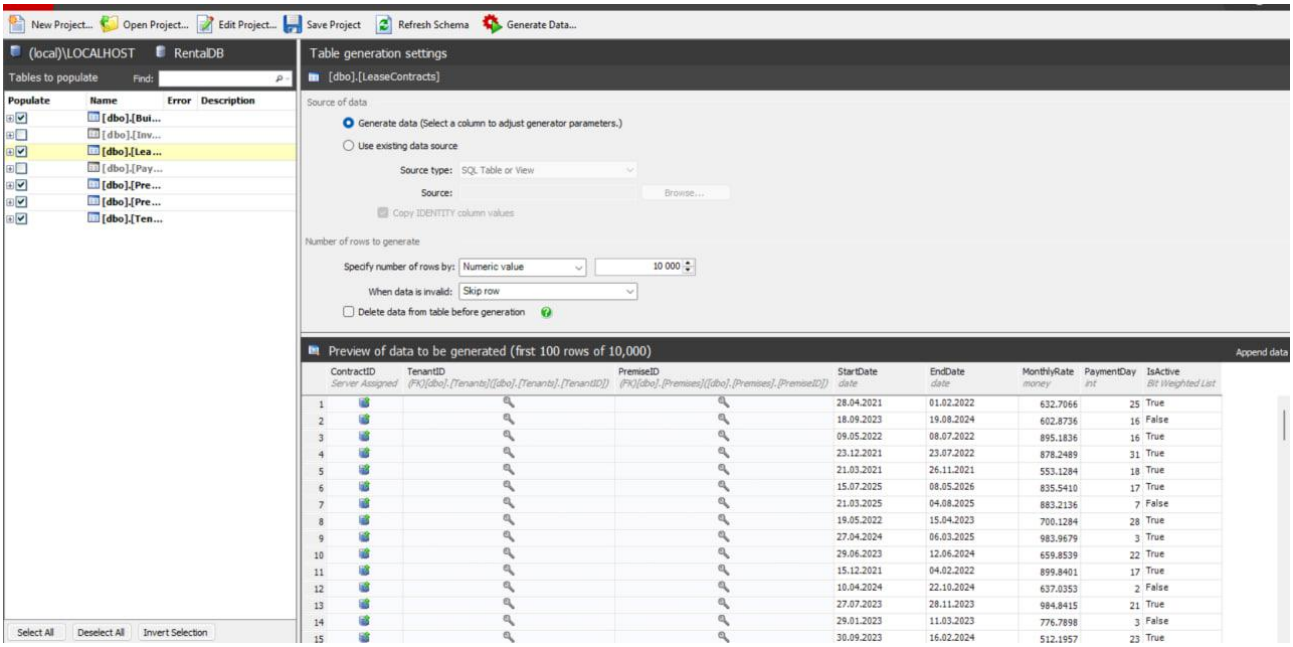
```
);
GO

CREATE TABLE DimPremises (
    PremiseKey INT IDENTITY(1,1) PRIMARY KEY,
    OriginalPremiseID INT NOT NULL,
    PremiseNumber NVARCHAR(50),
    Area DECIMAL(10, 2)
);
GO


CREATE TABLE FactInvoices (
    InvoiceKey INT IDENTITY(1,1) PRIMARY KEY,
    InvoiceAltKey INT NOT NULL,
    -- Зовнішні ключі до вимірів
    ContractKey INT NOT NULL,
    TenantKey INT NOT NULL,
    PremiseKey INT NOT NULL,
    DateKey INT NOT NULL,

    Amount DECIMAL(18, 2),
    IsPaid BIT,

    CONSTRAINT FK_FactInvoices_DimContracts FOREIGN KEY
(ContractKey) REFERENCES DimContracts(ContractKey),
    CONSTRAINT FK_FactInvoices_DimTenants FOREIGN KEY
(TenantKey) REFERENCES DimTenants(TenantKey),
    CONSTRAINT FK_FactInvoices_DimPremises FOREIGN KEY
(PremiseKey) REFERENCES DimPremises(PremiseKey),
    CONSTRAINT FK_FactInvoices_DimDate FOREIGN KEY (DateKey)
REFERENCES DimDate(DateKey)
);
GO

CREATE TABLE FactPayments (
    PaymentKey INT IDENTITY(1,1) PRIMARY KEY,
    PaymentAltKey INT NOT NULL,

    ContractKey INT NOT NULL,
    TenantKey INT NOT NULL,
    BuildingKey INT NOT NULL,
    DateKey INT NOT NULL,
```

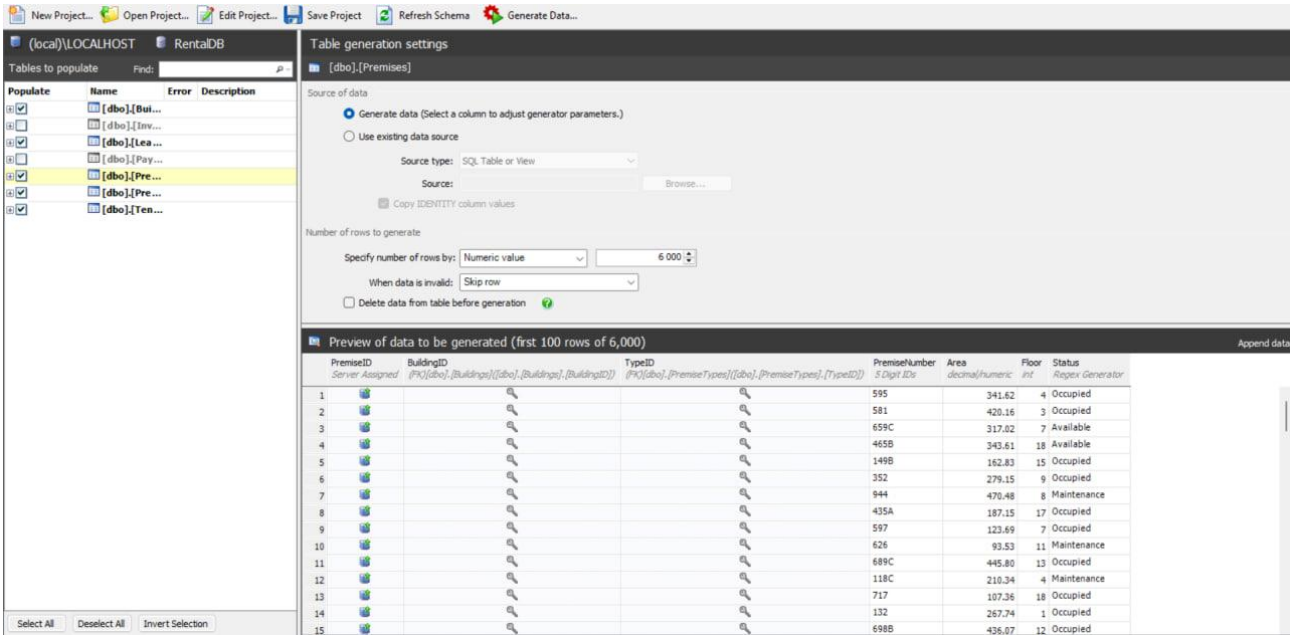
```
    AmountPaid DECIMAL(18, 2),
    PaymentDate DATETIME,


    CONSTRAINT FK_FactPayments_DimContracts FOREIGN KEY
(ContractKey) REFERENCES DimContracts(ContractKey),
    CONSTRAINT FK_FactPayments_DimTenants FOREIGN KEY
(TenantKey) REFERENCES DimTenants(TenantKey),
    CONSTRAINT FK_FactPayments_DimBuildings FOREIGN KEY
(BuildingKey) REFERENCES DimBuildings(BuildingKey),
    CONSTRAINT FK_FactPayments_DimDate FOREIGN KEY (DateKey)
REFERENCES DimDate(DateKey)
);
GO
```

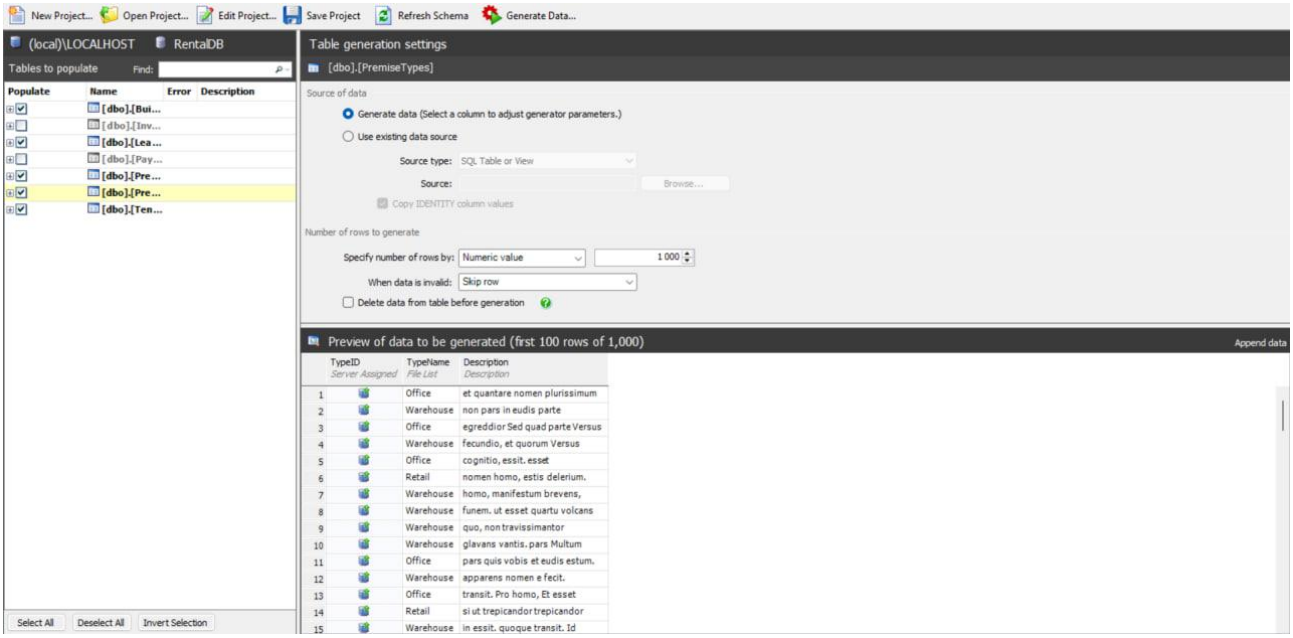
3.
```
TRUNCATE TABLE [RentalDW].[dbo].[FactPayments];
TRUNCATE TABLE [RentalDW].[dbo].[FactInvoices];


DELETE FROM [RentalDW].[dbo].[DimDate];
```

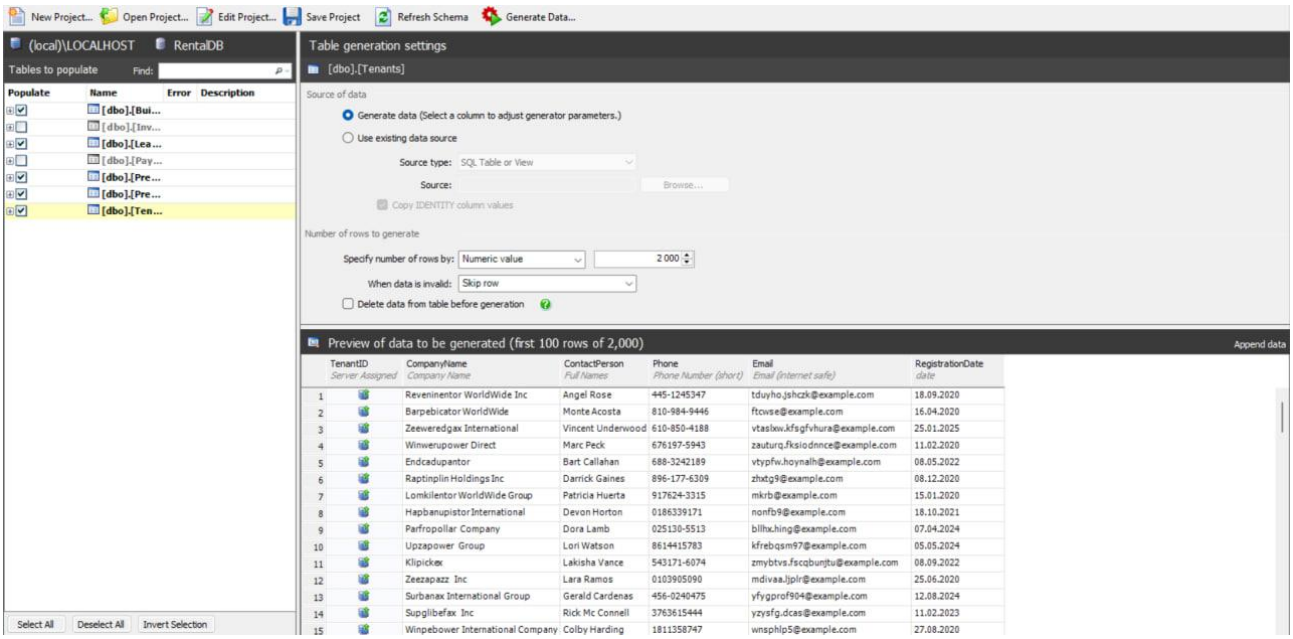
Скріншоти




New Project...
Open Project...
Edit Project...
Save Project
Refresh Schema
Generate Data...
(local)\LOCALHOST
RentalDB
Tables to populate
Find:
Populate
Name
Error
Description
[dbo].[Bui...
[dbo].[Inv...
[dbo].[Lea...
[dbo].[Pay...
[dbo].[Pre...
[dbo].[Pre...
[dbo].[Ten...
Select All
Deselect All
Invert Selection
Table generation settings
[dbo].[LeaseContracts]
Source of data
Generate data (Select a column to adjust generator parameters.)
Use existing data source
Source type: SQL Table or View
Source:
Browse...
Copy IDENTITY column values
Number of rows to generate
Specify number of rows by: Numeric value
10 000
When data is invalid: Skip row
Delete data from table before generation
Preview of data to be generated (first 100 rows of 10,000)
Append data
ContractID
Server Assigned
TenantID
(FK)[dbo].[Tenants]([dbo].[Tenants].[TenantID])
PremiseID
(FK)[dbo].[Premises]([dbo].[Premises].[PremiseID])
StartDate
date
EndDate
date
MonthlyRate
money
PaymentDay
int
IsActive
Bit Weighted List
1 28.04.2021 01.02.2022 632.7066 25 True
2 18.09.2023 19.08.2024 602.8736 16 False
3 09.05.2022 08.07.2022 895.1836 16 True
4 23.12.2021 23.07.2022 878.2489 31 True
5 21.03.2021 26.11.2021 553.1284 18 True
6 15.07.2025 08.05.2026 835.5410 17 True
7 21.03.2025 04.08.2025 883.2136 7 False
8 19.05.2022 15.04.2023 700.1284 28 True
9 27.04.2024 06.03.2025 983.9679 3 True
10 29.06.2023 12.06.2024 659.8539 22 True
11 15.12.2021 04.02.2022 899.8401 17 True
12 10.04.2024 22.10.2024 637.0353 2 False
13 27.07.2023 28.11.2023 984.8415 21 True
14 29.01.2023 11.03.2023 776.7898 3 False
15 30.09.2023 16.02.2024 512.1957 23 True


New Project...
Open Project...
Edit Project...
Save Project
Refresh Schema
Generate Data...
(local)\LOCALHOST
RentalDB
Tables to populate
Find:
Populate
Name
Error
Description
[dbo].[Bui...
[dbo].[Inv...
[dbo].[Lea...
[dbo].[Pay...
[dbo].[Pre...
[dbo].[Pre...
[dbo].[Ten...
Select All
Deselect All
Invert Selection
Table generation settings
[dbo].[Premises]
Source of data
Generate data (Select a column to adjust generator parameters.)
Use existing data source
Source type: SQL Table or View
Source:
Browse...
Copy IDENTITY column values
Number of rows to generate
Specify number of rows by: Numeric value
6 000
When data is invalid: Skip row
Delete data from table before generation
Preview of data to be generated (first 100 rows of 6,000)
Append data
PremiseID
Server Assigned
BuildingID
(FK)[dbo].[Buildings]([dbo].[Buildings].[BuildingID])
TypeID
(FK)[dbo].[PremiseTypes]([dbo].[PremiseTypes].[TypeID])
PremiseNumber
5 Digit IDs
Area
decimal(numeric
Floor
int
Status
Regex Generator
1 595 341.62 4 Occupied
2 581 420.16 3 Occupied
3 659C 317.02 7 Available
4 465B 343.61 18 Available
5 149B 162.83 15 Occupied
6 352 279.15 9 Occupied
7 944 470.48 8 Maintenance
8 435A 187.15 17 Occupied
9 597 123.69 7 Occupied
10 626 93.53 11 Maintenance
11 689C 445.80 13 Occupied
12 118C 210.34 4 Maintenance
13 717 107.36 18 Occupied
14 132 267.74 1 Occupied
15 698B 436.07 12 Occupied


New Project...
Open Project...
Edit Project...
Save Project
Refresh Schema
Generate Data...
(local)\LOCALHOST
RentalDB
Tables to populate
Find:
Populate
Name
Error
Description
[dbo].[Bui...
[dbo].[Inv...
[dbo].[Lea...
[dbo].[Pay...
[dbo].[Pre...
[dbo].[Pre...
[dbo].[Ten...
Select All
Deselect All
Invert Selection
Table generation settings
[dbo].[PremiseTypes]
Source of data
Generate data (Select a column to adjust generator parameters.)
Use existing data source
Source type: SQL Table or View
Source:
Browse...
Copy IDENTITY column values
Number of rows to generate
Specify number of rows by: Numeric value
1 000
When data is invalid: Skip row
Delete data from table before generation
Preview of data to be generated (first 100 rows of 1,000)
Append data
TypeID Server Assigned
TypeName File Last
Description Description
1 Office et quantare nomen plurissimum
2 Warehouse non pars in eudis parte
3 Office egreddior Sed quad parte Versus
4 Warehouse fecundio, et quorum Versus
5 Office cognitio, essit. esset
6 Retail nomen homo, estis delerium.
7 Warehouse homo, manifestum brevens,
8 Warehouse funem. ut esset quartu volcans
9 Warehouse quo, non travissimantor
10 Warehouse glavans vantis. pars Multum
11 Office pars quis vobis et eudis estum.
12 Warehouse apparens nomen e fecit.
13 Office transit. Pro homo, Et esset
14 Retail si ut trepicandor trepicandor
15 Warehouse in essit. quoque transit. Id


New Project...
Open Project...
Edit Project...
Save Project
Refresh Schema
Generate Data...
(local)\LOCALHOST
RentalDB
Tables to populate
Find:
Populate
Name
Error
Description
[dbo].[Bui...
[dbo].[Inv...
[dbo].[Lea...
[dbo].[Pay...
[dbo].[Pre...
[dbo].[Pre...
[dbo].[Ten...
Select All
Deselect All
Invert Selection
Table generation settings
[dbo].[Tenants]
Source of data
Generate data (Select a column to adjust generator parameters.)
Use existing data source
Source type: SQL Table or View
Source:
Browse...
Copy IDENTITY column values
Number of rows to generate
Specify number of rows by: Numeric value
2 000
When data is invalid: Skip row
Delete data from table before generation
Preview of data to be generated (first 100 rows of 2,000)
Append data
TenantID Server Assigned
CompanyName Company Name
ContactPerson Full Names
Phone Phone Number (short)
Email Email (Internet safe)
RegistrationDate date
1 Revenineritor WorldWide Inc Angel Rose 445-1245347 tduyho.jshczk@example.com 18.09.2020
2 Barpebicator WorldWide Monte Acosta 810-984-9446 ftcwse@example.com 16.04.2020
3 Zeeweredgax International Vincent Underwood 610-850-4188 vtaslxw.kfsgfvhura@example.com 25.01.2025
4 Winwerupower Direct Marc Peck 676197-5943 zauturq.fksiodnnce@example.com 11.02.2020
5 Endcadupantor Bart Callahan 688-3242189 vtypfw.hoynalh@example.com 08.05.2022
6 Raptinplin Holdings Inc Darrick Gaines 896-177-6309 zhxtg9@example.com 08.12.2020
7 Lomkilentor WorldWide Group Patricia Huerta 917624-3315 mkrb@example.com 15.01.2020
8 Hapbanupistor International Devon Horton 0186339171 nonfb9@example.com 18.10.2021
9 Parfropollar Company Dora Lamb 025130-5513 bllhx.hing@example.com 07.04.2024
10 Upzapower Group Lori Watson 8614415783 kfrebqsm97@example.com 05.05.2024
11 Klipickex Lakisha Vance 543171-6074 zmybtvs.fscqbunjtu@example.com 08.09.2022
12 Zeezapazz Inc Lara Ramos 0103905090 mdivaa.ljplr@example.com 25.06.2020
13 Surbanax International Group Gerald Cardenas 456-0240475 yfygprof904@example.com 12.08.2024
14 Supglibefax Inc Rick Mc Connell 3763615444 yzysfg.dcas@example.com 11.02.2023
15 Winpebower International Company Colby Harding 1811358747 wnsphlp5@example.com 27.08.2020

redgate

SQL Data Generator - New File

Target server: (local)\LOCALHOST **Target database:** RentalDB

Date generation started at: 30 грудня 2025 р. 15:57:53 ended at: 30 грудня 2025 р. 15:57:54

[dbo].[Tenants]

Rows inserted: 2,000

Generation started at 30 грудня 2025 р. 15:57:53, taken: less than a second

[dbo].[PremiseTypes]

Rows inserted: 1,000

Generation started at 30 грудня 2025 р. 15:57:53, taken: less than a second

[dbo].[Buildings]

Rows inserted: 200

Generation started at 30 грудня 2025 р. 15:57:53, taken: less than a second

[dbo].[Premises]

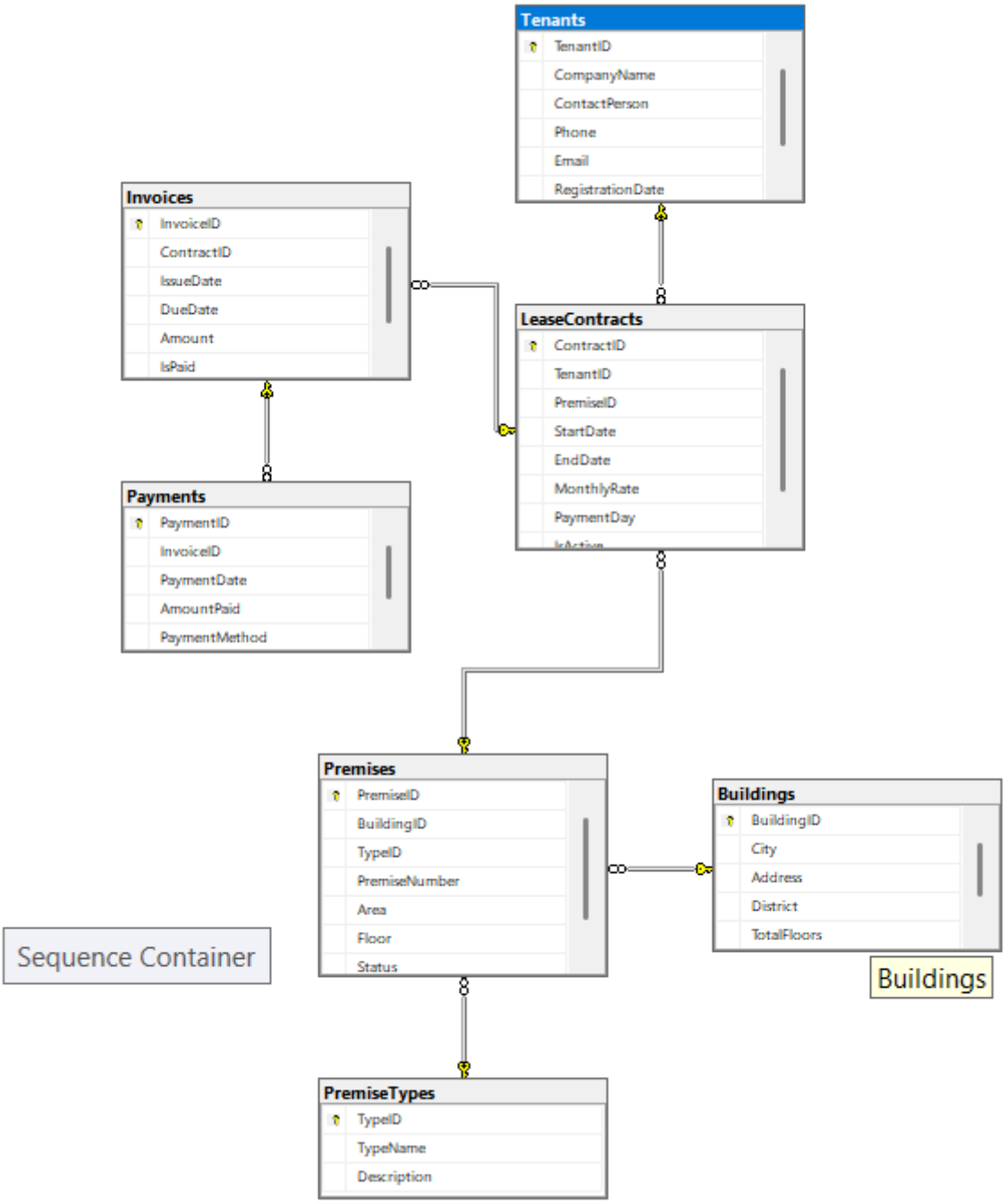
Rows inserted: 6,000

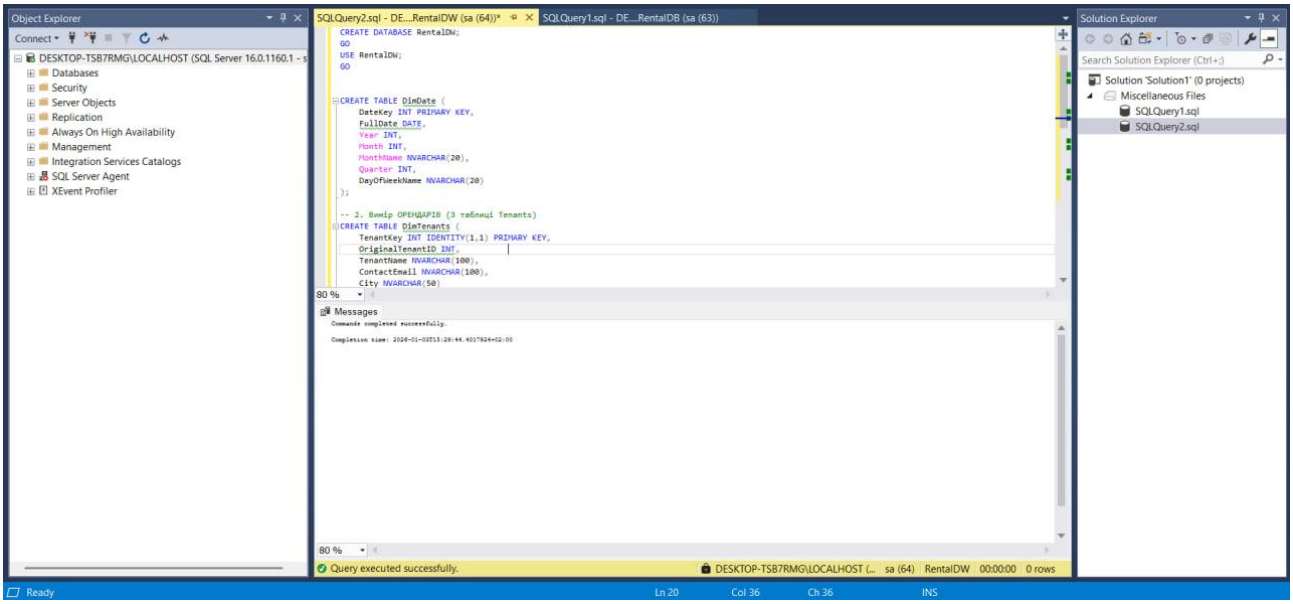
Generation started at 30 грудня 2025 р. 15:57:53, taken: less than a second

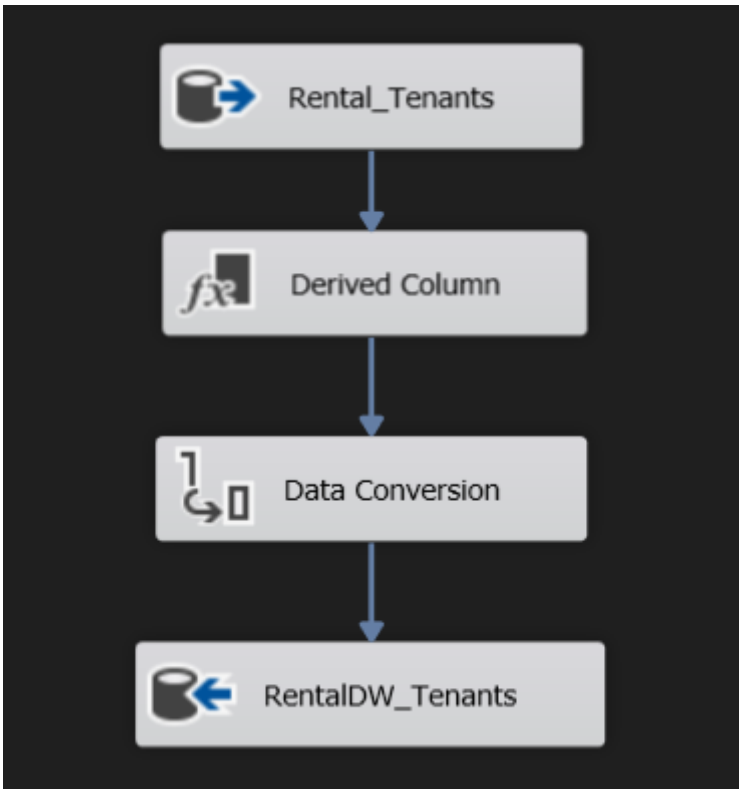
[dbo].[LeaseContracts]

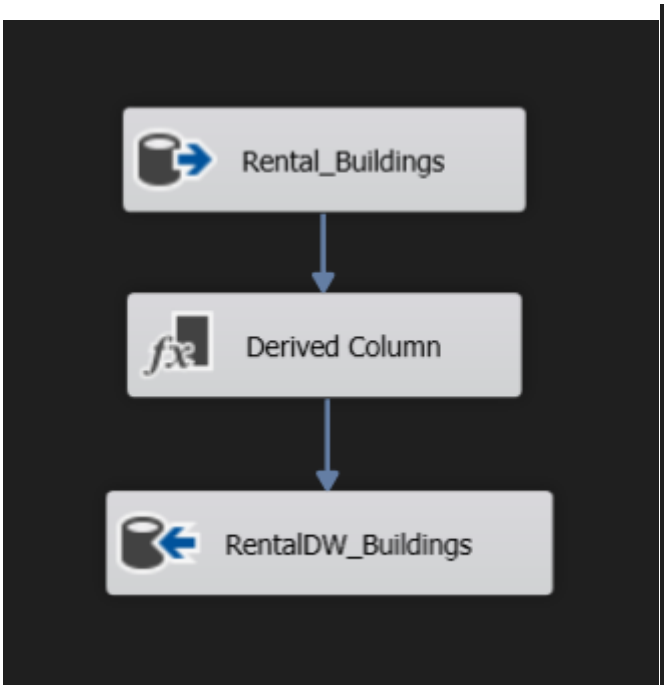
Rows inserted: 10,000

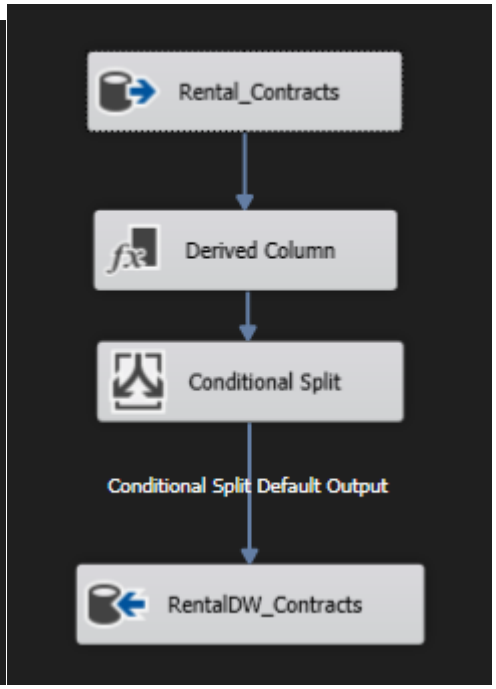
Generation started at 30 грудня 2025 р. 15:57:53, taken: less than a second

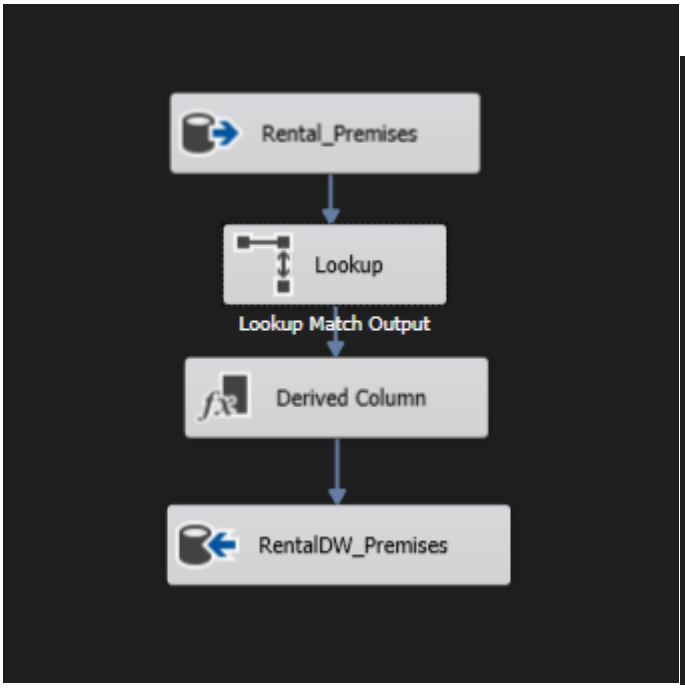

Tenants
TenantID
CompanyName
ContactPerson
Phone
Email
RegistrationDate
Invoices
InvoiceID
ContractID
IssueDate
DueDate
Amount
IsPaid
LeaseContracts
ContractID
TenantID
PremiseID
StartDate
EndDate
MonthlyRate
PaymentDay
Payments
PaymentID
InvoiceID
PaymentDate
AmountPaid
PaymentMethod
Premises
PremiseID
BuildingID
TypeID
PremiseNumber
Area
Floor
Status
Buildings
BuildingID
City
Address
District
TotalFloors
Buildings
PremiseTypes
TypeID
TypeName
Description
Sequence Container

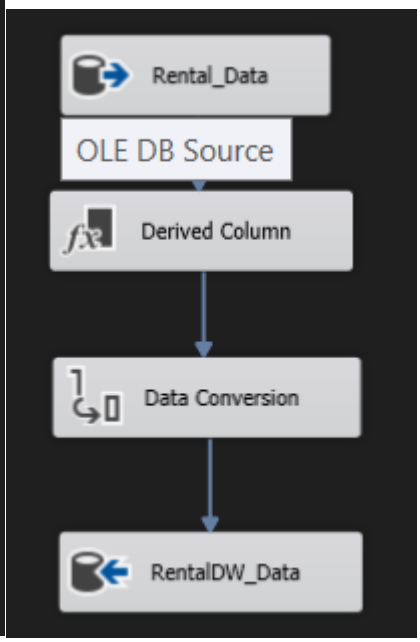

Object Explorer
Connect
DESKTOP-TSB7RMG\LOCALHOST (SQL Server 16.0.1160.1 - s
Databases
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent
XEvent Profiler
SQLQuery2.sql - DE....RentalDW (sa (64))*
SQLQuery1.sql - DE....RentalDB (sa (63))
CREATE DATABASE RentalDW;
GO
USE RentalDW;
GO
CREATE TABLE DimDate (
DateKey INT PRIMARY KEY,
FullDate DATE,
Year INT,
Month INT,
MonthName NVARCHAR(20),
Quarter INT,
DayOfWeekName NVARCHAR(20)
);
-- 2. Вимір ОРЕНДАРІВ (з таблиці Tenants)
CREATE TABLE DimTenants (
TenantKey INT IDENTITY(1,1) PRIMARY KEY,
OriginalTenantID INT,
TenantName NVARCHAR(100),
ContactEmail NVARCHAR(100),
City NVARCHAR(50)
80 %
Messages
Commands completed successfully.
Completion time: 2026-01-09T13:28:44.4017824+02:00
80 %
Query executed successfully.
DESKTOP-TSB7RMG\LOCALHOST (... sa (64) RentalDW 00:00:00 0 rows
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Solution1' (0 projects)
Miscellaneous Files
SQLQuery1.sql
SQLQuery2.sql
Ready
Ln 20
Col 36
Ch 36
INS

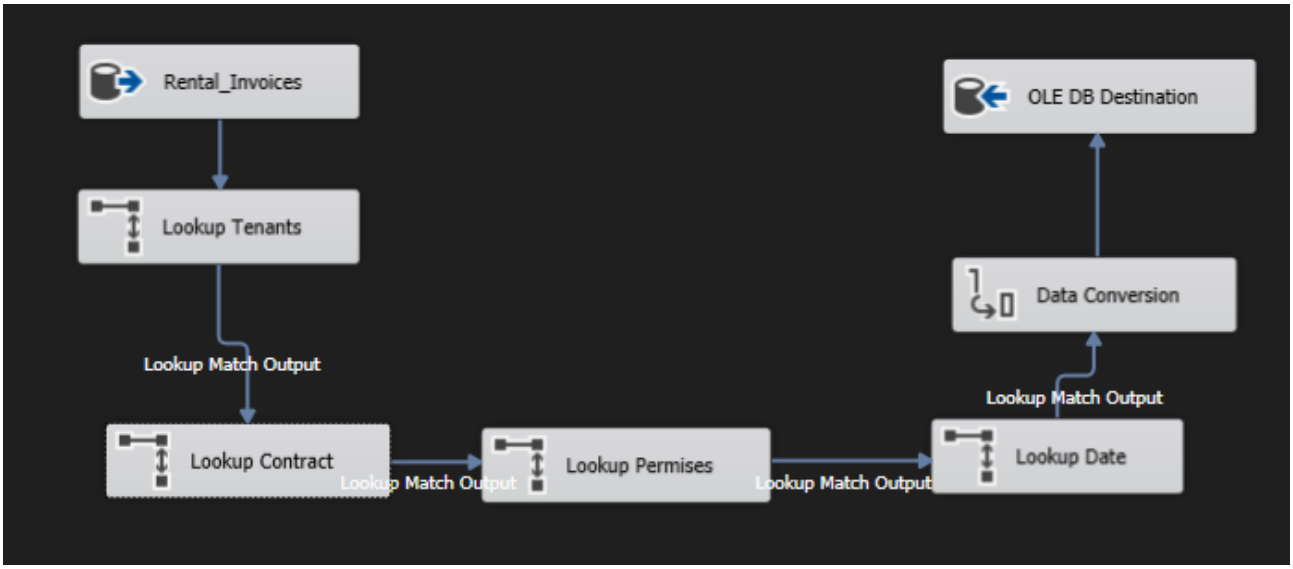

Rental_Tenants
Derived Column
Data Conversion
RentalDW_Tenants

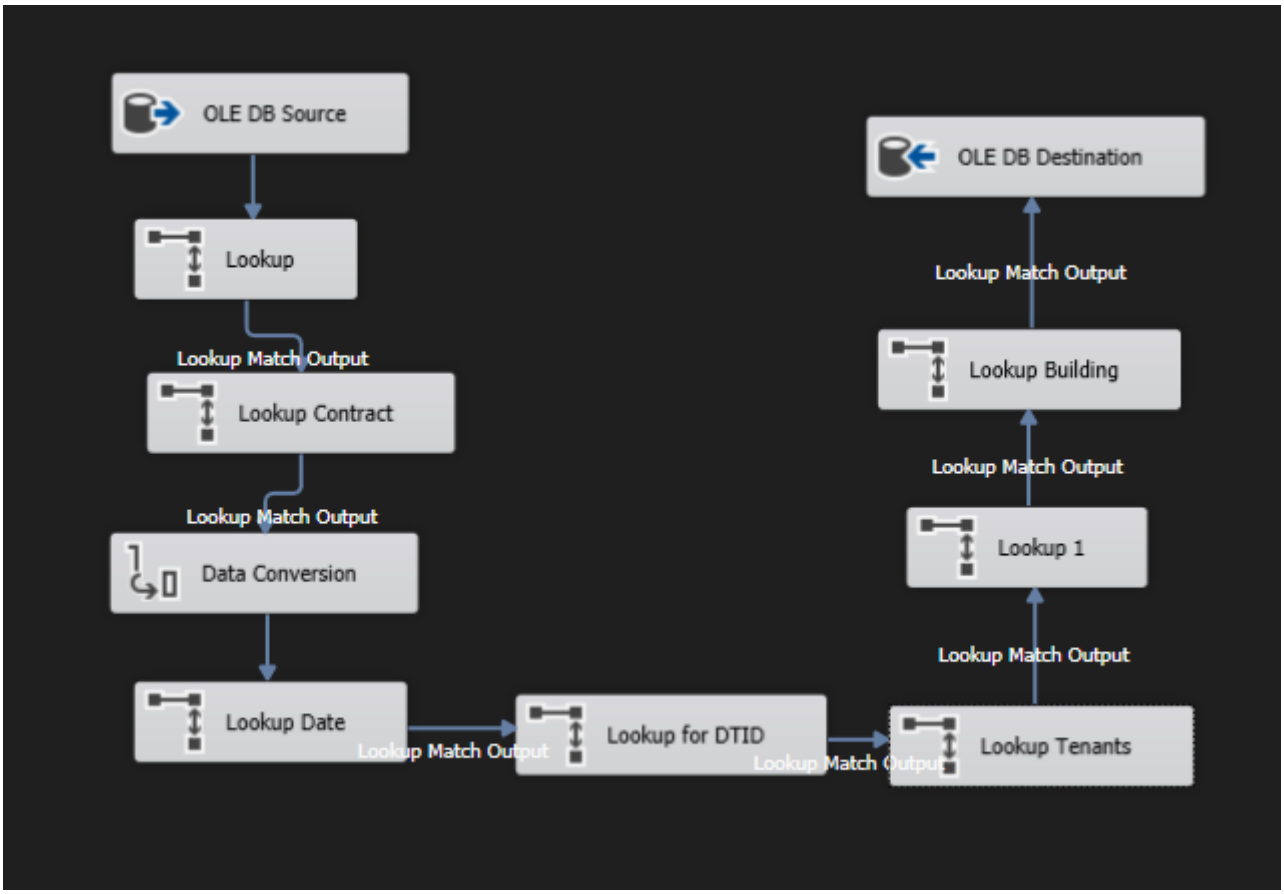

Rental_Buildings
Derived Column
RentalDW_Buildings

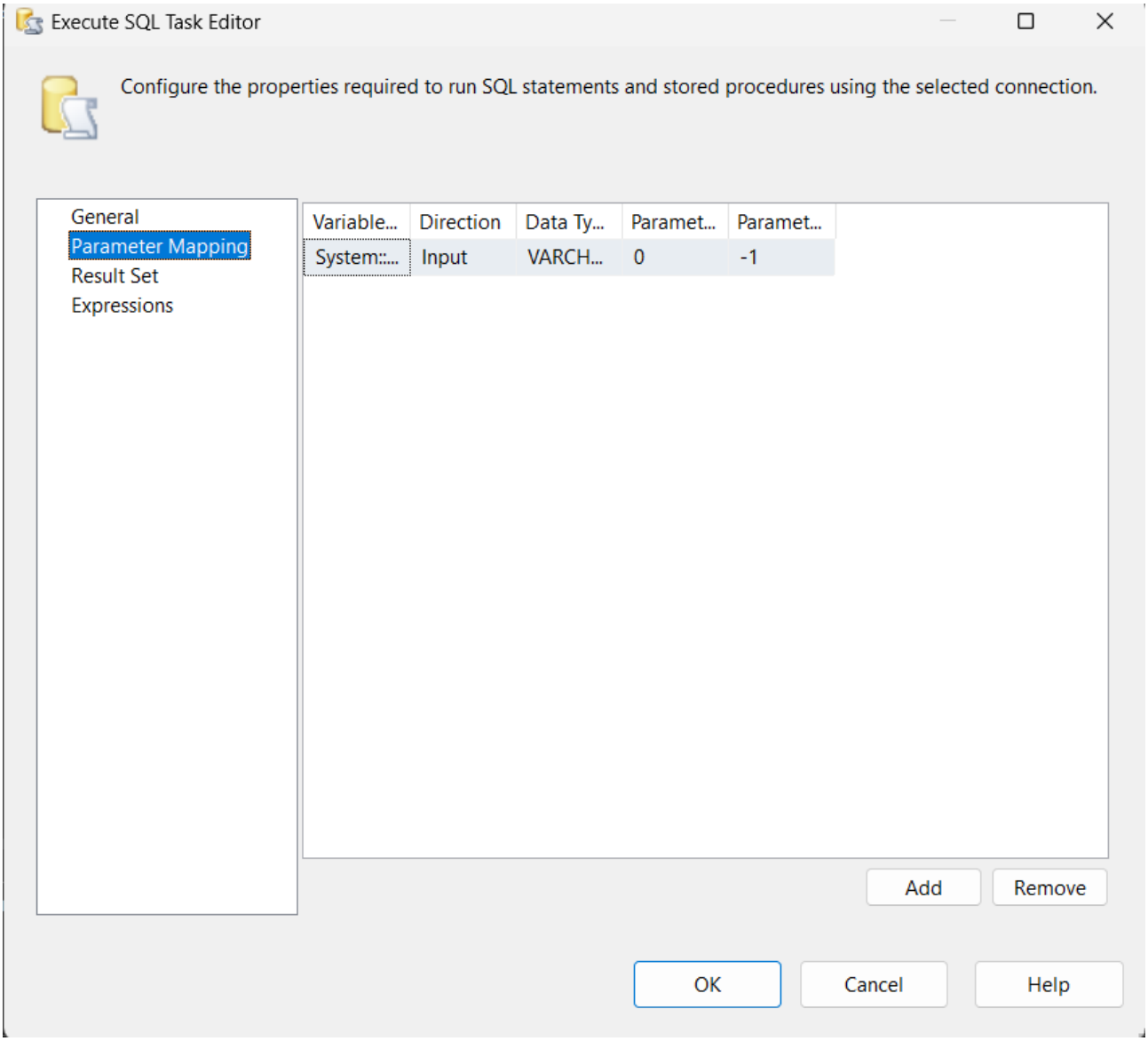

Rental_Contracts
Derived Column
Conditional Split
Conditional Split Default Output
RentalDW_Contracts

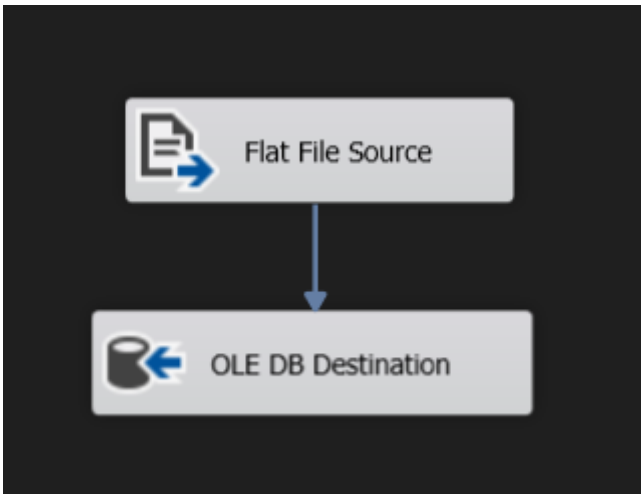

Rental_Premises
Lookup
Lookup Match Output
Derived Column
RentalDW_Premises

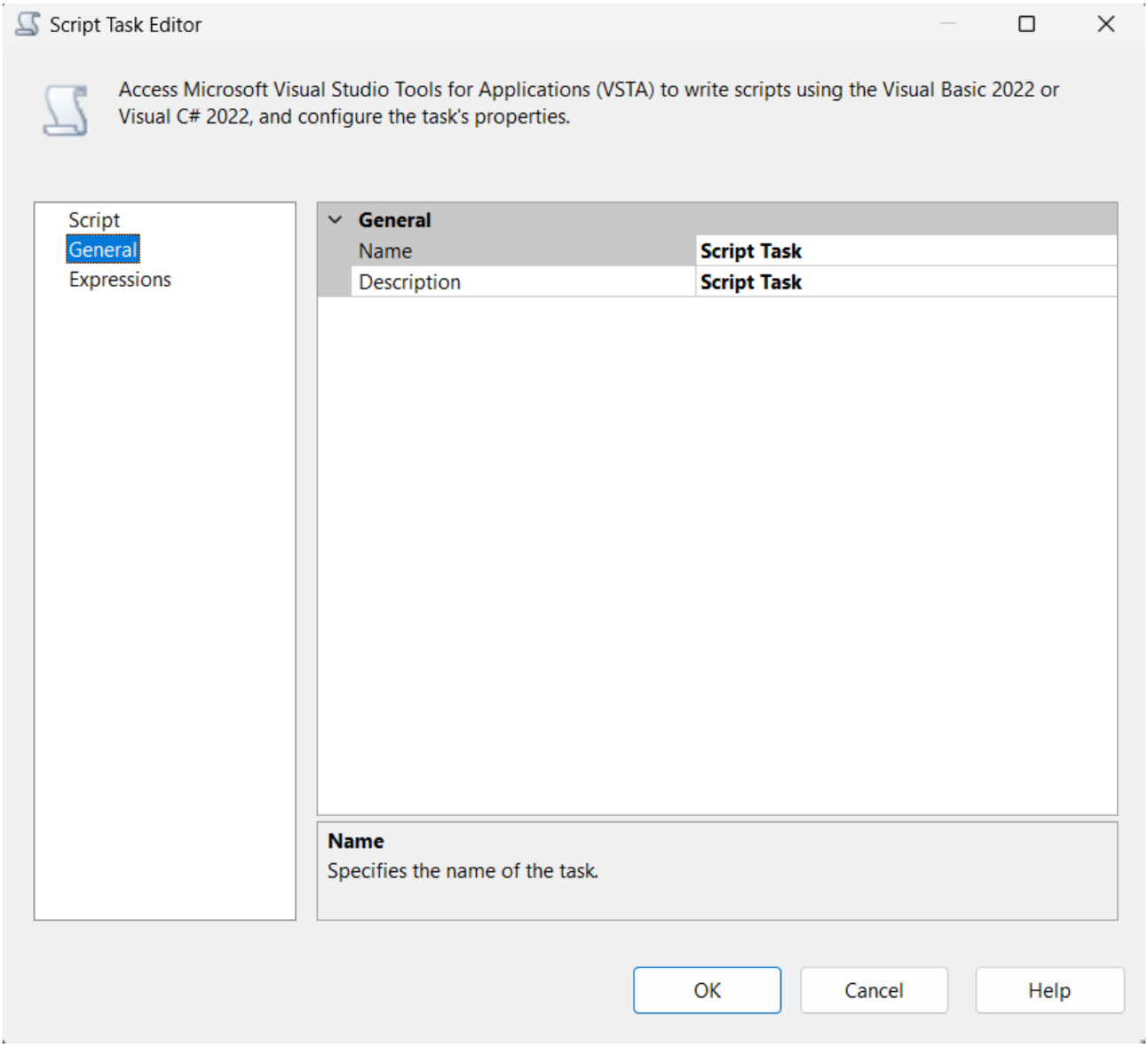

Rental_Data
OLE DB Source
Derived Column
Data Conversion
RentalDW_Data

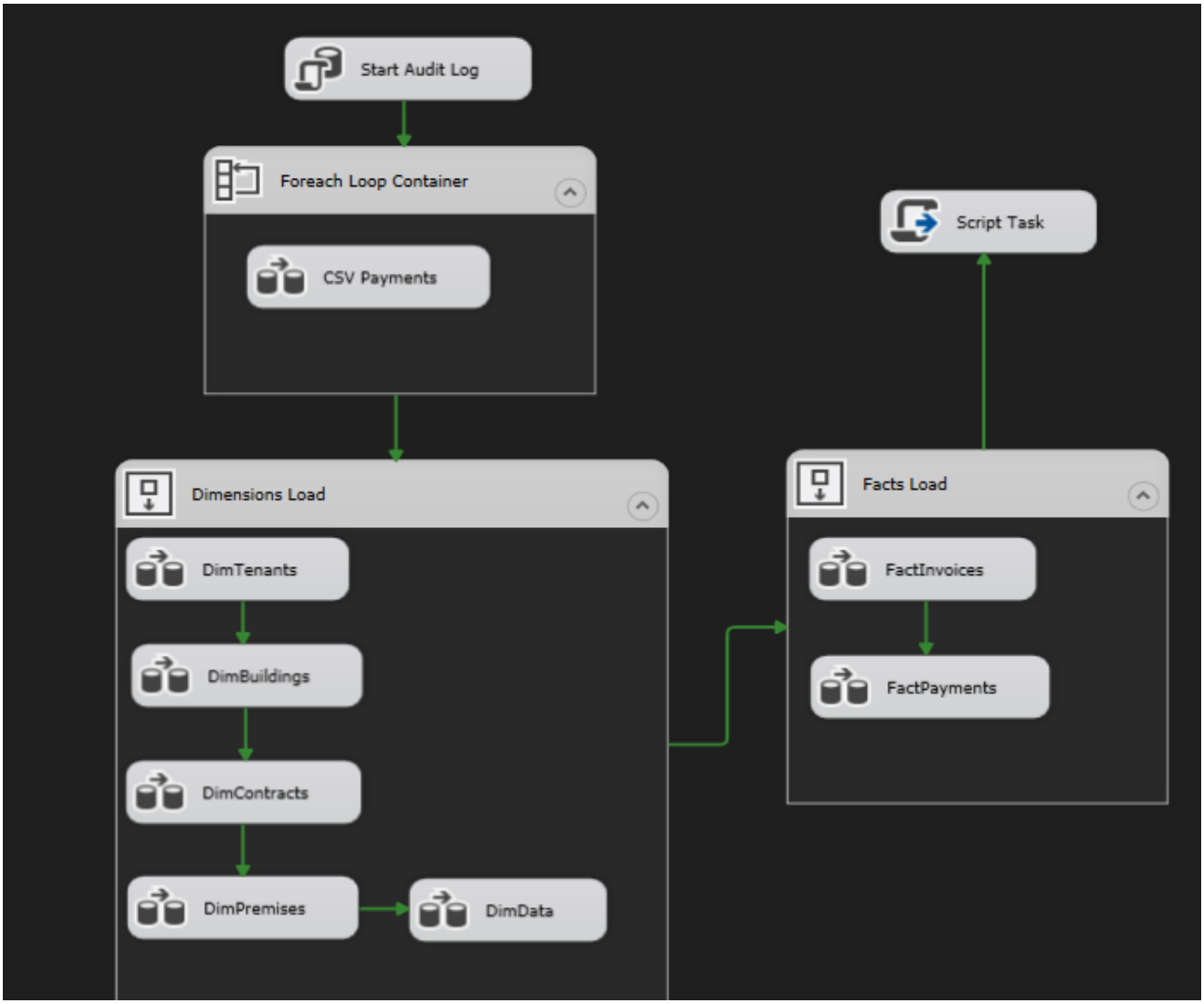

Rental_Invoices
Lookup Tenants
Lookup Match Output
Lookup Contract
Lookup Match Output
Lookup Permises
Lookup Match Output
Lookup Date
Lookup Match Output
Data Conversion
OLE DB Destination

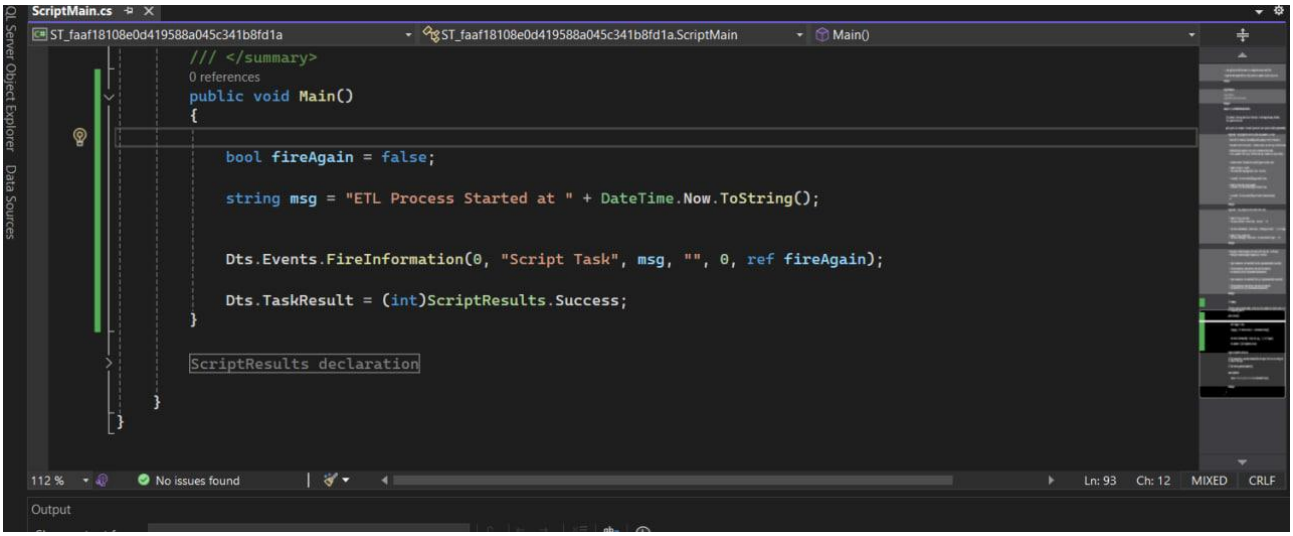

OLE DB Source
Lookup
Lookup Match Output
Lookup Contract
Lookup Match Output
Data Conversion
Lookup Match Output
Lookup Date
Lookup Match Output
Lookup for DTID
Lookup Match Output
Lookup Tenants
Lookup Match Output
Lookup 1
Lookup Match Output
Lookup Building
Lookup Match Output
OLE DB Destination

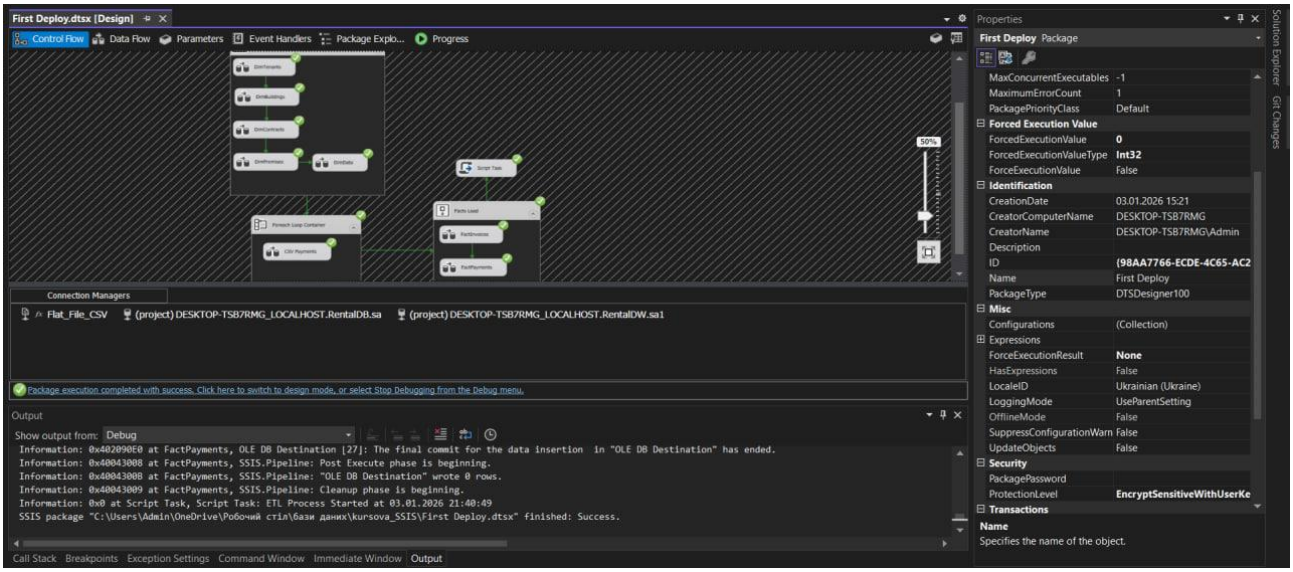

Execute SQL Task Editor
Configure the properties required to run SQL statements and stored procedures using the selected connection.
General
Parameter Mapping
Result Set
Expressions
Variable...
Direction
Data Ty...
Paramet...
Paramet...
System::...
Input
VARCH...
0
-1
Add
Remove
OK
Cancel
Help

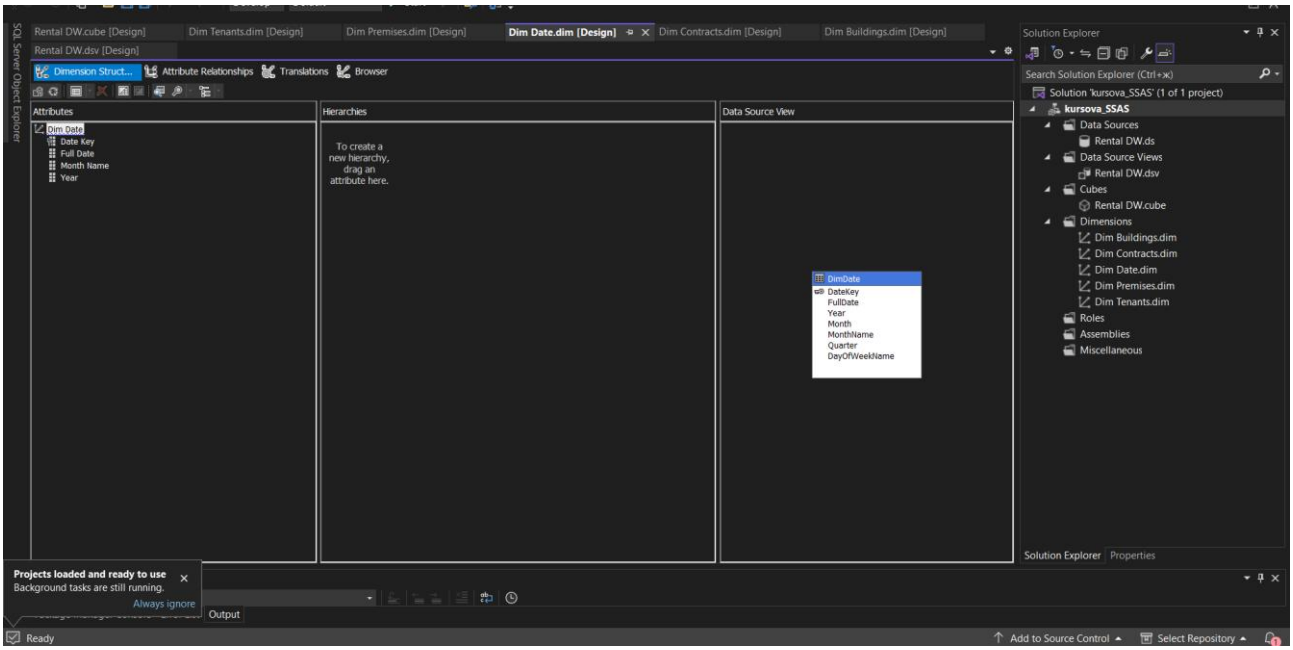

Flat File Source
OLE DB Destination

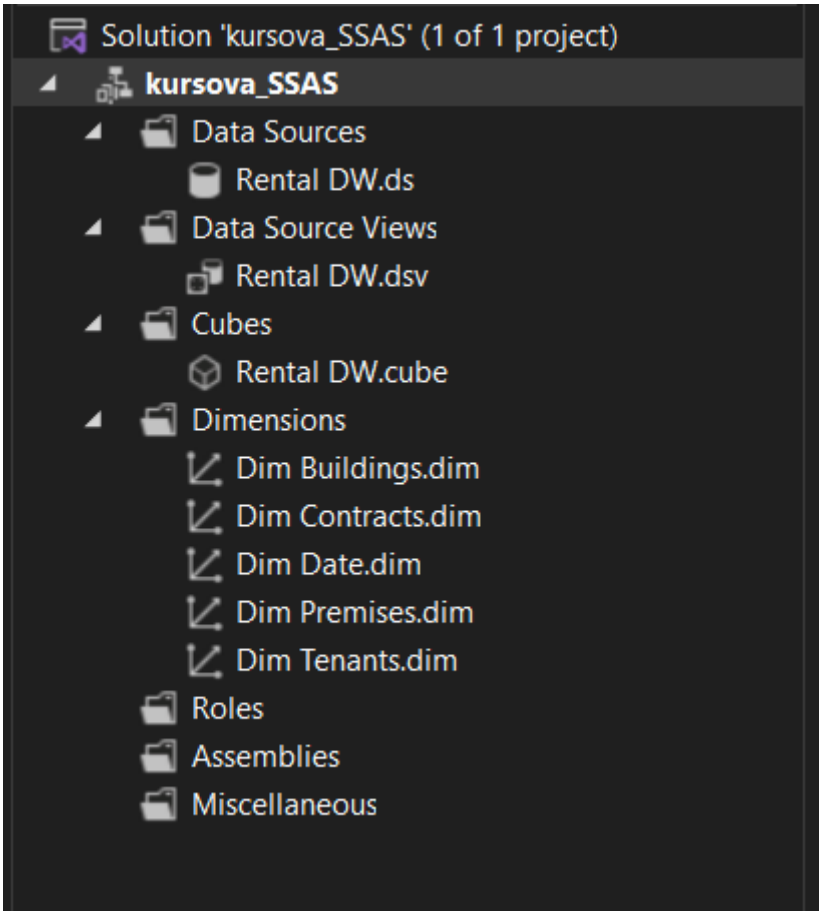

Script Task Editor
Access Microsoft Visual Studio Tools for Applications (VSTA) to write scripts using the Visual Basic 2022 or Visual C# 2022, and configure the task's properties.
Script
General
Expressions
General
Name
Script Task
Description
Script Task
Name
Specifies the name of the task.
OK
Cancel
Help

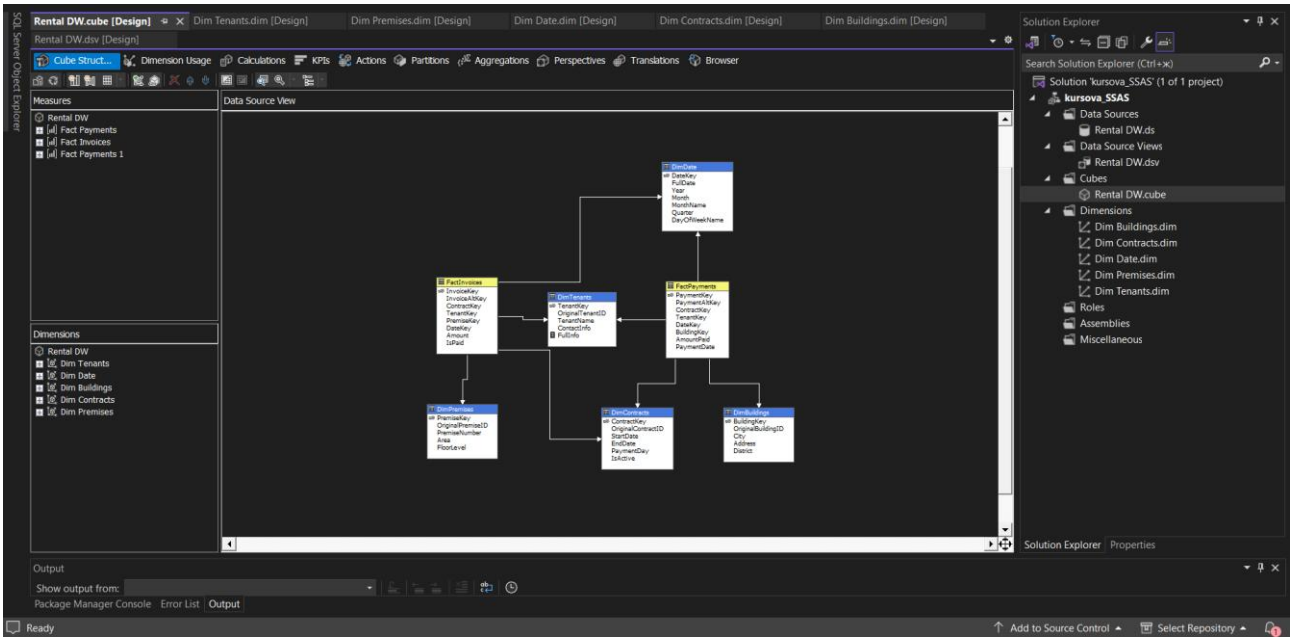

Start Audit Log
Foreach Loop Container
CSV Payments
Script Task
Dimensions Load
DimTenants
DimBuildings
DimContracts
DimPremises
DimData
Facts Load
FactInvoices
FactPayments

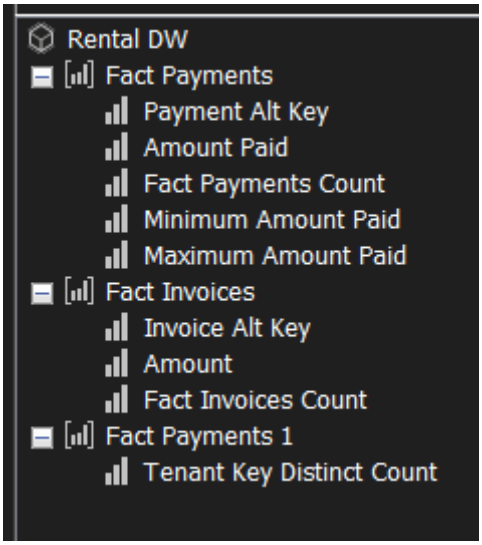

ScriptMain.cs
ST_faaf18108e0d419588a045c341b8fd1a
ST_faaf18108e0d419588a045c341b8fd1a.ScriptMain
Main()
/// </summary>
0 references
public void Main()
{
bool fireAgain = false;
string msg = "ETL Process Started at " + DateTime.Now.ToString();
Dts.Events.FireInformation(0, "Script Task", msg, "", 0, ref fireAgain);
Dts.TaskResult = (int)ScriptResults.Success;
}
ScriptResults declaration
}
}
No issues found
Ln: 93
Ch: 12
MIXED
CRLF
Output

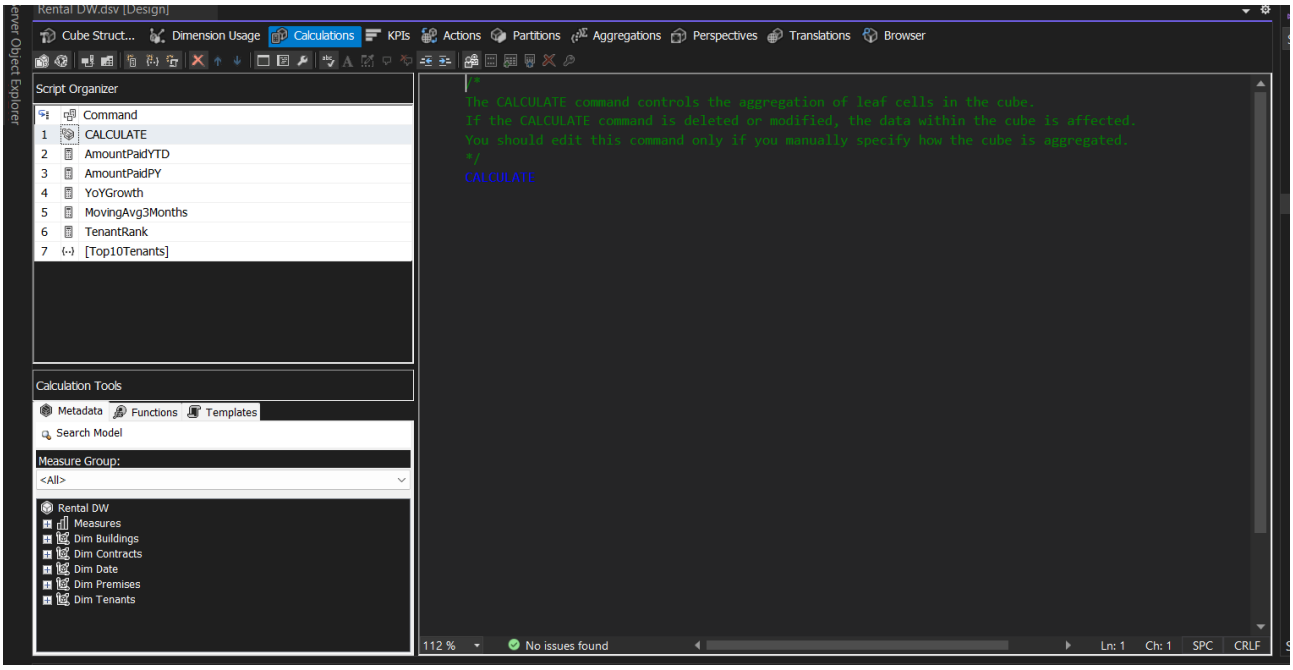

First Deploy.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explo...
Progress
Connection Managers
Flat_File_CSV
(project) DESKTOP-TSB7RMG_LOCALHOST.RentalDB.sa
(project) DESKTOP-TSB7RMG_LOCALHOST.RentalDW.sa1
Package execution completed with success. Click here to switch to design mode, or select Stop Debugging from the Debug menu.
Output
Show output from: Debug
Information: 0x402090E0 at FactPayments, OLE DB Destination [27]: The final commit for the data insertion in "OLE DB Destination" has ended.
Information: 0x40043008 at FactPayments, SSIS.Pipeline: Post Execute phase is beginning.
Information: 0x40043008 at FactPayments, SSIS.Pipeline: "OLE DB Destination" wrote 0 rows.
Information: 0x40043009 at FactPayments, SSIS.Pipeline: Cleanup phase is beginning.
Information: 0x0 at Script Task, Script Task: ETL Process Started at 03.01.2026 21:40:49
SSIS package "C:\Users\Admin\OneDrive\Робочий стіл\бази даних\kursova_SSIS\First Deploy.dtsx" finished: Success.
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Window
Output
Properties
First Deploy Package
MaxConcurrentExecutables -1
MaximumErrorCount 1
PackagePriorityClass Default
Forced Execution Value
ForcedExecutionValue 0
ForcedExecutionValueType Int32
ForceExecutionValue False
Identification
CreationDate 03.01.2026 15:21
CreatorComputerName DESKTOP-TSB7RMG
CreatorName DESKTOP-TSB7RMG\Admin
Description
ID {98AA7766-ECDE-4C65-AC2
Name First Deploy
PackageType DTSDesigner100
Misc
Configurations (Collection)
Expressions
ForceExecutionResult None
HasExpressions False
LocaleID Ukrainian (Ukraine)
LoggingMode UseParentSetting
OfflineMode False
SuppressConfigurationWarn False
UpdateObjects False
Security
PackagePassword
ProtectionLevel EncryptSensitiveWithUserKe
Transactions
Name
Specifies the name of the object.
Solution Explorer
Get Changes

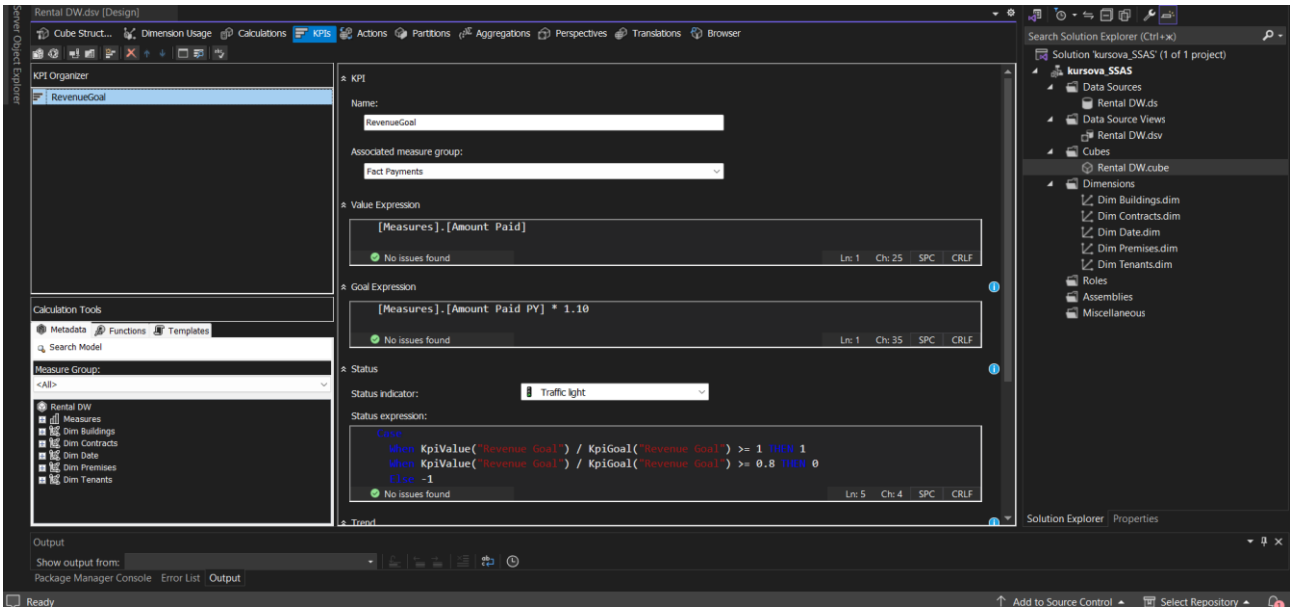

Rental DW.cube [Design]
Dim Tenants.dim [Design]
Dim Premises.dim [Design]
Dim Date.dim [Design]
Dim Contracts.dim [Design]
Dim Buildings.dim [Design]
Rental DW.dsv [Design]
Dimension Struct...
Attribute Relationships
Translations
Browser
Attributes
Dim Date
Date Key
Full Date
Month Name
Year
Hierarchies
To create a new hierarchy, drag an attribute here.
Data Source View
DimDate
DateKey
FullDate
Year
Month
MonthName
Quarter
DayOfWeekName
Solution Explorer
Search Solution Explorer (Ctrl+ж)
Solution 'kursova_SSAS' (1 of 1 project)
kursova_SSAS
Data Sources
Rental DW.ds
Data Source Views
Rental DW.dsv
Cubes
Rental DW.cube
Dimensions
Dim Buildings.dim
Dim Contracts.dim
Dim Date.dim
Dim Premises.dim
Dim Tenants.dim
Roles
Assemblies
Miscellaneous
Solution Explorer
Properties
Projects loaded and ready to use
Background tasks are still running.
Always ignore
Output
Ready
Add to Source Control
Select Repository

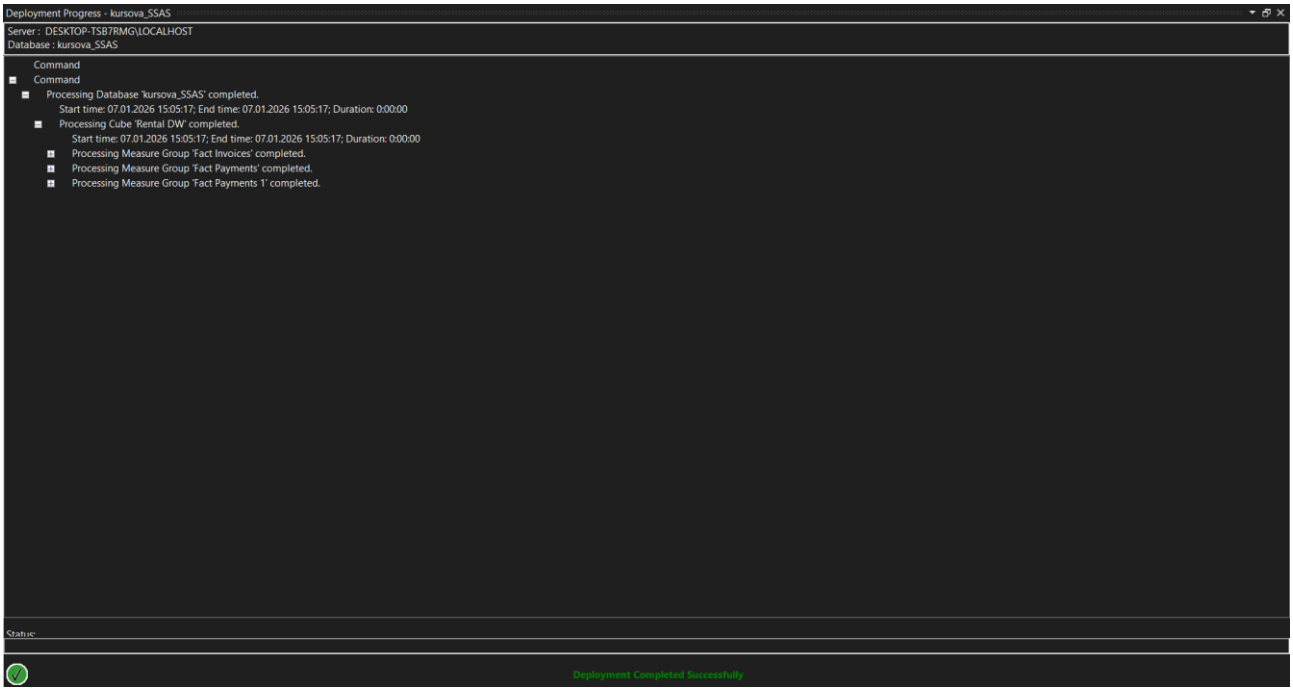

Solution 'kursova_SSAS' (1 of 1 project)
kursova_SSAS
Data Sources
Rental DW.ds
Data Source Views
Rental DW.dsv
Cubes
Rental DW.cube
Dimensions
Dim Buildings.dim
Dim Contracts.dim
Dim Date.dim
Dim Premises.dim
Dim Tenants.dim
Roles
Assemblies
Miscellaneous

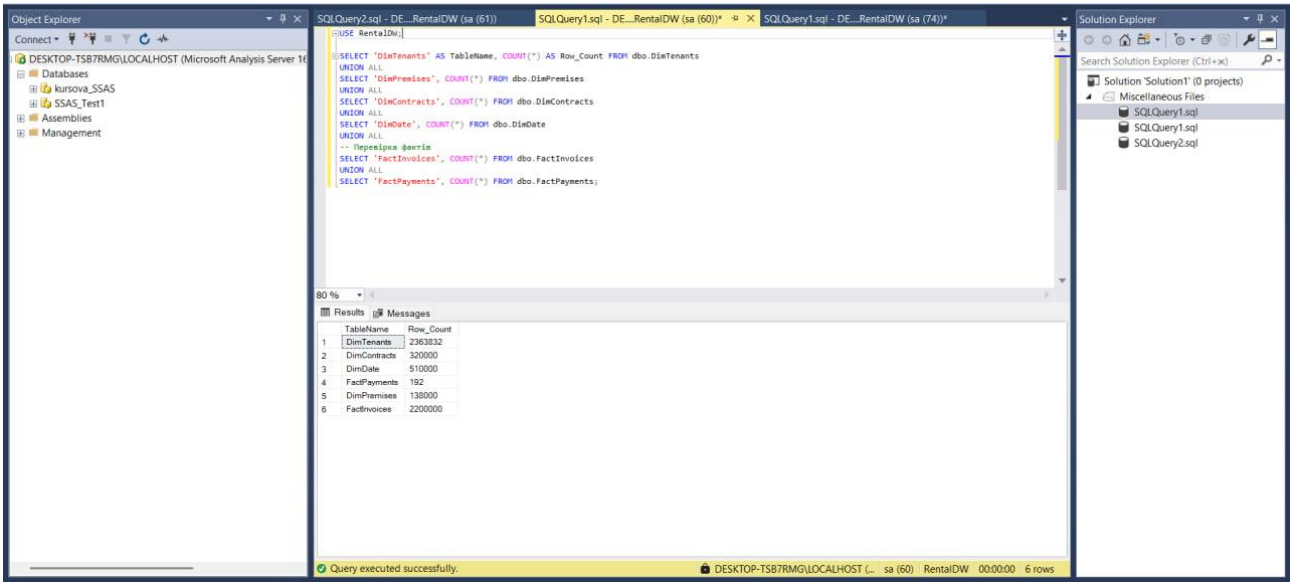

Rental DW.cube [Design]
Dim Tenants.dim [Design]
Dim Premises.dim [Design]
Dim Date.dim [Design]
Dim Contracts.dim [Design]
Dim Buildings.dim [Design]
Rental DW.dsv [Design]
Cube Struct...
Dimension Usage
Calculations
KPIs
Actions
Partitions
Aggregations
Perspectives
Translations
Browser
Measures
Rental DW
Fact Payments
Fact Invoices
Fact Payments 1
Data Source View
Dimensions
Rental DW
Dim Tenants
Dim Date
Dim Buildings
Dim Contracts
Dim Premises
Solution Explorer
Search Solution Explorer (Ctrl+ж)
Solution 'kursova_SSAS' (1 of 1 project)
kursova_SSAS
Data Sources
Rental DW.ds
Data Source Views
Rental DW.dsv
Cubes
Rental DW.cube
Dimensions
Dim Buildings.dim
Dim Contracts.dim
Dim Date.dim
Dim Premises.dim
Dim Tenants.dim
Roles
Assemblies
Miscellaneous
Solution Explorer
Properties
Output
Show output from:
Package Manager Console
Error List
Output
Ready
Add to Source Control
Select Repository

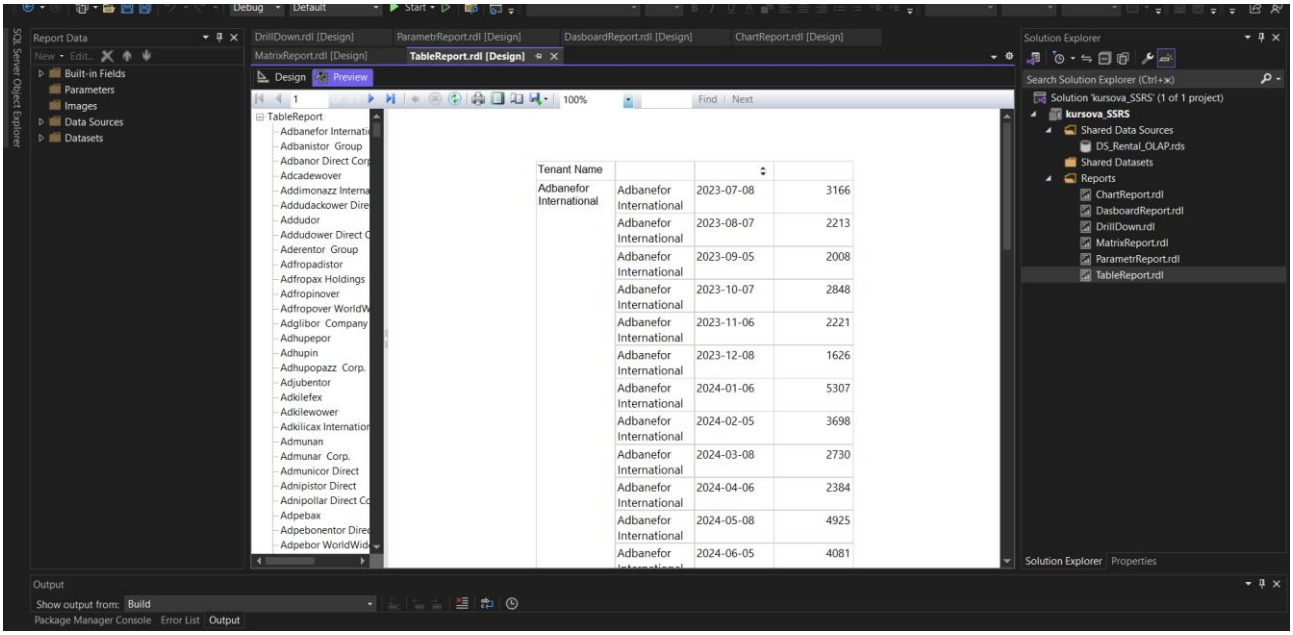

Rental DW
Fact Payments
Payment Alt Key
Amount Paid
Fact Payments Count
Minimum Amount Paid
Maximum Amount Paid
Fact Invoices
Invoice Alt Key
Amount
Fact Invoices Count
Fact Payments 1
Tenant Key Distinct Count


Rental DW.dsv [Design]
Cube Struct...
Dimension Usage
Calculations
KPIs
Actions
Partitions
Aggregations
Perspectives
Translations
Browser
Script Organizer
Command
1 CALCULATE
2 AmountPaidYTD
3 AmountPaidPY
4 YoYGrowth
5 MovingAvg3Months
6 TenantRank
7 [Top10Tenants]
/*
The CALCULATE command controls the aggregation of leaf cells in the cube.
If the CALCULATE command is deleted or modified, the data within the cube is affected.
You should edit this command only if you manually specify how the cube is aggregated.
*/
CALCULATE
Calculation Tools
Metadata
Functions
Templates
Search Model
Measure Group:
<All>
Rental DW
Measures
Dim Buildings
Dim Contracts
Dim Date
Dim Premises
Dim Tenants
112 %
No issues found
Ln: 1
Ch: 1
SPC
CRLF


Rental DW.dsv [Design]
KPI Organizer
RevenueGoal
KPI
Name:
RevenueGoal
Associated measure group:
Fact Payments
Value Expression
[Measures].[Amount Paid]
Goal Expression
[Measures].[Amount Paid PY] * 1.10
Status
Status indicator:
Traffic light
Status expression:
Case
When KpiValue("Revenue Goal") / KpiGoal("Revenue Goal") >= 1 THEN 1
When KpiValue("Revenue Goal") / KpiGoal("Revenue Goal") >= 0.8 THEN 0
Else -1
Trend
Search Solution Explorer (Ctrl+ж)
Solution 'kursova_SSAS' (1 of 1 project)
kursova_SSAS
Data Sources
Rental DW.ds
Data Source Views
Rental DW.dsv
Cubes
Rental DW.cube
Dimensions
Dim Buildings.dim
Dim Contracts.dim
Dim Date.dim
Dim Premises.dim
Dim Tenants.dim
Roles
Assemblies
Miscellaneous
Output
Package Manager Console
Error List
Output
Ready
Add to Source Control
Select Repository


Deployment Progress - kursova_SSAS
Server : DESKTOP-TSB7RMG\LOCALHOST
Database : kursova_SSAS
Command
Command
Processing Database 'kursova_SSAS' completed.
Start time: 07.01.2026 15:05:17; End time: 07.01.2026 15:05:17; Duration: 0:00:00
Processing Cube 'Rental DW' completed.
Start time: 07.01.2026 15:05:17; End time: 07.01.2026 15:05:17; Duration: 0:00:00
Processing Measure Group 'Fact Invoices' completed.
Processing Measure Group 'Fact Payments' completed.
Processing Measure Group 'Fact Payments 1' completed.
Status:
Deployment Completed Successfully


Object Explorer
Connect
DESKTOP-TSB7RMG\LOCALHOST (Microsoft Analysis Server 16
Databases
kursova_SSAS
SSAS_Test1
Assemblies
Management
SQLQuery2.sql - DE....RentalDW (sa (61))
SQLQuery1.sql - DE....RentalDW (sa (60))*
SQLQuery1.sql - DE....RentalDW (sa (74))*
USE RentalDW;
SELECT 'DimTenants' AS TableName, COUNT(*) AS Row_Count FROM dbo.DimTenants
UNION ALL
SELECT 'DimPremises', COUNT(*) FROM dbo.DimPremises
UNION ALL
SELECT 'DimContracts', COUNT(*) FROM dbo.DimContracts
UNION ALL
SELECT 'DimDate', COUNT(*) FROM dbo.DimDate
UNION ALL
-- Перевірка фактів
SELECT 'FactInvoices', COUNT(*) FROM dbo.FactInvoices
UNION ALL
SELECT 'FactPayments', COUNT(*) FROM dbo.FactPayments;
80 %
Results
Messages
TableName Row_Count
1 DimTenants 2363832
2 DimContracts 320000
3 DimDate 510000
4 FactPayments 192
5 DimPremises 138000
6 FactInvoices 2200000
Query executed successfully.
DESKTOP-TSB7RMG\LOCALHOST (... sa (60) RentalDW 00:00:00 6 rows
Solution Explorer
Search Solution Explorer (Ctrl+ж)
Solution 'Solution1' (0 projects)
Miscellaneous Files
SQLQuery1.sql
SQLQuery1.sql
SQLQuery2.sql


Tenant Name
Adbanefor International
Adbanefor International 2023-07-08 3166
Adbanefor International 2023-08-07 2213
Adbanefor International 2023-09-05 2008
Adbanefor International 2023-10-07 2848
Adbanefor International 2023-11-06 2221
Adbanefor International 2023-12-08 1626
Adbanefor International 2024-01-06 5307
Adbanefor International 2024-02-05 3698
Adbanefor International 2024-03-08 2730
Adbanefor International 2024-04-06 2384
Adbanefor International 2024-05-08 4925
Adbanefor International 2024-06-05 4081

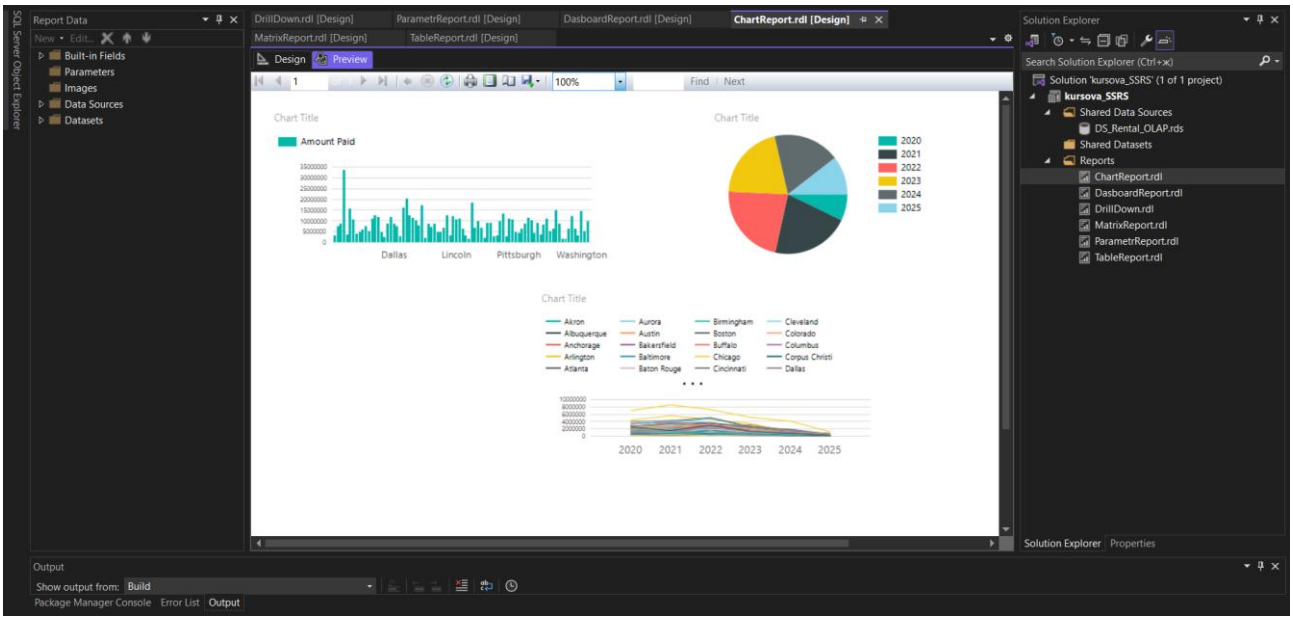
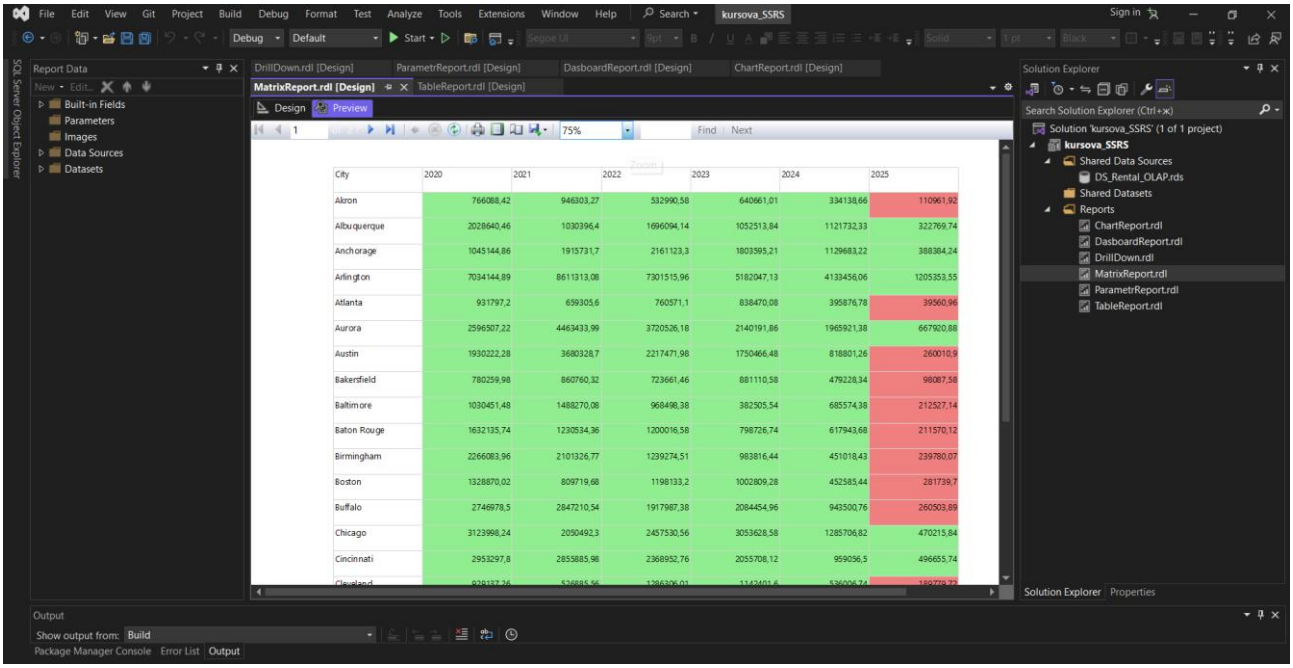
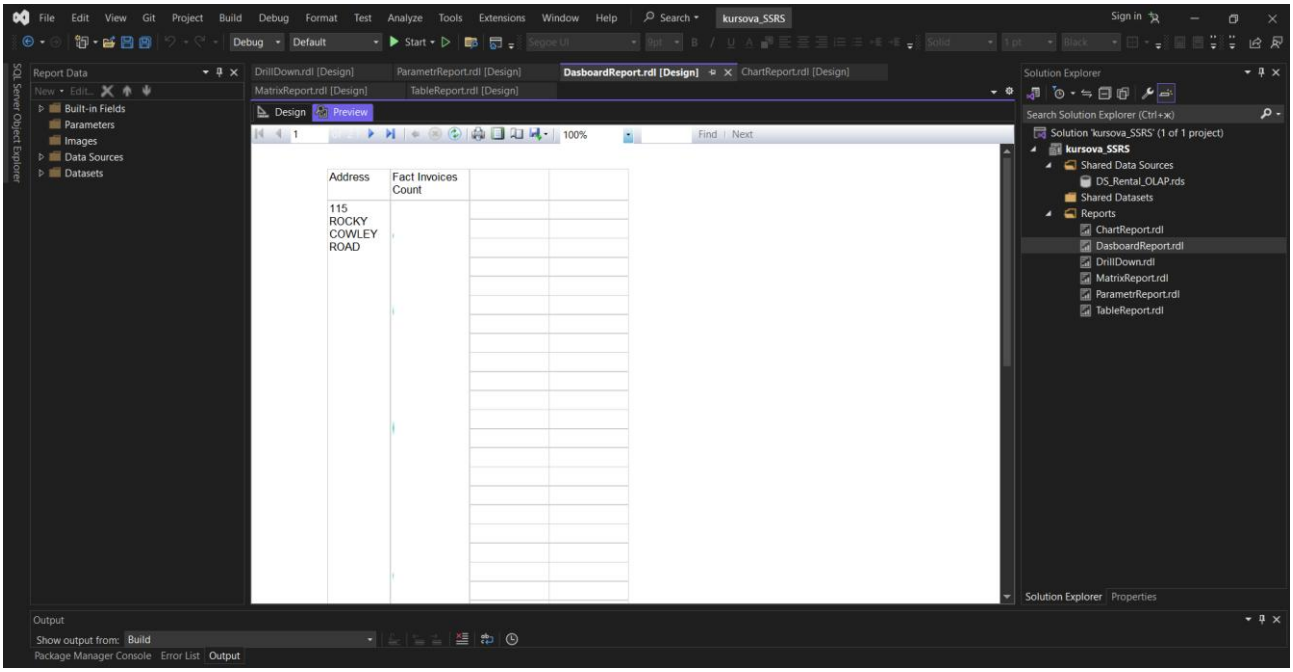
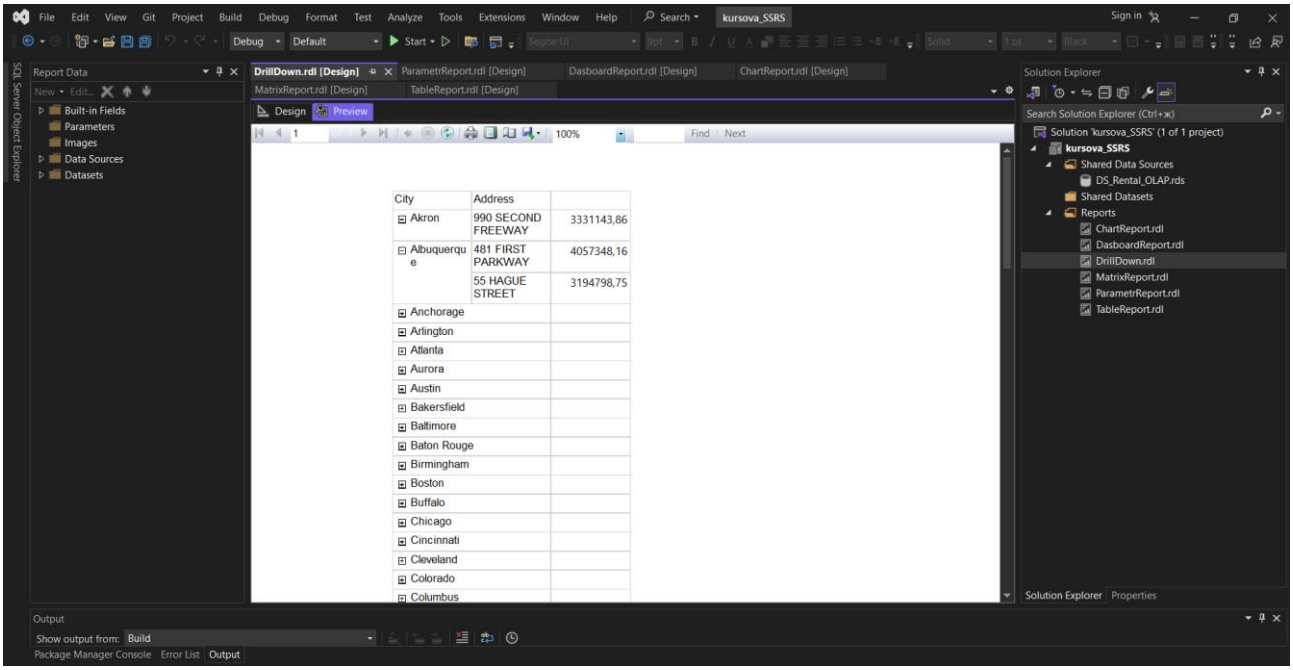
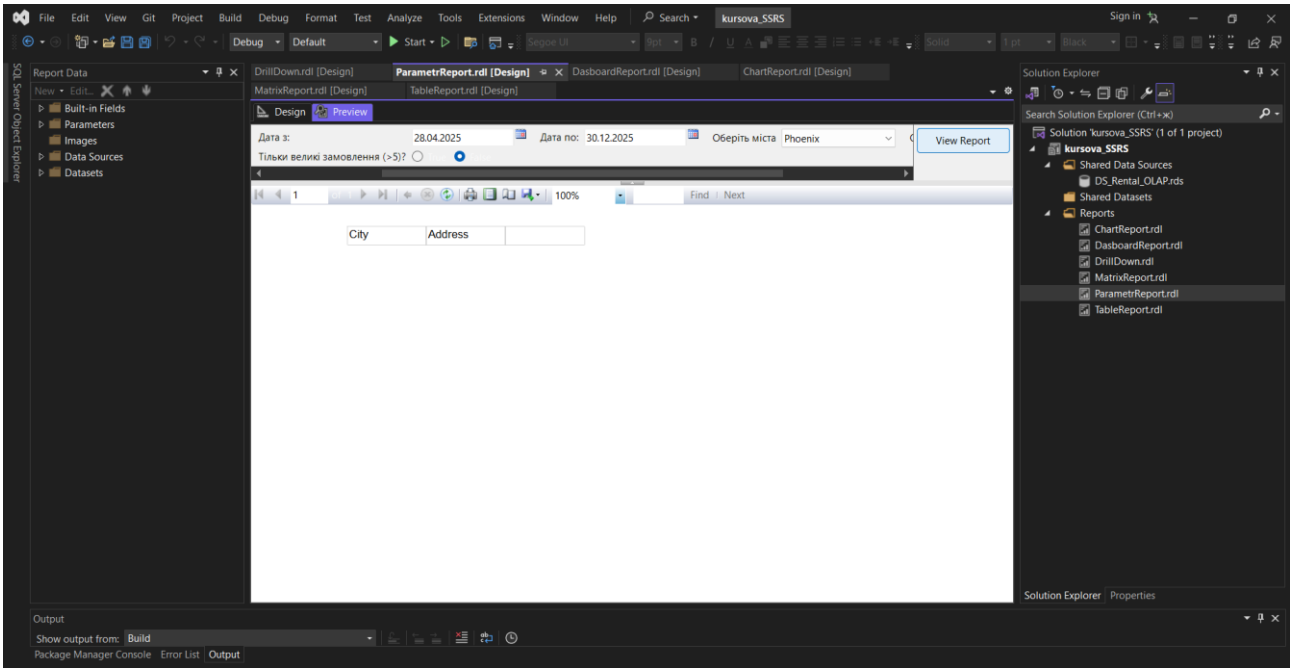


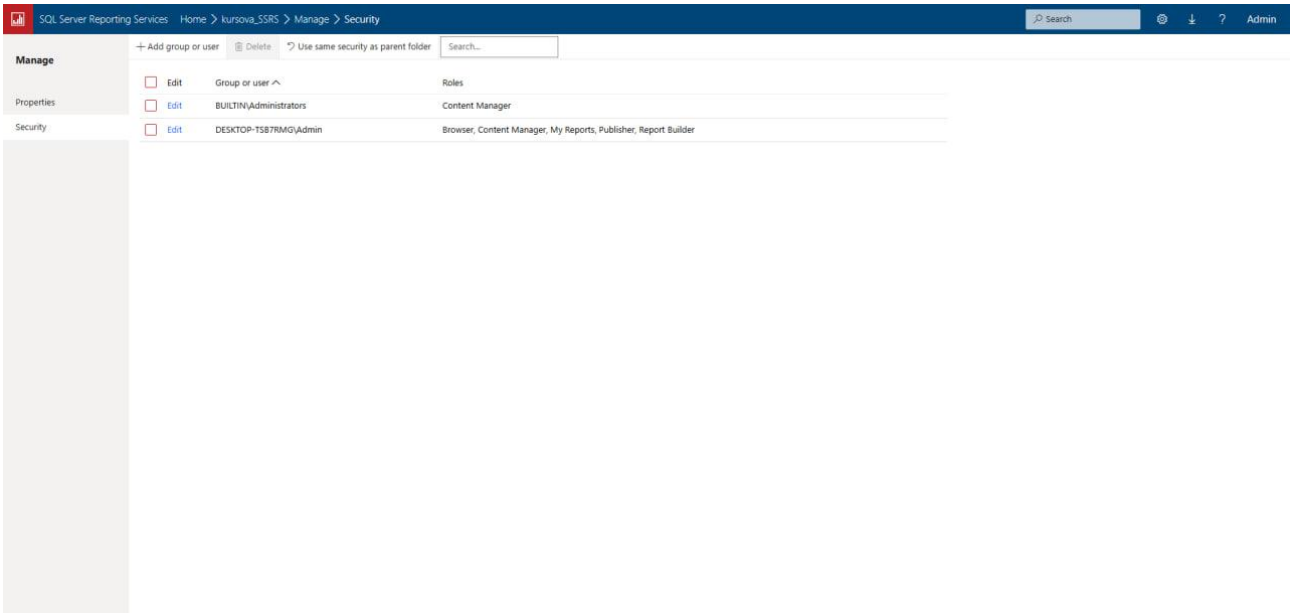
Chart Title
Amount Paid
Dallas
Lincoln
Pittsburgh
Washington
Chart Title
2020
2021
2022
2023
2024
2025
Chart Title

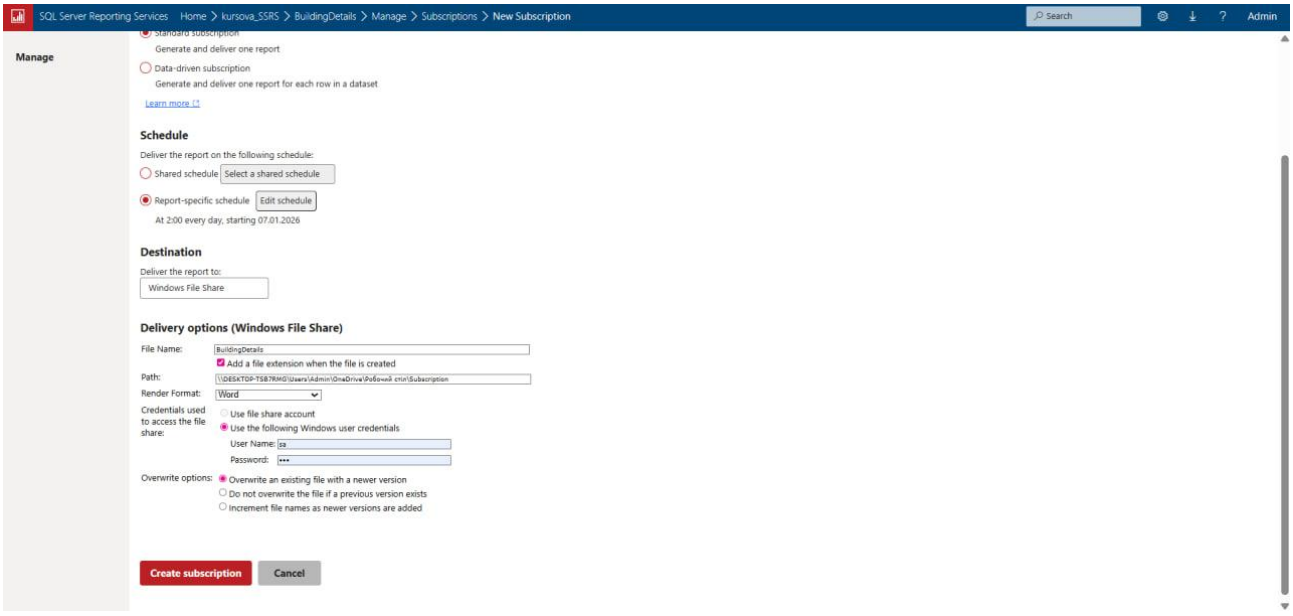

MatrixReport.rdl [Design]
TableReport.rdl [Design]
DrillDown.rdl [Design]
ParametrReport.rdl [Design]
DasboardReport.rdl [Design]
ChartReport.rdl [Design]
Report Data
Solution Explorer
kursova_SSRS
Solution 'kursova_SSRS' (1 of 1 project)
Shared Data Sources
DS_Rental_OLAP.rds
Shared Datasets
Reports
ChartReport.rdl
DasboardReport.rdl
DrillDown.rdl
MatrixReport.rdl
ParametrReport.rdl
TableReport.rdl
City
2020
2021
2022
2023
2024
2025
Akron
Albuquerque
Anchorage
Arlington
Atlanta
Aurora
Austin
Bakersfield
Baltimore
Baton Rouge
Birmingham
Boston
Buffalo
Chicago
Cincinnati
Cleveland

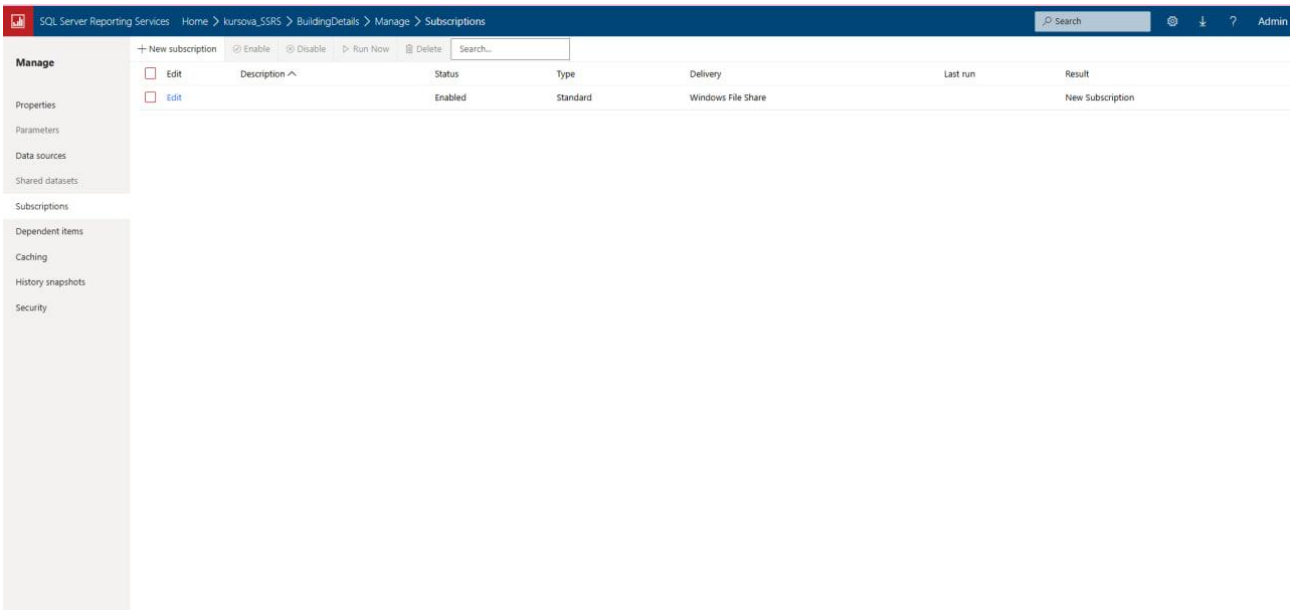

DasboardReport.rdl [Design]
Address
Fact Invoices Count
115 ROCKY COWLEY ROAD

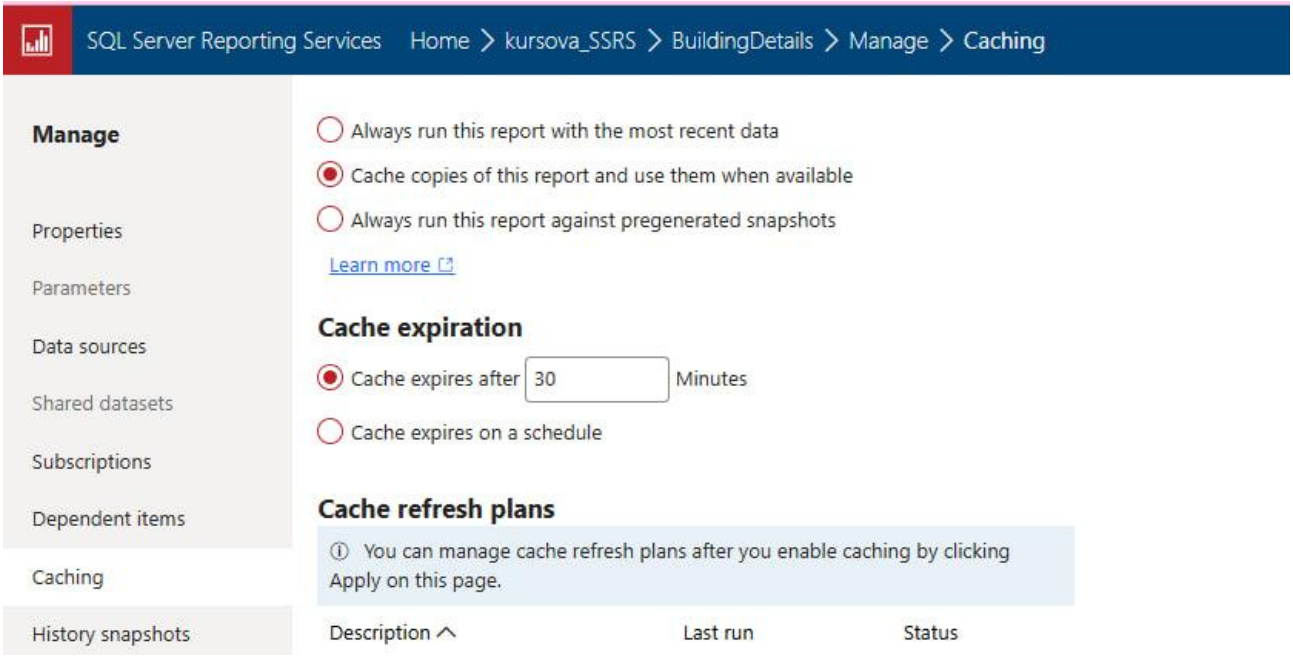

File Edit View Git Project Build Debug Format Test Analyze Tools Extensions Window Help
kursova_SSRS
Report Data
DrillDown.rdl [Design]
ParametrReport.rdl [Design]
DasboardReport.rdl [Design]
ChartReport.rdl [Design]
MatrixReport.rdl [Design]
TableReport.rdl [Design]
Design
Preview
City
Address
Akron
990 SECOND FREEWAY
3331143,86
Albuquerque
481 FIRST PARKWAY
4057348,16
55 HAGUE STREET
3194798,75
Anchorage
Arlington
Atlanta
Aurora
Austin
Bakersfield
Baltimore
Baton Rouge
Birmingham
Boston
Buffalo
Chicago
Cincinnati
Cleveland
Colorado
Columbus
Solution Explorer
Solution 'kursova_SSRS' (1 of 1 project)
kursova_SSRS
Shared Data Sources
DS_Rental_OLAP.rds
Shared Datasets
Reports
ChartReport.rdl
DasboardReport.rdl
DrillDown.rdl
MatrixReport.rdl
ParametrReport.rdl
TableReport.rdl
Output
Show output from: Build
Package Manager Console
Error List
Output


File Edit View Git Project Build Debug Format Test Analyze Tools Extensions Window Help
kursova_SSRS
Report Data
DrillDown.rdl [Design]
ParametrReport.rdl [Design]
DasboardReport.rdl [Design]
ChartReport.rdl [Design]
MatrixReport.rdl [Design]
TableReport.rdl [Design]
Design
Preview
Дата з:
28.04.2025
Дата по:
30.12.2025
Оберіть міста
Phoenix
Тільки великі замовлення (>5)?
View Report
City
Address
Solution Explorer
Solution 'kursova_SSRS' (1 of 1 project)
kursova_SSRS
Shared Data Sources
DS_Rental_OLAP.rds
Shared Datasets
Reports
ChartReport.rdl
DasboardReport.rdl
DrillDown.rdl
MatrixReport.rdl
ParametrReport.rdl
TableReport.rdl
Output
Show output from: Build
Package Manager Console
Error List
Output


SQL Server Reporting Services
Home > kursova_SSRS > Manage > Security
Search
Admin
Manage
Properties
Security
Add group or user
Delete
Use same security as parent folder
Search...
Edit
Group or user
Roles
Edit
BUILTIN\Administrators
Content Manager
Edit
DESKTOP-TSB7RMG\Admin
Browser, Content Manager, My Reports, Publisher, Report Builder


SQL Server Reporting Services
Home > kursova_SSRS > BuildingDetails > Manage > Subscriptions > New Subscription
Search
Admin
Manage
Generate and deliver one report
Data-driven subscription
Generate and deliver one report for each row in a dataset
Learn more
Schedule
Deliver the report on the following schedule:
Shared schedule
Select a shared schedule
Report-specific schedule
Edit schedule
At 2:00 every day, starting 07.01.2026
Destination
Deliver the report to:
Windows File Share
Delivery options (Windows File Share)
File Name:
BuildingDetails
Add a file extension when the file is created
Path:
Render Format:
Word
Credentials used to access the file share:
Use file share account
Use the following Windows user credentials
User Name:
sa
Password:
Overwrite options:
Overwrite an existing file with a newer version
Do not overwrite the file if a previous version exists
Increment file names as newer versions are added
Create subscription
Cancel


SQL Server Reporting Services
Home > kursova_SSRS > BuildingDetails > Manage > Subscriptions
Search
Admin
Manage
Properties
Parameters
Data sources
Shared datasets
Subscriptions
Dependent items
Caching
History snapshots
Security
New subscription
Enable
Disable
Run Now
Delete
Search...
Edit
Description
Status
Type
Delivery
Last run
Result
Edit
Enabled
Standard
Windows File Share
New Subscription


SQL Server Reporting Services
Home > kursova_SSRS > BuildingDetails > Manage > Caching
Manage
Properties
Parameters
Data sources
Shared datasets
Subscriptions
Dependent items
Caching
History snapshots
Always run this report with the most recent data
Cache copies of this report and use them when available
Always run this report against pregenerated snapshots
Learn more
Cache expiration
Cache expires after 30 Minutes
Cache expires on a schedule
Cache refresh plans
You can manage cache refresh plans after you enable caching by clicking Apply on this page.
Description
Last run
Status