

Міністерство освіти і науки України
Національний університет “Львівська політехніка”
Інститут прикладної математики та фундаментальних наук

Кафедра прикладної математики

Звіт
про виконання курсової роботи
з курсу “Надвеликі бази даних”

Виконала:
студентка групи ПМ-43
Агєєнко Анастасія

Перевірив:
Любінський Б.Б.

Львів - 2025

Вступ

Актуальність теми

У сучасних умовах цифровізації вищої освіти управління навчальним процесом потребує впровадження високоефективних інформаційних систем. Традиційні методи обліку успішності студентів, що базуються на паперових носіях або розрізнених електронних таблицях, вже не задовольняють потреби працівників адміністрації університетів у швидкому та точному аналізі даних. Створення єдиного простору для ведення обліку студентів, їх соціальних характеристик та академічних досягнень є критично важливим для прозорого нарахування стипендії та формування рейтингів успішності.

Дана робота присвячена розробці інформаційної системи, яка дозволяє не лише зберігати анкети студентів (ПІБ, номер студентського квитка, адреса проживання, сімейний стан, стипендія, проживання в гуртожитку, активність, тощо), а й здійснювати комплексний аналіз їхньої успішності з урахуванням викладачів, предметів та навчальних груп.

Мета роботи

Проектування та реалізація інформаційної системи управління навчальним процесом університету. Система має автоматизувати процес збору даних, їх перетворення та візуалізації у вигляді інтерактивних звітів.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- Провести аналіз предметної області та спроектувати архітектуру сховища даних.
- Реалізувати ETL-процеси для автоматичного завантаження даних про студентів, їхні хобі, стипендії та отримані бали.
- Побудувати багатовимірний OLAP-куб для миттєвого розрахунку таких метрик як AverageScore, ScholarshipPercent, CostPerPoint, StudentRank, WeightedAverage, PreviousYear, 3MonthAverageScore, ScorePer1000Currency, YearOverYearGrowth
- Розробити динамічні аналітичні звіти:
 - список студентів по групах;
 - рейтинг студентів (з параметром діапазону);
 - довідка студента про рейтинг та стипендію;
 - аналіз успішності по предметах та викладачах.

Об'єкт дослідження

Процеси обліку студентів та моніторинг їхньої успішності в закладі вищої освіти.

Предмет дослідження

Методи та засоби побудови аналітичних систем на основі стеку технологій Microsoft SQL Server (SSIS, SSAS, SSRS).

Розділ 1. Аналіз предметної області

Об'єктом дослідження є процеси обліку студентів та моніторингу їхньої успішності в рамках вишого навчального закладу. Предметна область охоплює життєвий цикл студента від моменту зарахування до групи до формування фінальних рейтингів для виплати стипендій.

Ключові компоненти:

- **студенти:** система зберігає повний профіль студента, включаючи персональні дані (ПІБ, номер студентського квитка, адреса проживання) та соціальні характеристики (місце народження, стать).
- **академічна структура:** студенти розподілені за групами (Group), які є ключовими адміністративними одиницями.
- **Навчальний процес:** включає взаємодію викладачів (Teacher) та предметів (Subject). Кожен предмет закріплений за викладачем у певному навчальному році (TeacherSubject).
- **Оцінювання та успішність:** центральною частиною є фіксація балів (StudentGrade) за виконання навчального плану.

Виявлення та опис основних бізнес-процесів

У системі виділено три основні потоки даних:

1. Облік та рух студентів:

- Реєстрація нових студентів та призначення їх до груп через сутність StudentGroup.
- Відстеження соціальних аспектів, зокрема захоплень студента (StudentHobby), що дозволяє формувати цілісний портрет особистості.

2. Контроль знань та оцінювання:

- Призначення викладачів на конкретні дисципліни.
- Фіксація результатів іспитів та заліків у таблиці StudentGrade. Це є джерелом для розрахунку AverageScore, який ми бачимо у звітах.

3. Формування аналітичної звітності та стипендіальне забезпечення:

- Розрахунок рейтингу: на основі накопичених балів за певний період (ScholarshipPeriod) формується підсумковий рейтинг (StudentRating).
- Виплата стипендій: автоматичне визначення суми та підтвердження виплати (ScholarshipPayment) на основі позиції студента в рейтингу успішності.

Функціональні вимоги до системи

На основі аналізу визначено такі вимоги:

- Гнучка фільтрація: можливість вибору конкретного студента або групи через динамічні параметри (StudentFullName).
- Багатовимірний аналіз: розрахунок середніх показників успішності за допомогою OLAP-технологій.
- Автоматизація виплати: система повинна блокувати нарахування стипендії студентам, чий рейтинг нижчий за встановлений поріг.

Бізнес-правила та обмеження

1. Унікальність: кожен студент повинен мати унікальний номер студентського квитка (StudentCardNo).
2. Цілісність: оцінка студента не може бути внесена в систему без прив'язки до існуючого зв'язку “Викладач-Предмет”
3. Періодичність: рейтинг розраховується за визначені періоди, що фіксуються в ScholarshipPeriod.

Концептуальне проектування

Концептуальна модель бази даних відображає основні об'єкти (сущності) предметної області та зв'язки між ними без прив'язки до конкретної СУБД.

Опис основних сущностей та їх атрибутів:

Студенти (Student):

- Опис: містить персональну та анкетовану інформацію про осіб, що навчаються.
- Атрибути:
 - StudentID
 - LastName
 - FirstName
 - MiddleName
 - StudentCardNo
 - BirthYear
 - BirthPlace
 - Address
 - Gender
 - MaritalStatus
 - ScholarshipAmount
 - DormRoomNum
 - IsActive

Групи (Group):

- Опис: колектив студентів, об'єднаний за роком вступу та напрямом навчання.
- Атрибути:
 - GroupID
 - GroupCode
 - CreationYear
 - LeaderStudentID

Предмети (Subject):

- Опис: навчальні дисципліни, які входять до програми навчання.
- Атрибути:
 - SubjectID
 - SubjectName
 - Credits

Викладачі (Teacher):

- Опис: співробітники університету, що забезпечують навчальний процес.
- Атрибути:
 - TeacherID
 - LastName
 - FirstName
 - MiddleName
 - Department
 - Email

Оцінки (StudentGrade):

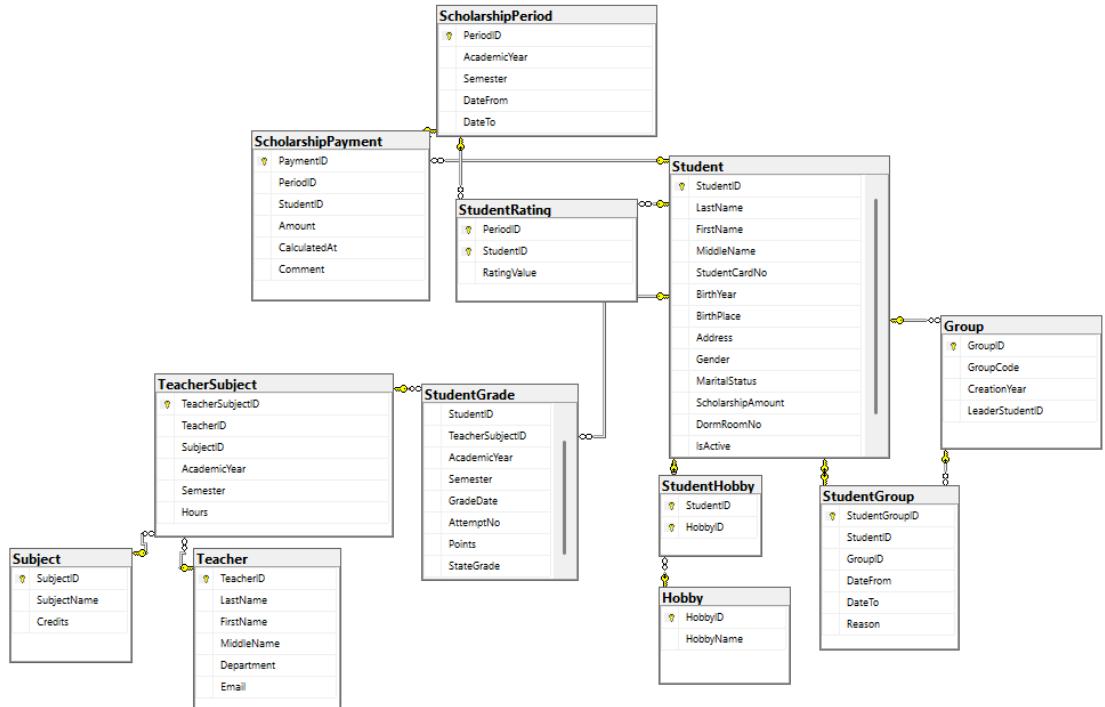
- Опис: результати контролю знань студентів.
- Атрибути:
 - StudentGradeID
 - StudentID
 - TeacherSubjectID
 - AcademicYear
 - Semester
 - GradeDate
 - AttemptNo
 - Points
 - StateGrade

Хобі (Hobby):

- Опис: довідник можливих захоплень студентів.

- Атрибуты:
 - HobbyID
 - HobbyName

Розділ 2. Проектування бази даних



Загальна характеристика бази даних

База даних UniversityDB призначена для зберігання, обробки та аналізу інформації про освітній процес у вищому навчальному закладі. Вона охоплює дані про студентів, навчальні групи, викладачів, дисципліни, оцінювання, рейтинги та стипендіальне забезпечення.

Проектування бази даних виконано з урахуванням таких вимог:

- підтримка великої кількості записів;
- забезпечення логічної та референтної цілісності;
- можливість історичного зберігання даних;

Логічна структура бази даних

Логічна модель бази даних включає такі групи таблиць:

- Довідкові таблиці
 - Hobby, Subject, Teacher, Group
- Операційні таблиці
 - Student, StudentGrade
- Аналітичні таблиці
 - StudentRating, ScholarshipPeriod

Таке розділення дозволяє спростити підтримку структури та масштабування системи.

Приклад створення довідникової таблиці Hobby:

```
CREATE TABLE [dbo].[Hobby](  
    [HobbyID] [int] IDENTITY(1,1) NOT NULL,  
    NOT NULL,  
    CONSTRAINT [PK_Hobby] PRIMARY KEY CLUSTERED ([HobbyID] ASC),  
    CONSTRAINT [UQ_Hobby_HobbyName] UNIQUE NONCLUSTERED  
    ([HobbyName] ASC)  
);
```

Наявність унікального обмеження на поле HobbyName запобігає дублюванню довідниківих значень під час генерації великих обсягів даних.

Забезпечення унікальності та цілісності

Для забезпечення цілісності даних у базі активно використовуються обмеження UNIQUE та CHECK.

Приклад унікального обмеження для таблиці навчальних груп:

```
CONSTRAINT [UQ_Group_GroupCode] UNIQUE NONCLUSTERED  
([GroupCode] ASC)
```

Це гарантує, що кожна навчальна група має унікальний код, що відповідає реальній організаційній структурі університету.

Перевірка допустимих значень (CHECK)

Для запобігання логічно некоректних даних застосовано перевірки діапазонів значень.

Приклад обмеження для року створення групи:

```
CONSTRAINT [CK_Group_CreationYear]  
CHECK ([CreationYear] >= 1990 AND [CreationYear] <= 2100)
```

Це обмеження забезпечує генерацію тільки реалістичних значень років.

Генерація великих обсягів даних

Фактографічні таблиці

Основне навантаження припадає на таблиці, що зберігають результати навчання та фінансові операції.

Приклад створення таблиці StudentGrade:

```
CREATE TABLE [dbo].[StudentGrade](  
    [StudentGradeID] [bigint] IDENTITY(1,1) NOT NULL,  
    [StudentID] [int] NOT NULL,  
    [TeacherSubjectID] [int] NOT NULL,  
    [AcademicYear] [smallint] NOT NULL,  
    [Semester] [tinyint] NOT NULL,  
    [GradeDate] [date] NOT NULL,  
    [AttemptNo] [tinyint] NOT NULL,  
    [Points] [decimal](5, 2) NOT NULL,  
    [StateGrade] [tinyint] NOT NULL,  
    CONSTRAINT [PK_StudentGrade] PRIMARY KEY CLUSTERED  
    ([StudentGradeID] ASC)  
);
```

Саме ця таблиця містить найбільшу кількість записів та використовується для розрахунку рейтингів.

Реалізація зовнішніх ключів

Зв'язність між таблицями забезпечується зовнішніми ключами.

Приклад зовнішнього ключа між StudentGrade та Student:

```
ALTER TABLE [dbo].[StudentGrade]  
ADD CONSTRAINT [FK_StudentGrade_Student]  
FOREIGN KEY ([StudentID])  
REFERENCES [dbo].[Student] ([StudentID]);
```

Це унеможливилоє створення оцінок для неіснуючих студентів.

Налаштування генератора даних

Генерація даних з урахуванням часових періодів

Для зберігання академічних періодів використовується таблиця ScholarshipPeriod.

Приклад її створення:

```
CREATE TABLE [dbo].[ScholarshipPeriod](  
    [PeriodID] [int] IDENTITY(1,1) NOT NULL,
```

```

[AcademicYear] [smallint] NOT NULL,
[Semester] [tinyint] NOT NULL,
[DateFrom] [date] NOT NULL,
[DateTo] [date] NOT NULL,
CONSTRAINT [PK_ScholarshipPeriod] PRIMARY KEY CLUSTERED
([PeriodID] ASC),
CONSTRAINT [CK_SP_Dates] CHECK ([DateTo] > [DateFrom])
);

```

Це дозволяє коректно моделювати навчальні семестри протягом кількох років.

Забезпечення часових даних

Історія перебування студентів у групах реалізована за допомогою таблиці StudentGroup.

Фрагмент реалізації:

```

CREATE TABLE [dbo].[StudentGroup](
[StudentGroupID] [int] IDENTITY(1,1) NOT NULL,
[StudentID] [int] NOT NULL,
[GroupID] [int] NOT NULL,
[DateFrom] [date] NOT NULL,
[DateTo] [date] NULL,
CONSTRAINT [CK_StudentGroup_Dates]
CHECK ([DateTo] IS NULL OR [DateTo] > [DateFrom])
);

```

Денормалізація для оптимізації

Поле ScholarshipAmount зберігається безпосередньо у таблиці Student.

Приклад значення за замовчуванням:

```

ALTER TABLE [dbo].[Student]
ADD CONSTRAINT [DF_Student_ScholarshipAmount]
DEFAULT ((0)) FOR [ScholarshipAmount];

```

Це дозволяє швидко отримувати поточний розмір стипендії без складних обчислень.

Фізичне проєктування

Збережені процедури

Для реалізації бізнес-логіки розрахунку стипендій використовується збережена процедура.

Фрагмент процедури usp_CalculateScholarship:

```
INSERT INTO dbo.StudentRating(PeriodID, StudentID, RatingValue)
SELECT
    @PeriodID,
    sg.StudentID,
    AVG(CAST(sg.Points AS DECIMAL(10,4)))
FROM dbo.StudentGrade sg
JOIN dbo.Student st
    ON st.StudentID = sg.StudentID
    AND st.IsActive = 1
WHERE sg.AcademicYear = @ay
    AND sg.Semester = @sem
GROUP BY sg.StudentID;
```

Цей фрагмент демонструє розрахунок рейтингу студентів на основі їхніх оцінок.

Забезпечення правил бізнес-логіки

Для складних правил використовуються тригери, які вмикаються спеціальною процедурою.

Приклад процедури увімкнення тригерів:

```
ENABLE TRIGGER dbo.tr_Student_DormRoomCapacity ON dbo.Student;
ENABLE TRIGGER dbo.tr_Group_LeaderMustBelong ON dbo.[Group];
```

Дані генерувалися за допомогою Redgate SQL Data Generator

Внизу прикріпляється звіт про генерацію даних

Target server: (local) LOCALHOST**Target database:** UniversityDB

Date generation started at: 16 грудня 2025 р. 19:47:34 ended at: 16 грудня 2025 р. 19:47:35

[dbo].[Teacher]

Rows inserted: 1,000

Generation started at 16 грудня 2025 р. 19:47:34, taken: less than a second

[dbo].[Subject]

Rows inserted: 100

Generation started at 16 грудня 2025 р. 19:47:35, taken: less than a second

[dbo].[Student]

Rows inserted: 10,000

Generation started at 16 грудня 2025 р. 19:47:35, taken: less than a second

[dbo].[Hobby]

Rows inserted: 1,000

Generation started at 16 грудня 2025 р. 19:47:35, taken: less than a second

[dbo].[Group]

Generation stopped. The generator for column GroupCode could not generate any more values.

Expected number of rows: 1,000

Actual rows inserted: 180

SQL Server did not insert some rows because they contained invalid data, such as duplicate values in a unique column.

Generation started at 16 грудня 2025 р. 19:47:35, taken: less than a second

Призначення та загальна характеристика сховища даних

Сховище даних UniversityDW призначено для аналітичної обробки інформації, отриманої з операційної бази даних університету (OLTP).

Основною метою створення сховища є підтримка формування аналітичних звітів, оцінювання навчальних результатів студентів, аналізу ефективності викладання та розрахунку стипендіального забезпечення.

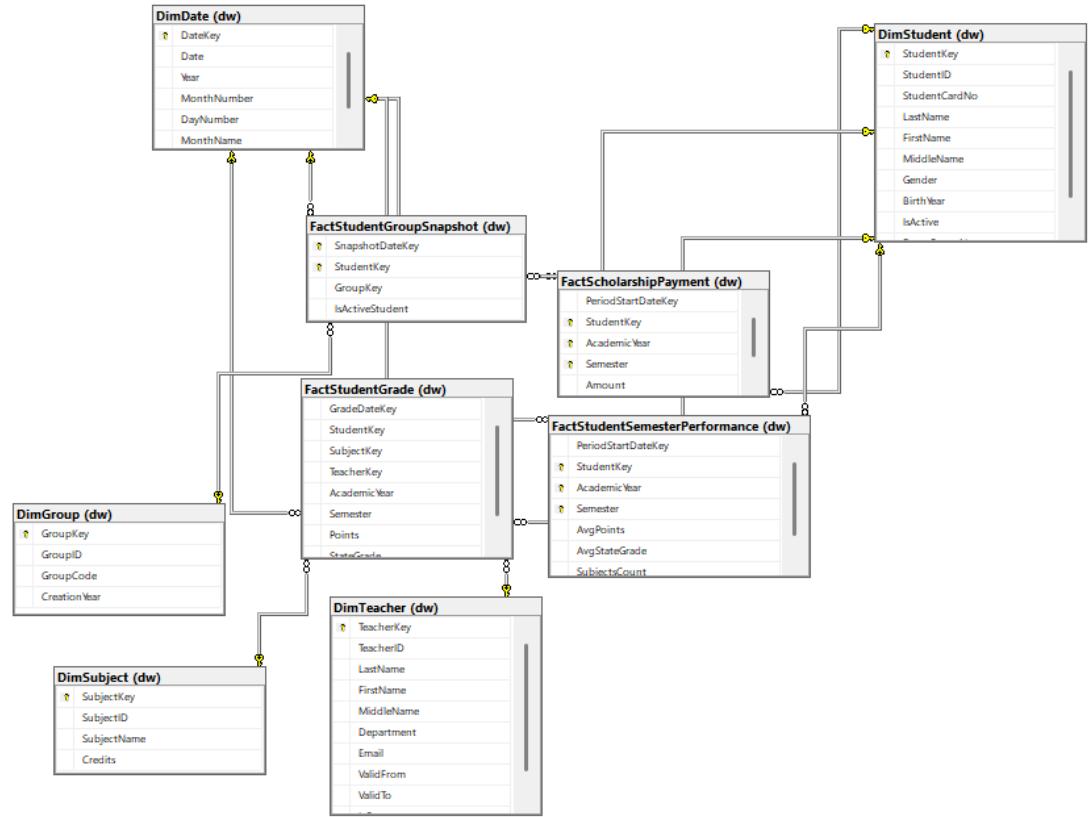
На відміну від транзакційної бази даних, UniversityDW оптимізоване для читання великих обсягів історичних даних і підтримує багатовимірний аналіз (OLAP). Завантаження даних у сховище передбачається виконувати окремими ETL-процесами, які не входять до даного етапу реалізації.

Архітектура сховища даних

Сховище даних реалізоване відповідно до зіркоподібної схеми (Star Schema), яка складається з:

- таблиць вимірів;
- таблиць фактів;

Для логічного розділення об'єктів сховища використовується окрема схема dw, що дозволяє чітко відокремити аналітичну модель від OLTP-рівня.



CREATE SCHEMA dw;

Керування версіями та безпечний повторний запуск

Для забезпечення можливості повторного виконання скрипта без помилок використано перевірки на існування об'єктів перед їх видаленням:

```
IF OBJECT_ID(N'dw.FactStudentGrade', 'U') IS NOT NULL  
DROP TABLE dw.FactStudentGrade;
```

Такий підхід дозволяє використовувати скрипт як baseline-структурну сховища даних.

Таблиці вимірів

Вимір дати (DimDate)

Таблиця DimDate є центральним календарним виміром, який використовується всіма таблицями фактів.

```
CREATE TABLE dw.DimDate
(
    DateKey INT NOT NULL PRIMARY KEY,
    [Date] DATE NOT NULL,
    [Year] SMALLINT NOT NULL,
    MonthNumber TINYINT NOT NULL,
    DayNumber TINYINT NOT NULL,
    MonthName NVARCHAR(20) NOT NULL
);
```

Поле DateKey реалізовано у форматі YYYYMMDD, що дозволяє уникнути обчислень при приєднанні до фактів.

Вимір студентів (DimStudent, SCD Type 1)

Таблиця DimStudent зберігає актуальні атрибути студентів без збереження історії змін (SCD Type 1).

```
CREATE TABLE dw.DimStudent
(
    StudentKey INT IDENTITY PRIMARY KEY,
    StudentID INT NOT NULL,
    LastName NVARCHAR(50) NOT NULL,
    FirstName NVARCHAR(50) NOT NULL,
    IsActive BIT NOT NULL,
    DormRoomNo INT NULL
);
```

Поле StudentKey є сурогатним ключем, тоді як StudentID — натуральним ключем з OLTP-системи.

Вимір навчальних груп (DimGroup)

```
CREATE TABLE dw.DimGroup
(
    GroupKey INT IDENTITY PRIMARY KEY,
    GroupID INT NOT NULL,
    GroupCode NVARCHAR(30) NOT NULL,
    CreationYear SMALLINT NOT NULL
);
```

Цей вимір використовується для групування студентів у звітах і звізах за роками набору.

Вимір дисциплін (DimSubject)

```
CREATE TABLE dw.DimSubject
(
    SubjectKey INT IDENTITY PRIMARY KEY,
    SubjectID INT NOT NULL,
    SubjectName NVARCHAR(120) NOT NULL,
    Credits TINYINT NULL
);
```

Таблиця забезпечує аналіз успішності за дисциплінами та кредитним навантаженням.

Вимір викладачів (DimTeacher, SCD Type 2)

Таблиця DimTeacher реалізує повну історичність змін (Slowly Changing Dimension Type 2).

```
CREATE TABLE dw.DimTeacher
(
    TeacherKey INT IDENTITY PRIMARY KEY,
    TeacherID INT NOT NULL,
    Department NVARCHAR(100),
    ValidFrom DATETIME2 NOT NULL,
    ValidTo DATETIME2 NOT NULL,
    IsCurrent BIT NOT NULL
);
```

Завдяки полям ValidFrom, ValidTo та IsCurrent стає можливим аналіз результатів студентів у контексті змін викладацького складу.

Таблиці фактів

Знімок студентів за групами (FactStudentGroupSnapshot)

```
CREATE TABLE dw.FactStudentGroupSnapshot
(
    SnapshotDateKey INT NOT NULL,
    StudentKey INT NOT NULL,
    GroupKey INT NOT NULL,
    IsActiveStudent BIT NOT NULL
);
```

Фактова таблиця використовується для звітів типу snapshot, наприклад:
кількість активних студентів у групах на конкретну дату.

Факт оцінювання студентів (FactStudentGrade)

```
CREATE TABLE dw.FactStudentGrade
(
    GradeDateKey INT NOT NULL,
    StudentKey INT NOT NULL,
    SubjectKey INT NOT NULL,
    TeacherKey INT NOT NULL,
    AcademicYear SMALLINT NOT NULL,
    Semester TINYINT NOT NULL,
    Points DECIMAL(5,2) NOT NULL
);
```

Це атомарна таблиця фактів, яка зберігає всі оцінки студентів і слугує джерелом для більш агрегованих фактів.

Семестрові показники (FactStudentSemesterPerformance)

```
CREATE TABLE dw.FactStudentSemesterPerformance
(
    StudentKey INT NOT NULL,
    AcademicYear SMALLINT NOT NULL,
```

```
Semester TINYINT NOT NULL,  
AvgPoints DECIMAL(10,4) NOT NULL,  
SubjectsCount INT NOT NULL  
);
```

Таблиця використовується для:

- рейтинг студентів;
- формування довідок;
- аналітики успішності.

Факт виплат стипендій (FactScholarshipPayment)

```
CREATE TABLE dw.FactScholarshipPayment  
(  
    StudentKey INT NOT NULL,  
    AcademicYear SMALLINT NOT NULL,  
    Semester TINYINT NOT NULL,  
    Amount MONEY NOT NULL  
);
```

Дана таблиця дозволяє аналізувати фінансове забезпечення студентів у розрізі семестрів.

Забезпечення цілісності даних

Для всіх таблиць фактів реалізовано зовнішні ключі до відповідних вимірів:

```
FOREIGN KEY (StudentKey) REFERENCES dw.DimStudent(StudentKey)
```

Це гарантує повну референтну цілісність сховища.

Переваги реалізованої моделі

- підтримка історичних змін (SCD2);
- оптимізація під аналітичні запити;
- масштабованість;

Розділ 3. Реалізація ETL-процесів

На етапі екстракції було реалізовано підключення до транзакційної бази даних UniversityDB за допомогою компонента OLE DB Source.

Стратегії витягування даних

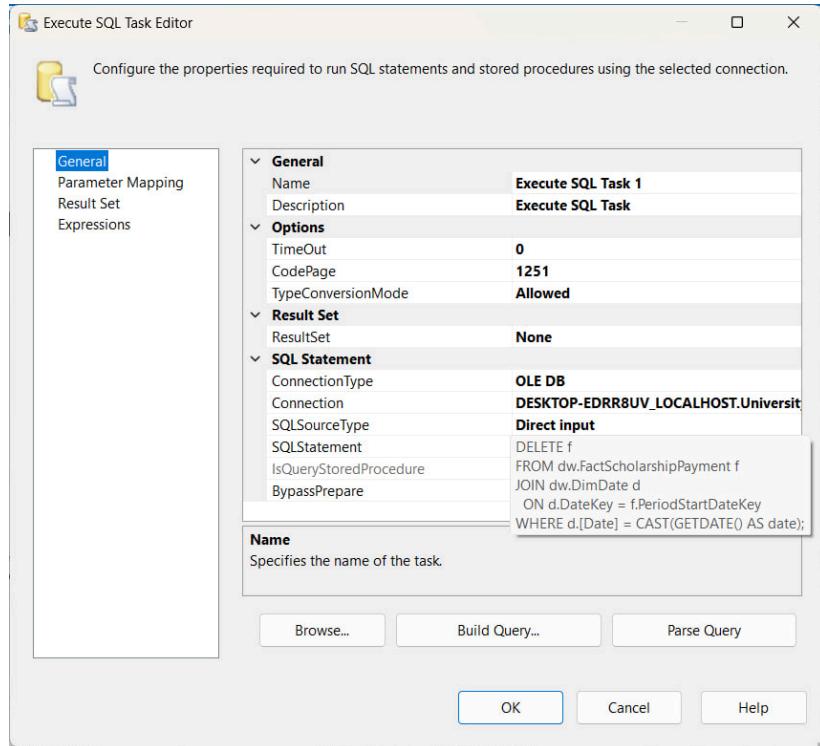
У проекті реалізовано дві основні стратегії завантаження залежно від типу даних:

- 1. Повне завантаження (Full Load):** застосовується для довідниковых таблиць вимірів, таких як dw.DimStudent та dw.DimTeacher. Оскільки обсяг цих даних відносно стабільний, під час кожного запуску ETL-пакета виконується повне перечитування таблиць джерела. Це гарантує актуальність персональних даних студентів та викладачів у сховищі.
- 2. Інкрементальне завантаження (Incremental Load):** застосовується для таблиць фактів, зокрема dw.FactStudentGrade та dw.FactScholarshipPayment. Враховуючи великий обсяг даних (понад 500 000 записів в основній таблиці фактів), витягування здійснюється частинами — у розрізі навчальних років та семестрів. Це дозволяє значно знизити навантаження на мережу та сервер бази даних, оновлюючи лише ті періоди, які були змінені або додані з моменту останнього завантаження.

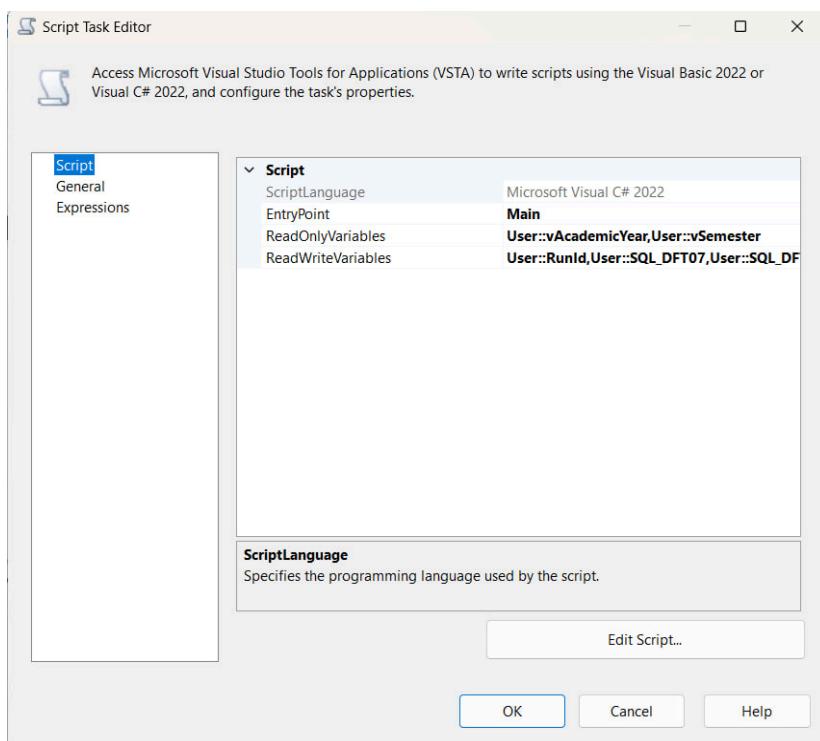
Додаткові компоненти SSIS

Реалізовано такі додаткові компоненти:

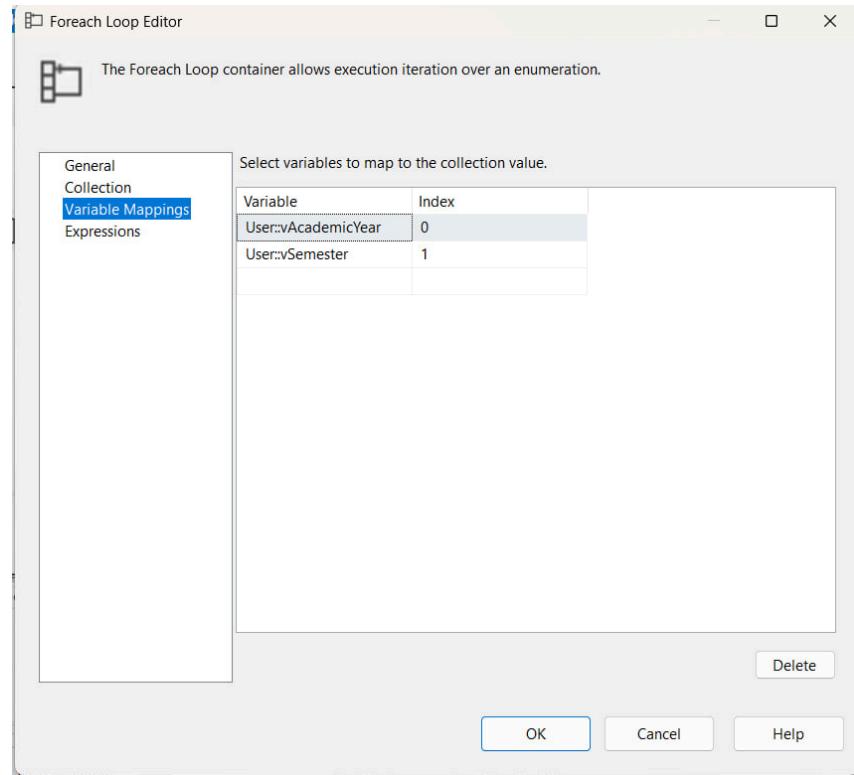
- Execute SQL Task - для підготовчих операцій



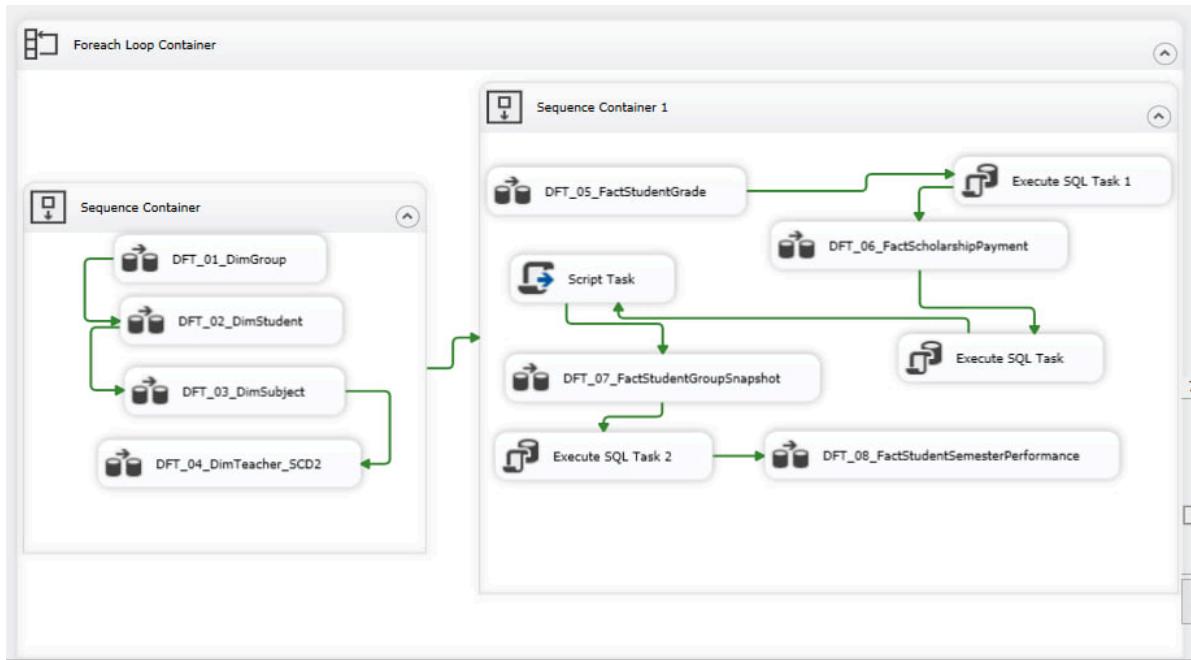
- Script task - для складної бізнес логіки



- For Each Loop Container - для масової обробки

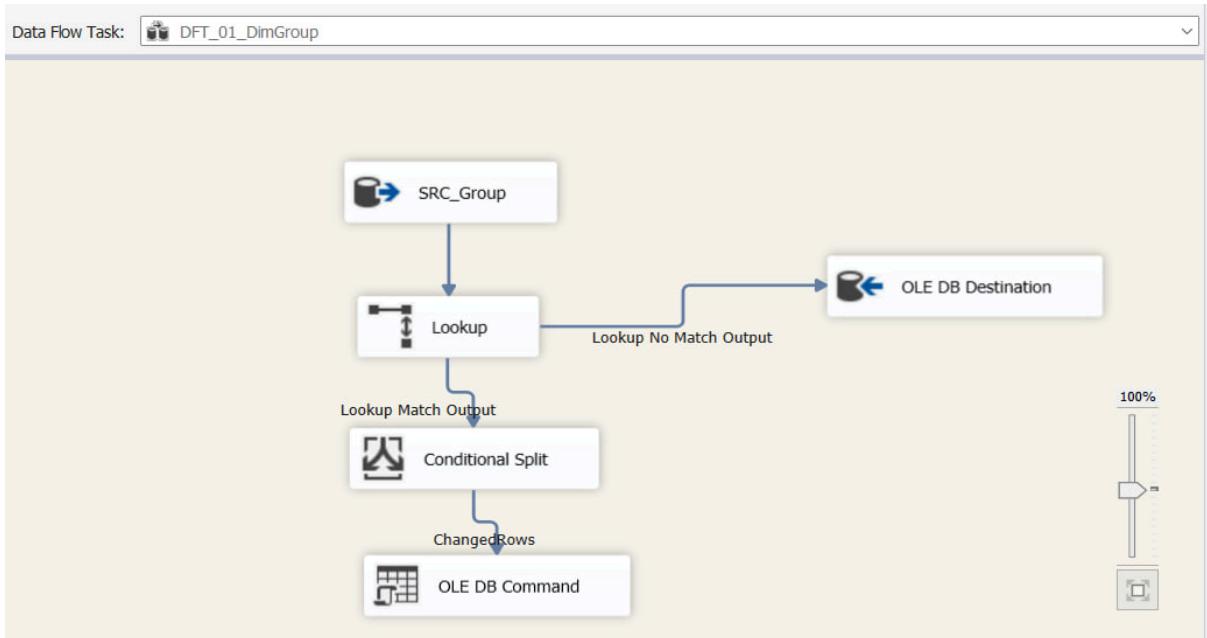


- Sequence Container - для організації пакетів

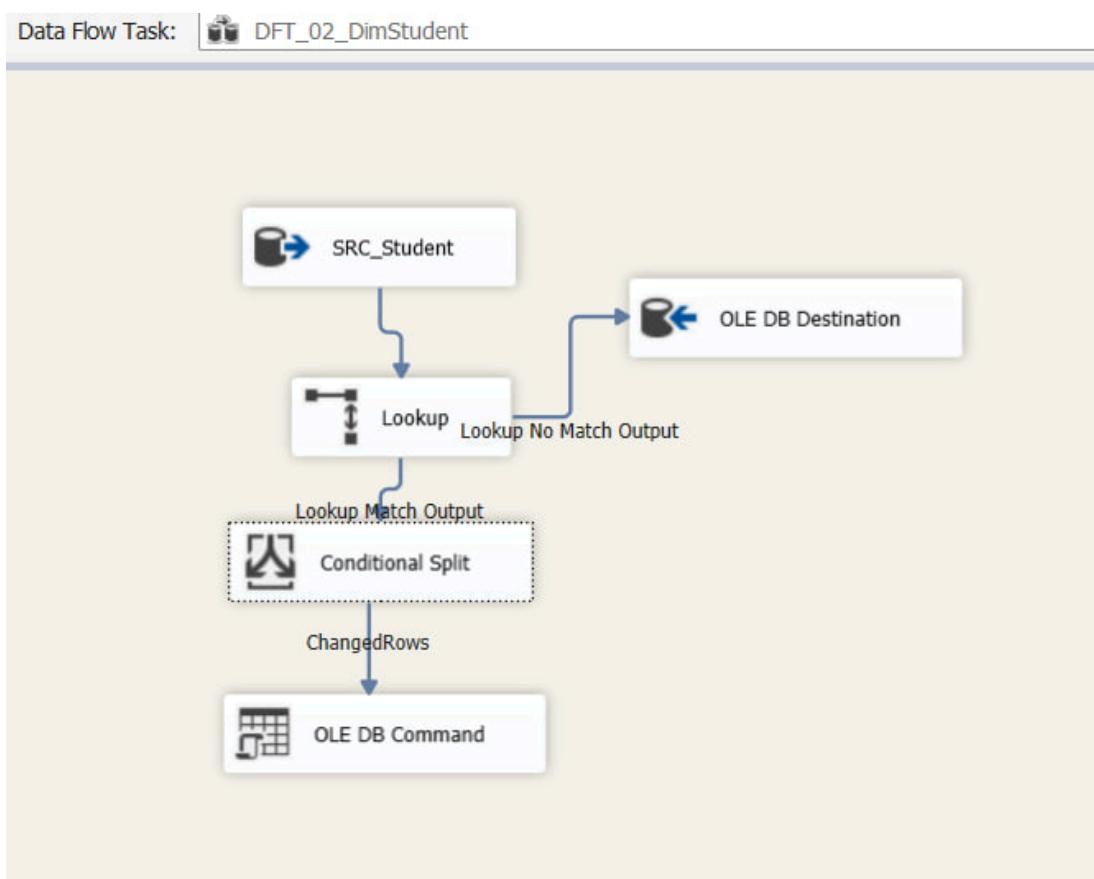


Структура таблиць вимірів

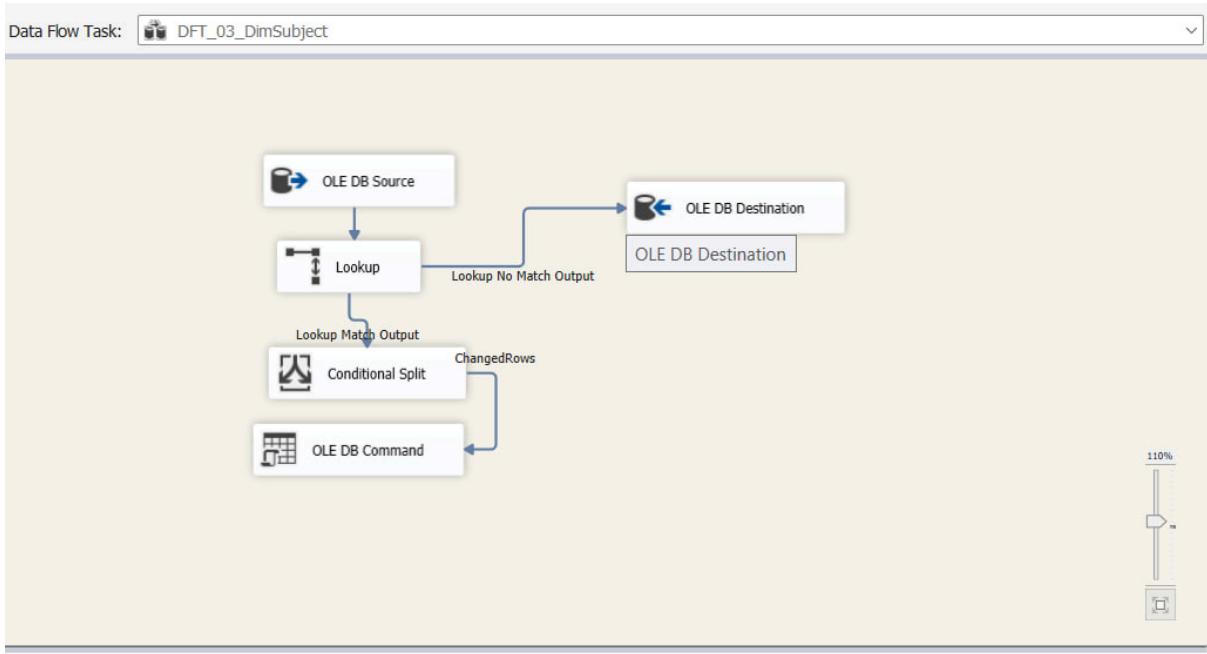
DimGroup



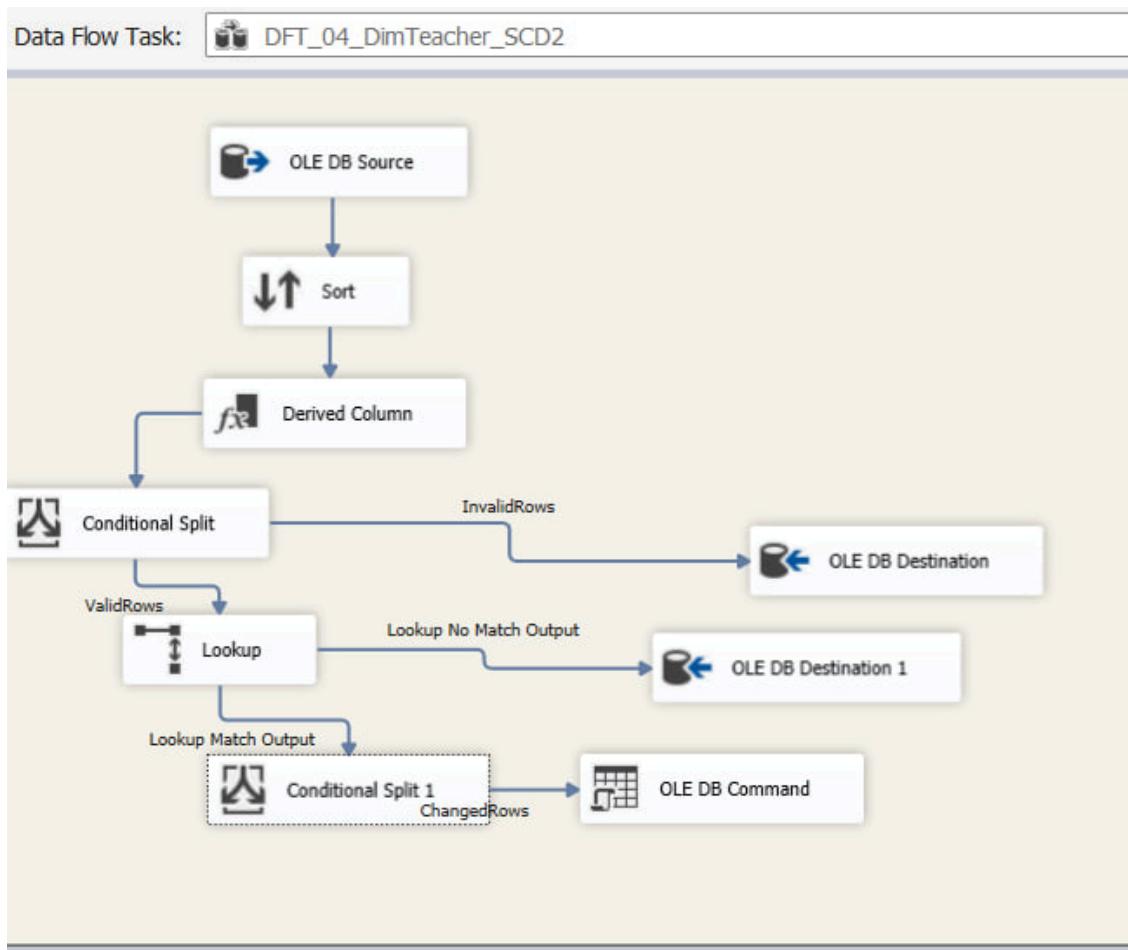
DimStudent



DimSubject



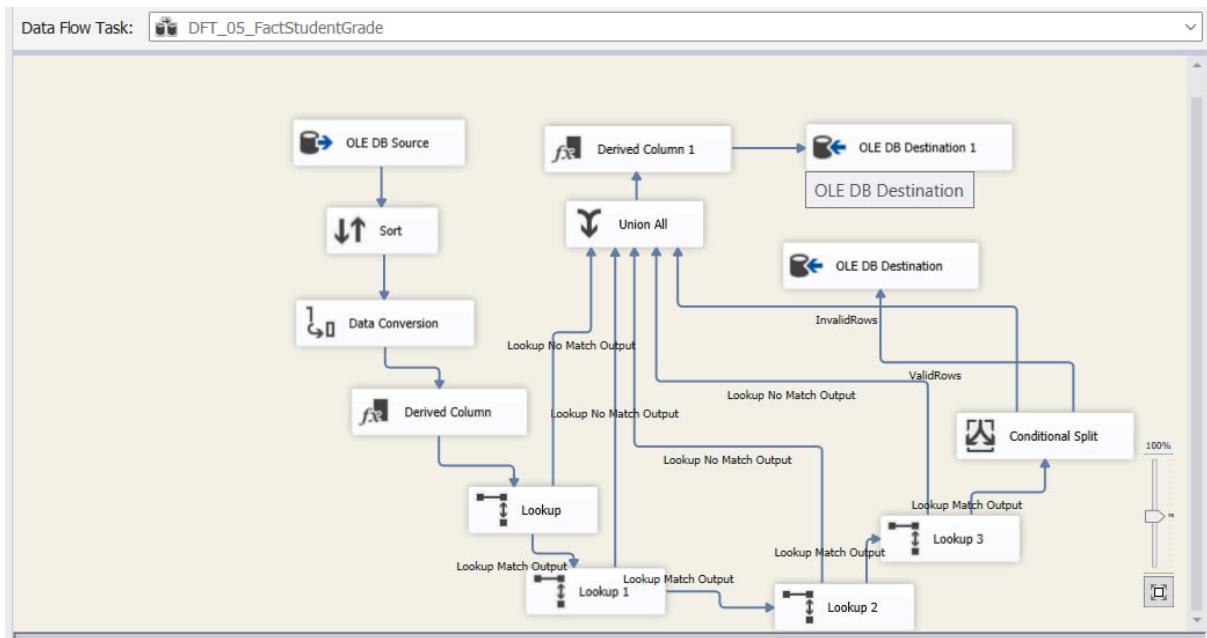
DimTeacher



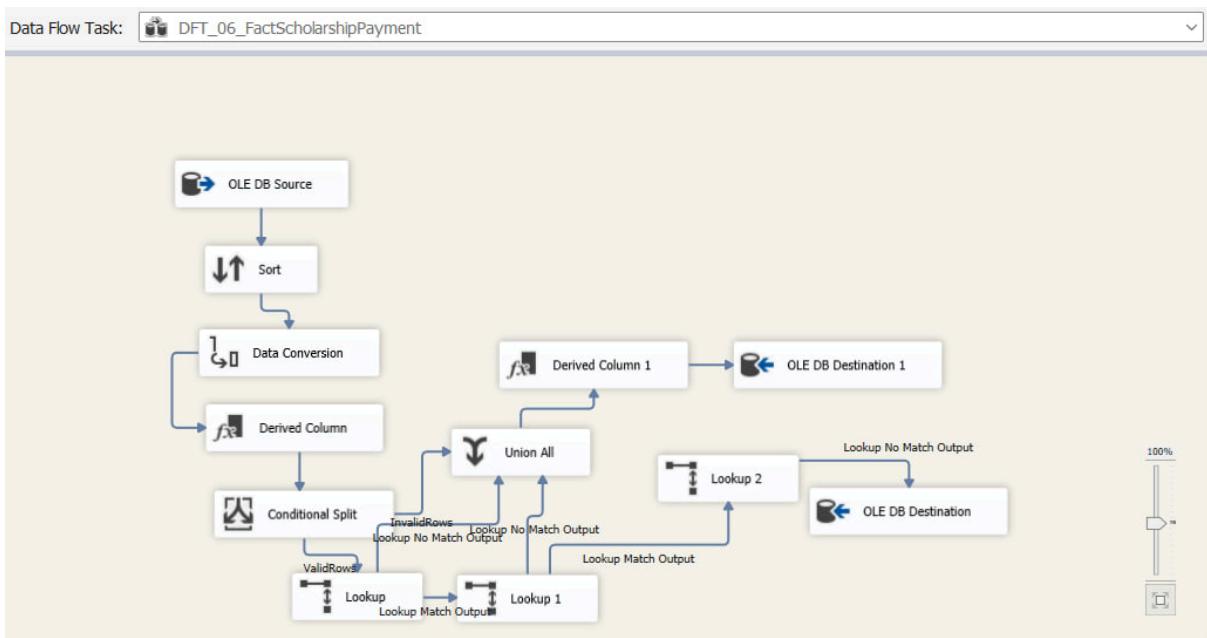
Виконані такі трансформації: Sort, Derived Column, Conditional Split, Lookup, OLE DB Command.

Структура таблиць фактів

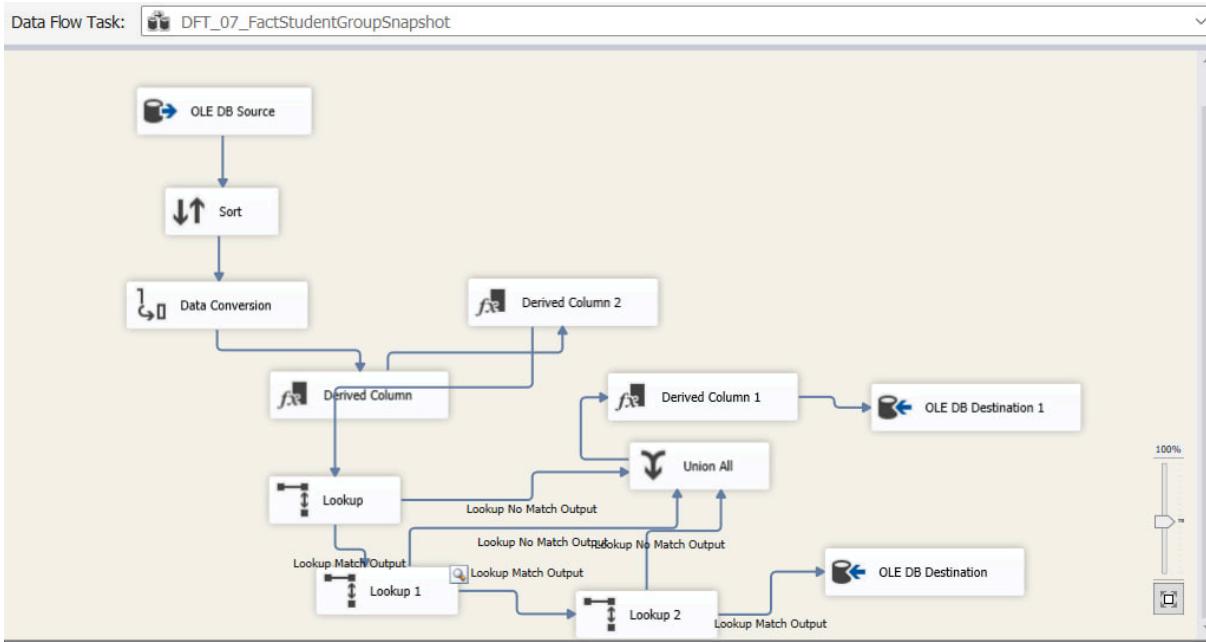
FactStudentGrade



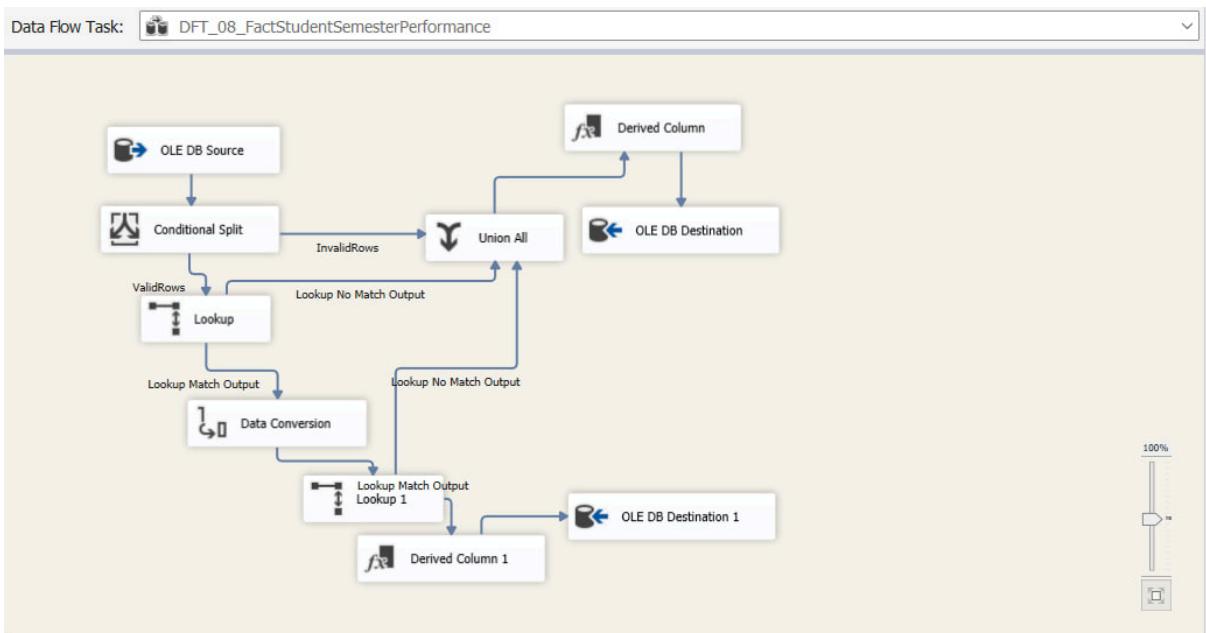
FactScholarshipPayment



FactStudentGroupSnapshot

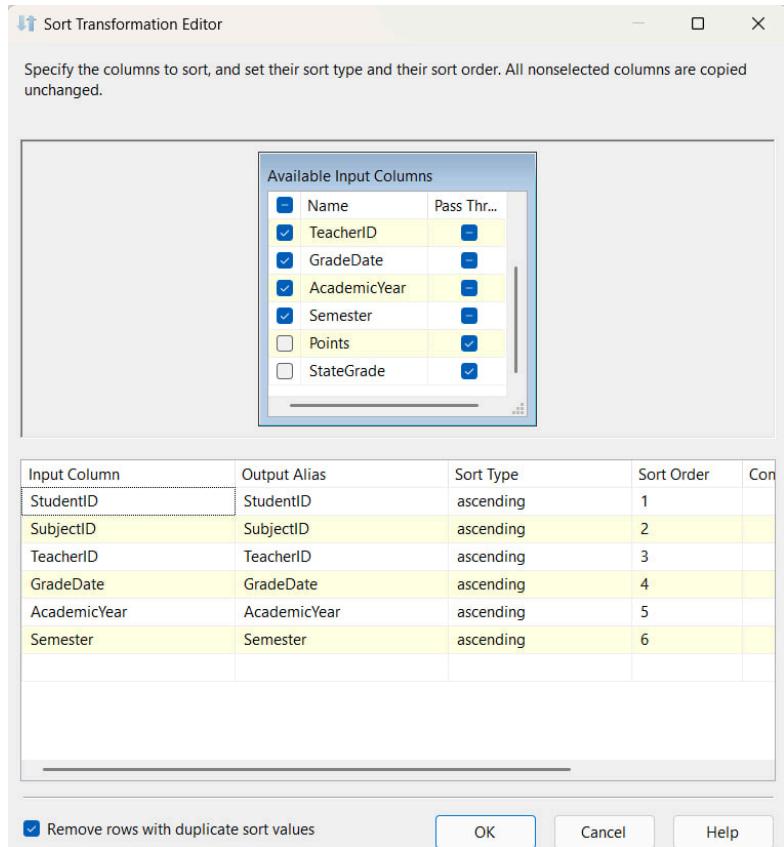


FactStudentSemesterPerformance

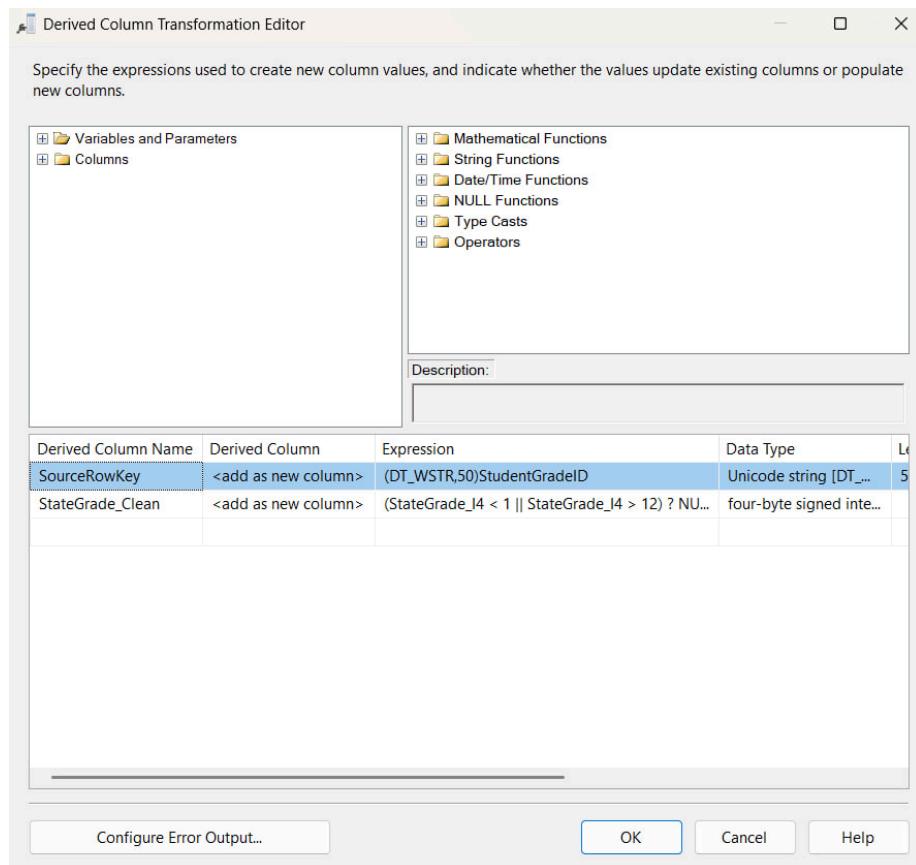


Використані трансформації: Conditional Split, Lookup, Data Conversion, Lookup, Derived Column, Union All.

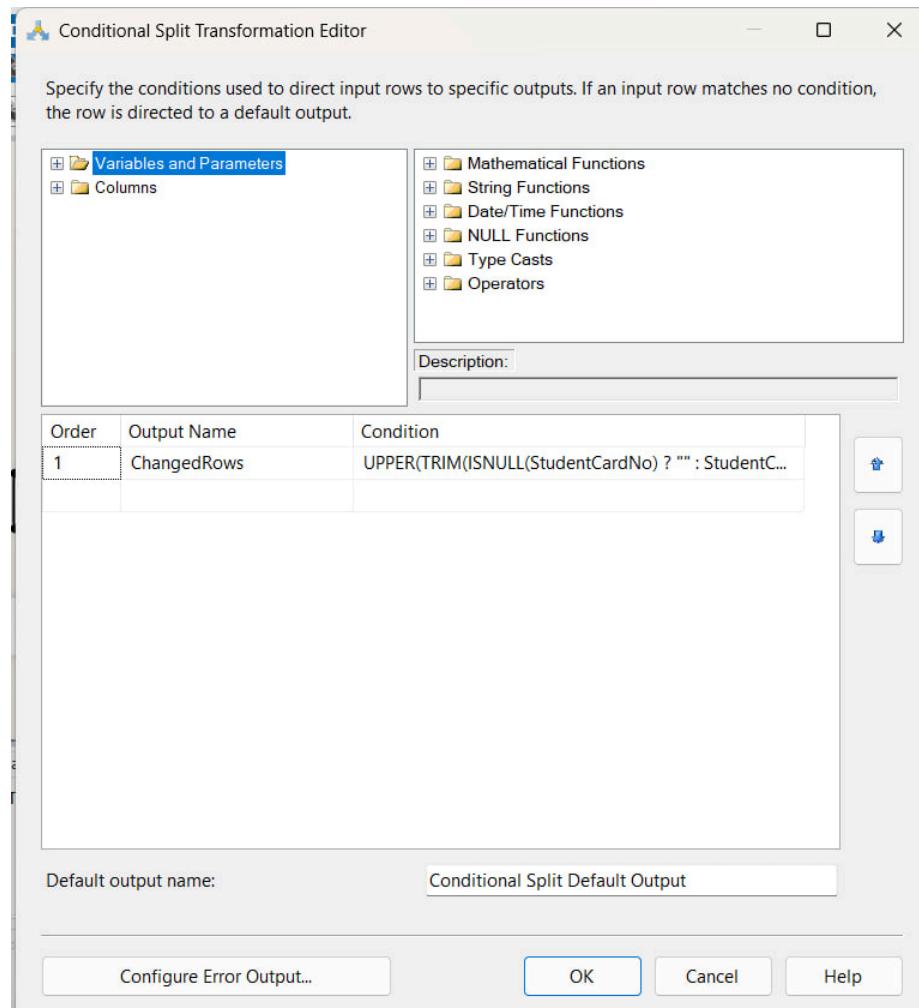
Sort - сортує потік даних за одним або кількома полями.



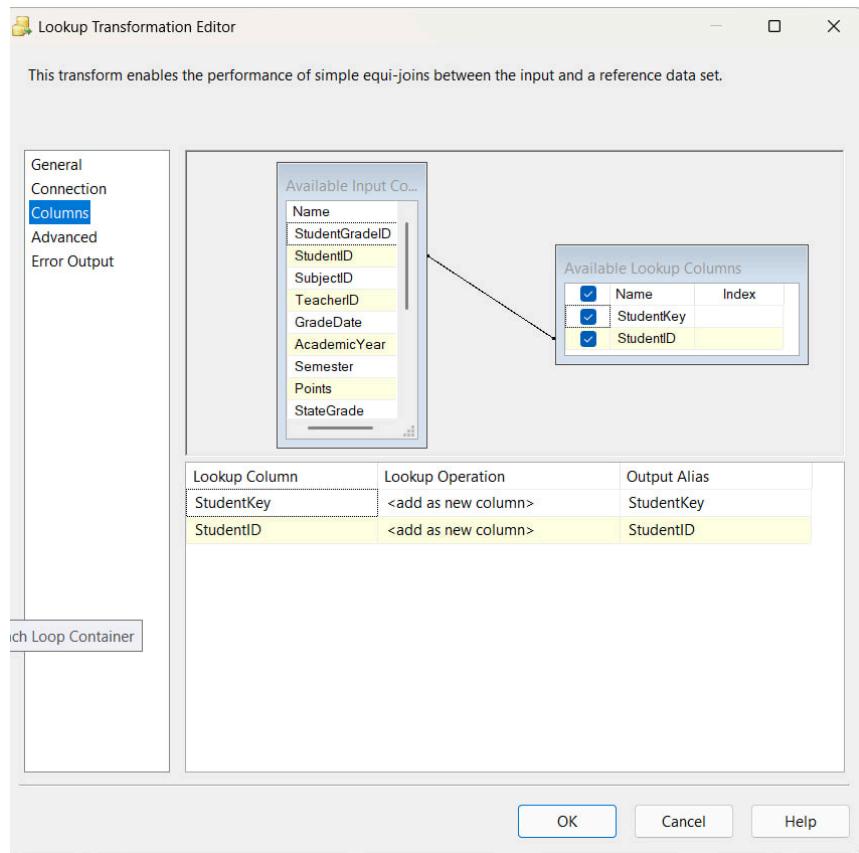
Derived Column - створює нову колонку або змінює значення існуючої на основі виразу.



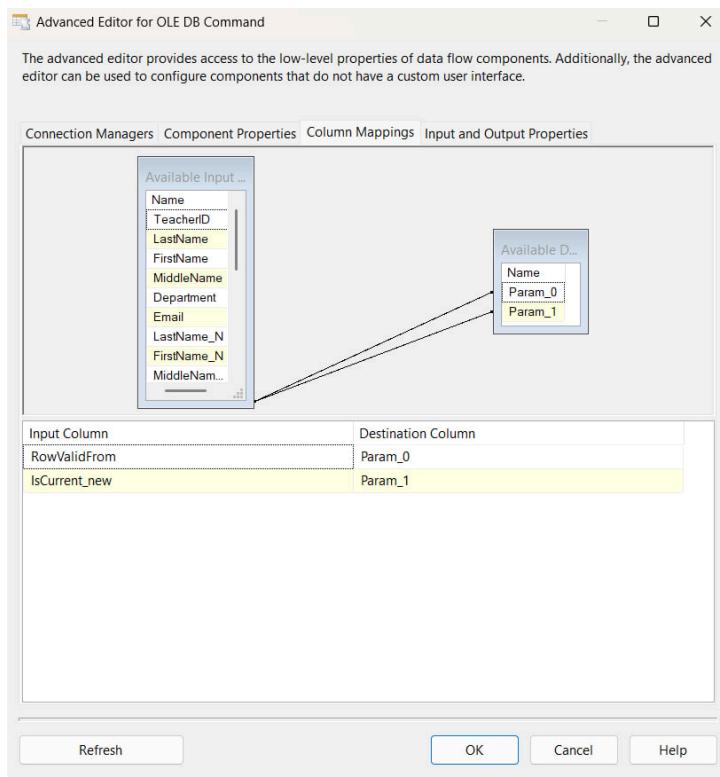
Conditional Split - розділяє потік даних на кілька гілок залежно від умова.



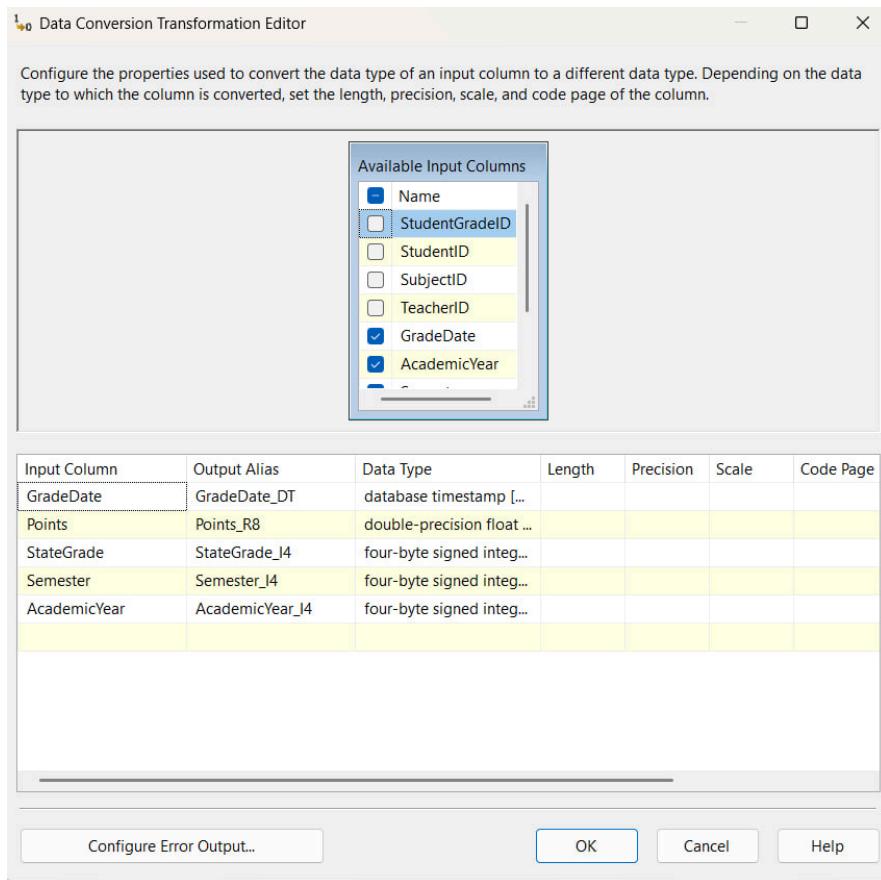
Lookup - знаходить відповідні значення в іншій таблиці.



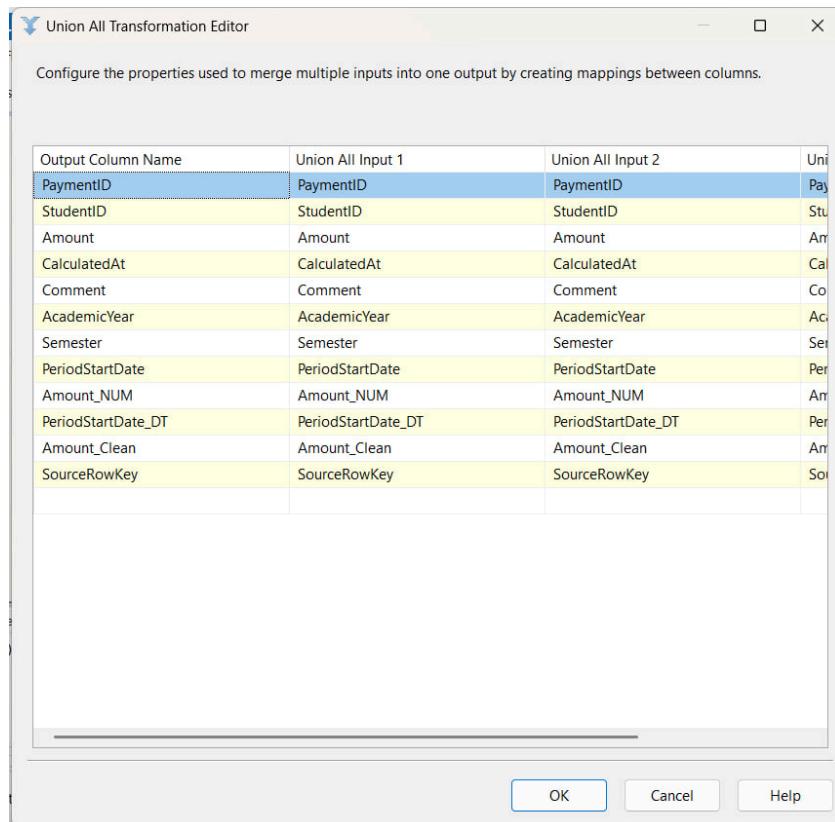
OLE DB Command - виконує SQL-команди для кожного рядка потоку даних.



Data Conversion - перетворює типи даних між різними форматами.

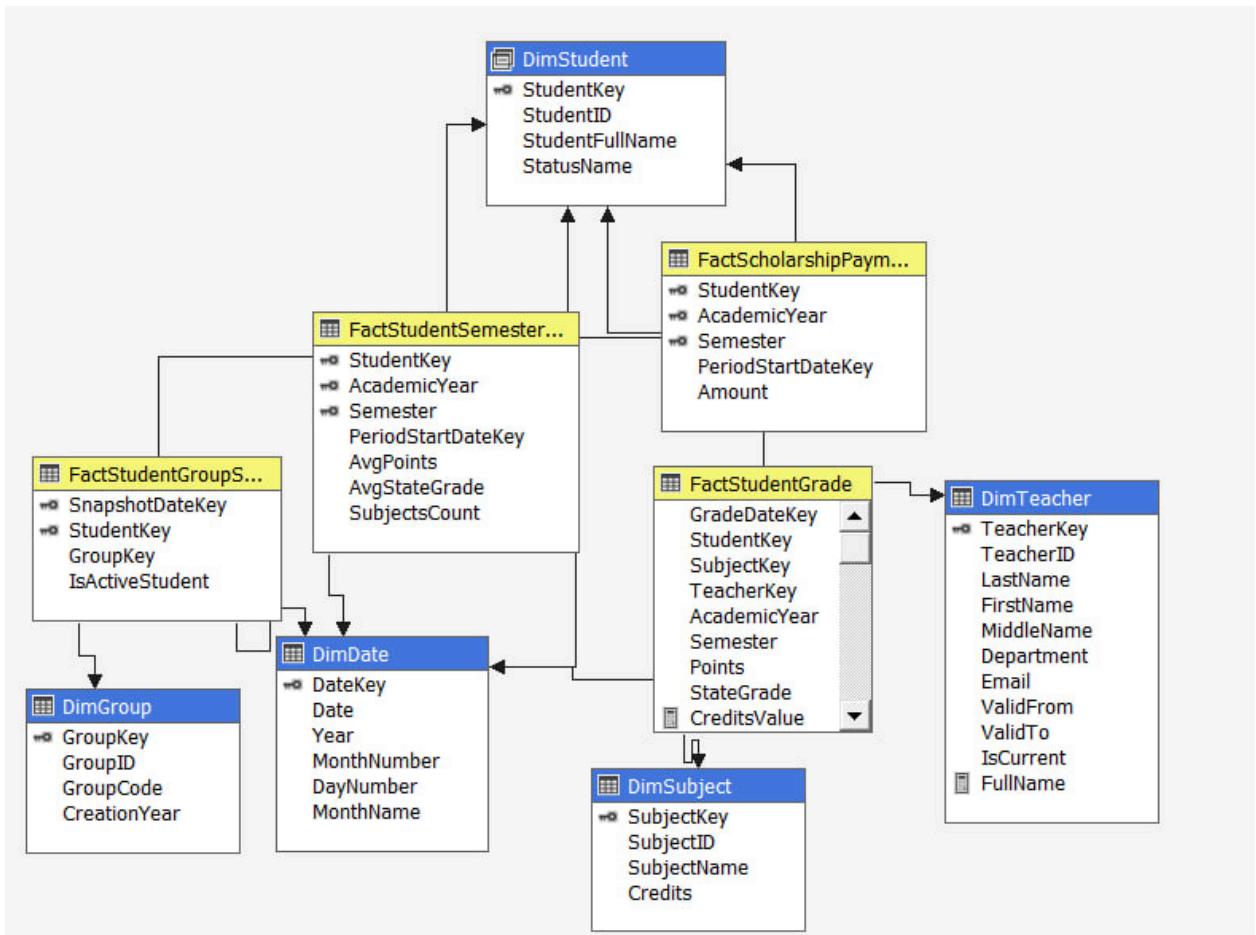


Union All - об'єднує декілька потоків даних в один.



Розділ 4. Побудова OLAP-куба та аналітичні звіти

Структура куба



Виміри та міри

Measure Groups					
Dimensions		[all] Fact Scholarship Payment	[all] Fact Student Grade	[all] Fact Student Group Snapshot	[all] Fact Student Semester Performa...
Dim Date	Date Key	Date Key	Date Key	Date Key	Date Key
Dim Group				Group Key	
Dim Student	Student Key	Student Key	Student Key	Student Key	Student Key
Dim Subject		Subject Key			
Dim Teacher	Teacher Key	Teacher Key			

Додаткові міри

[AverageScore]

[Measures].[AllPoints] / [Measures].[Fact Student Grade Count]

[ScholarshipPercent]

[Measures].[TotalAmount] / ([Measures].[TotalAmount], [Dim Student].[Student Full Name].[All])

[CostPerPoint]

[Measures].[TotalAmount] / [Measures].[AllPoints]

[StudentRank]

RANK([Dim Student].[Student Full Name].CurrentMember, [SortedStudents])

[WeightedAverage]

IIF(

[Measures].[AllCredits] = 0,

NULL,

[Measures].[SumWeightedPoints] / [Measures].[AllCredits]

)

[PreviousYear]

([Measures].[TotalAmount], PARALLELPERIOD([Dim Date].[Hierarchy].[Year], 1, [Dim Date].[Hierarchy].CurrentMember))

[3MonthAverageScore]

AVG(LASTPERIODS(3, [Dim Date].[Hierarchy].CurrentMember),
[Measures].[AllPoints] / [Measures].[Fact Student Grade Count])

[ScorePer1000Currency]

IIF(

[Measures].[TotalAmount] = 0,

0,

([Measures].[WeightedAverage] / [Measures].[TotalAmount]) * 1000

)

[YearOverYearGrowth]

IIF(ISEMPTY([Measures].[PreviousYear]) OR [Measures].[PreviousYear] = 0,
NULL,

([Measures].[AllPoints] - [Measures].[PreviousYear]) / [Measures].[PreviousYear])

Партиції

University D...cube [Design]*

Cube Struct... Dimension Usage Calculations KPIs Actions Partitions Aggregations Perspectives Translations Browser

Fact Scholarship Payment (1 Partition)

Item	Partition Name ↑	Source	Estimated Rows	Storage Mode	Aggregation Design
1	Fact Scholarship Payment	FactScholarshipPayment	24030	MOLAP	AggregationDesign

New Partition... Storage Settings...

Fact Student Grade (5 Partitions)

Item	Partition Name ↑	Source	Estimated Rows	Storage Mode	Aggregation Design
1	FactStudentGrade20	SELECT [StudentKey], [GradeDateKey], [SubjectKey], [Academ...]	100000	MOLAP	AggregationDesign
2	FactStudentGrade21	SELECT [StudentKey], [GradeDateKey], [SubjectKey], [Academ...]	100000	MOLAP	AggregationDesign
3	FactStudentGrade22	SELECT [StudentKey], [GradeDateKey], [SubjectKey], [Academ...]	100000	MOLAP	AggregationDesign
4	FactStudentGrade23	SELECT [StudentKey], [GradeDateKey], [SubjectKey], [Academ...]	100000	MOLAP	AggregationDesign
5	FactStudentGrade24	SELECT [StudentKey], [GradeDateKey], [SubjectKey], [Academ...]	100000	MOLAP	AggregationDesign

New Partition... Storage Settings...

Fact Student Group Snapshot (1 Partition)

Item	Partition Name ↑	Source	Estimated Rows	Storage Mode	Aggregation Design
1	Fact Student Group Snapshot	FactStudentGroupSnapshot	10000	MOLAP	AggregationDesign

New Partition... Storage Settings...

Fact Student Semester Performance (1 Partition)

Item	Partition Name ↑	Source	Estimated Rows	Storage Mode	Aggregation Design
1	Fact Student Semester Perfo...	FactStudentSemesterPerformance	10000	MOLAP	AggregationDesign

Агрегації

Cube Struct... Dimension Usage Calculations KPIs Actions Partitions Aggregations Perspectives Translations Browser

Aggregations Estimated Partition Size Partitions

Fact Scholarship Payment (1 Aggregation Design)	3	24030	Fact Scholarship Payment
Fact Student Grade (1 Aggregation Design)	5	100000	FactStudentGrade23, FactStudentGrade20, FactStudentGrade21, ...
Fact Student Group Snapshot (1 Aggregation Design)	4	10000	Fact Student Group Snapshot
Fact Student Semester Performance (1 Aggregation Design)	3	10000	Fact Student Semester Performance
AggregationDesign			

Перспективи

Cube Structure Overview					
	Cube Objects	Object Type	Perspective Name	Perspective Name	Perspective Name
	University DW	Name	Student Performance	Financials	Group Analytics
	Fact Student Grade Count	Measure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	MinPoint	Measure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	MaxPoint	Measure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	AllPoints	Measure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	AllCredits	Measure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	SumWeightedPoints	Measure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
-	Fact Student Group Snapshot	MeasureGroup	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	Fact Student Group Snap...	Measure	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
-	Fact Student Semester Perf...	MeasureGroup	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Fact Student Semester P...	Measure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
-	Dimensions				
	Dim Date	CubeDimension	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	Dim Group	CubeDimension	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Dim Student	CubeDimension	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Dim Subject	CubeDimension	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Dim Teacher	CubeDimension	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
-	KPIs				
	StudentPerformanceKPI	Kpi	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	ScholarshipBudgetKPI	Kpi	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
-	Calculations				
	SortedStudents	NamedSet	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	AverageScore	CalculatedMeasure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	ScholarshipPercent	CalculatedMeasure	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	CostPerPoint	CalculatedMeasure	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	StudentRank	CalculatedMeasure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	WeightedAverage	CalculatedMeasure	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	PreviousYear	CalculatedMeasure	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	3MonthAverageScore	CalculatedMeasure	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	ScorePer1000Currency	CalculatedMeasure	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	YearOverYearGrowth	CalculatedMeasure	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Звіти

Список студентів по групах

A screenshot of Microsoft Word showing a table with 10 rows and 2 columns. The first column contains student full names and the second column contains their average scores.

Student Full Name	Average Score
Gwendolyn	78,72
Herman	
Shanna	
Christy Marie	77,38
Priscilla	
Emma Megan	77,22
Virginia	
Leonardo	76,36
Aaron Tara	
Angie Laurie	76,06
Jami	
Donovan Alan	75,96
Douglas	
Tracy Irma	75,8
Marcus	
Patrick Carlos	75,78
Leslie	
Angel	75,68
Geffrey Indie	

Рейтинг студентів

Group Code	Student Full Name
AI20	Aaron Charles Enrique
	Aaron Darcy Gustavo
	Aaron Gabriel Betty
	Aaron Jaime Holly
	Aaron Jodie Crystal
	Aaron Kerry Ricardo
	Aaron Kristin Trisha
	Aaron Kurt Penny
	Aaron Rochelle Chan
	Aaron Scottie Shanda
	Aaron Seth Christina
	Aaron

Довідка студента про рейтинг та стипендію

Design Preview

Choose group AI23 FullStudentName Aaron Jodie Crystal

[View Report](#)

1 of 1 Find Next

Student name: Aaron Jodie Crystal
Student rank: 5
Total Amount: 0

Аналіз успішності по предметах та викладачах

Design		Preview			
1	of 2	100%	Find	Next	
BURNETT GINA GLORIA					
CAMPBELL GLENN ANNIE					
CISNEROS RANDAL TAMIKO					
COFFEY KIMBERLEY ALONZO					
COHEN LEON KISHA					
CORTEZ TERRI HERBERT					68,923895253 6825
CRAWFORD CASSIE ALFONSO	67,912438625 2046				
DAVID SUMMER RANDI					
DAVIDSON KARL DARLA				66,946791862 2848	
DODSON MISTI MICHAEL	66,807692307 6923				
FLYNN REBECCA TARYN			67,406885758 9984		
FRANK CHARLIE MARCI					
FREEMAN					

Висновки

У ході виконання курсової роботи було розроблено комплексну інформаційну систему обліку успішності студентів та автоматизації нарахування стипендій. На основі проведеного дослідження та практичної реалізації можна зробити такі висновки:

1. **Аналіз предметної області та проектування:** Було детально вивчено процеси адміністрування навчального процесу. Розроблена концептуальна та логічна моделі бази даних UniversityDB дозволили ефективно структурувати дані про студентів, викладачів, предмети та оцінки. Використання третьої нормальної форми (ЗНФ) забезпечило цілісність даних та відсутність надмірності.
2. **Генерація великих обсягів даних:** Для тестування продуктивності системи в умовах, наблизених до реальних, було реалізовано алгоритм генерації 500 000 записів у таблиці фактів оцінок. Це дозволило перевірити ефективність роботи індексів та швидкість виконання складних SQL-запитів.
3. **Автоматизація бізнес-логіки:** Створена збережена процедура `usp_CalculateScholarship` дозволила повністю автоматизувати процес рейтингування студентів. Використання віконних функцій та динамічного розподілу бюджету (виділення топ-10% для підвищеної стипендії) забезпечило точність та прозорість фінансових нарахувань.
4. **Розробка сховища даних та ETL:** Побудовано аналітичне сховище UniversityDW за архітектурою «Зірка». Розроблено пакети SSIS, які реалізують повний цикл ETL:
 - **Extract:** Витягування даних із транзакційної бази.
 - **Transform:** Очищення даних, обробка NULL-значень та реалізація інкрементального завантаження.
 - **Load:** Ефективне завантаження великих масивів даних. Створена система логування помилок у таблиці dw.ETL_ErrorLog забезпечила високу відмовостійкість процесу інтеграції.
5. **Багатовимірний аналіз (OLAP):** Реалізовано OLAP-куб засобами SSAS, що дозволило перетворити сирі дані на аналітичну інформацію. Завдяки ієрархіям вимірів (зокрема часовому вимірю DimDate) та обчислюваним мірам, час отримання складних звітів скоротився з десятків секунд до мілісекунд.
6. **Аналітична звітність:** На базі SSRS було спроектовано набір інтерактивних звітів, які надають керівництву навчального закладу

візуальну інформацію про динаміку успішності та структуру стипендіальних виплат.

Практична цінність роботи полягає у створенні масштабованої системи, яка здатна обробляти дані великого університету, забезпечуючи при цьому високу швидкість аналізу та мінімізацію людського фактора при прийнятті рішень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- 1. SQL Server Integration Services (SSIS) Documentation. Microsoft Learn.** URL: <https://learn.microsoft.com/en-us/sql/integration-services/>
- 2. SQL Server Analysis Services (SSAS) Documentation. Microsoft Learn.** URL: <https://learn.microsoft.com/en-us/sql/analysis-services/>