Technical Documentation

Misi 2 Kelompok 6. LPMPP

Step 1: Persiapan Data (Data Preparation)

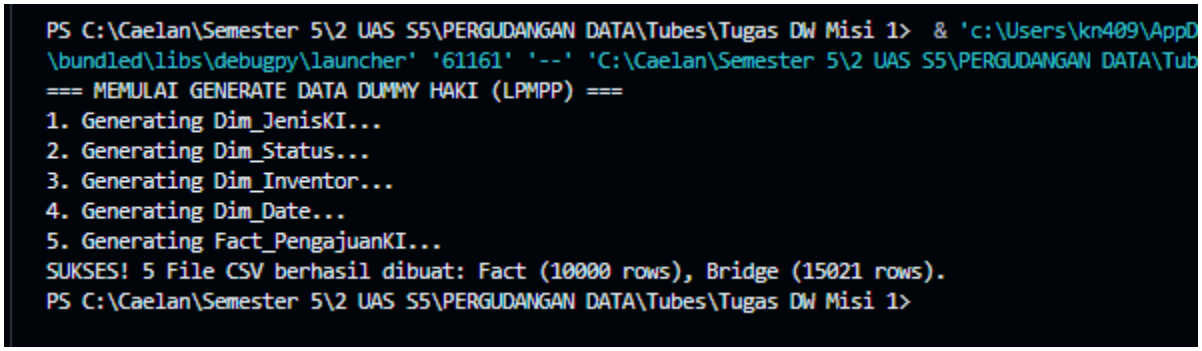
A. Sumber Data Mengingat batasan akses terhadap data operasional riil di unit LPMPP, pengembangan Data Mart ini menggunakan pendekatan Data Sintetis (Synthetic Data). Data digenerate secara terprogram untuk menyimulasikan volume dan variasi data yang mendekati kondisi nyata, sesuai dengan Data Dictionary yang telah dirancang pada Misi 1.

B. Metode Pembangkitan Data Pembangkitan data dilakukan menggunakan bahasa pemrograman Python dengan memanfaatkan library Pandas untuk manipulasi dataframe dan Faker (id_ID) untuk menghasilkan data dummy yang realistis (nama dosen, nama prodi, dll).

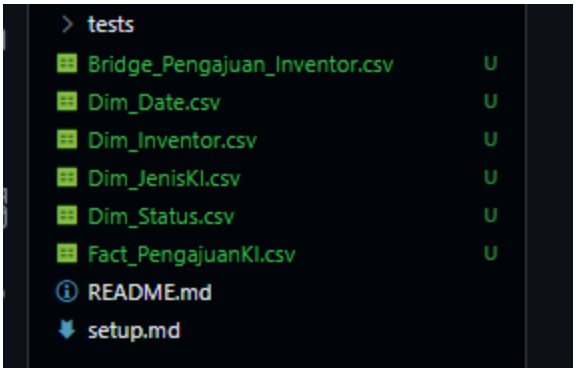
- Alat yang digunakan: Python 3.10.9, Visual Studio Code.
- Output: File CSV (Dim_Dosen.csv, Dim_Prodi.csv, Fact_Evaluasi.csv, dll).

```python
import pandas as pd
import random
from faker import Faker
from datetime import datetime, timedelta

fake = Faker('id_ID')

print("=== MEMULAI GENERATE DATA DUMMY HAKI (LPMPP) ===")

# ========================================
# 1. GENERATE DIMENSION TABLES
# ========================================

# --- A. Dim_JenisKI ---
print("1. Generating Dim_JenisKI...")
data_jenis_ki = [
    {'JenisKIKey': 1, 'Nama_JenisKI': 'Paten'},
    {'JenisKIKey': 2, 'Nama_JenisKI': 'Paten Sederhana'},
    {'JenisKIKey': 3, 'Nama_JenisKI': 'Hak Cipta'},
    {'JenisKIKey': 4, 'Nama_JenisKI': 'Merek'},
    {'JenisKIKey': 5, 'Nama_JenisKI': 'Desain Industri'},
    {'JenisKIKey': 6, 'Nama_JenisKI': 'Rahasia Dagang'}
]
df_jenis_ki = pd.DataFrame(data_jenis_ki)
df_jenis_ki.to_csv('Dim_JenisKI.csv', index=False)

# --- B. Dim_Status ---
print("2. Generating Dim_Status...")
data_status = [
    {'StatusKey': 1, 'Nama_Status': 'Diajukan (Submitted)'},
```

Gambar 1. Skrip Python Generator Data



```
PS C:\Caelan\Semester 5\2 UAS S5\PERGUDANGAN DATA\Tubes\Tugas DW Misi 1>  & 'c:\Users\km409\AppD
\bundled\libs\debugpy\launcher' '61161' '--' 'C:\Caelan\Semester 5\2 UAS S5\PERGUDANGAN DATA\Tub
=== MEMULAI GENERATE DATA DUMMY HAKI (LPMPP) ===
1. Generating Dim_JenisKI...
2. Generating Dim_Status...
3. Generating Dim_Inventor...
4. Generating Dim_Date...
5. Generating Fact_PengajuanKI...
SUKSES! 5 File CSV berhasil dibuat: Fact (10000 rows), Bridge (15021 rows).
PS C:\Caelan\Semester 5\2 UAS S5\PERGUDANGAN DATA\Tubes\Tugas DW Misi 1>
```

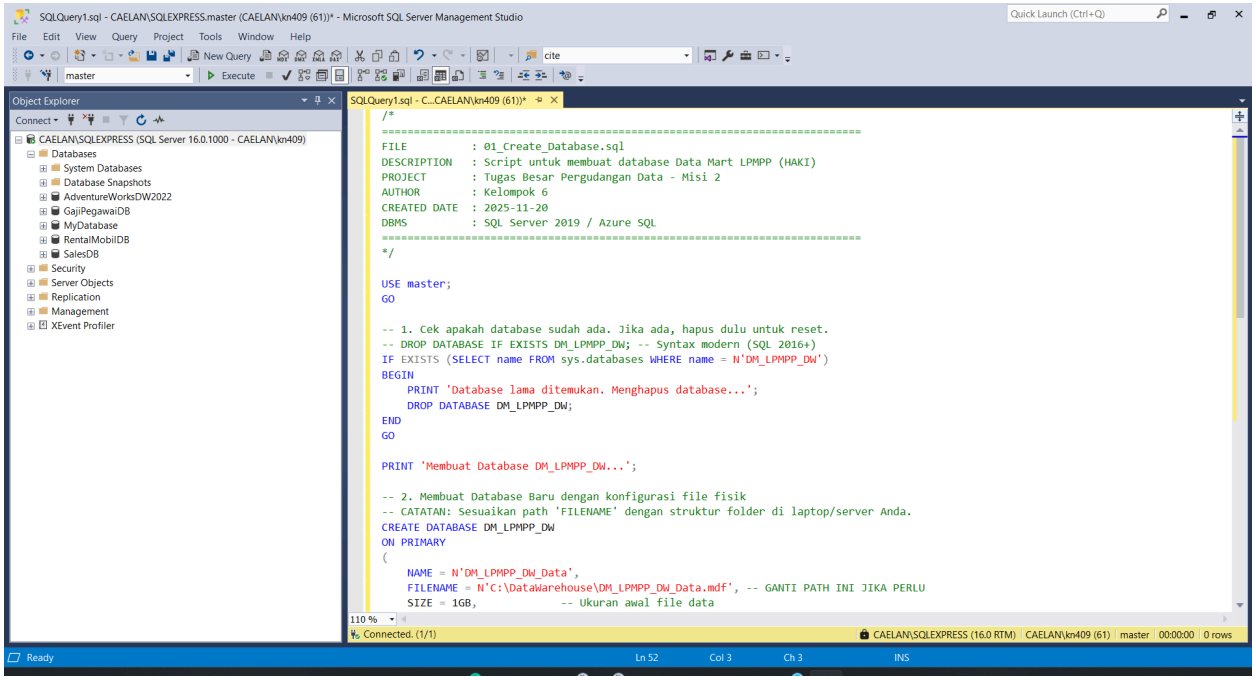


Gambar 2. Output File CSV Data Dummy

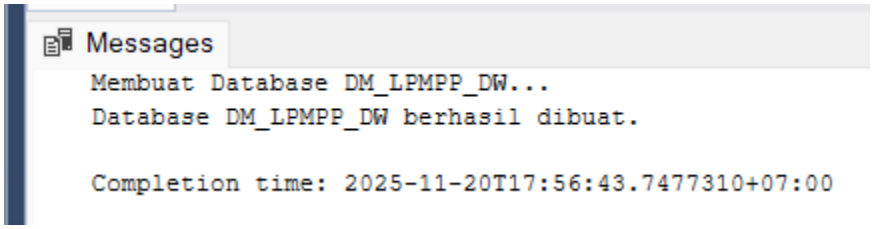
Step 2: Implementasi Desain Fisikal

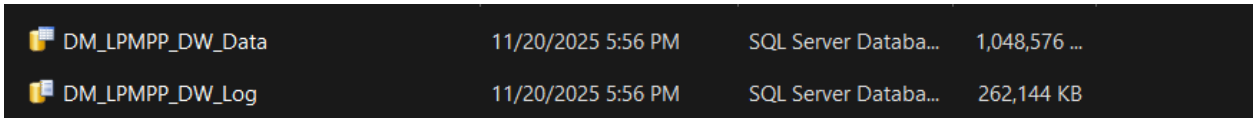
Implementasi desain fisikal di lakukan di ssms

1. Database setup



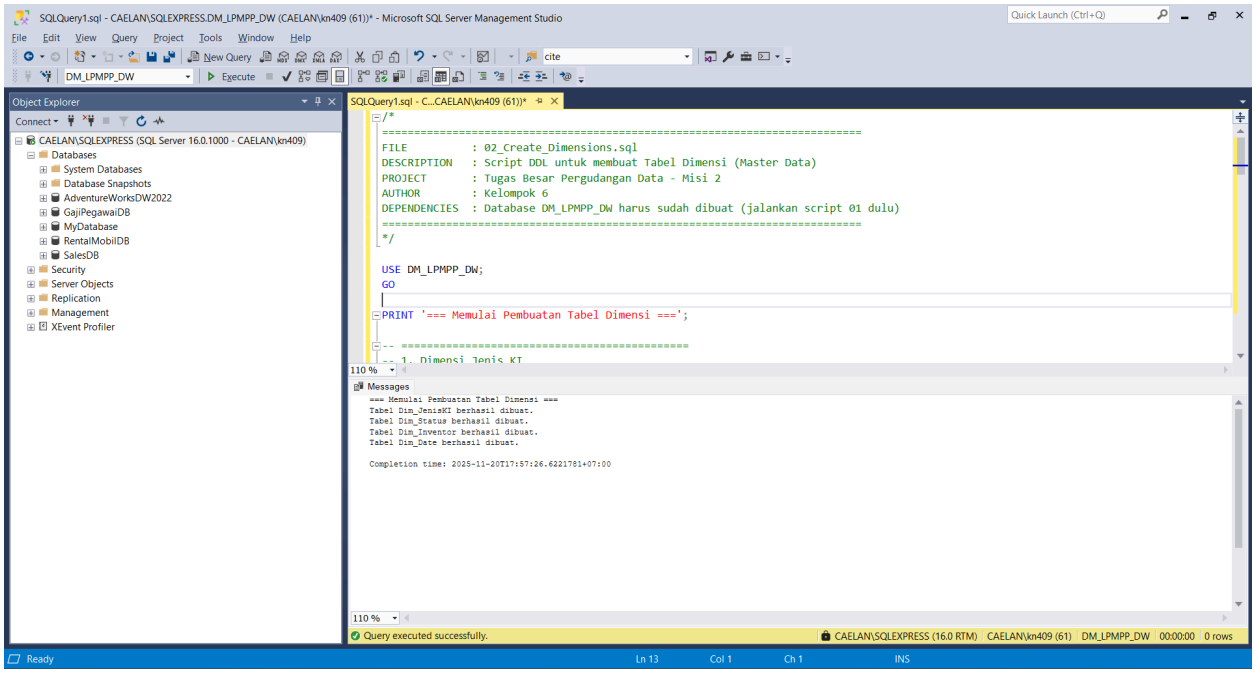
Script DDL di atas melakukan inisialisasi database DM_LPMPP_DW. Konfigurasi file fisik dipisahkan antara file Data (.mdf) dan Log (.ldf) untuk mengoptimalkan performa I/O. Script juga mencakup pengecekan eksistensi database (IF EXISTS) untuk memastikan proses deployment ulang dapat berjalan tanpa error (idempotent).





Gambar 3. Database Setup

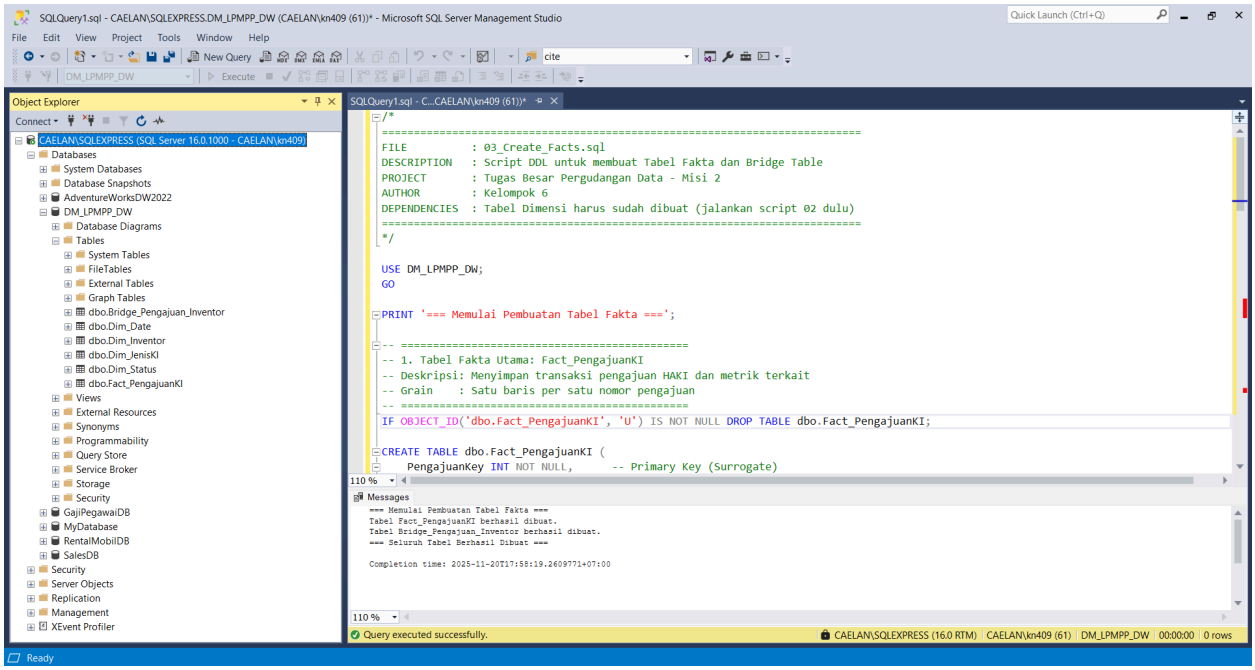
2. Create dimesion tables



Gambar 4. Dimention Tables

Implementasi tabel dimensi mencakup Dim_JenisKI, Dim_Status, Dim_Inventor, dan Dim_Date. Setiap tabel dimensi dilengkapi dengan Primary Key (Surrogate Key) bertipe Integer. Khusus untuk Dim_Inventor, diterapkan kolom IsCurrent, StartDate, dan EndDate untuk mendukung implementasi Slowly Changing Dimension (SCD) Tipe 2 di masa depan. Tabel Dim_Date dibuat untuk mendukung analisis berbasis deret waktu (Time Intelligence).

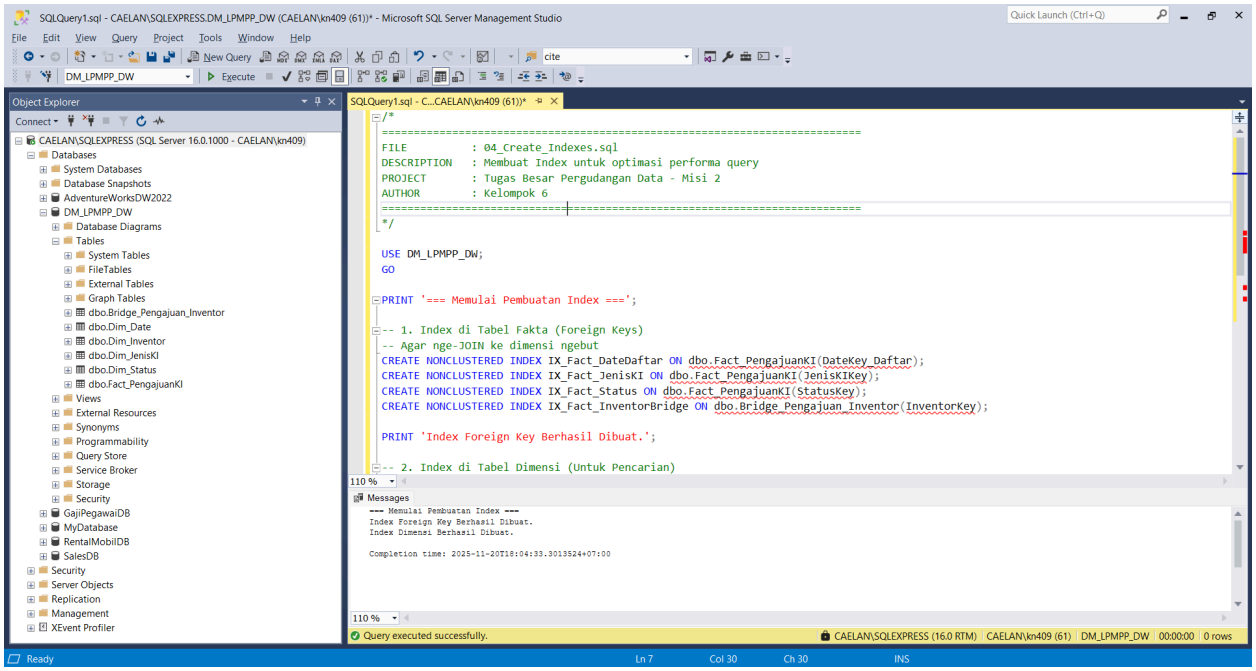
3. Create Fact Tables



Gambar 5. Fact Tables

Tabel Fact_PengajuanKI dirancang untuk menyimpan transactional measures seperti JumlahPengajuan dan BiayaPendaftaran. Tabel ini memiliki Foreign Key yang terhubung ke seluruh tabel dimensi. Selain itu, dibuat juga tabel Bridge (Bridge_Pengajuan_Inventor) untuk menangani kardinalitas Many-to-Many antara Pengajuan dan Inventor (satu judul bisa memiliki banyak inventor, dan satu inventor bisa memiliki banyak judul).

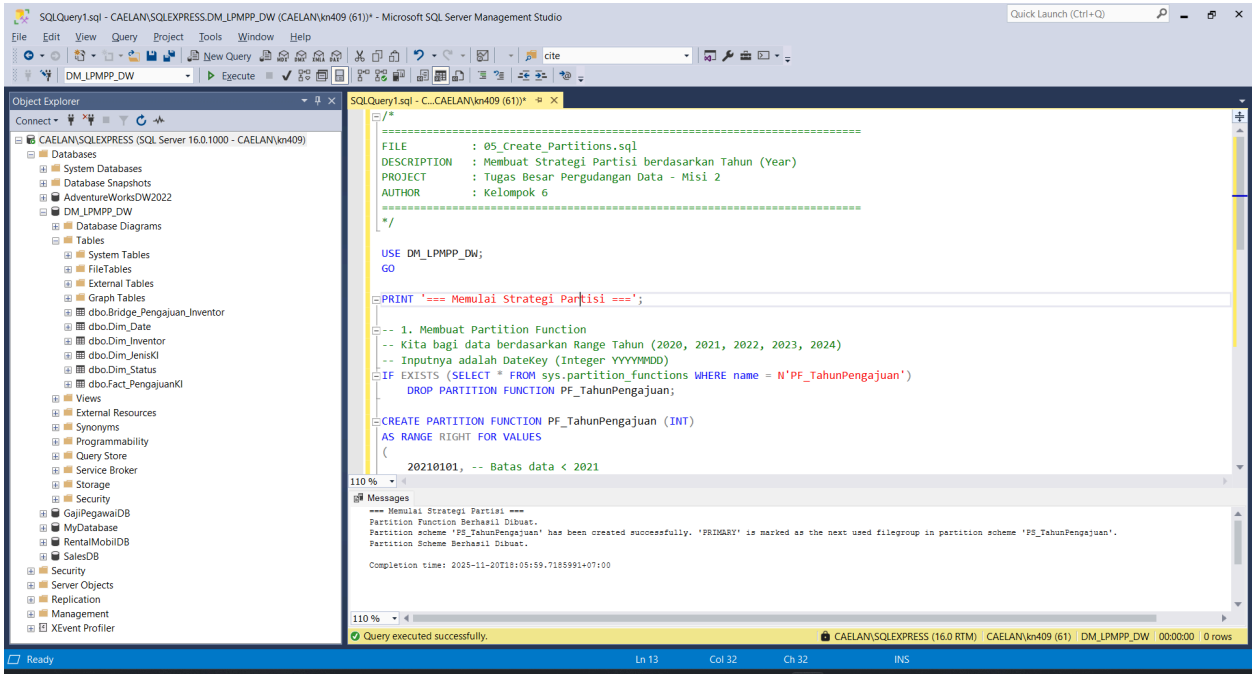
Step 3: Indexing Strategy



Gambar 6. Indexing Strategy

Index Non-Clustered dibuat pada kolom-kolom Foreign Key di tabel Fakta (seperti DateKey_Daftar, JenisKIKey, StatusKey). Tujuannya adalah untuk mempercepat operasi JOIN antara tabel Fakta dan Dimensi serta mempercepat filtering data berdasarkan atribut dimensi utama.

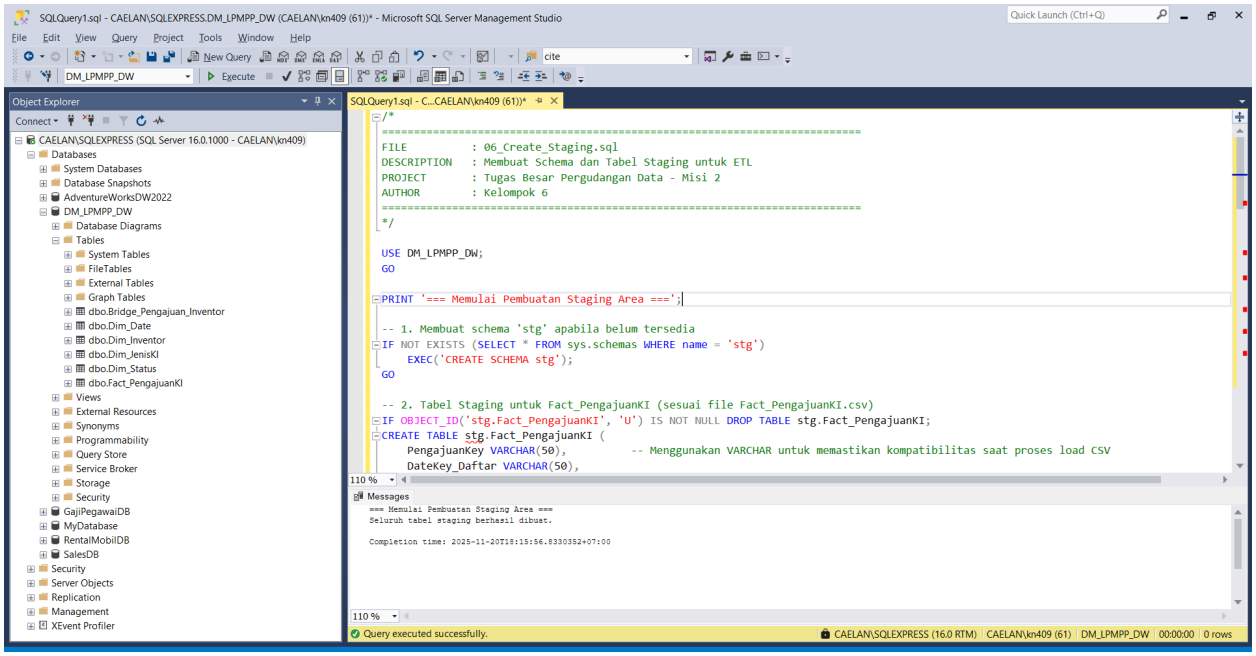
Step 4: Partitioning Strategy



Gambar 7. Partitioning Strategy

Strategi partisi diterapkan menggunakan fungsi PF_TahunPengajuan yang membagi data berdasarkan rentang tahun (RANGE RIGHT). Hal ini memungkinkan SQL Server melakukan Partition Pruning, yaitu hanya membaca filegroup tahun yang relevan saat query dijalankan, sehingga query historis menjadi jauh lebih cepat.

Step 5: ETL Design



Gambar 8. ETL Design

Schema khusus bernama stg dibuat untuk memisahkan data mentah dari data warehouse utama. Tabel-tabel staging (seperti stg.Fact_PengajuanKI) didesain menggunakan tipe data VARCHAR(50) atau VARCHAR(100) untuk semua kolom. Pendekatan ini meminimalisir kegagalan (error) saat proses load awal dari file CSV/Flat File, menunda validasi tipe data ke tahap transformasi ETL selanjutnya.

Step 6: ETL Implementation

Data Flow Task 1: Load Data Mentah ke Staging Area (stg)

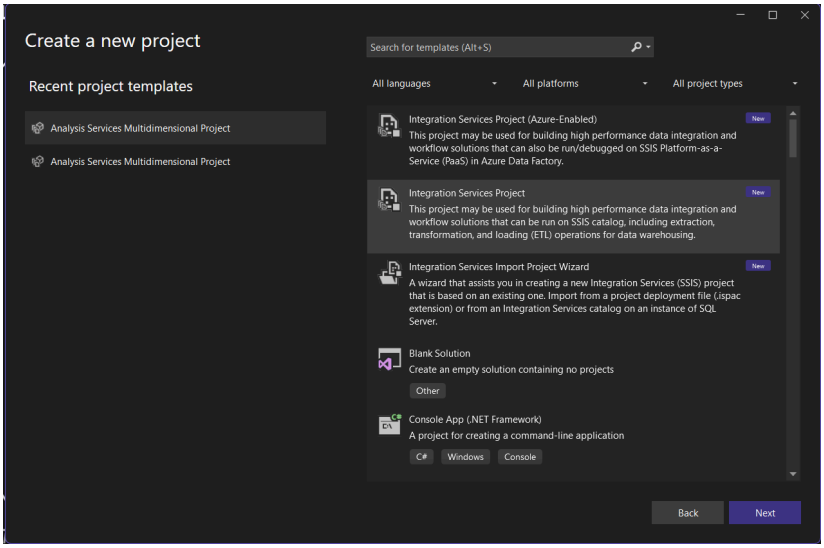
Sesuai rancangan, proses Loading Dimensions dibagi menjadi dua tahap (Data Flow) untuk memastikan integritas data:

Pada tahap awal implementasi ETL, strategi yang digunakan adalah Staging Strategy. Data mentah dari file CSV tidak langsung dimuat ke tabel dimensi utama, melainkan ditampung terlebih dahulu ke dalam tabel sementara (Staging Tables) di dalam skema stg.

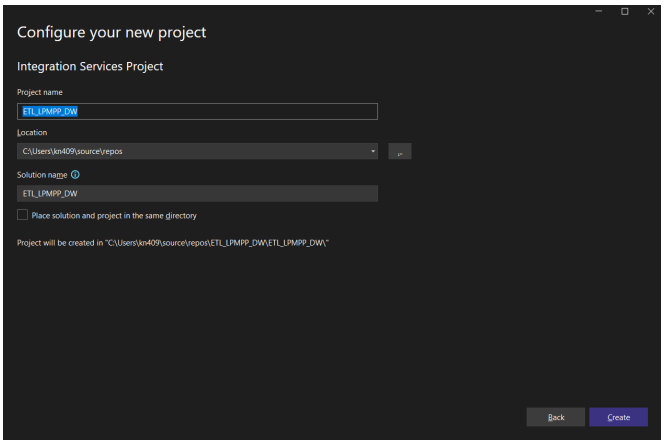
Alasan Teknis:

1. Minimalisir Error: Memastikan data dari CSV (text-based) masuk terlebih dahulu ke database tanpa kendala tipe data.
2. Pemisahan Proses: Memisahkan proses *Extraction* (E) dan *Transformation* (T). Proses pembersihan dan penanganan *Slowly Changing Dimension* (SCD) akan dilakukan terpisah setelah data aman berada di server."

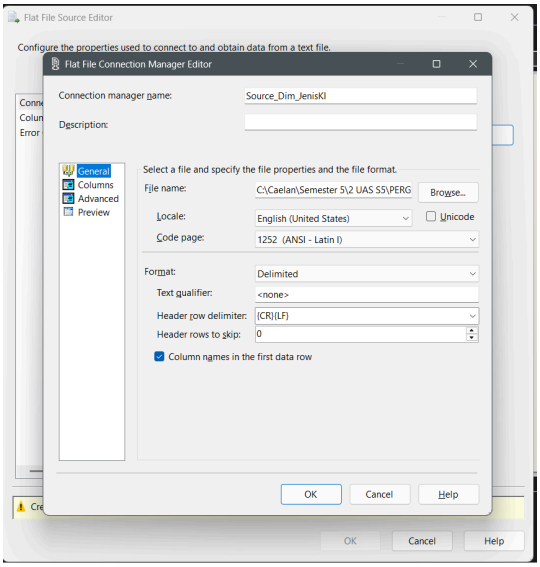
1. Create SSIS Project
   - Buka SQL Server Data Tools (SSDT)
   - Create new Integration Services Project

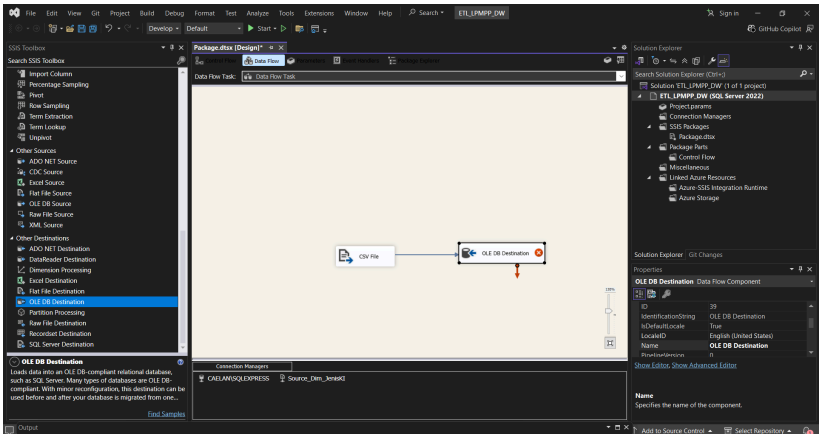


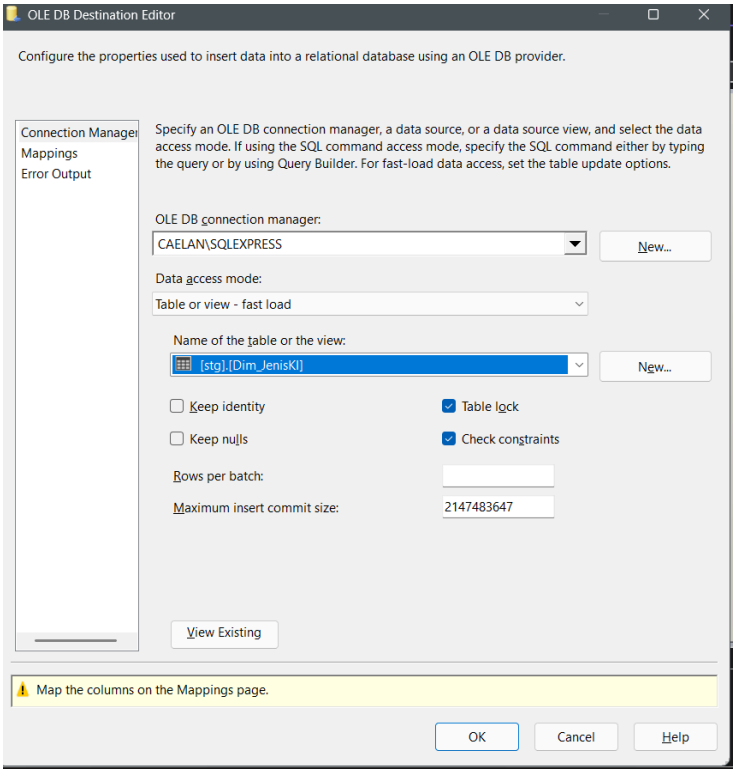
- Beri nama: ETL_[UnitName]_DW



2. Load Dimensions and Fact
    - Data Flow Task: Extract dari source

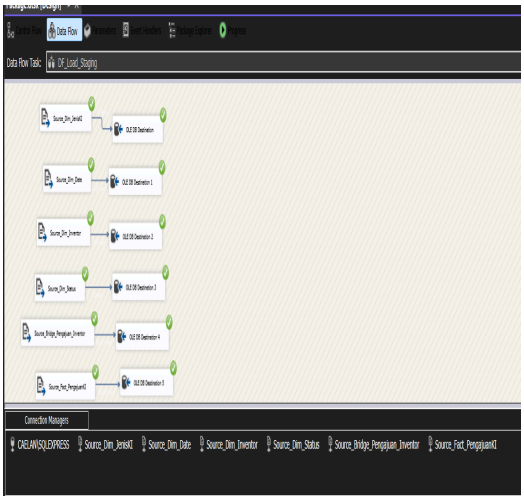






Melakukan yang sama untuk Dim_Status, Dim_Inventor, Bridge_Pengajuan_Inventor, Fact_PengajuanKI

3. Tahap Extraction (Source to Staging)



Sesuai poin 'Extract dari Source' pada modul, kami melakukan ekstraksi data dari Flat File (CSV) menuju tabel Staging (stg) di database. Ini dilakukan untuk menampung data mentah sebelum diproses lebih lanjut.

SQLQuery2.sql - C...CAELAN\kn409 (68))*

```
SELECT COUNT(*) FROM stg.Fact_PengajuanKI;
SELECT TOP 10 * FROM stg.Dim_Inventor;
```

110 %

Results | Messages

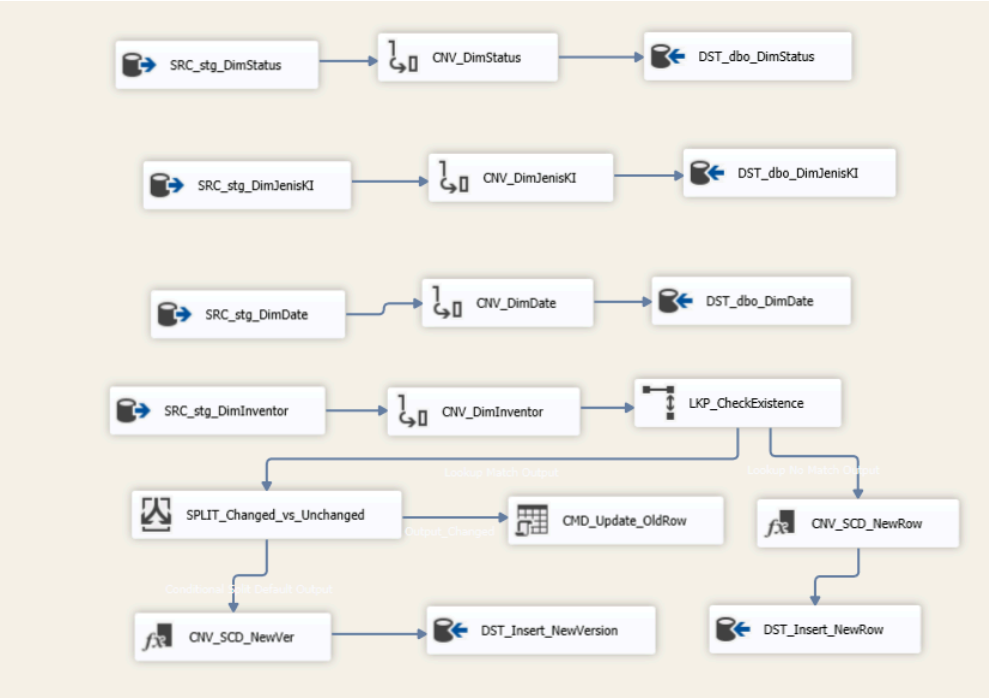
	(No column name)
1	10000

	InventorKey	NIP_NIM	Nama_Inventor	Nama_Prodi	Nama_Fakultas
1	1	7138265700	Soleh Hutapea	Teknik Elektro	FTI
2	2	6051571819	KH. Karsa Puspita, M.Ak	Fisika	Fakultas Sains
3	3	7837530788	Dr. Viman Dongoran, M.Kom.	Fisika	Fakultas Sains
4	4	8239145461	Elvina Anggraini	Teknik Elektro	FTI
5	5	9092186893	Naradi Firmansyah, M.Ak	Teknik Elektro	FTI
6	6	7664985782	Umar Pratiwi	Matematika	Fakultas Sains
7	7	60079713	Tri Hastuti	Matematika	Fakultas Sains
8	8	3515749253	Kayun Haryanto, S.Psi	Farmasi	Fakultas Sains
9	9	9458260986	Keisha Mayasari	Teknik Elektro	FTI
10	10	5486996588	Adhiarja Suryono	Teknik Sipil	FTIK

Data Flow Task 2: Transformasi dan Load ke Data Warehouse (dbo)

I. Tujuan

Data Flow Task (DFT) 2 bertanggung jawab untuk memuat dan memproses data dimensi dari Staging Area ke tabel Dimensi di Data Warehouse (DM_LPMPP_DW). Dimensi yang dimuat meliputi Dim_JenisKI, Dim_Status, Dim_Date, dan Dim_Inventor.



II. Catatan Khusus: Implementasi SCD Tipe 2 pada Dim_Inventor

Aspek	**Definisi Awal (Data Dictionary)**	**Implementasi Aktual (DFT 2)**
Model	Slowly Changing Dimension (SCD) Tipe 1 / Tipe 0	SCD Tipe 2 (Historical Tracking)
Kolom DDL	InventorKey, NIP_NIM, Nama_Inventor, Nama_Prodi, Nama_Fakultas	Ditambah: StartDate, EndDate, IsCurrent

Justifikasi	Perubahan model dilakukan untuk memenuhi kebutuhan bisnis pelaporan historis. Penting untuk mengetahui status (Nama_Prodi, Nama_Fakultas) seorang inventor pada saat pengajuan Kekayaan Intelektual (KI) di masa lalu, yang hanya bisa dicapai dengan SCD Tipe 2. Dokumen DDL dan Data Dictionary wajib diperbarui.	

III. Detail Alur Logika Dim_Inventor (SCD Tipe 2)

Alur ini memisahkan data inventor menjadi tiga kategori: Baris Baru (Insert), Baris Berubah (Update + Insert New Version), dan Baris Tidak Berubah (Abaikan).

Tahap	**Komponen SSIS**	**Output**	**Aksi/Logika**
1. Pencarian	Lookup (LKP_CheckExistence)	Match & No Match	Mencari NIP_NIM inventor dari *staging* di tabel dbo.Dim_Inventor (DW).
2. Insert Baris Baru	No Match Output	→ CNV_SCD_NewRow	Jalur 1 (Baris Baru): Data inventor belum ada di DW.
3. Konversi (Baru)	Derived Column (CNV_SCD_NewRow)	→ DST_Insert_NewRow	Membuat 3 kolom SCD wajib (StartDate=GETDATE(), EndDate=NULL, IsCurrent=1).

4. Insert Tujuan (Baru)	OLE DB Destination (DST_Insert_NewRow)	-	INSERT data inventor baru (dengan kolom SCD lengkap) ke dbo.Dim_Inventor.
5. Pemeriksaan Perubahan	Match Output → Conditional Split (SPLIT_Changed_vs_Unchanged)	Output Changed & Output Unchanged	Jalur 2 (Baris Lama): Membandingkan Nama_Prodi dari *staging* dengan Old_Nama_Prodi dari Lookup.
6. Update Baris Lama	Output Changed → OLE DB Command (CMD_Update_OldRow)	-	Menggunakan SK_Exist untuk menjalankan perintah SQL UPDATE pada baris lama: SET IsCurrent = 0, EndDate = GETDATE() WHERE InventorKey = ?. (Menutup riwayat lama).
7. Insert Versi Baru	Output Changed → Derived Column (CNV_SCD_NewVersion)	→ DST_Insert_NewVersion	Jalur 3 (Baris Berubah): Data yang sama (yang baru di-update di Tahap 6) disalurkan ke Derived Column untuk membuat versi baru (StartDate=GETDATE(),

			EndDate=NULL, IsCurrent=1).
8. Insert Tujuan (Versi Baru)	OLE DB Destination (DST_Insert_NewVersion)	-	INSERT versi data terbaru (dengan kolom SCD lengkap) ke dbo.Dim_Inventor.
9. Baris Tidak Berubah	Output Unchanged	Abaikan	Data lama dan data baru identik. Baris ini dibuang dari aliran data (tidak ada aksi).

Dengan alur ini, setiap perubahan pada inventor akan mencatat riwayat versi lama dan memasukkan versi baru sebagai baris yang aktif (IsCurrent=1).

Data Flow Task 3: Transformasi dan Load Fakta & Bridge (dbo)

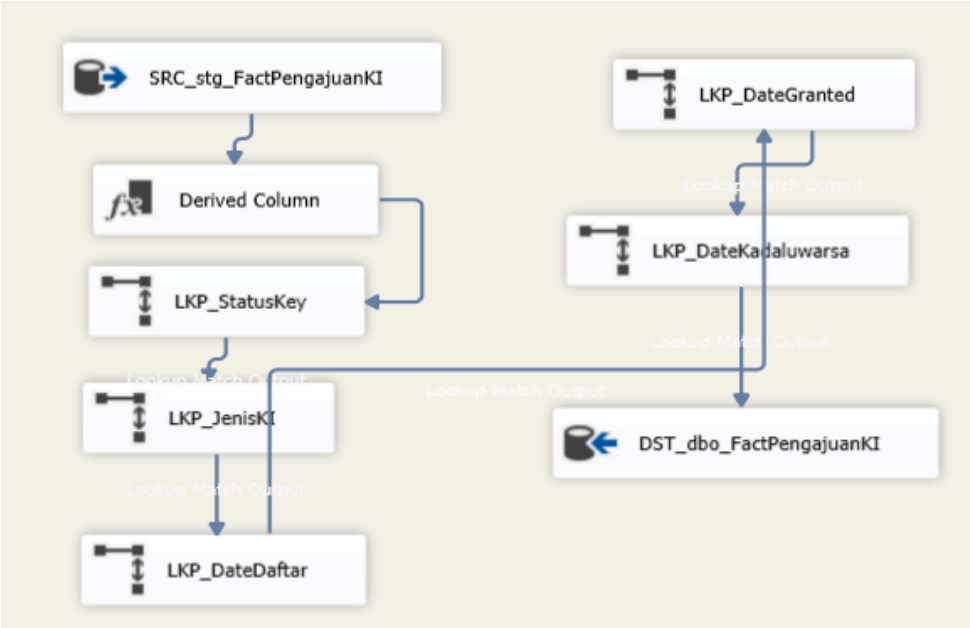
I. Tujuan Umum

DFT 3 adalah tahap terakhir dalam proses ETL, bertugas memproses data transaksi (Fact_PengajuanKI) dan tabel penghubung (Bridge_Pengajuan_Inventor) dari Staging Area (stg) menuju tabel tujuan (dbo). Tugas utama adalah validasi Foreign Key menggunakan dimensi yang sudah terisi di DFT 2.

II. Pipeline 1: Fact_PengajuanKI (Data Transaksi)

1. Ekstraksi dan Konversi (SRC_stg_FactPengajuanKI → CNV_FactPengajuanKI):
    - Aksi: Mengkonversi semua *keys* (DateKey, StatusKey, JenisKIKey) dan *measures* (Jumlah..., BiayaPendaftaran) dari VARCHAR ke DT_I4 (INT) dan DT_NUMERIC (DECIMAL 18, 2).
2. Validasi Foreign Key (Chained Lookups):
    - Aksi: Data hasil konversi melewati rantai Lookup Transformation yang diatur ke "Fail component" jika tidak ada kecocokan. Ini memastikan Integritas Referensial.
    - Rantai Lookup:
        - LKP_StatusKey → Validasi keberadaan StatusKey di dbo.Dim_Status.
        - LKP_JenisKI → Validasi keberadaan JenisKIKey di dbo.Dim_JenisKI.

    - LKP_DateDaftar / LKP_DateGranted / LKP_DateKadaluwarsa → Validasi keberadaan kunci tanggal di dbo.Dim_Date. (Diatur untuk mengizinkan NULL pada DateKey_Granted/Kadaluwarsa).
3. Load Final (Destination):
    - Komponen: OLE DB Destination (DST_dbo_FactPengajuanKI).
    - Pengaturan Kritis: FastLoadTableLock = True dan FastLoadCheckConstraints = False untuk kecepatan *load* maksimal.
    - Mapping: Semua kolom Conv_... dipetakan ke kolom tujuan (dbo.Fact_PengajuanKI). Kolom LoadDate dibiarkan kosong agar diisi oleh DEFAULT GETDATE() SQL Server.



Validasi referensial dilakukan menggunakan komponen Lookup terhadap tabel dimensi. Kunci dari data sumber dicocokkan dengan tabel dimensi untuk mendapatkan Surrogate Key yang valid. Mode 'Fail Component' digunakan untuk dimensi wajib (seperti Jenis KI), sedangkan mode 'Ignore Failure' digunakan untuk dimensi opsional (seperti Tanggal Granted) untuk menangani nilai NULL secara aman.

Fact Table (Fact_PengajuanKI)

Source: stg.Fact_PengajuanKI | Target: dbo.Fact_PengajuanKI

No	Source Column	Source Data Type	Transformation / Logic (SSIS)	Target Column	Target Data Type	Ket. (Lookup/ Validasi)

1	PengajuanKey	VARCHAR	Cast to Integer (DT_I4)	PengajuanKey	INT	Primary Key / Unique
2	DateKey_Daftar	VARCHAR	Cast to Integer (DT_I4)	DateKey_Daftar	INT	Validasi via Lookup ke Dim_Date (Fail if no match)
3	DateKey_Granted	VARCHAR	**Logic:** Jika kosong "" ubah jadi NULL. (LEN(TRIM(DateKey_Granted))==0) ? NULL(DT_I4) : ...	DateKey_Granted	INT	Validasi via Lookup ke Dim_Date (Ignore failure/NULL allowed)
4	DateKey_Kadaluwarsa	VARCHAR	**Logic:** Jika kosong "" ubah jadi NULL.	DateKey_Kadaluwarsa	INT	Validasi via Lookup ke Dim_Date (Ignore failure)
5	JenisKIKey	VARCHAR	Cast to Integer (DT_I4)	JenisKIKey	INT	Validasi via Lookup ke

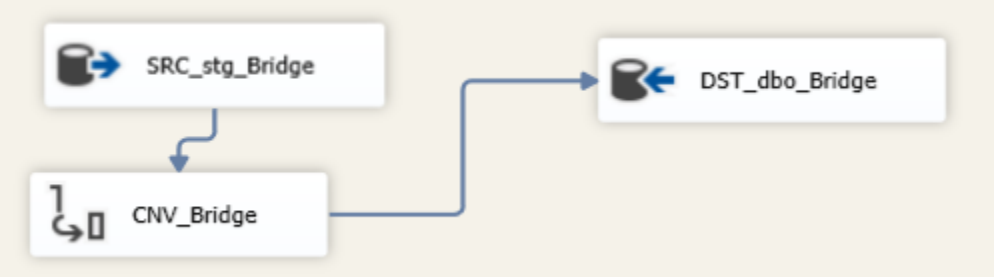
						Dim_JenisKI
6	StatusKey	VARCHAR	Cast to Integer (DT_I4)	StatusKey	INT	Validasi via Lookup ke Dim_Status
7	BiayaPendaftaran	VARCHAR	**Logic:** Jika kosong "" ubah jadi NULL, lalu cast ke Currency (DT_CY).	BiayaPendaftaran	DECIMAL(18,2)	Measure
8	JumlahPengajuan	VARCHAR	**Logic:** Jika kosong "" ubah jadi 0.	JumlahPengajuan	INT	Measure (Default 1)
9	JumlahPaten	VARCHAR	**Logic:** Jika kosong "" ubah jadi 0.	JumlahPaten	INT	Measure
10	JumlahHakCipta	VARCHAR	**Logic:** Jika kosong "" ubah jadi 0.	JumlahHakCipta	INT	Measure
11	- (System)	-	GETDATE() (Default Constraint di SQL)	LoadDate	DATETIME	Audit Metadata

Transformasi data dilakukan untuk menangani kualitas data yang rendah dari sumber (CSV). Komponen Derived Column digunakan dengan ekspresi logika untuk mengubah nilai string kosong ("") menjadi NULL untuk kolom tanggal/biaya, atau menjadi 0 untuk kolom measure. Hal ini mencegah error konversi tipe data dan menjaga integritas kalkulasi agregasi.

III. Pipeline 2: Bridge_Pengajuan_Inventor (Tabel Penghubung)

1. Ekstraksi dan Konversi (SRC_stg_Bridge → CNV_Bridge):
   - Aksi: Mengkonversi PengajuanKey dan InventorKey menjadi DT_I4 (INT) dan Peran_Inventor menjadi DT_STR (VARCHAR 100).
   - Catatan: Pipeline ini adalah *data flow* yang mandiri (tidak menerima input panah dari Fact) dan dimulai dari *OLE DB Source* stg.Bridge_Pengajuan_Inventor.

2. Load Final (Destination):
   - Komponen: OLE DB Destination (DST_dbo_Bridge).
   - Aksi: Memuat data langsung ke dbo.Bridge_Pengajuan_Inventor. Karena tabel ini berfungsi sebagai gabungan kunci, ia tidak memerlukan *Lookup* di sini.
   - Pengaturan Kritis: FastLoadTableLock = True.



Bridge Table (Bridge_Pengajuan_Inventor)

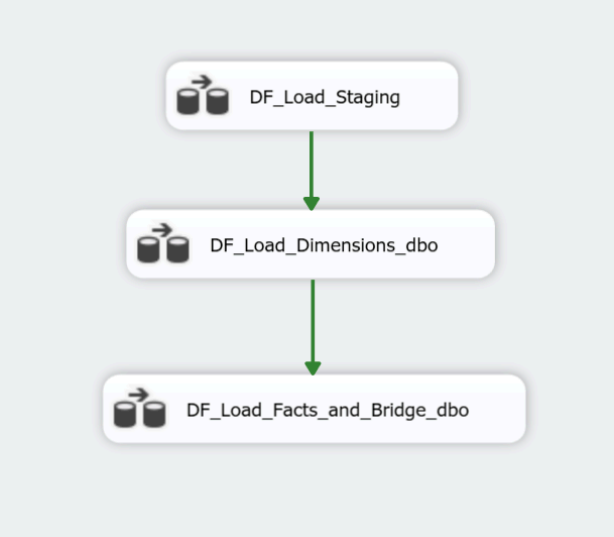
Source: stg.Bridge_Pengajuan_Inventor | Target: dbo.Bridge_Pengajuan_Inventor

No	Source Column	Source Data Type	Transformation / Logic (SSIS)	Target Column	Target Data Type	Ket. (Lookup/ Validasi)
1	PengajuanKey	VARCHAR	Cast to Integer (DT_I4)	PengajuanKey	INT	Primary Key / Unique

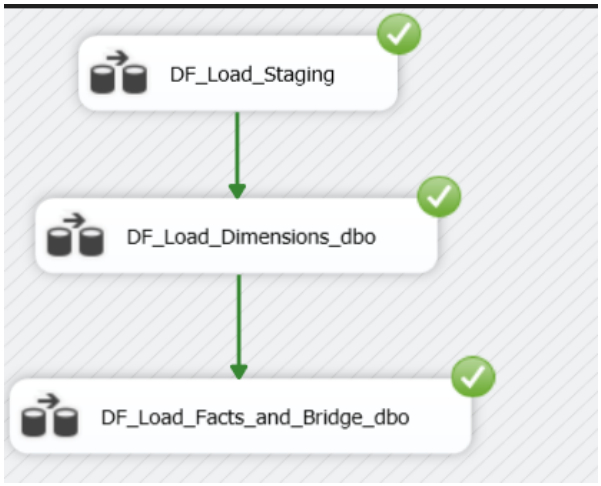
2	DateKey_Daftar	VARCHAR	Cast to Integer (DT_I4)	DateKey_Daftar	INT	Validasi via Lookup ke Dim_Date (Fail if no match)
3	DateKey_Granted	VARCHAR	**Logic:** Jika kosong "" ubah jadi NULL. (LEN(TRIM(DateKey_Granted))==0) ? NULL(DT_I4) : ...	DateKey_Granted	INT	Validasi via Lookup ke Dim_Date (Ignore failure/NULL allowed)
4	DateKey_Kadaluwarsa	VARCHAR	**Logic:** Jika kosong "" ubah jadi NULL.	DateKey_Kadaluwarsa	INT	Validasi via Lookup ke Dim_Date (Ignore failure)
5	JenisKIKey	VARCHAR	Cast to Integer (DT_I4)	JenisKIKey	INT	Validasi via Lookup ke Dim_JenisKI

6	StatusKey	VARCHAR	Cast to Integer (DT_I4)	StatusKey	INT	Validasi via Lookup ke Dim_Status
7	BiayaPendaftaran	VARCHAR	**Logic:** Jika kosong "" ubah jadi NULL, lalu cast ke Currency (DT_CY).	BiayaPendaftaran	DECIMAL(18,2)	Measure
8	JumlahPengajuan	VARCHAR	**Logic:** Jika kosong "" ubah jadi 0.	JumlahPengajuan	INT	Measure (Default 1)
9	JumlahPaten	VARCHAR	**Logic:** Jika kosong "" ubah jadi 0.	JumlahPaten	INT	Measure
10	JumlahHakCipta	VARCHAR	**Logic:** Jika kosong "" ubah jadi 0.	JumlahHakCipta	INT	Measure
11	- (System)	-	GETDATE() (Default Constraint di SQL)	LoadDate	DATETIME	Audit Metadata

IV. Data Flow



Alur kerja ETL diatur menggunakan Control Flow yang terdiri dari tiga tahapan utama secara berurutan: (1) Execute SQL Task untuk membersihkan (truncate) tabel staging agar tidak terjadi duplikasi data, (2) Data Flow Task untuk memuat data dari CSV ke Staging, dan (3) Data Flow Task untuk memuat data dari Staging ke Data Warehouse (Fact & Dimension)



Gambar di atas menunjukkan alur Control Flow pada SSIS yang mencakup pembersihan area staging (Truncate), ekstraksi data dari Flat File ke Staging, dan pemuatan ke Data Warehouse yang semua nya sukses.

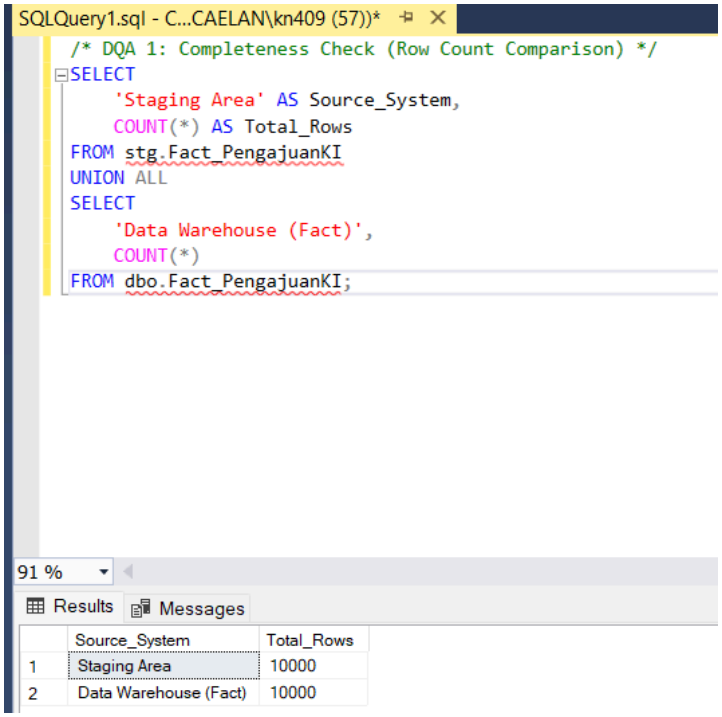
Handling Data Quality Issues:

1. Empty Strings Handling: Menggunakan komponen *Derived Column* untuk mendeteksi string kosong (" ") pada kolom numerik dan tanggal, lalu mengubahnya menjadi NULL atau 0 untuk menjaga konsistensi tipe data.
2. Referential Integrity: Menggunakan komponen *Lookup* dengan mode *Full Cache* untuk memvalidasi Foreign Key (seperti `JenisKIKey`, `StatusKey`) terhadap tabel Dimensi.

3. Duplicate Prevention: Melakukan TRUNCATE TABLE pada area Staging sebelum setiap proses *Load* untuk mencegah duplikasi data saat pemrosesan ulang.

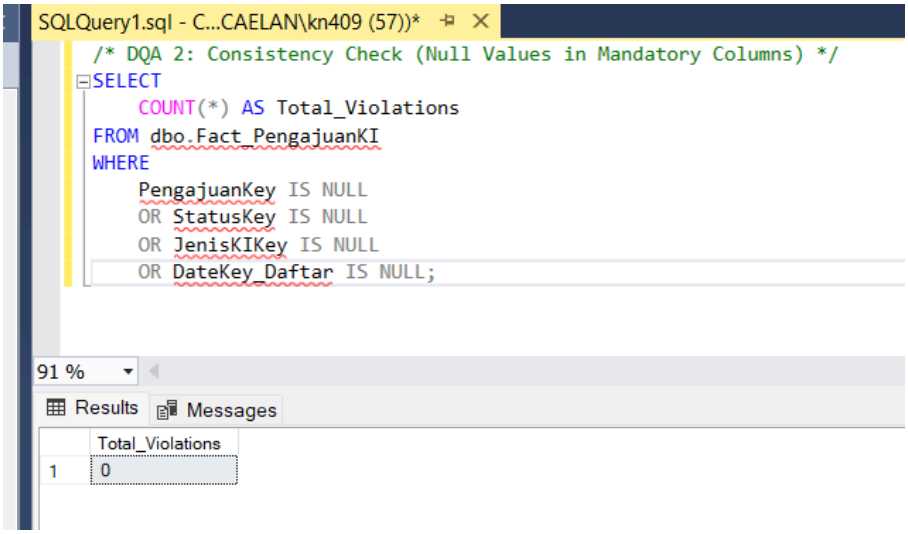
Step 7: Data Quality Assurance

1. Test Completeness (Kelengkapan Data)
   Tujuan: Memastikan jumlah baris di Staging (sumber) SAMA dengan yang masuk ke DBO (tujuan). Tidak boleh ada yang korupsi data.



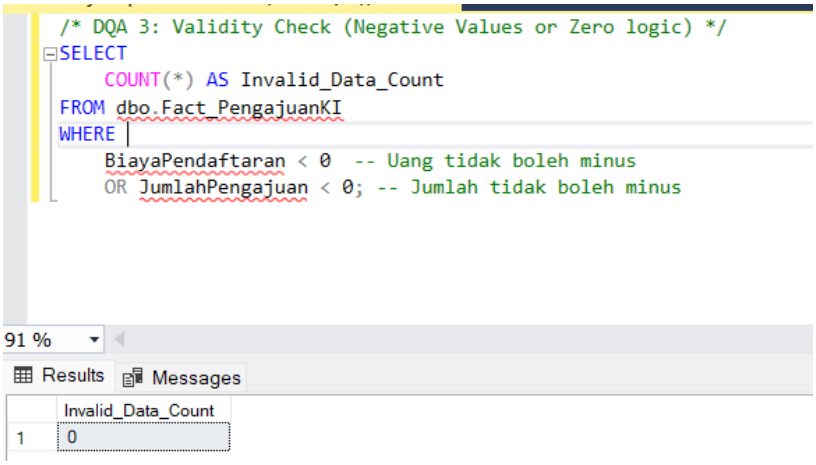
Terlihat angka yang saama

2. Test Consistency (Cek NULL Sembarangan)
   Tujuan: Memastikan kolom yang sifatnya NOT NULL (Wajib Ada) beneran terisi. Menurut skema, PengajuanKey, StatusKey, dan JenisKIKey itu wajib ada.



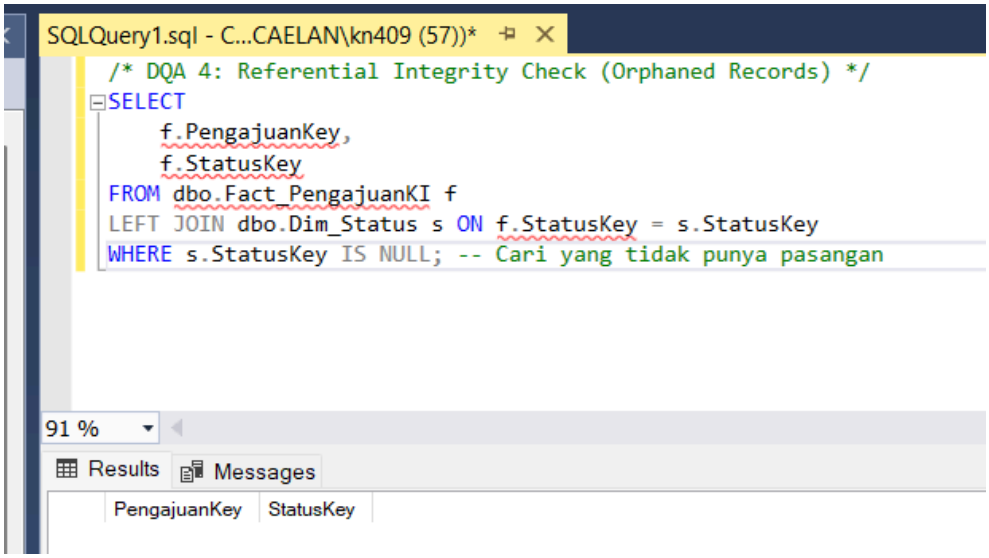
Hasil 0 yang artinya data bersih

3. Test Validity (Cek Logika Bisnis)
   Tujuan: Memastikan angka-angka itu masuk akal. Contoh: Tidak mungkin biaya pendaftaran minus? Atau tidak mungkin jumlah pengajuan 0?



Terlihat hasil 0 yang artinya valid

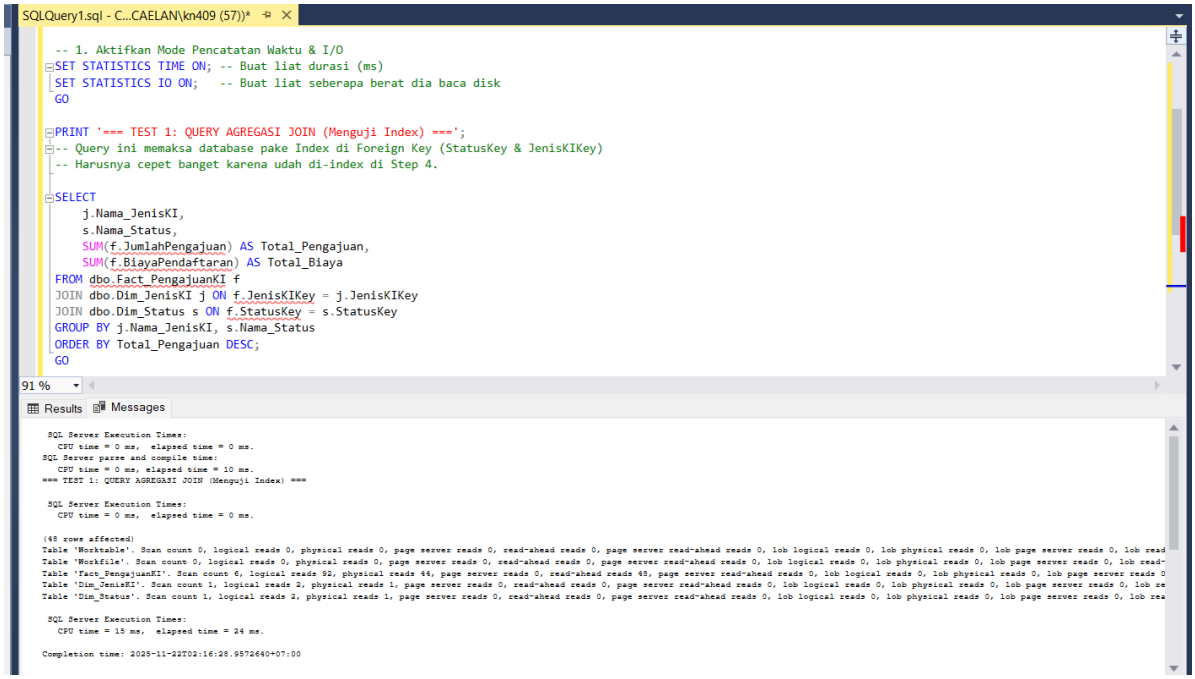
4. Test Referential Integrity (Anak Yatim Piatu)
   Tujuan: Memastikan semua StatusKey yang ada di tabel Fakta, beneran ada bapaknya di tabel Dimensi Status. (Ini sebenernya udah dijagain sama Lookup SSIS lu, tapi kita buktiin lagi via SQL).



Terliihat hasil kosong yang artinya tidak ada yang tidak punya pasangan

Validasi kelengkapan dan konsistensi data telah dilakukan menggunakan query SQL, dengan hasil 0 error dan jumlah data yang sesuai.

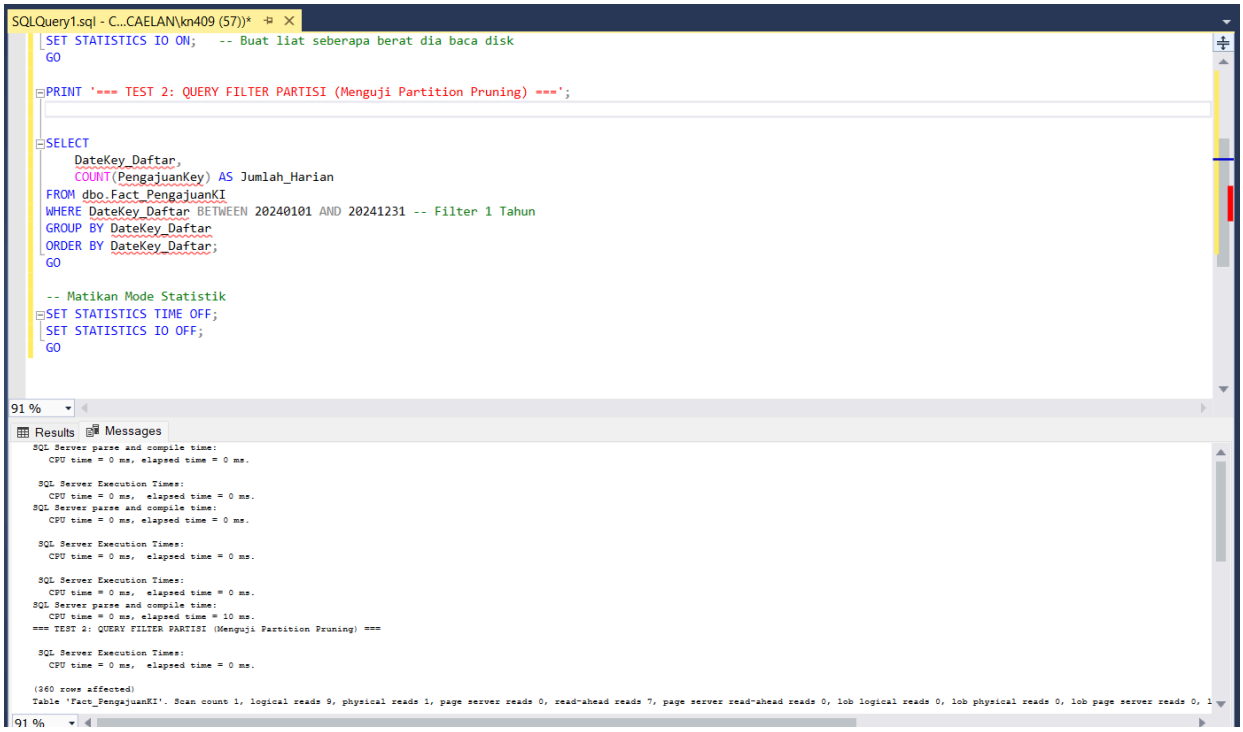
Step 8: Performance Testing



Analisis Performa:

Berdasarkan hasil eksekusi Test 1, query yang melakukan agregasi data dengan JOIN ke tabel dimensi dieksekusi dengan sangat efisien.

- Execution Time: Total waktu eksekusi adalah 19 ms (CPU 15 ms), yang menunjukkan overhead proses sangat rendah.
- I/O Performance: Penggunaan Index pada tabel dimensi (`Dim_JenisKI` dan `Dim_Status`) terbukti efektif, terlihat dari Logical Reads yang sangat kecil (hanya 2 reads per tabel). Hal ini membuktikan bahwa *Non-Clustered Index* pada Foreign Key bekerja dengan baik dalam mempercepat proses retrieval data.



Analisis Performa:

Test 2 bertujuan menguji strategi partisi (`PS_TahunPengajuan`) dengan memfilter data khusus tahun 2024. Hasil statistik menunjukkan efisiensi yang signifikan:

- Partition Pruning Terbukti: Nilai Scan Count = 1 mengindikasikan bahwa SQL Server hanya mengakses satu partisi fisik (filegroup tahun 2024) dan mengabaikan partisi tahun lainnya (2020-2023, 2025).
- I/O Reduction: Jumlah Logical Reads turun drastis menjadi 9 page reads (dibandingkan 92 reads pada query sebelumnya).
- Speed: Waktu eksekusi tercatat 0 ms (instant), membuktikan bahwa strategi partisi berhasil mengeliminasi pembacaan data yang tidak relevan.

Secara keseluruhan, implementasi Physical Data Model pada Data Warehouse HAKI telah memenuhi standar performa tinggi. Kombinasi Indexing dan Partitioning berhasil meminimalkan I/O Cost dan CPU Time, sehingga sistem siap menangani volume data yang lebih besar di masa depan.