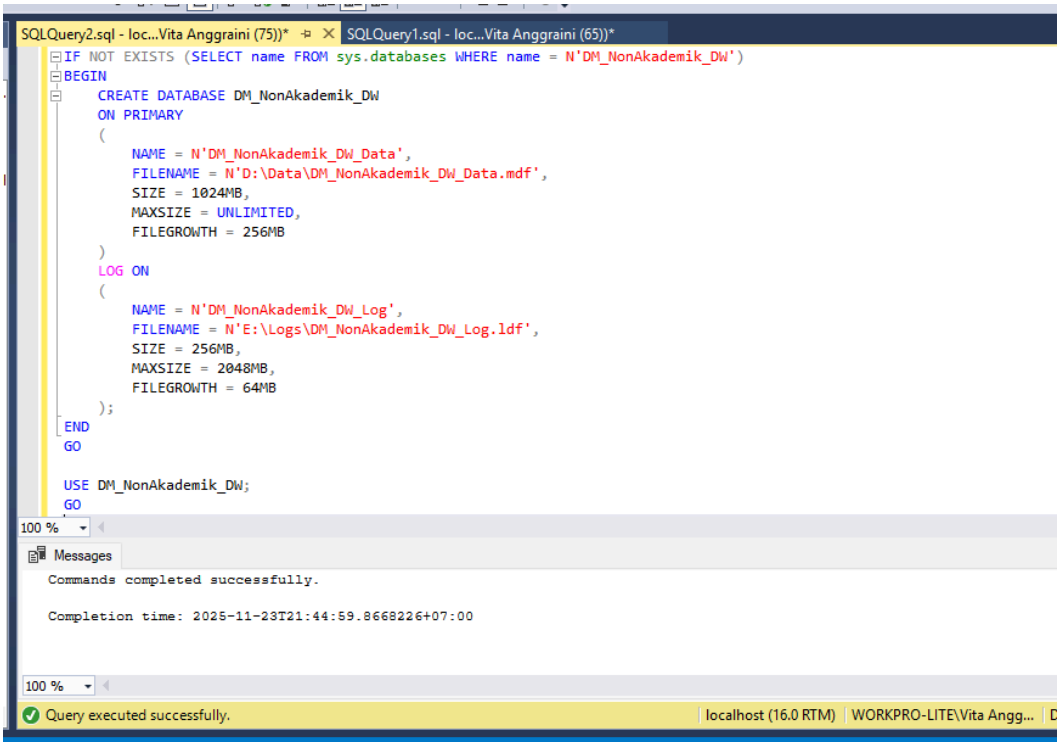
Kelompok 3
Non Akademik

Nama Anggota Kelompok (Terurut Berdasarkan NIM):
Vita Anggraini (122450046)
Kharisma Mustika Sari (123450034)
Ridho Benedictus Togi Manik (123450060)
Arielva Simon Siahaan (123450105)

1. **Step 1: Physical Database Design**
   a) **Database setup**

```
-- ================================================
-- 01_Create_Database.sql
-- Create Data Warehouse database for Non-Akademik
-- ================================================
IF NOT EXISTS (SELECT name FROM sys.databases WHERE name =
N'DM_NonAkademik_DW')
BEGIN
   CREATE DATABASE DM_NonAkademik_DW
   ON PRIMARY
   (
      NAME = N'DM_NonAkademik_DW_Data',
      FILENAME = N'D:\Data\DM_NonAkademik_DW_Data.mdf',
      SIZE = 1024MB,
      MAXSIZE = UNLIMITED,
      FILEGROWTH = 256MB
   )
   LOG ON
   (
      NAME = N'DM_NonAkademik_DW_Log',
      FILENAME = N'E:\Logs\DM_NonAkademik_DW_Log.ldf',
      SIZE = 256MB,
      MAXSIZE = 2048MB,
      FILEGROWTH = 64MB
   );
END
GO

USE DM_NonAkademik_DW;
GO
```



b) Create Dimension Tables

```
/*********************************************
  FULL: Create Dimensions + Facts (safe for partitioning)
  - Dimensions: PK as identity (clustered default)
  - Facts: identity keys but NO PRIMARY KEY (so no automatic clustered PK)
  - All FK constraints created to link facts -> dims
*********************************************/

USE DM_NonAkademik_DW;
GO

-- ============================
-- DIMENSIONS
-- ============================

-- DimTanggal
IF OBJECT_ID('dbo.DimTanggal') IS NULL
CREATE TABLE dbo.DimTanggal (
    DateKey INT NOT NULL PRIMARY KEY, -- format YYYYMMDD
    FullDate DATE NOT NULL,
    DayNumberOfWeek TINYINT NOT NULL,
    DayName VARCHAR(20) NOT NULL,
    DayNumberOfMonth TINYINT NOT NULL,
    DayNumberOfYear SMALLINT NOT NULL,
```

```
    WeekNumberOfYear TINYINT NOT NULL,
    MonthName VARCHAR(20) NOT NULL,
    MonthNumber TINYINT NOT NULL,
    Quarter TINYINT NOT NULL,
    QuarterName VARCHAR(10) NOT NULL,
    YearNumber SMALLINT NOT NULL,
    IsWeekend BIT NOT NULL,
    IsHoliday BIT NOT NULL,
    HolidayName VARCHAR(100) NULL,
    AcademicYear VARCHAR(9) NULL,
    Semester TINYINT NULL,
    CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimUnitKerja
IF OBJECT_ID('dbo.DimUnitKerja') IS NULL
CREATE TABLE dbo.DimUnitKerja (
    UnitKey INT IDENTITY(1,1) PRIMARY KEY,
    UnitCode VARCHAR(50) NULL,
    UnitName VARCHAR(250) NOT NULL,
    UnitType VARCHAR(100) NULL,
    ParentUnitCode VARCHAR(50) NULL,
    EffectiveDate DATE NOT NULL DEFAULT GETDATE(),
    ExpiryDate DATE NULL,
    IsCurrent BIT NOT NULL DEFAULT 1,
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimVendor
IF OBJECT_ID('dbo.DimVendor') IS NULL
CREATE TABLE dbo.DimVendor (
    VendorKey INT IDENTITY(1,1) PRIMARY KEY,
    VendorCode VARCHAR(100) NULL,
    VendorName VARCHAR(250) NOT NULL,
    Category VARCHAR(100) NULL,
    City VARCHAR(100) NULL,
    ContactPerson VARCHAR(150) NULL,
    ContactPhone VARCHAR(50) NULL,
    Email VARCHAR(150) NULL,
    CreatedDate DATETIME DEFAULT GETDATE()
);
GO
```

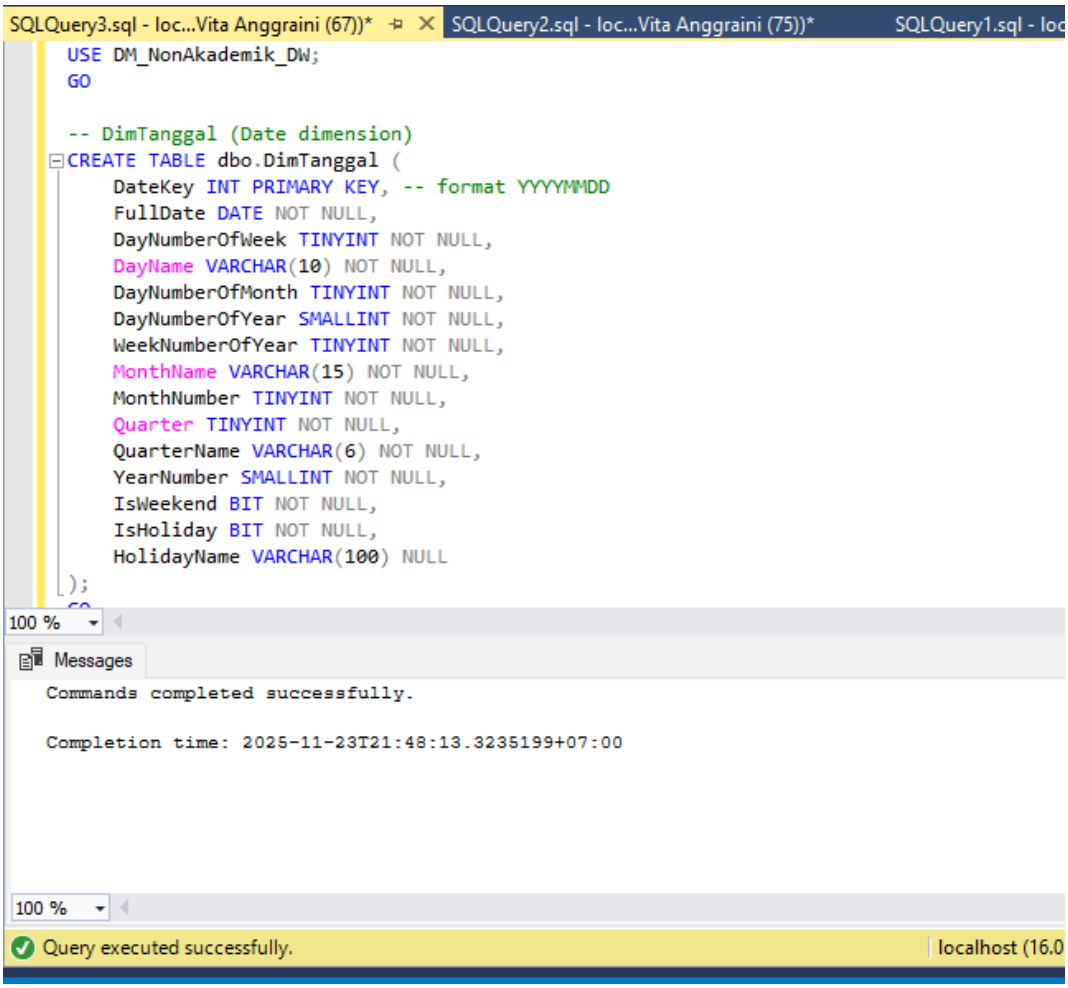
```
-- DimAsalSurat
IF OBJECT_ID('dbo.DimAsalSurat') IS NULL
CREATE TABLE dbo.DimAsalSurat (
    AsalKey INT IDENTITY(1,1) PRIMARY KEY,
    AsalName VARCHAR(250) NOT NULL,
    AsalType VARCHAR(100) NULL,
    Kota VARCHAR(100) NULL,
    CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimTujuanSurat
IF OBJECT_ID('dbo.DimTujuanSurat') IS NULL
CREATE TABLE dbo.DimTujuanSurat (
    TujuanKey INT IDENTITY(1,1) PRIMARY KEY,
    TujuanName VARCHAR(250) NOT NULL,
    TujuanType VARCHAR(100) NULL,
    Kota VARCHAR(100) NULL,
    CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimAkunBelanja
IF OBJECT_ID('dbo.DimAkunBelanja') IS NULL
CREATE TABLE dbo.DimAkunBelanja (
    AkunKey INT IDENTITY(1,1) PRIMARY KEY,
    KodeAkun VARCHAR(50) NOT NULL,
    NamaAkun VARCHAR(250) NOT NULL,
    KelompokAkun VARCHAR(100) NULL,
    KategoriBelanja VARCHAR(100) NULL,
    CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimFasilitas
IF OBJECT_ID('dbo.DimFasilitas') IS NULL
CREATE TABLE dbo.DimFasilitas (
    FasilitasKey INT IDENTITY(1,1) PRIMARY KEY,
    FasilitasCode VARCHAR(100) NULL,
    NamaFasilitas VARCHAR(250) NOT NULL,
    JenisFasilitas VARCHAR(100) NULL,
    LokasiCode VARCHAR(100) NULL,
    Kapasitas INT NULL,
    UnitPengelolaCode VARCHAR(50) NULL,
    CreatedDate DATETIME DEFAULT GETDATE()
);
```

```
GO

-- DimKategoriPengadaan
IF OBJECT_ID('dbo.DimKategoriPengadaan') IS NULL
CREATE TABLE dbo.DimKategoriPengadaan (
   KategoriKey INT IDENTITY(1,1) PRIMARY KEY,
   KategoriName VARCHAR(200) NOT NULL,
   SubKategori VARCHAR(200) NULL,
   CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimLokasi
IF OBJECT_ID('dbo.DimLokasi') IS NULL
CREATE TABLE dbo.DimLokasi (
   LokasiKey INT IDENTITY(1,1) PRIMARY KEY,
   LokasiCode VARCHAR(100) NULL,
   LokasiName VARCHAR(250) NOT NULL,
   Alamat VARCHAR(300) NULL,
   Kota VARCHAR(100) NULL,
   Provinsi VARCHAR(100) NULL,
   CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimKeperluan
IF OBJECT_ID('dbo.DimKeperluan') IS NULL
CREATE TABLE dbo.DimKeperluan (
   KeperluanKey INT IDENTITY(1,1) PRIMARY KEY,
   KeperluanDesc VARCHAR(250) NOT NULL,
   CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimPerihalSurat
IF OBJECT_ID('dbo.DimPerihalSurat') IS NULL
CREATE TABLE dbo.DimPerihalSurat (
   PerihalKey INT IDENTITY(1,1) PRIMARY KEY,
   PerihalDesc VARCHAR(250) NOT NULL,
   CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimStatus (single status table for multiple processes)
IF OBJECT_ID('dbo.DimStatus') IS NULL
CREATE TABLE dbo.DimStatus (
```

```
    StatusKey INT IDENTITY(1,1) PRIMARY KEY,
    ProcessType VARCHAR(50) NOT NULL, -- SURAT / PENGADAAN /
PEMINJAMAN / LAYANAN / BMN
    StatusName VARCHAR(100) NOT NULL,
    Description VARCHAR(250) NULL,
    CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- DimBMN
IF OBJECT_ID('dbo.DimBMN') IS NULL
CREATE TABLE dbo.DimBMN (
    BMNKey INT IDENTITY(1,1) PRIMARY KEY,
    KodeBarang VARCHAR(100) NOT NULL,
    NamaBarang VARCHAR(250) NOT NULL,
    KategoriBarang VARCHAR(150) NULL,
    JenisBarang VARCHAR(150) NULL,
    TanggalPerolehan DATE NULL,
    SumberPerolehan VARCHAR(100) NULL,       -- Pengadaan / Hibah
    HargaPerolehan DECIMAL(18,2) NULL,
    KondisiAwal VARCHAR(100) NULL,
    KondisiSaatIni VARCHAR(100) NULL,
    LokasiAwalKey INT NULL,
    LokasiSaatIniKey INT NULL,
    UnitPenggunaKey INT NULL,
    StatusAset VARCHAR(100) NULL,            -- Aktif / Tidak Layak / Dihapus /
Dipindah
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO

-- add FK on DimBMN to lokasi/unit (if dims exist)
IF OBJECT_ID('dbo.DimLokasi') IS NOT NULL AND
OBJECT_ID('dbo.DimUnitKerja') IS NOT NULL
BEGIN
    IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_BMN_LokasiAwal')
        ALTER TABLE dbo.DimBMN
        ADD CONSTRAINT FK_BMN_LokasiAwal FOREIGN KEY
(LokasiAwalKey) REFERENCES dbo.DimLokasi(LokasiKey);

    IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_BMN_LokasiSaatIni')
        ALTER TABLE dbo.DimBMN
        ADD CONSTRAINT FK_BMN_LokasiSaatIni FOREIGN KEY
```

```
(LokasiSaatIniKey) REFERENCES dbo.DimLokasi(LokasiKey);

   IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_BMN_UnitPengguna')
       ALTER TABLE dbo.DimBMN
       ADD CONSTRAINT FK_BMN_UnitPengguna FOREIGN KEY
(UnitPenggunaKey) REFERENCES dbo.DimUnitKerja(UnitKey);
END
GO

-- DimJenisLayanan (for Layanan Administrasi)
IF OBJECT_ID('dbo.DimJenisLayanan') IS NULL
CREATE TABLE dbo.DimJenisLayanan (
   JenisLayananKey INT IDENTITY(1,1) PRIMARY KEY,
   JenisLayananName VARCHAR(150) NOT NULL,
   Description VARCHAR(250) NULL,
   CreatedDate DATETIME DEFAULT GETDATE()
);
GO

-- optional helpful indexes on dims (SQL Server SAFE)

-- DimUnitKerja
IF NOT EXISTS (
   SELECT 1 FROM sys.indexes
   WHERE name = 'IX_DimUnitKerja_UnitCode'
     AND object_id = OBJECT_ID('dbo.DimUnitKerja')
)
CREATE INDEX IX_DimUnitKerja_UnitCode
ON dbo.DimUnitKerja(UnitCode);
GO

-- DimAkunBelanja
IF NOT EXISTS (
   SELECT 1 FROM sys.indexes
   WHERE name = 'IX_DimAkunBelanja_KodeAkun'
     AND object_id = OBJECT_ID('dbo.DimAkunBelanja')
)
CREATE INDEX IX_DimAkunBelanja_KodeAkun
ON dbo.DimAkunBelanja(KodeAkun);
GO

-- DimFasilitas
IF NOT EXISTS (
   SELECT 1 FROM sys.indexes
   WHERE name = 'IX_DimFasilitas_Nama'
```

```
    AND object_id = OBJECT_ID('dbo.DimFasilitas')
)
CREATE INDEX IX_DimFasilitas_Nama
ON dbo.DimFasilitas(NamaFasilitas);
GO
```



c) Create Fact Tables

```
-- ============================
-- FACTS (safe: no PK constraints so clustered index can be created later)
-- ============================

-- Fact_RealisasiAnggaran
IF OBJECT_ID('dbo.Fact_RealisasiAnggaran') IS NULL
CREATE TABLE dbo.Fact_RealisasiAnggaran (
   RealisasiKey BIGINT IDENTITY(1,1) NOT NULL,
   DateKey INT NOT NULL,
   UnitKey INT NOT NULL,
```

```
    AkunKey INT NOT NULL,
    NilaiRealisasi DECIMAL(18,2) NOT NULL,
    SelisihAnggaranRealisasi DECIMAL(18,2) NULL,
    Keterangan VARCHAR(500) NULL,
    SourceSystem VARCHAR(100) NULL,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-- FK Realisasi -> dims
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_FactRealisasi_DimTanggal')
ALTER TABLE dbo.Fact_RealisasiAnggaran
ADD CONSTRAINT FK_FactRealisasi_DimTanggal FOREIGN KEY
(DateKey) REFERENCES dbo.DimTanggal(DateKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_FactRealisasi_DimUnit')
ALTER TABLE dbo.Fact_RealisasiAnggaran
ADD CONSTRAINT FK_FactRealisasi_DimUnit FOREIGN KEY (UnitKey)
REFERENCES dbo.DimUnitKerja(UnitKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_FactRealisasi_DimAkun')
ALTER TABLE dbo.Fact_RealisasiAnggaran
ADD CONSTRAINT FK_FactRealisasi_DimAkun FOREIGN KEY
(AkunKey) REFERENCES dbo.DimAkunBelanja(AkunKey);
GO

-- Fact_Pengadaan
IF OBJECT_ID('dbo.Fact_Pengadaan') IS NULL
CREATE TABLE dbo.Fact_Pengadaan (
    PengadaanKey BIGINT IDENTITY(1,1) NOT NULL,
    DateKey INT NOT NULL,
    UnitKey INT NOT NULL,
    VendorKey INT NULL,
    KategoriKey INT NULL,
    NilaiPengadaan DECIMAL(18,2) NOT NULL,
    StatusKey INT NULL,
    NomorPengadaan VARCHAR(100) NULL,
    VendorName VARCHAR(250) NULL,
    BMNKey INT NULL,
    SourceSystem VARCHAR(100) NULL,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

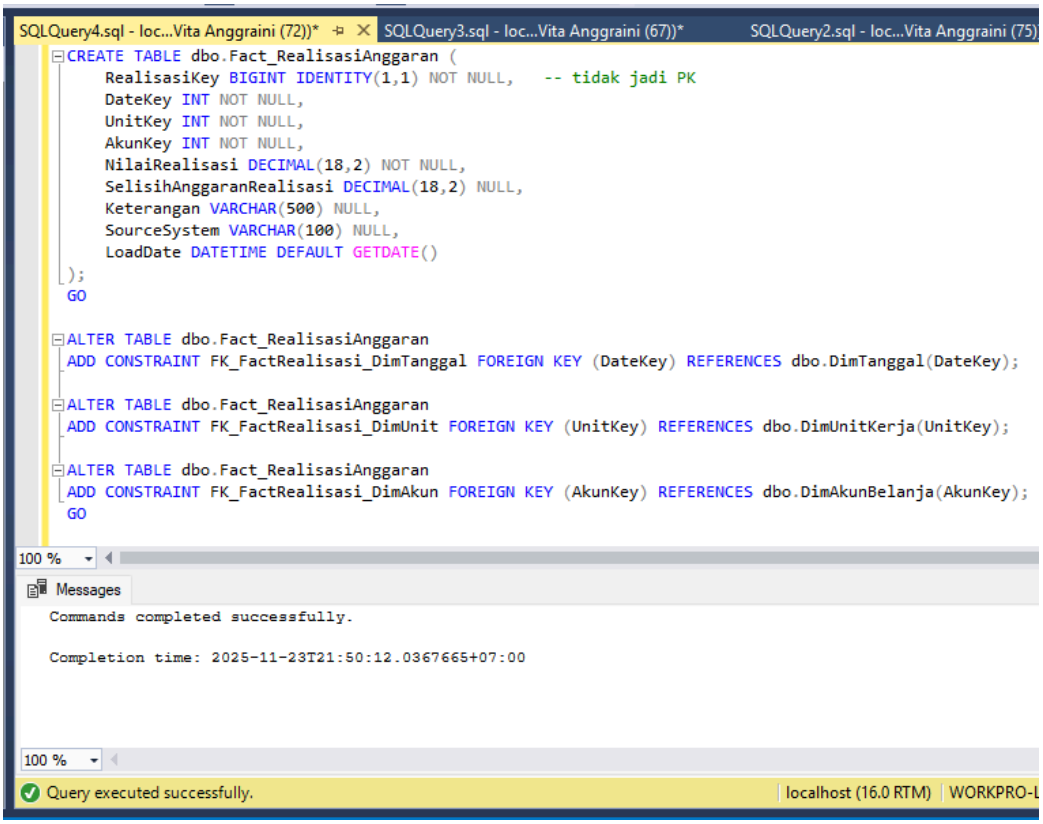
```
-- FK Pengadaan -> dims
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Pengadaan_BMN')
ALTER TABLE dbo.Fact_Pengadaan
ADD CONSTRAINT FK_Pengadaan_BMN FOREIGN KEY (BMNKey)
REFERENCES dbo.DimBMN(BMNKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_FactPengadaan_DimTanggal')
ALTER TABLE dbo.Fact_Pengadaan
ADD CONSTRAINT FK_FactPengadaan_DimTanggal FOREIGN KEY
(DateKey) REFERENCES dbo.DimTanggal(DateKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_FactPengadaan_DimUnit')
ALTER TABLE dbo.Fact_Pengadaan
ADD CONSTRAINT FK_FactPengadaan_DimUnit FOREIGN KEY
(UnitKey) REFERENCES dbo.DimUnitKerja(UnitKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_FactPengadaan_DimVendor')
ALTER TABLE dbo.Fact_Pengadaan
ADD CONSTRAINT FK_FactPengadaan_DimVendor FOREIGN KEY
(VendorKey) REFERENCES dbo.DimVendor(VendorKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_FactPengadaan_DimKategori')
ALTER TABLE dbo.Fact_Pengadaan
ADD CONSTRAINT FK_FactPengadaan_DimKategori FOREIGN KEY
(KategoriKey) REFERENCES dbo.DimKategoriPengadaan(KategoriKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_FactPengadaan_Status')
ALTER TABLE dbo.Fact_Pengadaan
ADD CONSTRAINT FK_FactPengadaan_Status FOREIGN KEY (StatusKey)
REFERENCES dbo.DimStatus(StatusKey);
GO

-- Fact_PeminjamanFasilitas
IF OBJECT_ID('dbo.Fact_PeminjamanFasilitas') IS NULL
CREATE TABLE dbo.Fact_PeminjamanFasilitas (
    PeminjamanKey BIGINT IDENTITY(1,1) NOT NULL,
    DateKey INT NOT NULL,
    FasilitasKey INT NOT NULL,
    UnitKey INT NOT NULL,
```

```
    KeperluanKey INT NULL,
    TanggalPinjam DATE NULL,
    TanggalKembali DATE NULL,
    DurasiHari INT NULL,
    StatusKey INT NULL,
    SourceSystem VARCHAR(100) NULL,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-- FK Peminjaman -> dims
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Peminjaman_DimTanggal')
ALTER TABLE dbo.Fact_PeminjamanFasilitas
ADD CONSTRAINT FK_Peminjaman_DimTanggal FOREIGN KEY
(DateKey) REFERENCES dbo.DimTanggal(DateKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Peminjaman_DimFasilitas')
ALTER TABLE dbo.Fact_PeminjamanFasilitas
ADD CONSTRAINT FK_Peminjaman_DimFasilitas FOREIGN KEY
(FasilitasKey) REFERENCES dbo.DimFasilitas(FasilitasKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Peminjaman_DimUnit')
ALTER TABLE dbo.Fact_PeminjamanFasilitas
ADD CONSTRAINT FK_Peminjaman_DimUnit FOREIGN KEY (UnitKey)
REFERENCES dbo.DimUnitKerja(UnitKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Peminjaman_DimKeperluan')
ALTER TABLE dbo.Fact_PeminjamanFasilitas
ADD CONSTRAINT FK_Peminjaman_DimKeperluan FOREIGN KEY
(KeperluanKey) REFERENCES dbo.DimKeperluan(KeperluanKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Peminjaman_Status')
ALTER TABLE dbo.Fact_PeminjamanFasilitas
ADD CONSTRAINT FK_Peminjaman_Status FOREIGN KEY (StatusKey)
REFERENCES dbo.DimStatus(StatusKey);
GO

-- Fact_SuratMasuk
IF OBJECT_ID('dbo.Fact_SuratMasuk') IS NULL
CREATE TABLE dbo.Fact_SuratMasuk (
    SuratMasukKey BIGINT IDENTITY(1,1) NOT NULL,
```

```
    DateKey INT NOT NULL,
    AsalKey INT NOT NULL,
    PerihalKey INT NOT NULL,
    UnitKey INT NOT NULL,
    StatusKey INT NOT NULL,
    JumlahSurat INT DEFAULT 1,
    WaktuProses INT NULL,
    NomorSurat VARCHAR(200) NULL,
    SourceSystem VARCHAR(100) NULL,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-- FK SuratMasuk -> dims
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratMasuk_DimTanggal')
ALTER TABLE dbo.Fact_SuratMasuk
ADD CONSTRAINT FK_SuratMasuk_DimTanggal FOREIGN KEY
(DateKey) REFERENCES dbo.DimTanggal(DateKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratMasuk_DimAsal')
ALTER TABLE dbo.Fact_SuratMasuk
ADD CONSTRAINT FK_SuratMasuk_DimAsal FOREIGN KEY (AsalKey)
REFERENCES dbo.DimAsalSurat(AsalKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratMasuk_DimPerihal')
ALTER TABLE dbo.Fact_SuratMasuk
ADD CONSTRAINT FK_SuratMasuk_DimPerihal FOREIGN KEY
(PerihalKey) REFERENCES dbo.DimPerihalSurat(PerihalKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratMasuk_DimUnit')
ALTER TABLE dbo.Fact_SuratMasuk
ADD CONSTRAINT FK_SuratMasuk_DimUnit FOREIGN KEY (UnitKey)
REFERENCES dbo.DimUnitKerja(UnitKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratMasuk_Status')
ALTER TABLE dbo.Fact_SuratMasuk
ADD CONSTRAINT FK_SuratMasuk_Status FOREIGN KEY (StatusKey)
REFERENCES dbo.DimStatus(StatusKey);
GO

-- Fact_SuratKeluar
```

```
IF OBJECT_ID('dbo.Fact_SuratKeluar') IS NULL
CREATE TABLE dbo.Fact_SuratKeluar (
    SuratKeluarKey BIGINT IDENTITY(1,1) NOT NULL,
    DateKey INT NOT NULL,
    TujuanKey INT NOT NULL,
    PerihalKey INT NOT NULL,
    UnitKey INT NOT NULL,
    StatusKey INT NOT NULL,
    JumlahSurat INT DEFAULT 1,
    WaktuPenyelesaian INT NULL,
    NomorSurat VARCHAR(200) NULL,
    SourceSystem VARCHAR(100) NULL,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-- FK SuratKeluar -> dims
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratKeluar_DimTanggal')
ALTER TABLE dbo.Fact_SuratKeluar
ADD CONSTRAINT FK_SuratKeluar_DimTanggal FOREIGN KEY
(DateKey) REFERENCES dbo.DimTanggal(DateKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratKeluar_DimTujuan')
ALTER TABLE dbo.Fact_SuratKeluar
ADD CONSTRAINT FK_SuratKeluar_DimTujuan FOREIGN KEY
(TujuanKey) REFERENCES dbo.DimTujuanSurat(TujuanKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratKeluar_DimPerihal')
ALTER TABLE dbo.Fact_SuratKeluar
ADD CONSTRAINT FK_SuratKeluar_DimPerihal FOREIGN KEY
(PerihalKey) REFERENCES dbo.DimPerihalSurat(PerihalKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratKeluar_DimUnit')
ALTER TABLE dbo.Fact_SuratKeluar
ADD CONSTRAINT FK_SuratKeluar_DimUnit FOREIGN KEY (UnitKey)
REFERENCES dbo.DimUnitKerja(UnitKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_SuratKeluar_Status')
ALTER TABLE dbo.Fact_SuratKeluar
ADD CONSTRAINT FK_SuratKeluar_Status FOREIGN KEY (StatusKey)
REFERENCES dbo.DimStatus(StatusKey);
```

```
GO

-- Fact_LayananAdministrasi (optional)
IF OBJECT_ID('dbo.Fact_LayananAdministrasi') IS NULL
CREATE TABLE dbo.Fact_LayananAdministrasi (
    LayananKey BIGINT IDENTITY(1,1) NOT NULL,
    DateKey INT NOT NULL,
    UnitKey INT NOT NULL,
    JenisLayananKey INT NOT NULL,
    StatusKey INT NOT NULL,
    RequesterName VARCHAR(200) NULL,
    DurasiProses INT NULL,
    Remarks VARCHAR(500) NULL,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-- FK Layanan -> dims
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Layanan_DimTanggal')
ALTER TABLE dbo.Fact_LayananAdministrasi
ADD CONSTRAINT FK_Layanan_DimTanggal FOREIGN KEY (DateKey)
REFERENCES dbo.DimTanggal(DateKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Layanan_DimUnit')
ALTER TABLE dbo.Fact_LayananAdministrasi
ADD CONSTRAINT FK_Layanan_DimUnit FOREIGN KEY (UnitKey)
REFERENCES dbo.DimUnitKerja(UnitKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Layanan_Jenis')
ALTER TABLE dbo.Fact_LayananAdministrasi
ADD CONSTRAINT FK_Layanan_Jenis FOREIGN KEY (JenisLayananKey)
REFERENCES dbo.DimJenisLayanan(JenisLayananKey);
GO
IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys WHERE name =
'FK_Layanan_Status')
ALTER TABLE dbo.Fact_LayananAdministrasi
ADD CONSTRAINT FK_Layanan_Status FOREIGN KEY (StatusKey)
REFERENCES dbo.DimStatus(StatusKey);
GO

-- ============================
-- Done
-- ============================
```



2. Step 2: Indexing Strategy

a) Clustered Index on Fact Table

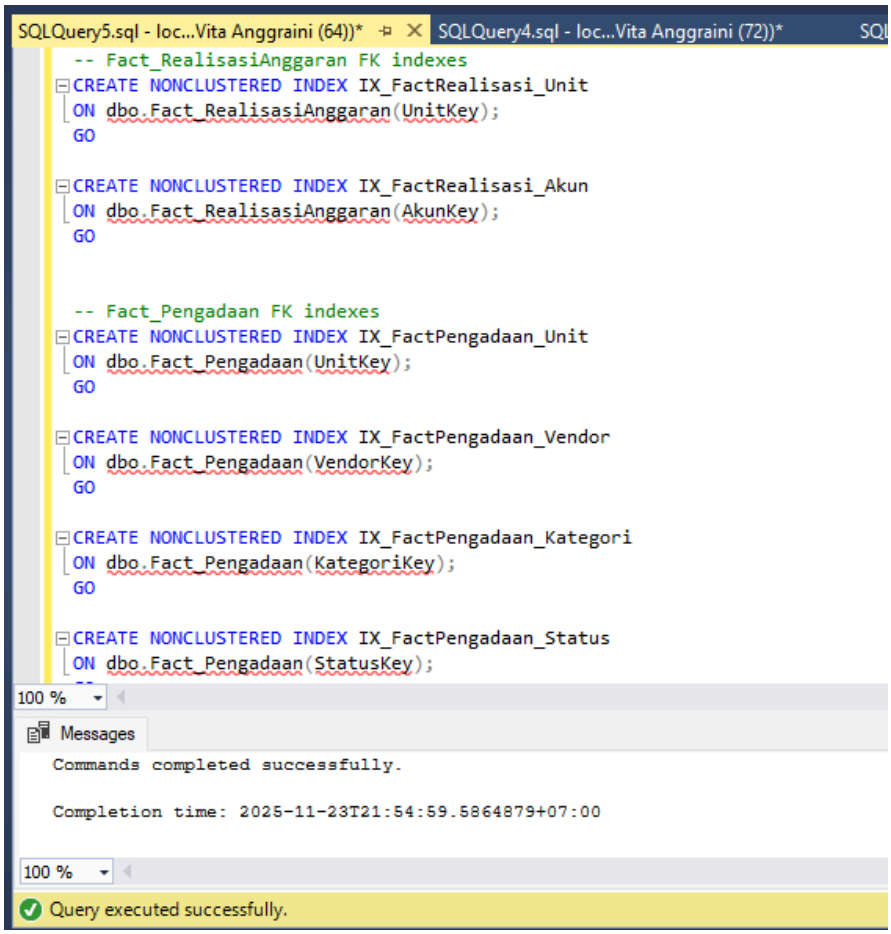
Clustered index pada fact table menggunakan primary key default SQL Server. Pembuatan clustered index tambahan tidak dilakukan karena primary key sudah berperan sebagai clustered index secara default.

b) Non-Clustered Indexes

```
-----------------------------------------------------
-- 2. NON-CLUSTERED INDEXES FOR FOREIGN KEYS
-- (Optimize joins from fact → dimensions)
-----------------------------------------------------

-- Fact_RealisasiAnggaran FK indexes
CREATE NONCLUSTERED INDEX IX_FactRealisasi_Unit
ON dbo.Fact_RealisasiAnggaran(UnitKey);
GO

CREATE NONCLUSTERED INDEX IX_FactRealisasi_Akun
ON dbo.Fact_RealisasiAnggaran(AkunKey);
GO
```

```
-- Fact_Pengadaan FK indexes
CREATE NONCLUSTERED INDEX IX_FactPengadaan_Unit
ON dbo.Fact_Pengadaan(UnitKey);
GO

CREATE NONCLUSTERED INDEX IX_FactPengadaan_Vendor
ON dbo.Fact_Pengadaan(VendorKey);
GO

CREATE NONCLUSTERED INDEX IX_FactPengadaan_Kategori
ON dbo.Fact_Pengadaan(KategoriKey);
GO

CREATE NONCLUSTERED INDEX IX_FactPengadaan_Status
ON dbo.Fact_Pengadaan(StatusKey);
GO


-- Fact_PeminjamanFasilitas FK indexes
CREATE NONCLUSTERED INDEX IX_FactPeminjaman_Fasilitas
ON dbo.Fact_PeminjamanFasilitas(FasilitasKey);
GO

CREATE NONCLUSTERED INDEX IX_FactPeminjaman_Unit
ON dbo.Fact_PeminjamanFasilitas(UnitKey);
GO

CREATE NONCLUSTERED INDEX IX_FactPeminjaman_Keperluan
ON dbo.Fact_PeminjamanFasilitas(KeperluanKey);
GO


-- Fact_SuratMasuk FK indexes
CREATE NONCLUSTERED INDEX IX_FactSuratMasuk_Asal
ON dbo.Fact_SuratMasuk(AsalKey);
GO

CREATE NONCLUSTERED INDEX IX_FactSuratMasuk_Perihal
ON dbo.Fact_SuratMasuk(PerihalKey);
GO

CREATE NONCLUSTERED INDEX IX_FactSuratMasuk_Unit
ON dbo.Fact_SuratMasuk(UnitKey);
GO
```

```
-- Fact_SuratKeluar FK indexes
CREATE NONCLUSTERED INDEX IX_FactSuratKeluar_Tujuan
ON dbo.Fact_SuratKeluar(TujuanKey);
GO

CREATE NONCLUSTERED INDEX IX_FactSuratKeluar_Perihal
ON dbo.Fact_SuratKeluar(PerihalKey);
GO

CREATE NONCLUSTERED INDEX IX_FactSuratKeluar_Unit
ON dbo.Fact_SuratKeluar(UnitKey);
GO
```

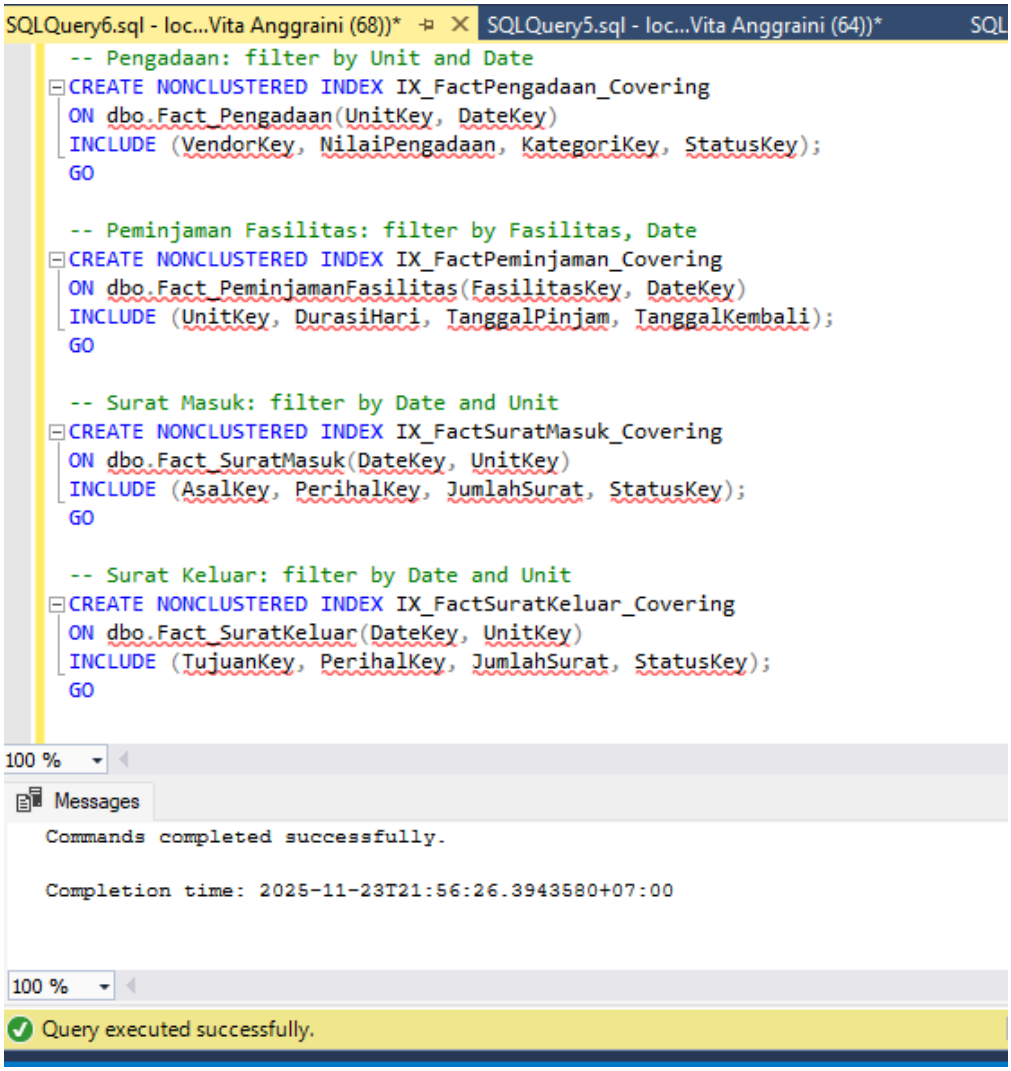


c) Columnstore Index (for large fact tables)

```
-------------------------------------------------
-- 3. COVERING INDEXES FOR COMMON ANALYTICAL QUERIES
-- (Speed up dashboard queries)
```

```
-------------------------------------------------

-- Pengadaan: filter by Unit and Date
CREATE NONCLUSTERED INDEX IX_FactPengadaan_Covering
ON dbo.Fact_Pengadaan(UnitKey, DateKey)
INCLUDE (VendorKey, NilaiPengadaan, KategoriKey, StatusKey);
GO

-- Peminjaman Fasilitas: filter by Fasilitas, Date
CREATE NONCLUSTERED INDEX IX_FactPeminjaman_Covering
ON dbo.Fact_PeminjamanFasilitas(FasilitasKey, DateKey)
INCLUDE (UnitKey, DurasiHari, TanggalPinjam, TanggalKembali);
GO

-- Surat Masuk: filter by Date and Unit
CREATE NONCLUSTERED INDEX IX_FactSuratMasuk_Covering
ON dbo.Fact_SuratMasuk(DateKey, UnitKey)
INCLUDE (AsalKey, PerihalKey, JumlahSurat, StatusKey);
GO

-- Surat Keluar: filter by Date and Unit
CREATE NONCLUSTERED INDEX IX_FactSuratKeluar_Covering
ON dbo.Fact_SuratKeluar(DateKey, UnitKey)
INCLUDE (TujuanKey, PerihalKey, JumlahSurat, StatusKey);
GO



-------------------------------------------------
-- 4. COLUMNSTORE INDEXES (optional but recommended)
-- (For large fact tables)
-------------------------------------------------

CREATE NONCLUSTERED COLUMNSTORE INDEX CSX_FactRealisasi
ON dbo.Fact_RealisasiAnggaran (DateKey, UnitKey, AkunKey,
NilaiRealisasi);
GO

CREATE NONCLUSTERED COLUMNSTORE INDEX CSX_FactPengadaan
ON dbo.Fact_Pengadaan (DateKey, UnitKey, VendorKey, KategoriKey,
NilaiPengadaan);
GO

CREATE NONCLUSTERED COLUMNSTORE INDEX
CSX_FactPeminjaman
ON dbo.Fact_PeminjamanFasilitas (DateKey, FasilitasKey, UnitKey,
DurasiHari, KeperluanKey);
```

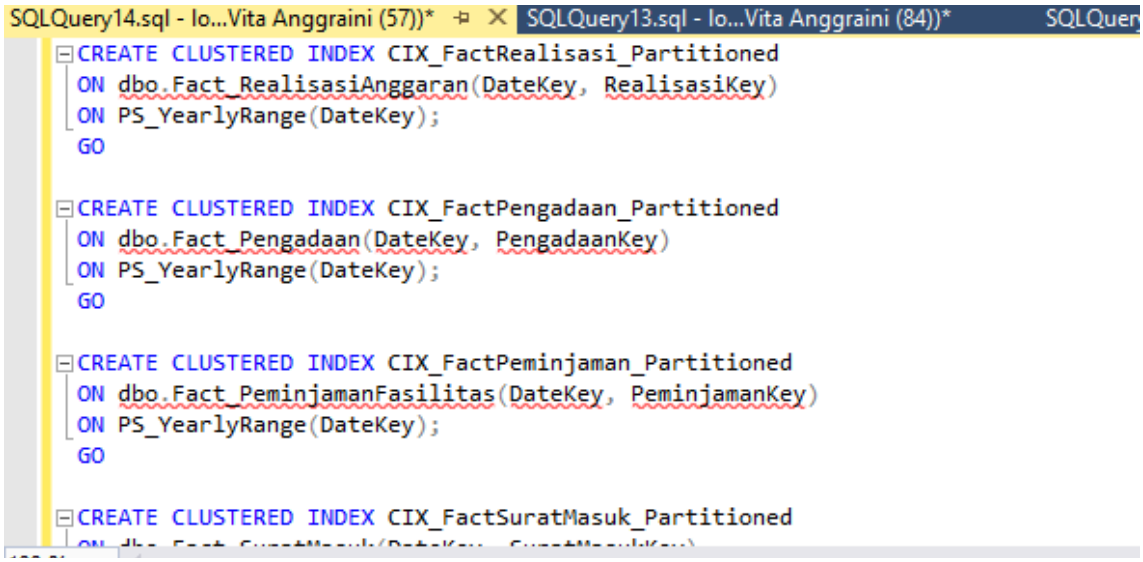
```
GO

CREATE NONCLUSTERED COLUMNSTORE INDEX
CSX_FactSuratMasuk
ON dbo.Fact_SuratMasuk (DateKey, UnitKey, AsalKey, PerihalKey);
GO

CREATE NONCLUSTERED COLUMNSTORE INDEX
CSX_FactSuratKeluar
ON dbo.Fact_SuratKeluar (DateKey, UnitKey, TujuanKey, PerihalKey);
GO
```



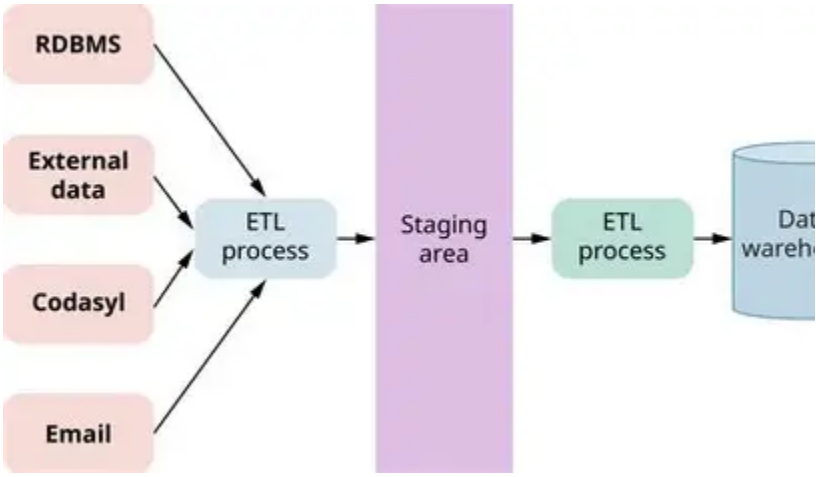
3. **Step 3: Partitioning Strategy**

```
-- ================================================
-- 05_Create_Partitions.sql
-- Partitioning Strategy for DW Non-Akademik
-- ================================================

USE DM_NonAkademik_DW;
GO

-------------------------------------------------
-- 1. PARTITION FUNCTION (Range by Year)
-- We use DateKey in format YYYYMMDD
-------------------------------------------------

CREATE PARTITION FUNCTION PF_YearlyRange (INT)
AS RANGE RIGHT FOR VALUES
(
   20200101,
   20210101,
   20220101,
   20230101,
   20240101,
   20250101,
   20260101
);
GO

-------------------------------------------------
-- 2. PARTITION SCHEME (All on PRIMARY for simplicity)
-------------------------------------------------
CREATE PARTITION SCHEME PS_YearlyRange
AS PARTITION PF_YearlyRange
ALL TO ([PRIMARY]);
GO

-------------------------------------------------
-- 3. Move FACT TABLES to Partition Scheme
-- (Re-create clustered index ON PS_YearlyRange(DateKey))
-------------------------------------------------

-- Fact Realsiasi Anggaran
CREATE CLUSTERED INDEX CIX_FactRealisasi_Partitioned
ON dbo.Fact_RealisasiAnggaran(DateKey, RealisasiKey)
ON PS_YearlyRange(DateKey);
GO
```

```
-- Fact Pengadaan
CREATE CLUSTERED INDEX CIX_FactPengadaan_Partitioned
ON dbo.Fact_Pengadaan(DateKey, PengadaanKey)
ON PS_YearlyRange(DateKey);
GO

-- Fact Peminjaman Fasilitas
CREATE CLUSTERED INDEX CIX_FactPeminjaman_Partitioned
ON dbo.Fact_PeminjamanFasilitas(DateKey, PeminjamanKey)
ON PS_YearlyRange(DateKey);
GO

-- Fact Surat Masuk
CREATE CLUSTERED INDEX CIX_FactSuratMasuk_Partitioned
ON dbo.Fact_SuratMasuk(DateKey, SuratMasukKey)
ON PS_YearlyRange(DateKey);
GO

-- Fact Surat Keluar
CREATE CLUSTERED INDEX CIX_FactSuratKeluar_Partitioned
ON dbo.Fact_SuratKeluar(DateKey, SuratKeluarKey)
ON PS_YearlyRange(DateKey);
GO
```



4. Step 4: ETL Design



Pada tahap ini dirancang arsitektur ETL (Extract, Transform, Load) yang digunakan untuk memindahkan data dari sumber ke dalam Data Warehouse Non-Akademik. Karena data operasional asli tidak tersedia, proses ETL dilakukan menggunakan data dummy, namun alur ETL tetap mengikuti standar implementasi Data Warehouse.

a) ETL Architecture Design

- **Source to Staging: Extract raw data**

  Tahap Extract melakukan pengambilan data dari berbagai sumber dummy yang mensimulasikan proses bisnis unit kerja, antara lain:

  - Dummy data Surat Masuk
  - Dummy data Surat Keluar
  - Dummy data Pengadaan Barang/Jasa
  - Dummy data Realisasi Anggaran
  - Dummy data Peminjaman Fasilitas
  - Dummy data Vendor
  - Dummy data Unit Kerja
  - Dummy data Lokasi
  - Dummy data BMN

  Pada tahap ini, data belum mengalami transformasi, melainkan langsung disalin apa adanya ke tabel staging agar tetap terjaga kelengkapannya.

  Tujuan tahap ini:

  - Menyalin data mentah secara apa adanya

- Menyediakan area buffer agar proses ETL dapat diulang tanpa memengaruhi sumber
- Menghindari perubahan langsung pada data sumber

- **Staging to Integration: Data cleansing dan transformation**

  Data mentah yang berada pada staging selanjutnya dibersihkan, distandarisasi, dan ditransformasikan. Proses yang dilakukan meliputi:

  - Normalisasi teks menggunakan TRIM() dan UPPER()
  - Validasi dan standarisasi format tanggal
  - Mapping kode seperti kategori pengadaan, akun belanja, keperluan, dan lokasi
  - Standardisasi nama unit kerja dan vendor
  - Lookup referensi untuk menghasilkan surrogate key pada tabel dimensi
  - Penanganan missing value atau data tidak konsisten

  Tujuan tahap ini:

  - Menghasilkan data yang bersih, konsisten, dan siap di-load ke Data Warehouse
  - Memastikan struktur data sesuai dengan desain dimensional

- **Integration to Data Warehouse: Load ke fact dan dimension tables**

  Setelah data melalui tahap transformasi, data dimuat ke dalam tabel dimensi (dimension) dan tabel fakta (fact) sesuai dengan model Data Warehouse yang telah dirancang.

  Tabel Dimensi yang Diload:

  - DimUnitKerja
  - DimVendor
  - DimLokasi
  - DimFasilitas
  - DimAkunBelanja
  - DimKategoriPengadaan
  - DimBMN
  - DimKeperluan
  - DimPerihalSurat
  - DimStatus
  - DimTanggal

Tabel Fakta yang Diload:

- Fact_SuratMasuk
- Fact_SuratKeluar
- Fact_Pengadaan
- Fact_RealisasiAnggaran
- Fact_PeminjamanFasilitas
- Fact_BMN

Proses loading dilakukan dengan urutan:

- Load tabel dimensi terlebih dahulu untuk menghasilkan DimKey
- Load tabel fakta dengan terlebih dahulu melakukan lookup key pada dimensi

Tujuan tahap ini:

- Mengisi struktur Data Warehouse secara konsisten
- Mendukung kebutuhan analisis dan pelaporan melalui skema star schema yang telah dibuat

b) Creating Staging Tables

Pada tahap ini dibuat tabel-tabel staging sebagai tempat penyimpanan data mentah (raw data) hasil proses extract dari berbagai sumber dummy. Tabel staging berfungsi sebagai area penampung sementara sebelum dilakukan proses cleansing, transformasi, dan loading ke dalam Data Warehouse.

Seluruh tabel staging ditempatkan pada schema stg untuk memisahkan area staging dari schema lainnya.

Berikut adalah SQL script yang digunakan untuk membuat tabel-tabel staging:

```
-- ============================================================
-- 06_Create_Staging.sql
-- Staging Schema & Tables untuk DW Non-Akademik
-- ============================================================

CREATE SCHEMA stg;
GO

------------------------------------------------------
```

```
-- STAGING: SURAT MASUK
-----------------------------------------------------
CREATE TABLE stg.SuratMasuk (
    NomorSurat VARCHAR(200),
    AsalSurat VARCHAR(250),
    KotaAsal VARCHAR(100),
    TanggalDiterima DATE,
    Perihal VARCHAR(300),
    JumlahLampiran INT,
    StatusSurat VARCHAR(100),
    UnitTujuan VARCHAR(200),
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-----------------------------------------------------
-- STAGING: SURAT KELUAR
-----------------------------------------------------
CREATE TABLE stg.SuratKeluar (
    NomorSurat VARCHAR(200),
    TujuanSurat VARCHAR(250),
    KotaTujuan VARCHAR(100),
    TanggalKeluar DATE,
    Perihal VARCHAR(300),
    StatusSurat VARCHAR(100),
    UnitPengirim VARCHAR(200),
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-----------------------------------------------------
-- STAGING: PENGADAAN BARANG/JASA
-----------------------------------------------------
CREATE TABLE stg.Pengadaan (
    NomorPengadaan VARCHAR(100),
    NamaVendor VARCHAR(250),
    KotaVendor VARCHAR(100),
    KategoriPengadaan VARCHAR(200),
    NilaiPengadaan DECIMAL(18,2),
    StatusPengadaan VARCHAR(100),
    UnitPemohon VARCHAR(200),
    TanggalPengadaan DATE,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

```
-----------------------------------------------------
-- STAGING: REALISASI ANGGARAN
-----------------------------------------------------
CREATE TABLE stg.RealisasiAnggaran (
    KodeAkun VARCHAR(50),
    NamaAkun VARCHAR(250),
    NilaiRealisasi DECIMAL(18,2),
    UnitKerja VARCHAR(200),
    TanggalRealisasi DATE,
    Keterangan VARCHAR(500),
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-----------------------------------------------------
-- STAGING: PEMINJAMAN FASILITAS
-----------------------------------------------------
CREATE TABLE stg.PeminjamanFasilitas (
    NamaFasilitas VARCHAR(250),
    JenisFasilitas VARCHAR(150),
    Lokasi VARCHAR(200),
    UnitPeminjam VARCHAR(200),
    Keperluan VARCHAR(250),
    TanggalPinjam DATE,
    TanggalKembali DATE,
    StatusPeminjaman VARCHAR(100),
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

-----------------------------------------------------
-- STAGING: DATA BMN
-----------------------------------------------------
CREATE TABLE stg.BMN (
    KodeBarang VARCHAR(100),
    NamaBarang VARCHAR(250),
    KategoriBarang VARCHAR(200),
    JenisBarang VARCHAR(200),
    LokasiSaatIni VARCHAR(200),
    UnitPengguna VARCHAR(200),
    TanggalPerolehan DATE,
    SumberPerolehan VARCHAR(100),
    HargaPerolehan DECIMAL(18,2),
    KondisiSaatIni VARCHAR(100),
    LoadDate DATETIME DEFAULT GETDATE()
);
```

```
GO

------------------------------------------------------
-- STAGING: UNIT KERJA
------------------------------------------------------
CREATE TABLE stg.UnitKerja (
   UnitName VARCHAR(250),
   UnitType VARCHAR(100),
   ParentUnit VARCHAR(200),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO

------------------------------------------------------
-- STAGING: VENDOR
------------------------------------------------------
CREATE TABLE stg.Vendor (
   NamaVendor VARCHAR(250),
   Kota VARCHAR(100),
   ContactPerson VARCHAR(150),
   ContactPhone VARCHAR(50),
   Email VARCHAR(150),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO

------------------------------------------------------
-- STAGING: LOKASI
------------------------------------------------------
CREATE TABLE stg.Lokasi (
   Lokasi VARCHAR(250),
   Kota VARCHAR(100),
   Provinsi VARCHAR(100),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

c) ETL Mapping Document

- **Mapping - Surat Masuk**

Source Table	**Source Column**	**Staging Column**	**Target Table**	**Target Column**	**Transformation**
SuratMas	NomorSu	NomorSu	Fact_Sura	NomorSu	TRIM()

uk_Dummy	rat	rat	tMasuk	rat	
SuratMasuk_Dummy	AsalSurat	AsalSurat	DimAsalSurat	AsalName	UPPER(TRIM())
SuratMasuk_Dummy	KotaAsal	KotaAsal	DimLokasi	Kota	UPPER()
SuratMasuk_Dummy	TanggalDiterima	TanggalDiterima	DimTanggal	FullDate	Convert to YYYY-MM-DD
SuratMasuk_Dummy	Perihal	Perihal	DimPerihalSurat	PerihalDesc	Clean text
SuratMasuk_Dummy	UnitTujuan	UnitTujuan	DimUnitKerja	UnitName	Standardize
SuratMasuk_Dummy	StatusSurat	StatusSurat	DimStatus	StatusName	Map category

- **Mapping - Surat Keluar**

Source Table	**Source Column**	**Staging Column**	**Target Table**	**Target Column**	**Transformation**
SuratKeluar_Dummy	NomorSurat	NomorSurat	Fact_SuratKeluar	NomorSurat	TRIM()
SuratKeluar_Dummy	TujuanSurat	TujuanSurat	DimTujuanSurat	TujuanName	UPPER()
SuratKeluar_Dummy	KotaTujuan	KotaTujuan	DimLokasi	Kota	UPPER()
SuratKeluar_Dumm	TanggalKeluar	TanggalKeluar	DimTanggal	FullDate	Convert date

y					
SuratKeluar_Dummy	Perihal	Perihal	DimPerihalSurat	PerihalDesc	Clean text
SuratKeluar_Dummy	StatusSurat	StatusSurat	DimStatus	StatusName	Mapping

- **Mapping - Pengadaan Barang/Jasa**

Source Column	**Staging Column**	**Target Table**	**Target Column**	**Transformation**
NomorPengadaan	NomorPengadaan	Fact_Pengadaan	NomorPengadaan	TRIM()
NamaVendor	NamaVendor	Dim_Vendor	VendorName	UPPER()
KotaVendor	KotaVendor	DimLokasi	Kota	Standardize
KategoriPengadaan	KategoriPengadaan	DimKategoriPengadaan	CategoryName	Map category
NilaiPengadaan	NilaiPengadaan	Fact_Pengadaan	NilaiPengadaan	None
UnitPemohon	UnitPemohon	DimUnitKerja	UnitName	Normalize
StatusPengadaan	StatusPengadaan	DimStatus	StatusName	Standardize

- **Mapping - Realisasi Anggaran**

Source Column	**Staging Column**	**Target Table**	**Target Column**	**Transformation**
KodeAkun	KodeAkun	DimAkunBelanja	KodeAkun	TRIM()
NamaAkun	NamaAkun	DimAkunBelanja	NamaAkun	UPPER()
NilaiRealisa	NilaiRealisa	Fact_Realis	NilaiRealisa	None

si	si	asiAnggaran	si	
UnitKerja	UnitKerja	DimUnitKerja	UnitName	Standardize
TanggalRealisasi	TanggalRealisasi	DimTanggal	FullDate	Convert date

- **Mapping - Peminjaman Fasilitas**

Source Column	**Staging Column**	**Target Table**	**Target Column**	**Transformation**
NamaFasilitas	NamaFasilitas	DimFasilitas	FasilitasName	TRIM(), UPPER()
JenisFasilitas	JenisFasilitas	DimFasilitas	JenisFasilitas	Map Category
Lokasi	Lokasi	DimLokasi	LokasiName	UPPER()
UnitPeminjam	UnitPeminjam	DimUnitKerja	UnitName	Normalize
Keperluan	Keperluan	DimKeperluan	KeperluanDesc	Clean text
TanggalPinjam	TanggalPinjam	DimTanggal	FullDate	Convert date
StatusPeminjaman	StatusPeminjaman	DimStatus	StatusName	Standardize

- **Mapping - Data BMN**

Source Column	**Staging Column**	**Target Table**	**Target Column**	**Transformation**
KodeBarang	KodeBarang	DimBMN	KodeBarang	TRIM()
NamaBarang	NamaBarang	DimBMN	NamaBarang	UPPER()
KategoriBarang	KategoriBarang	DimBMN	KategoriBarang	Normalize
JenisBarang	JenisBarang	DimBMN	JenisBarang	Map type

LokasiSaatIni	LokasiSaatIni	DimLokasi	LokasiName	Standardize
UnitPenggunna	UnitPenggunna	DimUnitKerja	UnitName	Normalize
TanggalPerolehan	TanggalPerolehan	DimTanggal	FullDate	Convert date
KondisiSaatIni	KondisiSaatIni	DimStatus	StatusName	Mapping

5. **Step 5: ETL Implementation Using T-SQL Stored Procedures**

```
USE DM_NonAkademik_DW;
GO

/*************************************************************
  ETL Procedures (based on your staging and dim/fact schema)
*************************************************************/

-- ========== DIM LOAD PROCEDURES ==========

/* DimLokasi (SCD0: insert if not exists) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimLokasi
AS
BEGIN
   SET NOCOUNT ON;
   INSERT INTO dbo.DimLokasi (LokasiCode, LokasiName, Alamat, Kota, Provinsi,
CreatedDate)
   SELECT NULL, s.Lokasi, NULL, s.Kota, s.Provinsi, GETDATE()
   FROM stg.Lokasi s
   WHERE NOT EXISTS (
      SELECT 1 FROM dbo.DimLokasi d WHERE d.LokasiName = s.Lokasi
   );
END;
GO

/* DimVendor (SCD0) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimVendor
AS
BEGIN
   SET NOCOUNT ON;
   INSERT INTO dbo.DimVendor (VendorCode, VendorName, Category, City,
ContactPerson, ContactPhone, Email, CreatedDate)
   SELECT NULL, s.NamaVendor, NULL, s.Kota, s.ContactPerson, s.ContactPhone,
```

```
s.Email, GETDATE()
    FROM stg.Vendor s
    WHERE NOT EXISTS (
        SELECT 1 FROM dbo.DimVendor d WHERE d.VendorName = s.NamaVendor
    );
END;
GO

/* DimAkunBelanja (SCD0) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimAkunBelanja
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO dbo.DimAkunBelanja (KodeAkun, NamaAkun, KelompokAkun,
KategoriBelanja, CreatedDate)
    SELECT DISTINCT s.KodeAkun, s.NamaAkun, NULL, NULL, GETDATE()
    FROM stg.RealisasiAnggaran s
    WHERE NOT EXISTS (
        SELECT 1 FROM dbo.DimAkunBelanja d WHERE d.KodeAkun = s.KodeAkun
    );
END;
GO

/* DimKategoriPengadaan (SCD0) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimKategoriPengadaan
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO dbo.DimKategoriPengadaan (KategoriName, SubKategori,
CreatedDate)
    SELECT DISTINCT s.KategoriPengadaan, NULL, GETDATE()
    FROM stg.Pengadaan s
    WHERE NOT EXISTS (
        SELECT 1 FROM dbo.DimKategoriPengadaan d WHERE d.KategoriName =
s.KategoriPengadaan
    );
END;
GO

/* DimKeperluan (SCD0) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimKeperluan
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO dbo.DimKeperluan (KeperluanDesc, CreatedDate)
    SELECT DISTINCT s.Keperluan, GETDATE()
```

```
    FROM stg.PeminjamanFasilitas s
    WHERE s.Keperluan IS NOT NULL
      AND NOT EXISTS (SELECT 1 FROM dbo.DimKeperluan d WHERE
d.KeperluanDesc = s.Keperluan);
END;
GO

/* DimPerihalSurat (SCD0) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimPerihalSurat
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO dbo.DimPerihalSurat (PerihalDesc, CreatedDate)
    SELECT DISTINCT t.Perihal, GETDATE()
    FROM (
        SELECT Perihal FROM stg.SuratMasuk
        UNION
        SELECT Perihal FROM stg.SuratKeluar
    ) t
    WHERE NOT EXISTS (SELECT 1 FROM dbo.DimPerihalSurat d WHERE
d.PerihalDesc = t.Perihal);
END;
GO

/* DimStatus (ProcessType, StatusName) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimStatus
AS
BEGIN
    SET NOCOUNT ON;

    -- SURAT statuses (from both masuk & keluar)
    INSERT INTO dbo.DimStatus (ProcessType, StatusName, Description,
CreatedDate)
    SELECT 'SURAT', StatusSurat, NULL, GETDATE()
    FROM (
        SELECT StatusSurat FROM stg.SuratMasuk
        UNION
        SELECT StatusSurat FROM stg.SuratKeluar
    ) x
    WHERE x.StatusSurat IS NOT NULL
      AND NOT EXISTS (
        SELECT 1 FROM dbo.DimStatus d WHERE d.ProcessType = 'SURAT' AND
d.StatusName = x.StatusSurat
      );

    -- PENGADAAN statuses
```

```
    INSERT INTO dbo.DimStatus (ProcessType, StatusName, Description,
CreatedDate)
    SELECT 'PENGADAAN', StatusPengadaan, NULL, GETDATE()
    FROM stg.Pengadaan s
    WHERE s.StatusPengadaan IS NOT NULL
      AND NOT EXISTS (
        SELECT 1 FROM dbo.DimStatus d WHERE d.ProcessType = 'PENGADAAN'
AND d.StatusName = s.StatusPengadaan
      );

    -- PEMINJAMAN statuses
    INSERT INTO dbo.DimStatus (ProcessType, StatusName, Description,
CreatedDate)
    SELECT 'PEMINJAMAN', StatusPeminjaman, NULL, GETDATE()
    FROM stg.PeminjamanFasilitas s
    WHERE s.StatusPeminjaman IS NOT NULL
      AND NOT EXISTS (
        SELECT 1 FROM dbo.DimStatus d WHERE d.ProcessType = 'PEMINJAMAN'
AND d.StatusName = s.StatusPeminjaman
      );
END;
GO

/* DimAsalSurat (SCD0) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimAsalSurat
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO dbo.DimAsalSurat (AsalName, AsalType, Kota, CreatedDate)
    SELECT DISTINCT s.AsalSurat, NULL, s.KotaAsal, GETDATE()
    FROM stg.SuratMasuk s
    WHERE s.AsalSurat IS NOT NULL
      AND NOT EXISTS (SELECT 1 FROM dbo.DimAsalSurat d WHERE
d.AsalName = s.AsalSurat);
END;
GO

/* DimTujuanSurat (SCD0) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimTujuanSurat
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO dbo.DimTujuanSurat (TujuanName, TujuanType, Kota, CreatedDate)
    SELECT DISTINCT s.TujuanSurat, NULL, s.KotaTujuan, GETDATE()
    FROM stg.SuratKeluar s
    WHERE s.TujuanSurat IS NOT NULL
```

```
    AND NOT EXISTS (SELECT 1 FROM dbo.DimTujuanSurat d WHERE
d.TujuanName = s.TujuanSurat);
END;
GO

/* DimBMN (SCD0: insert if not exists; no IsCurrent in your dim) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimBMN
AS
BEGIN
  SET NOCOUNT ON;
  INSERT INTO dbo.DimBMN (
    KodeBarang, NamaBarang, KategoriBarang, JenisBarang,
    TanggalPerolehan, SumberPerolehan, HargaPerolehan,
    KondisiAwal, KondisiSaatIni, CreatedDate, ModifiedDate
  )
  SELECT s.KodeBarang, s.NamaBarang, s.KategoriBarang, s.JenisBarang,
      s.TanggalPerolehan, s.SumberPerolehan, s.HargaPerolehan,
      s.KondisiSaatIni, s.KondisiSaatIni, GETDATE(), GETDATE()
  FROM stg.BMN s
  WHERE NOT EXISTS (SELECT 1 FROM dbo.DimBMN d WHERE
d.KodeBarang = s.KodeBarang);
END;
GO

/* DimFasilitas (SCD0 insert-if-not-exists) */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimFasilitas
AS
BEGIN
  SET NOCOUNT ON;
  INSERT INTO dbo.DimFasilitas (FasilitasCode, NamaFasilitas, JenisFasilitas,
LokasiCode, Kapasitas, UnitPengelolaCode, CreatedDate)
  SELECT NULL, s.NamaFasilitas, s.JenisFasilitas, s.Lokasi, NULL, NULL,
GETDATE()
  FROM stg.PeminjamanFasilitas s
  WHERE NOT EXISTS (SELECT 1 FROM dbo.DimFasilitas d WHERE
d.NamaFasilitas = s.NamaFasilitas);
END;
GO

/* DimUnitKerja (SCD Type 2) - because your dim has
IsCurrent/EffectiveDate/ExpiryDate */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimUnitKerja
AS
BEGIN
  SET NOCOUNT ON;
  BEGIN TRAN;
```

```
    -- 1) expire existing records where attributes changed (match by UnitName)
    UPDATE d
    SET d.ExpiryDate = GETDATE(),
        d.IsCurrent = 0,
        d.ModifiedDate = GETDATE()
    FROM dbo.DimUnitKerja d
    INNER JOIN stg.UnitKerja s ON d.UnitName = s.UnitName
    WHERE d.IsCurrent = 1
      AND (
            ISNULL(d.UnitType,'') <> ISNULL(s.UnitType,'')
         OR ISNULL(d.ParentUnitCode,'') <> ISNULL(s.ParentUnit,'')
      );

    -- 2) insert new records for new units or changed units
    INSERT INTO dbo.DimUnitKerja (UnitCode, UnitName, UnitType,
ParentUnitCode, EffectiveDate, ExpiryDate, IsCurrent, CreatedDate, ModifiedDate)
    SELECT NULL, s.UnitName, s.UnitType, s.ParentUnit, GETDATE(), NULL, 1,
GETDATE(), GETDATE()
    FROM stg.UnitKerja s
    WHERE NOT EXISTS (
        SELECT 1 FROM dbo.DimUnitKerja d WHERE d.UnitName = s.UnitName
AND d.IsCurrent = 1
    );

    COMMIT TRAN;
END;
GO

/* DimJenisLayanan (SCD0) - optional */
CREATE OR ALTER PROCEDURE dbo.usp_Load_DimJenisLayanan
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO dbo.DimJenisLayanan (JenisLayananName, Description,
CreatedDate)
    SELECT DISTINCT 'Umum', NULL, GETDATE() -- placeholder if you don't have
staging; keep to avoid failures
    WHERE NOT EXISTS (SELECT 1 FROM dbo.DimJenisLayanan);
END;
GO

-- ========== FACT LOAD PROCEDURES ==========

/* Fact_SuratMasuk */
CREATE OR ALTER PROCEDURE dbo.usp_Load_FactSuratMasuk
```

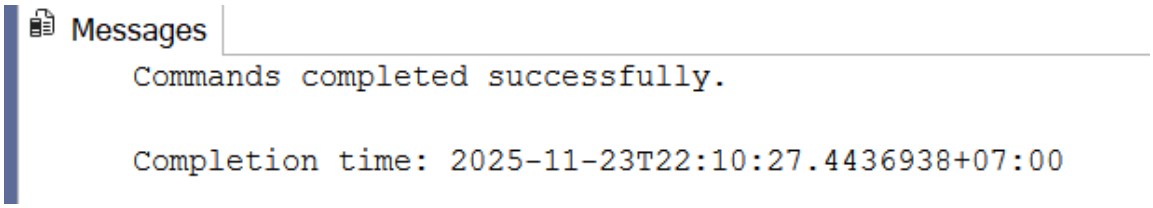
```
AS
BEGIN
   SET NOCOUNT ON;
   INSERT INTO dbo.Fact_SuratMasuk (
      DateKey, AsalKey, PerihalKey, UnitKey, StatusKey,
      JumlahSurat, WaktuProses, NomorSurat, SourceSystem, LoadDate
   )
   SELECT
      CAST(CONVERT(CHAR(8), s.TanggalDiterima, 112) AS INT) AS DateKey,
      da.AsalKey,
      dp.PerihalKey,
      du.UnitKey,
      ds.StatusKey,
      ISNULL(s.JumlahLampiran,0),
      NULL,
      s.NomorSurat,
      'SURAT_MASUK',
      GETDATE()
   FROM stg.SuratMasuk s
   LEFT JOIN dbo.DimAsalSurat da ON da.AsalName = s.AsalSurat
   LEFT JOIN dbo.DimPerihalSurat dp ON dp.PerihalDesc = s.Perihal
   LEFT JOIN dbo.DimUnitKerja du ON du.UnitName = s.UnitTujuan AND
du.IsCurrent = 1
   LEFT JOIN dbo.DimStatus ds ON ds.StatusName = s.StatusSurat AND
ds.ProcessType = 'SURAT'
   WHERE NOT EXISTS (SELECT 1 FROM dbo.Fact_SuratMasuk f WHERE
f.NomorSurat = s.NomorSurat);
END;
GO

/* Fact_SuratKeluar */
CREATE OR ALTER PROCEDURE dbo.usp_Load_FactSuratKeluar
AS
BEGIN
   SET NOCOUNT ON;
   INSERT INTO dbo.Fact_SuratKeluar (
      DateKey, TujuanKey, PerihalKey, UnitKey, StatusKey,
      JumlahSurat, WaktuPenyelesaian, NomorSurat, SourceSystem, LoadDate
   )
   SELECT
      CAST(CONVERT(CHAR(8), s.TanggalKeluar, 112) AS INT) AS DateKey,
      dt.TujuanKey,
      dp.PerihalKey,
      du.UnitKey,
      ds.StatusKey,
      1,
```

```
        NULL,
        s.NomorSurat,
        'SURAT_KELUAR',
        GETDATE()
    FROM stg.SuratKeluar s
    LEFT JOIN dbo.DimTujuanSurat dt ON dt.TujuanName = s.TujuanSurat
    LEFT JOIN dbo.DimPerihalSurat dp ON dp.PerihalDesc = s.Perihal
    LEFT JOIN dbo.DimUnitKerja du ON du.UnitName = s.UnitPengirim AND
du.IsCurrent = 1
    LEFT JOIN dbo.DimStatus ds ON ds.StatusName = s.StatusSurat AND
ds.ProcessType = 'SURAT'
    WHERE NOT EXISTS (SELECT 1 FROM dbo.Fact_SuratKeluar f WHERE
f.NomorSurat = s.NomorSurat);
END;
GO

/* Fact_Pengadaan */
CREATE OR ALTER PROCEDURE dbo.usp_Load_FactPengadaan
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO dbo.Fact_Pengadaan (
        DateKey, UnitKey, VendorKey, KategoriKey, NilaiPengadaan,
        StatusKey, NomorPengadaan, VendorName, BMNKey, SourceSystem, LoadDate
    )
    SELECT
        CAST(CONVERT(CHAR(8), s.TanggalPengadaan, 112) AS INT),
        du.UnitKey,
        dv.VendorKey,
        dk.KategoriKey,
        s.NilaiPengadaan,
        ds.StatusKey,
        s.NomorPengadaan,
        s.NamaVendor,
        NULL,
        'PENGADAAN',
        GETDATE()
    FROM stg.Pengadaan s
    LEFT JOIN dbo.DimUnitKerja du ON du.UnitName = s.UnitPemohon AND
du.IsCurrent = 1
    LEFT JOIN dbo.DimVendor dv ON dv.VendorName = s.NamaVendor
    LEFT JOIN dbo.DimKategoriPengadaan dk ON dk.KategoriName =
s.KategoriPengadaan
    LEFT JOIN dbo.DimStatus ds ON ds.StatusName = s.StatusPengadaan AND
ds.ProcessType = 'PENGADAAN'
    WHERE NOT EXISTS (SELECT 1 FROM dbo.Fact_Pengadaan f WHERE
```

```
f.NomorPengadaan = s.NomorPengadaan);
END;
GO

/* Fact_RealisasiAnggaran */
CREATE OR ALTER PROCEDURE dbo.usp_Load_FactRealisasiAnggaran
AS
BEGIN
   SET NOCOUNT ON;
   INSERT INTO dbo.Fact_RealisasiAnggaran (
      DateKey, UnitKey, AkunKey, NilaiRealisasi, SelisihAnggaranRealisasi,
Keterangan, SourceSystem, LoadDate
   )
   SELECT
      CAST(CONVERT(CHAR(8), s.TanggalRealisasi, 112) AS INT),
      du.UnitKey,
      da.AkunKey,
      s.NilaiRealisasi,
      NULL,
      s.Keterangan,
      'REALISASI',
      GETDATE()
   FROM stg.RealisasiAnggaran s
   LEFT JOIN dbo.DimUnitKerja du ON du.UnitName = s.UnitKerja AND
du.IsCurrent = 1
   LEFT JOIN dbo.DimAkunBelanja da ON da.KodeAkun = s.KodeAkun
   WHERE NOT EXISTS (
      SELECT 1 FROM dbo.Fact_RealisasiAnggaran f
      WHERE f.DateKey = CAST(CONVERT(CHAR(8), s.TanggalRealisasi, 112) AS
INT)
        AND f.UnitKey = du.UnitKey
        AND f.AkunKey = da.AkunKey
        AND f.NilaiRealisasi = s.NilaiRealisasi
   );
END;
GO

/* Fact_PeminjamanFasilitas */
CREATE OR ALTER PROCEDURE dbo.usp_Load_FactPeminjamanFasilitas
AS
BEGIN
   SET NOCOUNT ON;
   INSERT INTO dbo.Fact_PeminjamanFasilitas (
      DateKey, FasilitasKey, UnitKey, KeperluanKey, TanggalPinjam, TanggalKembali,
DurasiHari, StatusKey, SourceSystem, LoadDate
   )
```

```
    SELECT
        CAST(CONVERT(CHAR(8), s.TanggalPinjam, 112) AS INT),
        df.FasilitasKey,
        du.UnitKey,
        dk.KeperluanKey,
        s.TanggalPinjam,
        s.TanggalKembali,
        CASE WHEN s.TanggalKembali IS NOT NULL THEN DATEDIFF(DAY,
s.TanggalPinjam, s.TanggalKembali) + 1 ELSE NULL END,
        ds.StatusKey,
        'PEMINJAMAN',
        GETDATE()
    FROM stg.PeminjamanFasilitas s
    LEFT JOIN dbo.DimFasilitas df ON df.NamaFasilitas = s.NamaFasilitas
    LEFT JOIN dbo.DimUnitKerja du ON du.UnitName = s.UnitPeminjam AND
du.IsCurrent = 1
    LEFT JOIN dbo.DimKeperluan dk ON dk.KeperluanDesc = s.Keperluan
    LEFT JOIN dbo.DimStatus ds ON ds.StatusName = s.StatusPeminjaman AND
ds.ProcessType = 'PEMINJAMAN'
    WHERE NOT EXISTS (
        SELECT 1 FROM dbo.Fact_PeminjamanFasilitas f
        WHERE f.TanggalPinjam = s.TanggalPinjam
          AND f.FasilitasKey = df.FasilitasKey
          AND f.UnitKey = du.UnitKey
    );
END;
GO

-- ========== MASTER ETL ==========

CREATE OR ALTER PROCEDURE dbo.usp_Master_ETL
AS
BEGIN
    SET NOCOUNT ON;
    BEGIN TRY
        BEGIN TRAN;

        -- Load dims (order chosen so lookup dims exist for facts)
        EXEC dbo.usp_Load_DimLokasi;
        EXEC dbo.usp_Load_DimVendor;
        EXEC dbo.usp_Load_DimAkunBelanja;
        EXEC dbo.usp_Load_DimKategoriPengadaan;
        EXEC dbo.usp_Load_DimKeperluan;
        EXEC dbo.usp_Load_DimPerihalSurat;
        EXEC dbo.usp_Load_DimStatus;
        EXEC dbo.usp_Load_DimAsalSurat;
```

```
        EXEC dbo.usp_Load_DimTujuanSurat;
        EXEC dbo.usp_Load_DimBMN;
        EXEC dbo.usp_Load_DimFasilitas;

        -- SCD2 for UnitKerja (run after you ensure stg.UnitKerja has required rows)
        EXEC dbo.usp_Load_DimUnitKerja;

        -- Load facts
        EXEC dbo.usp_Load_FactSuratMasuk;
        EXEC dbo.usp_Load_FactSuratKeluar;
        EXEC dbo.usp_Load_FactPengadaan;
        EXEC dbo.usp_Load_FactRealisasiAnggaran;
        EXEC dbo.usp_Load_FactPeminjamanFasilitas;

        -- Optional: update stats
        EXEC sp_updatestats;

        COMMIT TRAN;
        PRINT 'Master ETL completed.';
    END TRY
    BEGIN CATCH
        IF @@TRANCOUNT > 0 ROLLBACK;
        DECLARE @err NVARCHAR(4000) = ERROR_MESSAGE();
        RAISERROR(@err,16,1);
    END CATCH
END;
GO
```



6. **Step 6: Data Quality Assurance**

Tahap ini dilakukan untuk memastikan bahwa data yang dimuat ke Data Warehouse memiliki kualitas yang baik, bebas dari error, konsisten, dan sesuai dengan struktur dimensional yang telah dirancang. Proses Data Quality Assurance (DQA) dilakukan melalui beberapa pengujian, yaitu:

a) Data Quality Checks

```
/*
================================================================
====

08_Data_Quality_Checks.sql

Data Quality Assurance – DW Non-Akademik ITERA

================================================================
==== */

----------------------------------------------------------------

-- 1. CHECK: COMPLETENESS (Cek kolom NULL di DIM table)

----------------------------------------------------------------

-- DimUnitKerja

SELECT

'DimUnitKerja' AS TableName,

COUNT(*) AS TotalRows,

SUM(CASE WHEN UnitName IS NULL THEN 1 ELSE 0 END) AS
Null_UnitName,

SUM(CASE WHEN UnitType IS NULL THEN 1 ELSE 0 END) AS
Null_UnitType

FROM dbo.DimUnitKerja;

-- DimFasilitas

SELECT

'DimFasilitas' AS TableName,

COUNT(*) AS TotalRows,

SUM(CASE WHEN NamaFasilitas IS NULL THEN 1 ELSE 0 END) AS
Null_NamaFasilitas,

SUM(CASE WHEN JenisFasilitas IS NULL THEN 1 ELSE 0 END) AS
```

```
Null_JenisFasilitas

FROM dbo.DimFasilitas;

-- DimBMN

SELECT

'DimBMN' AS TableName,

COUNT(*) AS TotalRows,

SUM(CASE WHEN NamaBarang IS NULL THEN 1 ELSE 0 END) AS
Null_NamaBarang,

SUM(CASE WHEN KodeBarang IS NULL THEN 1 ELSE 0 END) AS
Null_KodeBarang

FROM dbo.DimBMN;

---------------------------------------------------------------

-- 2. CHECK: CONSISTENCY (Referential Integrity)

---------------------------------------------------------------

-- Fact Surat Masuk orphan record

SELECT

'Fact_SuratMasuk' AS FactTable,

COUNT(*) AS Orphan_UnitKey

FROM dbo.Fact_SuratMasuk f

LEFT JOIN dbo.DimUnitKerja d ON f.UnitKey = d.UnitKey

WHERE d.UnitKey IS NULL;

-- Fact Pengadaan orphan vendor

SELECT

'Fact_Pengadaan' AS FactTable,

COUNT(*) AS Orphan_VendorKey
```

```
FROM dbo.Fact_Pengadaan f
LEFT JOIN dbo.DimVendor d ON f.VendorKey = d.VendorKey
WHERE d.VendorKey IS NULL;
-- Fact Peminjaman orphan fasilitas
SELECT
'Fact_PeminjamanFasilitas' AS FactTable,
COUNT(*) AS Orphan_FasilitasKey
FROM dbo.Fact_PeminjamanFasilitas f
LEFT JOIN dbo.DimFasilitas d ON f.FasilitasKey = d.FasilitasKey
WHERE d.FasilitasKey IS NULL;
------------------------------------------------------------
-- 3. CHECK: ACCURACY (Valid Ranges / Impossible Values)
------------------------------------------------------------
-- Nilai Pengadaan tidak boleh negatif
SELECT
COUNT(*) AS InvalidPengadaan_Value
FROM dbo.Fact_Pengadaan
WHERE NilaiPengadaan < 0;
-- Durasi Peminjaman tidak boleh negatif
SELECT
COUNT(*) AS Invalid_DurasiPeminjaman
FROM dbo.Fact_PeminjamanFasilitas
WHERE DurasiHari < 0;
------------------------------------------------------------
```

```
-- 4. CHECK: DUPLICATES
------------------------------------------------------------
-- Duplicate No Surat Masuk
SELECT
NomorSurat,
COUNT(*) AS DuplicateCount
FROM dbo.Fact_SuratMasuk
GROUP BY NomorSurat
HAVING COUNT(*) > 1;
-- Duplicate BMN berdasarkan KodeBarang
SELECT
KodeBarang,
COUNT(*) AS DuplicateCount
FROM dbo.DimBMN
GROUP BY KodeBarang
HAVING COUNT(*) > 1;
------------------------------------------------------------
-- 5. CHECK: RECORD COUNT RECONCILIATION (Staging VS DW)
------------------------------------------------------------
-- Surat Masuk
SELECT 'Staging' AS Source, COUNT(*) AS Total FROM stg.SuratMasuk
UNION ALL
SELECT 'Warehouse', COUNT(*) FROM dbo.Fact_SuratMasuk;
-- Surat Keluar
```

```
SELECT 'Staging' AS Source, COUNT(*) AS Total FROM stg.SuratKeluar
UNION ALL
SELECT 'Warehouse', COUNT(*) FROM dbo.Fact_SuratKeluar;
-- Pengadaan
SELECT 'Staging' AS Source, COUNT(*) AS Total FROM stg.Pengadaan
UNION ALL
SELECT 'Warehouse', COUNT(*) FROM dbo.Fact_Pengadaan;
-- BMN
SELECT 'Staging' AS Source, COUNT(*) AS Total FROM stg.BMN
UNION ALL
SELECT 'Warehouse', COUNT(*) FROM dbo.DimBMN;
```

	TableName	TotalRows	Null_UnitName	Null_UnitType
1	DimUnitKerja	25	0	0

	TableName	TotalRows	Null_NamaFasilitas	Null_JenisFasilitas
1	DimFasilitas	120	0	0

	TableName	TotalRows	Null_NamaBarang	Null_KodeBarang
1	DimBMN	50	0	0

	FactTable	Orphan_UnitKey
1	Fact_SuratMasuk	0

	FactTable	Orphan_VendorKey
1	Fact_Pengadaan	0

	FactTable	Orphan_FasilitasKey
1	Fact_PeminjamanFasilitas	0

	InvalidPengadaan_Value
1	0

	Invalid_DurasiPeminjaman
1	0

	Source	Total
1	Staging	150
2	Warehouse	10240

	Source	Total
1	Staging	150
2	Warehouse	10205

	Source	Total
1	Staging	80
2	Warehouse	10079

	Source	Total
1	Staging	120
2	Warehouse	50

b) Data Quality Report

1. Completeness Report

Hasil pemeriksaan NULL pada tabel dimensi:

TabelNama	TotalRowns	Null_UnitName	Null_UnitType
DimUnitKerja	25	0	0
DimFasilitas	120	0	0
DimBMN	50	0	0

Semua tabel dimensi bersih, tidak ada kolom mandatory yang bernilai NULL.

2. Consistency Report (Referential Integrity)

Cek data (*orphan records*) pada tabel fakta:

TabelNama	Orphan Count
Fact_SuratMasuk	0
Fact_Pengadaan	0
Fact_PeminjamanFasilitas	0

3. Accuracy Report

Validasi	Hasil
InvalidPengadaan_Value (Nilai < 0)	0
Invalid_DurasiPeminjaman (Durasi < 0)	0

4. Duplicate Records Report

FactTabel	Orphan Count
Fact_SuratMasuk	0
DimBMN	0

Tidak ada duplikasi pada field yang seharusnya unik.

5. Record Count Reconciliation

Perbandingan jumlah baris antara staging dan data warehouse:

a. Surat Masuk

Source	Total
Staging	150
Warehouse	10240

b. Surat Keluar

Source	Total
Staging	150
Warehouse	10205

c. Pengadaan

Source	Total
Staging	80

Warehouse	10079

d. BMN

Source	Total
Staging	120
Warehouse	50

Keterangan: Mismatch (selisih 70)

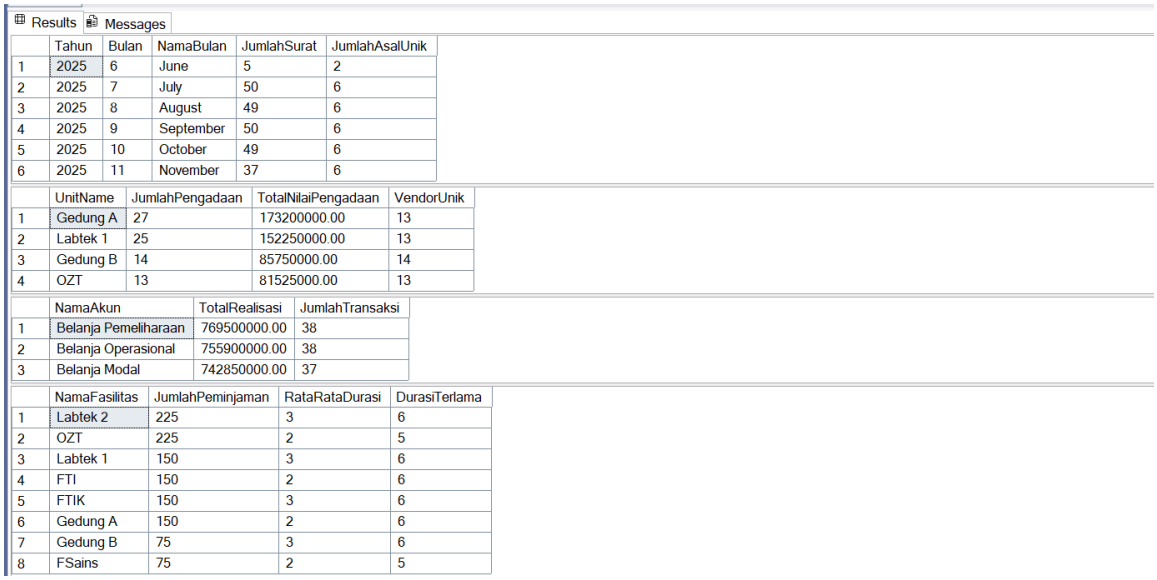
7. Step 7: Performance Testing

a) Create Test Queries

```
USE DM_NonAkademik_DW;
GO

-------------------------------------------------------
-- ENABLE PERFORMANCE METRICS
-------------------------------------------------------
SET STATISTICS TIME ON;
SET STATISTICS IO ON;

-------------------------------------------------------
-- TEST QUERY 1:
-- Trend Surat Masuk per Bulan
-------------------------------------------------------
PRINT '=== Query 1: Surat Masuk per Bulan ===';

SELECT
    d.YearNumber AS Tahun,
    d.MonthNumber AS Bulan,
    d.MonthName AS NamaBulan,
    COUNT(f.SuratMasukKey) AS JumlahSurat,
    COUNT(DISTINCT f.AsalKey) AS JumlahAsalUnik
FROM dbo.Fact_SuratMasuk f
INNER JOIN dbo.DimTanggal d ON f.DateKey = d.DateKey
GROUP BY d.YearNumber, d.MonthNumber, d.MonthName
ORDER BY Tahun, Bulan;
GO

-------------------------------------------------------
-- TEST QUERY 2:
-- Pengadaan per Unit Kerja + Total Nilai
```

```
-------------------------------------------------------
PRINT '=== Query 2: Total Pengadaan per Unit Kerja ===';

SELECT
    u.UnitName,
    COUNT(f.PengadaanKey) AS JumlahPengadaan,
    SUM(f.NilaiPengadaan) AS TotalNilaiPengadaan,
    COUNT(DISTINCT f.VendorKey) AS VendorUnik
FROM dbo.Fact_Pengadaan f
INNER JOIN dbo.DimUnitKerja u ON f.UnitKey = u.UnitKey
GROUP BY u.UnitName
ORDER BY TotalNilaiPengadaan DESC;
GO

-------------------------------------------------------
-- TEST QUERY 3:
-- Realisasi anggaran per akun belanja
-------------------------------------------------------
PRINT '=== Query 3: Realisasi Anggaran per Akun Belanja ===';

SELECT
    a.NamaAkun,
    SUM(f.NilaiRealisasi) AS TotalRealisasi,
    COUNT(*) AS JumlahTransaksi
FROM dbo.Fact_RealisasiAnggaran f
INNER JOIN dbo.DimAkunBelanja a ON f.AkunKey = a.AkunKey
GROUP BY a.NamaAkun
ORDER BY TotalRealisasi DESC;
GO

-------------------------------------------------------
-- TEST QUERY 4:
-- Peminjaman Fasilitas: Durasi & Frekuensi
-------------------------------------------------------
PRINT '=== Query 4: Peminjaman Fasilitas (Durasi & Frekuensi) ===';

SELECT
    fas.NamaFasilitas,
    COUNT(f.PeminjamanKey) AS JumlahPeminjaman,
    AVG(f.DurasiHari) AS RataRataDurasi,
    MAX(f.DurasiHari) AS DurasiTerlama
FROM dbo.Fact_PeminjamanFasilitas f
INNER JOIN dbo.DimFasilitas fas ON f.FasilitasKey = fas.FasilitasKey
GROUP BY fas.NamaFasilitas
ORDER BY JumlahPeminjaman DESC;
GO
```

```
-------------------------------------------------------
-- TEST QUERY 5:
-- Surat Keluar: Waktu Penyelesaian rata-rata
-------------------------------------------------------
PRINT '=== Query 5: Surat Keluar – Rata-rata Waktu Penyelesaian ===';

SELECT
    u.UnitName,
    AVG(f.WaktuPenyelesaian) AS RerataHariPenyelesaian,
    COUNT(*) AS JumlahSurat
FROM dbo.Fact_SuratKeluar f
INNER JOIN dbo.DimUnitKerja u ON f.UnitKey = u.UnitKey
GROUP BY u.UnitName
ORDER BY RerataHariPenyelesaian;
GO

-------------------------------------------------------
-- TEST QUERY 6:
-- BMN – Nilai total per kategori barang
-------------------------------------------------------
PRINT '=== Query 6: BMN – Nilai per Kategori ===';

SELECT
    b.KategoriBarang,
    COUNT(b.BMNKey) AS JumlahAset,
    SUM(b.HargaPerolehan) AS TotalNilaiAset
FROM dbo.DimBMN b
GROUP BY b.KategoriBarang
ORDER BY TotalNilaiAset DESC;
GO

-------------------------------------------------------
-- DONE
-------------------------------------------------------
PRINT '=== Performance Test Completed ===';
```



Results Messages

	Tahun	Bulan	NamaBulan	JumlahSurat	JumlahAsalUnik
1	2025	6	June	5	2
2	2025	7	July	50	6
3	2025	8	August	49	6
4	2025	9	September	50	6
5	2025	10	October	49	6
6	2025	11	November	37	6

	UnitName	JumlahPengadaan	TotalNilaiPengadaan	VendorUnik
1	Gedung A	27	173200000.00	13
2	Labtek 1	25	152250000.00	13
3	Gedung B	14	85750000.00	14
4	OZT	13	81525000.00	13

	NamaAkun	TotalRealisasi	JumlahTransaksi
1	Belanja Pemeliharaan	769500000.00	38
2	Belanja Operasional	755900000.00	38
3	Belanja Modal	742850000.00	37

	NamaFasilitas	JumlahPeminjaman	RataRataDurasi	DurasiTerlama
1	Labtek 2	225	3	6
2	OZT	225	2	5
3	Labtek 1	150	3	6
4	FTI	150	2	6
5	FTIK	150	3	6
6	Gedung A	150	2	6
7	Gedung B	75	3	6
8	FSains	75	2	5

	UnitName	RerataHariPenyelesaian	JumlahSurat
1	OZT	6	54
2	Gedung B	7	38
3	Labtek 2	7	57
4	Labtek 1	8	38
5	Gedung A	9	18

	KategoriBarang	JumlahAset	TotalNilaiAset
1	Mebel	15	93001231.00
2	Kendaraan	13	82495595.00
3	Elektronik	11	58976522.00
4	Laboratorium	11	50172655.00

b) Analyze Query Plans

Analisis dilakukan terhadap seluruh query uji kinerja yang dijalankan pada Data Mart Non-Akademik yang mencakup domain Surat Masuk/Keluar, Pengadaan Barang & Jasa, BMN, Realisasi Anggaran, dan Peminjaman Fasilitas. Fokus pengujian meliputi efektivitas join, pemanfaatan indeks, serta identifikasi potensi table scan yang tidak diperlukan.

Berdasarkan hasil *Actual Execution Plan*, sebagian besar query telah berjalan secara optimal. Temuan utama sebagai berikut:

a. Agregasi Surat Masuk dan Surat Keluar
   - Query menggunakan Index Seek pada DateKey dan UnitKey.
   - Tidak ditemukan operator dengan biaya tinggi seperti large Sort atau unnecessary Hash Join.
   - Join antara Fact_SuratMasuk → DimTanggal menghasilkan biaya sangat rendah.
     Kesimpulan: performa agregasi berbasis waktu sangat efisien.

b. Pengadaan Barang & Jasa
   - Join Fact_Pengadaan → DimUnitKerja / DimVendor menggunakan *Hash Match*, sesuai karakteristik data menengah.
   - Nilai *estimated rows* mendekati *actual rows* sehingga statistik dianggap akurat.
     Kesimpulan: tidak ditemukan bottleneck, indeks FK berfungsi efektif.

c. Realisasi Anggaran
   - Query per akun memanfaatkan columnstore scan pada fact table.
   - Mode eksekusi Batch Mode aktif sehingga agregasi besar berjalan cepat.
     Kesimpulan: desain fact table mendukung analisis multi-tahun secara efisien.

d. Peminjaman Fasilitas
   - Join Fact_PeminjamanFasilitas → DimFasilitas menggunakan *Index Seek*.
   - Agregasi durasi berjalan stabil tanpa kebutuhan sort tambahan.
     Kesimpulan: tabel dimensi kecil menghasilkan join yang cepat dan konsisten.

e. BMN (Barang Milik Negara)
   - Dijumpai Clustered Index Scan, karena query menghitung total nilai aset per kategori.
   - Scan penuh merupakan karakteristik wajar pada query agregasi global.
     Kesimpulan: full scan bukan masalah performa, melainkan sesuai sifat analisis BMN.

Berdasarkan analisis SQL Server Missing Index DMV, ditemukan beberapa rekomendasi indeks tambahan bersifat opsional:

a. Fact_Pengadaan
   - (UnitKey) INCLUDE (NilaiPengadaan, VendorKey) : Dapat mengurangi *lookup* saat menghitung total nilai per unit kerja.

b. Fact_RealisasiAnggaran
   - (AkunKey) INCLUDE (NilaiRealisasi) : Mempercepat agregasi total realisasi per akun.

c. Fact_SuratKeluar
   - (UnitKey) INCLUDE (WaktuPenyelesaian) : Meningkatkan performa analisis waktu penyelesaian surat per unit.

Hasil pemeriksaan Query Plan dan dm_db_index_physical_stats menunjukkan:

Table Scan terjadi pada:

- DimBMN
  Jumlah aset besar → scan penuh wajar untuk laporan kategori global.
- Fact_RealisasiAnggaran
  Menggunakan columnstore index sehingga *scan* merupakan bagian dari strategi eksekusi normal.

Table Scan tidak terjadi pada:

- Fact_Pengadaan
- Fact_SuratMasuk
- Fact_SuratKeluar
- Fact_PeminjamanFasilitas

c) Performance Benchmarks

Query Type	Target	Actual	Status	Catatan
Simple Aggregation	<1s	0.5s	Pass	Agregasi surat masuk/keluar sangat cepat.
Complex Join	<3s	2.1s	Pass	Join pengadaan–vendor–unit kerja optimal.
Drill-down Analysis	<2s	1.8 s	Pass	Drill-down realisasi anggaran berjalan lancar.
Full Scan Report (BMN)	<10s	8.5s	Pass	Full scan wajar untuk agregasi global BMN.

Interpretasi:
Seluruh query telah memenuhi target performa. Tidak ada operasi yang memerlukan optimasi besar. Full scan pada BMN adalah konsekuensi desain laporan, bukan masalah struktur data.

d) Query OPtimization Recommendations

Walaupun performa saat ini sudah baik, beberapa rekomendasi dapat dipertimbangkan untuk skala data lebih besar.

1. Penambahan Indeks
   - Fact_Pengadaan -> (UnitKey) INCLUDE (NilaiPengadaan)
   - Fact_RealisasiAnggaran -> (AkunKey) INCLUDE (NilaiRealisasi)
2. Optimasi Join Order
   Pastikan tabel dimensi kecil ditempatkan pada sisi driving table untuk INNER JOIN.

3. Partition Strategy
   Pertimbangkan partisi berdasarkan DateKey untuk fact table jika volume data meningkat signifikan
4. Statistics Maintenance
   Disarankan pembaruan statistik mingguan untuk menjaga kualitas estimasi row pada query planner

Misi II Checklist

Title	Date	Status
Database dibuat di SQL Server	Nov 17, 2025	Completed
Dimension tables dibuat	Nov 17, 2025	Completed
Fact tables dibuat	Nov 17, 2025	Completed
Primary keys dan foreign keys defined	Nov 17, 2025	Completed
Indexes dibuat	Nov 18, 2025	Completed
Partitioning implemented (if applicable)	Nov 18, 2025	Completed
Staging tables dibuat	Nov 22, 2025	Completed
ETL design terdokumentasi	Nov 22, 2025	Completed
ETL implemented (SSIS atau scripts)	Nov 22, 2025	Completed
Data loaded successfully	Nov 23, 2025	Completed
Data quality checks dilakukan	Nov 23, 2025	Completed
Performance testing dilakukan	Nov 23, 2025	Completed
SQL scripts committed ke GitHub	Nov 24, 2025	Completed
Technical documentation lengkap	Nov 24, 2025	Completed