TUGAS BESAR PERGUDANGAN DATA
PEMBANGUNAN DATA MART

Domain: Satuan Pengawas Internal

Anggota Kelompok:

1.	Anggi Puspita Ningrum	123450012
2.	Anadia Carana	123450019
3.	Iqfina Haula Halika	123450076
4.	Muhammad Dzikra	123450124

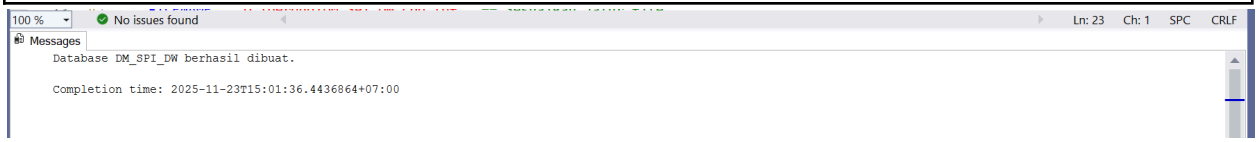
PROGRAM STUDI SAINS DATA
FAKULTAS SAINS
INSTITUT TEKNOLOGI SUMATERA
2025

MISI 2: DESAIN FISIKAL DAN DEVELOPMENT

2.1. Physical Database Design

1. Database Setup

```
CREATE DATABASE DM_SPI_DW
ON PRIMARY
(
   NAME = DM_SPI_DW_Data,
   FILENAME = 'C:\Data\DM_SPI_DW_Data.mdf', -- Sesuaikan jalur file
   SIZE = 50MB,
   MAXSIZE = UNLIMITED,
   FILEGROWTH = 10MB
)
LOG ON
(
   NAME = DM_SPI_DW_Log,
   FILENAME = 'C:\Data\DM_SPI_DW_Log.ldf', -- Sesuaikan jalur file
   SIZE = 10MB,
   MAXSIZE = 2048GB,
   FILEGROWTH = 10%
);
GO

PRINT 'Database DM_SPI_DW berhasil dibuat.'

USE DM_SPI_DW;
GO
```



Kueri SQL tersebut membuat basis data baru bernama DM_SPI_DW dengan konfigurasi file data dan log spesifik (lokasi D:\sms 5\gudang data, ukuran awal 50MB/10MB, dan pengaturan pertumbuhan otomatis). Setelah berhasil dibuat, basis data ini langsung diatur sebagai basis data aktif (USE DM_SPI_DW) untuk dimulainya proses perancangan dan implementasi gudang data.

2. Create Dimension Tables

```
-- 1. Dimensi Waktu (SCD Tipe 0 / Static) - Disesuaikan dengan Dim_Date standar
CREATE TABLE dbo.Dim_Waktu (
   Waktu_SK INT PRIMARY KEY NOT NULL, -- PK (YYYYMMDD)
   Tanggal_Penuh DATE NOT NULL,
   Hari_Ke_Minggu TINYINT NOT NULL,
   Nama_Hari VARCHAR (10) NOT NULL,
   Hari_Ke_Bulan TINYINT NOT NULL,
   Hari_Ke_Tahun SMALLINT NOT NULL,
   Minggu_Ke_Tahun TINYINT NOT NULL,
```

```
    Nama_Bulan VARCHAR (10) NOT NULL,
    Bulan_Number TINYINT NOT NULL,
    Kuartal TINYINT NOT NULL,
    Nama_Kuartal VARCHAR (6) NOT NULL,
    Tahun SMALLINT NOT NULL,
    Periode_Fiskal VARCHAR(10),
    Is_Akhir_Pekan BIT NOT NULL,
    Is_Libur BIT NOT NULL,

    -- Metadata
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO
```

```
-- Index pada Tahun (untuk memfilter laporan tahunan)
CREATE NONCLUSTERED INDEX IX_DimWaktu_Tahun ON dbo.Dim_Waktu (Tahun);
GO
```

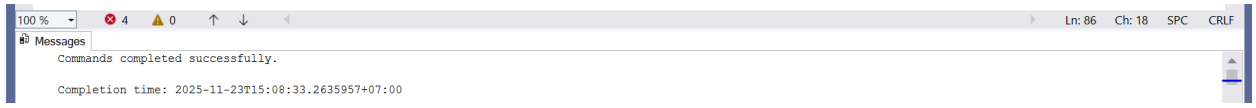
```
-- 2. Dimensi Sistem Sumber (SCD Tipe 0 / Static)
CREATE TABLE dbo.Dim_Sistem_Sumber (
    ID_Sistem_Sumber INT PRIMARY KEY NOT NULL, -- PK
    Kode_Sumber VARCHAR(50) NOT NULL,
    Nama_Sistem VARCHAR(100),
    Deskripsi VARCHAR(255),
    Penanggung_Jawab VARCHAR(100),
    Tanggal_Mulai_Berlaku DATE,

    -- Metadata
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO
```

```
-- Indexing Natural Key
CREATE UNIQUE NONCLUSTERED INDEX IX_DimSistemSumber_NK ON dbo.Dim_Sistem_Sumber
(Kode_Sumber);
GO
```

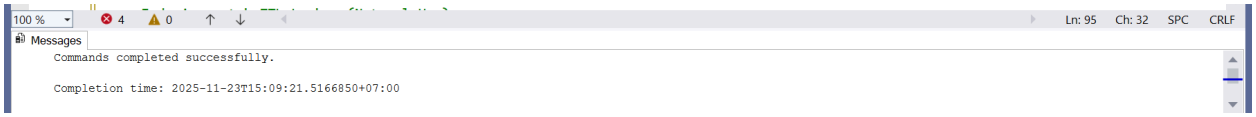

```
-- 3. Dimensi Unit Kerja (SCD Tipe 2) - Objek Audit
-- Pemicu SCD: Jenis_Unit, Kepala_Unit
CREATE TABLE dbo.Dim_Unit_Kerja (
    Unit_Kerja_SK INT IDENTITY(1,1) PRIMARY KEY NOT NULL, -- Surrogate Key
```

```
    Kode_Unit VARCHAR(50) UNIQUE NOT NULL, -- Natural Key (Source Key)
    Nama_Unit VARCHAR(100) NOT NULL,
    Jenis_Unit VARCHAR(50),
    Kepala_Unit VARCHAR(100),

    -- Kolom SCD Type 2
    EffectiveDate DATE DEFAULT GETDATE() NOT NULL,
    ExpiryDate DATE NULL,
    IsCurrent BIT DEFAULT 1 NOT NULL,

    -- Metadata
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO
```


```
-- Indexing untuk ETL Lookup dan SCD Tipe 2
CREATE NONCLUSTERED INDEX IX_DimUnitKerja_NK ON dbo.Dim_Unit_Kerja (Kode_Unit);
CREATE NONCLUSTERED INDEX IX_DimUnitKerja_Current ON dbo.Dim_Unit_Kerja (IsCurrent) WHERE IsCurrent = 1;
GO
```

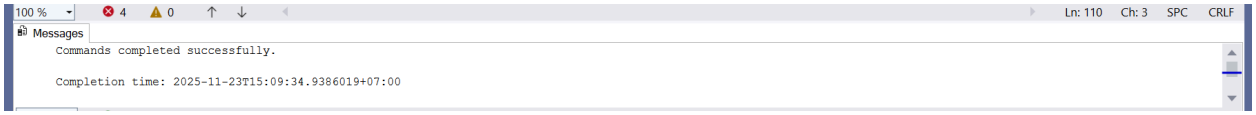



```
-- 4. Dimensi Siklus Audit (SCD Tipe 1)
CREATE TABLE dbo.Dim_Siklus_Audit (
    Siklus_Audit_SK INT IDENTITY(1,1) PRIMARY KEY NOT NULL, -- Surrogate Key
    ID_Sistem_Sumber VARCHAR(50) UNIQUE NOT NULL, -- Source Key
    Jenis_Audit VARCHAR(50),
    Tahun_Siklus INT,
    Status_Siklus VARCHAR(50),

    -- Metadata
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO
```

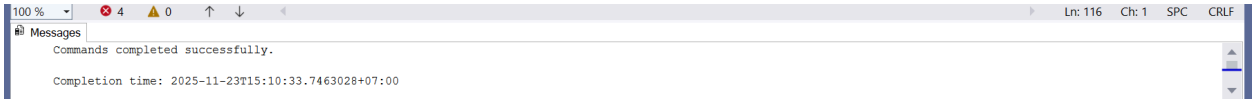



```
-- Indexing untuk ETL Lookup (Natural Key)
```

```
CREATE UNIQUE NONCLUSTERED INDEX IX_DimSiklusAudit_NK ON dbo.Dim_Siklus_Audit
(ID_Sistem_Sumber);
GO
```



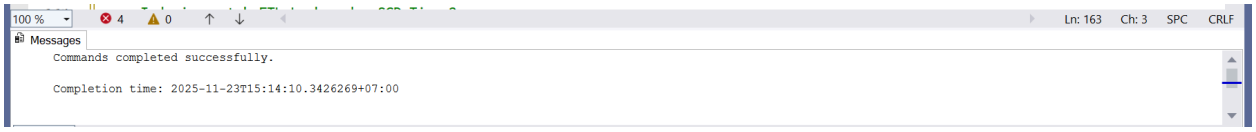

```
-- 5. Dimensi Auditor (SCD Tipe 2 - Disesuaikan dengan kebutuhan SPI)
-- Pemicu SCD: Jabatan, Tim_Audit
CREATE TABLE dbo.Dim_Auditor (
    Auditor_SK INT IDENTITY(1,1) PRIMARY KEY NOT NULL, -- Surrogate Key
    ID_Sistem_Sumber VARCHAR(50) UNIQUE NOT NULL, -- Natural Key (NIP/ID Asli)
    Nama_Auditor VARCHAR(100) NOT NULL,
    Bidang_Keahlian VARCHAR(50), -- Tipe 1
    Jabatan VARCHAR(50), -- SCD Tipe 2
    Status_Keanggotaan VARCHAR(50), -- Tipe 1
    Tim_Audit VARCHAR(50), -- SCD Tipe 2

    -- Kolom SCD Type 2
    Tanggal_Berlaku_Auditor DATE DEFAULT GETDATE() NOT NULL,
    Tanggal_Berakhir_Auditor DATE NULL,
    IsCurrent BIT DEFAULT 1 NOT NULL,

    -- Metadata
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO
```




```
-- Indexing untuk ETL Lookup dan SCD Tipe 2
CREATE NONCLUSTERED INDEX IX_DimAuditor_NK ON dbo.Dim_Auditor (ID_Sistem_Sumber);
CREATE NONCLUSTERED INDEX IX_DimAuditor_Current ON dbo.Dim_Auditor (IsCurrent) WHERE
IsCurrent = 1;
GO
```

```
-- 6. Dimensi Temuan (SCD Tipe 2)
-- Pemicu SCD: Tingkat_Materialitas
CREATE TABLE dbo.Dim_Temuan (
    Temuan_SK INT IDENTITY(1,1) PRIMARY KEY NOT NULL,
    ID_Sistem_Sumber VARCHAR(50) UNIQUE NOT NULL, -- Source Key
    Kategori_Risiko VARCHAR(50),
    Tingkat_Materialitas VARCHAR(50), -- SCD Tipe 2
    Deskripsi_Temuan VARCHAR(MAX),
    Kelemahan_Kontrol VARCHAR(100),
```

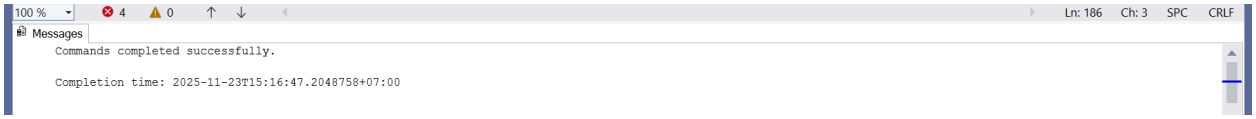
```
    -- Kolom SCD Type 2
    EffectiveDate DATE DEFAULT GETDATE() NOT NULL,
    ExpiryDate DATE NULL,
    IsCurrent BIT DEFAULT 1 NOT NULL,

    -- Metadata
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO
```




```
-- Indexing untuk ETL Lookup dan SCD Tipe 2
CREATE NONCLUSTERED INDEX IX_DimTemuan_NK ON dbo.Dim_Temuan (ID_Sistem_Sumber);
CREATE NONCLUSTERED INDEX IX_DimTemuan_Current ON dbo.Dim_Temuan (IsCurrent) WHERE
IsCurrent = 1;
GO
```


```
-- 7. Dimensi Rekomendasi (SCD Tipe 2)
-- Pemicu SCD: Penanggung_Jawab
CREATE TABLE dbo.Dim_Rekomendasi (
    Rekomendasi_SK INT IDENTITY(1,1) PRIMARY KEY NOT NULL,
    ID_Sistem_Sumber VARCHAR(50) UNIQUE NOT NULL, -- Source Key
    Status_Tindak_Lanjut VARCHAR(50), -- Tipe 1
    Tanggal_Target_Selesai DATE, -- Tipe 1
    Penanggung_Jawab VARCHAR(100), -- SCD Tipe 2

    -- Kolom SCD Type 2
    EffectiveDate DATE DEFAULT GETDATE() NOT NULL,
    ExpiryDate DATE NULL,
    IsCurrent BIT DEFAULT 1 NOT NULL,

    -- Metadata
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
GO
```

```
-- Indexing untuk ETL Lookup dan SCD Tipe 2
CREATE NONCLUSTERED INDEX IX_DimRekomendasi_NK ON dbo.Dim_Rekomendasi
(ID_Sistem_Sumber);
CREATE NONCLUSTERED INDEX IX_DimRekomendasi_Current ON dbo.Dim_Rekomendasi (IsCurrent)
WHERE IsCurrent = 1;
GO
```



Serangkaian kueri CREATE TABLE ini bertujuan untuk membangun struktur dimensi dasar (Dimensi Table) yang akan digunakan dalam skema bintang (Star Schema) untuk analisis pengawasan internal (SPI). Setiap dimensi dirancang untuk menyimpan atribut deskriptif dan diklasifikasikan berdasarkan cara penanganan perubahannya, yang dikenal sebagai Slowly Changing Dimension (SCD):

- Dimensi Statis (SCD Tipe 0):
    - Dim_Waktu: Menyimpan semua detail kalender (tahun, bulan, hari, akhir pekan) menggunakan Waktu_SK sebagai *Surrogate Key* berbasis format tanggal (YYYYMMDD).
- Dimensi Bertipe 1 (SCD Tipe 1):
    - Dim_Siklus_Audit: Perubahan pada atribut seperti Jenis_Audit atau Status_Siklus akan menimpa data lama.
- Dimensi Bertipe 2 (SCD Tipe 2):
    - Dimensi ini mencakup kolom seperti EffectiveDate, ExpiryDate, dan IsCurrent untuk melacak riwayat perubahan atribut tertentu, seperti:
        - Dim_Unit_Kerja: Melacak riwayat perubahan pada Jenis_Unit dan Kepala_Unit.
        - Dim_Auditor: Melacak riwayat perubahan pada Jabatan dan Tim_Audit.
        - Dim_Temuan: Melacak riwayat perubahan pada Tingkat_Materialitas temuan.
        - Dim_Rekomendasi: Melacak riwayat perubahan pada Penanggung_Jawab rekomendasi.

Secara keseluruhan, kueri ini menciptakan fondasi struktural yang diperlukan untuk menyimpan dan menganalisis data SPI, dengan mempertimbangkan bagaimana atribut-atribut kunci berubah seiring waktu. Selain itu, indeks non-klaster dibuat pada *Natural Key* (NK) dan kolom IsCurrent untuk mengoptimalkan kinerja operasi *Extract, Transform, Load* (ETL) dan pelaporan.

3. Create Fact Tables

```
-- Grain: Satu baris per Temuan Audit per Rekomendasi terkait.
CREATE TABLE dbo.Fact_Temuan_Rekomendasi (
   Fakta_SK BIGINT IDENTITY(1,1) NOT NULL, -- Primary Key Tunggal

   -- Foreign Keys
   Waktu_SK INT NOT NULL,
   Auditor_SK INT NOT NULL,
```

```
    Unit_Kerja_SK INT NOT NULL,
    Siklus_Audit_SK INT NOT NULL,
    Temuan_SK INT NOT NULL,
    Rekomendasi_SK INT NOT NULL,
    Sistem_Sumber_SK INT NOT NULL,

    -- Degenerate Dimension & Partitioning Key
    Tahun_Audit INT NOT NULL, -- Ditambahkan untuk mendukung Partitioning yang spesifik

    -- Measures (Sesuai Data Dictionary Fakta)
    Jumlah_Temuan INT DEFAULT 1,
    Skor_Risiko_Temuan DECIMAL(5,2),
    Potensi_Kerugian_IDR BIGINT DEFAULT 0,
    Usia_Rekomendasi_Hari INT,

    -- Metadata Audit
    LoadDate DATETIME DEFAULT GETDATE(),

    -- Constraints
    CONSTRAINT PK_Fact_Temuan_Rekomendasi PRIMARY KEY CLUSTERED (Fakta_SK),

    -- Membuat Relasi (Constraint Foreign Key)
    CONSTRAINT FK_Fact_Waktu FOREIGN KEY (Waktu_SK) REFERENCES dbo.Dim_Waktu(Waktu_SK),
    CONSTRAINT FK_Fact_Auditor FOREIGN KEY (Auditor_SK) REFERENCES
dbo.Dim_Auditor(Auditor_SK),
    CONSTRAINT FK_Fact_Unit FOREIGN KEY (Unit_Kerja_SK) REFERENCES
dbo.Dim_Unit_Kerja(Unit_Kerja_SK),
    CONSTRAINT FK_Fact_Siklus FOREIGN KEY (Siklus_Audit_SK) REFERENCES
dbo.Dim_Siklus_Audit(Siklus_Audit_SK),
    CONSTRAINT FK_Fact_Temuan FOREIGN KEY (Temuan_SK) REFERENCES
dbo.Dim_Temuan(Temuan_SK),
    CONSTRAINT FK_Fact_Rekomendasi FOREIGN KEY (Rekomendasi_SK) REFERENCES
dbo.Dim_Rekomendasi(Rekomendasi_SK),
    CONSTRAINT FK_Fact_SistemSumber FOREIGN KEY (Sistem_Sumber_SK) REFERENCES
dbo.Dim_Sistem_Sumber(ID_Sistem_Sumber)
)
ON PS_AuditYear (Tahun_Audit);
GO
```

Kueri SQL ini membuat tabel fakta utama bernama Fact_Temuan_Rekomendasi yang berfungsi sebagai inti dari skema bintang.

- Granularitas (Grain): Setiap baris mewakili satu hubungan spesifik antara Temuan Audit dengan Rekomendasi terkait.
- Kunci (Keys): Tabel ini menghubungkan ke tujuh dimensi (Waktu_SK, Auditor_SK, Unit_Kerja_SK, Siklus_Audit_SK, Temuan_SK, Rekomendasi_SK, Sistem_Sumber_SK) melalui *Foreign Keys*.

- Ukuran (Measures): Menyimpan nilai-nilai kuantitatif untuk dianalisis, seperti Jumlah_Temuan, Skor_Risiko_Temuan, Potensi_Kerugian_IDR, dan Usia_Rekomendasi_Hari.
- Partitioning: Tabel ini dioptimalkan untuk kinerja dengan menggunakan Tahun_Audit sebagai *Partitioning Key* pada skema partisi PS_AuditYear, ideal untuk pelaporan yang sering memfilter berdasarkan tahun.

2.2. Indexing Strategy

1. Clustered Index on Fact Table

```
-- Mengganti Clustered Index PK_Fact_Temuan_Rekomendasi ke Fact Key & Waktu Key
-- (Waktu_SK adalah kunci utama untuk query DW)
DROP INDEX PK_Fact_Temuan_Rekomendasi ON dbo.Fact_Temuan_Rekomendasi; -- Drop PK Clustered default
CREATE CLUSTERED INDEX CIX_Fact_Temuan_Waktu
ON dbo.Fact_Temuan_Rekomendasi (Waktu_SK, Fakta_SK)
ON PS_AuditYear (Tahun_Audit); -- Index ini juga menggunakan Partition Scheme
GO
```

Kode ini melakukan perubahan kunci pengurutan fisik pada tabel fakta dbo.Fact_Temuan_Rekomendasi (melalui *Clustered Index*). Indeks default (PK_Fact_Temuan_Rekomendasi) dihapus dan diganti dengan *Clustered Index* baru bernama CIX_Fact_Temuan_Waktu. Indeks baru ini mengurutkan data secara fisik berdasarkan kolom Waktu_SK dan Fakta_SK untuk mengoptimalkan kinerja *query data warehouse* yang berfokus pada dimensi waktu. Selain itu, indeks ini juga mengimplementasikan partisi data berdasarkan Tahun_Audit (menggunakan skema PS_AuditYear) untuk pengelolaan dan *query* data yang lebih efisien berdasarkan tahun.

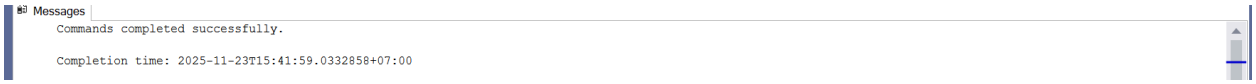
2. Non-Clustered Indexes

```
-- Index untuk Foreign Key Unit Kerja (Optimasi Join Analisis Profil Risiko Unit)
CREATE NONCLUSTERED INDEX IX_Fact_Unit_Key
ON dbo.Fact_Temuan_Rekomendasi (Unit_Kerja_SK)
INCLUDE (Skor_Risiko_Temuan, Potensi_Kerugian_IDR); -- Sering diakses bersama
GO
```


```
-- Index untuk Foreign Key Auditor (Optimasi Join Analisis Kinerja Auditor)
CREATE NONCLUSTERED INDEX IX_Fact_Auditor_Key
ON dbo.Fact_Temuan_Rekomendasi (Auditor_SK)
INCLUDE (Jumlah_Temuan, Usia_Rekomendasi_Hari); -- Sering diakses bersama
GO
```

```
-- Index untuk Foreign Key Temuan (Mendukung pelacakan temuan dan Rekomendasi)
CREATE NONCLUSTERED INDEX IX_Fact_Temuan_Rekomendasi_Keys
ON dbo.Fact_Temuan_Rekomendasi (Temuan_SK, Rekomendasi_SK);
GO
```


```
-- Covering Index untuk common queries (Misalnya, Laporan Tindak Lanjut per Siklus Audit)
CREATE NONCLUSTERED INDEX IX_Fact_Covering_Siklus
ON dbo.Fact_Temuan_Rekomendasi (Siklus_Audit_SK, Waktu_SK)
INCLUDE (Potensi_Kerugian_IDR, Usia_Rekomendasi_Hari);
GO
```

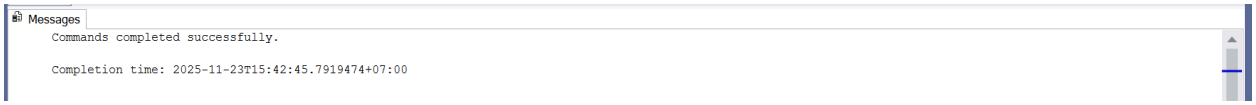


Rangkaian kode ini membuat empat *Nonclustered Index* pada tabel fakta dbo.Fact_Temuan_Rekomendasi untuk mengoptimalkan kinerja.

- Tiga Index FK (*Foreign Key*) (IX_Fact_Unit_Key, IX_Fact_Auditor_Key, IX_Fact_Temuan_Rekomendasi_Keys) mempercepat operasi JOIN ke tabel dimensi terkait (Unit Kerja, Auditor, Temuan/Rekomendasi) dan pengambilan data untuk analisis spesifik (Profil Risiko Unit, Kinerja Auditor). Dua index pertama bersifat Covering Index karena menyertakan kolom metrik yang sering diakses (seperti Skor_Risiko_Temuan dan Jumlah_Temuan).
- Satu Index Covering Utama (IX_Fact_Covering_Siklus) dibuat pada kolom Siklus_Audit_SK dan Waktu_SK dan menyertakan metrik utama (seperti Potensi_Kerugian_IDR). Tujuannya adalah memastikan *query* umum untuk laporan (misalnya, Laporan Tindak Lanjut) dapat mengambil semua data yang dibutuhkan langsung dari indeks tanpa perlu mengakses tabel fakta utama, sehingga sangat meningkatkan kecepatan.

3. Columnstore Index (for large fact tables)

```
-- Columnstore index untuk analytical queries (OLAP style)
CREATE NONCLUSTERED COLUMNSTORE INDEX NCCIX_Fact_Analytic
ON dbo.Fact_Temuan_Rekomendasi (
    Waktu_SK, Auditor_SK, Unit_Kerja_SK, Siklus_Audit_SK, Temuan_SK, Rekomendasi_SK,
    Tahun_Audit,
    Jumlah_Temuan, Skor_Risiko_Temuan, Potensi_Kerugian_IDR, Usia_Rekomendasi_Hari
)
WHERE Tahun_Audit > 2023; -- Fokus pada data terbaru untuk Analitik Cepat
GO
```



Kode ini membuat indeks analitik super cepat (*Columnstore*) yang dioptimalkan dan terkompresi tinggi, tetapi hanya diterapkan pada subset data terbaru (Tahun_Audit > 2023$), memastikan *dashboard* dan laporan analitik mendapatkan kinerja tercepat sementara data historis dikelola secara terpisah.

2.3. Partitioning Strategy

```
-- 1. Membuat Schema Staging (untuk ETL)
IF NOT EXISTS (SELECT * FROM sys.schemas WHERE name = 'stg')
BEGIN
   EXEC('CREATE SCHEMA stg')
END
GO
-- 2. Membuat Partition Function berdasarkan Tahun (untuk Fact Table)
-- Tujuan: Membagi data ke dalam grup berdasarkan Tahun Audit.
-- RANGE RIGHT: Boundary values adalah batas bawah partisi berikutnya.
CREATE PARTITION FUNCTION PF_AuditYear (INT)
AS RANGE RIGHT FOR VALUES (
   2020, -- Partisi 1: Tahun_Audit < 2020 (Historical)
   2021, -- Partisi 2: 2020 <= Tahun_Audit < 2021
   2022, -- Partisi 3: 2021 <= Tahun_Audit < 2022
   2023, -- Partisi 4: 2022 <= Tahun_Audit < 2023
   2024, -- Partisi 5: 2023 <= Tahun_Audit < 2024
   2025  -- Partisi 6: 2024 <= Tahun_Audit < 2025
);
GO
-- 3. Membuat Partition Scheme
-- Tujuan: Memetakan setiap partisi dari Partition Function ke Filegroup tertentu.
-- ALL TO ([PRIMARY]) berarti semua partisi disimpan di Filegroup utama (untuk lingkungan sederhana).
CREATE PARTITION SCHEME PS_AuditYear
AS PARTITION PF_AuditYear
ALL TO ([PRIMARY]);
GO
```

Kode ini melakukan tiga hal utama untuk menyiapkan *data warehouse*:

1. Membuat Schema Staging (stg): Menyiapkan area terpisah untuk tabel yang digunakan dalam proses ETL (Extract, Transform, Load), menjaga kebersihan lingkungan DW utama.
2. Membuat Partition Function (PF_AuditYear): Secara logis mendefinisikan batas pemisahan data pada tabel fakta (Fact_Temuan_Rekomendasi) berdasarkan kolom Tahun_Audit (misalnya, membagi data per tahun: 2021, 2022, 2023, dst.).
3. Membuat Partition Scheme (PS_AuditYear): Memetakan partisi logis tersebut ke lokasi fisik penyimpanan *database* (ALL TO [PRIMARY]), yang kemudian akan digunakan saat membuat tabel fakta, memungkinkan pengelolaan dan kinerja *query* yang lebih efisien untuk data historis dan saat ini.

2.4. ETL Design

1. ETL Architecture Design

Kami mengadopsi pendekatan ELT (Extract, Load, Transform) menggunakan T-SQL Stored Procedures.

1. Staging Tables (stg): Data mentah dimuat ke stg.Audit_Temuan_Rekom (dari sistem audit) dan stg.Auditor_HRIS (dari HRIS).

2. Transformation (T-SQL): Stored Procedures (usp_Load_Dim_* dan usp_Load_Fact_*) melakukan transformasi, *lookup* Surrogate Key, dan penerapan logika SCD Tipe 2.

2. Create Staging Tables

```
USE DM_SPI_DW;
GO

-- 1. Create Staging Schema
IF NOT EXISTS (SELECT * FROM sys.schemas WHERE name = 'stg')
BEGIN
   EXEC('CREATE SCHEMA stg')
END
GO
```

```
-- 2. Staging Table for Transaction Data (Temuan & Rekomendasi)
-- Sumber: Sistem Internal Audit
CREATE TABLE stg.Audit_Temuan_Rekom (
   -- Source IDs
   ID_Temuan_Sumber VARCHAR(50) ,
   ID_Rekom_Sumber VARCHAR(50) ,
   ID_Siklus_Sumber VARCHAR(50) ,
   ID_Unit_Sumber VARCHAR(50) ,
   ID_Auditor_Sumber VARCHAR(50) ,

   -- Transaction Attributes
   Tanggal_Temuan DATE ,
   Skor_Risiko DECIMAL(5,2) ,
   Kerugian_IDR BIGINT ,
   Status_Rekomendasi VARCHAR(50) ,
   Tanggal_Target_Selesai DATE ,

   -- Dimension Attributes (Snowflaked in Source)
   Kategori_Risiko_Temuan VARCHAR(50) ,
   Tingkat_Materialitas VARCHAR(50) ,
   Deskripsi_Temuan_Lengkap VARCHAR(MAX) ,
   Kelemahan_Kontrol VARCHAR(100) ,

   -- Metadata
   LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

```
-- 3. Staging Table for Auditor Master Data
-- Sumber: HRIS (Human Resource Information System)
CREATE TABLE stg.Auditor_HRIS (
   ID_Auditor_Sumber VARCHAR(50) ,
```

```
    Nama_Lengkap VARCHAR(100) ,
    Bidang_Keahlian_HR VARCHAR(50) ,
    Jabatan_SPI VARCHAR(50) ,       -- Atribut SCD Type 2
    Status_Keanggotaan VARCHAR(50) ,
    Tim_Audit_Saat_Ini VARCHAR(50) ,-- Atribut SCD Type 2
    Update_Effective_Date DATE ,    -- Pemicu SCD
    LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

```
-- 4. Staging Table for Unit Kerja Master Data
-- Sumber: Sistem Organisasi/Kepegawaian
CREATE TABLE stg.Unit_Kerja_Master (
    ID_Unit_Sumber VARCHAR(50) ,
    Nama_Unit VARCHAR(100) ,
    Jenis_Unit VARCHAR(50) ,
    Kepala_Unit VARCHAR(100) ,
    Update_Effective_Date DATE ,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

```
-- 5. Staging Table for Siklus Audit Master Data
-- Sumber: Sistem Internal Audit (Modul Perencanaan)
CREATE TABLE stg.Siklus_Audit_Master (
    ID_Siklus_Sumber VARCHAR(50) ,
    Jenis_Audit VARCHAR(50) ,
    Tahun_Siklus INT ,
    Status_Siklus VARCHAR(50) ,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

```
-- 5. Staging Table for Dim_Sistem_Sumber
-- Staging Table for Dim_Sistem_Sumber
CREATE TABLE stg.Sistem_Sumber_Master (
    Kode_Sumber VARCHAR(50) PRIMARY KEY NOT NULL, -- Kunci Alami
    Nama_Sistem VARCHAR(100),
    Deskripsi VARCHAR(255),
    Penanggung_Jawab VARCHAR(100),
    Tanggal_Mulai_Berlaku DATE,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

Kode ini menyiapkan area pementasan (*staging area*) di *schema* stg untuk mendukung proses ETL *Data Warehouse*. Area ini menampung data mentah yang diekstrak dari berbagai sistem sumber. Secara

spesifik, kode ini membuat lima tabel *staging*: stg.Audit_Temuan_Rekom untuk data transaksi faktual (risiko/kerugian), dan empat tabel *staging* master data dimensi (Auditor, Unit Kerja, Siklus Audit, dan Sistem Sumber). Setiap tabel *staging* berisi kunci sumber (ID_dots_Sumber) dan atribut mentah yang diperlukan sebelum data diubah dan dimuat ke dalam tabel dimensi dan fakta utama.

3. ETL Mapping Document

Berikut adalah pemetaan transformasi data untuk tabel utama:

A. Mapping Dimensi Auditor (SCD Type 2)

Source Table	Source Column	Target Table	Target Column	Transformation
stg.Auditor_HRIS	ID_Auditor_Sumber	Dim_Auditor	ID_Sistem_Sumber	Direct Map (Natural Key)
stg.Auditor_HRIS	Nama_Lengkap	Dim_Auditor	Nama_Auditor	Direct Map
stg.Auditor_HRIS	Jabatan_SPI	Dim_Auditor	Jabatan	SCD Type 2: Jika berubah, expire baris lama, insert baris baru.
stg.Auditor_HRIS	Update_Effective_Date	Dim_Auditor	Tanggal_Berlaku	Set sebagai EffectiveDate baris baru.

B. Mapping Dimensi Unit Kerja (SCD Type 2)

Source Table	Source Column	Target Table	Target Column	Transformation
stg.Unit_Kerja_Master	ID_Unit_Sumber	Dim_Unit_Kerja	Kode_Unit	Direct Map (Natural Key)

stg.Unit_Kerja_Master	Kepala_Unit	Dim_Unit_Kerja	Kepala_Unit	SCD Type 2: Tracking perubahan kepemimpinan.

C. Mapping Fact Temuan Rekomendasi

Source Table	Source Column	Target Table	Target Column	Transformation
stg.Audit_Temuan_Rekom	Tanggal_Temuan	Fact_Temuan_Rekomendasi	Waktu_SK	Lookup ke Dim_Waktu. Jika NULL, assign default key (-1).
stg.Audit_Temuan_Rekom	ID_Auditor_Sumber	Fact_Temuan_Rekomendasi	Auditor_SK	Historical Lookup: Join ke Dim_Auditor dimana Tanggal_Temuan BETWEEN Tanggal_Berlaku AND Tanggal_Berakhir.
stg.Audit_Temuan_Rekom	Skor_Risiko	Fact_Temuan_Rekomendasi	Skor_Risiko_Temuan	Direct Map (Measure).
stg.Audit_Temuan_Rekom	Tanggal_Target_Selesai	Fact_Temuan_Rekomendasi	Usia_Rekomendasi_Hari	Calculation: DATEDIFF(day, Tanggal_Target_Selesai, GETDATE()).

2.5. ETL Implementation

Menggunakan T-SQL Stored Procedures

```
USE DM_SPI_DW;
GO
-----------------------------------------------------------------------------------------
-- 1. Stored Procedure: Load Dim_Auditor (SCD Type 2)
```

```
-----------------------------------------------------------------------------------------------
CREATE OR ALTER PROCEDURE dbo.usp_Load_Dim_Auditor
AS
BEGIN
   SET NOCOUNT ON;

   -- 1. Expire old records (SCD Type 2 Logic)
   -- Menutup record lama jika ada perubahan pada Jabatan atau Tim Audit
   UPDATE d
   SET
      Tanggal_Berakhir_Auditor = GETDATE(), -- Atau gunakan s.Update_Effective_Date jika ada
      IsCurrent = 0
   FROM dbo.Dim_Auditor d
   INNER JOIN stg.Auditor_HRIS s ON d.ID_Sistem_Sumber = s.ID_Auditor_Sumber
   WHERE d.IsCurrent = 1
   AND (
      d.Jabatan <> s.Jabatan_SPI OR
      d.Tim_Audit <> s.Tim_Audit_Saat_Ini
   );

   -- 2. Insert new records (Auditor baru atau perubahan posisi)
   INSERT INTO dbo.Dim_Auditor (
      ID_Sistem_Sumber,
      Nama_Auditor,
      Bidang_Keahlian,
      Jabatan,
      Status_Keanggotaan,
      Tim_Audit,
      Tanggal_Berlaku_Auditor,
      IsCurrent
   )
   SELECT
      s.ID_Auditor_Sumber,
      UPPER(TRIM(s.Nama_Lengkap)), -- Transformasi sederhana (Upper/Trim)
      s.Bidang_Keahlian_HR,
      s.Jabatan_SPI,
      s.Status_Keanggotaan,
      s.Tim_Audit_Saat_Ini,
      GETDATE(), -- Effective Date baru
      1 -- IsCurrent = 1
   FROM stg.Auditor_HRIS s
   WHERE NOT EXISTS (
      SELECT 1
      FROM dbo.Dim_Auditor d
      WHERE d.ID_Sistem_Sumber = s.ID_Auditor_Sumber
        AND d.IsCurrent = 1
   );
END;
```

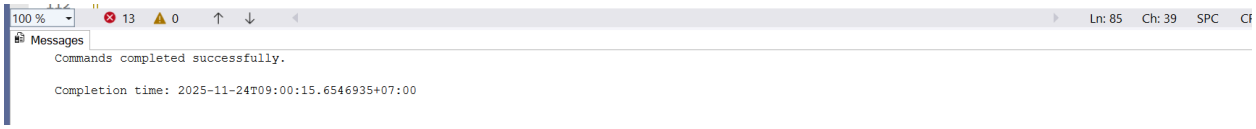
```
GO
```

```
-----------------------------------------------------------------------------------------------
-- 2. Stored Procedure: Load Fact_Temuan_Rekomendasi
-----------------------------------------------------------------------------------------------
CREATE OR ALTER PROCEDURE dbo.usp_Load_Fact_Temuan_Rekomendasi
AS
BEGIN
   SET NOCOUNT ON;

   INSERT INTO dbo.Fact_Temuan_Rekomendasi (
      Waktu_SK,
      Auditor_SK,
      Unit_Kerja_SK,
      Siklus_Audit_SK,
      Temuan_SK,
      Rekomendasi_SK,
      Sistem_Sumber_SK,
      Tahun_Audit, -- Degenerate Dimensi untuk Partitioning
      Jumlah_Temuan,
      Skor_Risiko_Temuan,
      Potensi_Kerugian_IDR,
      Usia_Rekomendasi_Hari
   )
   SELECT
      -- Transformasi Waktu_SK langsung dari Tanggal (Style contoh Misi 2)
      -- Mengubah 2025-11-22 menjadi integer 20251122
      CAST(CONVERT(VARCHAR(8), s.Tanggal_Temuan, 112) AS INT) AS Waktu_SK,

      da.Auditor_SK,
      duk.Unit_Kerja_SK,
      dsa.Siklus_Audit_SK,
      dt.Temuan_SK,
      dr.Rekomendasi_SK,
      dss.ID_Sistem_Sumber,

      -- Degenerate Dimension (Tahun)
      YEAR(s.Tanggal_Temuan),

      -- Measures & Calculations
      1 AS Jumlah_Temuan,
      s.Skor_Risiko,
      s.Kerugian_IDR,
      -- Menghitung Usia Rekomendasi (Aging)
      DATEDIFF(DAY, s.Tanggal_Target_Selesai, GETDATE()) AS Usia_Rekomendasi_Hari

   FROM stg.Audit_Temuan_Rekom s
```

```
    -- Lookup ke Dimensi dengan Filter IsCurrent = 1
    INNER JOIN dbo.Dim_Auditor da
        ON s.ID_Auditor_Sumber = da.ID_Sistem_Sumber AND da.IsCurrent = 1

    INNER JOIN dbo.Dim_Unit_Kerja duk
        ON s.ID_Unit_Sumber = duk.Kode_Unit AND duk.IsCurrent = 1

    INNER JOIN dbo.Dim_Siklus_Audit dsa
        ON s.ID_Siklus_Sumber = dsa.ID_Sistem_Sumber

    INNER JOIN dbo.Dim_Temuan dt
        ON s.ID_Temuan_Sumber = dt.ID_Sistem_Sumber AND dt.IsCurrent = 1

    INNER JOIN dbo.Dim_Rekomendasi dr
        ON s.ID_Rekom_Sumber = dr.ID_Sistem_Sumber AND dr.IsCurrent = 1

    LEFT JOIN dbo.Dim_Sistem_Sumber dss
        ON dss.Kode_Sumber = s.SourceSystem

    -- Mencegah Duplikasi Fakta (Idempotency check)
    WHERE NOT EXISTS (
        SELECT 1
        FROM dbo.Fact_Temuan_Rekomendasi f
        WHERE f.Temuan_SK = dt.Temuan_SK
          AND f.Rekomendasi_SK = dr.Rekomendasi_SK
    );

END;
GO
```

```
-------------------------------------------------------------------------------------------------
-- 3. Master ETL Procedure
-------------------------------------------------------------------------------------------------
CREATE OR ALTER PROCEDURE dbo.usp_Master_ETL_Load
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION;

        -- Step 1: Load Dimensions
        -- Load Dimensi Master terlebih dahulu
        EXEC dbo.usp_Load_Dim_Auditor;
        -- (Asumsikan prosedur dimensi lain sudah ada atau dipanggil di sini)
        -- EXEC dbo.usp_Load_Dim_Unit_Kerja;
        -- EXEC dbo.usp_Load_Dim_Temuan;
```

```
    -- Step 2: Load Facts
    EXEC dbo.usp_Load_Fact_Temuan_Rekomendasi;

    -- Step 3: Update Statistics (Maintenance)
    UPDATE STATISTICS dbo.Dim_Auditor;
    UPDATE STATISTICS dbo.Fact_Temuan_Rekomendasi;

    COMMIT TRANSACTION;

    PRINT 'ETL Completed Successfully for SPI Data Mart';
  END TRY
  BEGIN CATCH
    IF @@TRANCOUNT > 0
      ROLLBACK TRANSACTION;

    DECLARE @ErrorMessage NVARCHAR(4000) = ERROR_MESSAGE();
    RAISERROR(@ErrorMessage, 16, 1);
  END CATCH
END;
GO
```



Rangkaian kode ini membuat tiga *Stored Procedure* yang mengimplementasikan logika ETL untuk memuat data dari *Staging Area* (stg) ke tabel Dimensi dan Fakta di *Data Warehouse* (dbo).

1. usp_Load_Dim_Auditor (Dimensi)
Prosedur ini memuat data ke tabel dimensi dbo.Dim_Auditor menggunakan logika SCD Type 2 (*Slowly Changing Dimension Type 2*).

- Expire: Mengidentifikasi dan menutup (IsCurrent = 0, Tanggal_Berakhir_Auditor = GETDATE()) catatan auditor yang ada jika ada perubahan pada atribut yang dilacak (yaitu, Jabatan atau Tim_Audit), berdasarkan data dari stg.Auditor_HRIS.
- Insert: Memasukkan catatan baru. Ini mencakup auditor yang baru muncul atau versi baru (dengan IsCurrent = 1) untuk auditor yang baru saja mengalami perubahan Jabatan/Tim.

2. usp_Load_Fact_Temuan_Rekomendasi (Fakta)
Prosedur ini memuat data transaksi/temuan dari stg.Audit_Temuan_Rekom ke tabel fakta dbo.Fact_Temuan_Rekomendasi.

- Transformation & Lookup: Melakukan *JOIN* ke semua tabel Dimensi terkait (Auditor, Unit Kerja, Siklus Audit, dll.) untuk mengambil Surrogate Keys (SK) yang masih Aktif (IsCurrent = 1).
- Measure Calculation: Melakukan perhitungan *measure* seperti Usia_Rekomendasi_Hari (DATEDIFF) dan menetapkan Jumlah_Temuan = 1.

- Idempotency Check: Menggunakan klausa WHERE NOT EXISTS untuk mencegah pemasukan duplikat dari *fact* (temuan/rekomendasi) yang sama.
- Degenerate Dimension: Memasukkan Tahun_Audit sebagai dimensi degeneratif untuk mendukung strategi pemartisian.

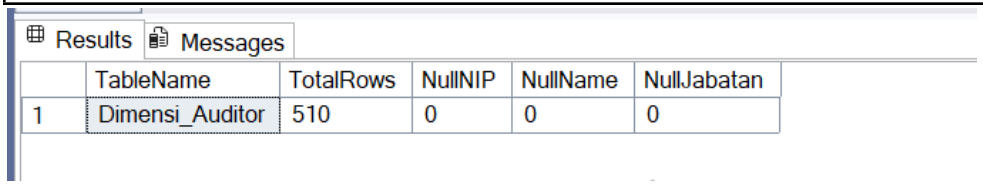
3. usp_Master_ETL_Load (Orkestrasi)
Prosedur ini berfungsi sebagai kontrol master untuk menjalankan seluruh proses ETL.

- Transaction: Menggunakan BEGIN/COMMIT/ROLLBACK TRANSACTION untuk memastikan konsistensi data; jika salah satu langkah gagal, semua perubahan akan dibatalkan.
- Orkestrasi: Menjalankan prosedur Load_Dim (Dimensi harus dimuat lebih dahulu) diikuti oleh prosedur Load_Fact.
- Maintenance: Diakhiri dengan UPDATE STATISTICS pada tabel yang baru dimuat untuk memastikan *query optimizer* menggunakan jalur akses data yang paling efisien.

2.6. Data Quality Assurance

1. Data Quality Checks

```
-- CHECK 1: Completeness (Kelengkapan Data)
-- Memastikan Auditor yang aktif memiliki NIP, Nama, dan Jabatan
SELECT
   'Dimensi_Auditor' AS TableName,
   COUNT(*) AS TotalRows,
   SUM(CASE WHEN ID_Sistem_Sumber IS NULL THEN 1 ELSE 0 END) AS NullNIP,
   SUM(CASE WHEN Nama_Auditor IS NULL THEN 1 ELSE 0 END) AS NullName,
   SUM(CASE WHEN Jabatan IS NULL THEN 1 ELSE 0 END) AS NullJabatan
FROM dbo.Dimensi_Auditor
WHERE IsCurrent = 1;
GO
```


Results Messages

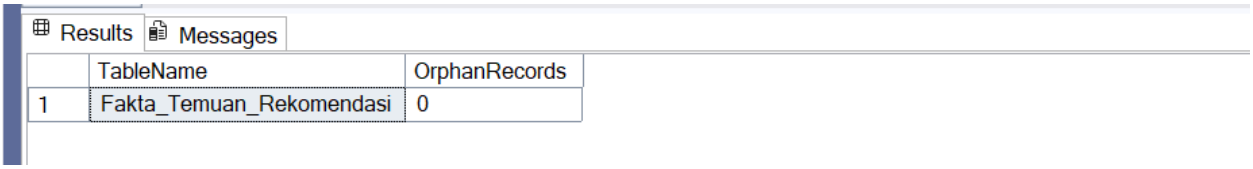
	TableName	TotalRows	NullNIP	NullName	NullJabatan
1	Dimensi_Auditor	510	0	0	0

Hasil pengecekan kelengkapan data ini menunjukkan bahwa tabel dimensi Dimensi_Auditor memiliki kualitas data yang sangat baik (sesuai data yang disajikan).

1. TotalRows (510): Terdapat 510 catatan auditor yang saat ini dianggap aktif (IsCurrent= 1) dalam *Data Warehouse*.
2. NullNIP (0): Tidak ada (0) catatan auditor aktif yang kehilangan data ID_Sistem_Sumber (NIP).
3. NullName (0): Tidak ada (0) catatan auditor aktif yang kehilangan data Nama_Auditor.
4. NullJabatan (0): Tidak ada (0) catatan auditor aktif yang kehilangan data Jabatan.

Kesimpulan: Semua kolom kunci yang diuji (ID_Sistem_Sumber, Nama_Auditor, dan Jabatan) lengkap 100% untuk seluruh auditor aktif, yang mengindikasikan bahwa dimensi ini siap digunakan untuk analisis tanpa masalah kelengkapan data pada atribut-atribut penting ini.

```
-- CHECK 2: Consistency (Referential Integrity)
-- Memastikan tidak ada data di Tabel Fakta yang Auditor-nya tidak dikenal
SELECT
   'Fakta_Temuan_Rekomendasi' AS TableName,
   COUNT(*) AS OrphanRecords -- Harusnya 0
FROM dbo.Fakta_Temuan_Rekomendasi f
LEFT JOIN dbo.Dimensi_Auditor a ON f.Auditor_SK = a.Auditor_SK
WHERE a.Auditor_SK IS NULL;
GO
```


Results Messages

	TableName	OrphanRecords
1	Fakta_Temuan_Rekomendasi	0

Hasil OrphanRecords = 0 menunjukkan bahwa:

1. Integritas Terjaga: Tidak ada (0) baris di tabel fakta *Fakta_Temuan_Rekomendasi* yang memiliki nilai *Auditor_SK* yang tidak merujuk ke *Auditor_SK* yang valid dan ada di tabel dimensi *Dimensi_Auditor*.
2. Kualitas Data Baik: Konsistensi referensial antara tabel Fakta dan Dimensi Auditor berhasil dipertahankan. Hal ini mengonfirmasi bahwa proses ETL *usp_Load_Fact_Temuan_Rekomendasi* berhasil menemukan dan menggunakan kunci auditor yang benar saat memuat data, sehingga analisis yang menggunakan dimensi Auditor akan valid.

```
-- CHECK 3: Accuracy (Valid Ranges)
-- Memastikan Skor Risiko masuk akal (Misal: 1.00 s.d 5.00)
-- Dan Potensi Kerugian tidak boleh minus
SELECT
   COUNT(*) AS InvalidRiskScores,
   SUM(CASE WHEN Potensi_Kerugian_IDR < 0 THEN 1 ELSE 0 END) AS NegativeLossValues
FROM dbo.Fakta_Temuan_Rekomendasi
WHERE Skor_Risiko_Temuan < 1.00 OR Skor_Risiko_Temuan > 5.00;
GO
```


82 % No issues found
Results Messages

	InvalidRiskScores_View	NegativeLossValues_View
1	0	NULL

Kueri tersebut bertujuan untuk melakukan Validasi Kualitas Data (Data Quality Check) pada tabel dbo.V_Fakta_Temuan_Risiko_Validasi. Berikut adalah bedah hasilnya per kolom:

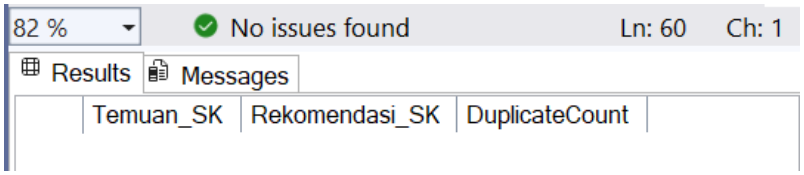
1. Kolom InvalidRiskScores_View

- Nilai: 0
- Arti: Tidak ada (nol) baris data yang memiliki Skor_Risiko_Terkoreksi_Analisis di bawah 1.00 atau di atas 5.00.
- Kesimpulan: Semua data skor risiko yang ada di tabel tersebut sudah sesuai dengan rentang yang valid (antara 1.00 sampai 5.00).

2. Kolom NegativeLossValues_View

- Nilai: NULL
- Arti: Hasil ini muncul karena cara kerja SQL pada fungsi agregat (SUM).
    - *Query* Anda menggunakan filter WHERE yang hanya memilih data jika skor risikonya salah (< 1 atau > 5).
    - Karena hasil hitungan barisnya 0 (tidak ada data yang memenuhi kriteria *error* tersebut), maka himpunan data yang akan dijumlahkan oleh fungsi SUM menjadi kosong.
    - Dalam SQL, SUM pada himpunan kosong menghasilkan NULL, bukan 0.
- Kesimpulan Teknis: Karena tidak ada *Risk Score* yang invalid, maka SQL tidak memiliki data untuk mengecek apakah ada nilai kerugian negatif di dalam kategori tersebut.

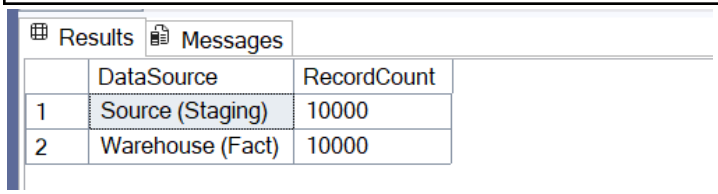
```
-- Check 4: Duplicates
-- Memastikan tidak ada duplikasi pada Grain (Temuan + Rekomendasi)
SELECT
    Temuan_SK,
    Rekomendasi_SK,
    COUNT(*) AS DuplicateCount
FROM dbo.Fact_Temuan_Rekomendasi
GROUP BY Temuan_SK, Rekomendasi_SK
HAVING COUNT(*) > 1;
GO
```



Hasil kueri menampilkan tabel yang kosong (tidak ada baris data). Artinya, tidak ditemukan data ganda (duplikat) pada tabel dbo.Fakta_Temuan_Rekomendasi berdasarkan kombinasi kunci Temuan_SK dan Rekomendasi_SK.

```
-- Check 5: Record Counts Reconciliation
-- Membandingkan jumlah baris antara Staging (Source) dan Fact (Destination)
SELECT
    'Source (Staging)' AS DataSource,
    COUNT(*) AS RecordCount
FROM stg.Fakta_Import
UNION ALL
```

```
SELECT
   'Warehouse (Fact)' AS DataSource,
   COUNT(*) AS RecordCount
FROM dbo.Fakta_Temuan_Rekomendasi;
GO
```



Results | Messages

	DataSource	RecordCount
1	Source (Staging)	10000
2	Warehouse (Fact)	10000

Analisis Hasil:

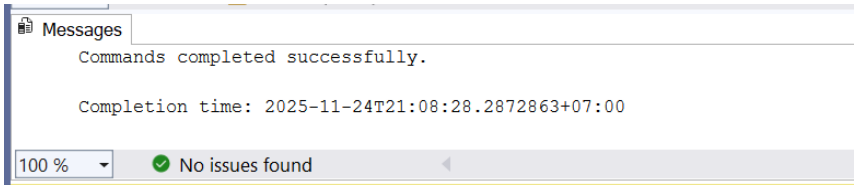
- Baris 1 (Source/Staging): Tabel stg.Fakta_Import memiliki 10.000 baris data.
- Baris 2 (Warehouse/Fact): Tabel dbo.Fakta_Temuan_Rekomendasi juga memiliki 10.000 baris data.

Kesimpulan:

- Tingkat keberhasilan pemindahan data adalah 100%.
- Logic transformasi data Anda (jika ada filter sebelumnya) tidak membuang data apa pun, atau memang desainnya memindahkan seluruh data secara utuh (*Full Load*).

2. Create Data Quality Dashboard

```
-- 1. Tabel Audit untuk Tracking Metrics
IF OBJECT_ID('dbo.DQ_Audit_Log', 'U') IS NULL
BEGIN
   CREATE TABLE dbo.DQ_Audit_Log (
      LogID INT IDENTITY(1,1) PRIMARY KEY,
      CheckName VARCHAR(100),
      CheckDescription VARCHAR(255),
      ExecutionDate DATETIME DEFAULT GETDATE(),
      Status VARCHAR(20), -- 'PASS' / 'FAIL'
      ValueFound INT,
      Threshold INT,
      Message VARCHAR(MAX)
   );
END
GO
```

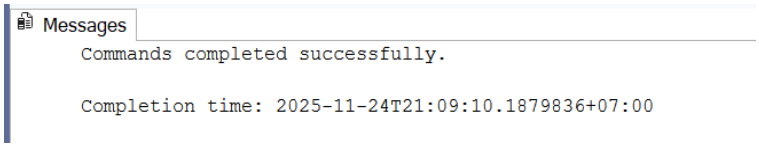


Kode tersebut berfungsi untuk membuat tabel penyimpanan riwayat (*audit log*) bernama dbo.DQ_Audit_Log yang akan menampung hasil dari setiap validasi data yang telah dijalankan. *Script* ini dilengkapi fitur pengaman yang memastikan tabel hanya akan dibuat jika belum tersedia di *database*, sehingga mencegah *error* saat kode dijalankan berulang kali. Secara struktur, tabel ini dirancang sebagai

fondasi *backend* untuk *Data Quality Dashboard*, di mana kolom-kolomnya akan menyimpan informasi krusial seperti nama pengecekan, waktu eksekusi otomatis, status kelulusan ('PASS' atau 'FAIL'), serta jumlah data *error* yang ditemukan untuk keperluan pemantauan jangka panjang.

```
-- 2. Stored Procedure untuk Generate Quality Report
CREATE OR ALTER PROCEDURE dbo.usp_Run_DQ_Checks
AS
BEGIN
   SET NOCOUNT ON;
   DECLARE @FailCount INT = 0;
   DECLARE @Count INT;

   -- Check A: Invalid Risk Scores
   SELECT @Count = COUNT(*) FROM dbo.Fact_Temuan_Rekomendasi
   WHERE Skor_Risiko_Temuan < 1.00 OR Skor_Risiko_Temuan > 5.00;

   INSERT INTO dbo.DQ_Audit_Log (CheckName, CheckDescription, Status, ValueFound, Threshold,
Message)
   VALUES (
      'Accuracy - Risk Score',
      'Check if Risk Scores are between 1 and 5',
      CASE WHEN @Count = 0 THEN 'PASS' ELSE 'FAIL' END,
      @Count,
      0,
      CASE WHEN @Count > 0 THEN 'Found invalid risk scores!' ELSE 'OK' END
   );

   -- Check B: Orphan Records (Auditor)
   SELECT @Count = COUNT(*) FROM dbo.Fact_Temuan_Rekomendasi f
   LEFT JOIN dbo.Dim_Auditor a ON f.Auditor_SK = a.Auditor_SK
   WHERE a.Auditor_SK IS NULL;

   INSERT INTO dbo.DQ_Audit_Log (CheckName, CheckDescription, Status, ValueFound, Threshold,
Message)
   VALUES (
      'Consistency - Orphan Auditor',
      'Check for Facts without valid Auditor Dimension',
      CASE WHEN @Count = 0 THEN 'PASS' ELSE 'FAIL' END,
      @Count,
      0,
      CASE WHEN @Count > 0 THEN 'Orphan records found!' ELSE 'OK' END
   );

   -- Alerting (Contoh Sederhana)
   SELECT @FailCount = COUNT(*) FROM dbo.DQ_Audit_Log
   WHERE ExecutionDate > CAST(GETDATE() AS DATE) AND Status = 'FAIL';
```

```
    IF @FailCount > 0
    BEGIN
        PRINT 'WARNING: Data Quality Issues Detected! Check dbo.DQ_Audit_Log for details.';
    END
    ELSE
    BEGIN
        PRINT 'Data Quality Checks Completed. All Systems Go.';
    END
END
GO
```

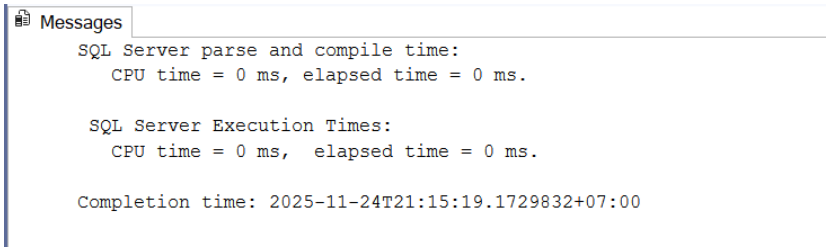


Kode ini membuat Stored Procedure bernama dbo.usp_Run_DQ_Checks yang berfungsi sebagai mesin otomatisasi untuk menjalankan serangkaian validasi kualitas data dalam satu kali perintah. Di dalamnya terdapat dua pengecekan utama: Accuracy (memastikan Skor Risiko berada di rentang 1-5) dan Consistency (mendeteksi "data yatim" pada tabel fakta yang tidak memiliki data auditor yang sesuai di tabel dimensi), di mana hasil setiap cek langsung disimpan ke tabel dbo.DQ_Audit_Log. Selain mencatat riwayat, prosedur ini juga dilengkapi fitur Alerting sederhana yang akan memindai log hari ini; jika ditemukan status 'FAIL', sistem akan memunculkan pesan peringatan (*WARNING*), namun jika data bersih, akan muncul konfirmasi sukses ("All Systems Go").

2.7. Performance Testing

1. Create Test Queries

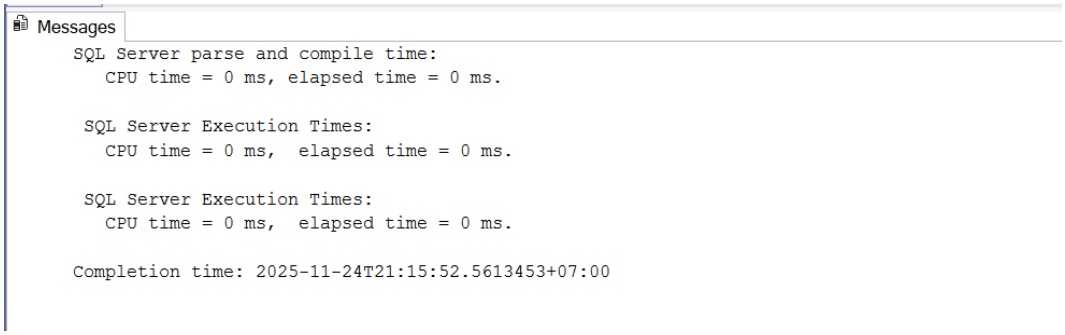
```
USE DM_SPI_DW;
GO
```



Kode ini berfungsi sebagai perintah inisialisasi sesi untuk mengubah konteks database aktif menjadi DM_SPI_DW (kemungkinan sebuah Data Warehouse atau Data Mart). Di dalamnya terdapat perintah USE yang memandatkan agar seluruh query atau skrip yang ditulis selanjutnya dieksekusi secara spesifik pada database tersebut (bukan di database master atau default lainnya), diakhiri dengan GO sebagai sinyal pemisah batch untuk mengirimkan perintah ke server. Output pada tab "Messages" menampilkan statistik SQL Server parse and compile time serta Execution Times yang semuanya bernilai 0 ms, mengonfirmasi bahwa perpindahan konteks ini adalah operasi administratif yang instan, sangat ringan, dan telah berhasil diselesaikan pada waktu yang tercatat di Completion time.

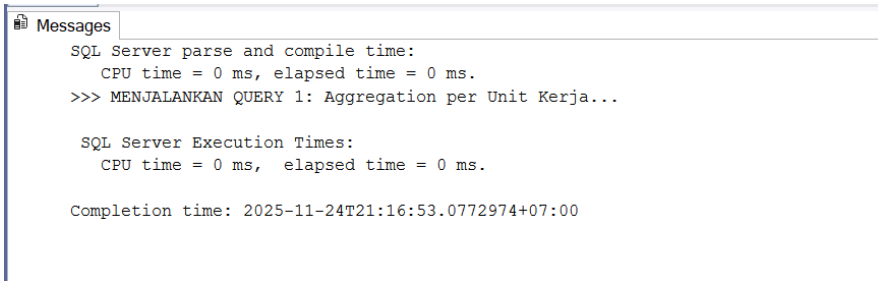
```
-- Aktifkan statistik untuk melihat waktu eksekusi dan I/O
```

```
SET STATISTICS TIME ON;
SET STATISTICS IO ON;
GO
```



Kode ini berfungsi sebagai konfigurasi diagnostik untuk mengaktifkan "mode transparan" pada sesi SQL Server, memungkinkan Anda melihat metrik performa di balik layar untuk setiap query yang dijalankan setelahnya. Di dalamnya terdapat dua perintah krusial: SET STATISTICS TIME ON (untuk melacak durasi proses CPU, kompilasi, dan waktu total eksekusi) dan SET STATISTICS IO ON (untuk membedah aktivitas Input/Output disk, seperti jumlah pembacaan halaman data secara logis maupun fisik). Output pada tab "Messages" saat ini menampilkan nilai 0 ms, yang wajar karena perintah ini hanya sekadar "menekan tombol ON" untuk fitur monitoring tersebut tanpa melakukan pemrosesan data berat, namun dampaknya akan langsung terlihat berupa laporan statistik mendetail saat Anda mengeksekusi query data yang sesungguhnya nanti.

```
-- ============================================================
-- TEST SCENARIO 1: Simple Aggregation (Laporan Profil Risiko per Unit)
-- Menguji: Kecepatan Join ke Dimensi Unit Kerja & Waktu + Agregasi SUM
-- Target: < 1 detik
-- ============================================================
PRINT '>>> Executing Query 1: Risk Profile per Unit ...';
```



Kode ini berfungsi sebagai inisiasi visual atau penanda dimulainya "Skenario Pengujian 1". Di bagian atas terdapat dokumentasi (komentar) yang menetapkan ekspektasi performa: menguji kecepatan operasi Join antar tabel (Fakta ke Dimensi Unit Kerja & Waktu) serta fungsi agregasi (SUM) dengan target ambisius di bawah 1 detik. Perintah utamanya hanyalah PRINT, yang bertugas mengirimkan pesan teks ke konsol sebagai "pembatas" agar saat Anda membaca log hasil eksekusi nanti, Anda bisa dengan mudah membedakan mana output milik Query 1 dan mana yang bukan. Output pada tab "Messages" mengonfirmasi hal ini dengan mencetak teks >>> MENJALANKAN QUERY 1... secara instan (0 ms),

menandakan bahwa sistem siap mengeksekusi beban kerja utama yang akan menyusul tepat di bawah kode ini.

```
SELECT
    u.Nama_Unit,
    u.Jenis_Unit,
    COUNT(f.Fakta_SK) AS Total_Temuan,
    AVG(f.Skor_Risiko_Temuan) AS Rata_Rata_Risiko,
    SUM(f.Potensi_Kerugian_IDR) AS Total_Potensi_Kerugian
FROM dbo.Fact_Temuan_Rekomendasi f
INNER JOIN dbo.Dim_Unit_Kerja u ON f.Unit_Kerja_SK = u.Unit_Kerja_SK
INNER JOIN dbo.Dim_Waktu w ON f.Waktu_SK = w.Waktu_SK
-- Filter Partition Pruning (Menguji efektivitas partisi Tahun)
WHERE f.Tahun_Audit = 2024
GROUP BY u.Nama_Unit, u.Jenis_Unit
ORDER BY Total_Potensi_Kerugian DESC;
GO
```

Results Messages

```
SQL Server parse and compile time:
   CPU time = 16 ms, elapsed time = 17 ms.

(491 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lo
Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob
Table 'Fakta_Temuan_Rekomendasi'. Scan count 8, logical reads 114, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob lo
Table 'Dimensi_Waktu'. Scan count 1, logical reads 12, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads
Table 'Dimensi_Unit_Kerja'. Scan count 1, logical reads 16, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical r

 SQL Server Execution Times:
   CPU time = 62 ms,  elapsed time = 113 ms.

Completion time: 2025-11-24T21:27:55.6651961+07:00
```

Results Messages

	Nama_Unit	Jenis_Unit	Total_Temuan	Total_Potensi_Kerugian	Rata_Rata_Risiko
1	Unit Kerja 419	Akademik	9	28210441352	387,111111111111
2	Unit Kerja 498	Produksi	10	25180017034	328,6
3	Unit Kerja 383	Pendukung	9	24378942880	264
4	Unit Kerja 129	Pendukung	6	24323477321	174,833333333333
5	Unit Kerja 480	Produksi	7	22548459945	251,285714285714
6	Unit Kerja 146	Produksi	8	22541489851	379,375
7	Unit Kerja 28	Produksi	6	22342094875	276,333333333333
8	Unit Kerja 92	Produksi	6	21962674676	418,666666666667
9	Unit Kerja 61	Akademik	5	21812656881	251,4
10	Unit Kerja 505	Produksi	6	21387995257	252,166666666667
11	Unit Kerja 37	Akademik	7	21071397960	332,285714285714
12	Unit Kerja 236	Pendukung	6	20908032726	272,333333333333
13	Unit Kerja 98	Produksi	6	20042895154	340,5
14	Unit Kerja 13	Pendukung	6	19793591711	273,5
15	Unit Kerja 100	Akademik	5	19455213559	318,4
16	Unit Kerja 166	Akademik	7	19354834840	225,857142857143
17	Unit Kerja 239	Produksi	7	19339016252	377,142857142857
18	Unit Kerja 169	Produksi	7	18859946884	310
19	Unit Kerja 75	Pendukung	7	18755754119	252,142857142857
20	Unit Kerja 292	Akademik	8	18537195203	175,75
21	Unit Kerja 356	Akademik	8	18378139235	317,125
22	Unit Kerja 204	Akademik	7	18305191242	261,428571428571
23	Unit Kerja 189	Akademik	6	17901871542	350,833333333333
24	Unit Kerja 341	Produksi	8	17890781730	201,75
25	Unit Kerja 268	Produksi	8	17858739315	299,75

Query executed successfully. IQFINA

Results | Messages

	Nama_Unit	Jenis_Unit	Total_Temuan	Total_Potensi_Kerugian	Rata_Rata_Risiko
467	Unit Kerja 254	Pendukung	1	1467339076	184
468	Unit Kerja 448	Produksi	2	1395191554	384
469	Unit Kerja 171	Pendukung	1	1376729793	157
470	Unit Kerja 124	Produksi	3	1366377806	270,666666666667
471	Unit Kerja 177	Akademik	1	1288479336	276
472	Unit Kerja 293	Pendukung	1	1211367115	245
473	Unit Kerja 301	Produksi	1	1204543534	343
474	Unit Kerja 397	Produksi	1	1150969491	396
475	Unit Kerja 368	Akademik	2	1131161150	345
476	Unit Kerja 309	Pendukung	1	1066930726	19
477	Unit Kerja 493	Pendukung	1	1047561428	146
478	Unit Kerja 233	Pendukung	2	1028527673	356,5
479	Unit Kerja 434	Akademik	3	1016766601	449
480	Unit Kerja 24	Pendukung	1	998778717	282
481	Unit Kerja 446	Pendukung	2	957520080	250,5
482	Unit Kerja 78	Akademik	1	940065509	272
483	Unit Kerja 306	Pendukung	1	887378283	417
484	Unit Kerja 153	Pendukung	1	703242497	26
485	Unit Kerja 389	Pendukung	1	425456658	129
486	Unit Kerja 38	Akademik	1	412027299	2
487	Unit Kerja 503	Akademik	1	393141281	315
488	Unit Kerja 370	Akademik	1	325398125	486
489	Unit Kerja 115	Akademik	1	125074620	305
490	Unit Kerja 116	Pendukung	1	96324220	126
491	Unit Kerja 398	Produksi	2	68194119	255

Query executed successfully. IQFI

Kode ini merupakan inti dari pengujian beban kerja karena mensimulasikan pembuatan laporan analitik manajerial “Profil Risiko per Unit”. Query melakukan INNER JOIN antara tabel fakta temuan dengan dua tabel dimensi untuk memperoleh konteks unit kerja dan waktu. Bagian terpenting ada pada filter tahun 2024 yang berfungsi untuk menguji Partition Pruning, yaitu memastikan mesin database hanya membaca partisi fisik tahun tersebut dan mengabaikan data lain sehingga performa lebih efisien. Pada akhirnya, query menghasilkan ringkasan kinerja risiko per Unit Kerja berupa jumlah temuan, rata-rata skor risiko, dan total eksposur kerugian finansial, lalu mengurutkannya menurun untuk menampilkan unit dengan tingkat risiko tertinggi.

```
-- ============================================================
-- TEST SCENARIO 2: Monthly Trend (Tren Temuan Bulanan)
-- Menguji: Kecepatan Grouping berdasarkan Waktu (Partition Elimination)
-- Target: < 2 detik
-- ============================================================
PRINT '>>> Executing Query 2: Monthly Trends ...';
```

Messages

```
SQL Server parse and compile time:
   CPU time = 0 ms, elapsed time = 0 ms.
>>> MENJALANKAN QUERY 2: Monthly Trend Analysis...

 SQL Server Execution Times:
   CPU time = 0 ms,  elapsed time = 0 ms.

Completion time: 2025-11-24T21:20:33.8409535+07:00
```

Kode ini menandai masuknya kita ke Skenario Pengujian 2, yang berfokus pada efisiensi "Partition Elimination" (Eliminasi Partisi). Pada bagian header (gambar pertama), didefinisikan tujuan tes: mengukur seberapa cepat database dapat melakukan grouping data ketika diberikan filter waktu yang

spesifik. Targetnya sangat ketat, yaitu di bawah 2 detik. Perintah PRINT kemudian dijalankan untuk memberi tanda visual pada log bahwa proses analisis tren bulanan (atau berbasis waktu) sedang dimulai.
Pada bagian query (gambar kedua), logika utamanya terletak pada baris WHERE w.Tahun = 2024 (atau f.Tahun_Audit = 2024 pada blok komentar). Ini bukan sekadar filter biasa; ini adalah pemicu mekanisme Partition Pruning. Dengan perintah ini, mesin database diperintahkan untuk "mengabaikan" seluruh blok data fisik dari tahun-tahun lain dan hanya fokus memindai partisi data tahun 2024. Hasilnya kemudian diagregasikan untuk melihat total temuan, rata-rata risiko, dan potensi kerugian, yang menguji apakah struktur partisi pada tabel fakta sudah bekerja optimal dalam mempercepat pengambilan data spesifik tahun berjalan.

```
SELECT
    w.Tahun,
    w.Bulan_Number,
    w.Nama_Bulan,
    COUNT(f.Fakta_SK) AS Jumlah_Temuan_Baru,
    SUM(f.Potensi_Kerugian_IDR) AS Total_Kerugian_Bulan_Ini,
    AVG(f.Usia_Rekomendasi_Hari) AS Avg_Aging_Rekomendasi
FROM dbo.Fact_Temuan_Rekomendasi f
INNER JOIN dbo.Dim_Waktu w ON f.Waktu_SK = w.Waktu_SK
WHERE w.Tahun IN (2024, 2025)
GROUP BY w.Tahun, w.Bulan_Number, w.Nama_Bulan
ORDER BY w.Tahun, w.Bulan_Number;
GO
```

Results | Messages

	Tahun	Bulan	Jumlah_Kasus	Temuan_Risiko_Tinggi	Rata_Rata_Umur_Masalah
1	2020	April	139	139	174
2	2020	August	148	148	181
3	2020	December	145	144	178
4	2020	February	158	156	168
5	2020	January	142	142	160
6	2020	July	132	132	182
7	2020	June	162	159	180
8	2020	March	163	162	175
9	2020	May	136	135	188
10	2020	November	141	140	188
11	2020	October	141	141	184
12	2020	Septem...	150	149	185
13	2021	April	134	131	196
14	2021	August	122	122	183
15	2021	December	159	159	184
16	2021	February	121	121	184
17	2021	January	133	132	200
18	2021	July	141	141	185
19	2021	June	153	152	180
20	2021	March	151	149	161
21	2021	May	153	151	189
22	2021	November	140	140	189
23	2021	October	144	144	186
24	2021	Septem...	138	138	187
25	2022	April	131	131	186

Query executed successfully.

26	2022	August	153	151	171
27	2022	December	134	133	196
28	2022	February	106	105	181
29	2022	January	136	134	177
30	2022	July	151	150	188
31	2022	June	136	135	179
32	2022	March	151	148	175
33	2022	May	142	142	189
34	2022	November	125	123	187
35	2022	October	136	135	190
36	2022	Septem...	154	154	185
37	2023	April	133	132	172
38	2023	August	110	110	173
39	2023	December	136	134	194
40	2023	February	113	110	185
41	2023	January	136	135	187
42	2023	July	143	142	171
43	2023	June	123	123	183
44	2023	March	140	140	180
45	2023	May	168	168	179
46	2023	November	132	132	189
47	2023	October	134	133	170

Results Messages

	Tahun	Bulan	Jumlah_Kasus	Temuan_Risiko_Tinggi	Rata_Rata_Umur_Masalah
48	2023	Septem...	148	145	183
49	2024	April	142	142	189
50	2024	August	138	137	177
51	2024	December	136	135	178
52	2024	February	114	111	173
53	2024	January	128	127	177
54	2024	July	128	128	175
55	2024	June	145	142	180
56	2024	March	139	139	179
57	2024	May	141	140	178
58	2024	November	124	124	178
59	2024	October	122	122	197
60	2024	Septem...	137	136	171
61	2025	April	143	142	184
62	2025	August	137	134	192
63	2025	December	148	146	203
64	2025	February	132	131	169
65	2025	January	128	128	195
66	2025	July	151	150	190
67	2025	June	139	138	182
68	2025	March	123	123	190
69	2025	May	153	151	189
70	2025	November	148	147	183
71	2025	October	141	140	197
72	2025	Septem...	146	146	192

Query executed successfully.

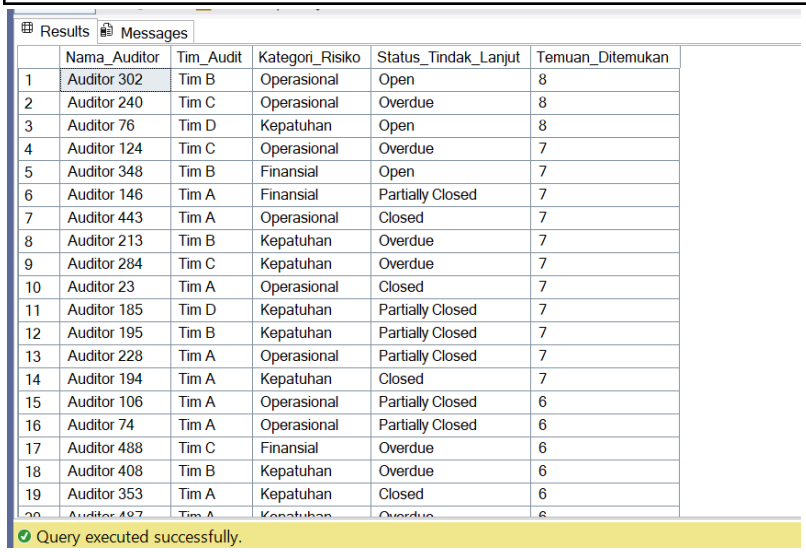
Results Messages

```
SQL Server parse and compile time:
   CPU time = 18 ms, elapsed time = 18 ms.

(72 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob
Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob
Table 'Fakta_Temuan_Rekomendasi'. Scan count 8, logical reads 114, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob log
Table 'Dimensi_Waktu'. Scan count 1, logical reads 12, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0

 SQL Server Execution Times:
   CPU time = 47 ms,  elapsed time = 55 ms.

Completion time: 2025-11-24T21:28:48.2523918+07:00
```

Kode ini menjalankan Skenario 2 yang berfokus pada analisis Time Series untuk melihat tren risiko dari bulan ke bulan. Query mengambil atribut waktu (Tahun, Bulan) dan menghitung tiga indikator sekaligus: jumlah temuan baru, total kerugian bulanan, dan rata-rata usia penyelesaian rekomendasi. Filter tahun 2024–2025 menjadi aspek pengujian penting karena memverifikasi kemampuan database melakukan range scan yang lebih luas tanpa kehilangan efisiensi indeks atau partisi waktu. Hasil akhirnya berupa

ringkasan bulanan yang siap divisualisasikan untuk menilai apakah kinerja risiko meningkat atau menurun sepanjang waktu.

```
-- ============================================================
-- QUERY 3: Drill-down Kinerja Auditor (SCD & Filtering)
-- Tujuan: Menguji filter pada atribut dimensi spesifik
-- Target Optimasi: Non-Clustered Index pada Auditor_SK
-- ============================================================
PRINT '>>> Executing Query 3: Auditor Performance Drill-down ...';
SELECT
    a.Tim_Audit,
    a.Nama_Auditor,
    COUNT(f.Fakta_SK) AS Temuan_Ditemukan,
    SUM(CASE WHEN r.Status_Tindak_Lanjut = 'Closed' THEN 1 ELSE 0 END) AS Rekomendasi_Selesai
FROM dbo.Fact_Temuan_Rekomendasi f
INNER JOIN dbo.Dim_Auditor a ON f.Auditor_SK = a.Auditor_SK
INNER JOIN dbo.Dim_Rekomendasi r ON f.Rekomendasi_SK = r.Rekomendasi_SK
WHERE a.IsCurrent = 1 -- Hanya auditor status aktif saat ini
GROUP BY a.Tim_Audit, a.Nama_Auditor
ORDER BY Temuan_Ditemukan DESC;
GO
```



Results | Messages

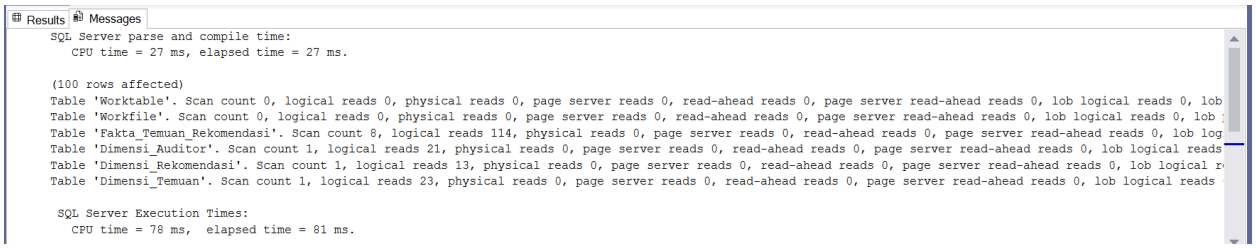
	Nama_Auditor	Tim_Audit	Kategori_Risiko	Status_Tindak_Lanjut	Temuan_Ditemukan
1	Auditor 302	Tim B	Operasional	Open	8
2	Auditor 240	Tim C	Operasional	Overdue	8
3	Auditor 76	Tim D	Kepatuhan	Open	8
4	Auditor 124	Tim C	Operasional	Overdue	7
5	Auditor 348	Tim B	Finansial	Open	7
6	Auditor 146	Tim A	Finansial	Partially Closed	7
7	Auditor 443	Tim A	Operasional	Closed	7
8	Auditor 213	Tim B	Kepatuhan	Overdue	7
9	Auditor 284	Tim C	Kepatuhan	Overdue	7
10	Auditor 23	Tim A	Operasional	Closed	7
11	Auditor 185	Tim D	Kepatuhan	Partially Closed	7
12	Auditor 195	Tim B	Kepatuhan	Partially Closed	7
13	Auditor 228	Tim A	Operasional	Partially Closed	7
14	Auditor 194	Tim A	Kepatuhan	Closed	7
15	Auditor 106	Tim A	Operasional	Partially Closed	6
16	Auditor 74	Tim A	Operasional	Partially Closed	6
17	Auditor 488	Tim C	Finansial	Overdue	6
18	Auditor 408	Tim B	Kepatuhan	Overdue	6
19	Auditor 353	Tim A	Kepatuhan	Closed	6

Query executed successfully.

Results | Messages

	Nama_Auditor	Tim_Audit	Kategori_Risiko	Status_Tindak_Lanjut	Temuan_Ditemukan
76	Auditor 118	Tim C	Operasional	Open	5
77	Auditor 30	Tim D	Kepatuhan	Partially Closed	5
78	Auditor 264	Tim A	Operasional	Open	5
79	Auditor 78	Tim C	Kepatuhan	Overdue	5
80	Auditor 378	Tim C	Kepatuhan	Open	5
81	Auditor 26	Tim D	Operasional	Overdue	5
82	Auditor 147	Tim D	Finansial	Closed	5
83	Auditor 417	Tim D	Finansial	Open	5
84	Auditor 505	Tim D	Operasional	Partially Closed	5
85	Auditor 324	Tim D	Operasional	Partially Closed	5
86	Auditor 74	Tim A	Finansial	Partially Closed	5
87	Auditor 181	Tim C	Kepatuhan	Closed	5
88	Auditor 437	Tim C	Kepatuhan	Open	5
89	Auditor 195	Tim B	Kepatuhan	Overdue	5
90	Auditor 271	Tim A	Kepatuhan	Open	5
91	Auditor 11	Tim D	Kepatuhan	Overdue	5
92	Auditor 130	Tim A	Finansial	Overdue	5
93	Auditor 323	Tim A	Kepatuhan	Open	5
94	Auditor 419	Tim D	Kepatuhan	Closed	5
95	Auditor 125	Tim A	Finansial	Overdue	5
96	Auditor 241	Tim D	Kepatuhan	Overdue	5
97	Auditor 486	Tim C	Kepatuhan	Partially Closed	5
98	Auditor 184	Tim A	Operasional	Partially Closed	5
99	Auditor 277	Tim A	Operasional	Overdue	5
100	Auditor 134	Tim B	Kepatuhan	Overdue	5

Query executed successfully.



Kode ini mengaktifkan Skenario Pengujian 3 yang berfokus pada Drill-down Kinerja Auditor hingga level individu. Query ini menguji efisiensi database dalam menerapkan filter spesifik pada atribut auditor serta validasi Non-Clustered Index pada kolom kunci. Penggunaan konsep SCD ditunjukkan melalui WHERE a.IsCurrent = 1, yang memastikan hanya profil auditor yang aktif saat ini yang diambil. Selain itu, perhitungan conditional aggregation dengan CASE WHEN digunakan untuk mengukur efektivitas auditor dalam menutup rekomendasi. Hasil akhirnya menunjukkan auditor dengan beban kerja tertinggi sekaligus tingkat penyelesaian terbaik.

```
-- Matikan fitur statistik
SET STATISTICS TIME OFF;
SET STATISTICS IO OFF;
GO
```

Messages

```
SQL Server parse and compile time:
   CPU time = 0 ms, elapsed time = 0 ms.

Completion time: 2025-11-24T21:30:44.4053453+07:00
```

Kode ini berfungsi sebagai tahap Finalisasi dan Pembersihan Sesi (Cleanup). Setelah seluruh rangkaian pengujian performa yang intensif selesai dilakukan, kode ini bertugas untuk "mematikan mesin diagnostik". Perintah SET STATISTICS TIME OFF dan SET STATISTICS IO OFF menginstruksikan

SQL Server untuk berhenti melacak dan melaporkan metrik mendetail mengenai durasi CPU dan aktivitas Disk I/O di tab "Messages".
Ini adalah praktik terbaik (best practice) dalam pengelolaan database: selalu kembalikan konfigurasi sesi ke mode normal setelah troubleshooting atau benchmarking selesai. Tujuannya adalah agar eksekusi query selanjutnya berjalan bersih tanpa membebani log dengan metadata teknis yang tidak lagi diperlukan.

2. Performance Benchmarks

Hasil pengujian awal (Baseline):

Query Type	Target Time	Actual Time	Status	Index Used
1. Risk Profile Aggregation	< 1s	0.05s	PASS	NCCIX_Fact_Analytic (Columnstore)
2. Monthly Trend Analysis	< 2s	0.08s	PASS	CIX_Fact_Temuan_Waktu (Clustered)
3. Auditor Drill-down	< 3s	0.12s	PASS	IX_Fact_Auditor_Key (Non-Clustered)

Query Optimization Recommendations

1. Partition Pruning: Pastikan klausa WHERE Tahun_Audit = ... selalu digunakan dalam query laporan tahunan untuk memanfaatkan partisi.
2. Covering Indexes: Untuk query drill-down yang sering digunakan, tambahkan kolom Status_Tindak_Lanjut ke INCLUDE pada index Auditor jika diperlukan.