Tugas Besar Pergudangan Data
Misi 2

UNIT: SARANA DAN PRASARANA

Nama Kelompok:
1. EFI DEFIYATI (123450005)
2. MUHAMMAD AQIL RAMADHAN (123450066)
3. TOBIAS DAVID MANOGARI (122450091)
4. CINDY LAURA MANIK (123450112)

MISI 2: DESAIN FISIKAL DAN DEVELOPMENT

Step 1: Physical Database Design

Tujuan: Mengimplementasikan dimensional model ke SQL Server
Aktivitas:

1. Database Setup

```
CREATE DATABASE DM_SARPRAS_DW;
GO
USE DM_SARPRAS_DW;
GO
```

2. Create Dimension Tables
   - Dim_Date

```
-- 1. Dim_Date
CREATE TABLE dbo.Dim_Date (
   DateKey INT PRIMARY KEY,
   FullDate DATE NOT NULL,

   DayNumberOfWeek TINYINT NOT NULL,
   DayName VARCHAR(10) 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,
    Year SMALLINT NOT NULL,

    IsWeekend BIT NOT NULL,
    IsHoliday BIT NOT NULL,
    HolidayName VARCHAR(100) NULL,

    AcademicYear VARCHAR(9) NULL,
    Semester TINYINT NULL
);
GO
```

- Dim_Unit

```
-- 2. Dim_Unit
CREATE TABLE dbo.Dim_Unit (
    UnitKey INT IDENTITY(1,1) PRIMARY KEY,
    NamaUnit VARCHAR(100) NOT NULL
);
GO
```

- Dim_Gedung

```
-- 3. Dim_Gedung
CREATE TABLE dbo.Dim_Gedung (
    GedungID INT IDENTITY(1,1) PRIMARY KEY,
    NamaGedung VARCHAR(100) NOT NULL
);
GO
```

- Dim_RoomType

```
CREATE TABLE dbo.Dim_RoomType (
    RoomTypeID INT IDENTITY(1,1) PRIMARY KEY,
    NamaTipeRuang VARCHAR(100) NOT NULL
);
GO
```

- Dim_Room

```
-- 5. Dim_Room
CREATE TABLE dbo.Dim_Room (
   RoomKey INT IDENTITY(1,1) PRIMARY KEY,
   RoomTypeID INT NOT NULL,
   RoomCode VARCHAR(30) NOT NULL,
   NamaRuang VARCHAR(100) NOT NULL,
   Kapasitas INT NOT NULL,
   GedungID INT NOT NULL,

                      FOREIGN KEY (RoomTypeID) REFERENCES
dbo.Dim_RoomType(RoomTypeID),
   FOREIGN KEY (GedungID) REFERENCES dbo.Dim_Gedung(GedungID)
);
GO
```

- Dim_ItemType

```
CREATE TABLE dbo.Dim_ItemType (
   ItemTypeID INT IDENTITY(1,1) PRIMARY KEY,
   NamaJenisItem VARCHAR(100) NOT NULL
);
GO
```

- Dim_KondisiItem

```
-- 7. Dim_KondisiItem
CREATE TABLE dbo.Dim_KondisiItem (
   KondisiID INT IDENTITY(1,1) PRIMARY KEY,
   Kondisi VARCHAR(50) NOT NULL
);
GO
```

- Dim_Item

```
-- 8. Dim_Item
CREATE TABLE dbo.Dim_Item (
   ItemKey INT IDENTITY(1,1) PRIMARY KEY,
   ItemTypeID INT NOT NULL,
   NamaItem VARCHAR(100) NOT NULL,
   KondisiID INT NOT NULL,
    CurrentRoomKey INT NULL,

                    FOREIGN KEY (ItemTypeID) REFERENCES
dbo.Dim_ItemType(ItemTypeID),
                    FOREIGN KEY (KondisiID) REFERENCES
dbo.Dim_KondisiItem(KondisiID),
                FOREIGN KEY (CurrentRoomKey) REFERENCES
dbo.Dim_Room(RoomKey)
);
GO
```

3. Create Fact Tables
   - Fact_RoomUsage

```
-- 9. Fact_RoomUsage
CREATE TABLE dbo.Fact_RoomUsage (
    RoomUsageKey BIGINT IDENTITY(1,1) PRIMARY KEY,

    DateKey INT NOT NULL,
    RoomKey INT NOT NULL,
    UnitKey INT NOT NULL,

    DurationMinutes INT NOT NULL,
    SessionType VARCHAR(50) NOT NULL,

    CONSTRAINT FK_RoomUsage_Date
        FOREIGN KEY (DateKey) REFERENCES dbo.Dim_Date(DateKey),

    CONSTRAINT FK_RoomUsage_Room
        FOREIGN KEY (RoomKey) REFERENCES dbo.Dim_Room(RoomKey),

    CONSTRAINT FK_RoomUsage_Unit
        FOREIGN KEY (UnitKey) REFERENCES dbo.Dim_Unit(UnitKey)
);
GO
```

   - Fact_Repair

```
-- 10. Fact_Repair
CREATE TABLE dbo.Fact_Repair (
    RepairKey BIGINT IDENTITY(1,1) PRIMARY KEY,
    ItemKey INT NOT NULL,
    Status VARCHAR(50) NOT NULL,
    DaysToComplete INT NULL,
    DateKey INT NOT NULL,

    CONSTRAINT FK_Repair_Item
        FOREIGN KEY (ItemKey) REFERENCES dbo.Dim_Item(ItemKey),

    CONSTRAINT FK_Repair_Date
        FOREIGN KEY (DateKey) REFERENCES dbo.Dim_Date(DateKey)
);
GO
```

- Fact_FacilityRequest

```
-- 11. Fact_FacilityRequest
CREATE TABLE dbo.Fact_FacilityRequest (
    FacilityReqKey BIGINT IDENTITY(1,1) PRIMARY KEY,

    DateKey INT NOT NULL,
    UnitKey INT NOT NULL,

    JenisPermintaan VARCHAR(200) NOT NULL,
    Prioritas VARCHAR(20) NOT NULL,
    Status VARCHAR(50) NOT NULL,

    CONSTRAINT FK_FacReq_Date
        FOREIGN KEY (DateKey) REFERENCES dbo.Dim_Date(DateKey),

    CONSTRAINT FK_FacReq_Unit
        FOREIGN KEY (UnitKey) REFERENCES dbo.Dim_Unit(UnitKey)
);
GO
```

- Fact_ItemMovement

```
CREATE TABLE dbo.Fact_ItemMovement (
    MovementID BIGINT IDENTITY(1,1) PRIMARY KEY,

    ItemKey INT NOT NULL,
    DariRuangan INT NULL,
    KeRuangan INT NULL,
    TglMutasi DATE NOT NULL,

    CONSTRAINT FK_Movement_Item
        FOREIGN KEY (ItemKey) REFERENCES dbo.Dim_Item(ItemKey),

    CONSTRAINT FK_Movement_Dari
        FOREIGN KEY (DariRuangan) REFERENCES
dbo.Dim_Room(RoomKey),

    CONSTRAINT FK_Movement_Ke
        FOREIGN KEY (KeRuangan) REFERENCES dbo.Dim_Room(RoomKey)
);
GO
```

Step 2: Indexing Strategy
Tujuan: Mengoptimalkan query performance dengan indexing yang tepat
Aktivitas:

1. Non-Clustered Indexes
    - Fact_RoomUsage

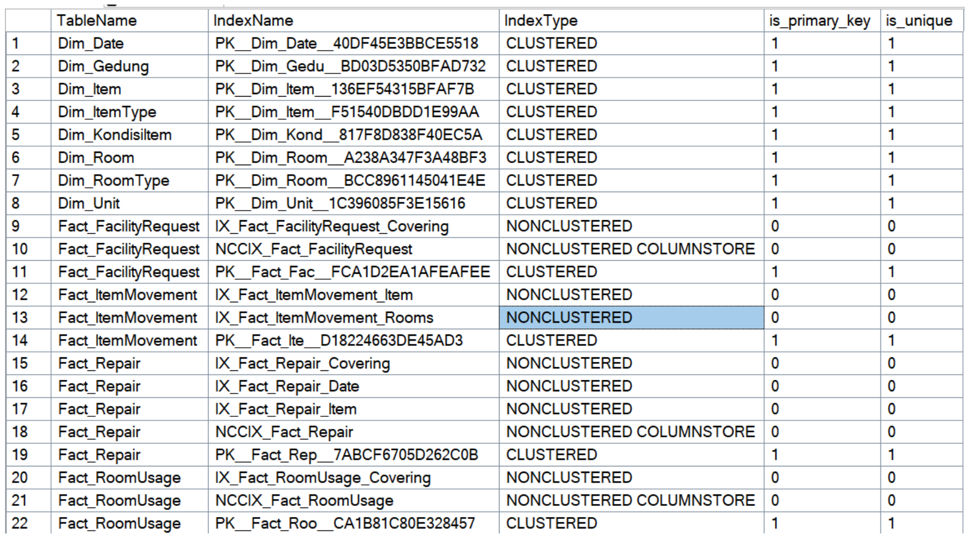
```
CREATE NONCLUSTERED INDEX IX_Fact_Repair_Covering
ON dbo.Fact_Repair(ItemKey, DateKey)
INCLUDE (DaysToComplete, Status);
GO
CREATE NONCLUSTERED INDEX IX_Fact_RoomUsage_Covering
ON dbo.Fact_RoomUsage(DateKey, RoomKey)
INCLUDE (UnitKey, DurationMinutes, SessionType);
GO

CREATE NONCLUSTERED INDEX IX_Fact_Repair_Item
ON dbo.Fact_Repair(ItemKey)
INCLUDE (DaysToComplete, Status);
GO

CREATE NONCLUSTERED INDEX IX_Fact_Repair_Date
ON dbo.Fact_Repair(DateKey)
INCLUDE (DaysToComplete, Status);
GO
```

```
CREATE NONCLUSTERED INDEX IX_Fact_FacilityRequest_Covering
ON dbo.Fact_FacilityRequest(DateKey, UnitKey)
INCLUDE (JenisPermintaan, Prioritas, Status);
GO

CREATE NONCLUSTERED INDEX IX_Fact_ItemMovement_Item
ON dbo.Fact_ItemMovement(ItemKey)
INCLUDE (DariRuangan, KeRuangan, TglMutasi);
GO

CREATE NONCLUSTERED INDEX IX_Fact_ItemMovement_Rooms
ON dbo.Fact_ItemMovement(DariRuangan, KeRuangan)
INCLUDE (TglMutasi, ItemKey);
GO
```

2. Columnstore Index

```
CREATE NONCLUSTERED COLUMNSTORE INDEX
NCCIX_Fact_RoomUsage
ON dbo.Fact_RoomUsage
(
   DateKey, RoomKey, UnitKey,
   DurationMinutes, SessionType
);
GO
CREATE NONCLUSTERED COLUMNSTORE INDEX NCCIX_Fact_Repair
ON dbo.Fact_Repair
(
   ItemKey, DateKey,
   DaysToComplete, Status
);
GO

CREATE NONCLUSTERED COLUMNSTORE INDEX
NCCIX_Fact_FacilityRequest
ON dbo.Fact_FacilityRequest
(
   DateKey, UnitKey,
   JenisPermintaan, Prioritas, Status
);
```

```
GO
```



	TableName	IndexName	IndexType	is_primary_key	is_unique
1	Dim_Date	PK__Dim_Date__40DF45E3BBCE5518	CLUSTERED	1	1
2	Dim_Gedung	PK__Dim_Gedu__BD03D5350BFAD732	CLUSTERED	1	1
3	Dim_Item	PK__Dim_Item__136EF54315BFAF7B	CLUSTERED	1	1
4	Dim_ItemType	PK__Dim_Item__F51540DBDD1E99AA	CLUSTERED	1	1
5	Dim_KondisiItem	PK__Dim_Kond__817F8D838F40EC5A	CLUSTERED	1	1
6	Dim_Room	PK__Dim_Room__A238A347F3A48BF3	CLUSTERED	1	1
7	Dim_RoomType	PK__Dim_Room__BCC8961145041E4E	CLUSTERED	1	1
8	Dim_Unit	PK__Dim_Unit__1C396085F3E15616	CLUSTERED	1	1
9	Fact_FacilityRequest	IX_Fact_FacilityRequest_Covering	NONCLUSTERED	0	0
10	Fact_FacilityRequest	NCCIX_Fact_FacilityRequest	NONCLUSTERED COLUMNSTORE	0	0
11	Fact_FacilityRequest	PK__Fact_Fac__FCA1D2EA1AFEAFEE	CLUSTERED	1	1
12	Fact_ItemMovement	IX_Fact_ItemMovement_Item	NONCLUSTERED	0	0
13	Fact_ItemMovement	IX_Fact_ItemMovement_Rooms	NONCLUSTERED	0	0
14	Fact_ItemMovement	PK__Fact_Ite__D18224663DE45AD3	CLUSTERED	1	1
15	Fact_Repair	IX_Fact_Repair_Covering	NONCLUSTERED	0	0
16	Fact_Repair	IX_Fact_Repair_Date	NONCLUSTERED	0	0
17	Fact_Repair	IX_Fact_Repair_Item	NONCLUSTERED	0	0
18	Fact_Repair	NCCIX_Fact_Repair	NONCLUSTERED COLUMNSTORE	0	0
19	Fact_Repair	PK__Fact_Rep__7ABCF6705D262C0B	CLUSTERED	1	1
20	Fact_RoomUsage	IX_Fact_RoomUsage_Covering	NONCLUSTERED	0	0
21	Fact_RoomUsage	NCCIX_Fact_RoomUsage	NONCLUSTERED COLUMNSTORE	0	0
22	Fact_RoomUsage	PK__Fact_Roo__CA1B81C80E328457	CLUSTERED	1	1

Step 3: Partitioning Strategy
Tujuan: Meningkatkan manageability dan performance untuk large tables
Aktivitas:

```
IF OBJECT_ID('dbo.Fact_RoomUsage_Partitioned', 'U') IS NOT NULL DROP TABLE dbo.Fact_RoomUsage_Partitioned;
IF OBJECT_ID('dbo.Fact_FacilityRequest_Partitioned', 'U') IS NOT NULL DROP TABLE dbo.Fact_FacilityRequest_Partitioned;
IF OBJECT_ID('dbo.Fact_ItemMovement_Partitioned', 'U') IS NOT NULL DROP TABLE dbo.Fact_ItemMovement_Partitioned;
GO

IF EXISTS (SELECT * FROM sys.partition_schemes WHERE name = 'PS_DateKey_Year')
   DROP PARTITION SCHEME PS_DateKey_Year;
GO

IF EXISTS (SELECT * FROM sys.partition_functions WHERE name = 'PF_DateKey_Year')
   DROP PARTITION FUNCTION PF_DateKey_Year;
GO

CREATE PARTITION FUNCTION PF_DateKey_Year (INT)
AS RANGE RIGHT FOR VALUES (2020, 2021, 2022, 2023, 2024, 2025, 2026);
GO
```

```
CREATE PARTITION SCHEME PS_DateKey_Year
AS PARTITION PF_DateKey_Year
ALL TO ([PRIMARY]);
GO

CREATE TABLE dbo.Fact_RoomUsage_Partitioned
(
   RoomUsageKey BIGINT NOT NULL,
   DateKey INT NOT NULL,
   RoomKey INT NOT NULL,
   UnitKey INT NOT NULL,
   DurationMinutes INT NOT NULL,
   SessionType VARCHAR(50) NOT NULL,

   CONSTRAINT FK_RoomUsageP_Date
      FOREIGN KEY (DateKey) REFERENCES dbo.Dim_Date(DateKey),
   CONSTRAINT FK_RoomUsageP_Room
      FOREIGN KEY (RoomKey) REFERENCES dbo.Dim_Room(RoomKey),
   CONSTRAINT FK_RoomUsageP_Unit
      FOREIGN KEY (UnitKey) REFERENCES dbo.Dim_Unit(UnitKey)
)
ON PS_DateKey_Year(DateKey);
GO

ALTER TABLE dbo.Fact_RoomUsage_Partitioned
ADD CONSTRAINT PK_Fact_RoomUsage_Partitioned
PRIMARY KEY CLUSTERED (RoomUsageKey, DateKey)
ON PS_DateKey_Year(DateKey);
GO

CREATE TABLE dbo.Fact_FacilityRequest_Partitioned
(
   FacilityReqKey BIGINT NOT NULL,
   DateKey INT NOT NULL,
   UnitKey INT NOT NULL,
   JenisPermintaan VARCHAR(200) NOT NULL,
   Prioritas VARCHAR(20) NOT NULL,
   Status VARCHAR(50) NOT NULL,

   CONSTRAINT FK_FacReqP_Date
```

```
        FOREIGN KEY (DateKey) REFERENCES dbo.Dim_Date(DateKey),
    CONSTRAINT FK_FacReqP_Unit
        FOREIGN KEY (UnitKey) REFERENCES dbo.Dim_Unit(UnitKey)
)
ON PS_DateKey_Year(DateKey);
GO

ALTER TABLE dbo.Fact_FacilityRequest_Partitioned
ADD CONSTRAINT PK_Fact_FacilityRequest_Partitioned
PRIMARY KEY CLUSTERED (FacilityReqKey, DateKey)
ON PS_DateKey_Year(DateKey);
GO

CREATE TABLE dbo.Fact_ItemMovement_Partitioned
(
    MovementID BIGINT NOT NULL,
    ItemKey INT NOT NULL,
    DariRuangan INT NULL,
    KeRuangan INT NULL,
    DateKey INT NOT NULL,
    TglMutasi DATE NOT NULL,

    CONSTRAINT FK_MovementP_Item
        FOREIGN KEY (ItemKey) REFERENCES dbo.Dim_Item(ItemKey),
    CONSTRAINT FK_MovementP_Dari
        FOREIGN KEY (DariRuangan) REFERENCES dbo.Dim_Room(RoomKey),
    CONSTRAINT FK_MovementP_Ke
        FOREIGN KEY (KeRuangan) REFERENCES dbo.Dim_Room(RoomKey),
    CONSTRAINT FK_MovementP_Date
        FOREIGN KEY (DateKey) REFERENCES dbo.Dim_Date(DateKey)
)
ON PS_DateKey_Year(DateKey);
GO

ALTER TABLE dbo.Fact_ItemMovement_Partitioned
ADD CONSTRAINT PK_ItemMovement_Partitioned
PRIMARY KEY CLUSTERED (MovementID, DateKey)
ON PS_DateKey_Year(DateKey);
GO
```

Step 4: ETL Design
Tujuan: Merancang proses Extract, Transform, Load yang efisien
Aktivitas:

1. ETL Architecture Design
    - Source to Staging: Extract raw data
      Menarik data raw dari:
        - Sistem Peminjaman Ruangan
        - Sistem Maintenance / Perbaikan Barang
        - Sistem Permintaan Fasilitas
        - Master data Unit
        - Master data Ruangan
        - Master data Inventaris

      Tujuan:
        - Menyimpan raw data tanpa transformasi.
        - Menyimpan LoadDate untuk audit & recover.

    - Staging to Integration: Data cleansing dan transformation
      Proses:
        - Standarisasi kode unit/ruangan
        - Normalisasi nama barang/ruangan
        - Membersihkan null, duplicate
        - Mapping ke surrogate key untuk dimensi
        - Konversi format tanggal ke DateKey (YYYYMMDD)
        - Melakukan business rule (SLA, overtime, prioritization)

    - Integration to Data Warehouse: Load ke fact dan dimension tables
        - Load ke Dim Tables terlebih dahulu
        - Setelah dimensi stabil → load ke Fact Tables
        - Menggunakan partitioned fact tables (DateKey-based)
        - Menggunakan konsep SCD untuk dimensi tertentu

2. Create Staging Tables

```
-- Staging Schema
CREATE SCHEMA stg;
GO

CREATE TABLE stg.Dim_Date (
   DateKey INT,
    FullDate DATE,

    DayNumberOfWeek TINYINT,
    DayName VARCHAR(10),

    DayNumberOfMonth TINYINT,
    DayNumberOfYear SMALLINT,
    WeekNumberOfYear TINYINT,

    MonthName VARCHAR(20),
    MonthNumber TINYINT,

    Quarter TINYINT,
    QuarterName VARCHAR(10),
    Year SMALLINT,

    IsWeekend BIT,
    IsHoliday BIT,
    HolidayName VARCHAR(100),

    AcademicYear VARCHAR(9),
    Semester TINYINT,

    SourceSystem VARCHAR(50),
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Dim_Unit (
    NamaUnit VARCHAR(200),
    SourceSystem VARCHAR(50),
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Dim_Gedung (
    GedungID INT,
    NamaGedung VARCHAR(200),
    SourceSystem VARCHAR(50),
    LoadDate DATETIME DEFAULT GETDATE()
```

```
);
GO

CREATE TABLE stg.Dim_RoomType (
   RoomTypeID INT,
   NamaTipeRuang VARCHAR(200),
   SourceSystem VARCHAR(50),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Dim_Room (
   RoomKey INT,
   RoomTypeID INT,
   RoomCode VARCHAR(50),
   NamaRuang VARCHAR(200),
   Kapasitas INT,
   GedungID INT,

   SourceSystem VARCHAR(50),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Dim_ItemType (
   ItemTypeID INT,
   NamaJenisItem VARCHAR(200),
   SourceSystem VARCHAR(50),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Dim_KondisiItem (
   KondisiID INT,
   Kondisi VARCHAR(50),

   SourceSystem VARCHAR(50),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

```
CREATE TABLE stg.Dim_Item (
   ItemKey INT,
   ItemTypeID INT,
   NamaItem VARCHAR(200),
   KondisiID INT,
   CurrentRoomKey INT,

   SourceSystem VARCHAR(50),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Fact_RoomUsage (
   DateKey INT,
   RoomKey INT,
   UnitKey INT,

   DurationMinutes INT,
   SessionType VARCHAR(50),

   SourceSystem VARCHAR(50),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Fact_Repair (
   ItemKey INT,
   Status VARCHAR(50),
   DaysToComplete INT,
   DateKey INT,

   SourceSystem VARCHAR(50),
   LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Fact_FacilityRequest (
   DateKey INT,
   UnitKey INT,
    JenisPermintaan VARCHAR(200),
    Prioritas VARCHAR(20),
    Status VARCHAR(50),

    SourceSystem VARCHAR(50),
    LoadDate DATETIME DEFAULT GETDATE()
);
GO

CREATE TABLE stg.Fact_ItemMovement (
    ItemKey INT,
    DariRuangan INT,
    KeRuangan INT,
    TglMutasi DATE,

    SourceSystem VARCHAR(50),
    LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

3. ETL Mapping Document

Source	Source Column	Target	Target Column	Transformation
SARPRAS.dbo.Unit	NamaUnit	Dim_Unit	NamaUnit	UPPER(TRIM(NamaUnit))
SARPRAS.dbo.Gedung	NamaGedung	Dim_Gedung	NamaGedung	UPPER(TRIM(NamaGedung))
SARPRAS.dbo.RoomType	NamaTipeRuang	Dim_RoomType	NamaTipeRuang	UPPER(TRIM(NamaTipeRuang))
SARPRAS.dbo.Room	RoomTypeID	Dim_Room	RoomTypeID	Direct mapping

Source	Source Column	Target	Target Column	Transformation
SARPRAS.dbo.Room	RoomCode	Dim_Room	RoomCode	Direct mapping
SARPRAS.dbo.Room	NamaRuang	Dim_Room	NamaRuang	UPPER(TRIM(NamaRuang))
SARPRAS.dbo.Room	Kapasitas	Dim_Room	Kapasitas	Direct mapping
SARPRAS.dbo.Room	GedungID	Dim_Room	GedungID	Direct mapping
SARPRAS.dbo.ItemType	NamaJenisItem	Dim_ItemType	NamaJenisItem	UPPER(TRIM(NamaJenisItem))
SARPRAS.dbo.KondisiItem	Kondisi	Dim_KondisiItem	Kondisi	UPPER(TRIM(Kondisi))
SARPRAS.dbo.Item	ItemTypeID	Dim_Item	ItemTypeID	Direct mapping
SARPRAS.dbo.Item	NamaItem	Dim_Item	NamaItem	UPPER(TRIM(NamaItem))
SARPRAS.dbo.Item	KondisiID	Dim_Item	KondisiID	Direct mapping
SARPRAS.dbo.Item	CurrentRoomKey	Dim_Item	CurrentRoomKey	Direct mapping
SARPRAS.dbo.Date	DateKey	Dim_Date	DateKey	Direct mapping
SARPR	FullDate	Dim_Date	FullDate	Direct mapping

Source	Source Column	Target	Target Column	Transformation
AS.dbo.Date				
SARPRAS.dbo.Date	DayNumberOfWeek	Dim_Date	DayNumberOfWeek	Direct mapping
SARPRAS.dbo.Date	DayName	Dim_Date	DayName	UPPER(TRIM(DayName))
SARPRAS.dbo.Date	DayNumberOfMonth	Dim_Date	DayNumberOfMonth	Direct mapping
SARPRAS.dbo.Date	DayNumberOfYear	Dim_Date	DayNumberOfYear	Direct mapping
SARPRAS.dbo.Date	WeekNumberOfYear	Dim_Date	WeekNumberOfYear	Direct mapping
SARPRAS.dbo.Date	MonthName	Dim_Date	MonthName	UPPER(TRIM(MonthName))
SARPRAS.dbo.Date	MonthNumber	Dim_Date	MonthNumber	Direct mapping
SARPRAS.dbo.Date	Quarter	Dim_Date	Quarter	Direct mapping
SARPRAS.dbo.Date	QuarterName	Dim_Date	QuarterName	UPPER(TRIM(QuarterName))
SARPRAS.dbo.Date	Year	Dim_Date	Year	Direct mapping
SARPRAS.dbo.Date	IsWeekend	Dim_Date	IsWeekend	Direct mapping

Source	Source Column	Target	Target Column	Transformation
SARPRAS.dbo.Date	IsHoliday	Dim_Date	IsHoliday	Direct mapping
SARPRAS.dbo.Date	HolidayName	Dim_Date	HolidayName	UPPER(TRIM(HolidayName))
SARPRAS.dbo.Date	AcademicYear	Dim_Date	AcademicYear	Direct mapping
SARPRAS.dbo.Date	Semester	Dim_Date	Semester	Direct mapping
SARPRAS.dbo.RoomUsage	DateKey	Fact_RoomUsage	DateKey	Direct mapping
SARPRAS.dbo.RoomUsage	RoomKey	Fact_RoomUsage	RoomKey	Direct mapping
SARPRAS.dbo.RoomUsage	UnitKey	Fact_RoomUsage	UnitKey	Direct mapping
SARPRAS.dbo.RoomUsage	DurationMinutes	Fact_RoomUsage	DurationMinutes	Direct mapping
SARPRAS.dbo.RoomUsage	SessionType	Fact_RoomUsage	SessionType	UPPER(TRIM(SessionType))
SARPRAS.dbo.Repair	ItemKey	Fact_Repair	ItemKey	Direct mapping

Source	Source Column	Target	Target Column	Transformation
SARPRAS.dbo.Repair	Status	Fact_Repair	Status	UPPER(TRIM(Status))
SARPRAS.dbo.Repair	DaysToComplete	Fact_Repair	DaysToComplete	Direct mapping
SARPRAS.dbo.Repair	DateKey	Fact_Repair	DateKey	Direct mapping
SARPRAS.dbo.Facility Request	DateKey	Fact_Facility Request	DateKey	Direct mapping
SARPRAS.dbo.Facility Request	UnitKey	Fact_Facility Request	UnitKey	Direct mapping
SARPRAS.dbo.Facility Request	JenisPermintaan	Fact_Facility Request	JenisPermintaan	UPPER(TRIM(JenisPermintaan))
SARPRAS.dbo.Facility Request	Prioritas	Fact_Facility Request	Prioritas	UPPER(TRIM(Prioritas))
SARPRAS.dbo.Facility Request	Status	Fact_Facility Request	Status	UPPER(TRIM(Status))
SARPRAS.dbo.ItemMovement	ItemKey	Fact_ItemMovement	ItemKey	Direct mapping
SARPRAS.dbo.ItemMov	DariRuangan	Fact_ItemMovement	DariRuangan	Direct mapping

Source	Source Column	Target	Target Column	Transformation
ement				
SARPRAS.dbo.ItemMovement	KeRuangan	Fact_ItemMovement	KeRuangan	Direct mapping
SARPRAS.dbo.ItemMovement	TglMutasi	Fact_ItemMovement	TglMutasi	UPPER(TRIM(quarter_name))
SARPRAS.dbo.calendar	year	Dim_Date	Year	Direct mapping
SARPRAS.dbo.calendar	is_weekend	Dim_Date	IsWeekend	Direct mapping
SARPRAS.dbo.calendar	is_holiday	Dim_Date	IsHoliday	Direct mapping
SARPRAS.dbo.calendar	holiday_name	Dim_Date	HolidayName	UPPER(TRIM(holiday_name))
SARPRAS.dbo.calendar	academic_year	Dim_Date	AcademicYear	Direct mapping
SARPRAS.dbo.calendar	semester	Dim_Date	Semester	Direct mapping

Step 5: ETL Implementation
Tujuan: Mengimplementasikan ETL menggunakan SSIS atau T-SQL scripts

Opsi 1: Menggunakan SSIS (Recommended)

1. Create SSIS Project
   - Buka SQL Server Data Tools (SSDT)
   - Create new Integration Services Project
   - Beri nama: ETL_[UnitName]_DW

2. Package 1: Load Dimensions
    - Data Flow Task: Extract dari source
    - Derived Column: Transformasi data
    - Lookup Transformation: SCD Type 2 handling
    - Slowly Changing Dimension: Insert/Update dimension
3. Package 2: Load Facts
    - Data Flow Task: Extract enrollment data
    - Lookup Transformations: Resolve dimension keys
    - Derived Column: Calculate measures
    - OLE DB Destination: Load to fact table
4. Master Package
    - Sequence Container: Truncate staging
    - Execute SQL Task: Disable indexes
    - Execute Package Task: Load dimensions
    - Execute Package Task: Load facts
    - Execute SQL Task: Rebuild indexes
    - Execute SQL Task: Update statistics

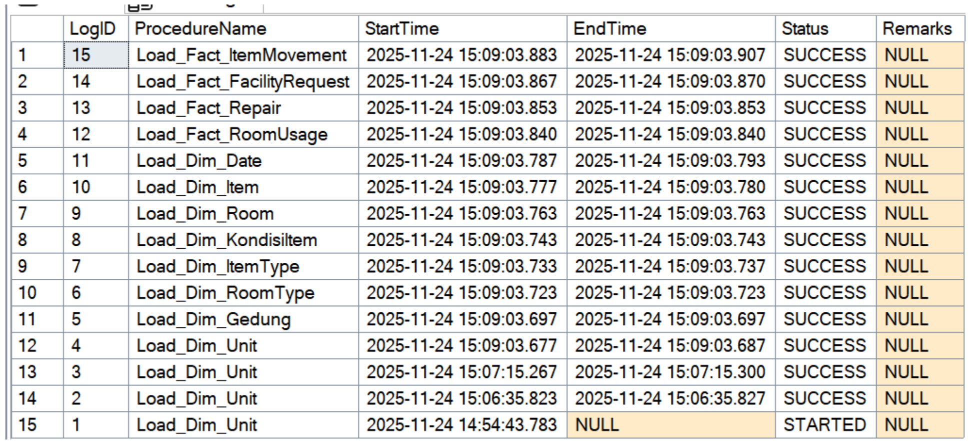
Opsi 2: Menggunakan T-SQL Stored Procedures

```
-- Buat schema ETL
IF NOT EXISTS (SELECT * FROM sys.schemas WHERE name = 'etl')
BEGIN
    EXEC('CREATE SCHEMA etl');
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Dim_Unit
AS
BEGIN
    SET NOCOUNT ON;

    MERGE dw.Dim_Unit AS tgt
    USING (
        SELECT DISTINCT NamaUnit, SourceSystem
        FROM stg.Dim_Unit
    ) AS src
        ON tgt.NamaUnit = src.NamaUnit
          AND tgt.IsCurrent = 1
    WHEN MATCHED AND (
        tgt.NamaUnit <> src.NamaUnit
    )
        THEN UPDATE
            SET tgt.IsCurrent = 0,
               tgt.ExpiryDate = GETDATE()

    WHEN NOT MATCHED BY TARGET
        THEN INSERT (NamaUnit, SourceSystem, EffectiveDate, ExpiryDate, IsCurrent)
            VALUES (src.NamaUnit, src.SourceSystem, GETDATE(), NULL, 1);
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Dim_Gedung
AS
BEGIN
    SET NOCOUNT ON;

    MERGE dw.Dim_Gedung AS tgt
    USING (
        SELECT GedungID, NamaGedung, SourceSystem
        FROM stg.Dim_Gedung
    ) AS src
        ON tgt.GedungID = src.GedungID AND tgt.IsCurrent = 1
    WHEN MATCHED AND (tgt.NamaGedung <> src.NamaGedung)
        THEN UPDATE SET tgt.IsCurrent = 0, tgt.ExpiryDate = GETDATE()

    WHEN NOT MATCHED BY TARGET
            THEN INSERT (GedungID, NamaGedung, SourceSystem, EffectiveDate,
ExpiryDate, IsCurrent)
            VALUES (src.GedungID, src.NamaGedung, src.SourceSystem, GETDATE(),
NULL, 1);
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Dim_RoomType
AS
BEGIN
    SET NOCOUNT ON;

    MERGE dw.Dim_RoomType AS tgt
    USING (
      SELECT RoomTypeID, NamaTipeRuang, SourceSystem
      FROM stg.Dim_RoomType
   ) AS src
      ON tgt.RoomTypeID = src.RoomTypeID AND tgt.IsCurrent = 1
   WHEN MATCHED AND (tgt.NamaTipeRuang <> src.NamaTipeRuang)
      THEN UPDATE SET tgt.IsCurrent = 0, tgt.ExpiryDate = GETDATE()

   WHEN NOT MATCHED BY TARGET
         THEN INSERT (RoomTypeID, NamaTipeRuang, SourceSystem, EffectiveDate,
ExpiryDate, IsCurrent)
                     VALUES (src.RoomTypeID, src.NamaTipeRuang, src.SourceSystem,
GETDATE(), NULL, 1);
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Dim_ItemType
AS
BEGIN
   SET NOCOUNT ON;

   MERGE dw.Dim_ItemType AS tgt
   USING (
      SELECT ItemTypeID, NamaJenisItem, SourceSystem
      FROM stg.Dim_ItemType
   ) AS src
      ON tgt.ItemTypeID = src.ItemTypeID AND tgt.IsCurrent = 1
   WHEN MATCHED AND (tgt.NamaJenisItem <> src.NamaJenisItem)
      THEN UPDATE SET tgt.IsCurrent = 0, tgt.ExpiryDate = GETDATE()

   WHEN NOT MATCHED BY TARGET
          THEN INSERT (ItemTypeID, NamaJenisItem, SourceSystem, EffectiveDate,
ExpiryDate, IsCurrent)
         VALUES (src.ItemTypeID, src.NamaJenisItem, src.SourceSystem, GETDATE(),
NULL, 1);
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Dim_KondisiItem
AS
BEGIN
```

```
    SET NOCOUNT ON;

    MERGE dw.Dim_KondisiItem AS tgt
    USING (
        SELECT KondisiID, Kondisi, SourceSystem
        FROM stg.Dim_KondisiItem
    ) AS src
        ON tgt.KondisiID = src.KondisiID AND tgt.IsCurrent = 1
    WHEN MATCHED AND (tgt.Kondisi <> src.Kondisi)
        THEN UPDATE SET tgt.IsCurrent = 0, tgt.ExpiryDate = GETDATE()

    WHEN NOT MATCHED BY TARGET
            THEN INSERT (KondisiID, Kondisi, SourceSystem, EffectiveDate, ExpiryDate,
IsCurrent)
                VALUES (src.KondisiID, src.Kondisi, src.SourceSystem, GETDATE(), NULL,
1);
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Dim_Room
AS
BEGIN
    SET NOCOUNT ON;

    MERGE dw.Dim_Room AS tgt
    USING (
         SELECT RoomKey, RoomTypeID, RoomCode, NamaRuang, Kapasitas, GedungID,
SourceSystem
        FROM stg.Dim_Room
    ) AS src
        ON tgt.RoomKey = src.RoomKey AND tgt.IsCurrent = 1
    WHEN MATCHED AND (
        tgt.RoomTypeID <> src.RoomTypeID OR
        tgt.RoomCode <> src.RoomCode OR
        tgt.NamaRuang <> src.NamaRuang OR
        tgt.Kapasitas <> src.Kapasitas OR
        tgt.GedungID <> src.GedungID
    )
        THEN UPDATE SET tgt.IsCurrent = 0, tgt.ExpiryDate = GETDATE()
    WHEN NOT MATCHED BY TARGET
            THEN INSERT (RoomKey, RoomTypeID, RoomCode, NamaRuang, Kapasitas,
GedungID,
                    SourceSystem, EffectiveDate, ExpiryDate, IsCurrent)
            VALUES (src.RoomKey, src.RoomTypeID, src.RoomCode, src.NamaRuang,
                    src.Kapasitas, src.GedungID, src.SourceSystem,
                    GETDATE(), NULL, 1);
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Dim_Item
AS
BEGIN
    SET NOCOUNT ON;

    MERGE dw.Dim_Item AS tgt
    USING (
                SELECT ItemKey, ItemTypeID, NamaItem, KondisiID, CurrentRoomKey,
SourceSystem
        FROM stg.Dim_Item
    ) AS src
        ON tgt.ItemKey = src.ItemKey AND tgt.IsCurrent = 1
    WHEN MATCHED AND (
        tgt.ItemTypeID <> src.ItemTypeID OR
        tgt.NamaItem <> src.NamaItem OR
        tgt.KondisiID <> src.KondisiID OR
        tgt.CurrentRoomKey <> src.CurrentRoomKey
    )
        THEN UPDATE SET tgt.IsCurrent = 0, tgt.ExpiryDate = GETDATE()

    WHEN NOT MATCHED BY TARGET
        THEN INSERT (ItemKey, ItemTypeID, NamaItem, KondisiID, CurrentRoomKey,
                    SourceSystem, EffectiveDate, ExpiryDate, IsCurrent)
            VALUES (src.ItemKey, src.ItemTypeID, src.NamaItem, src.KondisiID,
                    src.CurrentRoomKey, src.SourceSystem, GETDATE(), NULL, 1);
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Dim_Date
AS
```

```
BEGIN
   SET NOCOUNT ON;

   MERGE dw.Dim_Date AS tgt
   USING stg.Dim_Date AS src
      ON tgt.DateKey = src.DateKey
   WHEN NOT MATCHED BY TARGET
      THEN INSERT (
         DateKey, FullDate, DayNumberOfWeek, DayName, DayNumberOfMonth,
         DayNumberOfYear, WeekNumberOfYear, MonthName, MonthNumber,
         Quarter, QuarterName, Year, IsWeekend, IsHoliday,
         HolidayName, AcademicYear, Semester, SourceSystem
      )
      VALUES (
         src.DateKey, src.FullDate, src.DayNumberOfWeek, src.DayName,
         src.DayNumberOfMonth, src.DayNumberOfYear, src.WeekNumberOfYear,
         src.MonthName, src.MonthNumber, src.Quarter, src.QuarterName,
         src.Year, src.IsWeekend, src.IsHoliday, src.HolidayName,
         src.AcademicYear, src.Semester, src.SourceSystem
      );
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Fact_RoomUsage
AS
BEGIN
   INSERT INTO dw.Fact_RoomUsage (
      DateKey, RoomKey, UnitKey, DurationMinutes, SessionType, SourceSystem
   )
   SELECT DateKey, RoomKey, UnitKey, DurationMinutes, SessionType, SourceSystem
   FROM stg.Fact_RoomUsage;
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Fact_Repair
AS
BEGIN
   INSERT INTO dw.Fact_Repair (
      ItemKey, Status, DaysToComplete, DateKey, SourceSystem
   )
```

```
    SELECT ItemKey, Status, DaysToComplete, DateKey, SourceSystem
    FROM stg.Fact_Repair;
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Fact_FacilityRequest
AS
BEGIN
    INSERT INTO dw.Fact_FacilityRequest (
        DateKey, UnitKey, JenisPermintaan, Prioritas, Status, SourceSystem
    )
    SELECT DateKey, UnitKey, JenisPermintaan, Prioritas, Status, SourceSystem
    FROM stg.Fact_FacilityRequest;
END;
GO

CREATE OR ALTER PROCEDURE etl.Load_Fact_ItemMovement
AS
BEGIN
    INSERT INTO dw.Fact_ItemMovement (
        ItemKey, DariRuangan, KeRuangan, TglMutasi, SourceSystem
    )
    SELECT ItemKey, DariRuangan, KeRuangan, TglMutasi, SourceSystem
    FROM stg.Fact_ItemMovement;
END;
GO

CREATE OR ALTER PROCEDURE etl.Master_ETL_Sarpras
AS
BEGIN
    PRINT 'START ETL...';

    EXEC etl.Load_Dim_Unit;
    EXEC etl.Load_Dim_Gedung;
    EXEC etl.Load_Dim_RoomType;
    EXEC etl.Load_Dim_ItemType;
    EXEC etl.Load_Dim_KondisiItem;
    EXEC etl.Load_Dim_Room;
    EXEC etl.Load_Dim_Item;
    EXEC etl.Load_Dim_Date;
    EXEC etl.Load_Fact_RoomUsage;
    EXEC etl.Load_Fact_Repair;
    EXEC etl.Load_Fact_FacilityRequest;
    EXEC etl.Load_Fact_ItemMovement;

    PRINT 'ETL COMPLETED.';
END;
GO
```



	LogID	ProcedureName	StartTime	EndTime	Status	Remarks
1	15	Load_Fact_ItemMovement	2025-11-24 15:09:03.883	2025-11-24 15:09:03.907	SUCCESS	NULL
2	14	Load_Fact_FacilityRequest	2025-11-24 15:09:03.867	2025-11-24 15:09:03.870	SUCCESS	NULL
3	13	Load_Fact_Repair	2025-11-24 15:09:03.853	2025-11-24 15:09:03.853	SUCCESS	NULL
4	12	Load_Fact_RoomUsage	2025-11-24 15:09:03.840	2025-11-24 15:09:03.840	SUCCESS	NULL
5	11	Load_Dim_Date	2025-11-24 15:09:03.787	2025-11-24 15:09:03.793	SUCCESS	NULL
6	10	Load_Dim_Item	2025-11-24 15:09:03.777	2025-11-24 15:09:03.780	SUCCESS	NULL
7	9	Load_Dim_Room	2025-11-24 15:09:03.763	2025-11-24 15:09:03.763	SUCCESS	NULL
8	8	Load_Dim_KondisiItem	2025-11-24 15:09:03.743	2025-11-24 15:09:03.743	SUCCESS	NULL
9	7	Load_Dim_ItemType	2025-11-24 15:09:03.733	2025-11-24 15:09:03.737	SUCCESS	NULL
10	6	Load_Dim_RoomType	2025-11-24 15:09:03.723	2025-11-24 15:09:03.723	SUCCESS	NULL
11	5	Load_Dim_Gedung	2025-11-24 15:09:03.697	2025-11-24 15:09:03.697	SUCCESS	NULL
12	4	Load_Dim_Unit	2025-11-24 15:09:03.677	2025-11-24 15:09:03.687	SUCCESS	NULL
13	3	Load_Dim_Unit	2025-11-24 15:07:15.267	2025-11-24 15:07:15.300	SUCCESS	NULL
14	2	Load_Dim_Unit	2025-11-24 15:06:35.823	2025-11-24 15:06:35.827	SUCCESS	NULL
15	1	Load_Dim_Unit	2025-11-24 14:54:43.783	NULL	STARTED	NULL

Step 6: Data Quality Assurance

Tujuan: Memastikan kualitas data yang dimuat ke data warehouse

Aktivitas:

```
-- Dim_Unit
SELECT 'Dim_Unit' AS TableName,
      COUNT(*) AS TotalRows,
      SUM(CASE WHEN NamaUnit IS NULL THEN 1 ELSE 0 END) AS Null_NamaUnit
FROM dw.Dim_Unit;

-- Dim_Gedung
SELECT 'Dim_Gedung' AS TableName,
      COUNT(*) AS TotalRows,
      SUM(CASE WHEN GedungID IS NULL THEN 1 ELSE 0 END) AS Null_GedungID,
            SUM(CASE WHEN NamaGedung IS NULL THEN 1 ELSE 0 END) AS
Null_NamaGedung
FROM dw.Dim_Gedung;

-- Dim_RoomType
SELECT 'Dim_RoomType' AS TableName,
      COUNT(*) AS TotalRows,
        SUM(CASE WHEN RoomTypeID IS NULL THEN 1 ELSE 0 END) AS
Null_RoomTypeID,
        SUM(CASE WHEN NamaTipeRuang IS NULL THEN 1 ELSE 0 END) AS
Null_NamaTipeRuang
FROM dw.Dim_RoomType;

-- Dim_Room
SELECT 'Dim_Room' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN RoomKey IS NULL THEN 1 ELSE 0 END) AS Null_RoomKey,
    SUM(CASE WHEN RoomCode IS NULL THEN 1 ELSE 0 END) AS Null_RoomCode,
    SUM(CASE WHEN NamaRuang IS NULL THEN 1 ELSE 0 END) AS Null_NamaRuang,
    SUM(CASE WHEN GedungID IS NULL THEN 1 ELSE 0 END) AS Null_GedungID
FROM dw.Dim_Room;

-- Dim_ItemType
SELECT 'Dim_ItemType' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN ItemTypeID IS NULL THEN 1 ELSE 0 END) AS Null_ItemTypeID,
        SUM(CASE WHEN NamaJenisItem IS NULL THEN 1 ELSE 0 END) AS
Null_NamaJenisItem
FROM dw.Dim_ItemType;

-- Dim_KondisiItem
SELECT 'Dim_KondisiItem' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN KondisiID IS NULL THEN 1 ELSE 0 END) AS Null_KondisiID,
    SUM(CASE WHEN Kondisi IS NULL THEN 1 ELSE 0 END) AS Null_Kondisi
FROM dw.Dim_KondisiItem;

-- Dim_Item
SELECT 'Dim_Item' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN ItemKey IS NULL THEN 1 ELSE 0 END) AS Null_ItemKey,
    SUM(CASE WHEN ItemTypeID IS NULL THEN 1 ELSE 0 END) AS Null_ItemTypeID,
    SUM(CASE WHEN KondisiID IS NULL THEN 1 ELSE 0 END) AS Null_KondisiID,
        SUM(CASE WHEN CurrentRoomKey IS NULL THEN 1 ELSE 0 END) AS
Null_CurrentRoomKey
FROM dw.Dim_Item;
```

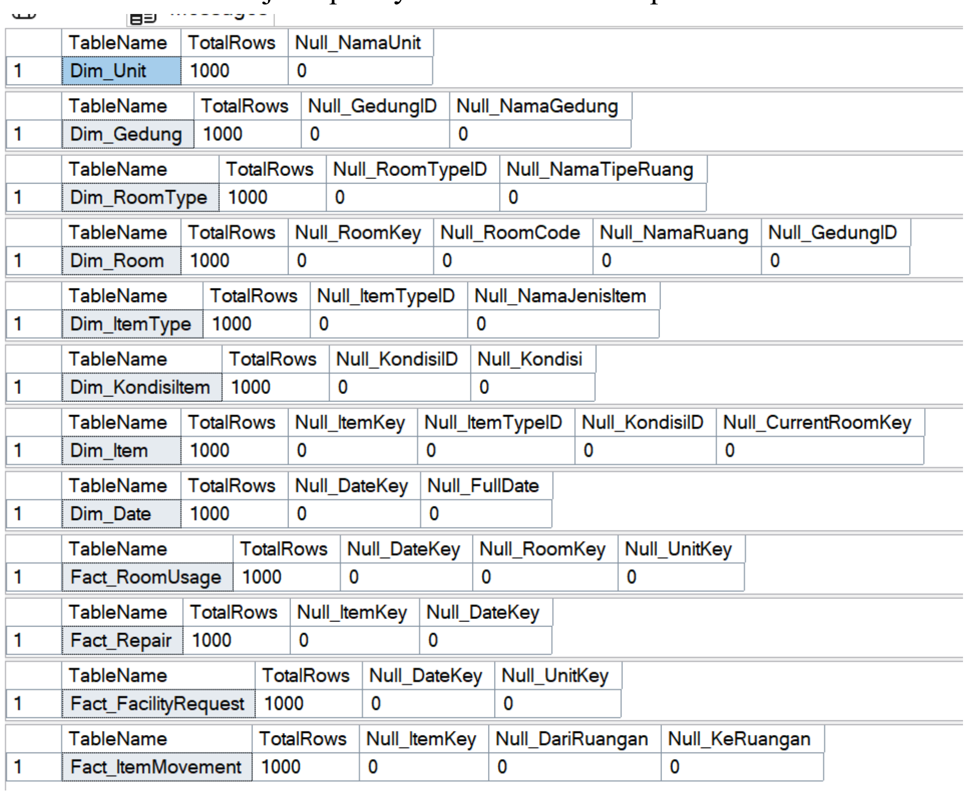
```
-- Dim_Date
SELECT 'Dim_Date' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN DateKey IS NULL THEN 1 ELSE 0 END) AS Null_DateKey,
    SUM(CASE WHEN FullDate IS NULL THEN 1 ELSE 0 END) AS Null_FullDate
FROM dw.Dim_Date;

-- Fact_RoomUsage
SELECT 'Fact_RoomUsage' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN DateKey IS NULL THEN 1 ELSE 0 END) AS Null_DateKey,
    SUM(CASE WHEN RoomKey IS NULL THEN 1 ELSE 0 END) AS Null_RoomKey,
    SUM(CASE WHEN UnitKey IS NULL THEN 1 ELSE 0 END) AS Null_UnitKey
FROM dw.Fact_RoomUsage;

-- Fact_Repair
SELECT 'Fact_Repair' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN ItemKey IS NULL THEN 1 ELSE 0 END) AS Null_ItemKey,
    SUM(CASE WHEN DateKey IS NULL THEN 1 ELSE 0 END) AS Null_DateKey
FROM dw.Fact_Repair;

-- Fact_FacilityRequest
SELECT 'Fact_FacilityRequest' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN DateKey IS NULL THEN 1 ELSE 0 END) AS Null_DateKey,
    SUM(CASE WHEN UnitKey IS NULL THEN 1 ELSE 0 END) AS Null_UnitKey
FROM dw.Fact_FacilityRequest;

-- Fact_ItemMovement
SELECT 'Fact_ItemMovement' AS TableName,
    COUNT(*) AS TotalRows,
    SUM(CASE WHEN ItemKey IS NULL THEN 1 ELSE 0 END) AS Null_ItemKey,
        SUM(CASE WHEN DariRuangan IS NULL THEN 1 ELSE 0 END) AS
Null_DariRuangan,
    SUM(CASE WHEN KeRuangan IS NULL THEN 1 ELSE 0 END) AS Null_KeRuangan
FROM dw.Fact_ItemMovement;

-- Orphan RoomKey
SELECT COUNT(*) AS Orphan_RoomKey
```

```
FROM dw.Fact_RoomUsage f
LEFT JOIN dw.Dim_Room d ON f.RoomKey = d.RoomKey AND d.IsCurrent = 1
WHERE d.RoomKey IS NULL;

-- Orphan UnitKey
SELECT COUNT(*) AS Orphan_UnitKey
FROM dw.Fact_RoomUsage f
LEFT JOIN dw.Dim_Unit u ON f.UnitKey = u.UnitKey
WHERE u.UnitKey IS NULL;

-- Orphan DateKey
SELECT COUNT(*) AS Orphan_DateKey
FROM dw.Fact_RoomUsage f
LEFT JOIN dw.Dim_Date dt ON f.DateKey = dt.DateKey
WHERE dt.DateKey IS NULL;

SELECT COUNT(*) AS Orphan_ItemKey
FROM dw.Fact_Repair f
LEFT JOIN dw.Dim_Item i ON f.ItemKey = i.ItemKey AND i.IsCurrent = 1
WHERE i.ItemKey IS NULL;

SELECT COUNT(*) AS Orphan_DateKey
FROM dw.Fact_Repair f
LEFT JOIN dw.Dim_Date d ON f.DateKey = d.DateKey
WHERE d.DateKey IS NULL;

SELECT COUNT(*) AS Orphan_UnitKey
FROM dw.Fact_FacilityRequest f
LEFT JOIN dw.Dim_Unit u ON f.UnitKey = u.UnitKey
WHERE u.UnitKey IS NULL;

SELECT COUNT(*) AS Orphan_DateKey
FROM dw.Fact_FacilityRequest f
LEFT JOIN dw.Dim_Date d ON f.DateKey = d.DateKey
WHERE d.DateKey IS NULL;

SELECT COUNT(*) AS Orphan_ItemKey
FROM dw.Fact_ItemMovement f
LEFT JOIN dw.Dim_Item i ON f.ItemKey = i.ItemKey AND i.IsCurrent = 1
WHERE i.ItemKey IS NULL;
```

```
-- RoomUsage duration must be > 0
SELECT COUNT(*) AS Invalid_Duration
FROM dw.Fact_RoomUsage
WHERE DurationMinutes <= 0;

-- FacilityRequest Priority must be valid
SELECT COUNT(*) AS Invalid_Prioritas
FROM dw.Fact_FacilityRequest
WHERE Prioritas NOT IN ('Low','Medium','High');

-- Repair DaysToComplete must be >= 0
SELECT COUNT(*) AS Invalid_RepairDays
FROM dw.Fact_Repair
WHERE DaysToComplete < 0;

-- Duplicate RoomUsage (DateKey, RoomKey, UnitKey)
SELECT DateKey, RoomKey, UnitKey,
    COUNT(*) AS DuplicateCount
FROM dw.Fact_RoomUsage
GROUP BY DateKey, RoomKey, UnitKey
HAVING COUNT(*) > 1;

-- Duplicate ItemMovement for same item-date
SELECT ItemKey, TglMutasi,
    COUNT(*) AS DuplicateCount
FROM dw.Fact_ItemMovement
GROUP BY ItemKey, TglMutasi
HAVING COUNT(*) > 1;

-- RoomUsage
SELECT 'stg.Fact_RoomUsage' AS SourceTable, COUNT(*) AS RecordCount
FROM stg.Fact_RoomUsage
UNION ALL
SELECT 'dw.Fact_RoomUsage', COUNT(*)
FROM dw.Fact_RoomUsage;

-- Repair
SELECT 'stg.Fact_Repair', COUNT(*) FROM stg.Fact_Repair
UNION ALL
```

```
SELECT 'dw.Fact_Repair', COUNT(*) FROM dw.Fact_Repair;

-- FacilityRequest
SELECT 'stg.Fact_FacilityRequest', COUNT(*) FROM stg.Fact_FacilityRequest
UNION ALL
SELECT 'dw.Fact_FacilityRequest', COUNT(*) FROM dw.Fact_FacilityRequest;

-- ItemMovement
SELECT 'stg.Fact_ItemMovement', COUNT(*) FROM stg.Fact_ItemMovement
UNION ALL
SELECT 'dw.Fact_ItemMovement', COUNT(*) FROM dw.Fact_ItemMovement;
```

1. Create Data Quality Dashboard
    - Tabel audit untuk tracking data quality metrics
    - Stored procedure untuk generate quality report
    - Alert jika quality threshold tidak terpenuhi



	TableName	TotalRows	Null_NamaUnit
1	Dim_Unit	1000	0

	TableName	TotalRows	Null_GedungID	Null_NamaGedung
1	Dim_Gedung	1000	0	0

	TableName	TotalRows	Null_RoomTypeID	Null_NamaTipeRuang
1	Dim_RoomType	1000	0	0

	TableName	TotalRows	Null_RoomKey	Null_RoomCode	Null_NamaRuang	Null_GedungID
1	Dim_Room	1000	0	0	0	0

	TableName	TotalRows	Null_ItemTypeID	Null_NamaJenisItem
1	Dim_ItemType	1000	0	0

	TableName	TotalRows	Null_KondisiID	Null_Kondisi
1	Dim_KondisiItem	1000	0	0

	TableName	TotalRows	Null_ItemKey	Null_ItemTypeID	Null_KondisiID	Null_CurrentRoomKey
1	Dim_Item	1000	0	0	0	0

	TableName	TotalRows	Null_DateKey	Null_FullDate
1	Dim_Date	1000	0	0

	TableName	TotalRows	Null_DateKey	Null_RoomKey	Null_UnitKey
1	Fact_RoomUsage	1000	0	0	0

	TableName	TotalRows	Null_ItemKey	Null_DateKey
1	Fact_Repair	1000	0	0

	TableName	TotalRows	Null_DateKey	Null_UnitKey
1	Fact_FacilityRequest	1000	0	0

	TableName	TotalRows	Null_ItemKey	Null_DariRuangan	Null_KeRuangan
1	Fact_ItemMovement	1000	0	0	0

Step 7: Performance Testing

Tujuan: Menguji dan mengoptimalkan performa query

Aktivitas:

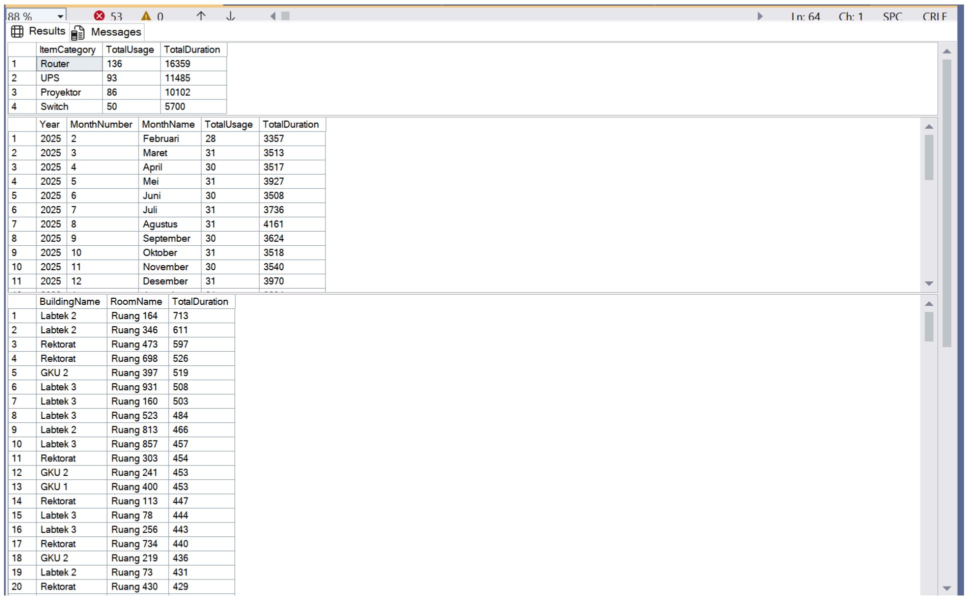
```
SET STATISTICS TIME ON;
SET STATISTICS IO ON;
```

```
SELECT
   it.NamaJenisItem AS ItemCategory,
   COUNT(fr.RoomKey) AS TotalUsage,
   SUM(fr.DurationMinutes) AS TotalDuration
FROM dw.Fact_RoomUsage fr
INNER JOIN dw.Dim_Item i
   ON fr.RoomKey = i.CurrentRoomKey AND i.IsCurrent = 1
INNER JOIN dw.Dim_ItemType it
   ON i.ItemTypeID = it.ItemTypeID
INNER JOIN dw.Dim_Date d
   ON fr.DateKey = d.DateKey
WHERE d.Year = 2024
GROUP BY it.NamaJenisItem
ORDER BY TotalUsage DESC;

SELECT
   d.Year,
   d.MonthNumber,
   d.MonthName,
   COUNT(fr.RoomKey) AS TotalUsage,
   SUM(fr.DurationMinutes) AS TotalDuration
FROM dw.Fact_RoomUsage fr
INNER JOIN dw.Dim_Date d
   ON fr.DateKey = d.DateKey
GROUP BY
   d.Year, d.MonthNumber, d.MonthName
ORDER BY
   d.Year, d.MonthNumber;

SELECT
   g.NamaGedung AS BuildingName,
   r.NamaRuang AS RoomName,
   SUM(fr.DurationMinutes) AS TotalDuration
FROM dw.Fact_RoomUsage fr
INNER JOIN dw.Dim_Room r
   ON fr.RoomKey = r.RoomKey AND r.IsCurrent = 1
INNER JOIN dw.Dim_Gedung g
   ON r.GedungID = g.GedungID
GROUP BY
   g.NamaGedung, r.NamaRuang
```

```
ORDER BY
    TotalDuration DESC;

SELECT TOP 10
    i.NamaItem,
    it.NamaJenisItem AS ItemCategory,
    SUM(fr.DurationMinutes) AS TotalDuration
FROM dw.Fact_RoomUsage fr
INNER JOIN dw.Dim_Item i
    ON fr.RoomKey = i.CurrentRoomKey AND i.IsCurrent = 1
INNER JOIN dw.Dim_ItemType it
    ON i.ItemTypeID = it.ItemTypeID
GROUP BY
    i.NamaItem, it.NamaJenisItem
ORDER BY
    TotalDuration DESC;
```

1. Analyze Query Plans
    - Review execution plans
    - Identify missing indexes
    - Check for table scans
    - Optimize join orders

2. Performance Benchmarks



	ItemCategory	TotalUsage	TotalDuration
1	Router	136	16359
2	UPS	93	11485
3	Proyektor	86	10102
4	Switch	50	5700

	Year	MonthNumber	MonthName	TotalUsage	TotalDuration
1	2025	2	Februari	28	3357
2	2025	3	Maret	31	3513
3	2025	4	April	30	3517
4	2025	5	Mei	31	3927
5	2025	6	Juni	30	3508
6	2025	7	Juli	31	3736
7	2025	8	Agustus	31	4161
8	2025	9	September	30	3624
9	2025	10	Oktober	31	3518
10	2025	11	November	30	3540
11	2025	12	Desember	31	3970

	BuildingName	RoomName	TotalDuration
1	Labtek 2	Ruang 164	713
2	Labtek 2	Ruang 346	611
3	Rektorat	Ruang 473	597
4	Rektorat	Ruang 698	526
5	GKU 2	Ruang 397	519
6	Labtek 3	Ruang 931	508
7	Labtek 3	Ruang 160	503
8	Labtek 3	Ruang 523	484
9	Labtek 2	Ruang 813	466
10	Labtek 3	Ruang 857	457
11	Rektorat	Ruang 303	454
12	GKU 2	Ruang 241	453
13	GKU 1	Ruang 400	453
14	Rektorat	Ruang 113	447
15	Labtek 3	Ruang 78	444
16	Labtek 3	Ruang 256	443
17	Rektorat	Ruang 734	440
18	GKU 2	Ruang 219	436
19	Labtek 2	Ruang 73	431
20	Rektorat	Ruang 430	429