Tugas Besar Pergudangan Data
Misi 3

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 3: IMPLEMENTASI PRODUKSI

Step 1: Production Deployment
Tujuan: Men-deploy database dan ETL ke Azure VM production
Aktivitas:
1. Environment Setup
2. Database Deployment
3. Initial Data Load
4. Schedule ETL Jobs

```
USE msdb;
GO

-- 1. Create Job
EXEC sp_add_job
   @job_name     = N'ETL_Daily_Load_SARPRAS',
   @enabled      = 1,
   @description  = N'Daily ETL Load untuk Data Mart Sarpras';
GO

-- 2. Create Job Step (jalankan Master ETL)
EXEC sp_add_jobstep
   @job_name        = N'ETL_Daily_Load_SARPRAS',
   @step_name       = N'Execute Master ETL Sarpras',
   @subsystem       = N'TSQL',
   @command         = N'EXEC dbo.usp_Master_ETL_Sarpras;',
   -- Ganti jika nama SP berbeda
   @database_name   = N'DM_Sarpras_DW',
   @retry_attempts  = 3,
    @retry_interval  = 5;  -- menit
GO

-- 3. Create Daily Schedule (02:00 AM)
EXEC sp_add_schedule
    @schedule_name     = N'Sarpras Daily at 2 AM',
    @freq_type         = 4,        -- Daily
    @freq_interval     = 1,        -- Setiap 1 hari
    @active_start_time = 020000;   -- 02:00 AM (HHMMSS)
GO

-- 4. Attach Schedule to the Job
EXEC sp_attach_schedule
    @job_name      = N'ETL_Daily_Load_SARPRAS',
    @schedule_name = N'Sarpras Daily at 2 AM';
GO

-- 5. Register Job on Server
EXEC sp_add_jobserver
    @job_name = N'ETL_Daily_Load_SARPRAS';
GO
```

Step 2: Dashboard Development
Tujuan: Membangun interactive dashboard untuk end-users
Aktivitas:

1. Create Analytical Views

```
-- View: Room Utilization Summary
CREATE VIEW dbo.vw_Room_Utilization AS
SELECT
    dr.RoomKey,
    dr.NamaRuang,
    dr.RoomCode,
    dr.Kapasitas,
    dg.NamaGedung,
    drt.NamaTipeRuang,

    COUNT(fu.RoomUsageKey) AS TotalUsage,
    SUM(fu.DurationMinutes) AS TotalMinutesUsed,
    CAST(
        AVG(CAST(fu.DurationMinutes AS FLOAT))
        AS DECIMAL(10,2)
    ) AS AvgDuration,

    COUNT(DISTINCT fu.UnitKey) AS TotalUnitsUsing,

    -- Presentase penggunaan terhadap waktu 1 bulan (43.200 menit)
    CAST(
        SUM(fu.DurationMinutes) * 100.0 / NULLIF(43200, 0)
        AS DECIMAL(5,2)
    ) AS UtilizationRate

FROM dbo.Fact_RoomUsage fu
INNER JOIN dbo.Dim_Room dr ON fu.RoomKey = dr.RoomKey
INNER JOIN dbo.Dim_Gedung dg ON dr.GedungID = dg.GedungID
INNER JOIN dbo.Dim_RoomType drt ON dr.RoomTypeID = drt.RoomTypeID
GROUP BY
    dr.RoomKey,
    dr.NamaRuang,
    dr.RoomCode,
    dr.Kapasitas,
    dg.NamaGedung,
    drt.NamaTipeRuang;
GO

-- View: Facility Request Analytics
CREATE VIEW dbo.vw_Facility_Request_Analytics AS
SELECT
    du.NamaUnit,
    d.Year AS Tahun,
    d.MonthNumber AS Bulan,

    COUNT(fr.FacilityReqKey) AS TotalRequests,

    SUM(CASE WHEN fr.Prioritas = 'High' THEN 1 ELSE 0 END) AS HighPriority,
      SUM(CASE WHEN fr.Prioritas = 'Medium' THEN 1 ELSE 0 END) AS
MediumPriority,
    SUM(CASE WHEN fr.Prioritas = 'Low' THEN 1 ELSE 0 END) AS LowPriority,

    SUM(CASE WHEN fr.Status = 'Completed' THEN 1 ELSE 0 END) AS Completed,
    SUM(CASE WHEN fr.Status = 'Pending' THEN 1 ELSE 0 END) AS Pending,
    SUM(CASE WHEN fr.Status = 'In Progress' THEN 1 ELSE 0 END) AS InProgress,

    CAST(
        SUM(CASE WHEN fr.Status = 'Completed' THEN 1 ELSE 0 END)
        * 100.0 / NULLIF(COUNT(*), 0)
        AS DECIMAL(5,2)
    ) AS CompletionRate

FROM dbo.Fact_FacilityRequest fr
INNER JOIN dbo.Dim_Unit du ON fr.UnitKey = du.UnitKey
INNER JOIN dbo.Dim_Date d ON fr.DateKey = d.DateKey
GROUP BY
    du.NamaUnit,
    d.Year,
    d.MonthNumber;
GO
```

2. Design Power BI Dashboards
   Format Dashboard → beri warna berbeda untuk kategori prioritas, tipe ruangan, atau status
   a. Dashboard 1: Executive Summary
      - KPI Cards: Total Aset Aktif, Tingkat Kesiapan Sarana (%), Total Peminjaman Bulanan
      - Line Chart: Tren penggunaan sarana dari waktu ke waktu
      - Bar Chart: Distribusi peminjaman aset per divisi
      - Map**:** Lokasi penyimpanan aset & titik persebaran sarana lapangan

   b. Dashboard 2: Operational Performance
      - Stacked Bar: Status kondisi aset (Baik, Perlu Perbaikan, Rusak) per kategori
      - Heat Map: Intensitas penggunaan sarana berdasarkan tanggal & jenis aset
      - Table: Daftar aset dengan frekuensi pemakaian tertinggi
      - Gauge: Persentase aset siap pakai dibanding target kesiapan

   c. Dashboard 3: Maintenance & Logistics Analysis
      - Area Chart: Tren permintaan perbaikan sarana per bulan
      - Pie Chart: Proporsi jenis sarana (meja, kursi, tenda, sound system, dan lainnya)
      - Waterfall: Rincian biaya pemeliharaan dari pengajuan hingga realisasi

        - Forecast: Proyeksi kebutuhan sarana & pemeliharaan untuk periode berikutnya

    d. Connect Power BI to SQL Server
        - Get Data → SQL Server
        - Server: [Azure VM IP/Hostname]
        - Database: DM_Sarpras_DW
        - Pilihan koneksi:
            - Import Mode – cocok untuk laporan statis
            - DirectQuery (disarankan untuk data penggunaan sarana yang real-time)
                - Load tabel seperti inventaris aset, log peminjaman, log perbaikan, serta view monitoring sarana
                - Dapat menggunakan DAX queries untuk agregasi kebutuhan sarana dan analisis performa penggunaan

3. Implement Interactivity
    - Slicers: Tahun Kegiatan, Jenis Sarana, Lokasi Penyimpanan, Status Kondisi
    - Drill-through: Dari ringkasan penggunaan sarana → detail aset per kategori atau per kegiatan
    - Cross-filtering: Antar visual saling memfilter, misalnya memilih kategori sarana akan otomatis memfilter grafik pemakaian, kondisi aset, dan biaya perawatan
    - Bookmarks: Untuk menampilkan berbagai tampilan seperti View Penggunaan, View Pemeliharaan, dan View Inventaris
    - Row-Level Security (opsional): Mengatur akses berdasarkan unit, misalnya staf Sarpras hanya melihat aset di bawah tanggung jawabnya

Step 3: Security Implementation
Tujuan: Mengimplementasikan access control dan audit trail
Aktivitas:

1. Create User Roles

```
-- Create Database Roles
CREATE ROLE db_executive;
CREATE ROLE db_analyst;
CREATE ROLE db_viewer;
CREATE ROLE db_etl_operator;
GO

-- Executive Full Access for SELECT and ETL Procedure
GRANT SELECT ON SCHEMA::dbo TO db_executive;
GRANT EXECUTE ON SCHEMA::dbo TO db_executive;
GO

-- Analyst: Can analyze and edit staging before loading
GRANT SELECT ON SCHEMA::dbo TO db_analyst;
GRANT SELECT, INSERT, UPDATE, DELETE ON SCHEMA::stg TO db_analyst;
GO

-- Viewer: Read-only access to all dimensional & fact tables
GRANT SELECT ON SCHEMA::dbo TO db_viewer;
GO

-- ETL Operator: Full access to staging + loading to DWH
GRANT EXECUTE ON SCHEMA::dbo TO db_etl_operator;
GRANT SELECT, INSERT, UPDATE, DELETE ON SCHEMA::stg TO
db_etl_operator;
GRANT INSERT, UPDATE ON SCHEMA::dbo TO db_etl_operator;
GO
```

2. Create Users and Assign Roles

```
-- Create SQL Logins
CREATE LOGIN executive_user WITH PASSWORD = 'Barcelona123';
CREATE LOGIN analyst_user WITH PASSWORD = 'Barcelona123';
CREATE LOGIN viewer_user WITH PASSWORD = 'Barcelona123';
CREATE LOGIN etl_service WITH PASSWORD = 'Barcelona123';
GO

-- Create Database Users
USE DM_SARPRAS_DW;
GO

CREATE USER executive_user FOR LOGIN executive_user;
CREATE USER analyst_user FOR LOGIN analyst_user;
CREATE USER viewer_user FOR LOGIN viewer_user;
CREATE USER etl_service FOR LOGIN etl_service;
GO

-- Assign Users to Roles
```

```
ALTER ROLE db_executive ADD MEMBER executive_user;
ALTER ROLE db_analyst ADD MEMBER analyst_user;
ALTER ROLE db_viewer ADD MEMBER viewer_user;
ALTER ROLE db_etl_operator ADD MEMBER etl_service;
GO
```

3. Implement Data Masking

```
-- Masking otomatis hanya jika kolom ditemukan

-- Nama Item (contoh jika NamaItem yang dipakai)
IF EXISTS (
   SELECT 1 FROM INFORMATION_SCHEMA.COLUMNS
   WHERE TABLE_NAME = 'Dim_Item' AND COLUMN_NAME = 'NamaItem'
)
BEGIN
   ALTER TABLE dbo.Dim_Item
   ALTER COLUMN NamaItem
   ADD MASKED WITH (FUNCTION = 'partial(1,"xxx",1)');
END

-- Deskripsi Item (alternative)
IF EXISTS (
   SELECT 1 FROM INFORMATION_SCHEMA.COLUMNS
   WHERE TABLE_NAME = 'Dim_Item' AND COLUMN_NAME = 'ItemDescription'
)
BEGIN
   ALTER TABLE dbo.Dim_Item
   ALTER COLUMN ItemDescription
   ADD MASKED WITH (FUNCTION = 'partial(1,"xxxxxxx",1)');
END

-- Serial Number / Code (default masking)
IF EXISTS (
   SELECT 1 FROM INFORMATION_SCHEMA.COLUMNS
   WHERE TABLE_NAME = 'Dim_Item' AND COLUMN_NAME = 'KodeItem'
)
BEGIN
   ALTER TABLE dbo.Dim_Item
   ALTER COLUMN KodeItem
```

```
    ADD MASKED WITH (FUNCTION = 'default()');
END
GO

-- Permission UNMASK
GRANT UNMASK TO db_executive;
GRANT UNMASK TO db_etl_operator;
GO
```

4. Implement Audit Trail
    - Create Audit Table

      ```
      -- Create Audit Table
      CREATE TABLE dbo.AuditLog (
          AuditID BIGINT IDENTITY(1,1) PRIMARY KEY,
          EventTime DATETIME2 DEFAULT SYSDATETIME(),
          UserName NVARCHAR(128) DEFAULT SUSER_SNAME(),
          EventType NVARCHAR(50),
          SchemaName NVARCHAR(128),
          ObjectName NVARCHAR(128),
          RowsAffected INT,
          SQLStatement NVARCHAR(MAX) NULL,
          HostName VARCHAR(128) NULL
      );
      GO
      ```

    - Audit Trigger Example (Fact Repair)

      ```
      CREATE TRIGGER trg_Audit_Fact_Repair
      ON dbo.Fact_Repair
      AFTER INSERT, UPDATE, DELETE
      AS
      BEGIN
          SET NOCOUNT ON;

          DECLARE @EventType NVARCHAR(50);
      ```

```
    IF EXISTS(SELECT * FROM inserted) AND EXISTS(SELECT * FROM deleted)
        SET @EventType = 'UPDATE';
    ELSE IF EXISTS(SELECT * FROM inserted)
        SET @EventType = 'INSERT';
    ELSE
        SET @EventType = 'DELETE';

    INSERT INTO dbo.AuditLog (EventType, SchemaName, ObjectName, RowsAffected)
    VALUES (@EventType, 'dbo', 'Fact_Repair', @@ROWCOUNT);
END;
GO
```

- Enable SQL Server Audit (Server-level)

```
USE master;
GO

IF EXISTS (SELECT * FROM sys.server_audits WHERE name = 'Sarpras_Audit')
    DROP SERVER AUDIT Sarpras_Audit;
GO

-- Buat audit target ke Application Log
CREATE SERVER AUDIT Sarpras_Audit
TO APPLICATION_LOG
WITH (ON_FAILURE = CONTINUE);
GO

ALTER SERVER AUDIT Sarpras_Audit
WITH (STATE = ON);
GO
```

- Create Database Audit Specification

```
CREATE DATABASE AUDIT SPECIFICATION Sarpras_DB_Audit
FOR SERVER AUDIT Sarpras_Audit
```

```
ADD (SELECT, INSERT, UPDATE, DELETE ON SCHEMA::dbo BY public);
GO

ALTER DATABASE AUDIT SPECIFICATION Sarpras_DB_Audit
WITH (STATE = ON);
GO
```

Step 4: Backup and Recovery Strategy
Tujuan: Menyiapkan backup untuk disaster recovery
Aktivitas:

```
-- Full Backup
BACKUP DATABASE DM_Sarpras_DW
TO DISK = N'/var/opt/mssql/backup/DM_Sarpras_DW_Full.bak'
WITH
   COMPRESSION,
   INIT,
   NAME = 'Full Backup',
   STATS = 10;
GO

-- Differential Backup
BACKUP DATABASE DM_Sarpras_DW
TO DISK = N'/var/opt/mssql/backup/DM_Sarpras_DW_Diff.bak'
WITH DIFFERENTIAL, COMPRESSION, INIT, STATS = 10;
GO

-- Transaction Log Backup
SELECT name, recovery_model_desc
FROM sys.databases WHERE name='DM_Sarpras_DW';

ALTER DATABASE DM_Sarpras_DW SET RECOVERY FULL;
GO

BACKUP LOG DM_Sarpras_DW
TO DISK = N'/var/opt/mssql/backup/DM_Sarpras_DW_Log.trn'
WITH COMPRESSION, INIT, STATS = 10;
GO
```

```
-- Schedule Backup Jobs
-- Full Backup: Weekly (Sunday 2 AM)
-- Differential Backup: Daily (2 AM)
-- Transaction Log Backup: Every 6 hours

-- Backup to Azure Blob Storage (Optional)
CREATE CREDENTIAL [AzureStorageCredential]
WITH IDENTITY = 'SHARED ACCESS SIGNATURE',
    SECRET = '<SAS_TOKEN>';
GO

BACKUP DATABASE DM_Sarpras_DW
TO URL = N'https://[storage_account].blob.core.windows.net/backups/DM_Sarpras_DW.bak'
WITH
  CREDENTIAL = 'AzureStorageCredential',
  COMPRESSION;
GO
```

Step 5: User Acceptance Testing
Tujuan: Validasi sistem dengan end-users
Aktivitas:

1. Create Test Cases
2. Conduct UAT Sessions
3. Performance Testing
4. Refinement

Step 6: Documentation
Tujuan: Menyiapkan dokumentasi lengkap untuk operasional
Dokumen yang Diperlukan:

1. System Architecture Document
2. Data Dictionary (Update dari Misi 1)
3. ETL Documentation
4. User Manual
5. Operations Manual
6. Security Documentation

Step 7: Final Presentation
Tujuan: Mempresentasikan hasil proyek kepada dosen dan stakeholders
Struktur Presentasi (20-25 menit):

1. Introduction
2. Requirements & Design
3. Implementation
4. Dashboard Demo
5. Technical Highlights
6. Lessons Learned & Future Work
7. Q&A