Rancangan dan Implementasi Arsitektur Data Warehouse Pada Kebun Raya

Disusun Oleh :

Kartini Lovian Simbolon	**122450003**
Nasywa Nur Afifah	**122450125**
Erma Daniar Safitri	**123450061**
Ginda Fajar Riadi Marpaung	**123450103**

PROGRAM STUDI SAINS DATA
FAKULTAS SAINS
INSTITUT TEKNOLOGI SUMATERA

2025

Fakta & Dimensi

Berdasarkan masalah dan tujuan bisnis, maka dapat diidentifikasi sejumlah kebutuhan bisnis yang dirincikan pada tabel berikut.

Tabel 1. 2 Fakta & Dimensi

Kebutuhan	Fakta	Dimensi
Analisis tanaman dan spesies tanaman	Fact_KoleksiTanaman	Dim_Tanaman,Dim_Spesies
Monitoring fasilitas dan pengunjung yang datang	Fact_Kunjungan	Dim_Fasilitas,Dim_pengunjung

Dari hasil identifikasi fakta dan dimensi, selanjutnya dirancang desain konseptual dalam bentuk diagram ERD untuk memberikan gambaran hubungan antar entitas dalam skema data warehouse yang akan dibangun.

Tabel Fakta

Fakta Pengunjung

kunjungan_key	Primary key
waktu_id	Foreign key
pengunjung_id	Foreign key
fasilitas_id	Foreign key
ulasan_id	Foreign key
jumlah_pengunjung	Banyaknya pengunjung yang datang
tujuan	Tempat yang didatangi

Tabel Dimensi

Dim_Waktu

waktu_id	Primary key
tanggal	tanggal
hari	senin-minggu

bulan	1-12
nama_bulan	januari-desember
tahun	tahun
kuartal	kuartal

Dim_Pengunjung

pengunjung_id	primary key
nama	Nama pengunjung
asal_Pengunjung	Asal pengunjung
kategori_pengunjung	(anak,dewasa,remaja)

Dim_Fasilitas

fasilitas_id	Primary key
nama_fasilitas	Nama fasilitas
kapasitas	Maksimal jumlah orang
lokasi	Lokasi fasilitas

Dim_Ulasan

ulasan_id	Primary key
isi_ulasan	Tanggapan pengunjung
rating	Foreign key
tanggal_ulasan	Lokasi fasilitas

Tabel Fakta

Fakta Koleksi Tanaman

koleksitanaman_key	Primary key
waktu_id	Foreign key
tanaman_id	Foreign key

spesies_id	Foreign key
jumlah_tanaman	Banyaknya tanaman yang ada

Tabel Dimensi

Dim_Tanaman

tanaman_id	Primary key
nama_tanaman	Nama tanaman
status_konservasi	Status tanaman
asal_tanaman	Asal tanaman

Dim_Spesies

spesies_id	Primary key
nama_spesies	Nama spesies
jenis_spesies	Jenis spesies
asal_spesies	Asal spesies

QUERY

Database

```
USE master;
GO

IF NOT EXISTS (SELECT name FROM sys.databases WHERE name = N'KebunRaya_DWH')
BEGIN
   CREATE DATABASE KebunRaya_DWH;
END
GO

USE KebunRaya_DWH;
GO
```

Tabel Dimensi

```
USE KebunRaya_DWH;
GO

-- Dim Waktu
CREATE TABLE Dim_Waktu (
    waktu_id INT IDENTITY(1,1) PRIMARY KEY,
    tanggal DATE UNIQUE,
    hari VARCHAR(20),
    bulan INT,
    nama_bulan VARCHAR(20),
    tahun INT,
    kuartal INT
);

-- Dim Pengunjung
CREATE TABLE Dim_Pengunjung (
    pengunjung_id INT PRIMARY KEY,
    nama VARCHAR(255),
    kategori VARCHAR(100),
    asal VARCHAR(100)
);

-- Dim Pengunjung
CREATE TABLE Dim_Pengunjung (
    pengunjung_id INT PRIMARY KEY,
    nama VARCHAR(255),
    kategori VARCHAR(100),
    asal VARCHAR(100)
);

-- Dim Fasilitas
CREATE TABLE Dim_Fasilitas (
    fasilitas_id INT PRIMARY KEY,
    nama_fasilitas VARCHAR(255),
    jenis VARCHAR(100),
    kapasitas INT,
    lokasi VARCHAR(255)
);
```

```
-- Dim Ulasan
CREATE TABLE Dim_Ulasan (
    ulasan_id INT PRIMARY KEY,
    isi_ulasan VARCHAR(255),
    rating INT CHECK (rating BETWEEN 1 AND 5),
    tanggal_ulasan DATE
);

- Dim Tanaman
CREATE TABLE Dim_Tanaman (
    tanaman_id INT PRIMARY KEY,
    nama_tanaman VARCHAR(255),
    status_konservasi VARCHAR(100),
    asal VARCHAR(100)
);

-- Dim Spesies
CREATE TABLE Dim_Spesies (
    spesies_id INT PRIMARY KEY,
    nama_spesies VARCHAR(255),
    jenis VARCHAR(100),
    asal VARCHAR(100)
);
GO
```

Tabel Fakta

```
USE KebunRaya_DWH;
GO

-- Fact Kunjungan
CREATE TABLE Fact_Kunjungan (
    kunjungan_key BIGINT IDENTITY(1,1) PRIMARY KEY,
    waktu_id INT,
    pengunjung_id INT,
    fasilitas_id INT,
    ulasan_id INT,
    jumlah_pengunjung INT,
    tujuan VARCHAR(100),
```

```
        CONSTRAINT FK_FKunj_Waktu FOREIGN KEY (waktu_id) REFERENCES Dim_Waktu(waktu_id),
      CONSTRAINT FK_FKunj_Peng FOREIGN KEY (pengunjung_id) REFERENCES Dim_Pengunjung(pengunjung_id),
        CONSTRAINT FK_FKunj_Fasil FOREIGN KEY (fasilitas_id) REFERENCES Dim_Fasilitas(fasilitas_id),
         CONSTRAINT FK_FKunj_Ulas FOREIGN KEY (ulasan_id) REFERENCES Dim_Ulasan(ulasan_id)
);

-- Fact Tanaman
CREATE TABLE Fact_Tanaman (
   tanaman_key BIGINT IDENTITY(1,1) PRIMARY KEY,
   waktu_id INT,
   tanaman_id INT,
   spesies_id INT,
   kuantitas INT DEFAULT 1,

         CONSTRAINT FK_FTan_Waktu FOREIGN KEY (waktu_id) REFERENCES Dim_Waktu(waktu_id),
      CONSTRAINT FK_FTan_Tanaman FOREIGN KEY (tanaman_id) REFERENCES Dim_Tanaman(tanaman_id),
        CONSTRAINT FK_FTan_Spesies FOREIGN KEY (spesies_id) REFERENCES Dim_Spesies(spesies_id)
);
GO
```

Indexing

```
USE KebunRaya_DWH;
GO

-------------------------------------------------
-- NONCLUSTERED INDEX (Optimasi Join)
-------------------------------------------------
CREATE NONCLUSTERED INDEX IX_FKunj_Waktu
ON Fact_Kunjungan(waktu_id);

CREATE NONCLUSTERED INDEX IX_FKunj_Pengunjung
ON Fact_Kunjungan(pengunjung_id);
```

```
CREATE NONCLUSTERED INDEX IX_FKunj_Fasilitas
ON Fact_Kunjungan(fasilitas_id);

CREATE NONCLUSTERED INDEX IX_FTanaman_Waktu
ON Fact_Tanaman(waktu_id);

CREATE NONCLUSTERED INDEX IX_FTanaman_Tanaman
ON Fact_Tanaman(tanaman_id);

CREATE NONCLUSTERED INDEX IX_FTanaman_Spesies
ON Fact_Tanaman(spesies_id);

-------------------------------------------------
-- COLUMNSTORE INDEX (Analitik)
-------------------------------------------------
CREATE NONCLUSTERED COLUMNSTORE INDEX CSX_Fact_Kunjungan
ON Fact_Kunjungan
(
   waktu_id, pengunjung_id, fasilitas_id, jumlah_pengunjung
);

CREATE NONCLUSTERED COLUMNSTORE INDEX CSX_Fact_Tanaman
ON Fact_Tanaman
(
   waktu_id, tanaman_id, spesies_id, kuantitas
);
GO
```

Indexing dilakukan dengan menambah nonclustered index pada setiap foreign key di fact tables agar proses join ke dimension tables lebih efisien. Clustered index otomatis diterapkan pada primary key fact table untuk mengoptimalkan proses loading dan menjaga integritas data. Columnstore index juga diterapkan pada fact tables guna meningkatkan kecepatan kueri analitik dan agregasi skala besar, sehingga performa akses data warehouse tetap optimal dan responsif.

Staging

```
USE KebunRaya_DWH;
GO

CREATE SCHEMA stg;
GO

CREATE TABLE stg.Kunjungan (
```

```
    waktu DATE,
    pengunjung_id INT,
    fasilitas_id INT,
    tujuan VARCHAR(100),
    jumlah INT,
    LoadDate DATETIME DEFAULT GETDATE()
);

CREATE TABLE stg.Tanaman (
    waktu DATE,
    tanaman_id INT,
    spesies_id INT,
    kuantitas INT,
    LoadDate DATETIME DEFAULT GETDATE()
);
GO
```

Staging area merupakan area penyimpanan sementara yang digunakan untuk menampung data mentah hasil ekstraksi dari berbagai sumber, seperti file Excel, CSV, atau sistem operasional lainnya, sebelum data tersebut diproses lebih lanjut. Pada tahap ini, data ditempatkan dalam tabel staging tanpa pembatasan atau constraint yang ketat, sehingga memudahkan proses pembersihan, validasi, dan transformasi sesuai kebutuhan data warehouse. Misalnya, setelah dijalankan query pembuatan tabel stg_tanaman, data koleksi tanaman yang diimpor dari sumber eksternal akan masuk lebih dulu ke staging area ini. Di sini, data dapat diperiksa untuk menghilangkan duplikasi, memperbaiki format yang tidak standar, dan menyempurnakan nilai yang kurang valid. Setelah proses cleansing dan transformasi selesai, barulah data tersebut di-load ke tabel utama dimension atau fact pada data warehouse. Dengan mengimplementasikan staging area, proses ETL menjadi lebih terstruktur, data error dapat ditangani lebih awal, serta memastikan kualitas dan integritas data di lingkungan data warehouse tetap terjaga.

ETL

```
USE KebunRaya_DWH;
GO
-- Load Fact_Kunjungan
CREATE OR ALTER PROCEDURE usp_Load_FactKunjungan
AS
BEGIN
                                                    INSERT                INTO
Fact_Kunjungan(waktu_id,pengunjung_id,fasilitas_id,ulasan_id,jumlah_pengunjung,tujua
n)
```

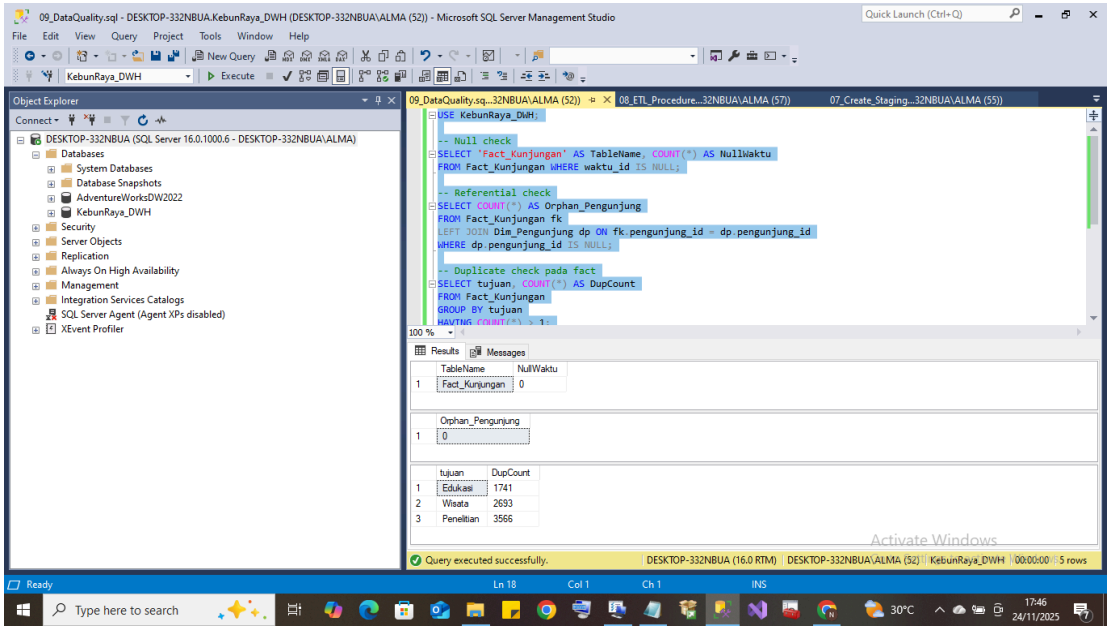
```
    SELECT
        w.waktu_id,
        s.pengunjung_id,
        s.fasilitas_id,
        NULL,
        s.jumlah,
        s.tujuan
    FROM stg.Kunjungan s
    JOIN Dim_Waktu w ON w.tanggal = s.waktu;
END
GO


-- Load Fact_Tanaman
CREATE OR ALTER PROCEDURE usp_Load_FactTanaman
AS
BEGIN
    INSERT INTO Fact_Tanaman(waktu_id,tanaman_id,spesies_id,kuantitas)
    SELECT
        w.waktu_id,
        t.tanaman_id,
        t.spesies_id,
        t.kuantitas
    FROM stg.Tanaman t
    JOIN Dim_Waktu w ON w.tanggal = t.waktu;
END
GO


-- Master ETL
CREATE OR ALTER PROCEDURE usp_Master_ETL
AS
BEGIN
    EXEC usp_Load_FactKunjungan;
    EXEC usp_Load_FactTanaman;
END
GO
```

Data Quality Check

```
USE KebunRaya_DWH;

-- Null check
```

```
SELECT 'Fact_Kunjungan' AS TableName, COUNT(*) AS NullWaktu
FROM Fact_Kunjungan WHERE waktu_id IS NULL;

-- Referential check
SELECT COUNT(*) AS Orphan_Pengunjung
FROM Fact_Kunjungan fk
LEFT JOIN Dim_Pengunjung dp ON fk.pengunjung_id = dp.pengunjung_id
WHERE dp.pengunjung_id IS NULL;

-- Duplicate check pada fact
SELECT tujuan, COUNT(*) AS DupCount
FROM Fact_Kunjungan
GROUP BY tujuan
HAVING COUNT(*) > 1;
```

Output



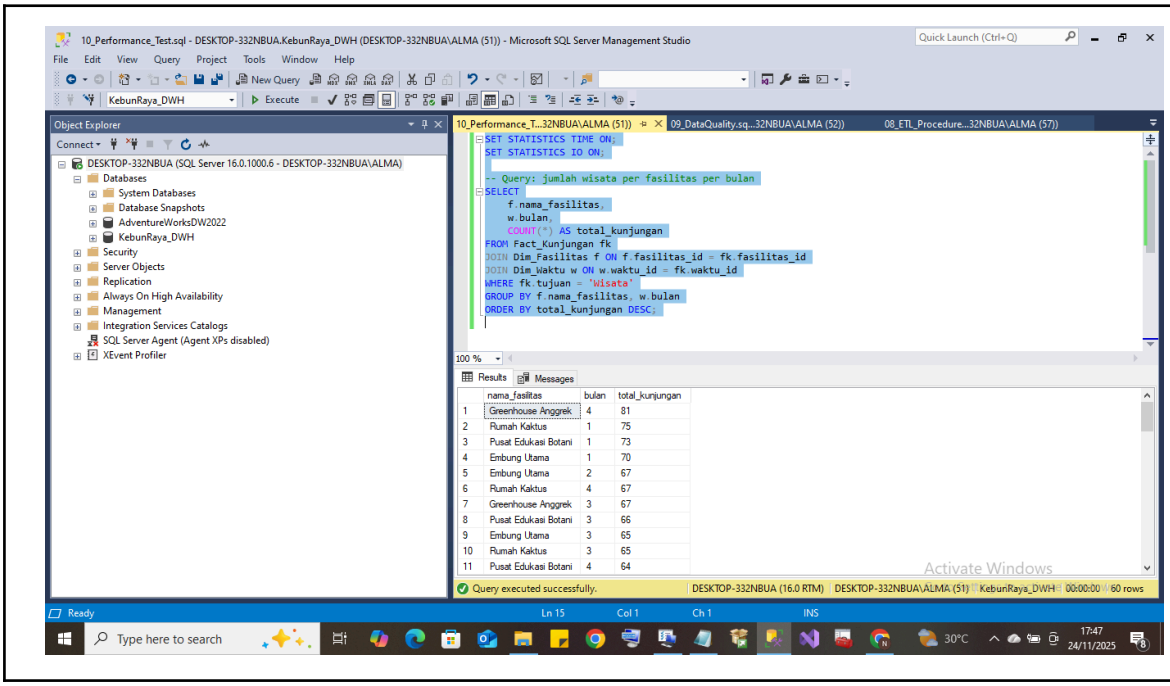
Pengecekan kualitas data pada tabel fact dilakukan untuk memastikan kelengkapan, konsistensi, dan integritas data hasil pemrosesan ETL. Berdasarkan hasil query, tidak ditemukan nilai NULL pada kolom kunci waktu, menandakan setiap data kunjungan memiliki informasi tanggal yang valid dan tidak ada catatan yang hilang. Selain itu, pemeriksaan referensi pengunjung menunjukkan seluruh foreign key telah terhubung dengan benar ke tabel dimension, sehingga tidak ditemukan data orphan yang berpotensi menyebabkan error dalam analisis selanjutnya.

Distribusi tujuan kunjungan yang tercatat dalam tabel fact juga telah sesuai dengan kategori aslinya, dengan jumlah entri yang konsisten menurut masing-masing jenis aktivitas. Hasil pengecekan ini menunjukkan proses pembersihan dan validasi data telah berjalan optimal, dan seluruh data warehouse kini siap digunakan untuk analisis dan pelaporan lebih lanjut. Upaya menjaga kualitas data ini penting agar informasi yang dihasilkan oleh sistem benar-benar akurat serta bermanfaat sebagai dasar pengambilan keputusan manajerial.

Performance Testing

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

-- Query: jumlah wisata per fasilitas per bulan
SELECT
    f.nama_fasilitas,
    w.bulan,
    COUNT(*) AS total_kunjungan
FROM Fact_Kunjungan fk
JOIN Dim_Fasilitas f ON f.fasilitas_id = fk.fasilitas_id
JOIN Dim_Waktu w ON w.waktu_id = fk.waktu_id
WHERE fk.tujuan = 'Wisata'
GROUP BY f.nama_fasilitas, w.bulan
ORDER BY total_kunjungan DESC;
```

Output :



Performance testing dilakukan untuk mengukur kecepatan dan efisiensi query pada data warehouse Kebun Raya ITERA. Salah satu uji yang dilakukan adalah query untuk menghitung jumlah kunjungan wisata per fasilitas pada setiap bulan. Query ini menggabungkan data dari fact table kunjungan dengan dimension fasilitas dan waktu, menggunakan operasi group by serta filter kategori kunjungan wisata. Hasil pengujian memperlihatkan bahwa proses eksekusi berjalan sukses tanpa error, dan tabel hasil dapat menampilkan jumlah kunjungan yang teragregasi sesuai masing-masing fasilitas dan bulan.

Dalam pengujian ini, eksekusi query berlangsung cepat dan sistem mampu menghasilkan output rekap kunjungan untuk seluruh fasilitas utama dalam waktu sangat efisien, seperti Greenhouse Anggrek, Rumah Kaktus, dan Embung Utama. Hal ini menunjukkan optimasi index dan struktur database yang digunakan mampu mendukung kebutuhan analitik operasional dengan baik. Performance testing semacam ini penting untuk memastikan data warehouse dapat menampung proses analisis skala besar secara responsif, serta siap digunakan dalam laporan periodik maupun dashboard interaktif oleh pengelola Kebun Raya ITERA.

Arsitektur Logikal Data Warehouse Kebun Raya ITERA

Arsitektur logikal Data Warehouse Kebun Raya ITERA dikembangkan berdasarkan skema konseptual dan logikal yang telah dirancang serta diimplementasikan pada Misi Pertama dan Kedua dari proyek ini. Struktur data mart ini terdiri dari dua fact table utama: Fact_Kunjungan (menyimpan data aktivitas kunjungan wisatawan dan aktivitas edukasi) dan Fact_Tanaman (mencatat proses penanaman, perubahan status konservasi, serta pertumbuhan koleksi tanaman). Kedua fact table dirancang dalam bentuk star schema yang terpisah secara fungsional namun terintegrasi secara semantik melalui *conformed dimensions*, yaitu Dim_Waktu, Dim_Tanaman, Dim_Fasilitas, Dim_Pengunjung, Dim_Ulasan, dan Dim_Spesies.

Setiap fact table beroperasi dengan granularitas harian, sesuai kebutuhan pelaporan dan analitik manajemen Kebun Raya ITERA yang membutuhkan insight berbasis waktu (hari, bulan, tahun). Dim_Waktu tidak hanya memuat atribut standar (tanggal, bulan, tahun, hari) namun juga label khusus seperti periode musim kunjungan, puncak liburan, dan event edukasi. Dim_Fasilitas merepresentasikan seluruh komponen fisik kebun (embung, greenhouse, labirin, gazebo, koleksi taman, dsb), serta kapasitas dan letaknya, yang digunakan untuk analisa distribusi pengunjung dan penggunaan fasilitas.

Hubungan antar dimensi Dim_Tanaman dan Dim_Spesies terhubung ke dua fakta berbeda—misalnya, *analisis pertumbuhan tanaman berdasarkan jenis spesies, asal daerah,*

dan status konservasi. Faktual data kunjungan dan ulasan juga dapat dikaitkan lewat dimensi pengunjung serta waktu, untuk membangun korelasi antara pengalaman wisata, musim kunjungan, fasilitas, dan rating. Mekanisme Slowly Changing Dimension Type 2 (SCD2) diterapkan pada status konservasi tanaman, riwayat perubahan koleksi, serta profil pengunjung yang mengalami update data penting (misal kategori pengunjung atau institusi asal), agar histori perubahan tetap terjaga secara longitudinal.

Seluruh dimensi dibangun dari data primer internal Kebun Raya ITERA (inventarisasi tanaman, log kunjungan harian, event) serta sumber sekunder seperti data statistik pariwisata lokal. Validasi dan pembersihan data telah dilakukan dalam pipeline ETL pada Misi Kedua, untuk memastikan data siap operasional dan reliabel. Relasi antartabel tidak hanya difungsikan untuk pelaporan standar (jumlah tanaman, tren kunjungan), namun juga mendukung analitik lanjutan seperti evaluasi efektivitas edukasi, tracking status konservasi, dan segmentasi pengunjung.

Dengan desain logikal ini, sistem mampu menjawab kebutuhan analitik manajemen Kebun Raya ITERA secara multidimensi, lintas proses, dan mendukung pelacakan historis yang akurat. Kombinasi *star schema* dan conformed dimensions menjamin fleksibilitas integrasi data lintas domain, serta mendukung prinsip transparansi dan pengambilan keputusan berbasis data untuk konservasi dan pengelolaan kebun raya secara modern.

Penjelasan ETL Pipeline (T-SQL)

ETL pipeline adalah rangkaian proses yang digunakan untuk mengekstrak data mentah dari berbagai sumber, melakukan transformasi (pembersihan dan penyesuaian format), lalu memuat data final ke dalam struktur data warehouse. Pada kasus ini, pipeline ETL diimplementasikan menggunakan skrip dan prosedur T-SQL secara terjadwal.

Alur ETL pipeline pada data mart:

1. Extract:
   Data dari sumber eksternal seperti file Excel, CSV, atau temp database diambil dan dimasukkan terlebih dahulu ke tabel staging menggunakan perintah INSERT.
   Misalnya, data pengunjung diekstrak ke tabel stg_pengunjung, data tanaman ke stg_tanaman.
2. Transform:
   Data dalam tabel staging dibersihkan dan disesuaikan. Transformasi dilakukan melalui query T-SQL, seperti menghapus duplikat, memperbaiki format tanggal, standarisasi penulisan, maupun validasi kategori. Prosedur ini juga dapat menambahkan atribut baru, melakukan lookup foreign key, dan menyesuaikan data sebelum di-load.
3. Load:
   Data yang telah melalui proses transformasi kemudian dimasukkan ke tabel utama data warehouse (dimension dan fact table) menggunakan query INSERT atau prosedur

T-SQL khusus. Misalnya, data hasil cleansing dari stg_pengunjung di-load ke dim_pengunjung, dan data kunjungan ke fact_kunjungan.

Pipeline ETL ini dijalankan secara batch, terjadwal setiap periode tertentu agar data warehouse selalu ter-update dan siap untuk dianalisis. Dengan implementasi berbasis T-SQL, proses ETL dapat dikontrol secara terstruktur, otomatis, dan mudah didokumentasikan—mendukung kebutuhan pelaporan bisnis dan analitik keberlanjutan.