

# Diseño e Implementación de Base de Datos Analítica - Día 5

Change Status

in-progress 40 min

## Learning Objectives

- 1 Comprender los principios del diseño de bases de datos analíticas
- 2 Aprender arquitecturas modernas para analytics (Lambda y Kappa)
- 3 Conocer estrategias de optimización para diferentes cargas de trabajo

Theory

Practice

Evidence

Quiz

## Activities and Learning

### Task 1: Arquitectura de Data Warehouse (10 minutos)

#### ¿Por qué necesitamos una arquitectura especial para analytics?

Los sistemas transaccionales (OLTP) están optimizados para operaciones individuales rápidas, pero los análisis requieren consultas complejas sobre grandes volúmenes de datos históricos. El data warehouse resuelve esta brecha.

#### Tres capas fundamentales:

**Staging Layer:** Área de preparación y validación inicial

**Integration Layer:** Datos normalizados y consistentes

**Access Layer:** Vistas optimizadas para consultas analíticas

#### Ejemplo conceptual - Staging Area:

```
-- Área de staging: datos crudos sin transformación
CREATE SCHEMA staging;

CREATE TABLE staging.raw_sales (
  transaction_id TEXT,
  customer_email TEXT,
  product_sku TEXT,
  quantity TEXT, -- Todavía como texto
  unit_price TEXT,
  sale_date TEXT,
  source_system TEXT,
  loaded_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

**Punto clave:** La staging area aísla las transformaciones del impacto en sistemas fuente.

### Task 2: Modelado Dimensional (10 minutos)

#### ¿Qué es el modelado dimensional?

Es una técnica para estructurar datos de manera intuitiva para analistas de negocio, usando dimensiones (quién, qué, cuándo, dónde) y hechos (medidas cuantificables).

#### Dimensiones vs Hechos:

**Dimensiones:** Atributos descriptivos (Cliente, Producto, Tiempo)

**Hechos:** Métricas medibles (Ventas, Cantidad, Ingresos)

#### Ejemplo simple - Esquema Estrella:

```
-- Dimensión Cliente
CREATE TABLE dim_customer (
  customer_id SERIAL PRIMARY KEY,
  natural_key TEXT UNIQUE, -- Email como clave natural
  full_name TEXT,
  registration_date DATE,
  customer_segment TEXT
);

-- Dimensión Producto
CREATE TABLE dim_product (
  product_id SERIAL PRIMARY KEY,
  sku TEXT UNIQUE,
  product_name TEXT,
  category TEXT,
  unit_cost DECIMAL(10,2)
```



```
);

-- Tabla de Hechos Ventas
CREATE TABLE fact_sales (
    sale_id SERIAL PRIMARY KEY,
    customer_id INTEGER REFERENCES dim_customer(customer_id),
    product_id INTEGER REFERENCES dim_product(product_id),
    sale_date DATE,
    quantity_sold INTEGER,
    unit_price DECIMAL(10,2),
    total_amount DECIMAL(10,2)
);
```

**Beneficio:** Consultas intuitivas como "ventas por categoría de cliente en el último trimestre".

**Task 3: Arquitecturas Modernas (10 minutos)**

**Arquitectura Lambda: Precisión histórica + Velocidad en tiempo real**

Combina dos caminos de procesamiento:

- Batch Layer:** Procesamiento preciso de datos históricos completos
- Speed Layer:** Aproximaciones rápidas para datos recientes

**Ventaja:** Balance entre precisión y velocidad.

**Arquitectura Kappa: Streaming-First**

Simplificación radical: todo fluye como streams, incluso el reprocesamiento histórico.

**Ventaja:** Simplicidad y facilidad de mantenimiento.

**Comparación conceptual:**

Aspecto	Lambda	Kappa
Complejidad	Alta (3 componentes)	Baja (1 pipeline)
Reprocesamiento	Complejo	Natural
Mantenimiento	Mayor costo	Menor costo

**Elección:** Lambda para analytics complejos, Kappa para streaming como fuente única.