

Guía de Estudio para la Evaluación a Título de Suficiencia Bases de Datos Avanzadas

Introducción

La presente guía tiene como objetivo orientar al estudiante en su preparación para la Evaluación a Título de Suficiencia (ETS) de la unidad de aprendizaje **Bases de Datos Avanzadas**. El contenido de este documento se basa en el temario oficial del programa de estudios y está enfocado en las habilidades prácticas que serán evaluadas.

Temario Oficial y Aplicación en el Examen

El examen está diseñado para evaluar los conocimientos del curso. A continuación, se desglosa el temario oficial y se explica cómo cada unidad se aplica en los dos ejercicios prácticos que componen la evaluación.

Unidades I, II y III: Business Intelligence y Almacenes de Datos

Estas tres unidades forman un bloque cohesivo que se evaluará íntegramente en el **Ejercicio 1**.

I. Sistemas de Soporte de Decisión y Almacenes de Datos

1.1 Sistemas de Soporte de decisiones (DSS)

1.1.1 Características de los DSS

1.1.2 Taxonomía de los DSS

1.1.3 Arquitecturas y plataformas para los DSS

1.2 Almacenes de Datos

1.2.1 Características de un Almacén de Datos

1.2.2 Arquitecturas de Data Warehouses y Data Marts

1.2.3 Aplicaciones de Data Warehouse

1.2.4 Data Lake (lagos de datos) y sus características

1.2.6 Arquitecturas de Data Lake

1.2.7 Comparativa entre Data Lakes y Data Marts

II. Diseño de Almacenes y Bancos de Datos

- 2.1 Diseño de Almacenes de datos (Data Warehouse)
- 2.2 Referencia Arquitectural de los Data Warehouse
- 2.3 Funcionamiento de los almacenes de Datos
- 2.4 Funciones Extracción, Transformación y Limpieza (ETL)

III. Diseño de Cubos OLAP

- 3.1 Cubos de Datos
 - 3.1.1 OLAP (On-Line Analytical Processing)
 - 3.1.2 Comparativa entre OLAP y OLTP
- 3.2 Modelado de Datos para almacenes de datos
 - 3.2.1 Modelo Relacional
 - 3.2.2 Modelo dimensional OLAP
 - 3.2.3 Tipos de Esquemas (Estrella, copo de nieve, etc.)
- 3.3 Operadores de cubos de datos
- 3.4 Cubos multidimensionales

Ejercicio 1: Implementación de un Data Warehouse para Análisis de Riesgo Sísmico

Descripción del Escenario

El Instituto Politécnico Nacional planea desarrollar un Sistema Nacional de Análisis de Riesgos Sísmicos para mejorar la toma de decisiones, investigación académica y planificación urbana en zonas de alto riesgo. El sistema debe integrar datos de sismos, demográficos y económicos.

Fuentes de Datos

Utilice los siguientes conjuntos de datos públicos:

- **Datos de Sismos:** Catálogo histórico del Servicio Sismológico Nacional (SSN).
 - <http://www2.ssn.unam.mx:8080/catalogo/>
- **Datos de Población:** Censo de Población y Vivienda 2020 del INEGI (Resultados por localidad ITER, archivo `iter_00_cpv2020_csv.zip`).
 - <https://www.inegi.org.mx/datosabiertos/>

- **Datos Económicos:** Censos Económicos 2019 del INEGI (Datos nacionales, archivo ce2019_nac_csv.zip).

- <https://www.inegi.org.mx/datosabiertos/>

Tareas a Realizar

1. **Tarea 1.- Diseño del Esquema del Data Warehouse:** Basado en los requerimientos del escenario, defina un esquema en estrella. A continuación, se presenta un modelo de referencia para las tablas del DWH que deberá implementar en PostgreSQL.

```
1 CREATE TABLE IF NOT EXISTS dim_economia (  
2     id_economia INT PRIMARY KEY,  
3     nombre_entidad VARCHAR(100),  
4     entidad DOUBLE PRECISION,  
5     produccion_bruta_total DOUBLE PRECISION,  
6     insumos_utilizados DOUBLE PRECISION,  
7     consumo_intermedio DOUBLE PRECISION,  
8     valor_agregado DOUBLE PRECISION,  
9     formacion_capital DOUBLE PRECISION,  
10    activos_fijos_adquiridos DOUBLE PRECISION  
11 );  
12  
13 CREATE TABLE IF NOT EXISTS dim_sismos (  
14     id_sismo INT PRIMARY KEY,  
15     magnitud DOUBLE PRECISION,  
16     latitud DOUBLE PRECISION,  
17     longitud DOUBLE PRECISION,  
18     profundidad DOUBLE PRECISION,  
19     referencia_de_localizacion VARCHAR(100),  
20     estado VARCHAR(100),  
21     nombre_estado VARCHAR(100)  
22 );  
23  
24 CREATE TABLE IF NOT EXISTS dim_tiempo (  
25     id_tiempo INT PRIMARY KEY,  
26     hora_utc TIME,  
27     fecha DATE,  
28     anio INT,  
29     mes INT,  
30     dia INT,  
31     trimestre INT  
32 );  
33  
34 CREATE TABLE IF NOT EXISTS dim_zonas (  
35     id_zonas INT PRIMARY KEY,  
36     entidad INT,  
37     nom_ent VARCHAR(100),  
38     pobtot BIGINT,  
39     pobfem BIGINT,  
40     pobmas BIGINT  
41 );  
42
```

```

43 CREATE TABLE fact_impacto_sismos_imputed (
44     id_sismo INT,
45     id_zonas INT,
46     id_economia INT,
47     id_tiempo INT,
48     poblacion_afectada BIGINT,
49     impacto_economico DOUBLE PRECISION,
50     determinante_de_riesgo DOUBLE PRECISION,
51     riesgo_proporcional DOUBLE PRECISION,
52     indice_zscore DOUBLE PRECISION,
53     FOREIGN KEY (id_sismo) REFERENCES dim_sismos(id_sismo),
54     FOREIGN KEY (id_zonas) REFERENCES dim_zonas(id_zonas),
55     FOREIGN KEY (id_economia) REFERENCES dim_economia(id_economia),
56     FOREIGN KEY (id_tiempo) REFERENCES dim_tiempo(id_tiempo)
57 );

```

Listing 1: DDL para el esquema del Data Warehouse de Sismos.

2. **Tarea 2.- Implementación del Proceso ETL:** Para poblar el Data Warehouse, deberá implementar un proceso ETL. Puede utilizar herramientas como **Pentaho Data Integration**, **KNIME**, o desarrollar un script a la medida con **Python** (usando librerías como Pandas y Psycopg2), por ejemplo, en un notebook de Google Colab. El objetivo es extraer los datos de los CSV, transformarlos (limpieza, normalización, creación de llaves subrogadas) y cargarlos en las tablas de su DWH en PostgreSQL.

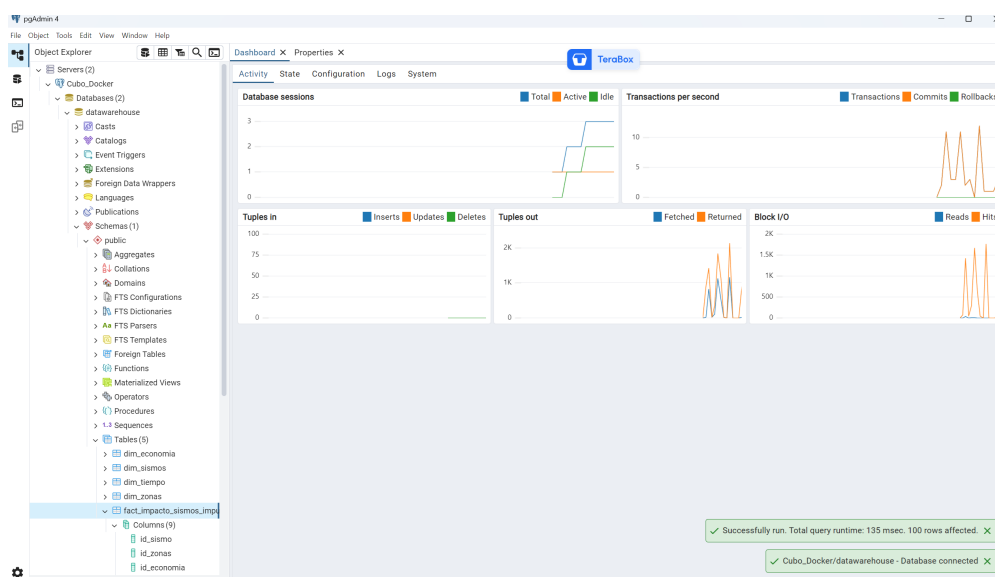


Figura 1: DWH funcional en PostgreSQL.

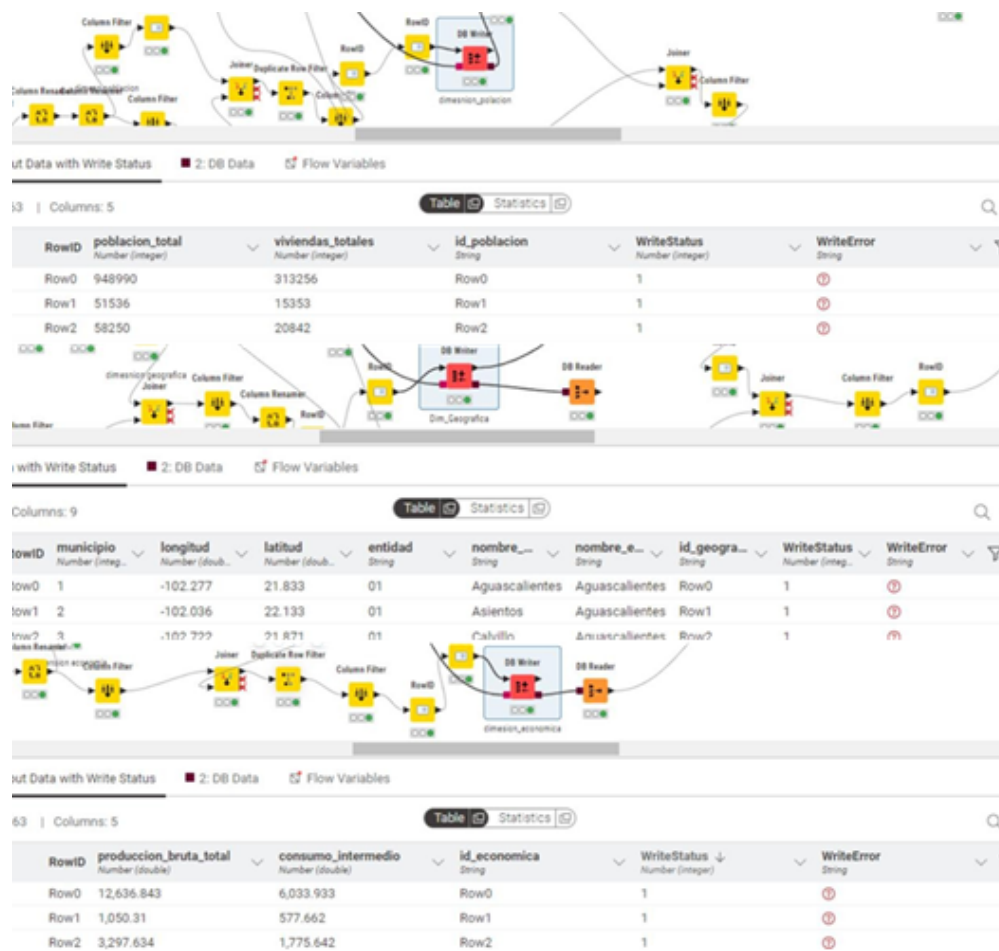


Figura 2: Proceso ETL diseñado.

El punto de partida de todo proceso ETL es la conexión a las fuentes de datos. En un entorno de desarrollo profesional, se utilizan nodos o conectores específicos para establecer la comunicación con las bases de datos donde residen los datos crudos. Para este ejercicio, deberá configurar la conexión a su base de datos PostgreSQL donde cargó inicialmente los archivos CSV del SSN e INEGI.

- **Herramientas:** Utilice los nodos PostgreSQL Connector y DB Query Reader en KNIME, o sus equivalentes en la herramienta ETL de su elección.
- **Objetivo:** Importar las tablas de sismos, población y economía a su flujo de trabajo para iniciar la transformación.

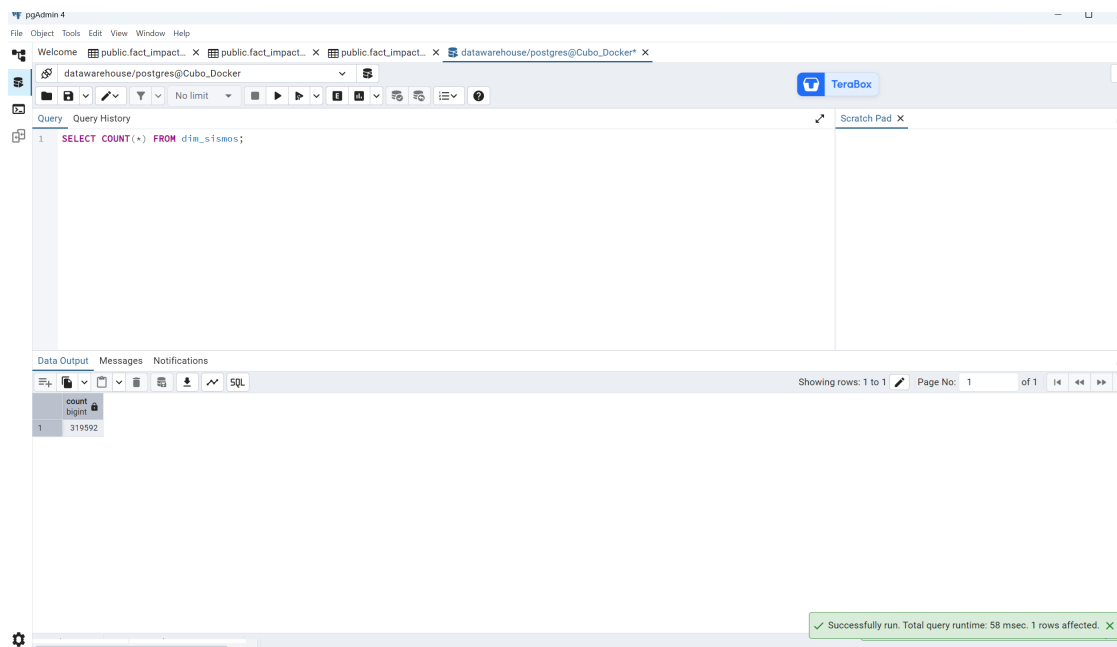


Figura 3: Datos transformados y cargados en PostgreSQL.

Tarea 3.- Transformación de Datos. Esta es la fase más importante, donde se limpia, enriquece y modela la información para adecuarla al esquema en estrella del Data Warehouse.

- **Limpieza y Normalización:** Utilice nodos como **Missing Value** para manejar datos faltantes, **String Manipulation** para estandarizar texto, y **Column Filter** para eliminar columnas innecesarias.
- **Creación de Dimensiones:** A partir de los datos extraídos, deberá construir cada una de las dimensiones. Por ejemplo, para la **dimensión de tiempo**, se utilizan nodos como **String to Date&Time** y **Date&Time Part Extractor** para descomponer una fecha en año, mes, trimestre y día.
- **Unión de Datos:** Utilice el nodo **Joiner** para combinar información de diferentes fuentes. Por ejemplo, unificar datos de población y economía para crear una dimensión geográfica consolidada.

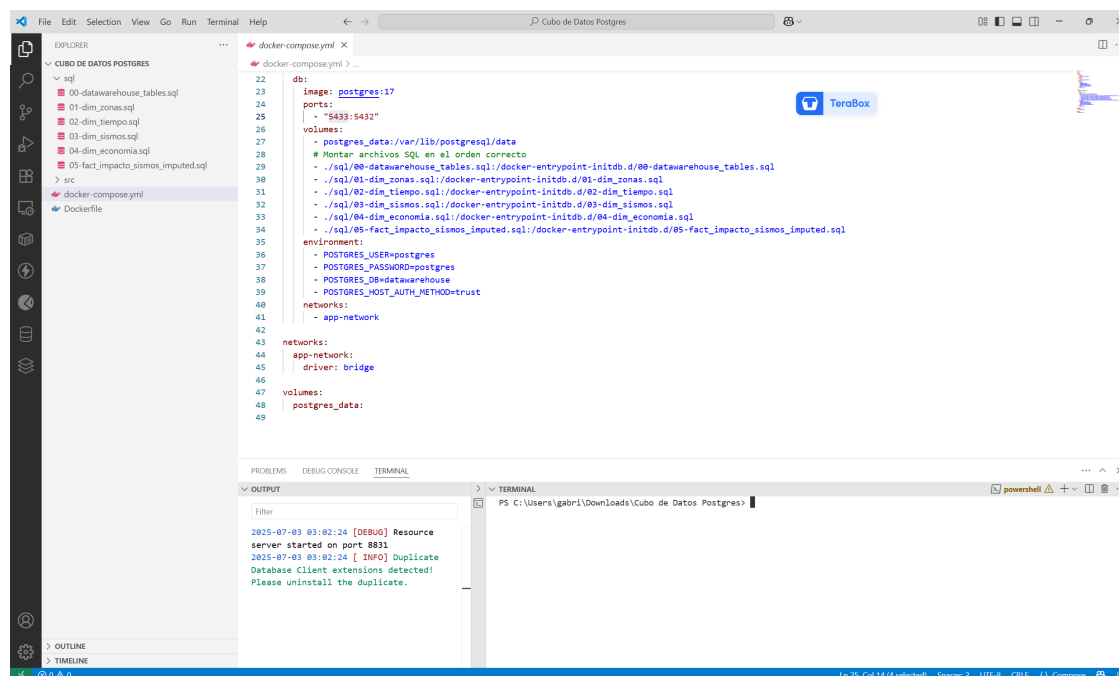


Figura 4: Ejemplo de un docker-compose.yml completo para la creación del DWH.

3. Tarea 4: Diseño, Construcción y Análisis del Cubo OLAP.

Una vez que el Data Warehouse está poblado y verificado, el paso final es construir la capa de análisis multidimensional. Esto se logra mediante un cubo OLAP, que permite consultas complejas y de alto rendimiento.

4.1. Diseño Conceptual del Cubo El objetivo es modelar un cubo que permita analizar el impacto de los sismos desde múltiples perspectivas. Basado en el esquema del DWH, se definen las siguientes dimensiones, jerarquías y medidas:

- **Dimensiones y Jerarquías:**
 - **Tiempo:** Permite analizar los eventos a lo largo del tiempo. Su jerarquía principal es: **Año ¿Mes ¿Día.**
 - **Geografía:** Organiza los datos espacialmente. Su jerarquía es: **Estado ¿Municipio.**
 - **Sismo:** Clasifica las características de cada evento. Se puede definir una jerarquía como: **Intensidad (categoría) ¿Profundidad.**
- **Medidas:** Son los valores numéricos que se desean analizar en las intersecciones del cubo, tales como: *Cantidad de Sismos, Población Afectada Total, e Impacto Económico Total.*

4.2. Implementación del Cubo Para la implementación, es mandatorio utilizar un **servidor OLAP dedicado** (como Pentaho BI Server o Apache Kylin). El diseño

conceptual se implementa utilizando una herramienta como **Pentaho Schema Workbench**, la cual genera un archivo de esquema Mondrian (‘.xml’) que se publica en el servidor.

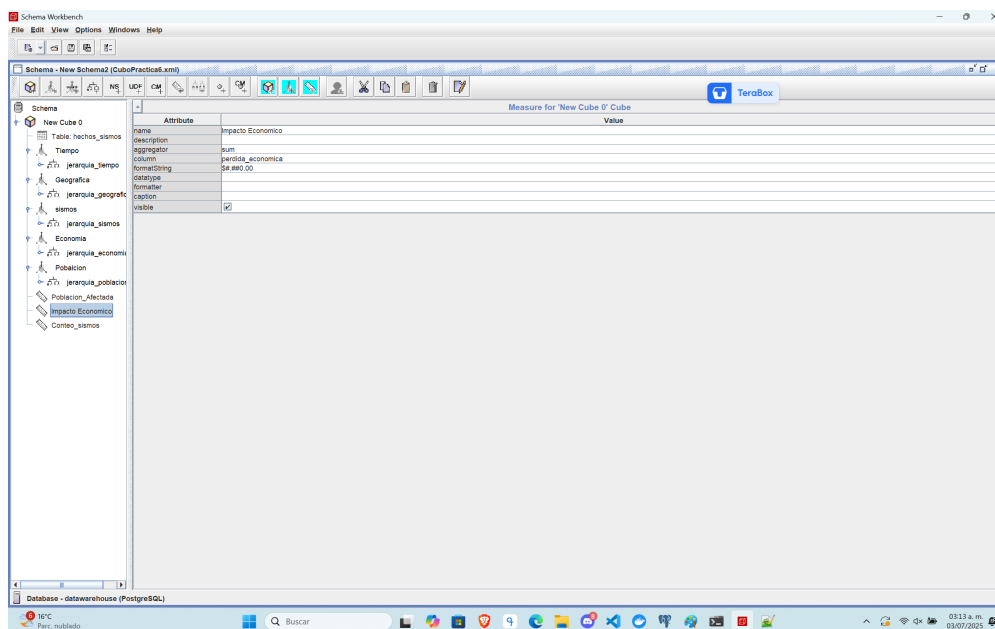


Figura 5: Ejemplo de la definición visual del Cubo OLAP.

4.3. Análisis del Cubo con Operaciones OLAP y MDX Con el cubo publicado, el análisis se realiza mediante consultas **MDX (Multidimensional Expressions)**. A continuación se muestran ejemplos de las operaciones fundamentales.

Drill Down (Profundizar): Navega de un nivel general a uno más detallado.

Pregunta de Negocio: Mostrar la población total afectada por mes durante el año 2018.

```

1  -- Muestra la poblacion total afectada por mes en el anio 2018
2  SELECT
3      { [Measures].[pob_tot] } ON COLUMNS,
4      NON EMPTY {
5          Order(
6              [Tiempo].[Año].[2018].Children,
7              Val([Tiempo].CurrentMember.Name),
8              BASC
9          )
10     } ON ROWS
11 FROM
12     [CUBO_test]

```

Listing 2: Consulta MDX para operación Drill Down.

Roll Up (Consolidar): Agrega datos desde un nivel de detalle a uno más general.

Pregunta de Negocio: Mostrar la población total afectada por año para el estado de Oaxaca.

```
1 -- Realiza una consulta para todo el estado de Oaxaca para mostrar
2 -- la poblacion afectada total por anio
3 SELECT
4     { [Measures].[pob_tot] } ON COLUMNS,
5     NON EMPTY { [Tiempo].[Año].Members } ON ROWS
6 FROM
7     [CUBO_test]
8 WHERE
9     ( [Geografia].[Estado].[Oaxaca] )
```

Listing 3: Consulta MDX para operación Roll Up.

Slice (Rebanar): Filtra el cubo por un único miembro de una dimensión.

Pregunta de Negocio: ¿Cuál fue la cantidad total de sismos en el estado de 'Guerrero'?

```
1 SELECT
2     {[Measures].[Cantidad Sismos]} ON COLUMNS
3 FROM
4     [CUBO_test]
5 WHERE
6     ([Geografia].[Estado].[Guerrero]);
```

Listing 4: Consulta MDX para operación Slice.

Pivot (Pivotar): Rota los ejes del cubo para cambiar la perspectiva de análisis.

Pregunta de Negocio: Crear una tabla que muestre la cantidad de sismos por año para cada categoría de magnitud.

```
1 SELECT
2     {[Sismo].[Categoria Magnitud].Members} ON COLUMNS,
3     NON EMPTY {[Tiempo].[Año].Members} ON ROWS
4 FROM
5     [CUBO_test]
6 WHERE
7     ([Measures].[Cantidad Sismos]);
```

Listing 5: Consulta MDX para operación Pivot.

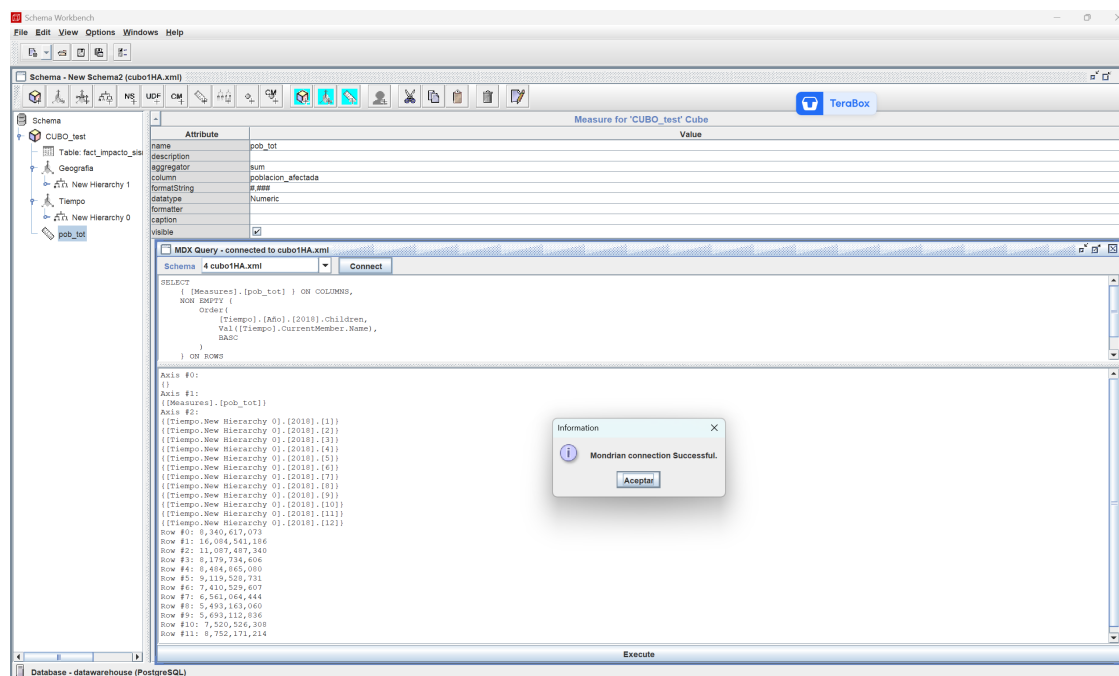


Figura 6: Ejemplo de drill-down en MDX.

Unidad IV: Bases de Datos No Relacionales

Esta unidad se evaluará íntegramente en el **Ejercicio Práctico 2**, donde deberá aplicar diferentes modelos NoSQL a un caso de uso empresarial.

IV. Temario de Bases de Datos No Relacionales

- 4.1 Bases de Datos orientadas a objetos
 - 4.1.1-4.1.4 Características, Arquitecturas, Ventajas/Desventajas y Aplicaciones.
- 4.2 Bases de Datos orientadas a documentos
 - 4.2.1-4.2.4 Características, Arquitecturas, Ventajas/Desventajas y Aplicaciones.
- 4.3 Bases de Datos clave-valor
 - 4.3.1-4.3.4 Características, Arquitecturas, Ventajas/Desventajas y Aplicaciones.
- 4.4 Bases de Datos orientadas a grafos
 - 4.4.1-4.4.4 Características, Arquitecturas, Ventajas/Desventajas y Aplicaciones.
- 4.5 Otras tendencias (Bases de datos móviles, geográficas, orientadas al tiempo).

Aplicación Práctica: Ejercicio 2 - Casos de Estudio para Modelado NoSQL

Para la evaluación de la Unidad IV, se presentan 8 escenarios de negocio. El alumno deberá desarrollar las transformaciones a los diferentes modelos NoSQL como se indica al final de esta sección.

Escenario 1: Clínica “GUADALUPANA” (Gestión de pacientes y médicos)

La clínica “GUADALUPANA” necesita informatizarse para gestionar de manera eficiente sus operaciones. El diseño de la base de datos deberá considerar:

- **Pacientes:** Código único, nombre, apellidos, dirección, colonia, estado, código postal, teléfono y fecha de nacimiento.
- **Médicos:** Código único, nombre, apellidos, teléfono y especialidad.
- **Ingresos:** Código de ingreso (incremental), número de consultorio, cama asignada y fecha de ingreso.
- **Restricciones:** Un médico puede atender varios ingresos, pero un ingreso solo puede estar asociado a un médico. Un paciente puede realizar múltiples ingresos.

Escenario 2: Centro de Desarrollo Infantil (CENDI IPN) - Biblioteca escolar

Se debe desarrollar una base de datos para gestionar el préstamo de libros a los alumnos, integrando información como autores, editoriales y especialidades.

- **Alumnos:** Código único, nombre, escuela y ciclo escolar.

- **Préstamos:** Fecha de préstamo y fecha de devolución.
- **Libros:** Código único, título, cantidad de páginas.
- **Autores:** Código único, nombre, email.
- **Editoriales:** Código único, nombre, dirección y teléfono.
- **Restricciones:** Múltiples relaciones muchos-a-muchos (alumno-libro, libro-autor).

Escenario 3: Fabricación de motores en Mastrettadesign - Tecnoidea La empresa expande sus operaciones al diseño y fabricación de motores, gestionando piezas, fabricantes, ensamblaje y diferentes tipos de motores.

- **Piezas:** ID único, código de pieza, código de fabricante, descripción, ruta de archivo CAD y material. Las piezas pueden ser compuestas (formadas por otras piezas básicas).
- **Motores:** ID único, descripción, número de piezas, ruta CAD. Con subtipos para **motocicletas** (caballos de fuerza, refrigeración) y **automóviles** (potencia fiscal, tipo de anclaje).
- **Operarios:** ID único, nombre, sueldo. Con un expediente especial para eventos extraordinarios.
- **Ensamblaje:** Se registran dos fases por cada pieza-motor: instalación y revisión, cada una con su operario responsable.

Escenario 4: Distribuidora de Agua Potable Una empresa de reparto de agua potable necesita una aplicación móvil con seguimiento en tiempo real y notificaciones push.

- **Usuarios:** Clientes, repartidores y superusuarios, cada uno con roles y permisos distintos.
- **Gestión:** Se deben gestionar clientes, productos (inventario), pedidos (con estado y repartidor asignado) y repartidores (con zonas de trabajo).
- **Funcionalidades Avanzadas:** Seguimiento GPS en tiempo real, notificaciones push y gestión de rutas.

Escenario 5: Sistema Médico Integral Un hospital privado busca digitalizar la gestión de expedientes clínicos y procesos administrativos, cumpliendo con normativas de salud como la NOM-004-SSA3-2012.

- **Usuarios:** Pacientes, médicos y administradores con diferentes niveles de acceso.
- **Gestión:** Perfiles de pacientes (alergias, historial), consultas médicas (diagnósticos, recetas), consentimientos informados y un sistema de tickets para soporte.

Escenario 6: Sistema de Asistencia Nutricional Un startup de salud necesita un sistema para que nutricionistas diseñen y monitoreen planes de dieta personalizados para sus pacientes en tiempo real.

- **Gestión:** Perfiles de pacientes (métricas diarias, peso), planes de dieta (alimentos, tabla nutricional, duración) y progreso del paciente (cumplimiento, logros).
- **Análisis:** El sistema debe generar reportes automáticos sobre el consumo calórico y el cumplimiento de metas.

Escenario 7: Sistema de Búsqueda y Recomendación Multimodal Un sistema que integra contenido de múltiples categorías (libros, series, videojuegos, anime) desde diversas APIs públicas para ofrecer recomendaciones personalizadas.

- **Contenido:** Se deben manejar atributos específicos para cada categoría (libros, series, etc.) y relacionarlos mediante tags semánticos.
- **Usuarios:** Gestionar perfiles, historial de búsquedas y listas de favoritos.
- **Recomendaciones:** Generar sugerencias cruzadas entre categorías basadas en el perfil del usuario y tendencias.

Escenario 8: Sistema de Búsqueda y Recomendación de Artículos Científicos Una plataforma para facilitar a estudiantes e investigadores el acceso a artículos científicos, integrando APIs como CrossRef o Semantic Scholar.

- **Gestión:** Perfiles de usuario con preferencias temáticas, almacenamiento de metadatos de artículos (título, autores, resumen, etc.), historial de búsquedas y favoritos.
- **Funcionalidad:** Búsquedas avanzadas con filtros y un motor de recomendaciones basado en intereses y similitud de contenido.

Guía de Implementación para el Ejercicio 2: Modelado NoSQL

Esta sección evalúa su dominio de la **Unidad IV: Bases de Datos No Relacionales**, aplicando los conceptos aprendidos en las prácticas 7, 8, 9 y 10. La tarea consiste en **seleccionar uno de los 8 escenarios** descritos anteriormente y modelar su solución utilizando cuatro paradigmas diferentes.

Para cada modelo, deberá diseñar la estructura de datos, justificar sus decisiones, implementar un prototipo y resolver 3 preguntas de negocio que usted mismo plantee. A continuación, se detalla lo que se espera para cada implementación.

Modelo Relacional-Objetual El objetivo aquí es extender el modelo relacional tradicional con características de la programación orientada a objetos, utilizando PostgreSQL.

- **Técnicas a Aplicar:** Deberá utilizar **tipos de datos compuestos** ('CREATE TYPE') para encapsular atributos complejos (ej. una dirección o las especificaciones de una pieza) y/o **herencia de tablas** ('INHERITS') para modelar jerarquías .es-un" (ej. en el escenario de motores, un 'MotorDeAuto' y un 'MotorDeMoto' heredan de una tabla base 'Motor').
- **Entregables Clave:** Un diagrama de clases UML que represente su modelo de objetos y el código SQL que implemente estos tipos y tablas.

```

cendi_orm.py > Autor
1 from sqlalchemy import create_engine, Column, Integer, String, ForeignKey, Date, Text
2 from sqlalchemy.orm import declarative_base
3 from sqlalchemy.orm import relationship, sessionmaker
4
5 # Define el motor de conexión
6 engine = create_engine('postgres://postgres:patricia15@localhost:5432/base_biblioteca')
7
8 # Base declarativa
9 Base = declarative_base()
10
11 # Tabla Autores
12 class Autor(Base):
13     __tablename__ = 'autores'
14
15     id_autor = Column(Integer, primary_key=True)
16     nombre = Column(String(30))
17     ap_paterno = Column(String(20))
18     ap_materno = Column(String(20))
19     email = Column(String(20))
20
21     libros = relationship('Libro', secondary='autores_libros', back_populates='autores')
22
23 FOREIGN KEY(id_libro) REFERENCES libros (id_libro)
24 )
25 ]
26 (Background on this error at: https://sqlalche.me/e/20/f405)
27
28 C:\Users\User\Desktop\practica7>python cendi_orm.py
29 Tablas creadas exitosamente.
30
31 C:\Users\User\Desktop\practica7>

```

Figura 7: Ejemplo del Modelo Relacional-Objetual

Modelo Orientado a Documentos Aquí deberá utilizar MongoDB para crear un modelo de datos flexible y semi-estructurado. La decisión de diseño más importante será cómo manejar las relaciones.

- **Diseño Estratégico:** Deberá justificar por qué elige **embeber** documentos dentro de otros (relación uno-a-pocos, acceso conjunto) o usar **referencias** (relación uno-a-muchos, datos que se actualizan con frecuencia). Por ejemplo, en el escenario de motores, las piezas podrían embeberse dentro del documento del motor, mientras que el fabricante podría ser una referencia.

- **Validación y Representación:** Es requisito crear un **JSON Schema** para validar la estructura de su colección principal y un **XML Schema (XSD)** que represente esa misma estructura.
- **Consultas:** Se espera que sus consultas de negocio aprovechen el **Aggregation Framework** de MongoDB para realizar análisis complejos.

```
test> db.createCollection("users", {
...   validator: {
...     $jsonSchema: {
...       bsonType: "object",
...       required: ["name", "email"],
...       properties: {
...         name: {
...           bsonType: "string",
...           description: "Nombre del usuario - requerido"
...         },
...         email: {
...           bsonType: "string",
...           pattern: "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$",
...           description: "Email válido - requerido"
...         },
...         bio: {
...           bsonType: "string",
...           description: "Biografía del usuario - opcional"
...         },
...         createdAt: {
...           bsonType: "date",
...           description: "Fecha de creación"
...         }
...       }
...     }
...   }
... });
{ ok: 1 }
```

Figura 8: Ejemplo de Schema JSON

Modelo Clave-Valor Esta sección requiere aplicar Redis para un caso de uso que demande alta velocidad y baja latencia. No se trata de modelar todo el sistema, sino de resolver un problema específico de manera eficiente.

- **Caso de Uso:** Deberá proponer un caso de uso apropiado para su escenario, como la implementación de un sistema de **caché** para datos consultados frecuentemente (ej. los detalles de una pieza en el sistema de motores) o el **seguimiento de estado en tiempo real** (ej. el estado actual del ensamblaje de un motor específico).
- **Diseño de Claves:** Defina una convención de nomenclatura clara y eficiente para sus claves (ej. 'pieza:12345:specs', 'motor:98765:status').

- **Estructuras de Datos:** Utilice los tipos de datos de Redis más adecuados para su solución (Strings, Hashes, Lists, etc.).

Modelo Orientado a Grafos El objetivo es modelar y explotar las **relaciones complejas** inherentes a su escenario utilizando Neo4j.

- **Modelado del Grafo:** Identifique las entidades principales como **Nodos** (ej. 'Pieza', 'Motor', 'Operario', 'Fabricante') y las acciones o conexiones como **Relaciones** (ej. 'CONTIENE', 'FABRICADO_POR', 'ENSAMBLADO_POR'). Debe presentar un diagrama de este modelo.
- **Lenguaje Cypher:** Utilice el lenguaje declarativo Cypher para crear la estructura del grafo y, fundamentalmente, para escribir consultas que atraviesen el grafo y respondan a preguntas complejas que serían muy difíciles de resolver en un modelo relacional (ej. "¿Qué otros motores utilizan piezas del mismo fabricante que las del motor 'Mastretta MXT'?").

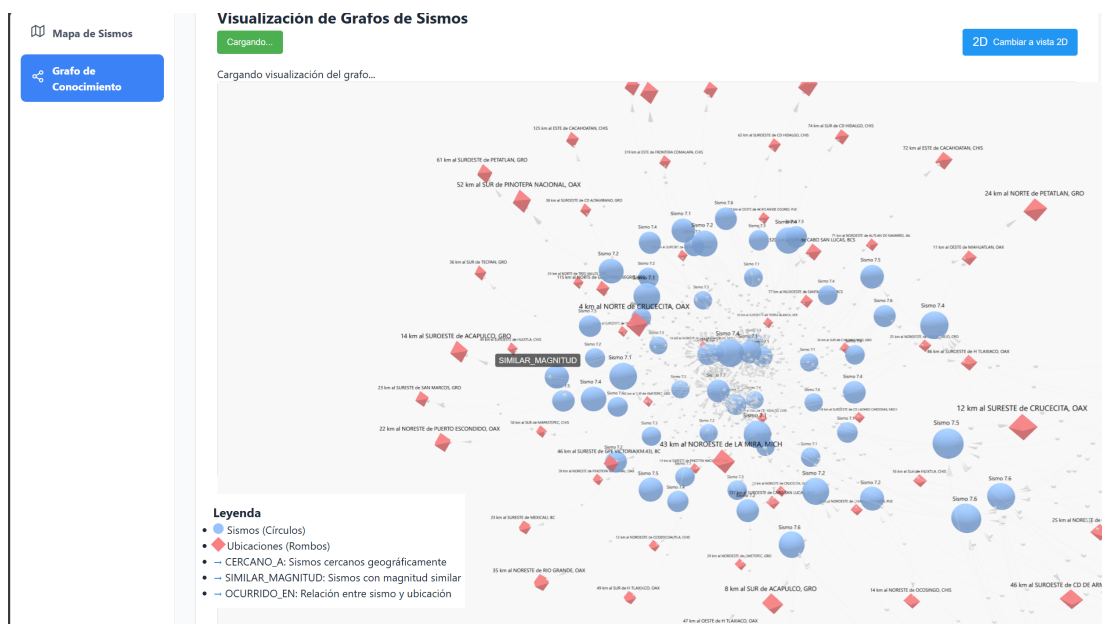


Figura 9: Ejemplo del Modelo Orientado a Grafos para el DWH de sismos.