Instituto Tecnológico de Buenos Aires

TPO

72.41 - Base de Datos II

Integrantes:

Curran, Luke - 62030

Dalton, Ian - 62317

Tordomar, Nicolás - 62250

Fecha de entrega: 20 de junio de 2025

Introducción

El objetivo del trabajo práctico es diseñar e implementar una capa de persistencia políglota utilizando al menos dos bases de datos NoSQL.

El primer paso fue analizar la información que se desea persistir y las consultas que se pretenden realizar. A partir de este análisis, se seleccionaron las bases de datos más adecuadas para cada tipo de dato, teniendo en cuenta la motivación detrás de cada elección, así como también las limitaciones propias de cada tecnología.

Una vez elegidas las bases, se dividieron las consultas y requerimientos entre ellas, buscando aprovechar las fortalezas particulares de cada motor de base de datos.

Elección de bases de datos

Se eligieron MongoDB y Neo4j como bases de datos NoSQL para implementar la capa de persistencia del sistema.

MongoDB fue utilizada para las consultas sobre datos estructurados y aquellas que requieren agregaciones simples (totales, conteos), sin necesidad de joins complejos. Su capacidad de escalar horizontalmente y su rendimiento en operaciones CRUD la convierten en una opción eficiente para manejar grandes volúmenes de información. Además, permite ordenar órdenes por fecha y generar vistas simuladas de manera flexible.

Neo4j, por su parte, fue seleccionada para resolver consultas que implican múltiples relaciones o saltos entre entidades, que en Mongo por ejemplo implicaría múltiples joins o subconsultas anidadas. En Neo4j, estas operaciones se expresan de forma sencilla mediante patrones MATCH, optimizando el rendimiento. Es ideal para modelar y consultar relaciones como proveedores ↔ órdenes ↔ productos, facilitando la navegación del grafo con gran eficiencia.

Pasos para Ejecutar el Proyecto

1. En Github, abrí el proyecto directamente en Codespaces con clic en "Code" y seleccionando "Open with Codespaces".

2. Levantar los contenedores con docker-compose up -d. Esto va a levantar los contenedores de MongoDB y Neo4j y cargar la app desarrollada con Fast API automáticamente.

3. Te va a avisar que puerto 8000 está corriendo, pero los datos todavía tienen que cargar. Se generará un enlace al servidor, que puede demorar unos segundos en estar disponible. Cuando te cuente que no hay atributos, sumá "/docs" al final de la URL y recarga la pantalla.

4. Ahí tendrás acceso a todas las 15 consultas que corresponden a las 15 consultas en el TP para probar y testear.

Queries

Query 1

Obtener los datos de los proveedores activos y habilitados junto con sus teléfonos.
Endpoint: /proveedores/activos-habilitados

Query 2

Obtener el/los teléfono/s y el código del/los proveedor/es que contengan la palabra “Tecnología” en su razón social.
Endpoint: /proveedores/telefonos-tecnologia

Query 3

Mostrar cada teléfono junto con los datos del proveedor
Endpoint: /proveedores/telefonos

Query 4

Obtener todos los proveedores que tengan registrada al menos una orden de pedido.
Endpoint: /proveedores/proveedores-con-orden

Query 5

Identificar todos los proveedores que NO tengan registrada ninguna orden de pedido.
Endpoint: /proveedores/proveedores-sin-orden

Query 6

Devolver todos los proveedores, con la cantidad de órdenes que tienen registradas y el monto total pedido, con y sin IVA incluido (si no tienen órdenes registradas considerar

cantidad y monto en 0).
Endpoint: /proveedores/proveedores-cantidad-ordenes

Query 7

Listar los datos de todas las órdenes que hayan sido pedidas al proveedor cuyo CUIT es 30-66060817-5.
Endpoint: /ordenes/ordenes-cuit

Query 8

Mostrar los productos que han sido pedido al menos 1 vez.
Endpoint: /productos/productos-pedidos

Query 9

Listar los datos de todas las órdenes de pedido que contengan productos de la marca "COTO"
Endpoint: /ordenes/ordenes-coto

Todas las vistas son creadas al momento del load. Lo que realizan las queries es acceder a los datos de la vista.

Query 10

Se necesita crear una vista que devuelva los datos de las órdenes de pedido ordenadas por fecha (incluyendo la razón social del proveedor y el total de la orden sin y con IVA).
Endpoint: /ordenes/ordenes-por-fecha

Query 11

Crear una vista que devuelva todos los productos que aún NO han sido pedidos.
Endpoint: /productos/productos-no-pedidos

Query 12

Crear una vista que devuelva los datos de los proveedores activos que están inhabilitados
.Endpoint: /proveedores/proveedores-activos-inhabilitados

Query 13

Implementar la funcionalidad que permita crear nuevos proveedores, eliminar y modificar los ya existentes.

El Endpoint base es /proveedores. Se accede con los métodos POST,PUT y DELETE

Query 14

Implementar la funcionalidad que permita crear nuevos productos y modificar los ya existentes.

El Endpoint base es /productos. Se accede con los métodos POST y PUT

Query 15

Implementar la funcionalidad que permita registrar nuevas órdenes de pedido a los proveedores si corresponde

El Endpoint base es /ordenes. Se accede con el método POST

Resultados

Se deja un link donde se pueden ver los resultados para mayor comodidad.

https://real-fireman-684.notion.site/Resultados-217888bd6415806987e7d3636ccd9c9f?pvs=73