Instituto Tecnológico de Buenos Aires

Base de Datos 2

Trabajo Práctico Especial

AÑO 2023

INFORME

Alumnos:

- Canevaro Bautista Ignacio - 62179
- Arnaude Juan Segundo - 62184
- Wodtke Matías - 62098

Fecha de entrega: 17/11/2023

Comisión: S

Profesores: Cecilia Rodríguez Babino y Guillermo Rodríguez

Introducción **3**
Queries-Vistas PostgreSQL **3**
Migración de datos hacia Mongo **3**
Queries-Vistas Mongo **5**
API para PostgreSQL **5**
Conclusiones **5**
Anexo **6**
Consultas PostgreSQL 6
Vistas PostgreSQL 7
Consultas MongoDB 8
Vistas MongoDB 11

Introducción

Buscando mejorar nuestras habilidades con el uso de las bases de datos relacionales y no relacionales, se desarrollaron consultas y vistas en PostgreSQL y MongoDB, y migración de datos de bases de datos relacionales a no relacionales. También se desarrollaron APIs que se pueden consultar en la base de datos PostgreSQL.

Queries-Vistas PostgreSQL

(Ver anexo para las queries)

La definición de las queries fue bastante sencilla, se siguieron lineamientos de lo que practicamos durante todo el año. Además, se contaba con diagramas de entidad relación que siempre hace más sencillo la creación de las queries, ya que ayudan a visualizar el modelado lo que simplifica mucho en momentos que es necesario hacer uso de joins.

No se presentaron muchas complicaciones en estos puntos.

Migración de datos hacia Mongo

La migración tuvo su complejidad. Ningún integrante del grupo había migrado alguna vez datos de una base de datos hacia otra, y el hecho de que una sea relacional y otra no parecía que iba a complicar el proceso.

Para solucionar esta problemática se optó por lo siguiente, primero debemos copiar las tablas a distintos archivos .csv dentro de nuestro container Docker, para esto corremos el script **from_sql_table_to_csv_file.sql**. Luego, en la consola corremos los siguientes comandos para copiar los archivos de docker a nuestra computadora local.

```
docker cp MypostgreSQL:/tmp/e01_cliente.csv
<direccion-donde-guardar-los-csv-localmente>/e01_cliente.csv

docker cp MypostgreSQL:/tmp/e01_detalle_factura.csv
<direccion-donde-guardar-los-csv-localmente>/e01_detalle_factura.csv

docker cp MypostgreSQL:/tmp/e01_factura.csv
<direccion-donde-guardar-los-csv-localmente>/e01_factura.csv

docker cp MypostgreSQL:/tmp/e01_producto.csv
<direccion-donde-guardar-los-csv-localmente>/e01_producto.csv
```

```
docker cp MypostgreSQL:/tmp/e01_telefono.csv
<direccion-donde-guardar-los-csv-localmente>/e01_telefono.csv
```

Podemos ver que los datos se encuentran en formato .csv en la dirección especificada.
En nuestro container de Docker que va a correr mongo creamos el siguiente directorio:

```
mkdir /usr/src/app
```

Ahora debemos pasar los archivos de nuestra máquina local a `/usr/src/app` entonces corremos los siguientes comandos:

```
docker cp
<direccion-donde-guardar-los-csv-localmente>/e01_cliente.csv
Mymongo:/usr/src/app/e01_cliente.csv

docker cp
<direccion-donde-guardar-los-csv-localmente>/e01_detalle_factura.csv
Mymongo:/usr/src/app/e01_detalle_factura.csv

docker cp
<direccion-donde-guardar-los-csv-localmente>/e01_factura.csv
Mymongo:/usr/src/app/e01_factura.csv

docker cp
<direccion-donde-guardar-los-csv-localmente>/e01_producto.csv
Mymongo:/usr/src/app/e01_producto.csv

docker cp
<direccion-donde-guardar-los-csv-localmente>/e01_telefono.csv
Mymongo:/usr/src/app/e01_telefono.csv
```

Ahora tenemos los archivos en el directorio `/usr/src/app` asi que nos queda migrar estos a nuestra base de datos de mongo:

```
docker exec Mymongo mongoimport -d mydb -c e01_cliente --type csv
--file /usr/src/app/e01_cliente.csv --headerline

docker exec Mymongo mongoimport -d mydb -c e01_detalle_factura
--type csv --file /usr/src/app/e01_detalle_factura.csv --headerline

docker exec Mymongo mongoimport -d mydb -c e01_factura --type csv
--file /usr/src/app/e01_factura.csv --headerline
```

```
docker exec Mymongo mongoimport -d mydb -c e01_producto --type csv
        --file /usr/src/app/e01_producto.csv --headerline

docker exec Mymongo mongoimport -d mydb -c e01_telefono --type csv
        --file /usr/src/app/e01_telefono.csv --headerline
```

Listo! Ya tenemos todos los archivos migrados a MongoDB dentro de la base de datos mydb.

Queries-Vistas Mongo

(Ver anexo para las queries)

Las consultas de Mongo tampoco presentaron mucha dificultad. Una vez ya hechas las de PostgreSQL se hizo sencillo el desarrollo en PostgreSQL debido a la similitud en la lógica de ambas consultas, aunque cabe aclarar, con distinta sintaxis.

Una problemática encontrada fue el hecho de tener que hacer consultas con agregación con lookups ya que los datos estaban normalizados debido a que los datos fueron migrados desde una base de datos relacional y el modelado de datos no es óptimo.

API para PostgreSQL

Para el desarrollo de la API habían muchas opciones para elegir: Django-Rest, Flask, Spring Boot, FastApi. Finalmente se decidió por hacer una API Rest con Express.js y Node.js en javascript por su simplicidad y popularidad.

Ningún integrante había hecho alguna vez una API, así que no fue sencillo ya que es algo que requiere muchos pasos y hubo que investigar en varios repositorios distintos y documentación.

Se utilizó Postman para el testeo de dicha API.

Conclusiones

Durante el desarrollo de este trabajo se encontraron problemas que fueron más complejos de abordar que otros. Sin embargo, se lograron cumplir todos los requisitos.

Un elemento que ayudó a que el desarrollo de este trabajo pueda ser finalizado es el hecho de que PostgreSQL y MongoDB (a pesar de ser relativamente joven) son dos de las bases de datos más populares, esto ayuda para que haya una gran cantidad documentación de calidad y también artículos que orientan cuando se presentan dudas.

Este proyecto ayudó a los integrantes a profundizar habilidades en Mongo y PostgreSQL, pero la introducción a creación de APIs y migración entre dichas base de datos es algo que aporta otra herramienta más a los integrantes del grupo.

Anexo

Consultas PostgreSQL

```
--1)
SELECT t.* FROM e01_cliente c JOIN e01_telefono t on c.nro_cliente
= t.nro_cliente WHERE c.nombre = 'Wanda' AND c.apellido = 'Baker'


--2)
SELECT * FROM e01_cliente  c WHERE c.nro_cliente IN (SELECT
f.nro_cliente FROM e01_factura f);


--3)
SELECT * FROM e01_cliente WHERE e01_cliente.nro_cliente NOT IN
(SELECT e01_factura.nro_cliente FROM e01_factura);


--4)
SELECT * FROM e01_producto p WHERE p.codigo_producto IN (SELECT
df.codigo_producto FROM e01_detalle_factura df);


--5)
SELECT c.*, t.codigo_area, t.nro_telefono FROM e01_cliente c LEFT
JOIN e01_telefono t ON c.nro_cliente = t.nro_cliente;
/*Si un cliente posee dos telefonos, aparecera en dos tuplas
distintas, una con cada teléfono*/


--6)
SELECT c.*, COUNT(f.nro_cliente) as cant_facturas FROM e01_cliente
c LEFT JOIN e01_factura f ON c.nro_cliente = f.nro_cliente GROUP
BY c.nro_cliente;


--7)
SELECT f.* FROM e01_cliente c JOIN e01_factura f on c.nro_cliente
= f.nro_cliente
WHERE c.nombre = 'Pandora' AND c.apellido = 'Tate'
```

```
--8)
SELECT f.* FROM e01_factura f WHERE f.nro_factura IN (SELECT
d.nro_factura FROM e01_detalle_factura d LEFT JOIN e01_producto p
ON  d.codigo_producto = p.codigo_producto WHERE p.marca = 'In
Faucibus Inc.' );


--9)
SELECT * FROM e01_telefono t INNER JOIN e01_cliente c ON
c.nro_cliente = t.nro_cliente;


--10)
SELECT c.nombre,c.apellido,f.total_con_iva FROM e01_cliente c JOIN
e01_factura f ON c.nro_cliente = f.nro_cliente
```

Vistas PostgreSQL

```
--1)
CREATE VIEW facturas_ordenadas_por_fecha AS
SELECT * FROM e01_factura ORDER BY fecha DESC;


--2)
CREATE VIEW productos_no_facturados AS SELECT * FROM e01_producto
p WHERE p.codigo_producto NOT IN (SELECT d.codigo_producto FROM
e01_detalle_factura d);
```

Consultas MongoDB

```
//Ejercicio 1
db.e01_cliente.aggregate([
      {
      $match: {
            nombre: 'Wanda',
            apellido: 'Baker'
      }
      },
      {
      $lookup: {
            from: "e01_telefono",
            localField: "nro_cliente",
```

```
            foreignField: "nro_cliente",
            as: "telefonos"
      }
      },
      {
      $unwind: "$telefonos"
      },
      {
      $replaceRoot: { newRoot: "$telefonos" }
      }
])


//Ejercicio 2
var facturaNumbers = db.e01_factura.distinct("nro_cliente");
db.e01_cliente.find({ nro_cliente: { $in: facturaNumbers } })


//Ejercicio 3
var facturaNumbers = db.e01_factura.distinct("nro_cliente");
db.e01_cliente.find({ nro_cliente: { $nin: facturaNumbers } })


//Ejercicio 4
var productoCodes =
db.e01_detalle_factura.distinct("codigo_producto");
db.e01_producto.find({ codigo_producto: { $in: productoCodes } })


//Ejercicio 5
db.e01_cliente.aggregate([
      {
      $lookup: {
            from: "e01_telefono",
            localField: "nro_cliente",
            foreignField: "nro_cliente",
            as: "telefonos"
      }
      },
      {
      $unwind: {
            path: "$telefonos",
            preserveNullAndEmptyArrays: true
      }
      },
      {
      $project: {
            _id: 0,
```

```
            nombre: 1,
            apellido: 1,
            codigo_area: "$telefonos.codigo_area",
            nro_telefono: "$telefonos.nro_telefono"
      }
      }
])


//Ejercicio 6
db.e01_cliente.aggregate([
      {
      $lookup: {
            from: "e01_factura",
            localField: "nro_cliente",
            foreignField: "nro_cliente",
            as: "facturas"
      }
      },
      {
      $unwind: {
            path: "$facturas",
            preserveNullAndEmptyArrays: true
      }
      },
      {
      $group: {
            _id: "$nro_cliente",
            nombre: { $first: "$nombre" },
            apellido: { $first: "$apellido" },
            cant_facturas: { $sum: 1 }
      }
      }
])


//Ejercio 7
db.e01_cliente.aggregate([
      {
      $match: {
            nombre: 'Pandora',
            apellido: 'Tate'
      }
      },
      {
      $lookup: {
            from: 'e01_factura',
            localField: 'nro_cliente',
```

```
            foreignField: 'nro_cliente',
            as: 'facturas'
      }
      },
      {
      $unwind: "$facturas"
      },
      {
      $replaceRoot: { newRoot: "$facturas" }
      }
])


//Ejercicio 8
db.e01_factura.find({nro_factura:{$in:db.e01_detalle_factura.disti
nct("nro_factura",{codigo_producto:{$in:db.e01_producto.distinct("
codigo_producto",{marca:"In Faucibus Inc."})}})}})

//Ejercicio 9
db.e01_cliente.aggregate([
      {
      $lookup: {
            from: "e01_telefono",
            localField: "nro_cliente",
            foreignField: "nro_cliente",
            as: "telefonos"
      }
      },
      {
      $unwind: "$telefonos"
      }
])


//Ejercicio 10
db.e01_factura.aggregate([
      {
      $lookup: {
            from: "e01_cliente",
            localField: "nro_cliente",
            foreignField: "nro_cliente",
            as: "cliente"
      }
      },
      {
      $unwind: "$cliente"
      },
      {
```

```
      $project: {
            _id: 0,
            nombre: "$cliente.nombre",
            apellido: "$cliente.apellido",
            total_con_iva: 1
      }
      }
])
```

Vistas MongoDB

```
//Ejercicio 1
db.createView(
      "facturas_ordenadas_por_fecha",
      "e01_factura",
      [
      { $sort: { fecha: -1 } } // Sort by 'fecha' in descending
order
      ]
 );


 //Ejercicio 2
var productCodesInDetailFactura =
db.e01_detalle_factura.distinct("codigo_producto");
db.createView(
   "productos_no_facturados",
   "e01_producto",
   [
     { $match: { codigo_producto: { $nin:
productCodesInDetailFactura } } }
   ]
);
```