Trabajo Final 2022

DISEÑO DE BASES DE DATOS - MG IS 2021

En este documento se presenta el trabajo final del curso, el cual dispondrá de dos posibles modalidades a elección:

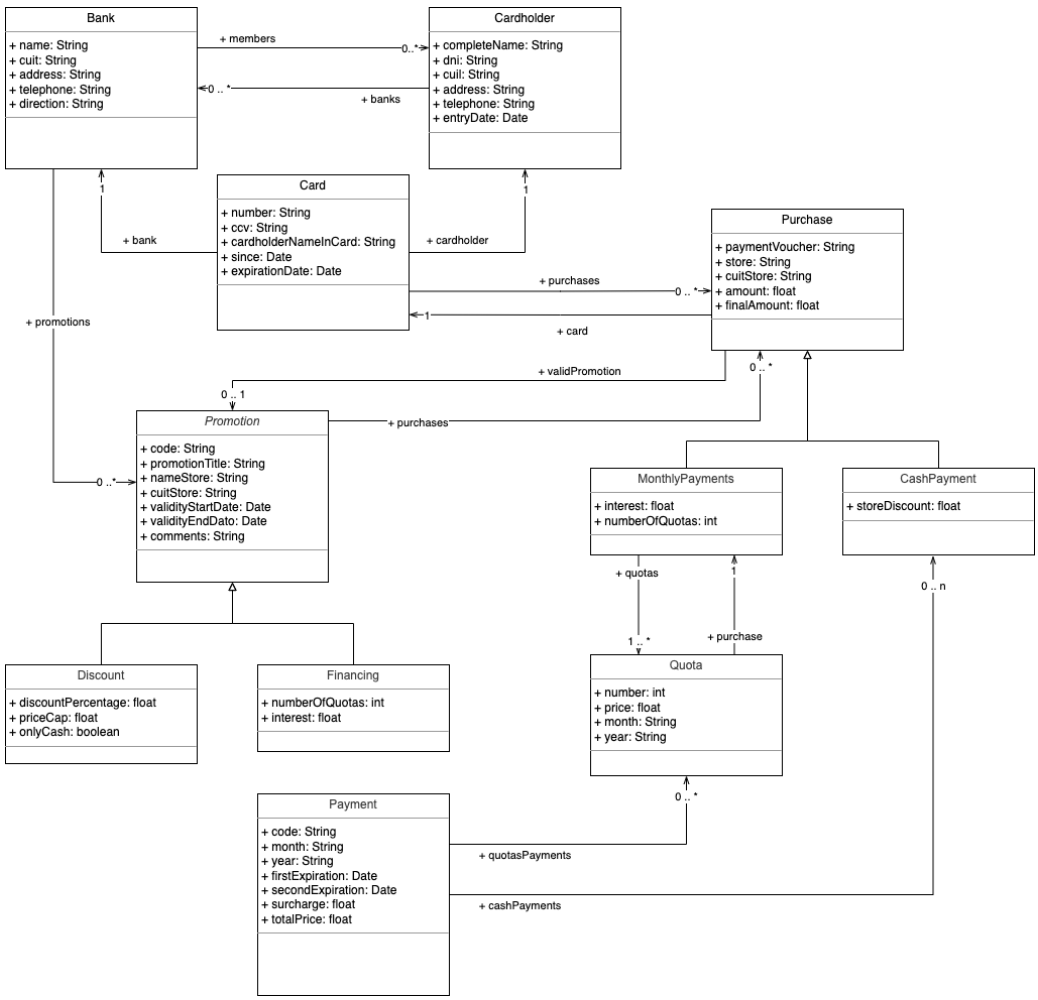
- Un proyecto de desarrollo: consiste en el mapeo de un proyecto, similar al visto durante la cursada, en una base de datos de tipo relacional, como MySQL, y a una de tipo no relacional, como MongoDB.
- Un proyecto de investigación: consiste en la definición de al menos dos BDs, que incluyan tipos relacionales y no relacionales y, con la carga de un conjunto grande de datos, hacer una prueba de performance sobre distintas consultas para finalmente plasmar en un documento las conclusiones.

El trabajo puede realizarse de manera individual o en grupos de dos personas. Informar sobre el tipo de trabajo y conformación del grupo a: federicodicla@gmail.com
Ante cualquier consulta que les surja durante el desarrollo del trabajo estaremos en contacto mediante el mismo mail.

La fecha limite para la entrega del trabajo es el 30/7/2023

Proyecto de desarrollo

Esta opción con consiste en un proyecto de desarrollo, el cual deberá persistirse en dos tipos de bases de datos, una de tipo relacional (como puede ser MySQL, visto durante la cursada) y a una base de datos de tipo no relacional (como Mongo, visto durante la cursada). Ambas persistencias no deben coexistir, sino que se trata de dos entregas de un mismo proyecto.



El proyecto consiste en un sistema de registro de pagos de diferentes tarjetas. La siguiente imagen muestra el diagrama de clase de la aplicación:

El sistema posee el listado de titulares de tarjetas (**CardHolder**) con sus respectivos datos, y de los diferentes bancos (**Bank**) sobre los cuales estos son clientes. Las diferentes tarjetas (**Card**) están asociados a un único titular y corresponde a un único banco, registrándose también los datos de la misma.

El sistema modela las diferentes compras (**Purchase**) con sus respectivos datos, incluyendo los de la tienda donde fue realizada, el valor inicial de la compra y el final (con los descuentos o intereses aplicados). Estas compras pueden ser de dos tipos: en un solo pago (**CashPurchase**), sobre la cual puede existir, ademas de los datos básicos de la compra, un porcentaje de descuento sobre el valor, ofrecido por el local; o la compra puede ser en cuotas (**MonthlyPayments**), se registra, si existe, un porcentaje adicional sobre el precio total de la compra, y un numero de cuotas.

Cada banco ofrece diferentes promociones (**Promotion**) sobre distintos locales asociados, identificados por su cuit, que tienen validez en un único rango de fechas. Cuando se realiza una compra en una tienda que dispone de promociones del banco asociado a la tarjeta con la que se realiza el pago, se aplican las promociones correspondientes. Estas promociones pueden ser de dos tipos: un descuento (**Discount**), el cual es un porcentaje que se resta al importe de las compras (este no tiene que ver con el aplicado por el local, y se suma a este), y puede o no tener un valor tope; también estas promociones pueden ser para compras en cuotas (**Financing**), especificando un numero de cuotas y un interés especifico, provisto por el banco. En caso de existir más de una posible descuento para una misma compra, estos pueden acumularse, exceptuando el caso donde existen diferentes opciones en cuotas, solo una aplicable, y cuando el descuento especifique que es solo al contado.

Por ultimo, los pagos (**Payments**) poseen un código interno, corresponden a cierto mes y año y tienen dos fechas de vencimiento, teniendo para cada vencimiento un porcentaje de incremento del valor del pago. Dicho pago corresponde a las cuotas (**Quota**) correspondientes a compras hechas con anterioridad y a la totalidad de pagos hechos al contado en el ultimo mes.

Entrega

Se deberá entregar el proyecto en dos resoluciones: una con persistencia de los objetos a una base de datos de tipo relacional (recomendamos MySQL), y otra a una base de datos de tipo no relacional (como MongoDB, orientada a documentos). Para ello se deberá implementar la persistencia de los objetos y sus relaciones, junto a la lógica correspondiente a la aplicación y las consultas para resolver los siguientes puntos:

- Agregar una nueva promoción de tipo descuento a un banco dado
- Editar las fecha de vencimiento de un pago con cierto código.
- Generar el total de pago de un mes dado, informando las compras correspondientes
- Obtener el listado de tarjetas que vencen en los siguientes 30 días.
- Obtener la información de una compra, incluyendo el listado de cuotas si esta posee.
- Eliminar una promoción a traves de su código (tener en cuenta que esta puede haber sido aplicada alguna compra)
- Obtener el precio total a pagar de una compra en cuotas (tener en cuenta que pueden existir promociones aplicadas)
- Obtener el listado de las promociones disponibles de un local entre dos fechas
- Obtener los titulares de las 10 tarjetas con más compras.
- Obtener la promoción mas utilizada en las compras registradas
- Obtener el nombre y cuit del local, que mas facturo en cierto mes
- Obtener el banco que registre la mayor sumatoria de los importes en pagos con su tarjeta.

Note que para el correcto funcionamiento de estos puntos, será necesario la realización de otras tareas, como los correspondientes creaciones y persistencia de las entidades relacionadas a las diferentes consultas.

Para la posterior prueba y corrección, estas funcionalidades pueden estar expuestas a través de endpoints de una API o pueden entregarse diferentes tests para su verificación.

El proyecto deberá estar acompañado de un breve informe en el que se describa el proyecto realizado, así como también las decisiones tomadas en el mapeo, sobre todo con respecto a los siguientes puntos:

- Utilización de relaciones bidireccionales
- Carga de objetos bajo demanda (o lazy)
- Operaciones en cascada
- Embeber objetos o utilización de referencias
- Uso de transacciones

Además el informe debe contener una descripcion de lo necesaria para levantar la aplicación (es decir, configuración de la base de datos, usuarios, etc.) y para su correspondiente prueba (descripción de los tests o endpoints de la API)

La entrega se realiza mediante un repositorio de GitHub al cual deben agregar como colaborador al usuario: fedediclaudio

Tecnologias

La tecnología es a elección de cada grupo, sin embargo, les proporcionamos un esquema ya inicializado de una aplicación en Java y utilizando las tecnologías que vimos como muestra durante la cursada.

El repositorio con el proyecto inicial es: https://github.com/fedediclaudio/dbdtp23_cardpurchases

Para la ejecución del proyecto dado, serán necesarias las siguientes tecnologías:

- Java 11
- Maven, como gestor de dependencias
- MySQL v5.7 en adelante, como base de datos relacional
- MongoDB, como base de datos no relacional

Se recomienda utilizar IntelliJ como IDE, aunque puede usarse el de su preferencia. La utilización de contenedores Docker es opcional.

Si bien ya cuenta con una configuración de conexión con las bases de datos (a través de las propiedades de la aplicación), esta les servirá de guía para generar su configuración.

Las dependencias necesarias, tanto para el mapeo en MySQL como en MongoDB, se encuentran ya cargadas dentro del archivo de dependencias “POM.xml”, pero se encuentran comentadas. Según cuál va a utilizarse, descomente las correspondientes.

Proyecto de Investigación

Como alternativa al proyecto de desarrollo, más orientado a aquellos que no estén cerca del código, les proponemos un trabajo de investigación que consiste en definir un conjunto de bases de datos y comparar la performance de estas en diferentes situaciones.

La prueba se debe realizar sobre al menos dos bases de datos, que quedan a elección del grupo, donde debe haber al menos una de tipo relacional (como MySQL, Postgres, etc.) y una de tipo no relacional (como Mongo, Elastic, etc.).

Sobre las bases de datos elegidas se debe definir un conjunto de entidades, al menos dos, que dispongan de una relación (uno a muchos, por ejemplo). Sobre estas entidades se realizarán las respectivas consultas de prueba. Estas consultas deben ser de diferentes tipos para abarcar distintos casos de uso de la base de datos, por ejemplo, debe haber inserciones, eliminaciones, actualizaciónes de datos y diferentes tipos de consultas, incluyendo innerjoins. Puede también hacerse pruebas con indices.

Para realizar estas consultas, la base de datos debe contar con diferentes cantidades significativas de datos. Proponemos que carguen tres diferentes cantidades de datos, con el fin de comprobar si al crecer las cantidades de datos se ve afectado el rendimiento de la base de datos. Por ejemplo, las pruebas pueden realizarse con 10000 (diez mil), 1000000 (un millón) y 100000000 (cien millones) de registros.

Por ultimo, se deben definir los distintos parámetros a evaluar sobre las diferentes consultas a realizar. Esto podría ser, por ejemplo, referidos al tiempo de las consultas (tiempo de respuesta, tiempo de carga de las información, etc.) o al espacio (espacio de almacenamiento ocupado, espacio de memoria, etc.).

La entrega consistirá de los recursos que se hayan utilizado para las diferentes pruebas, acompañada por un documento donde se describa, por un lado, las pruebas realizadas, incluyendo las decisiones tomadas con respecto a los puntos anteriores (bases de datos, entidades, consultas, cantidades de datos y parámetros definidos). Por otro lado, el informe debe mostrar los resultados de las diferentes pruebas, puede ayudarse con gráficos, y una conclusión obtenida sobre dichos resultados.

El informe debe enviarse a la siguiente dirección de correo: federicodicla@gmail.com