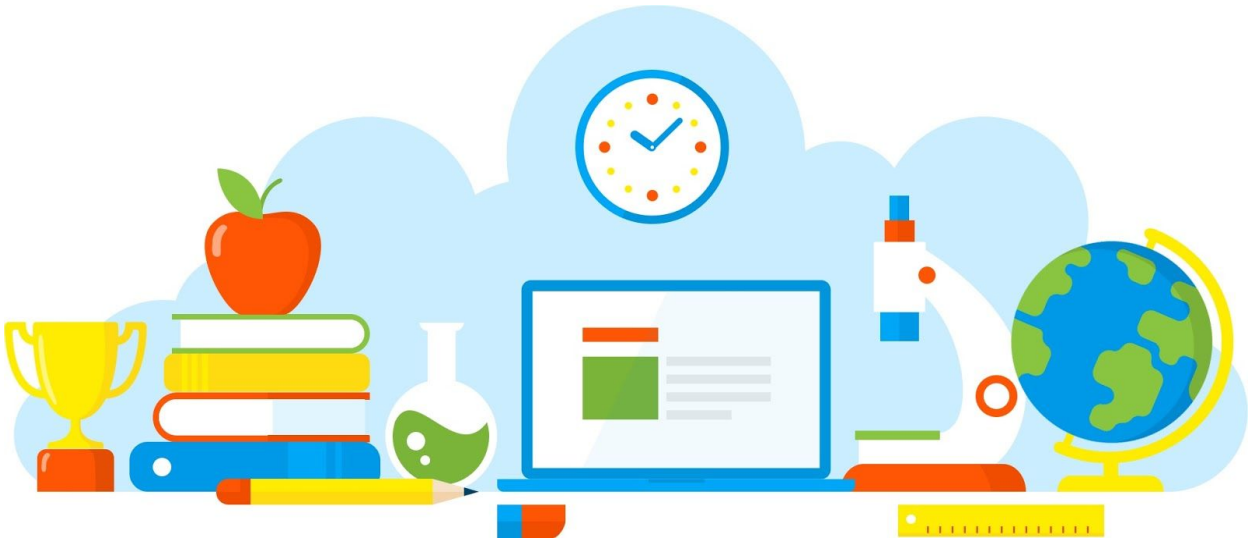

Rapport Partie 3

Application “Gestion Catalogue”

BOUFFARD-VERCELLI Florian JAROUDI Safaa

28/02/2019

Introduction

Il nous a été demandé dans cette dernière partie de proposer une application Java pouvant effectuer une gestion de produits provenant de plusieurs catalogues.

Nous reprenons donc le code de la partie 2 afin de continuer celle-ci.

Fonctionnalités

Les fonctionnalités de partie 2 sont également présentes dans celle-ci.

L'utilisateur a maintenant la possibilité de créer, supprimer et de sélectionner un catalogue afin de pouvoir classer ses produits.

Nous avons gardé les PreparedStatement pour les requêtes SQL mais nous avons également choisi de fournir une seconde possibilité pour effectuer les actions avec des procédures et des fonctions.

Les scripts sont disponibles dans le dossier dans un fichier SQL intitulé **pfscript_exemple.sql**

Néanmoins, nous les avons pas installées dans le programme par manque de temps.

Script SQL

Afin de pouvoir utiliser l'application avec JDBC, il est nécessaire de créer la table Produit et Catalogue avec le code ci-dessous.

```
DROP TABLE Catalogue CASCADE CONSTRAINTS;

CREATE TABLE Catalogue(
     idCatalogue INT,
     nomCatalogue VARCHAR(15),
     CONSTRAINT pk_catalogue PRIMARY KEY (idCatalogue),
     CONSTRAINT ck_nomCatalogue UNIQUE(nomCatalogue),
     CONSTRAINT nn_nomCatalogue CHECK (nomCatalogue IS NOT NULL)
);
```

```
DROP TABLE Produit CASCADE CONSTRAINTS;

CREATE TABLE Produit(
       idProduit CHAR(5),
       nomProduit VARCHAR(15),
       prixHT NUMBER(*,2),
       quantite NUMBER(5),
       catalogue INT,
       CONSTRAINT nn_nomProduit CHECK (nomProduit IS NOT NULL),
       CONSTRAINT nn_prixHT CHECK (prixHT IS NOT NULL),
       CONSTRAINT nn_quantite CHECK (quantite IS NOT NULL),
       CONSTRAINT pk_produit PRIMARY KEY (idProduit),
       CONSTRAINT fk_produit FOREIGN KEY (catalogue) REFERENCES
     catalogue (idCatalogue) ON DELETE CASCADE
);
```

→ Nous utilisons un séquenceur et un trigger lors de l'insertion d'un nouveau produit et d'un catalogue pour leur identifiant fictif.

```
CREATE SEQUENCE product_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE idCatalogue_seq START WITH 1 INCREMENT BY 1;
```

```
CREATE OR REPLACE TRIGGER insert_product_trigger
BEFORE INSERT ON Produit
FOR EACH ROW

BEGIN
  SELECT product_seq.NEXTVAL
  INTO   :new.idProduit
  FROM   dual;
END;
```

```
CREATE OR REPLACE TRIGGER insert_catalogue_trigger
BEFORE INSERT ON Catalogue
FOR EACH ROW

BEGIN
  SELECT idCatalogue_seq.NEXTVAL
  INTO   :new.idCatalogue
  FROM   dual;
END;
```

Diagrammes

Diagramme de classes de conception

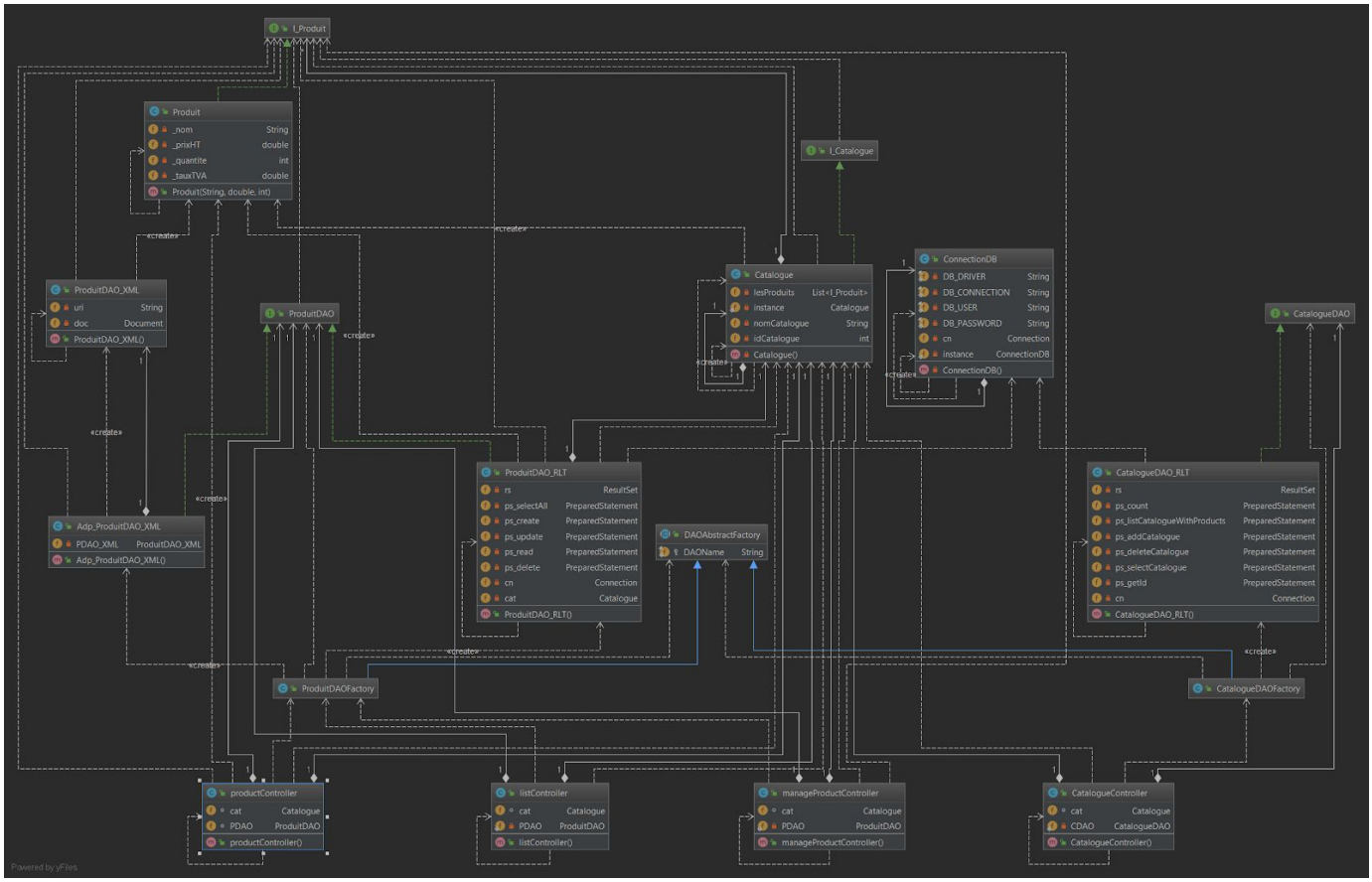

Diagramme de séquence (Initialisation de l'application)

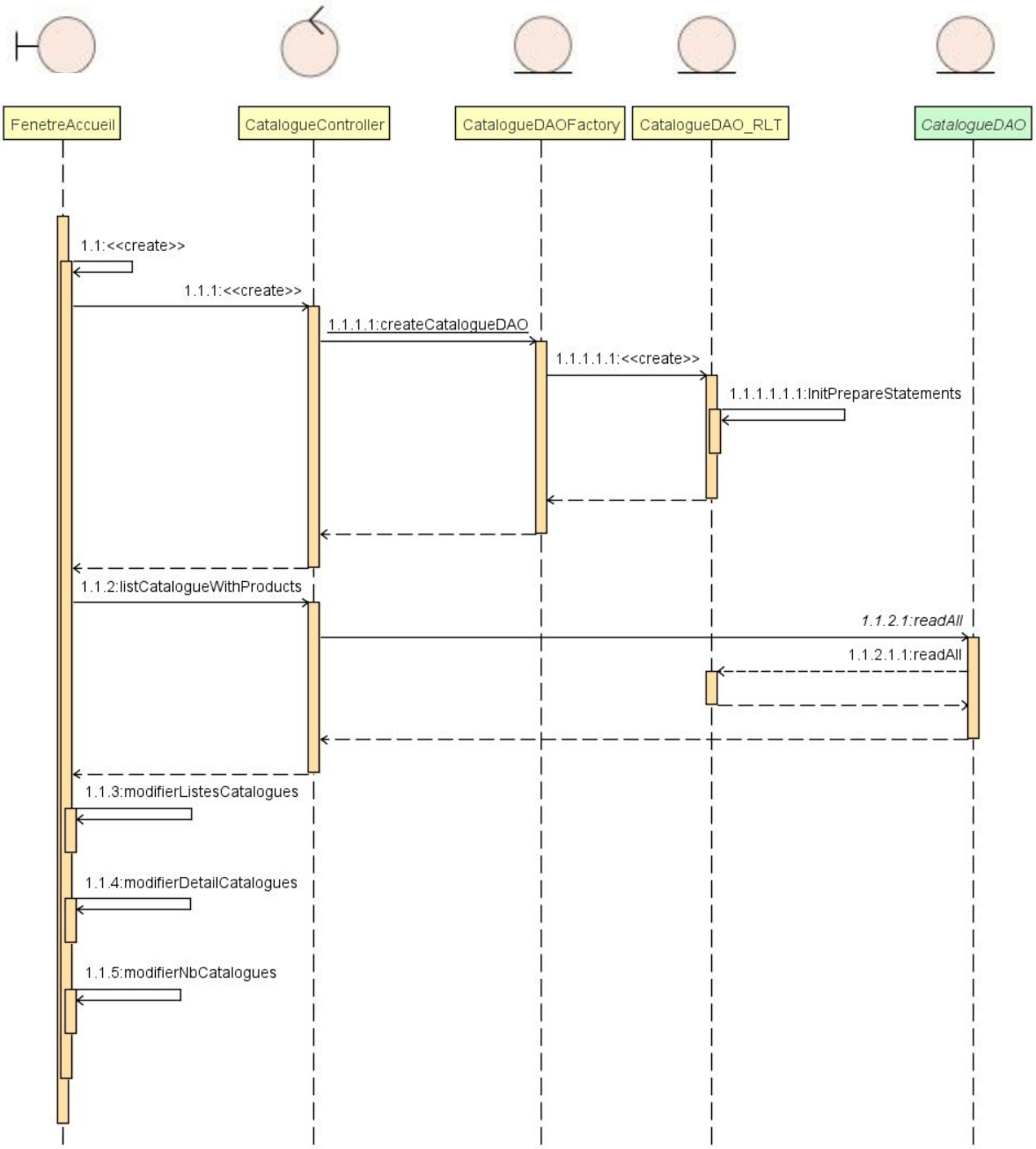

Diagramme de séquence (Sélectionner un catalogue)