

Projet E-shop

Projet de remise à niveau MBDS

Spécification de l’analyse de conception

MOMAS Lisa

TAMANINI Nicolas

Table des matières

[1 Description du sujet 3](#_Toc50974307)

[2 Description textuelle des requêtes de mise à jour 3](#_Toc50974308)

[3 Description textuelle des requêtes de suppression 3](#_Toc50974309)

[4 Description textuelle des requêtes de consultation 3](#_Toc50974310)

[5 Dictionnaire de données MERISE 3](#_Toc50974311)

[6 Description textuelle des associations 3](#_Toc50974312)

[7 Définition du Modèle Entité-Association MERISE 3](#_Toc50974313)

[8 Définition du Modèle Logique de Données 4](#_Toc50974314)

[9 Spécification des traitements avec des packages PLSQL 4](#_Toc50974315)

[10 Spécification des triggers 4](#_Toc50974316)

[11 Maquettes d’écran 4](#_Toc50974317)

# Description du sujet

Pour le projet de remise à niveau, nous avons décidé de choisir le sujet de la gestion d’une boutique de vêtements en ligne. L’application sera orientée sur la partie « arrière-boutique » (back office) d’une boutique de vêtement et doit intégrer les fonctionnalités suivantes qui permettront aux clients du site de commander des vêtements et de se faire livrer la commande :

* Visualisation d’une liste de vêtements et filtrage par catégorie, taille, couleur, etc.
* Création, modification, suppression de vêtements
* Gestion des stocks de vêtements
* Gestion des commandes
* Gestion des comptes utilisateur

# Description textuelle des requêtes de mise à jour

REQUETES DE MISE A JOUR

-- 2 requêtes impliquant 1 table :

-- Réduction de 10% sur tous les vêtements

UPDATE declinaison\_vetement SET prix\_declinaison = prix\_declinaison \* 0.90;

-- Réduction de 20% sur tous les vêtements dont le prix est supérieur à 80€

UPDATE declinaison\_vetement SET prix\_declinaison = prix\_declinaison \* 0.80 WHERE prix\_declinaison >= 80;

-- 2 requêtes impliquant 2 tables :

-- Reduction de 20% sur tous les vetements dont la taille est égale à XXL

UPDATE declinaison\_vetement dv JOIN taille t ON dv.id\_taille = t.id\_taille SET dv.prix\_declinaison = v.prix\_declinaison \* 0.80 WHERE t.nom\_taille = 'XXL' ; -- FONCTIONNE PAS

-- Augmentation de 5% sur les vetements rouges

UPDATE declinaison\_vetement dv JOIN couleur c ON dv.id\_couleur= c.id\_couleur SET dv.prix\_declinaison = dv.prix\_declinaison \* 1.05 WHERE dv.nom\_couleur = 'Rouge' ; -- FONCTIONNE PAS

-- 2 requêtes impliquant plus de 2 tables :

-- Augmentation de 5% sur les vetements rouges et de taille

UPDATE declinaison\_vetement dv JOIN couleur c ON dv.id\_couleur= c.id\_couleur JOIN taille t ON dv.id\_taille = t.id\_taille SET dv.prix\_declinaison = v.prix\_declinaison \* 1.05 WHERE c.nom\_couleur = 'Rouge' AND t.nom\_taille = 'S' ; -- FONCTIONNE PAS

-- Réduction de 5€ pour les vêtements rouges de taille M avec un prix supérieur à 9€--

UPDATE declinaison\_vetement dv JOIN couleur c ON dv.id\_couleur= c.id\_couleur JOIN taille t ON dv.id\_taille = t.id\_taille SET dv.prix\_declinaison = dv.prix\_declinaison - 5 WHERE dv.prix\_declinaison > 9 AND c.nom\_couleur = 'Rouge' AND t.nom\_taille = 'M' ; -- FONCTIONNE PAS

# Description textuelle des requêtes de suppression

REQUETES DE SUPPRESSION

-- 2 requêtes impliquant 1 table :

-- Supprimer une ligne de la table declinaison\_vetement (suppression d'un vêtement)

DELETE FROM declinaison\_vetement WHERE id\_declinaison = 1;

-- Supprimer tous les vêtements dont le prix est supérieur à 50€

DELETE FROM declinaison\_vetement WHERE prix\_declinaison > 50;

-- 2 requêtes impliquant 2 tables :

-- Supprimer toutes les commandes d'un utilisateur

DELETE (SELECT \* FROM commande JOIN utilisateur ON utilisateur.id\_utilisateur = commande.id\_utilisateur WHERE utilisateur.id\_utilisateur = 5);

-- Supprimer tous les vêtements dont la taille est 'M'

DELETE (SELECT \* FROM declinaison\_vetement JOIN taille ON declinaison\_vetement.id\_taille = taille.id\_taille WHERE taille.nom\_taille = 'M');

-- 2 requêtes impliquant plus de 2 tables :

-- Supprimer tous les vêtements dont la catégorie est "T-shirt"

DELETE (SELECT \* FROM declinaison\_vetement JOIN vetement ON declinaison\_vetement.id\_vetement = vetement.id\_vetement JOIN categorie ON vetement.id\_categorie = categorie.id\_categorie WHERE categorie.nom\_categorie = 'T-shirt');

--Supprimer les commandes contenant un vêtement dans la catégorie 'Pantalon'

DELETE (SELECT \* FROM commande JOIN est\_commande ON commande.id\_commande = est\_commande.id\_commande JOIN declinaison\_vetement ON est\_commande.id\_declinaison = declinaison\_vetement.id\_declinaison JOIN vetement ON declinaison\_vetement.id\_vetement = vetement.id\_vetement JOIN categorie ON vetement.id\_categorie = categorie.id\_categorie WHERE categorie.nom\_categorie = 'Pantalon');

# Description textuelle des requêtes de consultation

DESCRIPTION TEXTUELLES DES REQUETES DE CONSULTATION

-- 5 requêtes impliquant 1 table dont 1 avec un group By et une avec un Order By :

-- Trier les vêtements par ordre croissant du prix

SELECT id\_declinaison, prix\_declinaison FROM declinaison\_vetement ORDER BY prix\_declinaison;

-- Regrouper les vetements de quantité disponible par couleur

SELECT id\_couleur, COUNT(\*) FROM declinaison\_vetement GROUP BY id\_couleur HAVING quantite\_declinaison > 0; -- FONCTIONNE PAS

-- Filtrer les vêtements par gamme de prix. Ici c'est entre 20 et 40€.

SELECT \* FROM declinaison\_vetement WHERE prix\_declinaison BETWEEN 20 AND 40;

-- Lister les vêtements dont le prix est inférieur à 30€.

SELECT \* FROM declinaison\_vetement WHERE prix\_declinaison < 31;

-- Lister les vêtements dont le prix est égal à 30€.

SELECT \* FROM declinaison\_vetement WHERE prix\_declinaison = 30;

-- 5 requêtes impliquant 2 tables avec jointures internes dont 1 externe + 1 group by + 1 tri :

-- Donner le prix des commandes regroupées par utilisateur (GROUP BY)

SELECT utilisateur.id\_utilisateur, SUM(commande.prix\_commande) FROM utilisateur JOIN commande ON utilisateur.id\_utilisateur = commande.id\_utilisateur GROUP BY utilisateur.id\_utilisateur;

-- Donner la liste des utilisateurs avec les commandes qui leurs sont associées. Si un utilisateur n'a jamais commandé, on affiche quand même son nom

SELECT prenom\_utilisateur, nom\_utilisateur, id\_commande FROM utilisateur LEFT OUTER JOIN commande ON utilisateur.id\_utilisateur = commande.id\_utilisateur;

-- Donner la liste des déclinaisons de vêtements de taille 'M'

SELECT id\_declinaison, nom\_taille FROM declinaison\_vetement JOIN taille ON declinaison\_vetement.id\_taille = taille.id\_taille WHERE taille.nom\_taille = 'M';

-- Donner la liste des déclinaisons de vêtements de couleur rouge

SELECT id\_declinaison, nom\_couleur FROM declinaison\_vetement JOIN couleur ON declinaison\_vetement.id\_couleur = couleur.id\_couleur WHERE couleur.nom\_couleur = 'Rouge';

-- Donner la liste des vêtements de catégorie 'Pantalon'

SELECT nom\_vetement, nom\_categorie FROM vetement JOIN categorie ON vetement.id\_categorie = categorie.id\_categorie WHERE categorie.nom\_categorie = 'Pantalon';

--Trier les vêtements par catégorie

-- 5 requêtes impliquant plus de 2 tables avec jointures internes dont 1 externe + 1 group by + 1 tri :

-- Donner la liste des utilisateurs et de la ou des commande(s) contenant ayant vêtement de taille S que ces utilisateurs ont passé.

SELECT prenom\_utilisateur, nom\_utilisateur, commande.id\_commande FROM utilisateur JOIN commande ON utilisateur.id\_utilisateur = commande.id\_utilisateur JOIN est\_commande ON commande.id\_commande = est\_commande.id\_commande JOIN declinaison\_vetement ON est\_commande.id\_declinaison = declinaison\_vetement.id\_declinaison JOIN taille ON declinaison\_vetement.id\_taille = taille.id\_taille WHERE taille.nom\_taille = 'S';

-- Donner la liste des déclinaisons de vêtements dont la catégorie est 'T-Shirt' triés par prix décroissants

SELECT id\_declinaison, nom\_vetement, prix\_declinaison, quantite\_declinaison FROM declinaison\_vetement JOIN vetement ON declinaison\_vetement.id\_vetement = vetement.id\_vetement JOIN categorie ON vetement.id\_categorie = categorie.id\_categorie WHERE nom\_categorie='T-Shirt' ORDER BY prix\_declinaison DESC;

-- Donner la liste des déclinaisons de vêtements dont la taille est S et la couleur est noir

SELECT id\_declinaison, nom\_taille, nom\_couleur FROM declinaison\_vetement JOIN taille ON declinaison\_vetement.id\_taille = taille.id\_taille JOIN couleur ON declinaison\_vetement.id\_couleur = couleur.id\_couleur WHERE nom\_taille = 'S' AND nom\_couleur='Noir';

-- Donner le nom et la catégorie des vêtements étant associés à des déclinaisons. Si un vêtement n'est associé à aucune déclinaison, le nom et la catégorie de ce vêtement sont quand même affichés.

SELECT nom\_categorie, nom\_vetement, id\_declinaison FROM vetement LEFT OUTER JOIN categorie ON vetement.id\_categorie = categorie.id\_categorie LEFT OUTER JOIN declinaison\_vetement ON vetement.id\_vetement = declinaison\_vetement.id\_vetement;

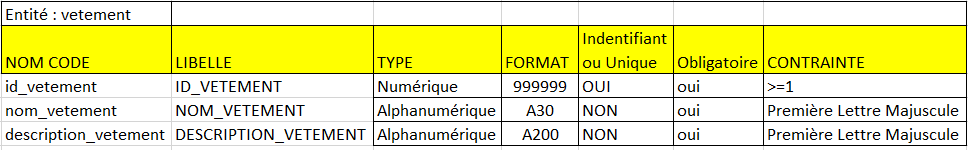
-- Donner la valeur marchande du stock de vêtements par catégorie trié par valeur marchande décroissante

SELECT categorie.nom\_categorie, SUM(declinaison\_vetement.prix\_declinaison)"Valeur marchande du stock par catégorie" FROM vetement JOIN categorie ON vetement.id\_categorie = categorie.id\_categorie JOIN declinaison\_vetement ON vetement.id\_vetement = declinaison\_vetement.id\_vetement GROUP BY categorie.nom\_categorie ORDER BY SUM(declinaison\_vetement.prix\_declinaison) DESC;

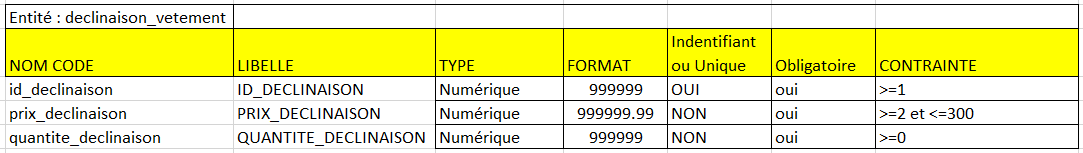
# Dictionnaire de données MERISE

Vous trouverez ci-joint le dictionnaire de donnée MERISE de chaque entité.

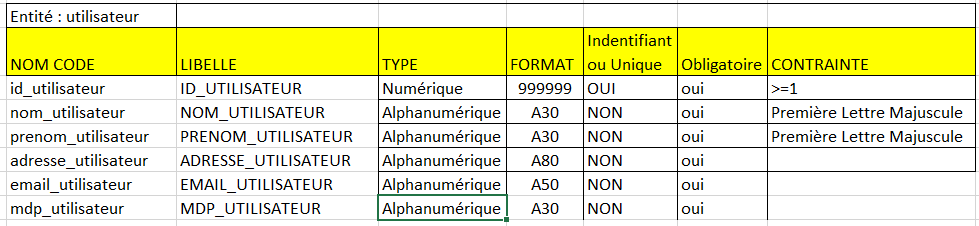
Entité VETEMENT



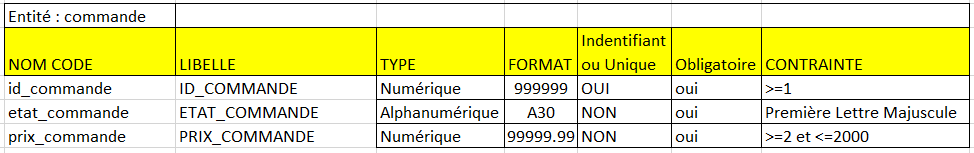
Entité DECLINAISON\_VETEMENT



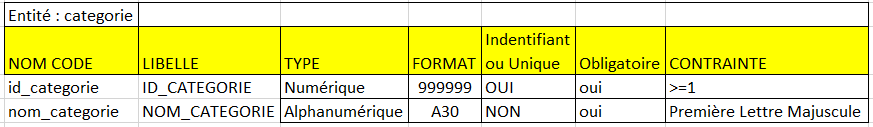
Entité UTILISATEUR



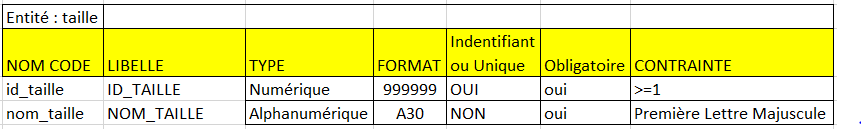
Entité COMMANDE



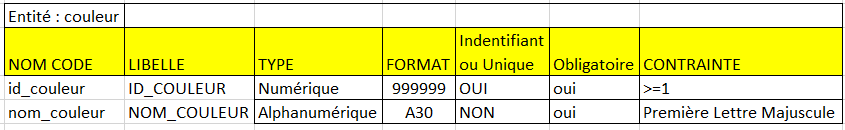
Entité CATEGORIE



Entité TAILLE



Entité COULEUR



# Description textuelle des associations

Nous avons 6 tables dans notre SGBD e-shop.

* Nous avons la table UTILISATEUR qui contient les données nécessaires pour se créer un compte utilisateur (id, nom, prénom, adresse, email). Cet utilisateur choisi ses vêtements en fonction des déclinaisons disponibles, nous avons donc une liaison entre UTILISATEUR et DECLINAISON\_VETEMENT. Il peut n’y avoir aucun ou plusieurs utilisateurs qui choisissent aucune ou plusieurs déclinaisons. 0 à n déclinaisons peuvent être choisi par 0 à n utilisateurs.

Après avoir choisis ses vêtements, l’utilisateur peut passer 0 ou plusieurs commandes comprenant ses déclinaisons de vêtement. Aucune ou plusieurs déclinaisons de vêtements de la table DECLINAISON\_VETEMENT peuvent donc être commandés par 0 ou plusieurs COMMANDE.

* Nous avons la table VETEMENT qui contient les attributs id, nom, description. Cette table peut comprendre un « type » de vêtement. Cela peut être un t-shirt avec une écriture propre dessus. Ce vêtement fait partie d’une CATEGORIE, dans notre cas, le t-shirt avec écriture ferait partie de la catégorie t-shirt. Nous avons donc un vêtement qui appartient à une catégorie.

1 vêtement appartient donc à 1 catégorie.

Cependant, une catégorie peut n’avoir aucun ou plusieurs vêtements dans son sein. Nous avons donc 0 ou n vêtements dans une catégorie.

* Nous avons la table DECLINAISON\_VETEMENT. Elle comprend les données des différentes déclinaisons d’un vêtement. En effet, nous pouvons avoir un vêtement de différentes tailles ou couleurs, c’est pour cela que cette table est liée aux tables COULEUR et TAILLE. En fonction de sa taille ou couleur, le prix ou la quantité d’un vêtement peut changer d’où un attribut prix et quantité. Cependant, ce vêtement à un nom et une description propre qu’importe sa déclinaison et fait partie d’une catégorie. C’est pour cela que la table est également liée à la table vêtement.

1 déclinaison de vêtement peut avoir que 1 taille assignée ainsi que 1 couleur. Cette déclinaison appartient qu’à 1 « type » de vêtement mais ce type de vêtement peut avoir n déclinaison.

# Définition du Modèle Entité-Association MERISE

0 à n UTILISATEUR peut passer 0 à n COMMANDE comprenant des déclinaisons de vêtement. 1 COMMANDE peut être passée par 1 UTILISATEUR.

0 à n DECLINAISON\_VETEMENT peuvent avoir 0 à n COMMANDE.

0 à n COMMANDES peuvent être passées par 0 à n UTILISATEUR et avoir 1 à n DECLINAISON\_VETEMENT.

Un UTILISATEUR peut choisir 0 à n DECLINAISON\_VETEMENT, et 0 à n DECLINAISON\_VETEMENT peuvent être choisit par 0 à n UTILISATEUR.

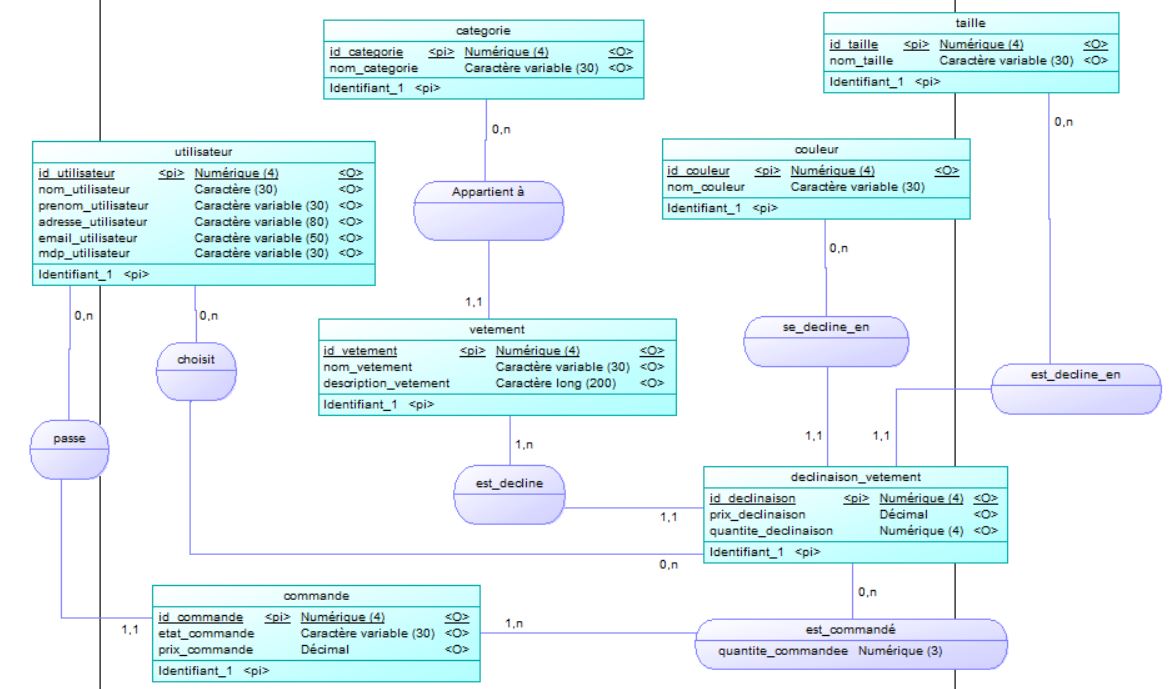
1 VETEMENT appartient à 1 CATEGORIE. Cependant, 1 CATEGORIE peut avoir 0 à N VETEMENT dans son sein.

1 DECLINAISON\_VETEMENT peut avoir que 1 TAILLE assignée ainsi que 1 COULEUR. 1 TAILLE ou COULEUR peut avoir 0 à n DECLINAISON\_VETEMENT.

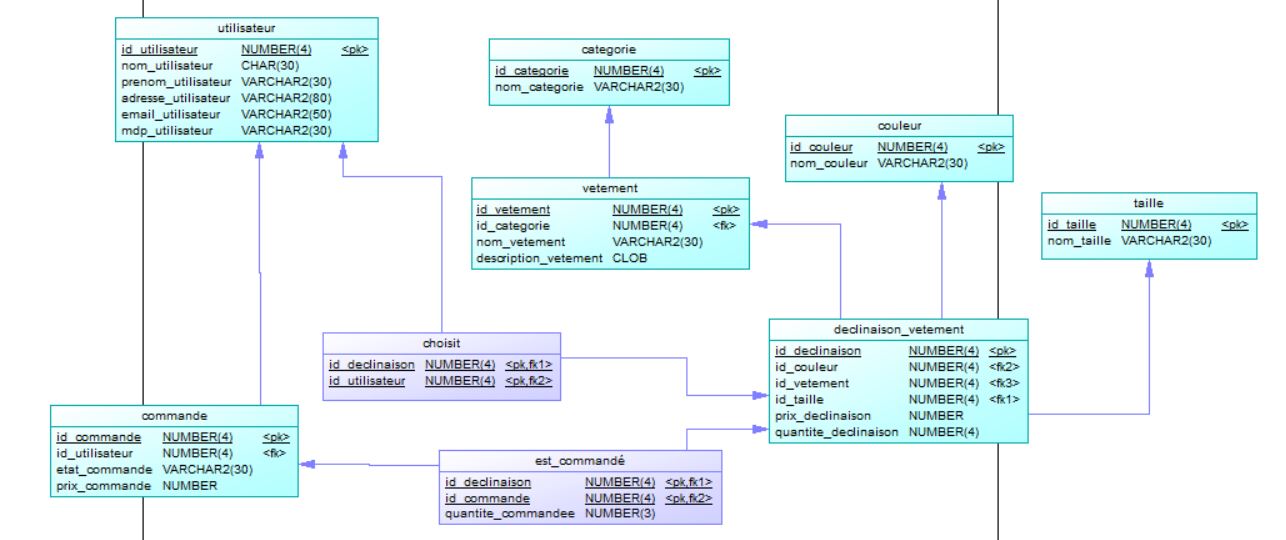
1 VETEMENT peut avoir 0 à n DECLINAISON\_VETEMENT ET 1 DECLINAISON\_VETEMENT peut avoir 1 à n VETEMENT.

# Définition du Modèle Logique de Données

Représentation du MCD réalisé sur PowerAMC



Représentation du MLD réalisé sur PowerAMC



# Spécification des traitements avec des packages PLSQL

Nous avons choisi de faire les fonctions et procédures PLSQL sur les tables UTILISATEUR (table A) et COMMANDE (table B).

Nous y avons réalisé des fonctions insérées permettant d’insérer dans la table une ligne avec tous ses attributs.

Les fonctions supprimer permet de supprimer une ligne du tableau correspondant à l’ID mis en paramètre de la fonction.

Pour la table UTILISATEUR, les fonctions modifiant les informations permettent de modifier un mot de passe (valeur nouvelle étant celle passée en paramètre) en fonction de l’id de l’utilisateur passé également en paramètre. Une autre permet de modifier les nom et prénoms par de nouvelles valeurs définis en paramètre en fonction de l’id rentrée en paramètre.

Pour la table COMMANDE, nous avons une fonction modifiant le prix d’une commande ainsi qu’une autre modifiant l’état de la commande (ex : nous passons d’un statut de commande « en cours » à un statue « expédié »).

Les fonctions lister permet d’afficher les informations du tableau de la forme suivante :

userId - userLastName| - userFirstName - userAddress - userEmail - userPassword. Elles listent les informations des utilisateurs.

# Spécification des triggers

Nous avons réalisé deux triggers.

Le premier se nomme USERDELETE\_TRIGGER et empêche la suppression des données de l’utilisateur lorsque celui-ci à déjà réalisé une commande. Cela permet de garder un suivi sur toutes les commandes réalisées au sein du e-shop.

Le deuxième trigger se nomme INSERTUSERTRIGGER et permet d’afficher à chaque insertion d’un utilisateur l’id, le nom et prénom du nouvel utilisateur.

Ces deux triggers sont spécifiés après un delete ou un insert.

# Maquettes d’écran

Nous avons fait des maquettes de notre site web sur AdobeXD. Notre logiciel devait idéalement remplir ces fonctionnalités et interfaces :

* Un écran de connexion

Cette page permet à l’utilisateur de se connecter.

L’adresse mail et le mot de passe de l’utilisateur doivent être intégrés dans la base de données Oracle. Dans le cas contraire :

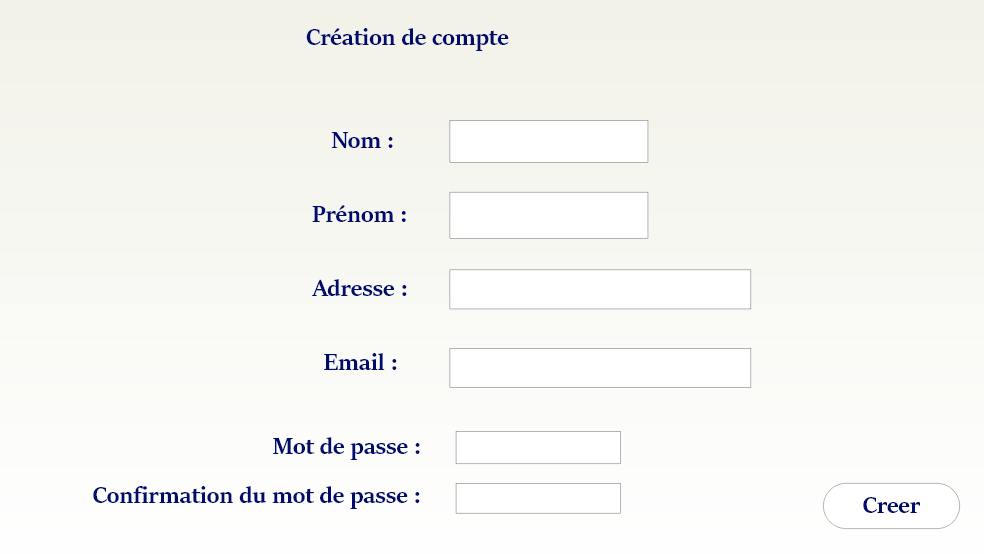
- L’utilisateur doit se créer de nouveaux identifiants en cliquant sur le lien hypertexte illustré sur le label « Se créer un compte ».

- L’utilisateur a oublié son mot de passe, il rentre l’adresse mail dans le champs dédiée à cela et cliquer sur le lien hypertexte « mot de passe oublié ».



* Ecran création de compte

Cette page permet à l’utilisateur de se créer un compte. Cela ajoute les informations à la base de données Oracle. Cette page est l’équivalent de notre page surprise.



* Un écran avec moteur de recherche (gérer la liste des données d’une table)

Dans cette écran, l’utilisateur peut réaliser plusieurs filtres. Il peut choisir de visualiser une liste de vêtement correspondant à la taille choisie et/ou à la couleur choisie et/ou la catégorie. Plus il y aura de filtre, moins il y aura de vêtements affichés dans la liste illustré dans le tableau situé au bas de la page. Les filtres sont cumulables et envoie un résultat instantané sur la liste.



* Ecran de gestion de vêtements

Cette page permet à l’interlocuteur de choisir d’ajouter ou supprimer un vêtement.

* La suppression d’un vêtement permet de supprimer toutes les déclinaisons liées à un vêtement. Les déclinaisons d’un vêtement choisis sont la taille, la couleur et la catégorie. L’utilisateur devra sélectionner le nom du vêtement qu’il veut supprimer.
* L’ajout d’un vêtement se fait d’abord par l’entrée du nom du vêtement et de sa description. L’utilisateur doit ensuite cliquer sur le bouton ajouter et cela renverra automatiquement à la page « écran de gestion de déclinaison ».



* Ecran de gestion de déclinaison

Cet écran permet à l’utilisateur d’ajoute ou supprimer des déclinaisons de vêtements.

Le remplissage de tous les champs est obligatoire lorsque l’utilisateur à auparavant cliqué sur le bouton « ajouter » de l’écran « gestion de vêtement ».

Dans le cas contraire, l’utilisateur peut ajouter ou supprimer une quantité d’une déclinaison de vêtement en sélectionnant dans un premier temps le nom du vêtement (comboBox situé en haut de l’écran), la couleur, la taille et la catégorie du vêtement. La démarche est la même pour supprimer une quantité d’une déclinaison du vêtement. Il faudra pour cela cliquer sur le bouton supprimer.

