

CAS Biocoop

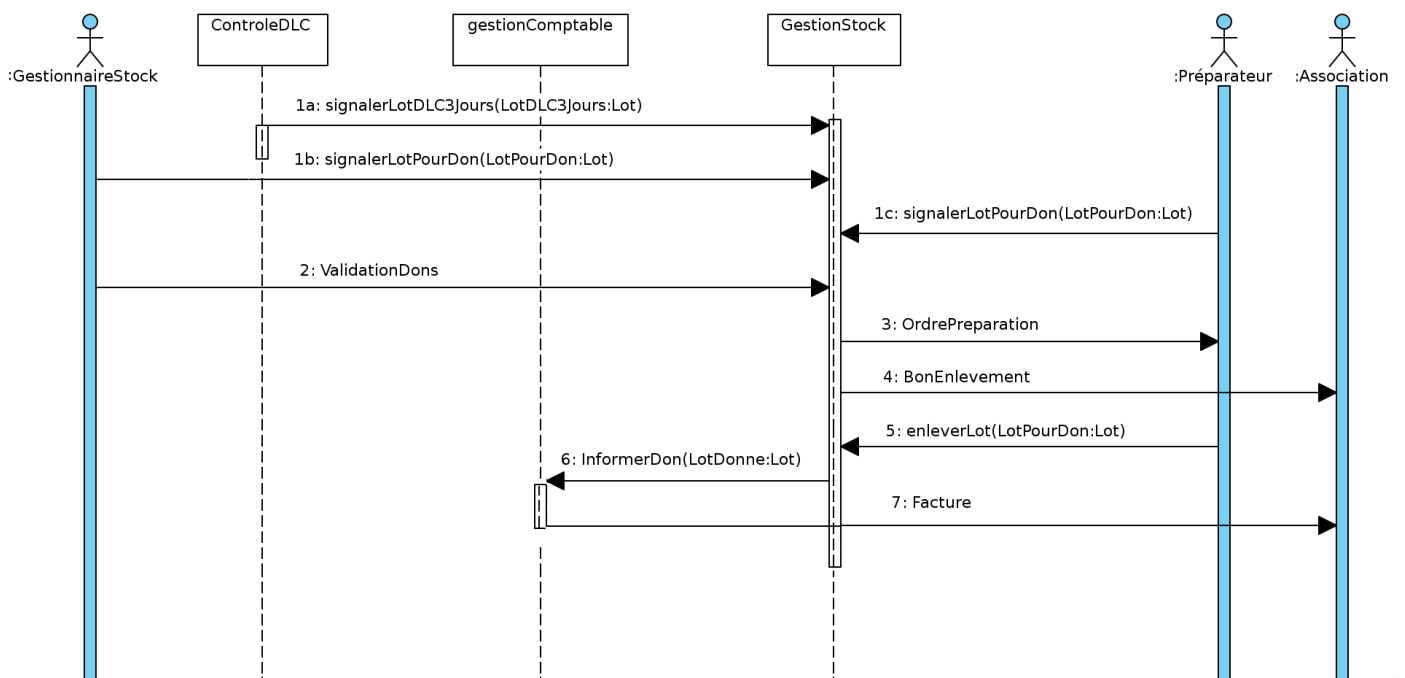
Proposition de corrigé

Dossier A – Mise en place de la gestion des dons

Mission A.1 – Mise à jour de la documentation

Question A.1.1

Réaliser le diagramme de séquence représentant le processus de gestion des dons.



Mission A.2 – Adaptation des éléments de solution de la base de données

Question A.2.1

Fournir les requêtes permettant d'obtenir ces résultats.

- a) la liste des lots (numéro du lot, désignation du produit concerné, nombre de produits du lot) **réceptionnés** en 2019 et dont l'état est « lot donné à une association caritative » ;

```
Select id as "numéro du lot", designation, nbrProduits as "nombre de produits du lot" from Lot
join Produit on refProduit = reference
where etat = 6
and year(dateReception) = 2019;
```

b) la liste des associations caritatives (nom) n'ayant pas reçu de don :

```
Select nom as "Associations n'ayant pas reçu de dons"
From AssociationCaritative
where id not in (select distinct idAssociation
                from FactureDon)
```

Remarque : la clause distinct n'est pas exigée

c) le montant cumulé de la valeur des marchandises données aux associations par année et par famille de produit. Pour répondre à ce besoin, vous utiliserez la vue SQL **ValeurDon** fournie dans le dossier documentaire :

```
Select year(dateFacture),libelle,sum(valeurMarchandise) as "total des dons"
From ValeurDon
Group by year(dateFacture),libelle
```

Question A.2.2

Modifier le déclencheur **before_update_Lot** pour prendre en compte les deux nouvelles valeurs 5 et 6 du code état du lot permettant de gérer les dons de lots.

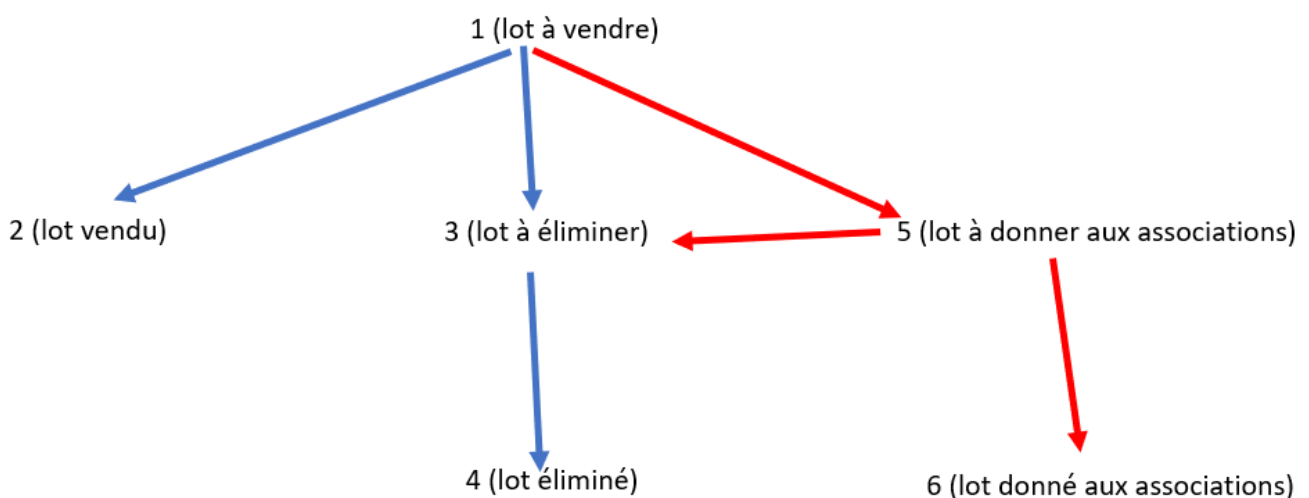
Il faut compléter la vérification de l'état du lot en tenant compte des nouvelles valeurs :

```
IF ((NEW.etat = 2 AND OLD.etat != 1)
    OR (NEW.etat = 3 AND OLD.etat != 1)
    OR (NEW.etat = 4 AND OLD.etat != 3)) THEN
```

devient :

```
IF ((NEW.etat = 2 AND OLD.etat !=1)
    OR (NEW.etat = 3 AND OLD.etat !=1 AND OLD.etat !=5)
    OR (NEW.etat = 4 AND OLD.etat !=3)
    OR (NEW.etat = 5 AND OLD.etat !=1)
    OR (NEW.etat = 6 AND OLD.etat !=5 )) THEN
```

Graphe d'état (non demandé) :



Ajouté : →

Initial : →

Dossier B – Transmission du catalogue des promotions

Mission B.1 – Développement des classes métiers et du gestionnaire d'accès aux données

Question B.1.1

Écrire le code du constructeur de la classe FamilleProduit.

```
// le constructeur
public function __construct($unId, $unLibelle)
{
    $this->id = $unId;
    $this->libelle = $unLibelle;
}
```

Question B.1.2

Donner deux arguments en faveur de l'utilisation d'une requête paramétrée dans la méthode *getLesPromotions* de la classe *GestionnaireBDD*.

Voici quelques arguments possibles pour répondre à la question. Tout autre argument cohérent sera accepté

Au niveau de la sécurité :

- Vérification du type ;
- échappement des caractères spéciaux pour une meilleure protection contre les attaques de type injection SQL.

Au niveau de l'efficacité :

- Une requête préparée peut être utilisée pour plusieurs exécutions dans un script.

Au niveau de la lisibilité :

- Une requête paramétrée est plus lisible qu'une requête comportant des apostrophes pour encadrer les valeurs de type chaîne et des opérateurs de concaténation pour assembler les éléments constants et les variables.

Question B.1.3

Écrire le code de la méthode *getUneFamille* de la classe *GestionnaireBDD*.

```
// fournit l'objet FamilleProduit à partir de son identifiant
// (ou l'objet null si l'identifiant n'existe pas)
public function getUneFamille($unIdFamille)
{
    // préparation de la requête et extraction des données
    $txtReq = "Select libelle from FamilleProduit where id = :idFamille";
    $req = $this->cnx->prepare($txtReq);
    $req->bindValue("idFamille", $unIdFamille, PDO::PARAM_STR);
    $req->execute();
    $uneLigne = $req->fetch(PDO::FETCH_OBJ);
    $req->closeCursor(); // libère les ressources du jeu de données

    $uneFamille = null;
    if ($uneLigne != false) {
        // création d'un objet FamilleProduit
        $unLibelle = $uneLigne->libelle;
        $uneFamille = new FamilleProduit($unIdFamille, $unLibelle);
    }
    return $uneFamille;
}
```

Question B.1.4

Écrire le code de la méthode *getPourcentageReduction* de la classe *LignePromotion*.

```
public function getPourcentageReduction()
{
    $tarifBase = $this->leProduit->getTarif() ;
    return 100 * ($tarifBase - $this->tarif)/ $tarifBase;
}
```

Question B.1.5

Écrire le code de la méthode *getLesLignesPromoRed* de la classe *Promotion* qui retourne une collection contenant les lignes de promotions pour lesquelles le pourcentage de réduction est supérieur ou égal à la valeur transmise à la méthode (par exemple : on fournira 20 pour 20%).

```
public function getLesLignesPromoRed($pourcentageReduc)
{
    $lesLignesARetourner = array();
    foreach($lesLignes as $uneLigne) {
        if ($uneLigne->getPourcentageReduction() >= $pourcentageReduc) {
            $lesLignesARetourner[] = $uneLigne;
        }
    }
    return $lesLignesARetourner;
}
```

Mission B.2 – Développement du contrôleur *PromotionsControleur***Question B.2.1**

Indiquer les adaptations de code à apporter à la méthode **lister** du contrôleur **PromotionsControleur** afin d'obtenir le résultat attendu. Pour chaque adaptation, préciser les numéros de ligne concernés.

Le code de la méthode *lister* est repris ci-dessous en barrant les lignes à supprimer et en mettant en gras les lignes à ajouter.

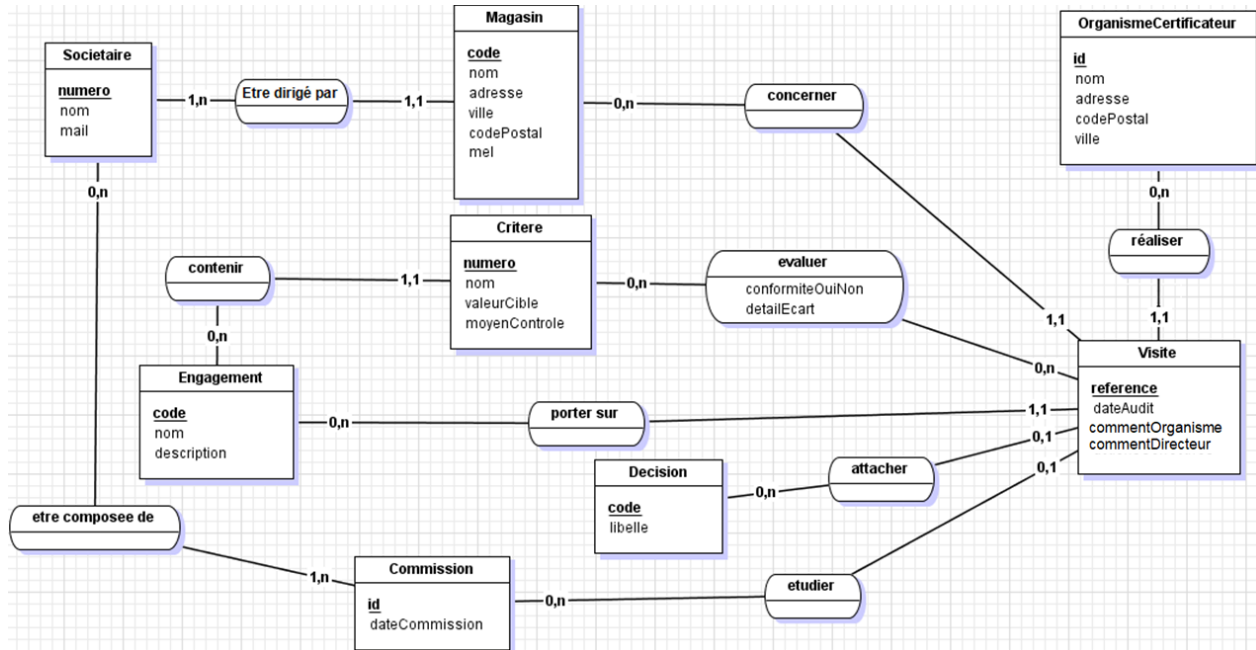
```
15.     $lesLignesDuTableau = array();
16.     foreach ($lesPromotions as $unePromotion) {
17.         // crée une ligne dans le tableau
18.         $uneLigne = array();
19.         $uneLigne["id"] = $unePromotion->getId();
20.         $uneLigne["libelle"] = $unePromotion->getLibelle();
21.         $uneLigne["mois"] = $unePromotion->getMois();
22.         $uneLigne["annee"] = $unePromotion->getAnnee();
23.         $uneLigne["idFamille"] = $unePromotion->getLaFamille()->getId();
24.         $lesLignesDuTableau[] = $uneLigne;
25.     }
26.     // 2-construction de l'élément "promotions"avec un tableau associatif,
27.     // ensemble de paires clé=>valeur
28.     $elt_promotions = [ "promotions" =>    $lesLignesDuTableau    ];
29.     // 3-construction de l'élément "data"
30.     $elt_data = [ "mois" =>    $mois,
31.                 "annee" =>    $annee,
32.                 "reponse" =>    $reponse,
33.                 "donnees" =>    $elt_promotions    ];
34.     // 4-construction de la racine
35.     $elt_racine = [ "data" =>    $elt_data    ];
```

Dossier C – Suivi des engagements sociétaires

Mission C.1 – Evolution de la structure de base de données

Question C.1.1

Modifier la structure de la base de données afin d'y intégrer la gestion des engagements, des visites et des commissions.



Remarque : on acceptera une fausse ternaire pour interpréter la phrase « Plusieurs commissions EES sont programmées à une date donnée »

Diagramme de classes

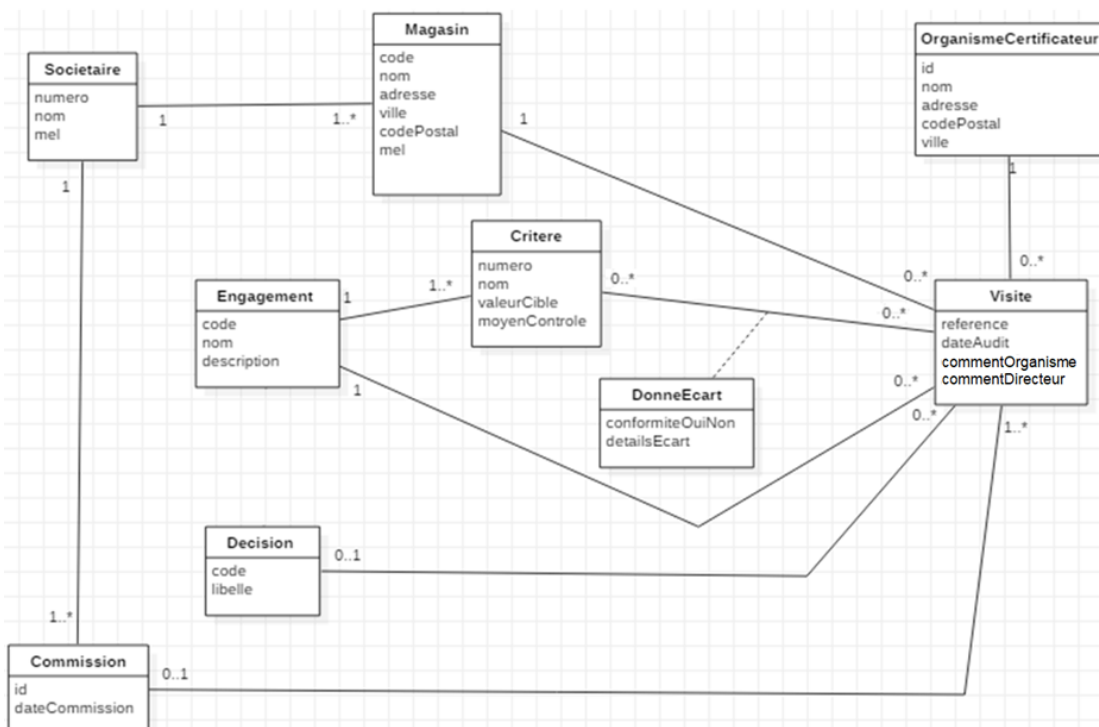


Schéma relationnel

Societaire (numero, nom, mel)

Clé primaire : numero

Magasin (code, nom, adresse, ville, codePostal, mel, numDirMagasin)

Clé primaire : code

Clé étrangère : numDirMagasin en référence à numero de Societaire

OrganismeCertificateur (id, nom, adresse, codePostal, ville)

Clé primaire : id

Engagement(code, nom, description)

Clé primaire : code

Visite (reference, dateAudit, commentOrganisme, commentDirecteur, codeMag, idOrg, codeEngagement, codeDecision, idCommission)

Clé primaire : référence

Clé étrangère : codeMag en référence à code de Magasin

idOrg en référence à id de OrganismeCertificateur

codeEngagement en référence à code de Engagement

codeDecision en référence à code de Decision

idCommission en référence à id de Commission

Decision (code, libelle)

Clé primaire : code

Critere (numero, nom, valeurCible, moyenControle)

Clé primaire : numero

Commission (id, dateCommission)

Clé primaire : id

Evaluer (refVisite, numCritere, detailsEcart)

Clé primaire : refVisite, numCritere

Clé étrangère : refVisite en référence à reference de Visite

numCritere en référence à numero de Critere

CompositionCommission (idCommission, numeroSocietaire)

Clé primaire : idCommission, numeroSocietaire

Clé étrangère : idCommission en référence à id de Commission

numeroSocietaire en référence à numero de Societaire

Mission C.2 – Prise en main de la plateforme d'intégration continue

Question C.2.1

Présenter deux avantages de la mise en place d'une plateforme d'intégration continue pour les applications développées par la DSI de Biocoop.

- Vérification automatique de la qualité du code source et tests de non régression sur chaque nouvelle version de code source,
- Tests automatisés,
- Détection au plus tôt des problèmes dans la chaîne de développement,
- Version toujours disponible pour une recette ou une démonstration => développement agile,
- Meilleure visibilité et suivi pour la DSI.

Question C.2.2

a) Donner les intitulés des scénarios de test manquants pour valider totalement le récit utilisateur « Consulter les visites de contrôle des magasins ».

Scénario : Rechercher une visite sur le seul critère de recherche engagement

Scénario : Rechercher une visite sur le seul critère de recherche magasin

Scénario : Rechercher une visite sur les deux critères de recherche magasin et engagement

Question C.2.2

b) Ecrire un des scénarios de test manquants de votre choix, en respectant la syntaxe utilisée et le jeu de données fourni

Scénario : Rechercher une visite sur le seul critère de recherche engagement

Etant donné que j'ai choisi Distribution produits magasin comme critère de recherche des engagements

Et que j'ai laissé à vide l'autre critère de recherche

Quand je valide ma recherche

Alors la liste suivante s'affiche

Date visite	Nom magasin	Engagement
01/04/2020	AmonBio	Distribution produits magasin
10/04/2020	Bioestbon	Distribution produits magasin
15/05/2020	AmonBio	Distribution produits magasin
15/05/2020	GrainBio	Distribution produits magasin

Ou bien

Scénario : Rechercher une visite sur le seul critère de recherche magasin

Etant donné que j'ai choisi Bioestbon comme critère de recherche magasin

Et que j'ai laissé à vide l'autre critère de recherche

Quand je valide ma recherche

Alors la liste suivante s'affiche

Date visite	Nom magasin	Engagement
10/04/2020	Bioestbon	Distribution produits magasin
25/04/2020	Bioestbon	Règlement de marque

Ou bien

Scénario : Rechercher une visite sur critère de recherche magasin et critère de recherche engagement

Etant donné que j'ai choisi Règlement de marque comme critère de recherche engagements et Bioestbon comme critère de recherche magasin

Quand je valide ma recherche

Alors la liste suivante s'affiche

<i>Date visite</i>	<i>Nom magasin</i>	<i>Engagement</i>
25/04/2020	Bioestbon	Règlement de marque

Question C.2.3

Présenter (nommer et définir brièvement) à votre cheffe de projet deux types de tests, autres que les tests fonctionnels, qui pourraient être également mis en place sur le serveur d'intégration continue.

Cette liste est non exhaustive :

- Tests unitaires : les tests portent sur un seul composant : classe, bibliothèque de fonctions
- Tests de charge : les tests visent à valider l'application lorsqu'elle est soumise à plusieurs demandes simultanées, le nombre de ces demandes pouvant évoluer par tranches
- Tests de performance : les tests visent à évaluer les temps de réponse de l'application et l'occupation en mémoire vive
- Tests de sécurité : les tests visent à évaluer le comportement de l'application face à certaines failles connues : injection SQL, faille XSS, faille CSRF, et forcer des accès non autorisés à certaines ressources.
- Test de non régression : tests réalisés après une modification, pour s'assurer que des défauts n'ont pas été introduits ou découverts dans des parties **non modifiées** de l'application. Ces tests sont effectués quand l'application ou son environnement est modifié.