Université de Toulouse
IUT de Blagnac
Département Informatique

Exploitation BD

Livrable 1

Nom : BOULOUIHA Yassir, DELAPLAGNE Titouan

Informatique 1ière Année

Table des matières :

Contexte

En ce début de semestre, il nous a été demandé d'aider la société Decasport. C'est une société spécialisée dans la revente de matériel de sport, dont la base de données ne correspond pas à leurs besoins actuels.

Ainsi notre travail consiste à rétroconcevoir cette base de données pour la mettre à jour, réparer ses défauts, et la perfectionner. Cette première semaine consistera à découvrir la base de données, notamment en rétro-concevant les différents schémas décrivant cette base.

Nous devons donc créer le Schéma Relationnel, le Dictionnaire de Données et le Diagramme de Classe.

Enfin nous devrons également lister les défauts actuels pour préparer les différentes modifications que nous effectuerons dans les prochaines semaines.

Schéma Logique (Relationnel)

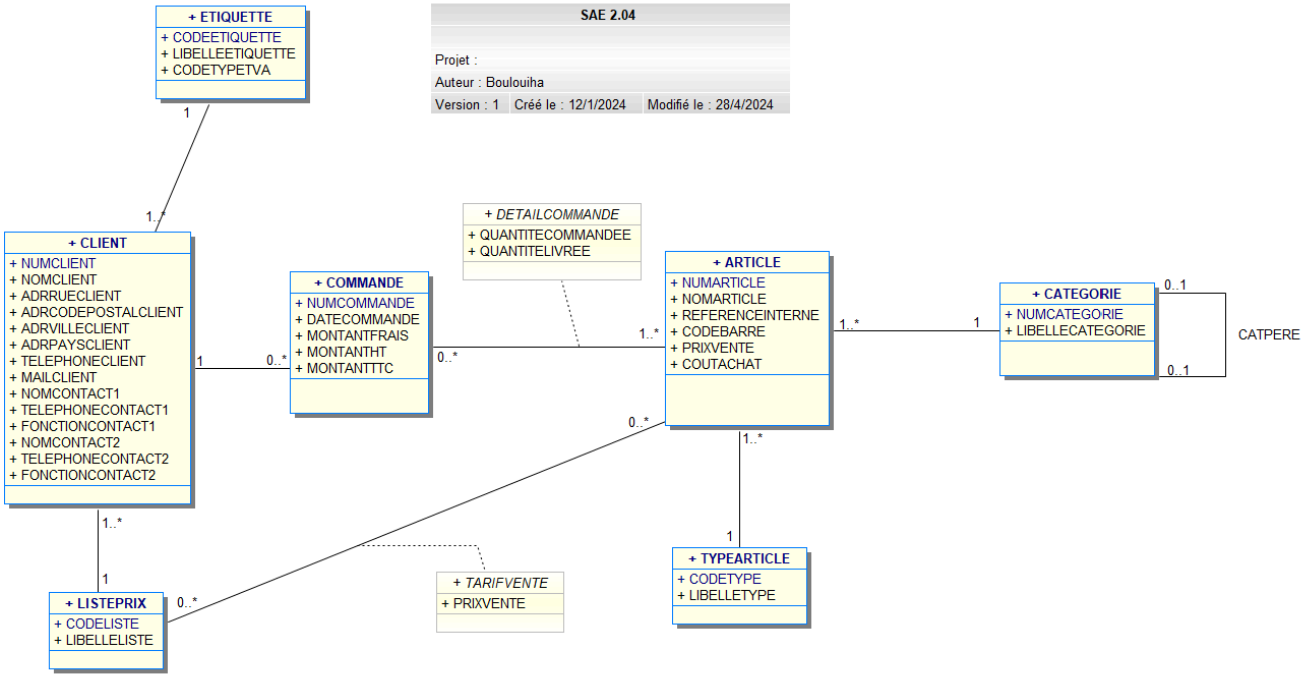
La première étape de la rétro-conception de cette base de données est de créer le schéma relationnel de la base.

Commande(NumCommande, NumClient#, DateCommande, MontantFrais, MontantHT,MontantTTC)
Client(NumClient, CodeListe#, CodeEtiquette#, NomClient, AdrRueClient, AdrCodePostalClient, AdrVilleClient, AdrPaysClient, TelephoneClient, MailClient, NomContact1, TelephoneContact1, FonctionContact1, NomContact2, TelephoneContact2, FonctionContact2)
Etiquette(CodeEtiquette, LibelleEtiquette, CodeTypeTVA)
TypeArticle(CodeType, LibelleType)
Article(NumArticle, NumCategorie#, CodeType#, NomArticle, ReferenceInterne, CodeBarre, PrixVente,CoutAchat)
Categorie(NumCategorie, NumCatPere#, LibelleCategorie)
TarifVente(NumArticle#, CodeListe#, PrixVente)
ListePrix(CodeListe, LibelleListe)
DetailCommande(NumComande#, NumArticle#, QuantiteCommandee, QuantiteLivree)

Schéma Conceptuel de la Base de Données

Avant de pouvoir créer le diagramme de classe de la base il est utile de commencer par créer le dictionnaire des données de celle-ci ainsi que le diagramme de classes UML. Ceux-ci permettent d'avoir une meilleure idée du fonctionnement de la Base de Données.

Diagramme de classes UML



Dictionnaire de données

Table	Nom de la colonne	Description	Type	Nullable	Contraintes	Calcul
Article	CODEBARRE	Code-barres	CHAR(13 BYTE)	Oui		
	COUTACHAT	Coût d'achat	NUMBER(10, 2)	Oui		
	NOMARTICLE	Nom de l'article	VARCHAR2(50 BYTE)	Oui		
	NUMARTICLE	Numéro de l'article	NUMBER(38, 0)	Non		
	PRIXVENTE	Prix de vente	NUMBER(10, 2)	Oui		
	REFERENCEINTERNE	Référence interne	VARCHAR2(10 BYTE)	Oui		
Categorie	LIBELLECATEGORIE	Libellé de la catégorie	VARCHAR2(40 BYTE)	Oui		
	NUMCATEGORIE	Numéro de la catégorie	NUMBER(38, 0)	Non		
	NUMCATPERE	Numéro de la catégorie parente	NUMBER(38, 0)	Oui		
Client	ADRCODEPOSTALCLIENT	Le code postal du client	VARCHAR2(10 BYTE)	Oui		
	ADRPAYSCLIENT	Pays du client	VARCHAR2(30 BYTE)	Oui		
	ADRRUECLIENT	Rue du client	VARCHAR2(50 BYTE)	Oui		
	ADRVILLECLIENT	Ville du client	VARCHAR2(40 BYTE)	Oui		
	FONCTIONCONTACT1	Fonction du contact 1	VARCHAR2(20 BYTE)	Oui		
	FONCTIONCONTACT2	Fonction du contact 2	VARCHAR2(20 BYTE)	Oui		

	MAILCLIENT	Adresse e-mail du client	VARCHAR2(60 BYTE)	Oui		
	NOMCLIENT	Nom du client	VARCHAR2(50 BYTE)	Oui		
	NOMCONTACT1	Nom du contact 1	VARCHAR2(50 BYTE)	Oui		
	NOMCONTACT2	Nom du contact 2	VARCHAR2(50 BYTE)	Oui		
	NUMCLIENT	Numéro du client	NUMBER(38,0)	Non		
	TELEPHONECLIENT	Téléphone du client	CHAR(12 BYTE)	Oui		
	TELEPHONECONTACT1	Téléphone du contact 1	CHAR(12 BYTE)	Oui		
	TELEPHONECONTACT2	Téléphone du contact 2	CHAR(12 BYTE)	Oui		
Commande	DATECOMMANDE	Date de la commande	DATE	Oui		
	MONTANTFRAIS	Montant des frais	NUMBER(10,2)	Oui		
	~~MONTANTHT~~	Montant hors taxes	NUMBER(10,2)	Oui		somme(PrixVente*QuantiteCommandee)
	~~MONTANTTTC~~	Montant toutes taxes comprises	NUMBER(10,2)	Oui		MONTANTHT*(TVA+1)
	NUMCOMMANDE	Numéro de commande	NUMBER(38,0)	Non		
DetailCommande	QUANTITECOMMANDEE	Quantité commandée	NUMBER(38,0)	Oui		
	QUANTITELIVREE	Quantité livrée	NUMBER(38,0)	Oui		
Etiquette	CODEETIQUETTE	Code étiquette	CHAR(2 BYTE)	Non		

	CODETYPETVA	Code de type TVA	NUMBER(38, 0)	Oui		
	LIBELLEETIQUETTE	Libellé de l'étiquette	VARCHAR2(70 BYTE)	Oui		
ListePrix	CODELISTE	Code de liste	CHAR(1 BYTE)	Non		
	LIBELLELISTE	Libellé de la liste	VARCHAR2(20 BYTE)	Oui		
TarifVente	PRIXVENTE	Prix de vente	NUMBER(10, 2)	Oui		
TypeArticle	CODETYPE	Code de type	CHAR(1 BYTE)	Non		
	LIBELLETYPE	Libellé du type	VARCHAR2(40 BYTE)	Oui		

Défauts actuels

Grâce à ces différents schémas nous avons pu remarquer différents soucis de conception que présente la base de données. Nous devons donc maintenant les lister, cette liste sera un outil plus qu'utile pour l'amélioration de la base de données.

Table(s) Concernée(s)	Défaut	Solution envisagée
Client	On pourrait trouver un dédoublement d'information quant aux deux contacts du client.	Il serait mieux de créer une table dédiée et simplement avoir les deux numéro de contact comme clés étrangère dans la table client.
Client	La table client contient 4 champs dédiés à	Il serait utile de créer une table dédiée à

	l'adresse, bien que cela ne soit pas une nécessité de les placer dans une table dédiée serait plus propre.	l'adresse des clients.
Client Etiquette ListePrix	Les tables Etiquette et ListePrix sont reliés au travers de la table Client, ce qui n'a pas de sens et provoquera certainement un dédoublement de certaines données. Cela complexifie également inutilement l'utilisation de la base.	Relier ces deux tables directement ou par une classe d'association
DetailCommande	Information quantitée commandée en double	La quantité commandée est celle qui doit être livrée normalement si on part du principe que la commande est expédiée en un seul coup et que cela ne servirait pas à assurer un suivi des articles qu'il reste à livrer.
Article Tarif Vente	Le prix de vente apparaît sur les deux tables qui sont pourtant reliés ainsi soit l'information et	Si l'information est dédoublée nous devons supprimer l'un des doublons sinon le nom de l'un d'entre eux doit

	dédoublée soit les deux ne décrivent pas la même chose.	être changé.

Si on part également du principe qu'il n'y a pas de contrôle de la saisie utilisateur avant d'entrer les données dans la base de données il faudrait ajouter des contraintes CHECK sur les attributs de prix pour qu'ils soient strictement positifs.

Conclusion

La Base de données utilisée par Decasport est remplie de défauts de conception. On y trouve des informations dédoublées, des liens entre tables qui sont erronés, ,des tables de trop et d'autres manquantes.

Ce premier exercice de rétro-conception nous a permis au travers des différents schémas effectués cette semaine de mieux comprendre cette base de données et ses défauts et seront des outils essentiels pour les différentes corrections que nous aurons à effectuer.

Université de Toulouse
IUT de Blagnac
Département Informatique

Exploitation BD

Livrable 2

Nom : BOULOUIHA Yassir, DELAPLAGNE Titouan

Informatique 1ière Année

Table des matières :

Contexte

En ce début de semestre, il nous a été demandé d'aider la société Decasport. C'est une société spécialisée dans la revente de matériel de sport, dont la base de données ne correspond pas à leurs besoins actuels.

En cette deuxième semaine, après avoir constaté les défauts de leur base de données actuelle, nous allons mettre à jour la base de données de façon à répondre aux nouveaux besoins de l'entreprise ainsi que de régler les soucis de conception déjà relevés auparavant.

Cela passera par :

- Concevoir et implémenter la gestion des articles par taille, des fournisseurs et des règles de réapprovisionnement.
- Mettre à jour le schéma relationnel, le dictionnaire de données et le diagramme de classes pour refléter les modifications apportées.
- Créer les scripts SQL nécessaires pour modifier la base de données existante et la mettre à jour selon le nouveau schéma.

Schéma Logique (Relationnel)

Voici le nouveau schéma logique de la base de données avec les modifications affichées en rouge.

Commande(NumCommande, NumClient#, DateCommande, MontantFrais, MontantHT,~~MontantTTC~~)

Client(NumClient, CodeListe#, CodeEtiquette#, NomClient, AdrRueClient, AdrCodePostalClient, AdrVilleClient, AdrPaysClient, TelephoneClient, MailClient, ~~NomContact1, TelephoneContact1, FonctionContact1, NomContact2, TelephoneContact2, FonctionContact2~~)

Etiquette(CodeEtiquette, LibelleEtiquette, CodeTypeTVA)

TypeArticle(CodeType, LibelleType)

Article(NumArticle, NumCategorie#, CodeType#, NomArticle, ReferenceInterne, CodeBarre, PrixVente,~~CoutAchat~~)

Categorie(NumCategorie, NumCatPere#, LibelleCategorie)

TarifVente(NumArticle#, CodeListe#, PrixVente)

ListePrix(CodeListe, LibelleListe)

DetailCommande(NumComande#, NumArticle#, QuantiteCommandee, QuantiteLivree)

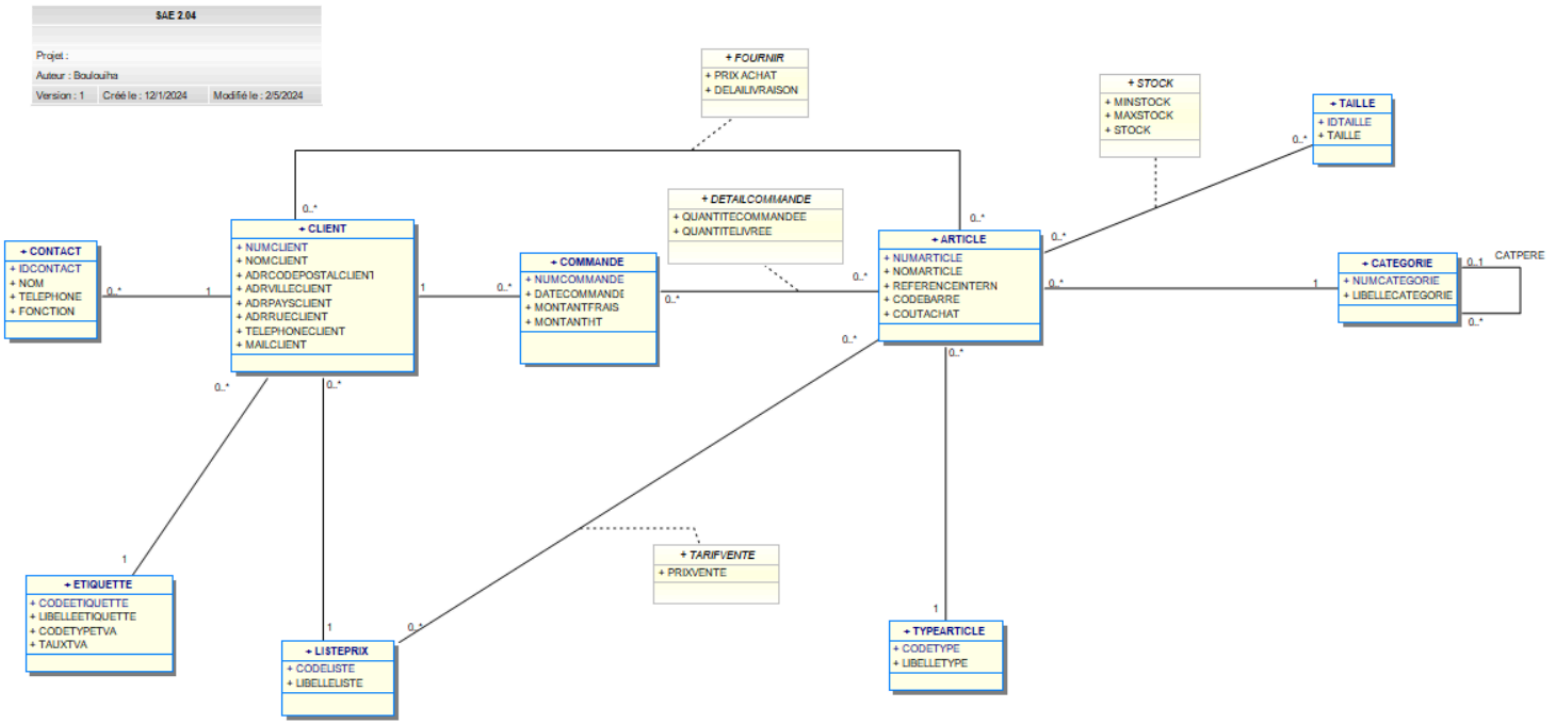
Taille(IdTaille, Taille)

Stock(NumArticle, IdTaille, MinStock, MaxStock, Stock)

Fournir(NumClient, NumArticle, Prix_Achat, DelaiLivraison)

Nouveau Schéma Conceptuel de la Base de Données

Ceci est le nouveau schéma conceptuel en tenant compte des erreurs de conception précédemment constatées et des besoins communiqués par l'entreprise Decasport.



Dictionnaire de données

Table	Nom de la colonne	Description	Type	Nullable	Contraintes	Calcul
Article	CODEBARRE	Code-barres	CHAR(13 BYTE)	Oui		
	COUTACHAT	Coût d'achat	NUMBER(10,2)	Oui		
	NOMARTICLE	Nom de l'article	VARCHAR2(50 BYTE)	Oui		
	NUMARTICLE	Numéro de l'article	NUMBER(38,0)	Non		
	REFERENCEINTERNE	Référence interne	VARCHAR2(10 BYTE)	Oui		
Categorie	LIBELLECATEGORIE	Libellé de la catégorie	VARCHAR2(40 BYTE)	Oui		
	NUMCATEGORIE	Numéro de la catégorie	NUMBER(38,0)	Non		
	NUMCATPERE	Numéro de la catégorie parente	NUMBER(38,0)	Oui		
Client	ADRCODEPOSTALCLIENT	Le code postal du client	VARCHAR2(10 BYTE)	Oui		
	ADRPAYSCLIENT	Pays du client	VARCHAR2(30 BYTE)	Oui		
	ADRRUECLIENT	Rue du client	VARCHAR2(50 BYTE)	Oui		
	ADRVILLECLIENT	Ville du client	VARCHAR2(40 BYTE)	Oui		

	MAILCLIENT	Adresse e-mail du client	VARCHAR2(60 BYTE)	Oui		
	NOMCLIENT	Nom du client	VARCHAR2(50 BYTE)	Oui		
	NUMCLIENT	Numéro du client	NUMBER(38,0)	Non		
	TELEPHONECLIENT	Téléphone du client	CHAR(12 BYTE)	Oui		
Commande	DATECOMMANDE	Date de la commande	DATE	Oui		
	MONTANTFRAIS	Montant des frais	NUMBER(10,2)	Oui		
	MONTANTHT	Montant hors taxes	NUMBER(10,2)	Oui		somme(PrixVente *QuantiteCommande)
	NUMCOMMANDE	Numéro de commande	NUMBER(38,0)	Non		
DetailCommande	QUANTITECOMMANDEE	Quantité commandée	NUMBER(38,0)	Oui		
	QUANTITELIVREE	Quantité livrée	NUMBER(38,0)	Oui		
Etiquette	CODEETIQUETTE	Code étiquette	CHAR(2 BYTE)	Non		
	CODETYPETVA	Code de type TVA	NUMBER(38,0)	Oui		
	TAUXTVA	Taux de TVA a appliquer	NUMBER(2,2)	Oui		
	LIBELLEETIQUETTE	Libellé de l'étiquette	VARCHAR2(70 BYTE)	Oui		
ListePrix	CODELISTE	Code de liste	CHAR(1 BYTE)	Non		
	LIBELLELISTE	Libellé de la liste	VARCHAR2(20 BYTE)	Oui		

TarifVente	PRIXVENTE	Prix de vente	NUMBER(10,2)	Oui		
TypeArticle	CODETYPE	Code de type	CHAR(1 BYTE)	Non		
	LIBELLETYPE	Libellé du type	VARCHAR2(40 BYTE)	Oui		
Taille	IDTAILLE	Identifiant de la taille	VARCHAR2(20)	Non		
	TAILLE	Taille de l'article	VARCHAR2(20)	Oui		
Fournir	NUMCLIENT	Numéro du client	NUMBER(38,0)	Non		
	NUMARTICLE	Numéro de l'article	NUMBER(38,0)	Non		
	PRIX_ACHAT	Prix d'achat de l'article	NUMBER(10,2)	Oui		
	DELAILIVRAISON	Délai de livraison	NUMBER(4,0)	Oui		
Stock	MINSTOCK	Stock minimal avant achat	NUMBER(6,0)	Oui		
	MAXSTOCK	Stock maximal après achat	NUMBER(6,0)	Oui		
	STOCK	Niveau actuel de stock	NUMBER(9,0)	Oui		
Contact	IDCONTACT	Identifiant du contact	VARCHAR2(12 BYTE)	Non		
	NOMCONTACT	Nom du contact	VARCHAR2(50 BYTE)	Oui		
	TELEPHONECONTACT	Numéro de téléphone du contact	CHAR(12 BYTE)	Oui		
	FONCTIONCONTACT	Fonction du contact	VARCHAR2(20 BYTE)	Oui		

Scripts SQL

SQL-LDD

```
CREATE TABLE TAILLE
   (
    IDTAILLE VARCHAR2(20)  NOT NULL,
    TAILLE VARCHAR2(20)  NULL,
CONSTRAINT PK_TAILLE PRIMARY KEY (IDTAILLE)
   ) ;

CREATE TABLE Contact(
        IDCONTACT VARCHAR(12),
        NUMCLIENT NUMBER(38,0),
        NOMCONTACT VARCHAR2(50 BYTE),
        TELEPHONECONTACT CHAR(12 BYTE),
        FONCTIONCONTACT VARCHAR2(20 BYTE),
        CONSTRAINT pk_Contact PRIMARY KEY (IDCONTACT),
      CONSTRAINT fk_Contact_Client FOREIGN KEY (NUMCLIENT) REFERENCES Client
    );

CREATE TABLE STOCK
   (
    NUMARTICLE NUMBER(38,0)  NOT NULL,
    IDTAILLE VARCHAR2(20)  NOT NULL,
    MINSTOCK NUMBER(6)  NULL,
    MAXSTOCK NUMBER(6)  NULL,
    STOCK NUMBER(9)  NULL,
    CONSTRAINT PK_STOCK PRIMARY KEY (NUMARTICLE, IDTAILLE),
   CONSTRAINT fk_stock_NumArticle FOREIGN KEY (NUMARTICLE) REFERENCES Article,
    CONSTRAINT fk_stock_IdTaille FOREIGN KEY (IDTAILLE) REFERENCES Taille
   ) ;
```

```
CREATE TABLE FOURNIR
   (
    NUMCLIENT NUMBER(38,0)  NOT NULL,
    NUMARTICLE NUMBER(38,0)  NOT NULL,
    PRIX_ACHAT NUMBER(10,2)  NULL,
    DELAILIVRAISON NUMBER(4)  NULL,
    CONSTRAINT PK_FOURNIR PRIMARY KEY (NUMCLIENT, NUMARTICLE),
    CONSTRAINT fk_Fournir_NumClient FOREIGN KEY (NUMCLIENT) REFERENCES Client,
  CONSTRAINT fk_Fournir_NumArticle FOREIGN KEY (NUMARTICLE) REFERENCES Article
   );


--MODIFICATION TABLE CLIENT
ALTER TABLE CLIENT DROP (NOMCONTACT1, NOMCONTACT2, TELEPHONECONTACT1,
TELEPHONECONTACT2, FONCTIONCONTACT1, FONCTIONCONTACT2);

--MODIFICATION TABLE Article
ALTER TABLE Article DROP (PRIXVENTE);

--MODIFICATION TABLE ETIQUETTE
ALTER TABLE ETIQUETTE ADD TAUXTVA DECIMAL(2,2);

--MODIFICATION TABLE COMMANDE
ALTER TABLE COMMANDE DROP (MONTANTTTC);
```

SQL-LMD

Client

```
INSERT INTO contact VALUES ('01', 1,'Jean', 0457589662, 'Manager');
INSERT INTO contact VALUES ('02', 1,'Stephane', 0454589662, 'Directeur des ventes');
INSERT INTO contact VALUES ('03', 4,'Aurélien', 0489961722, 'Manager régional');
INSERT INTO contact VALUES ('04', 3,'André', 0458966214, 'Responsable');
INSERT INTO contact VALUES ('05', 12,'Alaric', 0458627475, 'Gerant Local');
INSERT INTO contact VALUES ('06', 16,'Eduard', 0452789314, 'Directeur des ventes');
INSERT INTO contact VALUES ('07',20,'Jana', 0458713997, 'Manager régional');
INSERT INTO contact VALUES ('08', 8,'Hannah', 0458947514, 'Manager');
```

Taille

```
INSERT INTO Taille VALUES ('1', 'XS');
INSERT INTO Taille VALUES ('2', 'S');
INSERT INTO Taille VALUES ('3', 'M');
INSERT INTO Taille VALUES ('4', 'L');
INSERT INTO Taille VALUES ('5', 'XL');
```

Stock

```
INSERT INTO Stock VALUES (1, '1', 5, 30, 10);
INSERT INTO Stock VALUES (1, '2', 2, 10, 10);
INSERT INTO Stock VALUES (30, '4', 0, 10, 2);
INSERT INTO Stock VALUES (25, '5', NULL, NULL, 10);
```

Fournir

```
INSERT INTO Fournir VALUES (1,1, 3.50, 2);
INSERT INTO Fournir VALUES (1,15, 8.99, 3);
INSERT INTO Fournir VALUES (15,1, 2.50, 7);
INSERT INTO Fournir VALUES (2,8, 10.99, 1);
INSERT INTO Fournir VALUES (4,30, 349.99, 4);
INSERT INTO Fournir VALUES (4,24, 9, 3);
INSERT INTO Fournir VALUES (4,5, 10, 9);
INSERT INTO Fournir VALUES (4,7,2.99, 1);
```

Etiquette

```
UPDATE etiquette
SET TauxTVA=0.2
WHERE CODEETIQUETTE='FR';

UPDATE etiquette
SET TauxTVA=0.2
WHERE CODEETIQUETTE='GB';

UPDATE etiquette
SET TauxTVA=0.08
WHERE CODEETIQUETTE='CH';

UPDATE etiquette
SET TauxTVA=0.2
WHERE CODEETIQUETTE='AT';

UPDATE etiquette
SET TauxTVA=0.22
WHERE CODEETIQUETTE='IT';
```

Conclusion

Cette semaine nous avons pu corriger les erreurs remarquées la semaine passé et répondre aux nouveaux besoins de l'entreprise en produisant un nouveau diagramme des classes UML, le schéma relationnel (logique) correspondant, ainsi que le script SQL-LMD et LDD, que nous avons utilisé pour mettre en application notre correction de la BD en l'exécutant sur SQL Developer. Cela nous a permis de monter en compétences dans ces trois exercices en étant confrontés à une situation pratique réelle.

Commande(NumCommande, NumClient#, DateCommande, MontantFrais, MontantHT)

Client(NumClient, CodeListe#, CodeEtiquette#, NomClient, AdrRueClient, AdrCodePostalClient, AdrVilleClient, AdrPaysClient, TelephoneClient, MailClient)

Etiquette(CodeEtiquette, LibelleEtiquette, CodeTypeTVA)

TypeArticle(CodeType, LibelleType)

Article(NumArticle, NumCategorie#, CodeType#, NomArticle, ReferenceInterne, CodeBarre, PrixVente)

Categorie(NumCategorie, NumCatPere#, LibelleCategorie)

TarifVente(NumArticle#, CodeListe#, PrixVente)

ListePrix(CodeListe, LibelleListe)

DetailCommande(NumComande#, NumArticle#, QuantiteCommandee, QuantiteLivree)

Taille(IdTaille, Taille)

Stock(NumArticle, IdTaille, MinStock, MaxStock, Stock)

Fournir(NumClient, NumArticle, Prix_Achat, DelaiLivraison)

Université de Toulouse
IUT de Blagnac
Département Informatique

Exploitation BD

Livrable 3

Nom : BOULOUIHA Yassir, DELAPLAGNE Titouan

Informatique 1ière Année

Table des matières :

Contexte

Cette semaine il nous a été demandé de mettre en place des requêtes SQL qui permettraient d'interroger la base de données de l'entreprise Decasport, afin d'en tirer des données qui seraient utiles au pilotage de l'entreprise, comme par exemple les articles commandés le plus souvent, ventes correspondant à des pays spécifiques et chiffres d'affaires par client.

Requêtes SQL

```sql
--a)
--1 Articles jamais commandés
SELECT A.*
FROM Article A, Detail_Commande D
WHERE A.numarticle = D.numarticle (+)
AND D.numarticle IS NULL;

--2 Categories jamais commandées
SELECT C.*
FROM Categorie C, Article A, Detail_Commande D
WHERE C.numcategorie = A.numcategorie
AND A.numarticle = D.numarticle (+)
AND D.numarticle IS NULL;

--b)
SELECT TO_CHAR(dateCommande,'mm') as Month,
SUM((PrixVente*QuantiteLivree*(tauxTva+100))/100 + MontantFrais) as
TotalProfitSuisse
FROM Commande C, Client Cl, Etiquette E, Tarif_Vente T, Detail_Commande D, TVA
    WHERE C.NumClient=Cl.numClient
    AND E.CodeEtiquette=Cl.codeEtiquette
    AND C.numCommande=D.numCommande
    AND D.numArticle=T.numArticle
    AND E.CodeTypeTva=Tva.CodeTypeTva
    AND Cl.codeListe=T.codeListe
    AND Cl.codeEtiquette='CH'
    AND C.DateCommande>'01/01/24'
GROUP BY TO_CHAR(dateCommande,'mm');
```

```sql
SELECT TO_CHAR(dateCommande,'IW') as Week,
SUM((PrixVente*QuantiteLivree*(tauxTva+100))/100 + MontantFrais) as
TotalProfitSuisse
FROM Commande C, Client Cl, Etiquette E, Tarif_Vente T, Detail_Commande D, TVA
    WHERE C.NumClient=Cl.numClient
    AND E.CodeEtiquette=Cl.codeEtiquette
    AND C.numCommande=D.numCommande
    AND D.numArticle=T.numArticle
    AND E.CodeTypeTva=Tva.CodeTypeTva
    AND Cl.codeListe=T.codeListe
    AND Cl.codeEtiquette='CH'
    AND C.DateCommande>'01/01/24'
GROUP BY TO_CHAR(dateCommande,'IW');

--c)
SELECT C.*
FROM Client C, Commande Cde
WHERE C.numclient = Cde.numclient (+)
AND Cde.numclient IS NULL;

--d)
SELECT nomArticle, numArticle
FROM Article A, Categorie C
WHERE NOT EXISTS ( SELECT codeTaille
                   FROM Taille
                   WHERE LIBELLETAILLE LIKE 'pointure%'
                   MINUS
                   SELECT codeTaille
                   FROM Variante_Article V
                   WHERE V.numArticle=A.numArticle)
    AND C.NumCategorie= A.NumCategorie
    AND C.LibelleCategorie='Chaussures';
```

```sql
--e)
SELECT numcommande, numarticle, quantitecommandee,
quantitelivree,(quantitecommandee - quantitelivree) AS DIFFERENCE
FROM Detail_Commande
WHERE quantitecommandee > quantitelivree;

--f)
SELECT SUM((PrixVente*QuantiteLivree*tauxTva)/100 + MontantFrais) as totalVente
FROM Commande C1, Client C2, Etiquette E, Tarif_Vente T, Detail_Commande D, TVA
WHERE C1.NumClient=C2.numClient
    AND E.CodeEtiquette=C2.codeEtiquette
    AND C1.numCommande=D.numCommande
    AND D.numArticle=T.numArticle
    AND E.CodeTypeTva=Tva.CodeTypeTva
    AND C2.codeListe=T.codeListe
    AND E.CodeTypeTva='1';

SELECT SUM((PrixVente*QuantiteLivree*tauxTva)/100 + MontantFrais) as totalVente
FROM Commande C1, Client C2, Etiquette E, Tarif_Vente T, Detail_Commande D, TVA
WHERE C1.NumClient=C2.numClient
    AND E.CodeEtiquette=C2.codeEtiquette
    AND C1.numCommande=D.numCommande
    AND D.numArticle=T.numArticle
    AND E.CodeTypeTva=Tva.CodeTypeTva
    AND C2.codeListe=T.codeListe
    AND E.CodeTypeTva='2';


SELECT SUM((PrixVente*QuantiteLivree*tauxTva)/100 + MontantFrais) as totalVente
FROM Commande C1, Client C2, Etiquette E, Tarif_Vente T, Detail_Commande D, TVA
WHERE C1.NumClient=C2.numClient
    AND E.CodeEtiquette=C2.codeEtiquette
    AND C1.numCommande=D.numCommande
    AND D.numArticle=T.numArticle
    AND E.CodeTypeTva=Tva.CodeTypeTva
    AND C2.codeListe=T.codeListe
    AND E.CodeTypeTva='3';
```

```sql
--g)
SELECT A.numarticle, A.nomarticle, A.codebarre, A.coutachat
FROM Article A, Detail_Commande D
WHERE A.numarticle = D.numarticle
GROUP BY A.numarticle, A.nomarticle, A.codebarre, A.coutachat
HAVING COUNT(*) = (SELECT MAX(COUNT(*))
                   FROM Detail_Commande
                   GROUP BY numarticle );

--h)
SELECT A.numarticle,A.nomarticle, A.coutachat, SUM(quantitecommandee) AS
TOTAL_COMMANDEE
FROM Article A, Detail_Commande D
WHERE A.numarticle = D.numarticle
GROUP BY A.numarticle,A.nomarticle, A.coutachat
HAVING SUM(quantitecommandee) = (SELECT MAX(SUM(quantitecommandee))
                                 FROM Detail_Commande
                                 GROUP BY numarticle);

--i)
SELECT C.numclient, nomclient, ROUND(SUM((NVL(prixvente, 0)*NVL(quantitelivree,
0)*(NVL(tauxTva, 0)+100))/100 + NVL(MontantFrais,0))) AS TOTAL_CA
FROM Client C, Commande Cde, Detail_Commande Dcde, Tarif_Vente Tv, Etiquette E,
Tva T
WHERE C.numclient = Cde.numclient (+) AND Cde.numcommande = Dcde.numcommande (+)
AND Dcde.numarticle = Tv.numarticle (+) AND C.codeetiquette = E.codeetiquette (+)
AND E.codetypetva = T.codetypetva (+)
GROUP BY C.numclient, nomclient
ORDER BY TOTAL_CA DESC;
```

```sql
--j)
SELECT NumClient, NomClient
FROM Client C1
WHERE NOT EXISTS ( SELECT D.numArticle
                   FROM Commande C2, Client C3, Detail_Commande D
                   WHERE C3.numClient=C2.numClient
                    AND D.numCommande=C2.numCommande
                    AND C3.NomClient='The North Face Pavia'
                   MINUS
                   SELECT NumArticle
                   FROM Commande C2, Detail_Commande D
                   WHERE C2.numClient=C1.numClient
                    AND D.numCommande=C2.numCommande)
    AND NomClient!='The North Face Pavia';
```

Conclusion

Cette semaine nous avons pu appliquer dans un contexte pratique les connaissances que nous avons acquises lors de la ressource R2.06 Exploitation Bases de Données, en utilisant les formes de requêtes qui nous ont semblé les plus propices à chaque situation lors des différentes problématiques de l'entreprise.