SAÉ S2.04

Livrable 2

Table des matières :

Introduction :

Après avoir analysé la base de données existante de la société Occaski, il était nécessaire de constater les défauts présents dans celle-ci afin de pouvoir les rectifier. Néanmoins, pour avoir une base de données optimisée, nous ne pouvions pas nous contenter d'ajuster simplement ces problèmes.

Il est également indispensable d'avoir une base de données qui corresponde aux nouveaux besoins émis par la société Occaski dans le cahier des charges qui nous a été transmis, pour réaliser cette transformation.

En premier lieu nous réaliserons le schéma conceptuel de la BD existante accompagné de son dictionnaire de données avant d'en déduire le schéma logique de la nouvelle BD sans oublier le script SQL-LDD et LMD de transformation de la BD.

I. Diagramme de classes UML (conceptuel) :

À partir de la base de données préexistante qui nous a été présentée, nous avons pu réaliser un diagramme de classes UML dans la première version.

Pour concevoir cette nouvelle base de données, nous avons tenu compte des exigences de la société ainsi que les erreurs constatées dans la version précédente afin d'implémenter un nouveau diagramme de classes UML que voici :

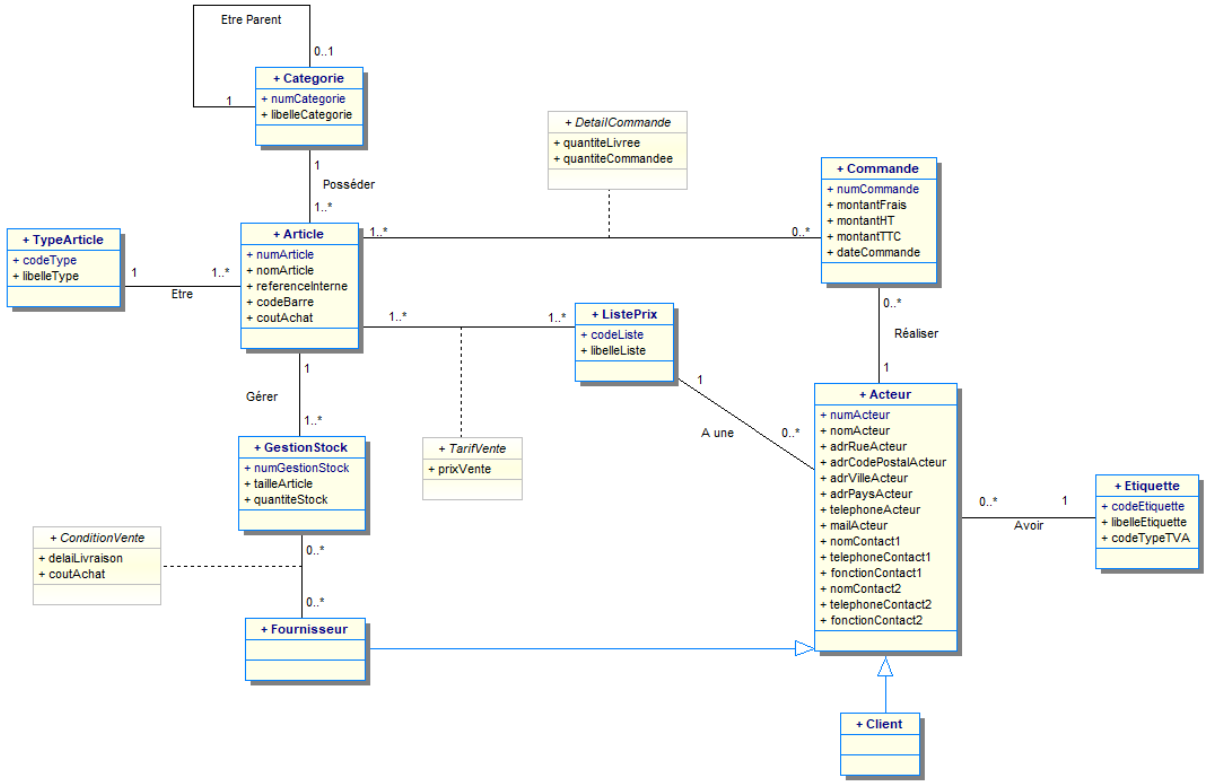

Diagramme de classes de la nouvelle base de données de "Occaski" réalisé avec l'outil Win'Design.

Remarque sur la TVA :

Il est intéressant de remarquer que le montant TTC que nous avons laissé dans la base de données tel quel, aurait pu être calculé à partir d'un attribut TVA. Pour respecter le format de la base de données existante, nous n'avons pas voulu implémenter cette solution.

II. Dictionnaire des données :

Voici le nouveau dictionnaire des données réalisé en fonction du schéma conceptuel ci-dessus. Il est important de mentionner que les relations CLIENT et FOURNISSEUR possèdent les mêmes attributs que ACTEUR (il est donc redondant de représenter à nouveau ces mêmes classes dans le dictionnaire de données) :

	Nom	Description	Type	Contrainte
Article	**numArticle**	N° (unique) d'un article	Nombre de longueur 22	
	nomArticle	Nom d'un article	Chaîne de 50 caractères	
	referenceInterne	Référence interne d'un client	Chaîne de 10 caractères	
	codeBarre	Code barre d'un article	Chaîne fixe de 13 caractères	
	coutAchat	Coût d'achat d'un article	Nombre de longueur 22	
Commande	**numCommande**	N° (unique) d'une commande	Nombre de longueur 22	
	montantFrais	Montant des frais d'une commande	Nombre de longueur 22	
	montantHT	Montant hors taxe d'une commande	Nombre de longueur 22	
	montantTTC	Montant tout tarif compris d'une commande	Nombre de longueur 22	
	dateCommande	Date de la commande effectuée	Date	

DetailCommande	quantiteLivree	Quantité livrée au client	Nombre de longueur 22	
	quantiteCommandee	Quantité commandée du client	Nombre de longueur 22	
TypeArticle	**codeType**	Code (unique) du type d'un article	Chaîne fixe d'un caractère	Ne doit pas être vide
	libelleType	Libellé du type de l'article	Chaîne de 40 caractères	
Categorie	**numCategorie**	N° (unique) de la catégorie	Nombre de longueur 22	
	libelleCategorie	Libellé de la catégorie	Chaîne de 40 caractères	
Acteur (Fournisseur et Client inclus)	**numActeur**	N° (unique) du client	Nombre de longueur 22	
	nomActeur	Nom du client	Chaîne de 50 caractères	
	adrRueActeur	Adresse concernant la rue du client	Chaîne de 50 caractères	
	adrCodePostalActeur	Adresse concernant le code postal du client	Chaîne de 10 caractères	
	adrVilleActeur	Adresse concernant la ville du client	Chaîne de 40 caractères	
	adrPaysActeur	Adresse concernant le pays du client	Chaîne de 30 caractères	
	telephoneActeur	Téléphone du client	Chaîne fixe de 12 caractères	

	mailActeur	Mail du client	Chaîne de 60 caractères	
	nomContact1	Nom du 1er contact	Chaîne de 50 caractères	
	telephoneContact1	Téléphone du 1er contact	Chaîne fixe de 12 caractères	
	fonctionContact1	Fonction du 1er contact	Chaîne de 20 caractères	
	nomContact2	Nom du deuxième contact	Chaîne de 50 caractères	
	telephoneContact2	Téléphone du deuxième contact	Chaîne fixe de 12 caractères	
	fonctionContact2	Fonction du second contact	Chaîne de 20 caractères	
Etiquette	**codeEtiquette**	Code (unique) de l'étiquette	Chaîne fixe de 2 caractères	
	libelleEtiquette	Libellé de l'étiquette	Chaîne de 70 caractères	
	codeTypeTVA	Code de type de TVA	Nombre de longueur 22	
ListePrix	**codeListe**	Code (unique) de la liste des prix	Chaîne fixe d'un caractère	Ne doit pas être vide
	libelleListe	Libellé de la liste des prix	Chaîne de 20 caractères	
TarifVente	prixVente	Tarif de prix de vente	Nombre de longueur 22	

GestionStock	**numGestionStock**	N° (unique) de la gestion du stock d'articles	Nombre de longueur 22	
	tailleArticle	Taille possible de l'article	Nombre de longueur 3	∈ {160, 180, 190, 200}
	quantiteStock	Quantité disponible de stock	Nombre de longueur 8	
ListePrix	delaiLivraison	Délai de livraison de l'article	Nombre de longueur 2	
	prixAchat	Prix d'achat réalisé par le fournisseur	Nombre de longueur 22	

III. Schéma logique de la nouvelle base de données:

Tout comme dans la première version, nous avons pu réaliser la traduction du diagramme de classes UML en schéma logique :

ARTICLE(NUMARTICLE, NUMCATEGORIE#, CODETYPE#, NOMARTICLE, REFERENCEINTERNE, CODEBARRE, COUTACHAT)

GESTIONSTOCK(NUMGESTIONSTOCK, NUMARTICLE#, TAILLEARTICLE, QUANTITESTOCK)

CONDITIONVENTE(NUMGESTIONSTOCK#,NUMACTEUR#,DELAILIVRAISON,COUTACHAT)

ACTEUR(NUMACTEUR, CODELISTE#, CODEETIQUETTE#, FONCTIONCONTACT1, NOMACTEUR, ADRRUEACTEUR, ADRCODEPOSTALACTEUR, ADRVILLEACTEUR, ADRPAYSACTEUR, TELEPHONEACTEUR, MAILACTEUR, NOMCONTACT1, TELEPHONECONTACT1, NOMCONTACT2, TELEPHONECONTACT2, FONCTIONCONTACT2)

FOURNISSEUR(NUMACTEUR#, CODELISTE#, CODEETIQUETTE#, FONCTIONCONTACT1, NOMACTEUR, ADRRUEACTEUR, ADRCODEPOSTALACTEUR, ADRVILLEACTEUR, ADRPAYSACTEUR, TELEPHONEACTEUR, MAILACTEUR, NOMCONTACT1, TELEPHONECONTACT1, NOMCONTACT2, TELEPHONECONTACT2, FONCTIONCONTACT2)

CLIENT(NUMACTEUR#, CODELISTE#, CODEETIQUETTE#, FONCTIONCONTACT1, NOMACTEUR, ADRRUEACTEUR, ADRCODEPOSTALACTEUR, ADRVILLEACTEUR, ADRPAYSACTEUR, TELEPHONEACTEUR, MAILACTEUR, NOMCONTACT1, TELEPHONECONTACT1, NOMCONTACT2, TELEPHONECONTACT2, FONCTIONCONTACT2))

COMMANDE(NUMCOMMANDE, NUMCLIENT#, MONTANTFRAIS, MONTANTHT, DATECOMMANDE, MONTANTTTC)

DETAILCOMMANDE(NUMCOMMANDE#, NUMARTICLE#, QUANTITELIVREE, QUANTITECOMMANDE)

TYPEARTICLE(CODETYPE, LIBELLETYPE)

CATEGORIE(NUMCATEGORIE, NUMCATPERE#, LIBELLECATEGORIE)

ETIQUETTE(CODEETIQUETTE, LIBELLEETIQUETTE, CODETYPETVA)

LISTEPRIX(CODELISTE, LIBELLELISTE)

TARIFVENTE(NUMARTICLE#,CODELISTE#, PRIXVENTE)

IV. Version linéaire (avec les règles de traduction):

Chaque relation avec ses multiplicités suit une règle de traduction. Vous trouverez ci-dessous, les différentes règles utilisées ainsi qu'un explicatif d'application de celles-ci :

R1 (classes d'objets) :

TYPEARTICLE(CODETYPE, LIBELLETYPE)

ETIQUETTE(CODEETIQUETTE, LIBELLEETIQUETTE, CODETYPETVA)

LISTEPRIX(CODELISTE, LIBELLELISTE)

GESTIONSTOCK(TAILLEARTICLE, QUANTITESTOCK)

ACTEUR(NUMCLIENT,FONCTIONCONTACT1, NOMCLIENT, ADRRUECLIENT, ADRCODEPOSTALCLIENT, ADRVILLECLIENT, ADRPAYSCLIENT, TELEPHONECLIENT, MAILCLIENT, NOMCONTACT1, TELEPHONECONTACT1, NOMCONTACT2, TELEPHONECONTACT2, FONCTIONCONTACT2)

COMMANDE(NUMCOMMANDE, MONTANTFRAIS, MONTANTHT, DATECOMMANDE, MONTANTTTC)

TYPEARTICLE(CODETYPE, LIBELLETYPE)

CATEGORIE(NUMCATEGORIE, LIBELLECATEGORIE)

ETIQUETTE(CODEETIQUETTE, LIBELLEETIQUETTE, CODETYPETVA)

LISTEPRIX(CODELISTE, LIBELLELISTE)

Chaque classe d'objets devient une relation de même nom dont le schéma comporte un attribut pour chaque attribut de la classe et contient une unique clé primaire.

R2 (association 1,n) :

ARTICLE(NUMARTICLE, NUMCATEGORIE#, CODETYPE#, NOMARTICLE, REFERENCEINTERNE, CODEBARRE, NUMARTICLE, COUTACHAT, PRIXVENTE)

COMMANDE(NUMCOMMANDE, NUMCLIENT#, MONTANTFRAIS, MONTANTHT, DATECOMMANDE, MONTANTTTC)

CATEGORIE(NUMCATEGORIE, NUMCATPERE#, LIBELLECATEGORIE)

ACTEUR(NUMCLIENT, CODELISTE#, CODEETIQUETTE#, FONCTIONCONTACT1, NOMCLIENT, ADRRUECLIENT, ADRCODEPOSTALCLIENT, ADRVILLECLIENT, ADRPAYSCLIENT, TELEPHONECLIENT, MAILCLIENT, NOMCONTACT1, TELEPHONECONTACT1, NOMCONTACT2, TELEPHONECONTACT2, FONCTIONCONTACT2)

GESTIONSTOCK(NUMGESTIONSTOCK, NUMARTICLE#, TAILLEARTICLE, QUANTITESTOCK)

Les attributs formant la clé primaire dans la relation issue de la classe d'objets du côté 1..1, sont dupliqués dans la relation de la classe opposée. Ces attributs forment une clé étrangère.

R3 (association n,m) :

DETAILCOMMANDE(NUMCOMMANDE#, NUMARTICLE#, QUANTITELIVREE, QUANTITECOMMANDE)

TARIFVENTE(NUMARTICLE#, CODELISTE#, PRIXVENTE)

CONDITIONVENTE(NUMGESTIONSTOCK#, NUMACTEUR#, DELAILIVRAISON, COUTACHAT)

D'après la règle de traduction R3, qui permet d'obtenir les classes d'associations (n, m), chaque classe d'associations devient une relation ayant pour clé primaire la concaténation des clés primaires des deux classes d'objets qui la constitue. Les classes DETAILCOMMANDE, TARIFVENTE et CONDITIONVENTE sont ainsi soumises à cette règle.

R5 (héritage sans partition) :

FOURNISSEUR(NUMACTEUR#, CODELISTE#, CODEETIQUETTE#,
FONCTIONCONTACT1, NOMCLIENT, ADRRUECLIENT,
ADRCODEPOSTALCLIENT, ADRVILLECLIENT, ADRPAYSCLIENT,
TELEPHONECLIENT, MAILCLIENT, NOMCONTACT1, TELEPHONECONTACT1,
NOMCONTACT2, TELEPHONECONTACT2, FONCTIONCONTACT2)

CLIENT(NUMACTEUR#, CODELISTE#, CODEETIQUETTE#,
FONCTIONCONTACT1, NOMCLIENT, ADRRUECLIENT,
ADRCODEPOSTALCLIENT, ADRVILLECLIENT, ADRPAYSCLIENT,
TELEPHONECLIENT, MAILCLIENT, NOMCONTACT1, TELEPHONECONTACT1,
NOMCONTACT2, TELEPHONECONTACT2, FONCTIONCONTACT2)

D'après la règle de traduction R5 sur l'héritage sans partition, la clé primaire de la classe mère est dupliquée dans l'ensemble de ses classes filles.
Ainsi, l'ensemble des clés étrangères présentes dans les classes filles auront la même valeur puisque la classe mère pourra correspondre aux classes filles (à contrario d'un héritage avec partition). Ainsi, les classes FOURNISSEUR et CLIENT, étant des classes filles, respectent cette règle.

V. Script SQL-LDD / LMD :

Vous trouverez ci-dessous, le script de cette deuxième version de la nouvelle base de données :

A. Création des tables :

```
DROP TABLE FOURNISSEUR ;
DROP TABLE GESTIONSTOCK ;
DROP TABLE CLIENT ;
DROP TABLE ARTICLE ;
DROP TABLE ACTEUR ;
DROP TABLE CONDITIONVENTE ;

CREATE TABLE FOURNISSEUR
    (
     numActeur NUMBER(22)  NOT NULL,
     nomActeur VARCHAR(50),
     adrRueActeur VARCHAR(128),
     adrCodePostalActeur VARCHAR(10),
     adrVilleActeur VARCHAR(40),
     adrPaysActeur VARCHAR(30),
     telephoneActeur CHAR(12),
     mailActeur VARCHAR(60),
     nomContact1 VARCHAR(50),
     telephoneContact1 CHAR(12),
     fonctionContact1 VARCHAR(20),
     nomContact2 VARCHAR(50),
     telephoneContact2 CHAR(12),
     fonctionContact2 VARCHAR(20),
     CONSTRAINT pk_FOURNISSEUR PRIMARY KEY (numActeur)) ;

CREATE TABLE GESTIONSTOCK
    (
     numGestionStock NUMBER(22)  NOT NULL,
     tailleArticle NUMBER(3)  NOT NULL,
     numArticle NUMBER(22)  NOT NULL,
     quantiteStock NUMBER(8),
     CONSTRAINT pk_GESTIONSTOCK PRIMARY KEY (numGestionStock)
    ) ;
```

```
CREATE TABLE CLIENT
    (
     numActeur NUMBER(22)  NOT NULL,
     nomActeur VARCHAR(50),
     adrRueActeur VARCHAR(128),
     adrCodePostalActeur VARCHAR(10),
     adrVilleActeur VARCHAR(40),
     adrPaysActeur VARCHAR(30),
     telephoneActeur CHAR(12),
     mailActeur VARCHAR(60),
     nomContact1 VARCHAR(50),
     telephoneContact1 CHAR(12),
     fonctionContact1 VARCHAR(20),
     nomContact2 VARCHAR(50),
     telephoneContact2 CHAR(12),
     fonctionContact2 VARCHAR(20),
     CONSTRAINT pk_CLIENT PRIMARY KEY (numActeur)
    ) ;

CREATE TABLE ARTICLE
    (
     numArticle NUMBER(22)  NOT NULL,
     codeType CHAR(1)  NOT NULL,
     numCategorie NUMBER(22)  NOT NULL,
     nomArticle VARCHAR(50),
     referenceInterne VARCHAR(10),
     codeBarre CHAR(13),
     coutAchat NUMBER(22,2),
     CONSTRAINT pk_ARTICLE PRIMARY KEY (numArticle)
    ) ;

CREATE TABLE ACTEUR
    (
     numActeur NUMBER(22)  NOT NULL,
     CODEETIQUETTE CHAR(2)  NOT NULL,
     CODELISTE CHAR(1)  NOT NULL,
     nomActeur VARCHAR(50),
     adrRueActeur VARCHAR(128),
     adrCodePostalActeur VARCHAR(10),
     adrVilleActeur VARCHAR(40),
     adrPaysActeur VARCHAR(30),
     telephoneActeur CHAR(12),
     mailActeur VARCHAR(60),
     nomContact1 VARCHAR(50),
```

```
    telephoneContact1 CHAR(12),
    fonctionContact1 VARCHAR(20),
    nomContact2 VARCHAR(50),
    telephoneContact2 CHAR(12),
    fonctionContact2 VARCHAR(20),
    CONSTRAINT pk_ACTEUR PRIMARY KEY (numActeur)
   ) ;

CREATE TABLE CONDITIONVENTE
   (
    numGestionStock NUMBER(22)  NOT NULL,
    numActeur NUMBER(22)  NOT NULL,
    delaiLivraison NUMBER(2),
    coutAchat NUMBER(22,2),
    CONSTRAINT pk_CONDITIONVENTE PRIMARY KEY
(numGestionArticle, numActeur)
   ) ;
```

B. Ajout des contraintes :

```
ALTER TABLE FOURNISSEUR ADD (
CONSTRAINT fk_fournisseur_acteur
FOREIGN KEY (numActeur)
REFERENCES ACTEUR (numActeur))   ;

ALTER TABLE GESTIONSTOCK ADD (
CONSTRAINT fk_gestionstock_article
FOREIGN KEY (numArticle)
REFERENCES ARTICLE (numArticle))   ;

ALTER TABLE CLIENT ADD (
CONSTRAINT fk_client_acteur
FOREIGN KEY (numActeur)
REFERENCES ACTEUR (numActeur))   ;

ALTER TABLE ARTICLE ADD (
CONSTRAINT fk_article_typeArticle
FOREIGN KEY (codeType)
REFERENCES TYPEARTICLE (codeType))   ;

ALTER TABLE ARTICLE ADD (
CONSTRAINT fk_article_categorie
FOREIGN KEY (numCategorie)
```

```
REFERENCES CATEGORIE (numCategorie))   ;

ALTER TABLE ACTEUR ADD (
CONSTRAINT fk_acteur_etiquette
FOREIGN KEY (codeEtiquette)
REFERENCES ETIQUETTE (codeEtiquette))   ;

ALTER TABLE ACTEUR ADD (
CONSTRAINT fk_acteur_LISTEPRIX
FOREIGN KEY (codeListe)
REFERENCES LISTEPRIX (codeListe))   ;

ALTER TABLE CONDITIONVENTE ADD (
CONSTRAINT fk_conditionvente_gestionStock
FOREIGN KEY (numGestionArticle)
REFERENCES GESTIONSTOCK (numGestionArticle))   ;

ALTER TABLE CONDITIONVENTE ADD (
CONSTRAINT fk_conditionvente_fournisseur
FOREIGN KEY (numActeur)
REFERENCES FOURNISSEUR (numActeur)) ;
```