SAE 1.4

Création d'une base de données

Nolan JACQUEMONT
James SANDALL-ROBERTSON
Cédric-Alexandre PASCAL

Présentation et contexte

Il nous a été demandé de concevoir une base de données pour l'entreprise Becquet. Cette dernière propose un catalogue de produits à la vente auprès de ces clients, qui établissent des commandes à partir de bons de commandes. Elle souhaitait donc pouvoir stocker toutes les données inhérentes aux bons de commandes passés par les clients de la société.

Nous avons donc conçu cette base de données à partir d'un bon de commande vierge. Pour cela, nous avons procédé en plusieurs étapes jusqu'à arriver à une base de données physique et fonctionnelle, que nous avons également pris le temps de tester. Voici donc un rapport complet de notre réalisation, structuré en 5 grandes étapes, listées dans le sommaire ci-dessous.

Sommaire

1) Schéma Conceptuel et Dictionnaire des Données

A partir du bon de commande, nous obtenons le dictionnaire des données suivant :

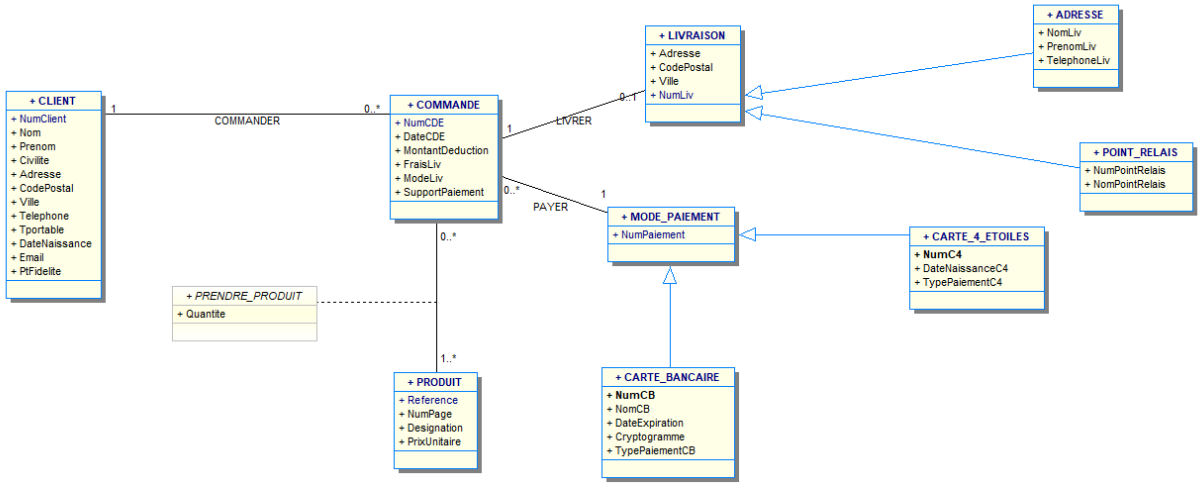
Nom	Description	Type	Contrainte	Calcul
NumClient	Numéro (unique) du Client	Chaîne de 12 caractères		
Civilité	Civilité du client	Chaîne de 4 caractères	Appartient à [MR ; MME; MLLE]	
Nom	Nom du Client	Chaîne de 30 caractères		
Prénom	Prénom du Client	Chaîne de 30 caractères		
Adresse	Libellé de l'adresse postale du client	Chaîne Fixe de 5 caractères		
Ville	Ville du client	Chaîne de 30 caractères		
CodePostal	Code postal du client	Chaîne Fixe de 5 caractères		
Téléphone	Téléphone fixe du client	Chaîne Fixe de 10 caractères		
Tportable	Téléphone portable	Chaîne Fixe de 10 caractères		
DateNaissance	Date de naissance du client	Date	JJ/MM/AAAA	
Email	Adresse mail du client	Chaîne de 30 caractères		
PtFidélité	Capital des points fidélités du client	Entier	>=0	
NumPage	Numéro de page du catalogue où apparaît l'article	Chaîne de 3 caractères		

Désignation	Désignation de l'article	Chaîne de 30 caractères		
Référence	Référence (unique) de l'article	Chaîne Fixe de 6 caractères		
Quantité	Quantitée commandée de l'article	Entier	>0	
NumCde	Numéro (unique) du bon de commande	Entier	> 0	
DateCde	Date saisie du Bon de Commande	Date	JJ/MM/AAAA	
Montant	*Montant sous-total avant déduction*	*Réel*	*>0*	*Quantité x PrixUnitaire*
MontantDéduction	Montant du chèque déduction	Entier	Apparatient {15;30;50;65;100}	
TotalAprèsDéduction	*Montant sous-total après déduction*	*Réel*	*>0*	*Montant - MontantDéduction*
MontantCde	*Montant de la commande*	*Réel*	*>0*	*TotalAprèsDéduction*
FraisTraitement	*Frais de participation du Bon de Commande*	*Réel*	*= 6.99*	*6,99 euros*
ModeLiv	Mode de livraison	Chaîne de 12 caractères	Appartient {PointRelais;Domicile;Express}	
FraisLiv	Frais de livraison	Réel	Appartient {0.0;2.0;9.9}	
SupportPaiement	Support de paiement choisi pour le Bon de Commande	Chaîne Fixe de 6 caractères	C(Cheque;Carte4;CarteB)	
TotalCde	*Montant total du Bon de Commande*	*Réel*	*>0*	*MontantCde + FraisTraitement + FraisLiv*
NumLiv	Numéro (unique) de livraison	Entier	> 0	

Adresse	Libellé de l'adresse postale de livraison (si différente de celle du client)	Chaîne de 30 caractères		
CodePostal	Code postal de livraison (si différente de celle du client)	Chaîne Fixe de 5 caractères		
VilleLiv	Ville de livraison (si différente de celle du client)	Chaîne de 30 caractères		
NumPointRelais	Numéro (unique) du point relais	Chaîne Fixe de 5 caractères		
NomPointRelais	Nom du point relais	Chaîne de 30 caractères		
NomLiv	Nom à la livraison	Chaîne de 30 caractères		
PrénomLiv	Prénom à la livraison	Chaîne de 30 caractères		
TéléphoneLiv	Téléphone à la livraison	Chaîne Fixe de 10 caractères		
NumC4	Numéro (unique) de Carte 4 Etoiles	Chaîne Fixe de 9 caractères		
DateNaissanceC4	Date de naissance du porteur de Carte 4 Etoiles	Date	JJ/MM/AAAA	
TypePaiementC4	Type de paiement de Carte 4 Etoiles choisi pour le bon de commande	Chaîne Fixe de 1 caractère	{C,M} C : Comptant; M: Mensuel	
NumCB	Numéro (unique) de Carte Bancaire	Chaîne Fixe de 16 caractères		
NomCB	Nom du porteur de la Carte Bancaire	Chaîne de 30 caractères		
DateExpiration	Date d'expiration de la	Chaîne Fixe	MM/AA	

	Carte Bancaire	de 5 caractères		
Cryptogramme	Cryptogramme (code à 3 chiffres) de la Carte Bancaire	Chaîne Fixe de 3 caractères		
TypePaiementCB	Type de paiement de Carte Bancaire choisi pour le Bon de Commande	Chaîne Fixe de 1 caractère	E {C;M] C: Comptant; M: Mensuel	
NumPaiement	Numéro de paiement	Entier		

Nous avons décidé d'organiser ces données en 9 classes d'objets différentes, à savoir : Client, Commande, Produit, Livraison, Adresse, Point_Relais, Mode_Paiement, Carte_Bancaire et Carte_4_Etoiles. Voici le diagramme de classes UML que nous avons réalisé à l'aide du logiciel Win'Design :



2) Schéma relationnel

Nous avons ensuite traduit notre schéma conceptuel en schéma relationnel. Pour cela, nous avons appliqué 5 règles de traduction, une après l'autre, afin d'obtenir notre nouveau schéma.

Légende :
Bleu : Ajouts
Attribut : Contrainte de clé primaire
Attribut# : Contrainte de clé étrangère

Règle 1 :

Selon la règle 1, chaque classe d'objet devient une relation avec comme attribut, les attributs de la classe d'objet. On choisit la clé primaire parmi les attributs de la relation qui garantissent l'identification des enregistrements.

Client(NumClient, Nom, Prenom, Civilite, Adresse, CodePostal, Ville, Telephone, Tportable, DateNaissance, Email, PtFidelite)
Commande(NumCDE, DateCDE, MontantDeduction, FraisLiv, SupportPaiement, ModeLiv)
Produit(Reference, NumPage, Designation, PrixUnitaire)
Mode_Paiement(NumPaiement)
Carte_4_Etoiles(NumC4, DateNaissanceC4, TypePaiementC4)
Carte_Bancaire(NumCB, NomCB, DateExpiration, Cryptogramme, TypePaiementCB)
Livraison(NumLiv, Adresse, CodePostal, Ville)
Adresse(NomLiv, PrenomLiv, TelephoneLiv)
Point_Relais(NumPointRelais, NomPointRelais)

Règle 2 :

Selon la règle 2, pour les classes d'associations (1,n) les attributs formant la clé primaire de la relation issue de la classe d'objets du côté 1..1, sont dupliqués dans la relation de la classe opposée et forment une clé étrangère.

Client(NumClient, Nom, Prenom, Civilite, Adresse, CodePostal, Ville, Telephone, Tportable, DateNaissance, Email, PtFidelite)
Commande(NumCDE, DateCDE, MontantDeduction, FraisLiv, SupportPaiement, ModeLiv, NumClient#,NumPaiement#)
Produit(Reference, NumPage, Designation, PrixUnitaire)
Mode_Paiement(NumPaiement)
Carte_4_Etoiles(NumC4, DateNaissanceC4, TypePaiementC4)
Carte_Bancaire(NumCB, NomCB, DateExpiration, Cryptogramme, TypePaiementCB)
Livraison(NumLiv, Adresse, CodePostal, Ville,NumCDE#)
Adresse(NomLiv, PrenomLiv, TelephoneLiv)
Point_Relais(NumPointRelais, NomPointRelais)

Règle 3 :

Selon la règle 3, pour les classes d'associations (n,m) elles deviennent une relation de même nom dont le schéma comporte les attributs dupliqués des clés primaires des relations issues des classes d'objets reliées, ces attributs forment des clés étrangères, un attribut pour chaque attribut de la classe d'associations et la clé primaire est formée par concaténation des clés étrangères.

Client(NumClient, Nom, Prenom, Civilite, Adresse, CodePostal, Ville, Telephone, Tportable, DateNaissance, Email, PtFidelite)
Commande(NumCDE, DateCDE, MontantDeduction, FraisLiv, SupportPaiement, ModeLiv, NumClient#,NumPaiement#)
Produit(Reference, NumPage, Designation, PrixUnitaire)
Mode_Paiement(NumPaiement)
Carte_4_Etoiles(NumC4, DateNaissanceC4, TypePaiementC4)
Carte_Bancaire(NumCB, NomCB, DateExpiration, Cryptogramme, TypePaiementCB)
Livraison(NumLiv, Adresse, CodePostal, Ville,*NumCDE#*)
Adresse(NomLiv, PrenomLiv, TelephoneLiv)
Point_Relais(NumPointRelais, NomPointRelais)
Prendre_Produit(NumCDE#,Reference#, Quantite)

Règle 4 :

Selon la règle 4, si une multiplicité (1,1) est présente, il faut appliquer R2 et ajouter une contrainte d'unicité sur la clé étrangère ajoutée sinon il faut appliquer R3 et ajouter une contrainte d'unicité sur chaque clé étrangère ajoutée.

Client(NumClient, Nom, Prenom, Civilite, Adresse, CodePostal, Ville, Telephone, Tportable, DateNaissance, Email, PtFidelite)
Commande(NumCDE, DateCDE, MontantDeduction, FraisLiv, SupportPaiement, ModeLiv, *NumClient#,NumPaiement#*)
Produit(Reference, NumPage, Designation, PrixUnitaire)
Mode_Paiement(NumPaiement)
Carte_4_Etoiles(NumC4, DateNaissanceC4, TypePaiementC4)
Carte_Bancaire(NumCB, NomCB, DateExpiration, Cryptogramme, TypePaiementCB)
Livraison(NumLiv, Adresse, CodePostal, Ville,*NumCDE#*)
Adresse(NomLiv, PrenomLiv, TelephoneLiv)
Point_Relais(NumPointRelais, NomPointRelais)
Prendre_Produit(*NumCDE#,Reference#*, Quantite)

Règle 5 :
Selon la règle 5, lorsque de l'on traduit l'héritage avec partition, on obtient:

Client(NumClient, Nom, Prenom, Civilite, Adresse, CodePostal, Ville, Telephone, Tportable, DateNaissance, Email, PtFidelite)
Commande(NumCDE, DateCDE, MontantDeduction, FraisLiv, SupportPaiement, ModeLiv, *NumClient#, NumPaiement#*)
Produit(Reference, NumPage, Designation, PrixUnitaire)
Mode_Paiement(NumPaiement)
Carte_4_Etoiles(NumPaiement#, NumC4, DateNaissanceC4, TypePaiementC4)
Carte_Bancaire(NumPaiement#, NumCB, NomCB, DateExpiration, Cryptogramme, TypePaiementCB)
Livraison(NumLiv, Adresse, CodePostal, Ville, *NumCDE#*)
Adresse(NumLiv#, NomLiv, PrenomLiv, TelephoneLiv)
Point_Relais(NumLiv#, NumPointRelais, NomPointRelais)
Prendre_Produit(*NumCDE#,Reference#*, Quantite)

Une fois que nous avons eu notre schéma relationnel, nous avons réalisé les requêtes SQL nécessaires à la création de la base de données.

3) Requêtes SQL-LDD : Créer la base de données

A partir du schéma relationnel, nous avons créé les tables SQL correspondantes. Pour cela, nous avons décidé d'utiliser une nouvelle fois le logiciel Win'Design afin de nous aider en générant le script SQL correspondant, que nous avons légèrement modifié pour mieux correspondre à notre schéma relationnel. Voici le script de création :

```
CREATE TABLE CARTE_BANCAIRE
   (
    NUMPAIEMENT DECIMAL(2)  NOT NULL,
    NUMCB VARCHAR(30)  NOT NULL,
    NOMCB VARCHAR(16)  NULL,
    DATEEXPIRATION CHAR(5) NULL,
    CRYPTOGRAMME CHAR(3)  NULL,
    TYPEPAIEMENTCB CHAR(1)  NULL
,   CONSTRAINT PK_CARTE_BANCAIRE PRIMARY KEY (NUMPAIEMENT)
   ) ;
```

```
CREATE TABLE COMMANDE
   (
    NUMCDE DECIMAL(2)  NOT NULL,
    NUMCLIENT DECIMAL(12)  NOT NULL,
    NUMPAIEMENT DECIMAL(2)  NOT NULL,
    DATECDE DATE  NULL,
    MONTANTDEDUCTION DECIMAL(3)  NULL,
    FRAISLIV DECIMAL(5,2)  NULL,
    MODELIV VARCHAR(30) NULL,
    SUPPORTPAIEMENT VARCHAR(30) NULL
,   CONSTRAINT PK_COMMANDE PRIMARY KEY (NUMCDE)
   ) ;

CREATE  INDEX I_FK_COMMANDE_CLIENT
     ON COMMANDE (NUMCLIENT ASC);
CREATE  INDEX I_FK_COMMANDE_MODE_PAIEMENT
     ON COMMANDE (NUMPAIEMENT ASC);

CREATE TABLE POINT_RELAIS
   (
    NUMLIV DECIMAL(2)  NOT NULL    CHECK (NUMLIV >= 1),
    NUMPOINTRELAIS VARCHAR(30)  NULL,
    NOMPOINTRELAIS VARCHAR(30)  NULL
,   CONSTRAINT PK_POINT_RELAIS PRIMARY KEY (NUMLIV)
   ) ;

CREATE TABLE LIVRAISON
   (
   NUMCDE DECIMAL(2)  NOT NULL,
    ADRESSE VARCHAR(30)  NULL,
    CODEPOSTAL VARCHAR(5)  NULL,
    VILLE VARCHAR(30)  NULL,
    NUMLIV DECIMAL(2)  NOT NULL    CHECK (NUMLIV >= 1)
,   CONSTRAINT PK_LIVRAISON PRIMARY KEY (NUMLIV)
   ) ;

CREATE UNIQUE INDEX I_FK_LIVRAISON_COMMANDE
     ON LIVRAISON (NUMCDE ASC) ;

CREATE TABLE ADRESSE
   (
    NUMLIV DECIMAL(2)  NOT NULL    CHECK (NUMLIV >= 1),
    NOMLIV VARCHAR(30)  NULL,
    PRENOMLIV VARCHAR(30)  NULL,
    TELEPHONELIV VARCHAR(30)  NULL
```

```
,   CONSTRAINT PK_ADRESSE PRIMARY KEY (NUMLIV)
   ) ;

CREATE TABLE CARTE_4_ETOILES
   (
    NUMPAIEMENT DECIMAL(2)  NOT NULL,
    NUMC4 CHAR(9)  NOT NULL,
    DATENAISSANCEC4 DATE  NULL,
    TYPEPAIEMENTC4 CHAR(1)  NULL
,   CONSTRAINT PK_CARTE_4_ETOILES PRIMARY KEY (NUMPAIEMENT)
   ) ;

CREATE TABLE CLIENT
   (
    NUMCLIENT DECIMAL(12)  NOT NULL,
    NOM VARCHAR(30)  NULL,
    PRENOM VARCHAR(30)  NULL,
    CIVILITE VARCHAR(4)  NULL,
    ADRESSE VARCHAR(30)  NULL,
    CODEPOSTAL CHAR(5)  NULL,
    VILLE VARCHAR(30)  NULL,
    TELEPHONE CHAR(10) NULL,
    TPORTABLE CHAR(10) NULL,
    DATENAISSANCE DATE  NULL,
    EMAIL VARCHAR(30) NULL,
    PTFIDELITE DECIMAL  NULL CHECK (PTFIDELITE >= 0)
,   CONSTRAINT PK_CLIENT PRIMARY KEY (NUMCLIENT)
   ) ;

CREATE TABLE PRODUIT
   (
    REFERENCE CHAR(6)  NOT NULL,
    NUMPAGE VARCHAR(3)  NULL,
    DESIGNATION VARCHAR(50)  NULL,
    PRIXUNITAIRE DECIMAL(5,2)  NULL CHECK(PRIXUNITAIRE>0)
,   CONSTRAINT PK_PRODUIT PRIMARY KEY (REFERENCE)
   ) ;

CREATE TABLE MODE_PAIEMENT
   (
    NUMPAIEMENT DECIMAL  NOT NULL
,   CONSTRAINT PK_MODE_PAIEMENT PRIMARY KEY (NUMPAIEMENT)
   ) ;
```

```
CREATE TABLE PRENDRE_PRODUIT
   (
    NUMCDE DECIMAL(2)  NOT NULL,
    REFERENCE CHAR(6)  NOT NULL,
    QUANTITE DECIMAL(2)  NULL CHECK (QUANTITE >= 1)
,   CONSTRAINT PK_PRENDRE_PRODUIT PRIMARY KEY (NUMCDE, REFERENCE)
   ) ;

CREATE  INDEX I_FK_PRENDRE_PRODUIT_COMMANDE
     ON PRENDRE_PRODUIT (NUMCDE ASC)
    ;
CREATE  INDEX I_FK_PRENDRE_PRODUIT_PRODUIT
     ON PRENDRE_PRODUIT (REFERENCE ASC)
    ;

ALTER TABLE CARTE_BANCAIRE ADD (
     CONSTRAINT FK_CARTE_BANCAIRE_MODE_PAIEMEN
          FOREIGN KEY (NUMPAIEMENT)
               REFERENCES MODE_PAIEMENT (NUMPAIEMENT))   ;

ALTER TABLE COMMANDE ADD (
     CONSTRAINT FK_COMMANDE_CLIENT
          FOREIGN KEY (NUMCLIENT)
               REFERENCES CLIENT (NUMCLIENT))   ;

ALTER TABLE COMMANDE ADD (
     CONSTRAINT FK_COMMANDE_MODE_PAIEMENT
          FOREIGN KEY (NUMPAIEMENT)
               REFERENCES MODE_PAIEMENT (NUMPAIEMENT))   ;

ALTER TABLE POINT_RELAIS ADD (
     CONSTRAINT FK_POINT_RELAIS_LIVRAISON
          FOREIGN KEY (NUMLIV)
               REFERENCES LIVRAISON (NUMLIV))   ;

ALTER TABLE LIVRAISON ADD (
     CONSTRAINT FK_LIVRAISON_COMMANDE
          FOREIGN KEY (NUMCDE)
               REFERENCES COMMANDE (NUMCDE))   ;

ALTER TABLE ADRESSE ADD (
     CONSTRAINT FK_ADRESSE_LIVRAISON
          FOREIGN KEY (NUMLIV)
               REFERENCES LIVRAISON (NUMLIV))   ;
```

```
ALTER TABLE CARTE_4_ETOILES ADD (
     CONSTRAINT FK_CARTE_4_ETOILES_MODE_PAIEME
          FOREIGN KEY (NUMPAIEMENT)
               REFERENCES MODE_PAIEMENT (NUMPAIEMENT))   ;
ALTER TABLE PRENDRE_PRODUIT ADD (
     CONSTRAINT FK_PRENDRE_PRODUIT_COMMANDE
          FOREIGN KEY (NUMCDE)
               REFERENCES COMMANDE (NUMCDE))   ;

ALTER TABLE PRENDRE_PRODUIT ADD (
     CONSTRAINT FK_PRENDRE_PRODUIT_PRODUIT
          FOREIGN KEY (REFERENCE)
               REFERENCES PRODUIT (REFERENCE))   ;
```

4) Requêtes SQL-LMD : Insérer des données dans la base

Nous avons ensuite ajouté des données tirées d'exemples de bon de commande, afin de tester au mieux que la base de données est bien fonctionnelle. Voici ci-dessous les requêtes SQL nécessaires à l'ajout de ces données dans la base :

-- Commande 1

```
INSERT INTO CLIENT VALUES(000100, 'Aztakes', 'Hélène', 'MME', 'Av de
Rangueil', '31000', 'Toulouse', '0600000000', null, null, null, 45);
INSERT INTO MODE_PAIEMENT VALUES(1);
INSERT INTO COMMANDE VALUES(1, 000100, 1,
TO_DATE('17/01/2022','dd/mm/yyyy'), 0, 2, 'Cheque', 'Domicile');
INSERT INTO PRODUIT VALUES(471147, null, 'Linge de lit chalet Drap
240x300', 14.95);
INSERT INTO PRODUIT VALUES(471159, null, 'Linge de lit chalet Drap
Housse', 17.45);
INSERT INTO PRODUIT VALUES(471162, null, 'Linge de lit chalet Taie
Traversin', 11.45);
INSERT INTO PRENDRE_PRODUIT VALUES(1, 471147, 1);
INSERT INTO PRENDRE_PRODUIT VALUES(1, 471159, 1);
INSERT INTO PRENDRE_PRODUIT VALUES(1, 471162, 1);
```

-- Commande 2

```
INSERT INTO CLIENT VALUES(000200, 'Assein', 'Marc', 'MR', 'Rue du
chêne', '31000', 'Toulouse', '0600000000', null, '01/12/2001',
'marc@orange.fr', 105);
INSERT INTO MODE_PAIEMENT VALUES(2);
```

```
INSERT INTO COMMANDE VALUES(2, 000200, 2,
TO_DATE('17/01/2022','dd/mm/yyyy'), 15, 0, 'CarteB', 'PointRelais');
INSERT INTO PRODUIT VALUES(905968, null, 'Drap de bain 100x150 gris
perle', 24.67);
INSERT INTO PRODUIT VALUES(905784, null, 'Drap de bain 60x100 gris
perle', 17.17);
INSERT INTO PRENDRE_PRODUIT VALUES(2, 905968, 3);
INSERT INTO PRENDRE_PRODUIT VALUES(2, 905784, 1);
INSERT INTO CARTE_BANCAIRE VALUES(2, '0001000100010001', 'ASSEIN',
'01/24', '001', 'C');
INSERT INTO LIVRAISON VALUES(2, 'Avenue de Revel', '81700',
'Puylaurens', 2);
INSERT INTO POINT_RELAIS VALUES(2, '52035', 'Intermarche contact');
```

-- Commande 3

```
INSERT INTO CLIENT VALUES(000300, 'Terrieur', 'Alex', 'MME', 'Rue de
la Caille', '81000', 'Albi' , '0500000000', '0600000060', null, null,
10);
INSERT INTO MODE_PAIEMENT VALUES(3);
INSERT INTO COMMANDE VALUES(3,  '000300', 3,
TO_DATE('17/01/2022','dd/mm/yyyy'), 50, 9.90, 'Cheque','Express');
INSERT INTO PRODUIT VALUES(950728, null, 'Couette dodo 240x220',
129.90);
INSERT INTO PRODUIT VALUES(950614, null, 'Oreiller ergonomique
60x60', 29.90);
INSERT INTO PRENDRE_PRODUIT VALUES(3, 950728, 1);
INSERT INTO PRENDRE_PRODUIT VALUES(3, 950614, 2);
```

-- Commande 4

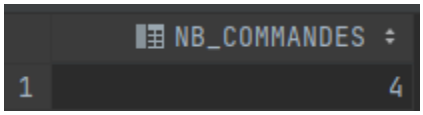
```
INSERT INTO MODE_PAIEMENT VALUES(4);
INSERT INTO COMMANDE VALUES(4, '000200', 4,
TO_DATE('17/01/2022','dd/mm/yyyy'), 100, 0, 'Cheque', 'PointRelais');
INSERT INTO LIVRAISON VALUES(4, 'Avenue de Revel', '81700',
'Puylaurens', 4);
INSERT INTO POINT_RELAIS VALUES(4, 52035, 'Intermarche contact');
INSERT INTO PRENDRE_PRODUIT VALUES(4, 950728, 3);
```

5) Requêtes SQL-LID : Chercher des informations tirées des données

Nous avons enfin voulu tester différentes requêtes de recherche dans les données, qui peuvent notamment se montrer utiles pour l'entreprise, dans des études statistiques par exemple. Chaque requête est présentée avec le code SQL correspondant ainsi que le résultat de sa requête sous forme de capture d'écran.

-- Nombre de commandes passées

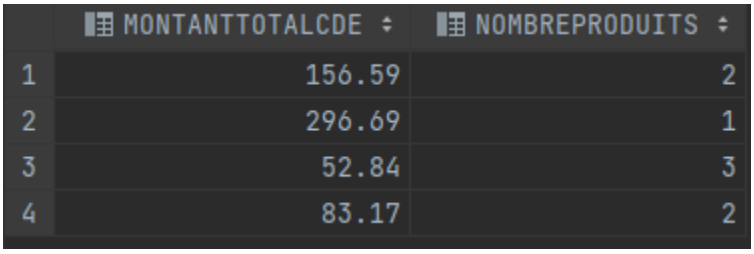
```
SELECT COUNT(NumCde) AS NB_COMMANDES
FROM Commande;
```



	NB_COMMANDES
1	4

-- Montant total des commandes et nombre de produits commandés

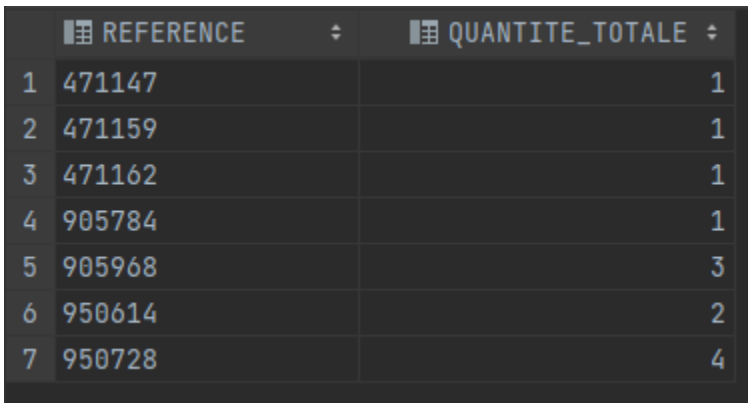
```
SELECT SUM(Pp.Quantite * P.PrixUnitaire) - C.MontantDeduction +
C.FraisLiv + 6.99 AS MontantTotalCde, COUNT(P.Reference) AS
NombreProduits
FROM Commande C, Prendre_Produit Pp, Produit P
WHERE C.NumCde = Pp.NumCde
  AND Pp.Reference = P.Reference
GROUP BY C.NumCde, C.FraisLiv, C.MontantDeduction;
```



	MONTANTTOTALCDE	NOMBREPRODUITS
1	156.59	2
2	296.69	1
3	52.84	3
4	83.17	2

-- Nombre de ventes par produits

```
SELECT P.Reference, SUM(Quantite) as Quantite_Totale
FROM Produit P, Prendre_Produit Pp
WHERE P.Reference = Pp.Reference
GROUP BY P.Reference;
```



	REFERENCE	QUANTITE_TOTALE
1	471147	1
2	471159	1
3	471162	1
4	905784	1
5	905968	3
6	950614	2
7	950728	4

-- Nombre de commandes passées par client

```
SELECT C.NUMCLIENT, COUNT(Cde.NumCde) AS NB_COMMANDES
FROM Client C, Commande Cde
WHERE C.NUMCLIENT = Cde.NUMCLIENT
GROUP BY C.NUMCLIENT;
```

	NUMCLIENT	NB_COMMANDES
1	100	1
2	200	2
3	300	1

-- Nombre de commandes passées et montant total par genre (Mr, Mme, Mlle) des clients

```
SELECT Cl.Civilite, COUNT(DISTINCT(C.NumCde)) AS NombreCde,
       SUM(Pp.Quantite * P.PrixUnitaire) -
SUM(DISTINCT(C.MontantDeduction)) + SUM(DISTINCT(C.FraisLiv)) + 6.99
* COUNT(DISTINCT(C.NumCde)) AS MontantTotal
FROM Commande C, Client Cl, Prendre_Produit Pp, Produit P
WHERE C.NumCde = Pp.NumCde
  AND C.NUMCLIENT = Cl.NUMCLIENT
  AND Pp.Reference = P.Reference
GROUP BY Cl.Civilite;
```

	CIVILITE	NOMBRECDE	MONTANTTOTAL
1	MME	2	209.43
2	MR	2	379.86

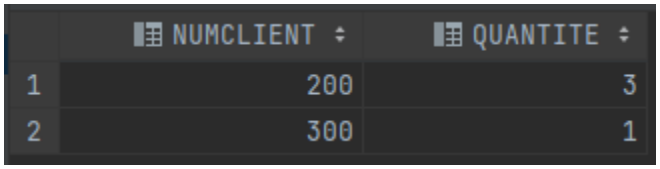
-- Obtenir le prix unitaire du produit n°950728

```
SELECT Reference, PrixUnitaire
FROM Produit
WHERE Reference = '950728';
```

	REFERENCE	PRIXUNITAIRE
1	950728	129.90

-- Obtenir les clients ayant commandé le produit n°950728 avec la quantité commandée

```
SELECT C.NumClient, Pp.Quantite
FROM Commande C, Prendre_Produit Pp, Produit P
WHERE C.NumCde = Pp.NumCde
      AND Pp.Reference = P.Reference
      AND P.Reference = '950728'
GROUP BY C.NumClient, Pp.Quantite;
```



	NUMCLIENT	QUANTITE
1	200	3
2	300	1