SAE 1.04

Création d'une base de données

Mickaël **FERNANDEZ**
Tilian **HURÉ**

4A

Sommaire :

Introduction :

Dans le cadre de la SAÉ 1.04, nous avions pour objectif, par groupe de deux, de concevoir une base de données pour l'entreprise française de vente à distance spécialisée dans le linge de maison et les objets décoratifs, BECQUET. Cette dernière propose un catalogue de produits, en version papier et numérique, à partir duquel les clients peuvent réaliser des commandes en complétant des bons de commande.

Nous devions donc répondre aux besoins de cette société, en mettant en place une base de données dans laquelle pourraient être saisies les données recueillies des bons de commandes.

Nous avons donc commencé par établir un Diagramme de Classes afin de conceptualiser notre travail, avant d'adapter ce dernier en Schéma Logique pour ensuite simplifier son implémentation physique en SQL.

Pour nous assurer du bon fonctionnement de notre base de données, nous avions également à exécuter diverses requêtes SQL à partir de bons de commandes donnés, et vérifier la coïncidence des informations obtenues.

Segmentation :

Nous avons d'abord segmenté les informations des bons de commande afin de définir quelles classes nous allions utiliser pour notre base de données, et comment ces dernières allaient être associées pour correspondre le mieux aux bons.

Classes d'objets :

- Client
- Article
- Paiement (spécialisée, avec partition)
- Livraison (spécialisée, avec partition)
- Bon (relié à Client)

Classes d'association :

- Commander (relié à Article et Bon)

Classes d'objets par spécialisation (héritage) :

- Cheque (relié à Bon et Paiement par hérédité)
- C4 (relié à Bon et Paiement par hérédité)
- CB (relié à Bon et Paiement par hérédité)
- Adresse (relié à Bon et Livraison par hérédité)
- Point_relais (relié à Bon et Livraison par hérédité)
- Adresse_client (relié à Bon et Livraison par hérédité)

Diagramme de Classes UML (conceptuel) :

À partir de la segmentation précédemment vue, nous avons pu élaborer un Diagramme de Classes UML pour conceptualiser la base de données, ses relations et leurs attributs, et ses associations.

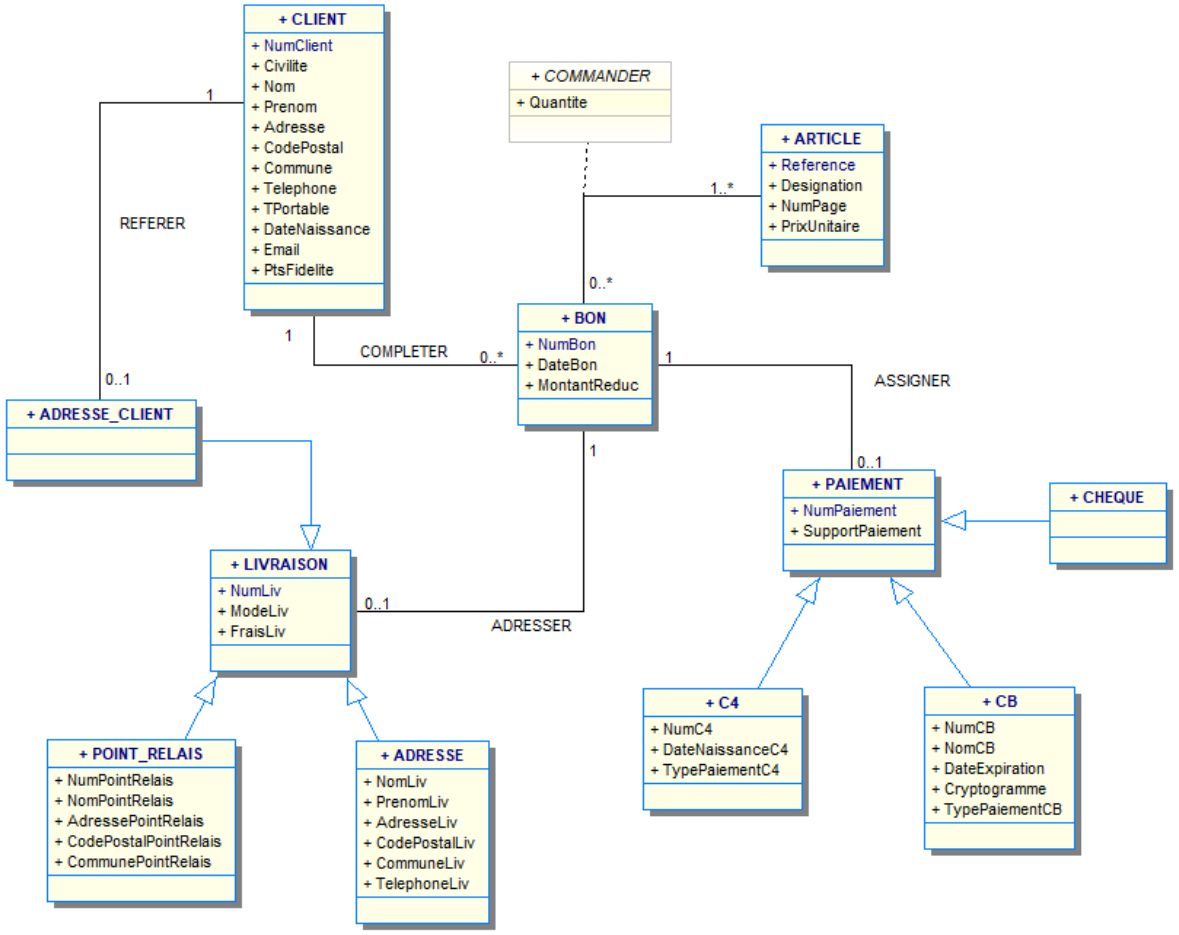
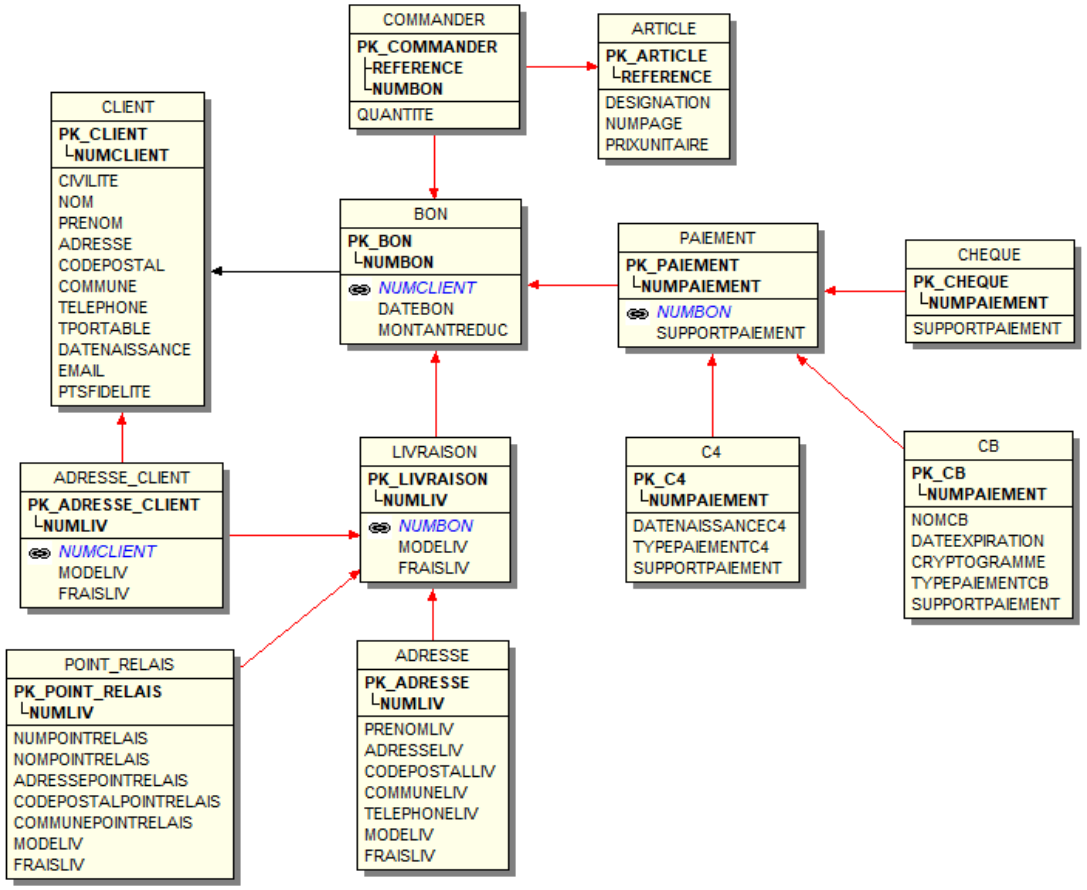


Diagramme de Classes de la base de données "Bon de commande" réaliser avec le logiciel Win'Design.

Schéma Relationnel (logique) :

Nous avons ensuite converti notre diagramme conceptuel en Schéma Logique à l'aide de l'outil Win'Design, ce qui nous a permis par la suite de simplifier l'implémentation physique de notre base de données.



Exemple de Schéma Logique de la base de données "Bon de commande" réalisé avec le logiciel Win'Design.

Version linéaires (avec les règles de traduction) :

Légende :

R1 : Classe d'objet
R2 : Association 1,n
R3 : Association n,m
R4 : Association 1,1
R5 : Héritage
Clé primaire
Clé étrangère#
Contrainte d'unicité

BON (**NumBon**, DateBon, MontantReduc, **NumClient#**)

CLIENT (**NumClient**, Civilite, Nom, Prenom, Adresse, CodePostal, Commune, Telephone, Tportable, DateNaissance, Email, PtsFidelite)

ARTICLE (**Reference**, NumPage, Designation, PrixUnitaire)

COMMANDER (**NumBon#, Reference#**, Quantite)

C4 (**NumPaiement**, NumC4, DateNaissanceC4, TypePaiementC4, SupportPaiement, ***NumBon#***)

CB (**NumPaiement**, NumCB, NomCB, DateExpiration, Cryptogramme, TypePaiementCB, SupportPaiement, ***NumBon#***)

CHEQUE (**NumPaiement**, SupportPaiement, ***NumBon#***)

ADRESSE_CLIENT (**NumLiv**, ModeLiv, FraisLiv, ***NumBon#***, ***NumClient#***)

ADRESSE (**NumLiv**, NomLiv, PrenomLiv, CodePostalLiv, CommuneLiv, TelephoneLiv, ModeLiv, FraisLiv, ***NumBon#***)

POINT_RELAIS (**NumLiv**, NumPointRelais, NomPointRelais, AdressePointRelais, CodePostalPointRelais, CommunePointRelais, ModeLiv, FraisLiv, ***NumBon#***)

Classes d'associations et règles de traduction appliquées :

D'après la règle de traduction **R1**, les relations BON, CLIENT, ARTICLE, C4, CB, CHEQUE, ADRESSE_CLIENT, ADRESSE et POINT_RELAIS, sont des classes d'objets possédant des attributs, l'en d'entre eux étant leur clé primaire.

D'après la règle de traduction **R2**, les classes BON et CLIENT forment une association (1, n), la classe BON ayant pour multiplicité 0..* dans cette association, c'est elle qui prend la clé primaire de la classe CLIENT en tant que clé étrangère.

Pour la règle de traduction **R3**, celle-ci correspond à une traduction des classes d'associations (n, m). Chaque association devient une relation comportant une clé primaire, étant formée par concaténation des clés étrangères des classes d'objets associées. C'est ce qui peut être constaté pour la classe COMMANDER formée à partir de l'association des classes BON et ARTICLE.

Pour la règle de traduction **R4**, celle-ci correspond à une traduction des classes d'associations (1, 1). Ainsi, comme nous avons la présence même d'une multiplicité (1, 1), nous appliquons la règle **R2**, en ajoutant une contrainte d'unicité sur la clé étrangère ajoutée. C'est ainsi ce qui est constaté entre la classe d'objets BON et les classes d'objets C4, CB, CHEQUE, mais également POINT_RELAIS, ADRESSE et ADRESSE_CLIENT qui prennent toutes la clé primaire de BON en tant que clé étrangère.

D'après la règle de traduction de l'héritage **R5**, la classe mère PAIEMENT se spécialise vers ses classes filles C4, CB et CHEQUE, de même pour la classe LIVRAISON qui se spécialise vers ADRESSE, ADRESSE_CLIENT et POINT_RELAIS. Avec partition, les tables mères PAIEMENT et LIVRAISON disparaissent, leurs clés primaires deviennent les clés primaires de leurs tables filles, et leurs attributs sont également dupliqués dans ses dernières.

Requêtes SQL (physique) :

Requêtes SQL-LDD / Création des tables :

```
CREATE TABLE Client (
    NumClient VARCHAR(12),
    Civilite VARCHAR(4),
    Nom VARCHAR(30),
    Prenom VARCHAR(30),
    Adresse VARCHAR(30),
    CodePostal VARCHAR(5),
    Commune VARCHAR(30),
    Telephone CHAR(10),
    Tportable CHAR(10),
    DateNaissance DATE,
    Email VARCHAR(30),
    PtsFidelite DECIMAL,
    CONSTRAINT pk_client PRIMARY KEY (NumClient),
    CONSTRAINT ck_client_civilite CHECK
          (Civilite IN ('MR', 'MLLE', 'MME')),
    CONSTRAINT ck_client_ptsfidelite CHECK (PtsFidelite >= 0)
);

CREATE TABLE Bon (
    NumBon VARCHAR(12),
    DateBon DATE,
    MontantReduc FLOAT,
    NumClient VARCHAR(12),
    CONSTRAINT bon PRIMARY KEY (NumBon),
    CONSTRAINT ck_bon_montantreduc CHECK
          (MontantReduc IN (15, 30, 50, 65, 100)),
    CONSTRAINT ck_bon_ptsfidelite CHECK (MontantReduc >= 0),
    CONSTRAINT fk_bon_numclient FOREIGN KEY (NumClient)
          REFERENCES Client(NumClient)
);
```

```sql
CREATE TABLE Adresse_Client (
    NumLiv VARCHAR(12),
    ModeLiv VARCHAR(10),
    FraisLiv FLOAT,
    NumBon VARCHAR(12) UNIQUE,
    NumClient VARCHAR(12) UNIQUE,
    CONSTRAINT pk_adresse_client PRIMARY KEY (NumLiv),
    CONSTRAINT ck_adresse_client_modeliv CHECK
          (ModeLiv IN ('PointRelais', 'Domicile', 'Express')),
    CONSTRAINT ck_adresse_client_fraisliv CHECK
          (FraisLiv IN (0.0, 2.0, 9.9)),
    CONSTRAINT fk_adresse_client_numbon FOREIGN KEY (NumBon)
          REFERENCES Bon(NumBon),
    CONSTRAINT fk_adresse_client_numclient FOREIGN KEY
          (NumClient) REFERENCES Client(NumClient)
);

CREATE TABLE Adresse (
    NumLiv VARCHAR(12),
    ModeLiv VARCHAR(10),
    FraisLiv FLOAT,
    NumBon VARCHAR(12) UNIQUE,
    NomLiv VARCHAR(30),
    PrenomLiv VARCHAR(30),
    CodePostalLiv CHAR(5),
    CommuneLiv VARCHAR(30),
    TelephoneLiv CHAR(10),
    CONSTRAINT pk_adresse PRIMARY KEY (NumLiv),
    CONSTRAINT ck_adresse_modeliv CHECK
          (ModeLiv IN ('PointRelais', 'Domicile', 'Express')),
    CONSTRAINT ck_adresse_fraisliv CHECK
          (FraisLiv IN (0.0, 2.0, 9.9)),
    CONSTRAINT fk_adresse_numbon FOREIGN KEY (NumBon)
          REFERENCES Bon(NumBon)
);
```

```sql
CREATE TABLE Point_Relais (
    NumLiv VARCHAR(12),
    ModeLiv VARCHAR(11),
    FraisLiv FLOAT,
    NumBon VARCHAR(12) UNIQUE,
    NumPointRelais VARCHAR(12),
    NomPointRelais VARCHAR(30),
    AdressePointRelais VARCHAR(30),
    CodePostalPointRelais CHAR(5),
    CommunePointRelais VARCHAR(30),
    CONSTRAINT pk_point_relais PRIMARY KEY (NumLiv),
    CONSTRAINT ck_point_relais_modeliv CHECK
          (ModeLiv IN ('PointRelais', 'Domicile', 'Express')),
    CONSTRAINT ck_point_relais_fraisliv CHECK
          (FraisLiv IN (0.0, 2.0, 9.9)),
    CONSTRAINT fk_point_relais_numbon FOREIGN KEY (NumBon)
          REFERENCES Bon(NumBon)
);

CREATE TABLE Article (
    Reference CHAR(6),
    NumPage VARCHAR(3),
    Designation VARCHAR(120),
    PrixUnitaire FLOAT NOT NULL,
    CONSTRAINT pk_article PRIMARY KEY (Reference),
    CONSTRAINT ck_article_prixunitaire CHECK
          (PrixUnitaire > 0)
);

CREATE TABLE Commander (
     NumBon VARCHAR(12),
     Reference CHAR(6),
     Quantite DECIMAL NOT NULL,
     CONSTRAINT fk_commander_numbon FOREIGN KEY (NumBon)
          REFERENCES BON(NumBon),
     CONSTRAINT fk_commander_reference FOREIGN KEY (Reference)
          REFERENCES ARTICLE(Reference),
     CONSTRAINT pk_commander PRIMARY KEY (NumBon, Reference),
     CONSTRAINT ck_commander_quantité CHECK (Quantite > 0)
);
```

```
CREATE TABLE C4 (
     NumBon VARCHAR(12),
     NumPaiement VARCHAR(12),
     SupportPaiement VARCHAR(6),
     NumC4 CHAR(9),
     DateNaissanceC4 DATE,
     TypePaiementC4 CHAR(1),
     CONSTRAINT pk_c4 PRIMARY KEY (NumPaiement),
     CONSTRAINT ck_c4_supportpaiement CHECK
          (SupportPaiement = 'C4'),
     CONSTRAINT fk_c4_numbon FOREIGN KEY (NumBon)
          REFERENCES BON(NumBon),
     CONSTRAINT uk_c4_numbon UNIQUE (NumBon)
);

CREATE TABLE CB     (
     NumBon VARCHAR(12),
     NumPaiement VARCHAR(12),
     SupportPaiement VARCHAR(6),
     NumCB CHAR(16),
     NomCB VARCHAR(30),
     DateExpiration CHAR(4),
     Cryptogramme CHAR(3),
     TypePaiementCB CHAR(1),
     CONSTRAINT pk_cb PRIMARY KEY (NumPaiement),
     CONSTRAINT fk_cb_numbon FOREIGN KEY (NumBon)
          REFERENCES BON(NumBon),
     CONSTRAINT uk_cb_numbon UNIQUE (NumBon),
     CONSTRAINT ck_cb_typepaiementcb CHECK
          (TypePaiementCB IN ('C', 'M')),
     CONSTRAINT ck_cb_supportpaiement CHECK
          (SupportPaiement = 'CB')
);

CREATE TABLE Cheque (
     NumBon VARCHAR(12),
     NumPaiement VARCHAR(12),
     SupportPaiement VARCHAR(6),
     CONSTRAINT pk_cheque PRIMARY KEY (NumPaiement),
     CONSTRAINT ck_cheque_supportpaiement CHECK
          (SupportPaiement = 'Chèque'),
     CONSTRAINT fk_cheque_numbon FOREIGN KEY (NumBon)
          REFERENCES BON(NumBon),
     CONSTRAINT uk_cheque_numbon UNIQUE (NumBon)
);
```

Requêtes SQL-LMD / Insertion des données :

```
/*1er bon*/
INSERT INTO Client
(NumClient, Civilite, Nom, Prenom, Adresse, CodePostal,
     Commune, Telephone, PtsFidelite)
VALUES ('000100', 'MME', 'AZTAKES', 'Hélène',
     'Av de Rangueil', '31000', 'Toulouse', '0600000000', 45);

INSERT INTO Bon
(NumBon, DateBon, MontantReduc, NumClient)
VALUES ('1', '', 0, '000100');

INSERT INTO Article
(Reference, Designation, PrixUnitaire)
VALUES ('471147', 'Linge de lit chalet Drap 240x300', 14.95);
INSERT INTO Commander
(NumBon, Reference, Quantite)
VALUES ('1', '471147', 1);

INSERT INTO Article
(Reference, Designation, PrixUnitaire)
VALUES ('471159', 'Linge de lit chalet Drap housse', 17.45);
INSERT INTO Commander
(NumBon, Reference, Quantite)
VALUES ('1', '471159', 1);

INSERT INTO Article
(Reference, Designation, PrixUnitaire)
VALUES ('471162', 'Linge de lit chalet Taie traversin',
     11.45);
INSERT INTO Commander
(NumBon, Reference, Quantite)
VALUES ('1', '471162', 1);

INSERT INTO Cheque
(NumPaiement, SupportPaiement, NumBon)
VALUES ('1', 'Chèque', '1');

INSERT INTO Adresse_Client
(NumLiv, ModeLiv, FraisLiv, NumBon, NumClient)
VALUES ('1', 'Domicile', 2.0, '1', '000100');
```

```
/*2e bon*/
INSERT INTO Client
(NumClient, Civilite, Nom, Prenom, Adresse, CodePostal,
     Commune, Telephone, DateNaissance, Email, PtsFidelite)
VALUES ('000200', 'MR', 'ASSEIN', 'Marc', 'rue du chêne',
     '31000', 'TOULOUSE', '600000006', '01/12/2001',
     'marc@orange.fr', 105);

INSERT INTO Bon
(NumBon, DateBon, MontantReduc, NumClient)
VALUES ('2', '', 15.0, '000200');

INSERT INTO Article
(Reference, Designation, PrixUnitaire)
VALUES ('905968', 'Drap de bain 100x150 grisperle', 24.67);
INSERT INTO Commander
(NumBon, Reference, Quantite)
VALUES ('2', '905968', 3);

INSERT INTO Article
(Reference, Designation, PrixUnitaire)
VALUES ('905784', 'Tapis de bain 60x100 grisperle', 17.17);
INSERT INTO Commander
(NumBon, Reference, Quantite)
VALUES ('2', '905784', 1);

INSERT INTO CB
(NumPaiement, NumCB, NomCB, DateExpiration, Cryptogramme,
     TypePaiementCB, SupportPaiement, NumBon)
VALUES ('2', '0001000100010001', 'ASSEIN', '01/24', '001',
     'C', 'CB', '2');

INSERT INTO Point_Relais
(NumLiv, NumPointRelais, NomPointRelais, AdressePointRelais,
     CodePostalPointRelais, CommunePointRelais, ModeLiv,
     FraisLiv, NumBon)
VALUES ('2', '52035', 'INTERMARCHE CONTACT',
     'AVENUE DE REVEL', '81700', 'PUYLAURENS', 'PointRelais',
     0.0, '1');
```

```sql
/*3e bon*/
INSERT INTO Client
(NumClient, Civilite, Nom, Prenom, Adresse, CodePostal,
     Commune, Telephone, TPortable, PtsFidelite)
VALUES ('000300', 'MME', 'TERRIEUR', 'Alex',
     'rue de la caille', '81000', 'ALBI', '0500000000',
     '0600000060', 10);

INSERT INTO Bon
(NumBon, DateBon, MontantReduc, NumClient)
VALUES ('3', '', 50, '000300');

INSERT INTO Article
(Reference, Designation, PrixUnitaire)
VALUES ('950728', 'Couette Dodo 240x220', 129.90);
INSERT INTO Commander
(NumBon, Reference, Quantite)
VALUES ('3', '950728', 1);

INSERT INTO Article
(Reference, Designation, PrixUnitaire)
VALUES ('950614', 'Oreiller ergonomique 60x60', 29.90);
INSERT INTO Commander
(NumBon, Reference, Quantite)
VALUES ('3', '950614', 2);

INSERT INTO Cheque
(NumPaiement, SupportPaiement, NumBon)
VALUES ('3', 'Chèque', '3');

INSERT INTO Adresse_Client
(NumLiv, ModeLiv, FraisLiv, NumBon, NumClient)
VALUES ('3', 'Express', 9.9, '3', '000300');
```

```sql
/*4e bon*/
INSERT INTO Bon
(NumBon, DateBon, MontantReduc, NumClient)
VALUES ('4', '', 100, '000200');

INSERT INTO Commander
(NumBon, Reference, Quantite)
VALUES ('4', '950728', 3);

INSERT INTO Cheque
(NumPaiement, SupportPaiement, NumBon)
VALUES ('4', 'Chèque', '4');

INSERT INTO Point_Relais
(NumLiv, ModeLiv, FraisLiv, NumBon, NumPointRelais,
     NomPointRelais, AdressePointRelais,
     CodePostalPointRelais, CommunePointRelais)
VALUES ('4', 'PointRelais', 0.0, '4', '52035',
     'INTERMARCHE CONTACT', 'AVENUE DE REVEL', '81700',
     'PUYLAURENS');
```

Requêtes SQL-LID / Recueil et calcul de données :

(Les résultats en noir et en gras sont donnés en dessous des requêtes SQL correspondantes)

```
/*Nombre de commandes passées*/
SELECT COUNT(NumBon) AS Nb_Commandes
FROM Bon;
```

```
NB_COMMANDES
------------
           4
```

```
/*Montant total et montant des commandes*/
SELECT Bon.NumBon,
     SUM((Commander.Quantite * Article.PrixUnitaire))
     - Bon.MontantReduc AS Montant_Bon,
     SUM(Commander.Quantite * Article.PrixUnitaire) + 6.99
     + COALESCE(Adresse.FraisLiv, 0)
     + COALESCE(Adresse_Client.FraisLiv, 0)
     + COALESCE(Point_Relais.FraisLiv, 0)
     - Bon.MontantReduc AS Montant_Total_Bon
FROM Bon, Article, Commander, Adresse, Adresse_Client,
     Point_Relais
WHERE Commander.NumBon = Bon.NumBon
AND Commander.Reference = Article.Reference
AND Adresse.NumBon(+) = Bon.NumBon
AND Adresse_Client.NumBon(+) = Bon.NumBon
AND Point_Relais.NumBon(+) = Bon.NumBon
GROUP BY Bon.NumBon, Bon.MontantReduc, Adresse.FraisLiv,
     Adresse_Client.FraisLiv, Point_Relais.FraisLiv
ORDER BY Bon.NumBon ASC;
```

NUMBON	MONTANT_BON	MONTANT_TOTAL_BON
1	43,85	52,84
2	76,18	83,17
3	139,7	156,59
4	289,7	296,69

```
/*Nombre de ventes par produits*/
SELECT Article.Designation,
     SUM(DISTINCT Commander.Quantite) AS Quantité
FROM Article, Commander
WHERE Commander.Reference = Article.Reference
GROUP BY Article.Designation;
```

DESIGNATION	QUANTITÉ
Oreiller ergonomique 60x60	2
Linge de lit chalet Drap 240x300	1
Linge de lit chalet Taie traversin	1
Linge de lit chalet Drap housse	1
Drap de bain 100x150 grisperle	3
Tapis de bain 60x100 grisperle	1
Couette Dodo 240x220	4

7 lignes sélectionnées.

```
/*Nombre de commandes passées par clients*/
SELECT Client.Prenom, Client.Nom,
     COUNT(Bon.NumBon) AS Nb_Commandes
FROM Bon, Client
WHERE Bon.NumClient = Bon.NumClient
GROUP BY Client.Prenom, Client.Nom;
```

PRENOM	NOM	NB_COMMANDES
Marc	ASSEIN	4
Alex	TERRIEUR	4
Hélène	AZTAKES	4

```
/*Nombre de commandes passées et montant total par genre
     (civilité) des clients*/
SELECT DISTINCT Client.Civilite AS Genre,
     COUNT(DISTINCT Bon.NumBon) AS Nb_Commandes,
     SUM(Article.PrixUnitaire * Commander.Quantite)
     + SUM(DISTINCT COALESCE(Adresse.FraisLiv, 0))
     + SUM(DISTINCT COALESCE(Adresse_Client.FraisLiv, 0))
     + SUM(DISTINCT COALESCE(Point_Relais.FraisLiv, 0))
     + 6.99*COUNT(DISTINCT(Bon.NumBon))
     - SUM(DISTINCT Bon.MontantReduc) AS Montant_Total
FROM Client, Bon, Article, Commander, Adresse_Client, Adresse,
     Point_Relais
WHERE Commander.NumBon = Bon.NumBon
AND Client.NumClient = Bon.NumClient
AND Article.Reference = Commander.Reference
AND Adresse_Client.NumBon(+) = Bon.NumBon
AND Point_Relais.NumBon(+) = Bon.NumBon
AND Adresse.NumBon(+) = Bon.NumBon
GROUP BY Client.Civilite;
```

```
GENRE NB_COMMANDES MONTANT_TOTAL
----  ------------ -------------
MME              2        209,43
MR               2        379,86
```

```
/*Obtenir le prix unitaire du produit n°950728*/
SELECT PrixUnitaire
FROM Article
WHERE Reference = '950728';
```

```
PRIXUNITAIRE
------------
       129,9
```

```
/*Obtenir les clients ayant commandé le produit n°950728 avec
     la quantité commandée*/
SELECT Client.Prenom, Client.Nom,
     SUM(DISTINCT Commander.Quantite) AS Quantité
FROM Client, Commander, Bon
WHERE Commander.Reference = '950728'
AND Bon.NumBon = Commander.NumBon
AND Client.NumClient = Bon.NumClient
GROUP BY Client.Prenom, Client.Nom;
```

```
PRENOM NOM      QUANTITÉ
------ -------- --------
Marc   ASSEIN          3
Alex   TERRIEUR        1
```

Conclusion :

Cette SAÉ nous a permis de confirmer nos compétences dans le domaine des bases de données en mettant en pratique de manière autonome, les nouvelles notions acquises en partant d'un schéma conceptuel à une implémentation physique en SQL. Nous proposons une solution simple, mais adaptée aux besoins exprimés.

La coopération a été au cœur du projet, nous avons en effet conceptualisé la base de données ensemble jusqu'à son état logique, à partir duquel nous nous sommes réparti la charge de travail pour procéder chacun à une implémentation physique plus efficace, tout en nous entraidant lors des difficultés rencontrées.

Nous avons pu acquérir davantage d'expérience, mais aussi une vision plus claire de la conception d'une base de données réelle à partir de besoins clairement exprimés. Tout cela en alliant les compétences de chacun, indispensables pour la progression du travail en groupe.