ÉCOLE NATIONALE D'INGÉNIEURS DE BREST

UNION CYCLISTE INTERNATIONALE

ENIB semestre S3P :
Informatique - Bases de données relationnelles

Union Cycliste Internationale : Gestion des coureurs appartenants aux différentes équipes de la fédération.

Florian Pasco S3P-A

Version 2.0 : 10 mai 2022

Table des matières

Table des figures

1 Cahier des charges initial

L'Union Cycliste Internationale fait appel à nos services en Informatique pour mettre en place un Système d'Information pour gérer les coureurs appartenant aux différentes équipes de la fédération à partir du cahier des charges suivant : "Les coureurs sont des personnes dont on connaît le nom et le prénom, la taille, la date de naissance et l'équipe à laquelle ils appartiennent. Une équipe est identifiée par son nom, elle possède un budget, un directeur sportif qui est une personne dont on connaît le nom, le prénom et la date de naissance. Elle est financée par des sponsors qui peuvent varier selon les années et dont on connaît le nom, l'adresse et le domaine d'activité. Une course correspond à un nom de course (ex. «Tour de France»), on en connaît la distance totale à parcourir. Elle peut comporter une ou plusieurs étapes, dont on connaît le numéro d'ordre (ex. «3è étape»), la date, le type (ex. «Contre la montre individuel»), la ville de départ et celle d'arrivée. Pour chaque coureur ayant participé à une étape d'une course, on connaît le classement qu'il a obtenu lors de cette étape. Pour chaque course, on connaît le vainqueur final et l'équipe à laquelle il appartient. Pour chaque course, les équipes emploient des soigneurs qui sont des personnes, dont on connaît le nom, le prénom, la date de naissance et la nationalité. On note aussi, à chaque étape, quelle dose de quel(s) produit(s) a administré un soigneur à un coureur. Un produit est identifié par un numéro de produit, a un nom, une indication (ex.«douleur musculaire»), une contre-indication (ex. «ne pas administrer en dessous de 20 ans») et uneposologie (ex. «1 comprimé par jour »). Dans cette base de donnée de production, seules les informations courantes (concernant l'édition en cours) de la course, des coureurs, des équipes, etc. sont stockées."

2 Identification des entités

À partir de ce cahier des charges, on relève le nom des entités les plus importantes et les informations qui y sont liées.

- **un coureur** est **une personne** dont on connaît le nom et le prénom, la taille, la date de naissance et **l'équipe** à laquelle il appartient.
- **une équipe** est identifiée par son nom, elle possède un budget, a **un directeur sportif** et est financée par **des sponsors**.
- **un directeur sportif** qui est **une personne** dont on connaît le nom, le prénom et la date de naissance.
- **un sponsor** est identifié par son nom, son adresse et son domaine d'activité.
- **une course** correspond à un nom de course et on en connaît la distance totale à parcourir.
- **une étape** a un numéro d'ordre (ex. «3è étape»), une date, un type, une ville de départ et une d'arrivée. un classement donne la place d'un coureur lors de l'étape d'une course.
- **un soigneur** est **une personne**, dont on connaît le nom, le prénom, la date de naissance et la nationalité.
- **un produit** est identifié par un numéro de produit et a un nom, une indication, une contre-indication et une posologie.
- **une classement** obtenu par **un coureur** lors d'une étape est connue.

3 Identification des associations

Les associations relient les entités entre elles. On rélève les associations à partir de phrases (Sujet-Verbe-Complément) du cahier des charges. Le Verbe représente l'association entre deux entités (Sujet, Complément).

3.1 Entité-Association

À partir du cahier des charges de l'Union Cycliste Internationale, on retient les phrases importantes reliant les entités par des verbes :

— une personne **est** un coureur.
— une personne **est** une personneun directeur sportif.
— une personne **est** un soigneur.
— un coureur **fait partie d'**une équipe.
— un soigneur **soigne** un coureur.
— une étape **est gagnée par** un coureur.
— un classement **donne la place d'**un coureur.
— un classement **est réalisé lors d'**une étape.
— un course **comporte** des étapes.
— une étape **est gagnée par** une équipe.
— un soigneur **soigne lors d'**une course.
— un soigneur **soigne à la demande de** une équipe.

3.2 Modélisation UML (version 1.1)

A partir de ces associations entre entités on peut proposer une première version de la structuration d'une base de données à l'aide du formalisme UML.

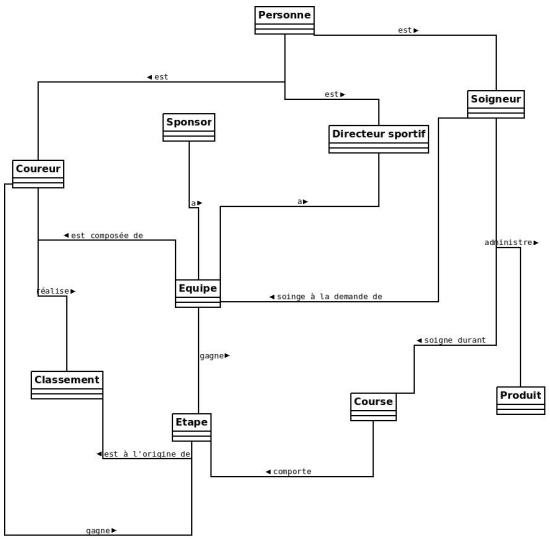


FIGURE 1 – Modèle : Entité-Association

4 Identification des cardinalités

A partir de ces phrases du cahier des charges ont peut également identifier la cardinalité des associations (un-un, un-plusieurs, plusieurs-plusieurs, etc ...). Par exemple, un soin est donné lors d'une ou plusieurs courses, une équipe est financée par au moins un sponsors . . .

4.1 Association-Cardinalité

L'identification des cardinalités est très importante afin de pouvoir gérer les contraintes référentielles entre les entités. Ces cardinalités permettront de savoir sur quelle entité (table) il sera nécessaire de placer une référence (clé étrangère) sur l'autre entité (table) en association ou s'il est nécessaire de définir une nouvelle entité pour relier les deux entités, ce qui sera le cas pour les associations plusieurs-à-plusieurs (many-to-many).

- **une** personne **peut être (mais pas nécessairement) un** coureur, directeur sportif ou soigneur.
- **une** équipe est composée de **plusieurs** coureurs.
- **un** coureur gagne **(ou pas) des** étapes.
- **un** coureur réalise **(ou pas) des** classements.
- **une** étape est à l'origine d'**au moins un** classement.
- **une** course comporte **au moins une** étape.
- **une** équipe gagne **(ou pas) des** étapes.
- **plusieurs** soigneurs soignent lors de **plusieurs** courses.
- **plusieurs** soigneurs soignent à la demande de **plusieurs** équipes.
- **plusieurs** soigneurs administrent **plusieurs** produits.
- **une** équipe a **nécessairement un** directeur sportif.
- **une** équipe a **au moins un** sponsor.

4.2 Modélisation UML (version 1.2)

À partir des cardinalités identifiées précédemment, on peut proposer une deuxième version du modèle UML.

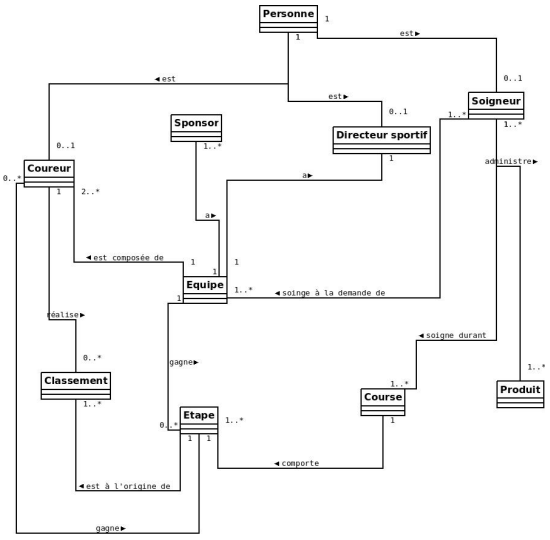


FIGURE 2 – Modèle : Cardinalité d'associations

4.3 Modélisation UML (version 1.3)

Le modèle précédent UML fait appaître différentes liaisons Plusieurs-à-Plusieurs que nous devrons systématiquement décomposer en deux associations Un-à-Pusieurs. Nous pouvons donc exprimer plus précisément cette décomposition en intégrant des tables (entités) associatives et en reformulant les associations entre ces nouvelles tables et les deux autres entités.

D'abord :

— **plusieurs** soigneurs soignent lors de **plusieurs** courses.

devient

— **un même** soin est réalisée lors de **plusieurs** courses.

— **plusieurs** soigneurs donnent **un** soin.

Ensuite :

— **plusieurs** soigneurs soignent à la demande de **plusieurs** équipes.

devient

— **Plusieurs** soigneurs donnent **un** soin.

— **Plusieurs** équipes reçoivent **un** soin.

Pour finir :

— **plusieurs** soigneurs administrent **plusieurs** produits.

devient

— **plusieurs** soigneurs administrent **une** dose.

— **plusieurs** produits sont utilisés avec **la même** dose.

Nous obtiendrons alors la modélisation suivante :

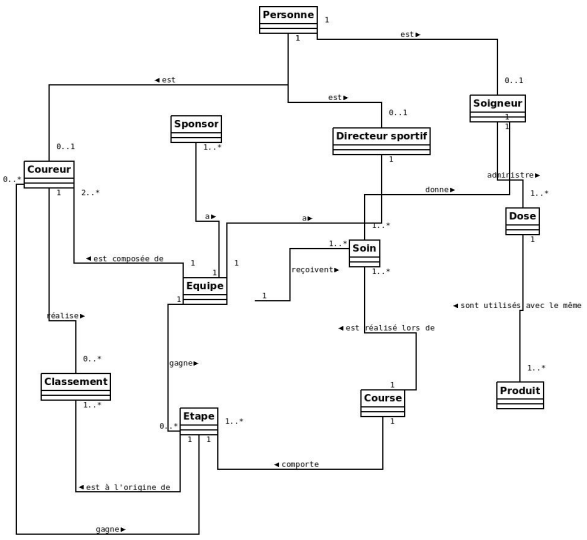


FIGURE 3 – Modèle : Cardinalité d'associations

5 Entité-Attributs

Après avoir identifié les entités importantes, à partir du cahier des charges on doit pouvoir relever les informations les caractérisant.

5.1 Identification des attributs

Dans notre cahier des charges on peut retenir les caractéristiques suivantes :
— une personne est identifié par un **nom** et un **prénom** et sa **date de naissance**.
— un coureur est identifié par une **taille**.
— une équipe est identifiée par son **nom** et a un **budget**.
— un sponsor a **nom**, une **adresse** et un **domaine d'activité**.
— une course a un **nom** et une **distance totale**.
— une étape a un **numéro d'ordre**, une **date**, un **type**, une **ville de départ** et une **d'arrivée**.
— un classement est une **place** obtenu.
— un soigneur est identifié par une **nationalité**.
— une dose est une **quantité** de produit.
— un produit a un **numéro**, un **nom**, une **indication**, une **contre-indication** et une **posologie**.

5.2 Modélisation UML (version 1.4)

On peut représenter les attributs des entités dans une troisème version du modèle UML.

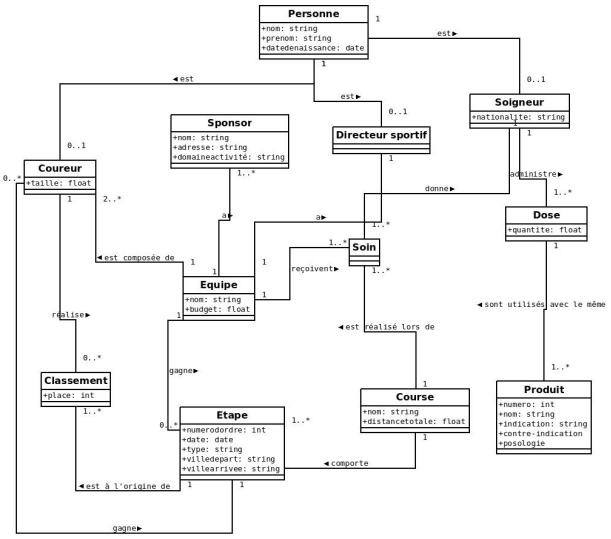


FIGURE 4 – Modèle : Attributs

Cette version peut-être considérée comme étant une première version que l'on peut proposer au client par rapport au cahier des charges initial.

6 de UML à SQL

Le passage d'une représentation à l'aide du formalisme UML à une structuration sous forme de base de données relationnelles nécessite de faire apparaître sur les entités des attributs qui représenteront les clés primaires de tables et les clés étrangères qui référenceront les clés primaires des entités (tables) en association.

6.1 Modélisation UML (version 1.4)

A partir des cardinalités des associations on peut représenter sur le modèle UML précédent les clés primaires et étrangères personne.id, soigneur.personne_id ...).

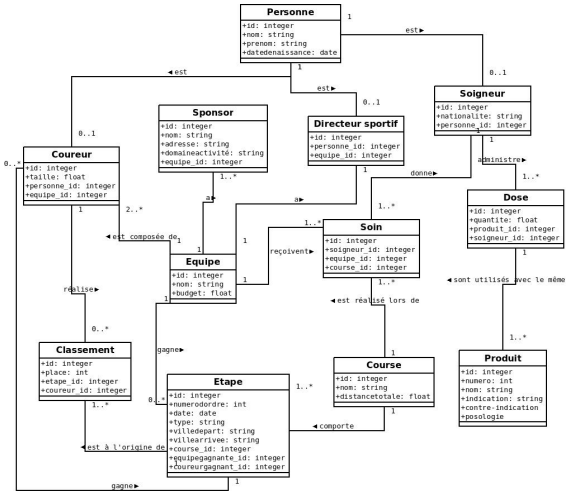


FIGURE 5 – Modèle : Attributs clés

6.2 Structuration de la base de données

A partir de ce schéma UML, on pourra mettre en œuvre la structuration de la base de données en SQL.

- personne (id, nom, prenom, datedenaissance)
- coureur (id, taille, #personne_id, #equipe_id)
- sponsor (id, nom, adresse, domaineactivité, #equipe_id)
- directeur sportif (id, #personne_id, #equipe_id)
- soigneur (id, nationalite, #personne_id)
- equipe (id, nom, budget, #ds_id)
- soin (#soigneur_id, #equipe_id, #course_id)
- dose (#produit_id, #soinguer_id, quantite)
- classement (id, place, #etape_id, #coureur_id)
- etape (id, numerodordre, date, type, villedepart, villearrivee, #course_id, #equipegagnante_id, #coureurgagnante_id)
- course (id, quantite, #produit_id, #soinguer_id)
- produit (id, quantite, #produit_id, #soinguer_id)

Dans cette représentation, les clés primaires sont soulignées, les clés étrangères sont "hashtaguées". En SQLite il n'est pas nécessaire (et même non-recommandé !) de définir un identifiant (PRIMARY KEY) incrémenté automatiquement (AUTOINCREMENT), en effet un identifiant (rowid) est accessible directement.

On peux ainsi réaliser la structuration de la base de donnée.

```
--creation de coureur
CREATE TABLE coureur (
  taille FLOAT,
  personne_id INTEGER,
  equipe_id INTEGER,
  FOREIGN KEY ( personne_id ) REFERENCES personnes,
  FOREIGN KEY ( equipe_id ) REFERENCES equipe
);

-- postes primary key : rowid
SELECT rowid from coureur ;
rowid
-- ---
1
2
...
```

On trouvera en annexe la création de l'ensemble des tables du modèle de données en SQL.

6.3 Insertion d'informations dans la base de données

A partir de cette structuration on peut alimenter la base de données en insérant des informations et fasisant des mises à jour à l'aide de commandes SQL (INSERT INTO ... VALUES ... ; UPDATE ... SET ... WHERE ... ;)

```sql
-- insertion de personne
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Bernard' , 'Hinault' , '14-11-1954')
    ;
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Thomas' , 'Voeckler' , '22-06-1979')
    ;
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Bryan' , 'Coquard' , '25-04-1992');
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Morgan' , 'Lamoisson' , '07-09-1988'
    );
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Jean' , 'Dupont' , '04-03-1960');

-- insertion de equipe
INSERT INTO equipe ( nom, budget ) VALUES ( 'Vendée U', 80000.15);
INSERT INTO equipe ( nom, budget ) VALUES ( 'Gitane-Campagnolo', 40010.25);

-- insertion de coureur
INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
    1.70,
    (SELECT rowid FROM personne WHERE prenom="Thomas" AND nom="Voeckler"),
    (SELECT rowid FROM equipe WHERE nom="Vendée U"));

INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
    1.75,
    (SELECT rowid FROM personne WHERE prenom="Bryan" AND nom="Coquard"),
    (SELECT rowid FROM equipe WHERE nom="Vendée U"));

INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
    1.80,
    (SELECT rowid FROM personne WHERE prenom="Bernard" AND nom="Hinault"),
    (SELECT rowid FROM equipe WHERE nom="Gitane-Campagnolo"));

-- insertion de directeur sportif
INSERT INTO directeur_sportif ( personne_id , equipe_id ) VALUES (
  (SELECT rowid FROM personne WHERE prenom = 'Morgan' AND nom = 'Lamoisson'),
  (SELECT rowid FROM equipe WHERE nom = 'Vendée U'));

-- insertion de soigneur
INSERT INTO soigneur ( nationalite, personne_id) VALUES (
  'Francais',
  (SELECT rowid FROM personne WHERE prenom = 'Jean' AND nom = 'Dupont'));

-- insertion de sponsor
INSERT INTO sponsor ( nom, adresse, domaineactivite, equipe_id ) VALUES ( 'Système U',
    '20 RUE D ARCUEIL PARC TERTIAIRE SILIC BATIMENT MO 94150 RUNGIS',
    'Grande distribution',
    (SELECT rowid FROM equipe WHERE nom = 'Vendée U'));

INSERT INTO sponsor ( nom, adresse, domaineactivite, equipe_id ) VALUES ( 'Département Vendée',
    '2 Av. Gordon Bennett, 75016 Paris',
    'Département',
    (SELECT rowid FROM equipe WHERE nom = 'Vendée U'));

-- insertion de course
INSERT INTO course ( nom, distancetotale ) VALUES ( 'Tour de France',  3000);
INSERT INTO course ( nom, distancetotale ) VALUES ( 'Tour d Espagne',  1000);

-- insertion de soin
INSERT INTO soin ( soigneur_id, equipe_id, course_id ) VALUES (
  (SELECT s.rowid FROM personne p CROSS JOIN soigneur s WHERE prenom = 'Jean' AND nom = 'Dupont' AND s
    .personne_id = p.rowid),
  (SELECT rowid FROM equipe WHERE nom = 'Vendée U'),
  (SELECT rowid FROM course WHERE nom = 'Tour de France'));

-- insertion de produit
INSERT INTO produit ( numero, nom, indication, contre_indication, posologie ) VALUES ( 1547,
  'Salbutamol',
  'douleur musculaire',
  'ne pas administrer en dessous de 20 ans',
  '1 comprimé par jour');

-- insertion de dose
INSERT INTO dose ( quantite, produit_id, soigneur_id ) VALUES ( '2.85',
  (SELECT rowid FROM produit WHERE nom = 'Salbutamol'),
```

```
  (SELECT s.rowid FROM personne p CROSS JOIN soigneur s WHERE prenom = 'Jean' AND nom = 'Dupont' AND s
    .personne_id = p.rowid));

-- insertion de etape
INSERT INTO etape ( numeroordre, date, type, villedepart, villearivee, course_id, equipegagnante_id,
    coureurgagnant_id ) VALUES (
    '1re étape',
    '15-01-2015',
    'Contre la montre individuel',
    'Lorient',
    'Brest',
    (SELECT rowid FROM course WHERE nom = 'Tour de France'),
    (SELECT rowid FROM equipe WHERE nom = 'Vendée U'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO etape ( numeroordre, date, type, villedepart, villearivee, course_id, equipegagnante_id,
    coureurgagnant_id ) VALUES (
    '2nd étape',
    '16-02-2015',
    'Course en ligne',
    'Brest',
    'Lannion',
    (SELECT rowid FROM course WHERE nom = 'Tour de France'),
    NULL,
    NULL);

INSERT INTO etape ( numeroordre, date, type, villedepart, villearivee, course_id, equipegagnante_id,
    coureurgagnant_id ) VALUES (
    '3ème étape',
    '17-02-2015',
    'Course en ligne',
    'Lannion',
    'Rennes',
    (SELECT rowid FROM course WHERE nom = 'Tour de France'),
    NULL,
    NULL);

INSERT INTO etape ( numeroordre, date, type, villedepart, villearivee, course_id, equipegagnante_id,
    coureurgagnant_id ) VALUES (
    '4ème étape',
    '18-02-2015',
    'Course en ligne',
    'Rennes',
    'Nantes',
    (SELECT rowid FROM course WHERE nom = 'Tour de France'),
    NULL,
    NULL);

-- insertion de classement
INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '1er',
    (SELECT rowid FROM etape WHERE numeroordre = '1re étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '3ème',
    (SELECT rowid FROM etape WHERE numeroordre = '3ème étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '2nd',
    (SELECT rowid FROM etape WHERE numeroordre = '2nd étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '4ème',
    (SELECT rowid FROM etape WHERE numeroordre = '4ème étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '2nd',
    (SELECT rowid FROM etape WHERE numeroordre = '1re étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
```

```
    '1er',
    (SELECT rowid FROM etape WHERE numeroordre = '3ème étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '1er',
    (SELECT rowid FROM etape WHERE numeroordre = '2nd étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '3ème',
    (SELECT rowid FROM etape WHERE numeroordre = '1re étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bryan' AND nom = 'Coquard'
    AND c.personne_id = p.rowid));
```

6.4 Jointures : récupération des informations

Lorsque la base de données est correctement structurée on doit pouvoir récupérer toutes les informations utiles en réalisant des jointures entre tables. la requête SQL ci-dessous représente la jointure entre toutes les tables de la base de données. On remarquera le renommage des colonnes ayant le même nom sur certaines tables.

```
-- les coureurs de l'équipe 'Vendée U'
SELECT p.nom
FROM equipe e, coureur c, personne p
WHERE e.rowid = c.equipe_id AND p.rowid = c.personne_id AND e.nom = 'Vendée U';

-- jointure entre toutes les tables ( renommage de colonnes de meme ... nom )
SELECT DISTINCT per.nom AS "NOM",
         per.prenom AS "PRENOM",
         per.datedenaissance,
         eq.nom AS "EQUIPE",
         eq.budget,
         cr.taille ,
         sg.nationalite,
         spon.nom AS "SPONSOR",
         spon.adresse,
         spon.domaineactivite ,
         cs.nom AS "COURSE",
         cs.distancetotale ,
         pro.numero,
         pro.nom  AS "PRODUIT",
         pro.indication,
         pro.contre_indication,
         pro.posologie,
         do.quantite AS "DOSE",
         et.numeroordre AS "ETAPE",
         et.date,
         et.type,
         et.villedepart,
         et.villearivee,
         cla.place AS "CLASSEMENT"
FROM personne per, equipe eq ,coureur cr,
   directeur_sportif ds, soigneur sg, sponsor spon,
   course cs, soin s, produit pro, dose do, etape et,
   classement cla
WHERE per.rowid = cr.personne_id
 AND eq.rowid = cr.equipe_id
-- AND per.rowid = ds.personne_id
 AND eq.rowid = ds.equipe_id
-- AND per.rowid = sg.personne_id
 AND eq.rowid = spon.equipe_id
 AND sg.rowid = s.soigneur_id
 AND eq.rowid = s.equipe_id
 AND cs.rowid = s.course_id
 AND pro.rowid = do.produit_id
 AND sg.rowid = do.soigneur_id
 AND cs.rowid = et.course_id
 AND eq.rowid = et.equipegagnante_id
 AND cr.rowid = et.coureurgagnant_id
 AND et.rowid = cla.etape_id
 AND cr.rowid = cla.coureur_id;
```

7 Cas d'usage

A partir de ce modèle de données on peut représenter les cas d'usage qui permettront de tester les requêtes :

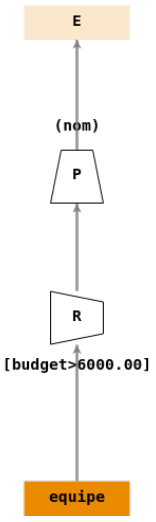
1. sur une seule table avec projections ($\prod$) et critères de restriction (σ)
2. sur plusieurs tables par jointure ($\bowtie$) et critères de restriction (σ)
3. par des requêtes ensemblistes($\cup, \cap,$)
4. avec une division relationnelle ($\div$)
5. en appliquant des fonctions d'agrégats (count(),sum(),max(),min(),avg() ...)
6. en faisant des groupements (GROUP BY)
7. puis des groupements avec critères de restrictions (GROUP BY ... HAVING)

7.1 Recherches des informations

Sur ce modèle de gestion des entreprises on pourrait mettre en œuvre les cas d'usage suivant :

1. $\prod$, σ : le nom de toutes les équipes avec un budget supérieur à 6000.00 €
2. $\prod$, $\bowtie$, σ : la taille de tous les coureurs de l'équipe "Vendée U"
3. $\cup$, $\cap$, : le nom des sponsors de l'équipe "Vendée U" qui n'ont pas pour domaine d'activité la grande distribution
4. $\div$: l'étape où serait classé tous les coureurs de l'équipe "Vendée U"
5. count(), sum(), max(), min(), avg() ... : la distance totale moyenne des courses
6. GROUP BY : le nombre de classement par coureur de l'équipe "Vendée U"
7. GROUP BY,HAVING : le nombre de classement par coureur de l'équipe "Vendée U" lorsque ce nombre est supérieur à 3

1. le nom de toutes les équipes avec un budget supérieur à 6000.00 €
   - calcul relationnel :
     - E = {(e.nom) | e ∈ equipe ∧ e.budget > 6000.00}
   - algèbre relationnelle :
     - E = $\prod_{(nom)}(\sigma_{[budget>6000.00]}(equipe))$
   - arbre de requête :

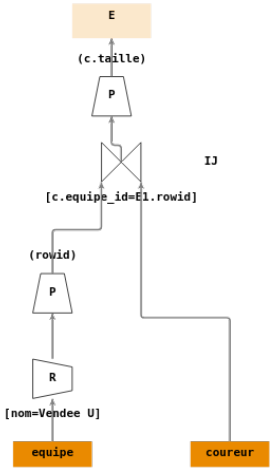


   - requête SQL :

```sql
SELECT nom
FROM equipe
WHERE budget > 6000.00;
```

2. la taille de tous les coureurs de l'équipe "Vendée U"
   - calcul relationnel :
     - E1 : l'équipe "Vendée U"
     - E1 = {e.rowid | equipe(e) ∧ e.nom = 'Vendée U' }
     - E = {c.taille | coureur(c) ∧ E1(e1) ∧ c.equipe_id = e1.rowid}
   - algèbre relationnelle :
     - E1 : l'équipe "Vendée U"
     - E1 = $\prod_{(rowid)}(\sigma_{[nom=VendeeU]}(equipe))$
     - E = $\prod_{(c.taille)}(\bowtie_{[c.equipe\_id=E1.rowid]}(coureur\ c, E1))$

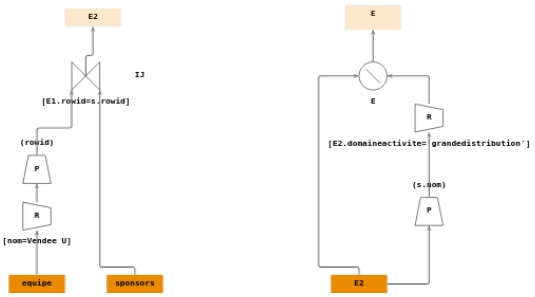
— arbre de requête :



— requête SQL :

```sql
SELECT p.prenom || ' ' || p.nom AS "coureur", taille
FROM coureur c, equipe e, personne p
WHERE c.equipe_id = e.rowid AND e.nom = 'Vendée U' and c.personne_id = p.rowid;
```

3. le nom des sponsors de l'équipe "Vendée U" qui n'ont pas pour domaine d'activité la grande distribution
   — calcul relationnel :
      — E1 : l'équipe "Vendée U"
      — E1 = {e.rowid | equipe(e) ∧ e.nom = 'Vendée U' }
      — E2 : Les sponsors de l'équipe "Vendée U"
      — E2 = {s.rowid | sponsor(s) ∧ E1(e1) ∧ e1.rowid = s.rowid}
      — E3 : Les sponsors de l'équipe "Vendée U" ayant pour domaine d'activité la grande distribution
      — E3 = {e2.rowid | E2(e2) ∧ e2.domaineactivite = 'grande distribution'}
      — E = {e2.nom | e2 ∈ E2 ∧ e2 = ∉ E3 }
   — algèbre relationnelle :
      — E1 : l'équipe "Vendée U"
      — E1 = $\prod_{(rowid)}(\sigma_{[nom=VendeeU]}(equipe))$
      — E2 : Les sponsors de l'équipe "Vendée U"
      — E2 = $\prod_{(s.nom)}(\bowtie_{[E1.rowid=s.rowid]}\ (sponsor\ s, E1))$
      — E3 : Les sponsors de l'équipe "Vendée U" ayant pour domaine d'activité la grande distribution
      — E3 = $\prod_{(E2.nom)}(\sigma_{[E2.domaineactivite='grandedistribution']}(E2))$
      — E = $\setminus(E2, E3)$
   — arbre de requête :

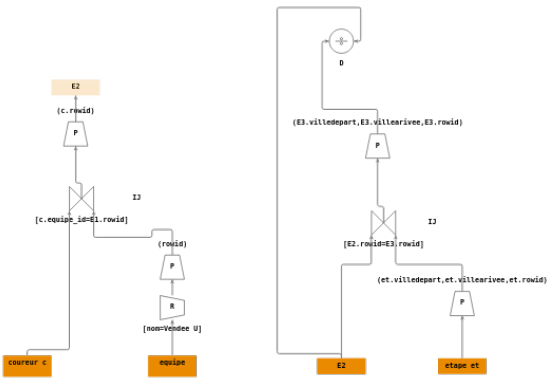


— requête SQL :

```sql
SELECT s.nom FROM sponsor s, equipe e WHERE e.nom = 'Vendée U' AND s.equipe_id = e.rowid
EXCEPT
```

```sql
SELECT s.nom FROM sponsor s, equipe e WHERE e.nom = 'Vendée U' AND s.equipe_id = e.rowid AND
    s.domaineactivite = "Grande distribution";
```

4. nom de l'étape où serait classé tous les coureurs de l'équipe "Vendée U"
   - calcul relationnel :
     - E1 : l'équipe "Vendée U"
     - E1 = {e.rowid | equipe(e) ∧ e.nom ='Vendée U' }
     - E2 : les coureurs de l'équipe "Vendée U"
     - E2 = {c.rowid | coureur(c) ∧ E1(e1) ∧ c.equipe_id = e1.rowid }
     - E3 : nom(villedepart-villearivee) et identifiant des étapes
     - E3 = {et.villedepart, et.villearivee, et.rowid | etape(et) }
     - E4 : nom(villedepart-villearivee) et identifiant des étapes où sont classés les coureurs de l'équipe "Vendée U"
     - E4 = {e3.villedepart, e3.villearivee, e3.rowid | E2(e2),E3(e3) ∧ e2.rowid = e3.rowid }
     - E : nom (villedepart-villearivee) de l'étape associé aux identifiants de tous les coureurs de l'équipe "Vendée U"
     - E = {CONCAT(e4.villedepart, '-', e4.villearivee) | ∀ E2(e2), (e4.villedepart, e4.villearivee, e2.rowid) ∈ E4(e4)}
   - algèbre relationnelle :
     - E1 : l'équipe "Vendée U"
     - E1 = $\prod_{(rowid)}(\sigma_{[nom=VendeeU]}(equipe))$
       item E2 : les coureurs de l'équipe "Vendée U"
     - E2 = $\prod_{(c.rowid)}(\bowtie_{[c.equipe\_id=E1.rowid]}\ (coureur\ c, E1))$
     - E3 : nom(villedepart-villearivee) et identifiant des étapes
     - E3 = $\prod_{(et.villedepart,et.villearivee,et.rowid)}(etape\ et)$
     - E4 : nom(villedepart-villearivee) et identifiant des étapes où sont classés les coureurs de l'équipe "Vendée U"
     - E4 = $\prod_{(E3.villedepart,E3.villearivee,E3.rowid)}(\bowtie_{[E2.rowid=E3.rowid]}\ (E2, E3)$
     - E = ÷(E4,E2)
   - arbre de requête :

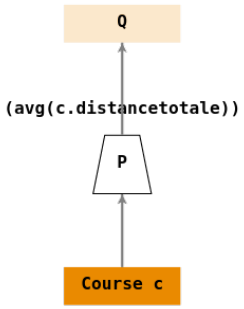


   - requête SQL :

```sql
SELECT et1.villedepart || '-' || et1.villearivee as 'Départ-Arrivée'
FROM etape et1
WHERE NOT EXISTS (
                    SELECT *
                    FROM coureur c,equipe eq
                    WHERE c.equipe_id=eq.rowid AND eq.nom='Vendée U'
                     AND NOT EXISTS (
                                      SELECT *
                                      FROM etape et2, classement cl
                                      WHERE cl.coureur_id = c.rowid  AND cl.etape_id = et1.rowid
                                      AND et1.rowid=et2.rowid
                                    )
);
```

5. la distance totale moyenne des courses
   - calcul relationnel :
     - E = avg(c.distancetotale) | Course(c)

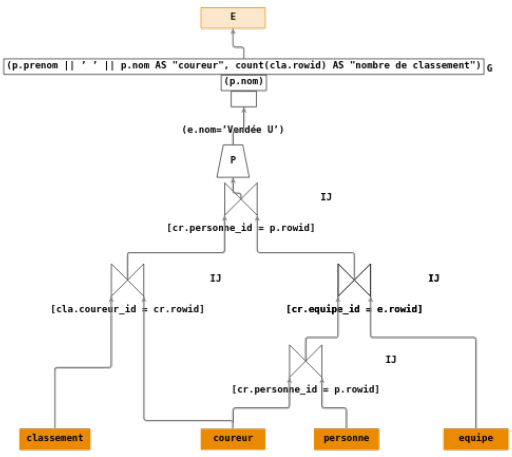
— algèbre relationnelle :
  — E = $\prod_{(avg(c.distancetotale)}(Course)$
— arbre de requête :



— requête SQL :

```sql
SELECT AVG(distancetotale)
FROM course;
```

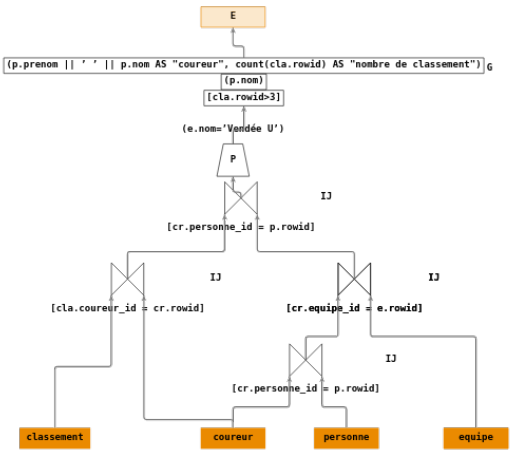
6. le nombre de classement par coureur de l'équipe "Vendée U"
— calcul relationnel :
  — E1 : l'équipe "Vendée U"
  — E1 = {e.rowid | equipe(e) ∧ e.nom ='Vendée U' }
  — E2 : les coureurs de l'équipe "Vendée U"
  — E2 = {c.rowid | coureur(c) ∧ E1(e1) ∧ c.equipe_id = e1.rowid }
  — E3 : les classements réalisé par les coureurs de l'équipe "Vendée U"
  — E3 = {cl.rowid | classement(cl) ∧ E2(e2) ∧ cl.coureur_id = e2.rowid }
  — E = {count(e3.rowid) | p ∈ $P_{(coureur\_id)}(E3)$ }
— algèbre relationnelle :
  — E1 : l'équipe "Vendée U"
  — E1 = $\prod_{(rowid)}(\sigma_{[nom=VendeeU]}(equipe))$
  item E2 : les coureurs de l'équipe "Vendée U"
  — E2 = $\prod_{(c.rowid)}(\bowtie_{[c.equipe\_id=E1.rowid]}\ (coureur\ c, E1))$
  — E3 : les classements réalisé par les coureurs de l'équipe "Vendée U"
  — E3 = $\prod_{(cl.rowid,)}(\bowtie_{[cl.coureur\_id=E2.rowid]}\ (classement, E2)$
  — E = $G^{(count(e3.rowid))}_{(coureur\_id)}(E3)$
— arbre de requête :



— requête SQL :

```sql
SELECT p.prenom || ' ' || p.nom AS "coureur", count(cla.rowid) AS "nombre de classement"
FROM coureur cr, classement cla, equipe e, personne p
WHERE e.nom='Vendée U' AND cla.coureur_id = cr.rowid and cr.equipe_id = e.rowid and cr.
    personne_id = p.rowid
GROUP BY p.nom;
```

7. le nombre de classement par coureur de l'équipe "Vendée U" lorsque ce nombre est supérieur à 3
   - calcul relationnel :
     - E1 : l'équipe "Vendée U"
     - E1 = {e.rowid | equipe(e) ∧ e.nom ='Vendée U' }
     - E2 : les coureurs de l'équipe "Vendée U"
     - E2 = {c.rowid | coureur(c) ∧ E1(e1) ∧ c.equipe_id = e1.rowid }
     - E3 : les classements réalisé par les coureurs de l'équipe "Vendée U"
     - E3 = {cl.rowid | classement(cl) ∧ E2(e2) ∧ cl.coureur_id = e2.rowid }
     - E4 : regroupement par coureur de l'équipe "Vendée U"
     - E4 = {p | p ∈ $P_{(coureur\_id)}(E3)$ }
     - E = {count(e4.rowid) | E4(e4) ∧ count(e4.rowid) > 3}
   - algèbre relationnelle :
     - E4 = $\prod_{(e3.villedepart,e3.villearivee,e3.rowid)}(\bowtie_{[e2.rowid=e3.rowid]} (E2, E3)$
     - E1 : l'équipe "Vendée U"
     - E1 = $\prod_{(rowid)}(\sigma_{[nom=VendeeU]}(equipe))$
       item E2 : les coureurs de l'équipe "Vendée U"
     - E2 = $\prod_{(c.rowid)}(\bowtie_{[c.equipe\_id=E1.rowid]} (coureur, E1))$
     - E3 : les classements réalisé par les coureurs de l'équipe "Vendée U"
     - E3 = $\prod_{(cl.rowid)}(\bowtie_{[cl.coureur\_id=E2.rowid]} (classement, E2)$
     - E = $G^{(count(e4.rowid))}_{(coureur\_id),[count(E3.rowid)>3]}(E3)$
   - arbre de requête :



   - requête SQL :

```sql
SELECT p.prenom || ' ' || p.nom AS "coureur", count(cla.rowid) AS "nombre de classement"
FROM coureur cr, classement cla, equipe e, personne p
WHERE e.nom='Vendée U' AND cla.coureur_id = cr.rowid and cr.equipe_id = e.rowid and cr.
    personne_id = p.rowid
GROUP BY p.nom
HAVING COUNT(cla.rowid) > 3;
```

8 Annexes

8.1 Création de tables

```sql
DROP TABLE IF EXISTS personne;
DROP TABLE IF EXISTS equipe;
DROP TABLE IF EXISTS coureur;
DROP TABLE IF EXISTS directeur_sportif;
DROP TABLE IF EXISTS soigneur;
DROP TABLE IF EXISTS sponsor;
DROP TABLE IF EXISTS course;
DROP TABLE IF EXISTS soin;
DROP TABLE IF EXISTS produit;
DROP TABLE IF EXISTS dose;
DROP TABLE IF EXISTS etape;
DROP TABLE IF EXISTS classement;

--creation de personne
CREATE TABLE personne (
  nom varchar (20),
  prenom varchar (20),
  datedenaissance TEXT
);

--creation de equipe
CREATE TABLE equipe (
  nom varchar(20),
  budget FLOAT
);

--creation de coureur
CREATE TABLE coureur (
  taille FLOAT,
  personne_id INTEGER,
  equipe_id INTEGER,
  FOREIGN KEY ( personne_id ) REFERENCES personnes,
  FOREIGN KEY ( equipe_id ) REFERENCES equipe
);

--creation de directeur sportif
CREATE TABLE directeur_sportif(
  personne_id INTEGER,
  equipe_id INTEGER,
  FOREIGN KEY ( personne_id ) REFERENCES personnes,
  FOREIGN KEY ( equipe_id ) REFERENCES equipe
);

--creation de soigneur
CREATE TABLE soigneur(
  nationalite varchar(20),
  personne_id INTEGER,
  FOREIGN KEY ( personne_id ) REFERENCES personnes
);

--creation de sponsor
CREATE TABLE sponsor(
  nom varchar(20),
  adresse varchar(30),
  domaineactivite varchar(20),
  equipe_id INTEGER,
  FOREIGN KEY ( equipe_id ) REFERENCES equipe
);

--creation de course
CREATE TABLE course(
  nom varchar(20),
  distancetotale FLOAT
);

--creation de soin
CREATE TABLE soin(
  soigneur_id INTEGER,
  equipe_id INTEGER,
  course_id INTEGER,
  FOREIGN KEY ( equipe_id ) REFERENCES equipe,
  FOREIGN KEY ( course_id ) REFERENCES course
);

--creation de produit
CREATE TABLE produit(
```

```
  numero INTEGER,
  nom varchar(20),
  indication varchar(30),
  contre_indication varchar(30),
  posologie varchar(30)
);

--creation de dose
CREATE TABLE dose(
  quantite FLOAT,
  produit_id INTEGER,
  soigneur_id INTEGER,
  FOREIGN KEY ( produit_id ) REFERENCES produit,
  FOREIGN KEY ( soigneur_id ) REFERENCES soigneur
);

--creation de etape
CREATE TABLE etape(
  numeroordre varchar(20),
  date TEXT,
  type varchar(30),
  villedepart varchar(20),
  villearivee varchar(20),
  course_id INTEGER,
  equipegagnante_id INTEGER,
  coureurgagnant_id INTEGER,
  FOREIGN KEY ( equipegagnante_id ) REFERENCES equipe,
  FOREIGN KEY ( coureurgagnant_id ) REFERENCES coureur
);

--creation de classement
CREATE TABLE classement(
  place varchar(10),
  etape_id INTEGER,
  coureur_id INTEGER,
  FOREIGN KEY ( etape_id ) REFERENCES etape,
  FOREIGN KEY ( coureur_id ) REFERENCES coureur
);
```

8.2 Insertion d'informations

```
-- insertion de personne
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Bernard' , 'Hinault' , '14-11-1954')
    ;
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Thomas' , 'Voeckler' , '22-06-1979')
    ;
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Bryan' , 'Coquard' , '25-04-1992');
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Morgan' , 'Lamoisson' , '07-09-1988'
    );
INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Jean' , 'Dupont' , '04-03-1960');

-- insertion de equipe
INSERT INTO equipe ( nom, budget ) VALUES ( 'Vendée U', 80000.15);
INSERT INTO equipe ( nom, budget ) VALUES ( 'Gitane-Campagnolo', 40010.25);

-- insertion de coureur
INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
    1.70,
    (SELECT rowid FROM personne WHERE prenom="Thomas" AND nom="Voeckler"),
    (SELECT rowid FROM equipe WHERE nom="Vendée U"));

INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
    1.75,
    (SELECT rowid FROM personne WHERE prenom="Bryan" AND nom="Coquard"),
    (SELECT rowid FROM equipe WHERE nom="Vendée U"));

INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
    1.80,
    (SELECT rowid FROM personne WHERE prenom="Bernard" AND nom="Hinault"),
    (SELECT rowid FROM equipe WHERE nom="Gitane-Campagnolo"));

-- insertion de directeur sportif
INSERT INTO directeur_sportif ( personne_id , equipe_id ) VALUES (
  (SELECT rowid FROM personne WHERE prenom = 'Morgan' AND nom = 'Lamoisson'),
  (SELECT rowid FROM equipe WHERE nom = 'Vendée U'));

-- insertion de soigneur
INSERT INTO soigneur ( nationalite, personne_id) VALUES (
  'Francais',
  (SELECT rowid FROM personne WHERE prenom = 'Jean' AND nom = 'Dupont'));

```

```sql
-- insertion de sponsor
INSERT INTO sponsor ( nom, adresse, domaineactivite, equipe_id ) VALUES ( 'Système U',
    '20 RUE D ARCUEIL PARC TERTIAIRE SILIC BATIMENT MO 94150 RUNGIS',
    'Grande distribution',
    (SELECT rowid FROM equipe WHERE nom = 'Vendée U'));

INSERT INTO sponsor ( nom, adresse, domaineactivite, equipe_id ) VALUES ( 'Département Vendée',
    '2 Av. Gordon Bennett, 75016 Paris',
    'Département',
    (SELECT rowid FROM equipe WHERE nom = 'Vendée U'));

-- insertion de course
INSERT INTO course ( nom, distancetotale ) VALUES ( 'Tour de France',  3000);
INSERT INTO course ( nom, distancetotale ) VALUES ( 'Tour d Espagne',  1000);

-- insertion de soin
INSERT INTO soin ( soigneur_id, equipe_id, course_id ) VALUES (
  (SELECT s.rowid FROM personne p CROSS JOIN soigneur s WHERE prenom = 'Jean' AND nom = 'Dupont' AND s
    .personne_id = p.rowid),
  (SELECT rowid FROM equipe WHERE nom = 'Vendée U'),
  (SELECT rowid FROM course WHERE nom = 'Tour de France'));

-- insertion de produit
INSERT INTO produit ( numero, nom, indication, contre_indication, posologie ) VALUES ( 1547,
  'Salbutamol',
  'douleur musculaire',
  'ne pas administrer en dessous de 20 ans',
  '1 comprimé par jour');

-- insertion de dose
INSERT INTO dose ( quantite, produit_id, soigneur_id ) VALUES ( '2.85',
  (SELECT rowid FROM produit WHERE nom = 'Salbutamol'),
  (SELECT s.rowid FROM personne p CROSS JOIN soigneur s WHERE prenom = 'Jean' AND nom = 'Dupont' AND s
    .personne_id = p.rowid));

-- insertion de etape
INSERT INTO etape ( numeroordre, date, type, villedepart, villearivee, course_id, equipegagnante_id,
    coureurgagnant_id ) VALUES (
    '1re étape',
    '15-01-2015',
    'Contre la montre individuel',
    'Lorient',
    'Brest',
    (SELECT rowid FROM course WHERE nom = 'Tour de France'),
    (SELECT rowid FROM equipe WHERE nom = 'Vendée U'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO etape ( numeroordre, date, type, villedepart, villearivee, course_id, equipegagnante_id,
    coureurgagnant_id ) VALUES (
    '2nd étape',
    '16-02-2015',
    'Course en ligne',
    'Brest',
    'Lannion',
    (SELECT rowid FROM course WHERE nom = 'Tour de France'),
    NULL,
    NULL);

INSERT INTO etape ( numeroordre, date, type, villedepart, villearivee, course_id, equipegagnante_id,
    coureurgagnant_id ) VALUES (
    '3ème étape',
    '17-02-2015',
    'Course en ligne',
    'Lannion',
    'Rennes',
    (SELECT rowid FROM course WHERE nom = 'Tour de France'),
    NULL,
    NULL);

INSERT INTO etape ( numeroordre, date, type, villedepart, villearivee, course_id, equipegagnante_id,
    coureurgagnant_id ) VALUES (
    '4ème étape',
    '18-02-2015',
    'Course en ligne',
    'Rennes',
    'Nantes',
    (SELECT rowid FROM course WHERE nom = 'Tour de France'),
    NULL,
    NULL);

```

```sql
-- insertion de classement
INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '1er',
    (SELECT rowid FROM etape WHERE numeroordre = '1re étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '3ème',
    (SELECT rowid FROM etape WHERE numeroordre = '3ème étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '2nd',
    (SELECT rowid FROM etape WHERE numeroordre = '2nd étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '4ème',
    (SELECT rowid FROM etape WHERE numeroordre = '4ème étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '2nd',
    (SELECT rowid FROM etape WHERE numeroordre = '1re étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '1er',
    (SELECT rowid FROM etape WHERE numeroordre = '3ème étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '1er',
    (SELECT rowid FROM etape WHERE numeroordre = '2nd étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
    AND c.personne_id = p.rowid));

INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
    '3ème',
    (SELECT rowid FROM etape WHERE numeroordre = '1re étape'),
    (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bryan' AND nom = 'Coquard'
    AND c.personne_id = p.rowid));
```

8.3 Mise à jour d'informations

```sql
UPDATE coureur
SET taille = 1.82
WHERE personne_id = (SELECT rowid FROM personne WHERE nom = 'Thomas' AND prenom = 'Voeckler');

UPDATE soigneur
SET nationalite = 'Allemand/Francais'
WHERE personne_id = (SELECT rowid FROM personne WHERE nom = 'Jean' AND prenom = 'Dupont');

UPDATE sponsor
SET adresse = '2bis rue Louis Armand, 75015 PARIS'
WHERE nom = 'Système U';
```

8.4 Jointures entre tables

```sql
-- les coureurs de l'équipe 'Vendée U'
SELECT p.nom
FROM equipe e, coureur c, personne p
WHERE e.rowid = c.equipe_id AND p.rowid = c.personne_id AND e.nom = 'Vendée U';

-- jointure entre toutes les tables ( renommage de colonnes de meme ... nom )
SELECT DISTINCT per.nom AS "NOM",
         per.prenom AS "PRENOM",
         per.datedenaissance,
         eq.nom AS "EQUIPE",
         eq.budget,
         cr.taille ,
         sg.nationalite,
         spon.nom AS "SPONSOR",
         spon.adresse,
```

```
16          spon.domaineactivite ,
17          cs.nom AS "COURSE",
18          cs.distancetotale ,
19          pro.numero,
20          pro.nom  AS "PRODUIT",
21          pro.indication,
22          pro.contre_indication,
23          pro.posologie,
24          do.quantite AS "DOSE",
25          et.numeroordre AS "ETAPE",
26          et.date,
27          et.type,
28          et.villedepart,
29          et.villearivee,
30          cla.place AS "CLASSEMENT"
31 FROM personne per, equipe eq ,coureur cr,
32    directeur_sportif ds, soigneur sg, sponsor spon,
33    course cs, soin s, produit pro, dose do, etape et,
34    classement cla
35 WHERE per.rowid = cr.personne_id
36  AND eq.rowid = cr.equipe_id
37 -- AND per.rowid = ds.personne_id
38  AND eq.rowid = ds.equipe_id
39 -- AND per.rowid = sg.personne_id
40  AND eq.rowid = spon.equipe_id
41  AND sg.rowid = s.soigneur_id
42  AND eq.rowid = s.equipe_id
43  AND cs.rowid = s.course_id
44  AND pro.rowid = do.produit_id
45  AND sg.rowid = do.soigneur_id
46  AND cs.rowid = et.course_id
47  AND eq.rowid = et.equipegagnante_id
48  AND cr.rowid = et.coureurgagnant_id
49  AND et.rowid = cla.etape_id
50  AND cr.rowid = cla.coureur_id;
```