

ENIB Semester S3P
Computer Science - Relational Databases

Union Cycliste Internationale: Managing Riders in Different Teams

Florian Pasco

S3P-A



Version 2.0 - May 10, 2022

Contents

1	Initial Specifications	3
2	Entity Identification	3
3	Identification of Associations	3
3.1	Entity-Association	3
3.2	UML Modeling (Version 1.1)	4
4	Identification of Cardinalities	4
4.1	Association-Cardinality	4
4.2	UML Modeling (Version 1.2)	5
4.3	UML Modeling (Version 1.3)	6
5	Entity-Attributes	6
5.1	Attribute Identification	6
5.2	UML Modeling (Version 1.4)	7
6	From UML to SQL	7
6.1	UML Modeling (Version 1.4)	7
6.2	Database Structuring	8
6.3	Inserting Data into the Database	9
6.4	Joins: Retrieving Information	11
7	Use Case	11
7.1	Information Retrieval	12
8	Appendices	13
8.1	Table Creation	13
8.2	Data Insertion	15
8.3	Data Update	17
8.4	Table Joins	17

List of Figures

1	Model: Entity-Association	4
2	Model: Association Cardinalities	5
3	Model: Association Cardinalities	6
4	Model: Attributes	7
5	Model: Key Attributes	8

1 Initial Specifications

The Union Cycliste Internationale has requested our IT services to develop an Information System to manage the riders belonging to the various teams of the federation based on the following specifications:

"Riders are individuals for whom we record their first and last name, height, date of birth, and the team they belong to. A team is identified by its name, has a budget, and a sports director who is a person with a known first and last name and date of birth. Teams are funded by sponsors, which may vary each year, and for each sponsor, we record their name, address, and field of activity.

A race corresponds to a race name (e.g., 'Tour de France'), and we know the total distance to be covered. It can include one or more stages, each identified by its sequence number (e.g., 'Stage 3'), date, type (e.g., 'Individual Time Trial'), starting city, and finishing city.

For each rider who participated in a stage of a race, we record the ranking they achieved in that stage. For each race, we track the final winner and the team they belong to.

For every race, teams employ soigneurs, who are individuals for whom we store their first and last name, date of birth, and nationality. Additionally, at each stage, we record which doses of which product(s) a soigneur administered to a rider.

A product is identified by a product number, has a name, an indication (e.g., 'muscle pain'), a contraindication (e.g., 'do not administer to individuals under 20 years old'), and a dosage recommendation (e.g., '1 tablet per day').

In this production database, only current information (related to the ongoing edition) about the race, riders, teams, etc., is stored."

2 Entity Identification

Based on these specifications, we identify the most important entities and the related information.

- A **rider** is a **person** for whom we record their first and last name, height, date of birth, and the **team** they belong to.
- A **team** is identified by its name, has a budget, a **sports director**, and is funded by **sponsors**.
- A **sports director** is a **person** for whom we store their first and last name and date of birth.
- A **sponsor** is identified by its name, address, and field of activity.
- A **race** corresponds to a race name, and we know the total distance to be covered.
- A **stage** has an order number (e.g., "Stage 3"), a date, a type, a starting city, and a finishing city. A ranking determines a rider's position during a stage of a race.
- A **soigneur** is a **person** for whom we record their first and last name, date of birth, and nationality.
- A **product** is identified by a product number and has a name, an indication, a contraindication, and a dosage recommendation.
- A **ranking** obtained by a **rider** during a stage is recorded.

3 Identification of Associations

Associations link entities together. We identify associations from subject-verb-object sentences in the specifications. The verb represents the association between two entities (Subject, Object).

3.1 Entity-Association

Based on the Union Cycliste Internationale specifications, we extract key sentences that connect entities through verbs:

- A person **is** a rider.
- A person **is** a sports director.
- A person **is** a soigneur.
- A rider **belongs to** a team.
- A soigneur **treats** a rider.
- A stage **is won by** a rider.
- A ranking **determines the position of** a rider.

- A ranking **is achieved during** a stage.
- A race **includes** stages.
- A stage **is won by** a team.
- A soigneur **provides treatment during** a race.
- A soigneur **treats at the request of** a team.

3.2 UML Modeling (Version 1.1)

From these associations between entities, we can propose an initial version of the database structure using UML formalism.

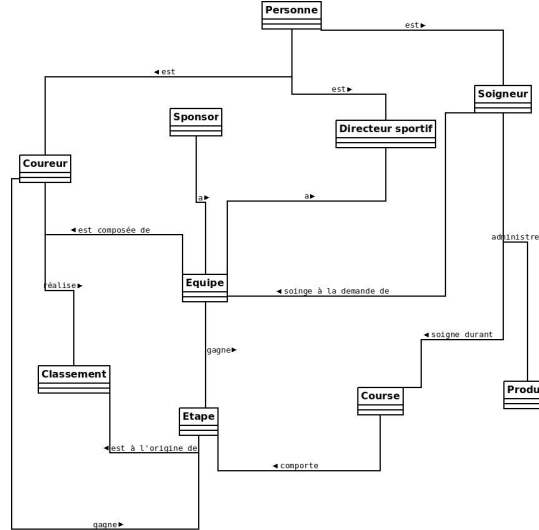


Figure 1: Model: Entity-Association

4 Identification of Cardinalities

From these sentences in the specifications, we can also identify the cardinality of associations (one-to-one, one-to-many, many-to-many, etc.). For example, a treatment is administered during one or multiple races, and a team is funded by at least one sponsor.

4.1 Association-Cardinality

Identifying cardinalities is crucial to managing referential constraints between entities. These cardinalities determine which entity (table) should hold a reference (foreign key) to another entity (table) in an association or whether a new entity is required to link the two entities. This is especially necessary for many-to-many (M:N) relationships.

- A person **can be (but is not necessarily)** a rider, a sports director, or a soigneur.
- A team consists of **multiple** riders.
- A rider **may or may not** win stages.
- A rider **may or may not** achieve rankings.
- A stage results in **at least one** ranking.
- A race consists of **at least one** stage.
- A team **may or may not** win stages.
- **Multiple** soigneurs provide treatment during **multiple** races.
- **Multiple** soigneurs provide treatment at the request of **multiple** teams.
- **Multiple** soigneurs administer **multiple** products.
- A team **must have exactly one** sports director.
- A team has **at least one** sponsor.

4.2 UML Modeling (Version 1.2)

Based on the identified cardinalities, we can propose a second version of the UML model.

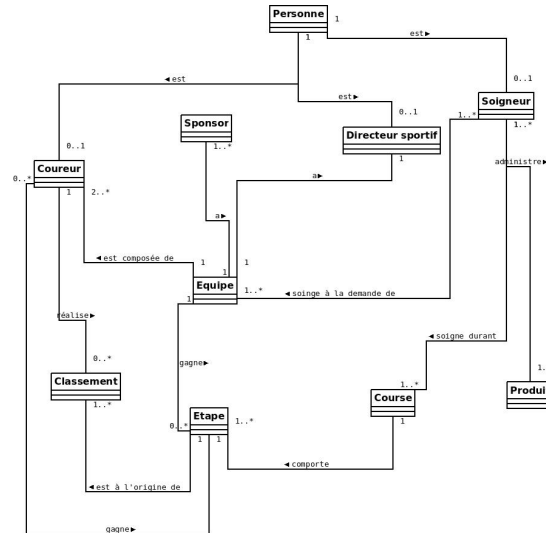


Figure 2: Model: Association Cardinalities

4.3 UML Modeling (Version 1.3)

The previous UML model reveals several many-to-many relationships, which must systematically be broken down into two one-to-many associations. We can express this refinement more precisely by introducing associative tables (entities) and redefining the associations between these new tables and the two original entities.

First:

- **Multiple** soigneurs provide treatment during **multiple** races.

becomes:

- **A single** treatment is administered during **multiple** races.
- **Multiple** soigneurs provide **a single** treatment.

Then:

- **Multiple** soigneurs provide treatment at the request of **multiple** teams.

becomes:

- **Multiple** soigneurs provide **a single** treatment.
- **Multiple** teams receive **a single** treatment.

Finally:

- **Multiple** soigneurs administer **multiple** products.

becomes:

- **Multiple** soigneurs administer **a single** dose.
- **Multiple** products are used within **the same** dose.

This results in the following model:

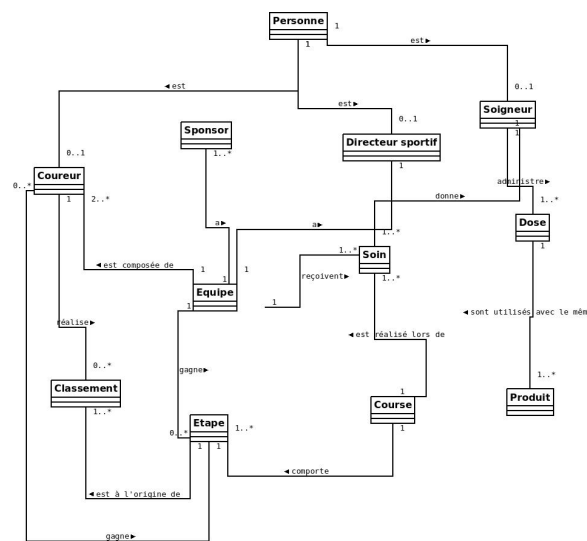


Figure 3: Model: Association Cardinalities

5 Entity-Attributes

After identifying the key entities from the specifications, we must extract the information that characterizes them.

5.1 Attribute Identification

From our specifications, we can retain the following characteristics:

- A person is identified by a **name**, a **surname**, and their **date of birth**.
- A rider is identified by their **height**.
- A team is identified by its **name** and has a **budget**.

- A sponsor has a **name**, an **address**, and a **business sector**.
- A race has a **name** and a **total distance**.
- A stage has an **order number**, a **date**, a **type**, a **departure city**, and an **arrival city**.
- A ranking corresponds to a **position** obtained.
- A soigneur is identified by their **nationality**.
- A dose is a **quantity** of a product.
- A product has a **number**, a **name**, an **indication**, a **contraindication**, and a **dosage**.

5.2 UML Modeling (Version 1.4)

We can represent the entity attributes in a third version of the UML model.

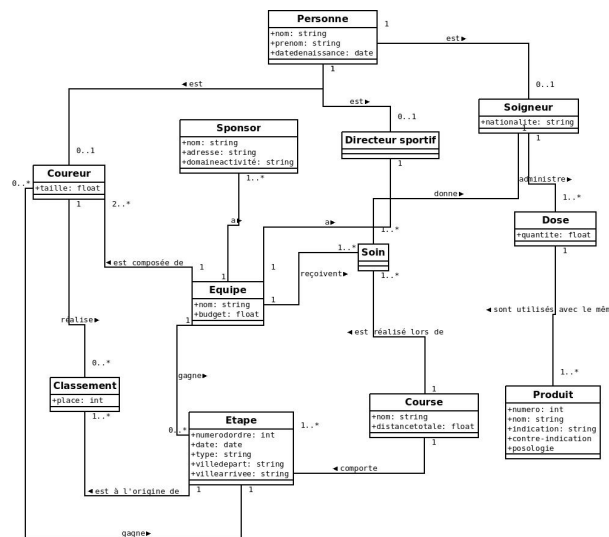


Figure 4: Model: Attributes

This version can be considered as the first version to be proposed to the client based on the initial specifications.

6 From UML to SQL

The transition from a UML-based representation to a relational database structure requires defining attributes within entities that will serve as primary keys for tables and foreign keys referencing the primary keys of associated entities (tables).

6.1 UML Modeling (Version 1.4)

Based on the cardinalities of the associations, we can represent primary and foreign keys in the previous UML model (e.g., `personne.id`, `soigneur.personne_id`, etc.).

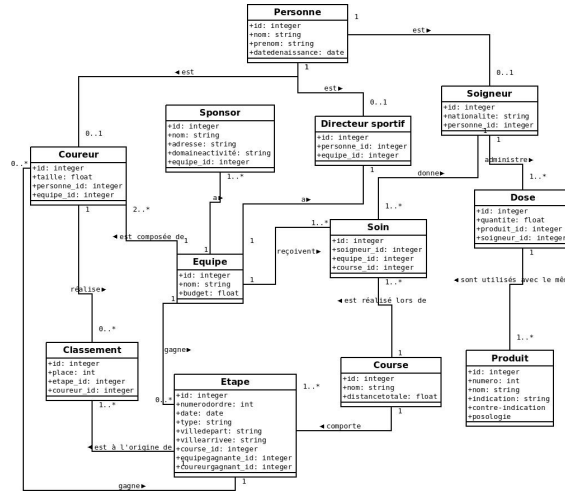


Figure 5: Model: Key Attributes

6.2 Database Structuring

From this UML schema, we can implement the database structure in SQL.

- personne (id, nom, prenom, datedenaissance)
- coureur (id, taille, #personne_id, #equipe_id)
- sponsor (id, nom, adresse, domaineactivite, #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, #soigneur_id, quantite)
- classement (id, place, #etape_id, #coureur_id)
- etape (id, numerodordre, date, type, villedepart, villearrivee, #course_id, #equipegagnante_id, #coureurgagnant_id)
- course (id, quantite, #produit_id, #soigneur_id)
- produit (id, quantite, #produit_id, #soigneur_id)

In this representation, primary keys are underlined, and foreign keys are prefixed with a hashtag (#). In SQLite, it is neither necessary nor recommended to define an automatically incremented (AUTOINCREMENT) primary key, as a rowid is directly available.

With this structure, we can create the database.

```

1 --creation de coureur
2 CREATE TABLE coureur (
3     taille FLOAT,
4     personne_id INTEGER,
5     equipe_id INTEGER,
6     FOREIGN KEY ( personne_id ) REFERENCES personnes ,
7     FOREIGN KEY ( equipe_id ) REFERENCES equipe
8 );
9
10 -- postes primary key : rowid
11 SELECT rowid from coureur ;
12 rowid
13 -- ---
14 1
15 2
16 ...

```

The full SQL script for creating all database tables is available in the appendix.

6.3 Inserting Data into the Database

With this structure in place, we can populate the database by inserting data and performing updates using SQL commands such as:

```
1 INSERT INTO ... VALUES (...);
2 UPDATE ... SET ... WHERE ...;

1 -- insertion de personne
2 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Bernard' , 'Hinault' , '14-11-1954')
;
3 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Thomas' , 'Voeckler' , '22-06-1979')
;
4 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Bryan' , 'Coquard' , '25-04-1992');
5 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Morgan' , 'Lamoisson' , '07-09-1988'
);
6 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Jean' , 'Dupont' , '04-03-1960');
7
8 -- insertion de equipe
9 INSERT INTO equipe ( nom, budget ) VALUES ( 'Vend e_U', 80000.15);
10 INSERT INTO equipe ( nom, budget ) VALUES ( 'Gitane-Campagnolo', 40010.25);
11
12 -- insertion de coureur
13 INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
14     1.70,
15     (SELECT rowid FROM personne WHERE prenom="Thomas" AND nom="Voeckler"),
16     (SELECT rowid FROM equipe WHERE nom="Vend e_U"));
17
18 INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
19     1.75,
20     (SELECT rowid FROM personne WHERE prenom="Bryan" AND nom="Coquard"),
21     (SELECT rowid FROM equipe WHERE nom="Vend e_U"));
22
23 INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
24     1.80,
25     (SELECT rowid FROM personne WHERE prenom="Bernard" AND nom="Hinault"),
26     (SELECT rowid FROM equipe WHERE nom="Gitane-Campagnolo"));
27
28 -- insertion de directeur sportif
29 INSERT INTO directeur_sportif ( personne_id , equipe_id ) VALUES (
30     (SELECT rowid FROM personne WHERE prenom = 'Morgan' AND nom = 'Lamoisson'),
31     (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'));
32
33 -- insertion de soigneur
34 INSERT INTO soigneur ( nationalite, personne_id) VALUES (
35     'Francais',
36     (SELECT rowid FROM personne WHERE prenom = 'Jean' AND nom = 'Dupont'));
37
38 -- insertion de sponsor
39 INSERT INTO sponsor ( nom, adresse, domaineactivite, equipe_id ) VALUES ( 'Syst me_U',
40     '20_RUE_D_ARCUEIL_PARC_TERTIAIRE_SILIC_BATIMENT_MO_94150_RUNGIS',
41     'Grande_distribution',
42     (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'));
43
44 INSERT INTO sponsor ( nom, adresse, domaineactivite, equipe_id ) VALUES ( 'D partement_Vend e',
45     '2_Av._Gordon_Bennett,_75016_Paris',
46     'D partement',
47     (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'));
48
49 -- insertion de course
50 INSERT INTO course ( nom, distancetotale ) VALUES ( 'Tour_de_France', 3000);
51 INSERT INTO course ( nom, distancetotale ) VALUES ( 'Tour_d_Espagne', 1000);
52
53 -- insertion de soin
54 INSERT INTO soin ( soigneur_id, equipe_id, course_id ) VALUES (
55     (SELECT s.rowid FROM personne p CROSS JOIN soigneur s WHERE prenom = 'Jean' AND nom = 'Dupont'
AND s.personne_id = p.rowid),
56     (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'),
57     (SELECT rowid FROM course WHERE nom = 'Tour_de_France'));
58
59 -- insertion de produit
60 INSERT INTO produit ( numero, nom, indication, contre_indication, posologie ) VALUES ( 1547,
61     'Salbutamol',
62     'douleur_musculaire',
63     'ne_pas_administrer_en_dessous_de_20_ans',
64     '1_comprim _par_jour');
65
66 -- insertion de dose
67 INSERT INTO dose ( quantite, produit_id, soigneur_id ) VALUES ( '2.85',
68     (SELECT rowid FROM produit WHERE nom = 'Salbutamol'),
```

```

69      (SELECT s.rowid FROM personne p CROSS JOIN soigneur s WHERE prenom = 'Jean' AND nom = 'Dupont'
70      AND s.personne_id = p.rowid));
71 -- insertion de etape
72 INSERT INTO etape ( numeroordre, date, type, villeddepart, villearivee, course_id, equipegagnante_id,
73      coureurgagnant_id ) VALUES (
74      '1re tape ',
75      '15-01-2015',
76      'Contre la montre individuel',
77      'Lorient',
78      'Brest',
79      (SELECT rowid FROM course WHERE nom = 'Tour de France'),
80      (SELECT rowid FROM equipe WHERE nom = 'Vendée U'),
81      (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
82      AND c.personne_id = p.rowid));
83
84 INSERT INTO etape ( numeroordre, date, type, villeddepart, villearivee, course_id, equipegagnante_id,
85      coureurgagnant_id ) VALUES (
86      '2nd tape ',
87      '16-02-2015',
88      'Course en ligne',
89      'Brest',
90      'Lannion',
91      (SELECT rowid FROM course WHERE nom = 'Tour de France'),
92      NULL,
93      NULL);
94
95 INSERT INTO etape ( numeroordre, date, type, villeddepart, villearivee, course_id, equipegagnante_id,
96      coureurgagnant_id ) VALUES (
97      '3 me tape ',
98      '17-02-2015',
99      'Course en ligne',
100      'Lannion',
101      'Rennes',
102      (SELECT rowid FROM course WHERE nom = 'Tour de France'),
103      NULL,
104      NULL);
105
106 INSERT INTO etape ( numeroordre, date, type, villeddepart, villearivee, course_id, equipegagnante_id,
107      coureurgagnant_id ) VALUES (
108      '4 me tape ',
109      '18-02-2015',
110      'Course en ligne',
111      'Rennes',
112      'Nantes',
113      (SELECT rowid FROM course WHERE nom = 'Tour de France'),
114      NULL,
115      NULL);
116
117 -- insertion de classement
118 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
119      '1er',
120      (SELECT rowid FROM etape WHERE numeroordre = '1re tape '),
121      (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
122      AND c.personne_id = p.rowid));
123
124 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
125      '3 me ',
126      (SELECT rowid FROM etape WHERE numeroordre = '3 me tape '),
127      (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
128      AND c.personne_id = p.rowid));
129
130 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
131      '2nd',
132      (SELECT rowid FROM etape WHERE numeroordre = '2nd tape '),
133      (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
134      AND c.personne_id = p.rowid));
135
136 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
137      '4 me ',
138      (SELECT rowid FROM etape WHERE numeroordre = '4 me tape '),
139      (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
140      AND c.personne_id = p.rowid));
141
142 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
143      '2nd',
144      (SELECT rowid FROM etape WHERE numeroordre = '1re tape '),
145      (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
146      AND c.personne_id = p.rowid));
147
148 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (

```

```

139     '1er',
140     (SELECT rowid FROM etape WHERE numeroordre = '3 me tape '),
141     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
142         AND c.personne_id = p.rowid));
143 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
144     '1er',
145     (SELECT rowid FROM etape WHERE numeroordre = '2nd tape '),
146     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
147         AND c.personne_id = p.rowid));
148 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
149     '3 me ',
150     (SELECT rowid FROM etape WHERE numeroordre = '1re tape '),
151     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bryan' AND nom = 'Coquard'
152         AND c.personne_id = p.rowid));

```

6.4 Joins: Retrieving Information

Once the database is properly structured, we must be able to retrieve all necessary information using table joins. The SQL query below demonstrates a join across all database tables. Column renaming is applied where multiple tables contain attributes with the same name.

```

1  -- les coureurs de l' equipe 'Vend e U'
2  SELECT p.nom
3  FROM equipe e, coureur c, personne p
4  WHERE e.rowid = c.equipe_id AND p.rowid = c.personne_id AND e.nom = 'Vend e U';
5
6  -- jointure entre toutes les tables ( renommage de colonnes de meme ... nom )
7  SELECT DISTINCT per.nom AS "NOM",
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
    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.villearrivee,
    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 Use Case

Based on this data model, we can represent the use cases that will allow us to test the queries:

1. on a single table with projections (Π) and restriction criteria (σ)
2. on multiple tables using joins (\bowtie) and restriction criteria (σ)
3. using set queries (\cup, \cap, \setminus)
4. with a relational division (\div)
5. applying aggregate functions ($\text{count}()$, $\text{sum}()$, $\text{max}()$, $\text{min}()$, $\text{avg}()$...)
6. performing groupings (GROUP BY)
7. then groupings with restriction criteria (GROUP BY ... HAVING)

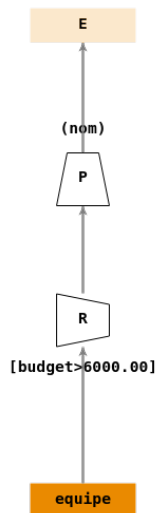
7.1 Information Retrieval

Using this company management model, we could implement the following use cases:

1. Π, σ : the name of all teams with a budget greater than €6000.00
2. Π, \bowtie, σ : the height of all cyclists in the "Vendée U" team
3. \cup, \cap, \setminus : the name of the sponsors of the "Vendée U" team that do not have retail as their business domain
4. \div : the stage where all the cyclists of the "Vendée U" team would be ranked
5. $\text{count}()$, $\text{sum}()$, $\text{max}()$, $\text{min}()$, $\text{avg}()$... : the average total distance of the races
6. GROUP BY: the number of rankings per cyclist of the "Vendée U" team
7. GROUP BY, HAVING: the number of rankings per cyclist of the "Vendée U" team when this number is greater than 3

1. The name of all teams with a budget greater than €6000.00

- Relational Calculus:
 - $E = \{(e.name) \mid e \in \text{team} \wedge e.budget > 6000.00\}$
- Relational Algebra:
 - $E = \Pi_{(name)}(\sigma_{[budget > 6000.00]}(team))$
- Query Tree:



- SQL Query:

```

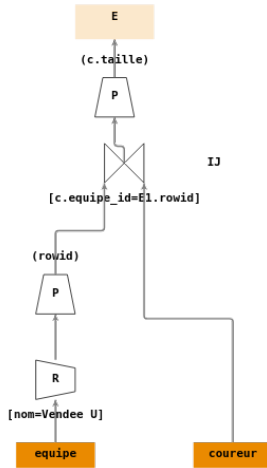
1 SELECT nom
2 FROM equipe
3 WHERE budget > 6000.00;

```

2. The height of all cyclists in the "Vendée U" team

- Relational Calculus:
 - E1: The "Vendée U" team

- $E1 = \{e.rowid \mid team(e) \wedge e.name = 'Vendée U' \}$
- $E = \{c.height \mid cyclist(c) \wedge E1(e1) \wedge c.team_id = e1.rowid\}$
- Relational Algebra:
 - $E1$: The "Vendée U" team
 - $E1 = \prod_{(rowid)} (\sigma_{[name=VendeU]}(team))$
 - $E = \prod_{(c.height)} (\bowtie_{[c.team_id=E1.rowid]} (cyclist\ c, E1))$
- Query Tree:



- SQL Query:

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

8 Appendices

8.1 Table Creation

```
1 DROP TABLE IF EXISTS personne;
2 DROP TABLE IF EXISTS equipe;
3 DROP TABLE IF EXISTS coureur;
4 DROP TABLE IF EXISTS directeur_sportif;
5 DROP TABLE IF EXISTS soigneur;
6 DROP TABLE IF EXISTS sponsor;
7 DROP TABLE IF EXISTS course;
8 DROP TABLE IF EXISTS soin;
9 DROP TABLE IF EXISTS produit;
10 DROP TABLE IF EXISTS dose;
11 DROP TABLE IF EXISTS etape;
12 DROP TABLE IF EXISTS classement;
13
14 --creation de personne
15 CREATE TABLE personne (
16     nom varchar (20),
17     prenom varchar (20),
18     datedenaissance TEXT
19 );
20
21 --creation de equipe
22 CREATE TABLE equipe (
23     nom varchar(20),
24     budget FLOAT
25 );
26
27 --creation de coureur
28 CREATE TABLE coureur (
29     taille FLOAT,
30     personne_id INTEGER,
31     equipe_id INTEGER,
32     FOREIGN KEY ( personne_id ) REFERENCES personnes,
33     FOREIGN KEY ( equipe_id ) REFERENCES equipe
```

```

34 );
35
36 --creation de directeur sportif
37 CREATE TABLE directeur_sportif(
38     personne_id INTEGER,
39     equipe_id INTEGER,
40     FOREIGN KEY ( personne_id ) REFERENCES personnes,
41     FOREIGN KEY ( equipe_id ) REFERENCES equipe
42 );
43
44 --creation de soigneur
45 CREATE TABLE soigneur(
46     nationalite varchar(20),
47     personne_id INTEGER,
48     FOREIGN KEY ( personne_id ) REFERENCES personnes
49 );
50
51 --creation de sponsor
52 CREATE TABLE sponsor(
53     nom varchar(20),
54     adresse varchar(30),
55     domaineactivite varchar(20),
56     equipe_id INTEGER,
57     FOREIGN KEY ( equipe_id ) REFERENCES equipe
58 );
59
60 --creation de course
61 CREATE TABLE course(
62     nom varchar(20),
63     distancetotale FLOAT
64 );
65
66 --creation de soin
67 CREATE TABLE soin(
68     soigneur_id INTEGER,
69     equipe_id INTEGER,
70     course_id INTEGER,
71     FOREIGN KEY ( equipe_id ) REFERENCES equipe,
72     FOREIGN KEY ( course_id ) REFERENCES course
73 );
74
75 --creation de produit
76 CREATE TABLE produit(
77     numero INTEGER,
78     nom varchar(20),
79     indication varchar(30),
80     contre_indication varchar(30),
81     posologie varchar(30)
82 );
83
84 --creation de dose
85 CREATE TABLE dose(
86     quantite FLOAT,
87     produit_id INTEGER,
88     soigneur_id INTEGER,
89     FOREIGN KEY ( produit_id ) REFERENCES produit,
90     FOREIGN KEY ( soigneur_id ) REFERENCES soigneur
91 );
92
93 --creation de etape
94 CREATE TABLE etape(
95     numeroordre varchar(20),
96     date TEXT,
97     type varchar(30),
98     villedepart varchar(20),
99     villearrivee varchar(20),
100     course_id INTEGER,
101     equipegagnante_id INTEGER,
102     coureurgagnant_id INTEGER,
103     FOREIGN KEY ( equipegagnante_id ) REFERENCES equipe,
104     FOREIGN KEY ( coureurgagnant_id ) REFERENCES coureur
105 );
106
107 --creation de classement
108 CREATE TABLE classement(
109     place varchar(10),
110     etape_id INTEGER,
111     coureur_id INTEGER,
112     FOREIGN KEY ( etape_id ) REFERENCES etape,
113     FOREIGN KEY ( coureur_id ) REFERENCES coureur
114 );

```

8.2 Data Insertion

```
1 -- insertion de personne
2 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Bernard' , 'Hinault' , '14-11-1954')
3 ;
4 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Thomas' , 'Voeckler' , '22-06-1979')
5 ;
6 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Bryan' , 'Coquard' , '25-04-1992');
7 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Morgan' , 'Lamoisson' , '07-09-1988'
8 );
9 INSERT INTO personne ( prenom , nom , datedenaissance ) VALUES ( 'Jean' , 'Dupont' , '04-03-1960');
10
11 -- insertion de equipe
12 INSERT INTO equipe ( nom, budget ) VALUES ( 'Vend e_U', 80000.15);
13 INSERT INTO equipe ( nom, budget ) VALUES ( 'Gitane-Campagnolo', 40010.25);
14
15 -- insertion de coureur
16 INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
17     1.70,
18     (SELECT rowid FROM personne WHERE prenom="Thomas" AND nom="Voeckler"),
19     (SELECT rowid FROM equipe WHERE nom="Vend e_U"));
20
21 INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
22     1.75,
23     (SELECT rowid FROM personne WHERE prenom="Bryan" AND nom="Coquard"),
24     (SELECT rowid FROM equipe WHERE nom="Vend e_U"));
25
26 INSERT INTO coureur(taille, personne_id, equipe_id) VALUES (
27     1.80,
28     (SELECT rowid FROM personne WHERE prenom="Bernard" AND nom="Hinault"),
29     (SELECT rowid FROM equipe WHERE nom="Gitane-Campagnolo"));
30
31 -- insertion de directeur sportif
32 INSERT INTO directeur_sportif ( personne_id , equipe_id ) VALUES (
33     (SELECT rowid FROM personne WHERE prenom = 'Morgan' AND nom = 'Lamoisson'),
34     (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'));
35
36 -- insertion de soigneur
37 INSERT INTO soigneur ( nationalite, personne_id) VALUES (
38     'Francais',
39     (SELECT rowid FROM personne WHERE prenom = 'Jean' AND nom = 'Dupont'));
40
41 -- insertion de sponsor
42 INSERT INTO sponsor ( nom, adresse, domaineactivite, equipe_id ) VALUES ( 'Syst me_U',
43     '20_RUE_D_ARCUEIL_PARC_TERTIAIRE_SILIC_BATIMENT_MO_94150_RUNGIS',
44     'Grande_distribution',
45     (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'));
46
47 INSERT INTO sponsor ( nom, adresse, domaineactivite, equipe_id ) VALUES ( 'D partement_Vend e',
48     '2_Av._Gordon_Bennett,_75016_Paris',
49     'D partement',
50     (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'));
51
52 -- insertion de course
53 INSERT INTO course ( nom, distancetotale ) VALUES ( 'Tour_de_France', 3000);
54 INSERT INTO course ( nom, distancetotale ) VALUES ( 'Tour_d_Espagne', 1000);
55
56 -- insertion de soin
57 INSERT INTO soin ( soigneur_id, equipe_id, course_id ) VALUES (
58     (SELECT s.rowid FROM personne p CROSS JOIN soigneur s WHERE prenom = 'Jean' AND nom = 'Dupont'
59     AND s.personne_id = p.rowid),
60     (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'),
61     (SELECT rowid FROM course WHERE nom = 'Tour_de_France'));
62
63 -- insertion de produit
64 INSERT INTO produit ( numero, nom, indication, contre_indication, posologie ) VALUES ( 1547,
65     'Salbutamol',
66     'douleur_musculaire',
67     'ne_pas_administrer_en_dessous_de_20_ans',
68     '1_comprim _par_jour');
69
70 -- insertion de dose
71 INSERT INTO dose ( quantite, produit_id, soigneur_id ) VALUES ( '2.85',
72     (SELECT rowid FROM produit WHERE nom = 'Salbutamol'),
73     (SELECT s.rowid FROM personne p CROSS JOIN soigneur s WHERE prenom = 'Jean' AND nom = 'Dupont'
74     AND s.personne_id = p.rowid));
75
76 -- insertion de etape
77 INSERT INTO etape ( numeroordre, date, type, villedepart, villearrivee, course_id, equipegagnante_id,
78     coureurgagnant_id ) VALUES (
79     '1re_tape ',
```

```

74      '15-01-2015',
75      'Contre_lamontre_individuel',
76      'Lorient',
77      'Brest',
78      (SELECT rowid FROM course WHERE nom = 'Tour_de_France'),
79      (SELECT rowid FROM equipe WHERE nom = 'Vend e_U'),
80      (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
      AND c.personne_id = p.rowid));
81
82 INSERT INTO etape ( numeroordre, date, type, villeddepart, villearivee, course_id, equipegagnante_id,
      coureurgagnant_id ) VALUES (
83      '2nd_tape ',
84      '16-02-2015',
85      'Course_en_ligne',
86      'Brest',
87      'Lannion',
88      (SELECT rowid FROM course WHERE nom = 'Tour_de_France'),
89      NULL,
90      NULL);
91
92 INSERT INTO etape ( numeroordre, date, type, villeddepart, villearivee, course_id, equipegagnante_id,
      coureurgagnant_id ) VALUES (
93      '3 me _tape ',
94      '17-02-2015',
95      'Course_en_ligne',
96      'Lannion',
97      'Rennes',
98      (SELECT rowid FROM course WHERE nom = 'Tour_de_France'),
99      NULL,
100     NULL);
101
102 INSERT INTO etape ( numeroordre, date, type, villeddepart, villearivee, course_id, equipegagnante_id,
      coureurgagnant_id ) VALUES (
103     '4 me _tape ',
104     '18-02-2015',
105     'Course_en_ligne',
106     'Rennes',
107     'Nantes',
108     (SELECT rowid FROM course WHERE nom = 'Tour_de_France'),
109     NULL,
110     NULL);
111
112 -- insertion de classement
113 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
114     '1er',
115     (SELECT rowid FROM etape WHERE numeroordre = '1re_tape '),
116     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
      AND c.personne_id = p.rowid));
117
118 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
119     '3 me ',
120     (SELECT rowid FROM etape WHERE numeroordre = '3 me _tape '),
121     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
      AND c.personne_id = p.rowid));
122
123 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
124     '2nd',
125     (SELECT rowid FROM etape WHERE numeroordre = '2nd_tape '),
126     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
      AND c.personne_id = p.rowid));
127
128 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
129     '4 me ',
130     (SELECT rowid FROM etape WHERE numeroordre = '4 me _tape '),
131     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Thomas' AND nom = 'Voeckler'
      AND c.personne_id = p.rowid));
132
133 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
134     '2nd',
135     (SELECT rowid FROM etape WHERE numeroordre = '1re_tape '),
136     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
      AND c.personne_id = p.rowid));
137
138 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
139     '1er',
140     (SELECT rowid FROM etape WHERE numeroordre = '3 me _tape '),
141     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
      AND c.personne_id = p.rowid));
142
143 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
144     '1er',

```



```

145     (SELECT rowid FROM etape WHERE numeroordre = '2nd tape '),
146     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bernard' AND nom = 'Hinault'
      AND c.personne_id = p.rowid));
147
148 INSERT INTO classement ( place, etape_id, coureur_id ) VALUES (
149     '3 me ',
150     (SELECT rowid FROM etape WHERE numeroordre = '1re tape '),
151     (SELECT c.rowid FROM coureur c CROSS JOIN personne p WHERE prenom = 'Bryan' AND nom = 'Coquard'
      AND c.personne_id = p.rowid));

```

8.3 Data Update

```

1 UPDATE coureur
2 SET taille = 1.82
3 WHERE personne_id = (SELECT rowid FROM personne WHERE nom = 'Thomas' AND prenom = 'Voeckler');
4
5 UPDATE soigneur
6 SET nationalite = 'Allemand/Francais'
7 WHERE personne_id = (SELECT rowid FROM personne WHERE nom = 'Jean' AND prenom = 'Dupont');
8
9 UPDATE sponsor
10 SET adresse = '2bis rue Louis Armand, 75015 PARIS'
11 WHERE nom = 'Système U';

```

8.4 Table Joins

```

1 -- les coureurs de l' equipe 'Vendée U'
2 SELECT p.nom
3 FROM equipe e, coureur c, personne p
4 WHERE e.rowid = c.equipe_id AND p.rowid = c.personne_id AND e.nom = 'Vendée U';
5
6 -- jointure entre toutes les tables ( renommage de colonnes de meme ... nom )
7 SELECT DISTINCT per.nom AS "NOM",
8                 per.prenom AS "PRENOM",
9                 per.datedenaissance,
10                eq.nom AS "EQUIPE",
11                eq.budget,
12                cr.taille,
13                sg.nationalite,
14                spon.nom AS "SPONSOR",
15                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.villearrivee,
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;

```