Université catholique de Louvain
École polytechnique de Louvain

Analyse et conception de systèmes d'information
LINGE1322 – Projet 2016

Conception d'un système d'information LocatIn

Repository : github.com/SebStreb/LocatIn
Hébergement : locatin.herokuapp.com
Rapport en ligne : locatin.herokuapp.com/rapport

Sébastien Strebelle
8143-1300

Professeur :
Stéphane Faulkner

14 juin 2016

Table des matières

Chapitre 1

Introduction

Dans le cadre du cours LINGE1322 - Analyse et Conception de systèmes d'informations, il nous a été demandé de concevoir une base de données pour une société de location de voitures. Ce rapport décrit les différentes étapes de l'analyse, la conception et enfin l'implémentation d'un système d'information complet pour ce client.

Ce document est la version imprimable du rapport. La version en ligne permet une plus grande interactivité, avec une meilleur visibilité du code, des images de meilleur qualité ainsi qu'une vidéo de démonstration.

Chapitre 2

Analyse

2.1 Cas d'étude

Pour ce projet, nous avons reçu la description d'un cas d'étude. Il donne l'analyse de l'organisation des données d'une entreprise de location de voiture.

Dans cette entreprise, les véhicules sont classés par modèles. Ceux-ci sont caractérisées par des options, ainsi qu'une classe de tarification. Selon cette tarification, un véhicule aura un contrat d'assurance toujours proposé par le même assureur.

L'entreprise possède également 3 formules différentes (journée, semaine et week-end), définissant le temps d'une location ainsi qu'un nombre de kilomètre à ne pas dépasser avec la voiture. Le prix de base d'une location dépendra de la formule et de la classe de tarification.

Avant de pouvoir effectuer une location, un client doit faire une réservation. Au jour où il a demandé, si le véhicule souhaité est effectivement disponible le client a alors la possibilité de démarrer une location. Un contrat est créé, afin d'enregistrer la date de départ, le kilométrage initial de la voiture ainsi que le permis de conduire du client. Celui-ci a aussi la possibilité de payer une caution. Si le véhicule n'était pas disponible il peut effectuer une nouvelle réservation. Le client a également la possibilité d'annuler sa réservation à tout moment.

Après sa location, le client peut ramener la voiture à l'entreprise. Celle-ci note alors la date d'arrivée et la distance parcourue. Une facture est créée et le client a la possibilité de la payer. Le montant de la facture dépend du montant forfaitaire de la location, du paiement de la caution, du dépassement du nombre de kilomètres forfaitaires ainsi que du retard qu'il aurait pu avoir à la remise du véhicule, ceci selon sa classe de tarification. Le client est aussi dédommagé s'il avait dû changer de réservation à cause d'une indisponibilité.

2.2 Modèle de données

2.2.1 Modélisation des entités

Après l'analyse approfondie du cas d'étude, j'ai pu modéliser les données de l'entreprise en différentes entités possédant des relations entre elles au moyen d'un schéma entité-association. Le schéma créé est donné dans la figure 2.1.

Par rapport au cas d'étude, j'ai dû rajouter une entité *User*, afin de modéliser un opérateur de l'entreprise qui va utiliser le système d'information que j'ai créé par après.

2.2.2 Modélisation de la base de données

Sur base du schéma entité-association que j'ai élaboré, j'ai été capable de créer une base de données relationnelle représentant le modèle de données du cas d'étude. Le schéma relationnel de cette base de données est repris dans la figure 2.2.

Sur base de ce schéma, j'ai pu créer la base de données au moyen de requêtes *SQL*. J'avais initialement implémenté cette base de données au moyen de *SQLite*. Néanmoins, la faiblesse des interfaces de communication avec ce *SGBD* m'a convaincu d'utiliser un serveur *MySQL* pour ma base de données.

Le code *SQL* de création des tables est repris dans l'annexe A de ce rapport. Le code *SQL* permettant d'insérer les données d'exemple, quant-à-lui, est repris dans l'annexe B. La base de données *MySQL* en ligne du système d'information est accessible au moyen des identifiants suivant :

Host : eu-cdbr-west-01.cleardb.com

User : b1bb440ebb3969

Password : 74aea619

Database : heroku_577e100f766e1eb

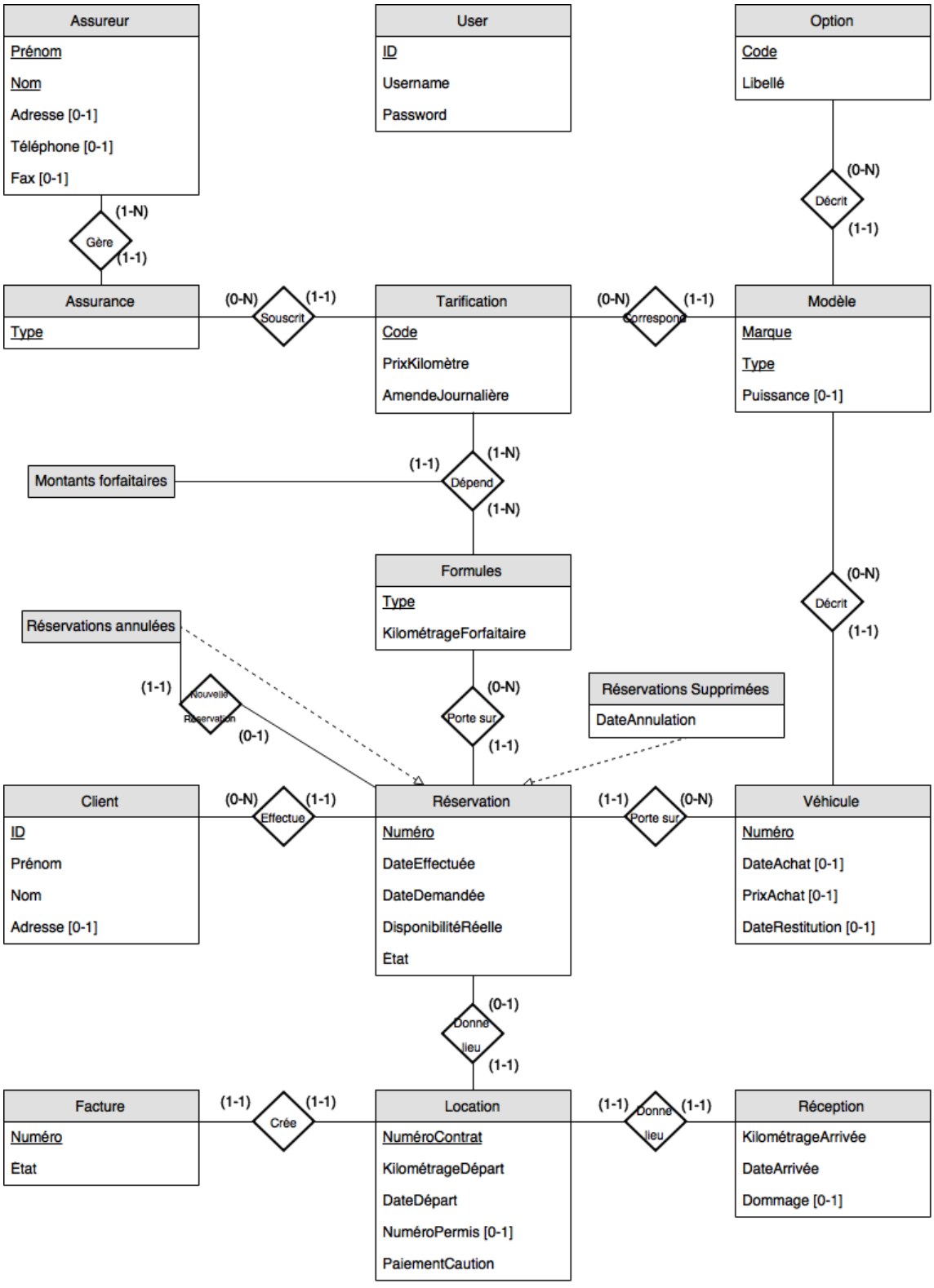

FIGURE 2.1 – Schéma entité-association

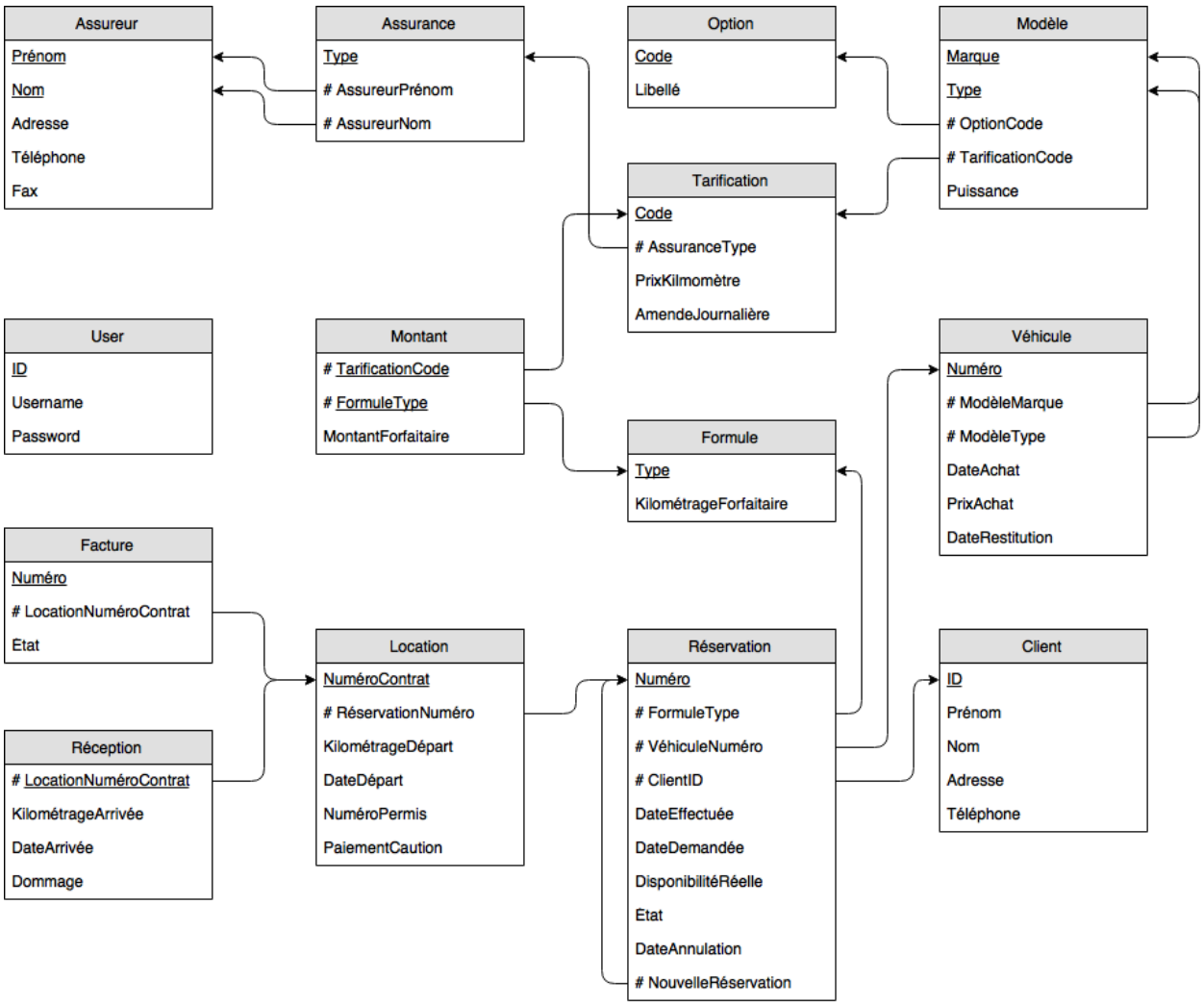

FIGURE 2.2 – Schéma relationnel de la base de données

2.3 Fonctionnalités

Une fois que le modèle de données était créé, j'ai voulu proposer un système d'information complet pour l'entreprise du cas d'étude. La première étape était alors d'analyser les besoins du client afin de trouver les fonctionnalités qui lui seraient pertinentes.

J'ai d'abord décidé d'orienter mon système d'information pour une utilisation par un opérateur de l'entreprise. Afin d'assurer la sécurité de l'entreprise, celui-ci devra se connecter avant de pouvoir effectuer des opérations. Trois types d'opérations me semblaient importantes à réaliser.

D'abord, l'opérateur doit pouvoir accueillir un client et répondre à ses besoins. Il doit pouvoir s'inscrire dans la base de données, réserver une voiture, gérer sa réservation, commencer une location, retourner un véhicule, payer ses factures ainsi que voir sa fidélité.

Ensuite, l'opérateur doit pouvoir avoir une vue d'ensemble sur l'état de l'entreprise. Pour cela, il doit pouvoir obtenir un rapport sur le catalogue de véhicules, ceux qui sont en stock, ceux qui sont en réservation, ceux qui sont en location et enfin un rapport sur l'historique des dernières réservations.

Enfin l'opérateur doit pouvoir modifier rapidement l'état de l'entreprise. Il peut vouloir donner l'accès à un nouvel opérateur, rajouter un véhicule, rajouter une nouvelle classe de tarification, modifier les tarifications des véhicules ou encore modifier les prix de l'entreprise.

Toutes ces fonctionnalités ne doivent pas pouvoir être accessibles à tous les opérateurs. Certaines opérations sensibles, comme les modifications de prix par exemple, ne peuvent être accessibles qu'à certains comptes afin de garantir la sécurité de l'entreprise.

Chapitre 3

Conception

3.1 Implémentation

J'avais à l'origine décidé d'implémenter ce système d'information au moyen d'une application java. Seulement, ce langage n'est pas le meilleur pour créer une interface ou interagir avec une base de données. J'ai ensuite voulu créer un site web avec PHP. Néanmoins en concevant le site, il me manquait de la souplesse du côté serveur pour concevoir ce que j'avais en tête. J'ai finalement décider de recréer toute l'application avec les dernières technologies du web, un serveur Node.js. La figure 3.1 montre la page d'accueil du système d'information créé. Il est hérgergé au moyen de *Heroku* à la page locatin.herokuapp.com

Comme spécifié dans les fonctionnalités, il faut se connecter pour pouvoir agir sur le site. Sinon les seules pages disponibles sont la page d'accueil, la page d'à propos ainsi que la page de connection. La barre de navigation propose trois menus déroulants pour les différentes opérations que voudrait faire l'opérateur.

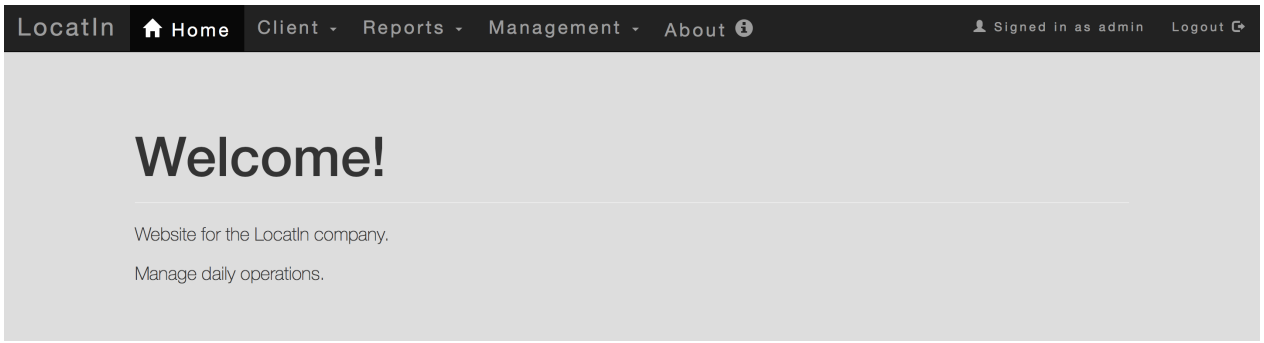

FIGURE 3.1 – Page d'accueil du système d'information

D'abord, lorsqu'un client se présente dans l'entreprise, l'opérateur peut trouver dans l'onglet *Client* les opérations qu'il pourrait vouloir faire avec lui.

— Inscription d'un nouveau client dans la base de données
— Recherche et réservation d'un véhicule
— Récupération d'une voiture et départ de location
— Retour d'une voiture et fin de location
— Paiement d'une facture
— Rapport sur la fidélité d'un client

Ensuite, l'onglet *Reports* donne un accès à une vision de l'état actuel de l'entreprise.

— Rapport sur le catalogue de véhicules à louer
— Rapport sur le stock actuel de véhicules
— Rapport sur les véhicules réservés
— Rapport sur les véhicules en location
— Rapport sur les dernières locations

Enfin, l'onglet *Management* permet de modifier l'état de l'entreprise. Ces opérations sensibles ne sont accessibles que pour l'utilisateur *admin*.

— Ajout de l'accès au système d'information pour un nouvel opérateur
— Ajout d'un nouveau véhicule
— Ajout d'une nouvelle classe de tarification
— Changement de classe de tarification pour un véhicule
— Modifications des prix de l'entreprise
— Exécuter une requête *SQL*

3.2 Requêtes importantes

Durant la création de ce système d'information, j'ai utilisé différentes requêtes *SQL*. Dans cette section, je vais montrer certaines requêtes particulièrement intéressantes.

3.2.1 Stock actuel de véhicules

```
1  SELECT
2          CONCAT(M.marque, ' ', M.type) AS 'Model',
3          COUNT(V.numero) AS 'Total number of cars',
4          O.libelle AS 'Options',
5          CONCAT(Mo.montantForfaitaire, ' €') AS 'Price for a day',
6          CONCAT(T.prixKilometre, ' €') AS 'Exceded kilometer price'
               ,
7          CONCAT(T.amendeJournaliere, ' €') AS 'Exceded day price'
8  FROM Vehicule V
9          LEFT JOIN Modele M ON V.modeleMarque = M.marque AND V.
               modeleType = M.type
10         LEFT JOIN Options O ON M.optionCode = O.code
11         LEFT JOIN Tarification T On M.tarificationCode = T.code
12         LEFT JOIN Montant Mo ON Mo.tarificationCode = T.code
13 WHERE
14         Mo.formuleType = 'Journée'
15         AND V.numero NOT IN(
16                 SELECT R.vehiculeNumero
17                 FROM Reservation R
18                 WHERE R.etat = 'Effectif'
19         )
20 GROUP BY M.marque, M.type
21 ;
```

Cette requête permet d'avoir une vision du stock actuel de véhicules dans l'entreprise. Dans cette requête je trie les véhicules pour ne garder que ceux dont il n'y a pas réservation effective.

3.2.2 Véhicules en location

```
1  SELECT
2          V.numero AS 'Car number',
```

```
3           CONCAT(V.modeleMarque, ' ', V.modeleType) AS 'Model',
4           R.numero AS 'Reservation number',
5           CONCAT(C.prenom, ' ', C.nom) AS 'Client',
6           CASE
7                   WHEN F.type = 'Journée' THEN
8                           R.dateDemandee + INTERVAL 1 DAY
9                   WHEN F.type = 'Semaine' THEN
10                          R.dateDemandee + INTERVAL 1 WEEK
11                  WHEN F.type = 'Week-end' THEN
12                          R.dateDemandee + INTERVAL 3 DAY
13          END AS 'Return date'
14  FROM Reservation R
15          LEFT JOIN Vehicule V ON R.vehiculeNumero = V.numero
16          LEFT JOIN Client C ON R.clientId = C.Id
17  WHERE
18          R.etat = 'Effectif'
19          AND EXISTS (
20                  SELECT L.numeroContrat
21                  FROM Location L
22                  WHERE L.reservationNumero = R.numero
23          )
24  ;
```

Cette requête permet de voir les véhicules qui sont actuellement en location. Dans cette requête, j'utilise une instruction *Case* afin de trouver la date de retour supposée selon la formule choisie. Je trie aussi les réservations en ne gardant que celles dont une location est liée.

3.2.3 Propositions de véhicules

```
1   SELECT
2           CONCAT(M.marque, ' ', M.type) AS 'Model',
3           COUNT(V.numero) AS 'Total number of cars',
4           O.libelle AS 'Options',
5           CONCAT(Mo.montantForfaitaire, ' €') AS 'Price for a day',
6           CONCAT(T.prixKilometre, ' €') AS 'Exceded kilometer price'
                ,
7           CONCAT(T.amendeJournaliere, ' €') AS 'Exceded day price'
8   FROM Vehicule V
9           LEFT JOIN Modele M ON V.modeleMarque = M.marque AND V.
                modeleType = M.type
```

```
10          LEFT JOIN Options O ON M.optionCode = O.code
11          LEFT JOIN Tarification T On M.tarificationCode = T.code
12          LEFT JOIN Montant Mo ON Mo.tarificationCode = T.code
13  WHERE
14          Mo.formuleType = 'Journée'
15          AND CONCAT(M.marque, ' ', M.type) = 'Audi A3'
16          AND O.code = 'O1'
17          AND V.numero NOT IN(
18                  SELECT R.vehiculeNumero
19                  FROM Reservation R
20                  WHERE R.etat = 'Effectif'
21          )
22  GROUP BY M.marque, M.type
23  ;
```

Cette requête permet de voir quels véhicules peuvent être proposés au client selon ses choix. Dans cette requête, j'utilise une fonction *Count* afin de compter le nombre de véhicules correspondants aux critères. Je groupe également les résultats par modèle de voiture au moyen de l'instruction *Group by*.

3.2.4 Factures d'un client

```
1   SELECT
2           V.factureNumero AS 'Bill Number',
3           CONCAT(C.prenom, ' ', C.nom) AS 'Client',
4           C.adresse AS 'Billing address',
5           V.montantBase + V.fraisPaiementCaution + V.fraisJoursSupps
                + V.fraisKmSupps - V.compensationAnnulation AS 'Total',
6           F.etat AS 'State'
7   FROM Client C
8           RIGHT JOIN VFacture V ON V.clientId = C.id
9           LEFT JOIN Facture F ON V.factureNumero = F.numero
10  WHERE C.id = 1
11  ORDER BY V.factureNumero DESC
12  ;
```

Cette requête permet de voir l'état de toutes le factures liées à un client. Dans cette requête, j'utilise la vue *VFacture* dont la définition est reprise dans l'annexe C du rapport afin de faciliter la requête. Cette vue donne la décomposition du calcul du montant total pour chaque facture.

3.2.5 Fidélité d'un client

```
1 SELECT
2         CONCAT(C.prenom, ' ', C.nom) AS 'Client',
3         COUNT(V.factureNumero) AS 'Number of reservations',
4         SUM(V.montantBase + V.fraisPaiementCaution + V.
              fraisJoursSupps + V.fraisKmSupps - V.
              compensationAnnulation) AS 'Sum'
5 FROM Client C
6         LEFT JOIN VFacture V ON V.clientId = C.id
7 WHERE C.id = 1
8 ;
```

Cette requête permet de voir la fidélité d'un client dans l'entreprise. Dans cette requête, j'utilise également la vue *VFacture*. De plus, j'utilise les fonction *Count* et *Sum* pour avoir respectivement le nombre de réservations et la somme totale dépensée par un client.

Chapitre 4

Conclusion

Pour conclure, je dirais que ce projet m'a permis d'approfondir mes connaissances des bases de données relationnelles. J'ai pu aussi expérimenter la programmation web.

Le système d'information créé pourrait être amélioré de plusieurs façons.

D'abord, il ne gère pas très bien la sécurité des données de l'entreprise. Les mots de passes sont stockés en clair dans la base de données, certains champs sont sensibles à l'injection *SQL*...

Ensuite, il serait intéressant de mieux gérer les privilèges dans le site. Actuellement, l'onglet *Management* est protégé en vérifiant simplement si l'utilisateur actuel de la session est nommé *admin*. Il faudrait pouvoir donner des privilèges à certains utilisateurs, leur permettant d'accéder à certaines pages ou non.

Enfin, certains formulaires sont peu exploités. Pour faciliter les requêtes *SQL*, je ne récupère que les champs les plus critiques. Également, il faudrait pouvoir montrer quels champs sont obligatoires.

Malgré ces points-là, ce système d'information a été pensé pour répondre aux besoins du client et devrait pouvoir répondre à ses attentes.

Annexe A

Création des tables

```
1  CREATE TABLE IF NOT EXISTS Assureur(
2          prenom VARCHAR(20) NOT NULL,
3          nom VARCHAR(20) NOT NULL,
4          adresse VARCHAR(60),
5          telephone VARCHAR(10),
6          fax VARCHAR(10),
7          PRIMARY KEY(prenom, nom)
8  );
9
10 CREATE TABLE IF NOT EXISTS Assurance(
11         type VARCHAR(20) PRIMARY KEY NOT NULL,
12         assureurPrenom VARCHAR(20) NOT NULL,
13         assureurNom VARCHAR(20) NOT NULL,
14         FOREIGN KEY (assureurPrenom, assureurNom) REFERENCES
               Assureur(prenom, nom)
15 );
16
17 CREATE TABLE IF NOT EXISTS Options(
18         code CHAR(2) PRIMARY KEY NOT NULL,
19         libelle VARCHAR(20) NOT NULL
20 );
21
22 CREATE TABLE IF NOT EXISTS Tarification(
23         code CHAR(2) PRIMARY KEY NOT NULL,
24         assuranceType VARCHAR(20) NOT NULL,
25         prixKilometre FLOAT NOT NULL,
26         amendeJournaliere FLOAT NOT NULL,
27         FOREIGN KEY (assuranceType) REFERENCES Assurance(type),
```

```
28          CHECK (prixKilometre > 0),
29          CHECK (amendeJournaliere > 0)
30  );
31
32  CREATE TABLE IF NOT EXISTS Modele(
33          marque VARCHAR(20) NOT NULL,
34          type VARCHAR(20) NOT NULL,
35          optionCode CHAR(2) NOT NULL,
36          tarificationCode CHAR(2) NOT NULL,
37          puissance FLOAT DEFAULT 150.0,
38          PRIMARY KEY (marque, type),
39          FOREIGN KEY (optionCode) REFERENCES Options(code),
40          FOREIGN KEY (tarificationCode) REFERENCES Tarification(
                code),
41          CHECK (puissance > 0)
42  );
43
44  CREATE TABLE IF NOT EXISTS Vehicule(
45          numero INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
46          modeleMarque VARCHAR(20) NOT NULL,
47          modeleType VARCHAR(20) NOT NULL,
48          dateAchat DATETIME DEFAULT NOW(),
49          prixAchat FLOAT DEFAULT 15000.0,
50          dateRestitution DATETIME,
51          FOREIGN KEY (modeleMarque, modeleType) REFERENCES Modele(
                marque, type),
52          CHECK (prixAchat > 0)
53  );
54
55  CREATE TABLE IF NOT EXISTS Formule(
56          type VARCHAR(20) PRIMARY KEY NOT NULL,
57          kilometrageForfaitaire FLOAT NOT NULL,
58          CHECK (kilometrageForfaitaire > 0)
59  );
60
61  CREATE TABLE IF NOT EXISTS Montant(
62          tarificationCode CHAR(2) NOT NULL,
63          formuleType VARCHAR(20) NOT NULL,
64          montantForfaitaire FLOAT NOT NULL,
65          PRIMARY KEY (tarificationCode, formuleType),
66          FOREIGN KEY (tarificationCode) REFERENCES Tarification(
                code),
67          FOREIGN KEY (formuleType) REFERENCES Formule(type),
```

```
68          CHECK (montantForfaitaire > 0)
69  );
70
71  CREATE TABLE IF NOT EXISTS Client(
72          id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
73          prenom VARCHAR(20) NOT NULL,
74          nom VARCHAR(20) NOT NULL,
75          adresse VARCHAR(60),
76          telephone VARCHAR(10)
77  );
78
79  CREATE TABLE IF NOT EXISTS Reservation(
80          numero INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
81          formuleType VARCHAR(20) NOT NULL,
82          vehiculeNumero INT NOT NULL,
83          clientId INT NOT NULL,
84          dateEffectuee DATETIME DEFAULT NOW(),
85          dateDemandee DATETIME DEFAULT NOW(),
86          disponibiliteReelle TINYINT DEFAULT 1,
87          etat VARCHAR(50),
88          dateAnnulation DATETIME,
89          nouvelleReservationNumero INT,
90          FOREIGN KEY (formuleType) REFERENCES Formule(type),
91          FOREIGN KEY (vehiculeNumero) REFERENCES Vehicule(numero),
92          FOREIGN KEY (clientId) REFERENCES Client(id),
93          FOREIGN KEY (nouvelleReservationNumero) REFERENCES
                Reservation(numero)
94  );
95
96  CREATE TABLE IF NOT EXISTS Location(
97          numeroContrat INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
98          reservationNumero INT NOT NULL,
99          kilometrageDepart FLOAT NOT NULL,
100         dateDepart DATETIME DEFAULT NOW(),
101         numeroPermis VARCHAR(10),
102         paiementCaution TINYINT DEFAULT 1,
103         FOREIGN KEY (reservationNumero) REFERENCES Reservation(
                numero),
104         CHECK (kilometrageDepart > 0)
105 );
106
107 CREATE TABLE IF NOT EXISTS Reception(
108         locationNumeroContrat INT PRIMARY KEY NOT NULL,
```

```
109         kilometrageArrivee FLOAT NOT NULL,
110         dateArrivee DATETIME DEFAULT NOW(),
111         dommage VARCHAR(50),
112         FOREIGN KEY (locationNumeroContrat) REFERENCES Location(
                numeroContrat),
113         CHECK (kilometrageArrivee > 0)
114 );
115
116 CREATE TABLE IF NOT EXISTS Facture(
117         numero INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
118         locationNumeroContrat INT,
119         etat VARCHAR(20) DEFAULT 'non-payé',
120         FOREIGN KEY (locationNumeroContrat) REFERENCES Location(
                numeroContrat)
121 );
```

Annexe B

Insertion des données d'exemple

```
1  INSERT OR REPLACE INTO Assureur (
2          prenom,
3          nom
4  ) VALUES (
5          'Jean-luc',
6          'Durant'
7  );
8
9  INSERT OR REPLACE INTO Assureur (
10         prenom,
11         nom
12 ) VALUES (
13         'Michel',
14         'Bubois'
15 );
16
17 INSERT OR REPLACE INTO Assurance (
18         type,
19         assureurPrenom,
20         assureurNom
21 ) VALUES (
22         'Omnium',
23         'Jean-luc',
24         'Durant'
25 );
26
27 INSERT OR REPLACE INTO Assurance (
28         type,
```

```
29          assureurPrenom,
30          assureurNom
31  ) VALUES (
32          'Kilométrage',
33          'Jean-luc',
34          'Durant'
35  );
36
37  INSERT OR REPLACE INTO Assurance (
38          type,
39          assureurPrenom,
40          assureurNom
41  ) VALUES (
42          'Annuelle',
43          'Michel',
44          'Bubois'
45  );
46
47  INSERT OR REPLACE INTO Option (
48          Code,
49          Libelle
50  ) VALUES (
51          'O1',
52          'Break - 5 portes'
53  );
54
55  INSERT OR REPLACE INTO Option (
56          Code,
57          Libelle
58  ) VALUES (
59          'O2',
60          'Berline - 5 portes'
61  );
62
63  INSERT OR REPLACE INTO Tarification (
64          code,
65          assuranceType,
66          prixKilometre,
67          amendeJournaliere
68  ) VALUES (
69          'T1',
70          'Omnium',
71          1,
```

```
72          30
73  );
74
75  INSERT OR REPLACE INTO Tarification (
76          code,
77          assuranceType,
78          prixKilometre,
79          amendeJournaliere
80  ) VALUES (
81          'T2',
82          'Kilométrage',
83          0.5,
84          40.0
85  );
86
87  INSERT OR REPLACE INTO Modele (
88          marque,
89          type,
90          optionCode,
91          tarificationCode
92  ) VALUES (
93          'Audi',
94          'A3',
95          'O1',
96          'T2'
97  );
98
99  INSERT OR REPLACE INTO Modele (
100         marque,
101         type,
102         optionCode,
103         tarificationCode
104 ) VALUES (
105         'BMW',
106         '520',
107         'O1',
108         'T1'
109 );
110
111 INSERT OR REPLACE INTO Vehicule (
112         modeleMarque,
113         modeleType
114 ) VALUES (
```

```
115             'Audi',
116             'A3'
117 );
118
119 INSERT OR REPLACE INTO Vehicule (
120             modeleMarque,
121             modeleType
122 ) VALUES (
123             'Audi',
124             'A3'
125 );
126
127 INSERT OR REPLACE INTO Vehicule (
128             modeleMarque,
129             modeleType
130 ) VALUES (
131             'BMW',
132             '520'
133 );
134
135 INSERT OR REPLACE INTO Formule (
136             type,
137             kilometrageForfaitaire
138 ) VALUES (
139             'Journée',
140             50
141 );
142
143 INSERT OR REPLACE INTO Formule (
144             type,
145             kilometrageForfaitaire
146 ) VALUES (
147             'Semaine',
148             250
149 );
150
151 INSERT OR REPLACE INTO Formule (
152             type,
153             kilometrageForfaitaire
154 ) VALUES (
155             'Week-end',
156             100
157 );
```

```
158
159 INSERT OR REPLACE INTO Montant (
160         tarificationCode,
161         formuleType,
162         montantForfaitaire
163 ) VALUES (
164         'T1',
165         'Journée',
166         100
167 );
168
169 INSERT OR REPLACE INTO Montant (
170         tarificationCode,
171         formuleType,
172         montantForfaitaire
173 ) VALUES (
174         'T2',
175         'Journée',
176         80
177 );
178
179 INSERT OR REPLACE INTO Montant (
180         tarificationCode,
181         formuleType,
182         montantForfaitaire
183 ) VALUES (
184         'T1',
185         'Semaine',
186         500
187 );
188
189 INSERT OR REPLACE INTO Montant (
190         tarificationCode,
191         formuleType,
192         montantForfaitaire
193 ) VALUES (
194         'T2',
195         'Semaine',
196         450
197 );
198
199 INSERT OR REPLACE INTO Montant (
200         tarificationCode,
```

```
201             formuleType,
202             montantForfaitaire
203     ) VALUES (
204             'T1',
205             'Week-end',
206             160
207     );
208
209     INSERT OR REPLACE INTO Montant (
210             tarificationCode,
211             formuleType,
212             montantForfaitaire
213     ) VALUES (
214             'T2',
215             'Week-end',
216             160
217     );
218
219     INSERT OR REPLACE INTO Client (
220             prenom,
221             nom
222     ) VALUES (
223             'Bernad',
224             'Duchêne'
225     );
226
227     INSERT OR REPLACE INTO Client (
228             prenom,
229             nom
230     ) VALUES (
231             'Monique',
232             'Ale'
233     );
234
235     INSERT OR REPLACE INTO Reservation (
236             formuleType,
237             vehiculeNumero,
238             clientId,
239             etat,
240             dateAnnulation,
241             nouvelleReservationNumero
242     ) VALUES (
243             'Journée',
```

```
244             1,
245             1,
246             'Effectif',
247             null,
248             null
249 );
250
251 INSERT OR REPLACE INTO Reservation (
252             formuleType,
253             vehiculeNumero,
254             clientId,
255             etat,
256             dateAnnulation,
257             nouvelleReservationNumero
258 ) VALUES (
259             'Journée',
260             2,
261             1,
262             'Effectif',
263             null,
264             null
265 );
266
267 INSERT OR REPLACE INTO Reservation (
268             formuleType,
269             vehiculeNumero,
270             clientId,
271             etat,
272             dateAnnulation,
273             nouvelleReservationNumero
274 ) VALUES (
275             'Semaine',
276             3,
277             2,
278             'Supprimée',
279             NOW() + INTERVAL 1 DAY,
280             null
281 );
282
283 INSERT OR REPLACE INTO Reservation (
284             formuleType,
285             vehiculeNumero,
286             clientId,
```

```
287             etat,
288             dateAnnulation,
289             nouvelleReservationNumero
290 ) VALUES (
291             'Week-end',
292             3,
293             1,
294             'Terminée',
295             null,
296             null
297 );
298
299 INSERT OR REPLACE INTO Reservation (
300             formuleType,
301             vehiculeNumero,
302             clientId,
303             etat,
304             dateAnnulation,
305             nouvelleReservationNumero
306 ) VALUES (
307             'Journée',
308             2,
309             1,
310             'Terminée',
311             null,
312             null
313 );
314
315 UPDATE Reservation SET etat = 'Annulée',
316             nouvelleReservationNumero = 5
317 WHERE numero = 1;
318
319 INSERT OR REPLACE INTO Location (
320             reservationNumero,
321             kilometrageDepart,
322             dateDepart,
323             paiementCaution
324 ) VALUES (
325             1,
326             20000,
327             NOW(),
328             1
329 );
```

```
330
331 INSERT OR REPLACE INTO Location (
332         reservationNumero,
333         kilometrageDepart,
334         dateDepart,
335         paiementCaution
336 ) VALUES (
337         4,
338         50000,
339         NOW(),
340         0
341 );
342
343 INSERT OR REPLACE INTO Location (
344         reservationNumero,
345         kilometrageDepart,
346         dateDepart,
347         paiementCaution
348 ) VALUES (
349         5,
350         18000,
351         NOW(),
352         1
353 );
354
355 INSERT OR REPLACE INTO Reception (
356         locationNumeroContrat,
357         kilometrageArrivee,
358         dateArrivee
359 ) VALUES (
360         2,
361         50050,
362         NOW() + INTERVAL 1 DAY
363 );
364
365 INSERT OR REPLACE INTO Reception (
366         locationNumeroContrat,
367         kilometrageArrivee,
368         dateArrivee
369 ) VALUES (
370         3,
371         20050,
372         NOW() + INTERVAL 20 HOUR
```

```
373 );
374
375 INSERT OR REPLACE INTO Facture (
376         locationNumeroContrat
377 ) VALUES (
378         2
379 );
380
381 INSERT OR REPLACE INTO Facture (
382         locationNumeroContrat
383 ) VALUES (
384         3
385 );
```

Annexe C

Vue pour les factures

```
1  CREATE OR REPLACE VIEW VFacture AS
2          SELECT
3                  Fa.numero AS factureNumero,
4                  L.numeroContrat AS numeroContrat,
5                  C.id AS clientId,
6                  M.montantForfaitaire AS montantBase,
7                  CASE
8                          WHEN L.paiementCaution = 0 THEN
9                                  (M.montantForfaitaire / 100) * 3
10                         ELSE
11                                 0
12                 END AS fraisPaiementCaution,
13                 CASE
14                         WHEN Fo.type = 'Journée' AND Rec.
                               dateArrivee > L.dateDepart + INTERVAL 1
                               DAY THEN
15                                 (DATEDIFF(Rec.dateArrivee, L.
                                       dateDepart) - 1) * T.
                                       amendeJournaliere
16                         WHEN Fo.type = 'Semaine' AND Rec.
                               dateArrivee > L.dateDepart + INTERVAL 1
                               WEEK THEN
17                                 (DATEDIFF(Rec.dateArrivee, L.
                                       dateDepart) - 7) * T.
                                       amendeJournaliere
18                         WHEN Fo.type = 'Week-end' AND Rec.
                               dateArrivee > L.dateDepart + INTERVAL 3
                               DAY THEN
```

```
19                                  (DATEDIFF(Rec.dateArrivee, L.
                                        dateDepart) - 3) * T.
                                        amendeJournaliere
20                          ELSE
21                                  0
22                  END AS fraisJoursSupps,
23                  CASE
24                          WHEN Rec.kilometrageArrivee - L.
                                kilometrageDepart > Fo.
                                kilometrageForfaitaire THEN
25                                  (Rec.kilometrageArrivee - L.
                                        kilometrageDepart - Fo.
                                        kilometrageForfaitaire) * T.
                                        prixKilometre
26                          ELSE
27                                  0
28                  END AS fraisKmSupps,
29                  CASE
30                          WHEN EXISTS (SELECT R.numero FROM
                                Reservation R WHERE R.
                                nouvelleReservationNumero = Res.numero)
                                THEN
31                                  T.amendeJournaliere
32                          ELSE
33                                  0
34                  END AS compensationAnnulation
35          FROM Client C, Facture Fa, Formule Fo, Location L, Montant
                M, Modele Mo, Reception Rec, Reservation Res,
                Tarification T, Vehicule V
36          WHERE
37                  Fa.locationNumeroContrat = L.numeroContrat AND
38                  Rec.locationNumeroContrat = L.numeroContrat AND
39                  L.reservationNumero = Res.numero AND
40                  Res.formuleType = Fo.type AND
41                  Res.vehiculeNumero = V.numero AND
42                  Res.clientId = C.id AND
43                  M.tarificationCode = T.code AND
44                  M.formuleType = Fo.type AND
45                  V.modeleMarque = Mo.marque AND
46                  V.modeleType = Mo.type AND
47                  Mo.tarificationCode = T.code
48  ;
```