Base de données

Stéphane Devismes

26 août 2020

Table des matières

1 La base de données *ZOO* **5**
- 1.1 Position du problème . . . 5
- 1.2 Les tables . . . 5
- 1.3 Exercices . . . 6
  - 1.3.1 Schéma relationnel . . . 6
  - 1.3.2 Compréhension des relations . . . 6
  - 1.3.3 Requêtes . . . 6
  - 1.3.4 Cohérence . . . 7

2 La grande bouffe **9**
- 2.1 Position du problème . . . 9
  - 2.1.1 Schémas des relations . . . 9
  - 2.1.2 Expression de requêtes . . . 10

3 Agence de voyages **11**
- 3.1 Position du problème . . . 11
- 3.2 Schéma des relations . . . 12
- 3.3 Questions . . . 13

4 Conception de Bases de Données **15**
- 4.1 Exercice 1 . . . 15
  - 4.1.1 Problème . . . 15
  - 4.1.2 Questions . . . 15
- 4.2 Exercice 2 . . . 16
  - 4.2.1 Problème . . . 16
  - 4.2.2 Questions . . . 16
- 4.3 Exercice 3 . . . 16
  - 4.3.1 Problème . . . 16
  - 4.3.2 Questions . . . 17
- 4.4 Exercice 4 . . . 17
  - 4.4.1 Problème . . . 17
  - 4.4.2 Questions . . . 17
- 4.5 Exercice 5 . . . 18
  - 4.5.1 Problème . . . 18
  - 4.5.2 Questions . . . 19

5 Conception en SQL **21**
- 5.1 La base de donnée *ZOO* . . . 21
- 5.2 Schéma relationnel . . . 21
  - 5.2.1 Les relations . . . 21
  - 5.2.2 Domaines . . . 22

5.2.3 Contraintes d'intégrité . . . 22
5.3 Création . . . 22
5.4 Remplir la base . . . 22
5.5 Déclencheurs (triggers) . . . 22

6 JDBC/Servlet **25**
6.1 Mise en place . . . 25
6.2 Utilisation de `ant` . . . 26
6.3 Questions . . . 27
6.4 La base de donnée *ZOO* . . . 27
6.4.1 Les relations . . . 27
6.4.2 Domaines . . . 27
6.4.3 Contraintes d'intégrité . . . 28

7 Normalisation **29**
7.1 Exercice 1 . . . 29
7.2 Exercice 2 . . . 29
7.3 Exercice 3 . . . 30
7.4 Exercice 4 . . . 30
7.5 Exercice 5 . . . 30

A Découvrir Oracle **31**
A.1 Connexion à Oracle . . . 31
A.2 Travailler depuis chez soi . . . 32
A.3 Exécution d'une requête en ligne . . . 32
A.4 Exécution d'une requête à partir d'un fichier . . . 32
A.5 Stockage des résultats des requêtes dans un fichier . . . 33
A.6 Requêtes paramétrées . . . 33

Chapitre 1

La base de données *ZOO*

1.1 Position du problème

Le directeur d'un *zoo* a informatisé la gestion de son établissement. Dans ce zoo, on trouve des animaux répertoriés par *type* (lion, léopard, girafe, escargot, ...). Chaque animal possède un *nom* (Charly, Arthur, Enzo, ...) qui l'identifie de façon unique, une *date de naissance* et un *pays d'origine*. On retient également les *maladies* que chaque animal a contractées depuis son arrivée au zoo. Les animaux sont logés dans des *cages*. Chaque cage peut recevoir un ou plusieurs animaux. Certaines cages peuvent être inoccupées. Une cage correspond à une certaine *fonctionnalité*, qui n'est pas forcément liée à un type d'animal donné (par exemple une cage peut convenir à la fois aux girafes, aux éléphants et aux fauves, une autre aux grands oiseaux, ...). Une cage est identifié par un *numéro*, elle est située dans une *allée*, identifiée aussi par un numéro. Des personnes sont employées par le zoo pour entretenir les cages et soigner les animaux. Chaque employé est identifié par son *nom*, et on connaît la ville où il réside. Il existe deux types de *postes* pour les employés : *gardien* ou *responsable*. Chaque employé est affecté à un unique poste : soit gardien, soit responsable. Un gardien s'occupe d'une ou plusieurs cages, et un responsable a la charge de toutes les cages de une ou plusieurs allées. Une *allée* est sous la responsabilité d'un seul employé et toute cage occupée par au moins un animal est gardée par au moins un gardien ; les cages inoccupées ne sont pas gardées.

1.2 Les tables

Ci-dessous nous fournissons les tables de la base de données du zoo.

LesAnimaux					
`nomA`	`sexe`	`type`	`pays`	`anNais`	`noCage`
Charly	mâle	lion	Kenya	1990	12
Arthur	mâle	ours	France	1980	1
Chloé	femelle	pie	France	1991	3
Milou	mâle	léopard	France	1993	11
Tintin	mâle	léopard	France	1993	11
Charlotte	femelle	lion	Kenya	1992	12

LesCages		
`noCage`	`fonction`	`noAllée`
11	enclos	10
1	fosse	1
2	aquarium	1
3	volière	2
4	grand aquarium	1
12	enclos	10

LesResponsables	
`noAllée`	`nomE`
10	Peyrin
1	Adiba
2	Voiron

LesMaladies	
nomA	nomM
Charly	rage de dents
Charly	grippe
Milou	angine
Chloé	grippe

LesEmployés	
nomE	adresse
Peyrin	Nouméa
Berrut	Sartène
Sicard	Calvi
Voiron	Pointe à Pitre
Scholl	Ushuaïa
Adiba	Papeete

LesGardiens	
noCage	nomE
11	Scholl
12	Berrut
12	Sicard
11	Berrut
11	Sicard
1	Scholl
3	Scholl
12	Scholl

1.3 Exercices

1.3.1 Schéma relationnel

1. Donnez la clé primaire de chacune des tables (relations) de la base *Zoo*.
2. Donnez la spécification des relations `LesAnimaux` et `LesCages`.

1.3.2 Compréhension des relations

1. Indiquez les n-uplets que l'on doit insérer dans la ou les relation(s) concernée(s), pour représenter les informations contenues dans le texte ci-dessous :
   — Chloé, qui est au zoo depuis longtemps, vient de contracter une rage de dents en même temps que la grippe.
   — Brigitte est une libellule femelle qui est arrivée au zoo il y a quelques jours. Les libellules sont des animaux de type "insecte archiptère". Brigitte a été placée dans la cage numéro 3. Depuis son arrivée au zoo, elle n'a contracté aucune maladie.
   — Mme Bruandet, résidant à Papeete, vient d'être employée par le zoo. Elle a été affectée au gardiennage des cages 4, 11, et 12.
   — On vient de construire une cage piscine, qui conviendra par exemple aux tortues. Elle a le numéro 40 et est située au bout de l'allée 10.
2. On souhaite maintenant pouvoir stocker pour chaque animal et pour chacune des maladies qu'il contracte, la date à laquelle il a contracté cette maladie. Quel est l'impact de cette modification sur le modèle relationnel précédent ?
3. Quelles sont les caractéristiques de l'énoncé qui ne sont pas directement traduites par les relations ?

1.3.3 Requêtes

Donnez l'expression relationnelle, l'expression SQL, ainsi que le résultat des requêtes données ci-dessous :

1. Le nom des animaux du zoo.
2. Les fonctionnalités disponibles dans le zoo.
3. Les noms des *léopards*.

4. Les maladies contractées au moins une fois par des animaux du zoo.
5. Les noms et numéros de cage des animaux *mâles* qui sont originaires du *Kenya* et dont la date de naissance est antérieure à *1992*.
6. Une requête produisant l'affichage suivant :

```
Peyrin vit à Nouméa
Berrut vit à Sartène
Sicard vit à Calvi
Voiron vit à Pointe à Pitre
Scholl vit à Ushuaia
Adiba vit à Papeete
```

7. Le nom et l'âge des animaux en *2020*.
8. Le nom des gardiens qui habitent *Ushuaïa*.
9. La fonctionnalité et le nom du gardien des cages gardées par un employé habitant *Calvi*.
10. Le nom des animaux ainsi que des employés qui en sont soit les gardiens soit les responsables.
11. Le nom des gardiens gardant tous les animaux.
12. Les noms et types des animaux qui n'ont jamais été malades.
13. Les noms des animaux originaires du *Kenya* ayant déjà contractés une *grippe*.
14. Les numéros et fonctionnalités des cages qui sont inoccupées.
15. Donner pour chaque animal mâle l'ensemble des maladies qu'il a contractées (ensemble des couples nom d'animal, nom de maladie).
16. Les numéros et fonctionnalités des cages qui sont partagées par des animaux de types différents. En d'autres termes, ce sont les cages qui contiennent au moins deux animaux de types différents.
17. Les noms des responsables et les noms des gardiens de *Charly*.
18. Le nom et le pays d'origine de l'animal doyen du zoo (il peut y en avoir plusieurs).
19. Le nom, le type et l'année de naissance des animaux qui ont contracté toutes les maladies (connues) du zoo.
20. Le nom, le type et le pays d'origine des animaux qui partagent la cage de *Charly*.
21. Le nom et l'adresse des employés qui sont gardiens d'animaux de tous types, on fait référence aux types des animaux du zoo.

1.3.4 Cohérence

Ecrire les requêtes (relationnelles et SQL) qui permettent de vérifier que la base de données satisfait les règles de cohérence suivantes :

1. Un employé est soit un gardien, soit un responsable.
2. Il n'y a pas de gardien affecté à des cages vides.

Chapitre 2

La grande bouffe

2.1 Position du problème

Une maîtresse de maison a constitué une base de données relationnelle sur les *amis* qu'elle invite. Elle représente, pour chaque repas identifié par une *date*, les noms des *plats* qui ont été servis ainsi que leur *type*, et pour chaque plat, le *vin* qui l'accompagnait. Elle connaît les *plats préférés* de tous ses amis. Un invité peut être ou non un ami et inversement.

2.1.1 Schémas des relations

Cette maîtresse de maison fort avisée définit le schéma de relations suivant (les identifiants des relations sont les attributs soulignés). Dans LesRepas, on parle des invités (amis ou non). Dans LesPréférences, on parle des amis (invités ou non).

LesRepas (dateR, nomI)
$(d, i) \in$ LesRepas $\Leftrightarrow$ la personne de nom i, a été invitée au repas identifié par la date d

LeMenu (dateR, nomP, nomV)
$(d, p, v) \in$ LeMenu $\Leftrightarrow$ le plat p accompagné par le vin v a été servi au repas d

LesPreferences (nomA, nomP)
$(n, p) \in$ LesPreferences $\Leftrightarrow$ l'ami de nom n aime le plat de nom p

LesPlats (nomP, typeP)
$(p, t) \in$ LesPlats $\Leftrightarrow$ le plat p est du type t

domaine(dateR) = date
domaine(nomI) = domaine(nomA) = {"Pierre", "Paul", etc..}
domaine(nomP) = {"Médaillon langouste", "Mousse chocolat", etc...}
domaine(nomV) = {"don pérignon 1991", "chateau la-pompe 1920", etc..}
domaine(typeP) = {"entrée chaude", "dessert", etc..}

LeMenu[dateR] = LesRepas[dateR]
LesPreferences[nomP] $\subseteq$ LesPLats[nomP]
LeMenu[nomP] $\subseteq$ LesPlats[nomP]

2.1.2 Expression de requêtes

Exprimer en algèbre relationnelle et dans le langage SQL les requêtes ci-dessous. Les requêtes devront construire des résultats sans répétition de valeurs. Dans le langage SQL, la clause `distinct` sera utilisée uniquement lorsque nécessaire.

1. Donner les noms des invités du repas organisé le 31 Décembre 2004.
2. Donner les noms des vins qui ont été servis avec un médaillon de langouste.
3. Donner les noms des plats (avec le vin qui les accompagne) qui ont été servis le 21 octobre 2003.
4. Donner les noms des invités à qui il a été servi au moins une fois du foie gras.
5. Donner les noms des amis qui n'ont jamais été invités.
6. Donner les repas (date, plats et vins servis) auxquels Thomas et Patrick ont été invités (ensemble).
7. Donner les noms des amis qui ont eu, au moins une fois, un plat de leurs préférences.
8. Donner les noms des invités qui n'aiment que les desserts.
9. Donner les noms des amis qui n'ont jamais eu un de leurs plats préférés.
10. Donner les noms des invités qui ont mangé du foie gras, mais qui n'aiment pas cela.
11. Pour chaque invité, donner le nombre de repas auxquels il a été convié.
12. Donner le nombre moyen d'invités par repas.
13. Donner les noms des desserts qui ont été servis au moins 2 fois.
14. Pour l'(les) ami(s) invité(s) le plus souvent, donner son(leur) nom ainsi que ses(leurs) plats préférés.
15. Donner les noms des amis qui aiment tous les types de plats, c'est-à-dire au moins un plat de chaque type.
16. Pour chaque ami, donner le nombre de repas auxquels il a été convié.
17. Notre maîtresse de maison décide qu'elle invitera à son prochain repas, qui aura lieu le 1er janvier 2010, tous les amis qui n'ont pas encore été invités. Former la requête qui prédit comment la table LesRepas sera mise à jour après le repas du 1er janvier 2010.

Chapitre 3

Agence de voyages

3.1 Position du problème

Une agence de voyage a informatisé la gestion des voyages qu'elle propose (itinéraires, monuments visités, réservations, etc.).

La base de données a été construite à partir du cahier des charges suivant :

Les circuits : Un circuit est identifié par un numéro, il est décrit par une ville de départ, une ville d'arrivée et une séquence non vide d'étapes. Une étape se déroule pendant un nombre donné de jours, dans une ville donnée. Au cours de chaque étape, tous les monuments de la ville, lorsqu'il y en a, sont visités. Les monuments des villes de départ et d'arrivée ne sont pas visités.

Un même circuit ne repasse jamais plusieurs fois dans la même ville étape, mais il peut arriver qu'une ville départ ou arrivée figure aussi dans l'ensemble des villes étapes.

Les monuments et les villes sont identifiés par leur nom.

Un circuit peut être programmé plusieurs fois, à des dates différentes, pour un nombre de places qui peut varier selon la programmation. Le prix d'un circuit est fixé, toujours le même quelque soit sa programmation. Un circuit dure un nombre de jours égal à la somme des durées de chacune de ses étapes.

Les réservations : Une réservation, identifiée par un numéro, est effectuée pour le compte d'un client (identifié par son nom) et concerne une programmation d'un circuit. Plusieurs places pour la même programmation du même circuit peuvent être réservées en une seule fois.

3.2 Schéma des relations

Le schéma retenu pour la base de données est constitué des relations suivantes (les identifiants sont soulignés) :

LESVILLES (NOMV, PAYS)
$(n, p) \in$ LESVILLES $\Leftrightarrow$ la ville dont le nom est n, est située dans le pays p.

LESMONUMENTS (NOMM, NOMV, PRIX)
$(nm, nv, p) \in$ LESMONUMENTS $\Leftrightarrow$ le monument identifié par son nom nm, est situé dans la ville nv. Son prix de visite est p (en euros). Certaines villes n'ont pas de monument à visiter.

LESCIRCUITS (NUMC, VDEP, VARR, PRIX)
$(n, vd, va, pr) \in$ LESCIRCUITS $\Leftrightarrow$ le circuit touristique identifié par le numéro n, part de la ville vd et se termine dans la ville va. Son prix est de pr, qui ne prend pas en compte le prix des monuments visités. La ville de départ représente le point de rendez-vous avec les accompagnateurs.

LESETAPES (NUMC, RANG, VETAPE, NBJOURS)
$(n, r, ve, nbj) \in$ LESETAPES $\Leftrightarrow$ la $r^{\text{ième}}$ étape du circuit n se déroule dans la ville ve, où le séjour est de nbj jours. On fait comme hypothèse que lorsqu'une ville est dans un circuit, tous ses monuments sont visités. De plus, $r \geq 1$, et les villes de départ et d'arrivée (VARR et VDEP de LESCIRCUITS) sont dans LESETAPES lorsqu'elles sont visitées.

LESPROGRAMMATIONS (NUMC, DATEDEP, NBPLACES)
$(n, d, nbl) \in$ LESPROGRAMMATIONS $\Leftrightarrow$ le circuit identifié par le numéro n, programmé à la date d dispose encore de nbl places disponibles. Le même circuit peut être programmé à différentes dates.

LESRESERVATIONS (NUMR, NOMC, NUMC, DATEDEP, NBRES)
$(nr, no, nc, d, nbr) \in$ LESRESERVATIONS $\Leftrightarrow$ le client de nom no, a effectué une réservation identifiée par nr, sur le circuit nc programmé à la date d. Il a réservé nbr places.

Les domaines associés sont :

domaine(NOMC) = {'Bonemine', 'Corto', etc.}
domaine(NOMM) = {'Tower Bridge', 'Madame Tussau', etc.}
domaine(VETAPE) = domaine(VDEP) = domaine(VARR) = domaine(NOMV) = {'Paris', 'Florence', 'Briançon', etc.}
domaine(PAYS) = {'Italie', 'Finlande', 'France', etc.}
domaine(NUMC) = domaine(NUMR) = domaine(RANG) = domaine(NBRES) = domaine(NBJOURS) = domaine(PRIX) = entiers > 0
domaine(NBPLACES) = entiers ≥ 0
domaine(DATEDEP) = dates

Les contraintes d'intégrité référentielle sont :

LESRESERVATIONS[NUMC, DATEDEP] $\subseteq$ LESPROGRAMMATIONS[NUMC, DATEDEP]
LESPROGRAMMATIONS[NUMC] $\subseteq$ LESCIRCUITS[NUMC]
LESETAPES[NUMC] $\subseteq$ LESCIRCUITS[NUMC]
LESETAPES[VETAPE] $\subseteq$ LESVILLES[NOMV]
LESCIRCUITS[VDEP] $\subseteq$ LESVILLES[NOMV]
LESCIRCUITS[VARR] $\subseteq$ LESVILLES[NOMV]
LESMONUMENTS[NOMV] $\subseteq$ LESVILLES[NOMV]

Remarque 1

- *On dit qu'un circuit passe par une ville v, lorsque v est une des étapes, ou la ville arrivée, ou la ville de départ (ou inclusif) ;*
- *On dit qu'un circuit visite une ville lorsque celle-ci est une fois une étape de ce circuit (Cf. 3.1).*

3.3 Questions

Remarque 2 *Afin de ne programmer aucune constante les requêtes devront être paramétrées. Les résultats seront ordonnés et sans répétition de valeurs. Par exemple, la requête ci-dessous est paramétrée par un numéro du circuit et un nom de ville :*

```
select * from Circuit where nc = &num and vdep = '&villed' ;
```

L'usage des requêtes paramétrées peut être facilité en utilisant les instructions prompt *et* accept *proposées par SQL (voir l'annexe ou la documentation en ligne :* help prompt *et* help accept*).*

Traduisez en algèbre relationnelle et en SQL les requêtes suivantes :

1. Donner le numéro, les villes de départ et d'arrivée des circuits qui démarrent après une date donnée.
2. Donner les noms des clients qui ont réservé au moins un circuit qui passe par un pays donné.
3. Donner le nom des clients qui ne visitent aucun monument.
4. Donner le numéro, le prix de base (sans tenir compte du prix des monuments visités) et la date de départ des circuits qui ont encore des places disponibles et dont le nombre de jours est inférieur ou égal à un entier donné.
5. Donner le numéro, le nombre total de jours et le prix de base des circuits qui n'ont aucune réservation.
6. Donner les noms des villes visitées par Milou (c'est-à-dire les villes étapes).
7. Donner les noms des villes et des monuments visités par Milou. Le résultat devra faire apparaître toutes les villes étapes, même celles où aucun monument n'est visité.
8. Donner les noms des villes visitées par Milou avec le nombre de monuments dans chaque ville.
9. Pour chacune des villes visitées par Milou, donner le prix total pour visiter tous les monuments de cette ville (ce prix est nul s'il n'y a aucun monument visité).
10. Donner les noms des pays, des villes et des monuments visités par un client donné. Le résultat devra faire apparaître aussi les pays et les villes sans monument.
11. Donner la liste des numéros des circuits, en ordre croissant, qui passent dans toutes les villes d'un pays donné.
12. Pour chaque circuit restant dans le même pays, donner ce pays et le numéro du circuit.
13. Pour tous les circuits qui passent dans un pays donné, donner les dates de toutes leurs programmations et, à chaque fois, la date d'arrivée dans le pays en question.
14. Etablir un récapitulatif faisant apparaître, pour chaque circuit :
    — numéro du circuit
    — coût total (prix + visite des monuments)
    — taux de remplissage moyen
    Taux moyen de remplissage = taux moyen de remplissage d'un circuit sur toutes les programmations.
    Taux de remplissage pour une programmation de circuit = nombre de places réservées / nombre de places offertes.
15. Pour chaque circuit, donné par son numéro, quelle est la date de programmation pour laquelle il y a le plus de places réservées ?

Chapitre 4

Conception de Bases de Données

4.1 Exercice 1

4.1.1 Problème

Une cinémathèque désire automatiser son système d'information. Celui-ci concerne au maximum 500 films, 20 réalisateurs et 40 acteurs principaux. En moyenne, on constate les points suivants :

— Un film a deux acteurs principaux.
— Un réalisateur a tourné 25 films.
— Un acteur a tourné dans 25 films.

En outre, on a noté les règles de gestion suivantes :

— Tout film a un et un seul réalisateur.
— Tout réalisateur a fait au moins un film.
— Tout film a au moins un acteur principal.
— Tout acteur principal a joué dans au moins un film.

Ci-dessous, vous avez les indications sur les données nécessaires :

Attribut	**Libellé**	**Domaine**
NOFILM	Identifiant du film	Numérique
TITRE	Titre du film	Chaîne de caractères
NOR	Identifiant d'un réalisateur	Numérique
NOMR	Nom d'un réalisateur	Chaîne de caractères
PRENOMR	Prénom d'un réalisateur	Chaîne de caractères
DATEENG	Date d'engagement d'un acteur (principal) dans un film	Date
NOA	Identifiant d'un acteur	Numérique
NOMA	Nom d'un acteur	Chaîne de caractères
PRENOMA	Prénom d'un acteur	Chaîne de caractères

4.1.2 Questions

— Donner le diagramme de classes correspondant à l'énoncé.
— Traduire le diagramme en schéma relationnel.

4.2 Exercice 2

4.2.1 Problème

Une auto-école comprend des moniteurs et des véhicules. Elle prépare les élèves à passer un permis d'un type donné. A chaque type de permis correspond un ensemble de caractéristiques rassemblées sous forme d'un libellé. Les moniteurs donnent des leçons aux élèves. On souhaite stocké le nom et le prénom des moniteurs ainsi que ceux des élèves. Les moniteurs et les élèves sont identifiés par un numéro unique. Chaque leçon est donnée par un seul moniteur, pour un seul élève à la fois et une certaine durée. Un élève ne passe qu'un seul type de permis et n'apprend à conduire que sur un seul véhicule. Chaque véhicule a un nom (clio, audi `TT`) et est identifié par son immatriculation. Le prix d'un permis est forfaitaire, quel que soit le nombre de leçons.

4.2.2 Questions

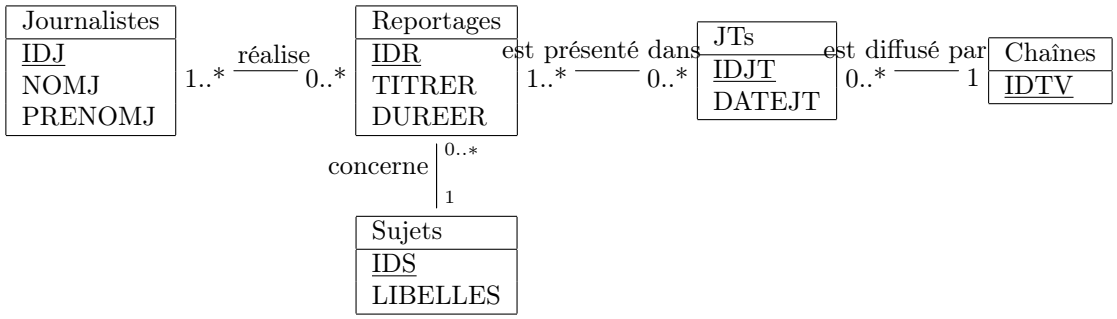
— Donner le diagramme de classes correspondant à l'énoncé.
— Traduire le diagramme en schéma relationnel.

4.3 Exercice 3

4.3.1 Problème

L'*AFP* (*Agence Flamande de Presse*) est une agence de presse spécialisée dans la production de reportages télévisuels. Son métier : Proposer des reportages aux télévisions du monde entier.

Voici une partie du diagramme de classes nécessaire à la gestion de l'agence :



Le dictionnaire (simplifié) des données du diagramme est le suivant :

Attribut	Désignation	Domaine	Commentaires
IDJ	Identifiant d'un journaliste	Chaîne de caractères	12 caractères
NOMJ	Nom d'un journaliste	Chaîne de caractères	
PRENOMJ	Prénom d'un journaliste	Chaîne de caractères	
IDR	Identifiant d'un reportage	Chaîne de caractères	13 caractères (code-barre)
TITRER	Titre d'un reportage	Chaîne de caractères	
DUREER	Durée d'un reportage	Réel	Exprimé en secondes
IDS	Identifiant d'un sujet	Chaîne de caractères	Code à 3 caractères
LIBELLES	Descriptif d'un sujet	Chaîne de caractères	
IDJT	Identifiant d'un journal télévisé	Entier naturel	Numéro automatique
DATEJT	Date d'un journal télévisé	Date	Ex : '10/09/2009'
IDTV	Identifiant d'une chaîne de télévision	Chaîne de caractères	Ex : CNN, RTBF, ARTE

Un reportage est vendu à une chaîne TV pour une diffusion lors d'un JT. L'acquéreur (une chaîne TV) a la possibilité de rediffuser le reportage autant de fois qu'il le souhaite dès lors que le reportage n'est pas coupé ou monté différemment.

4.3.2 Questions

1. Sur les seules informations fournies par le diagramme de classes ci-dessus, répondre aux questions suivantes (vos réponses doivent être justifiées) :

   (a) Un reportage peut-il être réalisé par plusieurs journalistes ?

   (b) Un reportage peut-il ne pas être réalisé par un journaliste ?

   (c) Un reportage peut-il être présenté dans des JTs de chaînes concurrentes ?

   (d) Un reportage peut-il ne jamais être présenté dans un JT ?

   (e) Peut-il y avoir plusieurs reportages sur le même sujet dans le même JT ?

   (f) Un sujet détermine-t-il un unique reportage ?

2. Pour chaque reportage, l'équipe de rédaction de l'*AFP* veut également pouvoir gérer les techniciens ayant collaboré au tournage ainsi que les personnalités interviewées. Les informations nécessaires à la gestion de ces personnes (techniciens et personnalités) sont identiques à celles gérées pour les journalistes. Par ailleurs, certains reportages doivent pouvoir faire l'objet d'une exclusivité avec certaines chaînes de télévision.

   (a) Compléter le diagramme de classes de telle sorte que ces nouvelles informations soient prises en compte.

   (b) Justifier brièvement les nouvelles cardinalités.

   (c) Donner le schéma relationnel **complet** du diagramme de classes.

4.4 Exercice 4

4.4.1 Problème

Un office HLM gère des cités composées de bâtiments, eux-mêmes composés d'appartements. Ces appartements sont libres ou loués à des locataires. Un locataire peut louer plusieurs appartements. Un appartement, s'il est loué, n'a qu'un seul locataire. Certaines informations sont nécessaires à la bonne gestion de chaque appartement comme, par exemple, l'étage où il se trouve, le montant de son loyer, si un garage y est affecté, la ville et l'adresse où il se trouve, le nom et le prénom de son locataire (s'il en a un)...

Dans chaque cité, les bâtiments sont numérotés séquentiellement à partir de 1. De même, dans un bâtiment, les appartements sont numérotés séquentiellement à partir de 1.

Chaque cité est sous la responsabilité d'un unique gardien, lui-même pouvant avoir plusieurs cités à sa charge. On souhaite stocker le nom et le prénom de chaque gardien. Un gardien est identifié par un numéro unique.

Chaque appartement appartient à type de logement (F1 à F6) correspondant à un certain nombre de caractéristiques précises regroupées sous la forme d'un libellé.

Pour chaque appartement, des opérations d'entretien peuvent être effectuées par des artisans. On souhaite stocker les noms et prénoms des artisans, ces derniers sont identifiés par un numéro. Pour chaque opérations d'entretiens, on souhaite stocker le prix et le détail de l'opération.

4.4.2 Questions

— Donner le diagramme de classes correspondant à l'énoncé.
— Traduire le diagramme en schéma relationnel.

4.5 Exercice 5

4.5.1 Problème

(Toute ressemblance avec des personnes, des organismes ou des faits existant ou ayant existé ne serait que pure coïncidence)

Vous avez été embauché le 1er mars 2020 par la société *THG-EPO*, société spécialisée dans l'approvisionnement de médicaments, remèdes et substances stimulantes pour sportifs. Les clients de *THG-EPO* sont des sportifs de haut niveau. Ses fournisseurs sont des sociétés pharmaceutiques pour la plupart implantées à l'étranger (Belgique, Etats-Unis, Pays-Bas, Uruguay, ...).

A sa création en 1998, *THG-EPO* ne comptait qu'une dizaine de fournisseurs belges pour une vingtaine de cyclistes réunis dans la seule équipe *Fastinos*. Depuis *THG-EPO* fournit plus de cinq mille sportifs dans des disciplines très diverses. Par, exemple, en athlétisme, football, ou même aux jeux d'échecs. *THG-EPO* emploie 530 personnes pour un chiffre d'affaire annuel d'environ 300 millions d'euros. Ses activités s'étendent sur l'Europe entière.

Malheureusement, cette croissance exponentielle a été entachée par quelques affaires, « surmédiatisées » selon Mr Ritchie Vorinquo, PDG de *THG-EPO*. Fort de cette expérience et afin de prévenir tout nouvel incident, Mr Vorinquo a décidé qu'il était désormais nécessaire de suivre précisément la pharmacopée prescrite par les médecins sportifs aux clients de *THG-EPO*. Cette décision devrait modifier en profondeur la stratégie marketing de la société, en quête d'image désormais irréprochable, basée sur le contrôle des substances absorbées par les sportifs.

THG-EPO vous a donc recruté au début du mois de mars pour une durée indéterminée en qualité de responsable du groupe de pilotage chargé de la mise en oeuvre du nouveau schéma directeur. Depuis le début du mois de mars, votre analyse de l'existant et des besoins vous a conduit à recenser les informations suivantes :

- La société dispose d'un fichier clients comportant pour chaque client (sportif), son identifiant, ses nom et prénom, son âge, son adresse, sa nationalité et un numéro de compte en Suisse.
- Les médecins sont identifiés par un numéro. Pour chaque docteur, on connaît ses nom et prénom, ses spécialités (toutes les spécialités étant elle-même répertoriées) et ses années d'expériences en tant que médecin sportif.
- Les fournisseurs (des sociétés pharmaceutiques) sont identifiés par leur numéro d'inscription IPF (International Pharmaceutical Federation). On a besoin de leur nom, leur adresse, leur numéro de téléphone et de fax.
- Chaque substance pharmaceutique (médicament) est identifiée par un numéro. Les informations nécessaires sont le nom de la molécule et sa formule chimique. Chaque médicament est fourni par une société pharmaceutique donnée. Lorsqu'une société pharmaceutique arrête de fournir *THG-EPO* en médicaments, les informations la concernant ne sont pas gardées (y compris les médicaments qu'elle fournissait).
- *THG-EPO* dispose de centres de distribution (succursales) implantés dans différentes villes européennes. Chaque centre est identifié par un nom d'inscription sur le registre du commerce du pays où il se trouve. Outre le nom, les informations concernant ces centres sont leur adresse, leur pays d'implantation, le nom de leur directeur, leur numéros de téléphone et de fax.
- Chaque centre est libre de pratiquer pour chaque médicament le prix de vente qu'il souhaite (le prix dépend souvent du pays, de la difficulté de se procurer le produit dans le pays). Les centres se fournissent directement auprès des sociétés pharmaceutiques. Les centres et les compagnies pharmaceutiques sont liés par contrats, chaque compagnie pouvant passer des contrats avec plusieurs centres, chaque centre pouvant passer des contrats avec plusieurs compagnies. Pour chaque contrat, on doit connaître la date d'entrée en vigueur du contrat, la date de fin du contrat et le texte du contrat.
- Les médecins prescrivent des médicaments aux sportifs. Un médecin peut prescrire un ou plusieurs médicaments à plusieurs sportifs. Un unique médecin est identifié pour chaque sportif comme son médecin traitant. Chaque sportif peut recevoir des prescriptions par un médecin autre que son médecin

traitant. Chaque prescription d'un médicament est datée et on a besoin de stocker la posologie prescrite (quantité, fréquence).

4.5.2 Questions

— Donner le diagramme de classes correspondant à l'énoncé.
— Traduire le diagramme en schéma relationnel.
— Pour chacune des hypothèses suivantes, le diagramme est-il modifié ? Si tel est le cas, redessinez la partie qui est modifiée. Sinon, expliquez en quelques lignes pourquoi le diagramme n'est pas modifié.
  — **Hypothèse A :** Au lieu d'être libre dans chaque centre de distribution, le prix de chaque médicament est fixé de façon unique pour tous les centres.
  — **Hypothèse B :** Chaque médicament peut être vendu par plusieurs sociétés pharmaceutiques.

Chapitre 5

Conception en SQL

5.1 La base de donnée *ZOO*

Le directeur d'un *zoo* a informatisé la gestion de son établissement. Dans ce zoo, on trouve des animaux répertoriés par *type* (lion, léopard, girafe, escargot, ...). Chaque animal possède un *nom* (Charly, Arthur, Enzo, ...) qui l'identifie de façon unique, une *date de naissance* et un *pays d'origine*. On retient également les *maladies* que chaque animal a contractées depuis son arrivée au zoo. Les animaux sont logés dans des *cages*. Chaque cage peut recevoir un ou plusieurs animaux. Certaines cages peuvent être inoccupées. Une cage correspond à une certaine *fonctionnalité*, qui n'est pas forcément liée à un type d'animal donné (par exemple une cage peut convenir à la fois aux girafes, aux éléphants et aux fauves, une autre aux grands oiseaux, ...). Une cage est identifiée par un *numéro*, elle est située dans une *allée*, identifiée aussi par un numéro. Des personnes sont employées par le zoo pour entretenir les cages et soigner les animaux. Chaque employé est identifiée par son *nom*, et on connaît la ville où elle réside. Il existe deux types de *postes* pour les employés : *gardien* ou *responsable*. Chaque employé est affecté à un unique poste : soit gardien, soit responsable. Un gardien s'occupe de une ou de plusieurs cages, et un responsable a la charge de toutes les cages de une ou de plusieurs allées. Une *allée* est sous la responsabilité d'un seul employé et toute cage occupée par au moins un animal est gardée par au moins un gardien ; les cages inoccupées ne sont pas gardées.

5.2 Schéma relationnel

Ci-dessous nous proposons un schéma relationnel traduisant l'énoncé.

5.2.1 Les relations

LesEmployes(<u>nomE</u>,adresse)
LesResponsables(<u>noAllee</u>,nomE)
LesCages(<u>noCage</u>,fonction,noAllee)
LesGardiens(<u>noCage,nomE</u>)
LesAnimaux(<u>nomA</u>,sexe,type,pays,anNais,noCage)
LesMaladies(<u>nomA,nomM</u>)

5.2.2 Domaines

Attribut	Domaine	Contraintes
nomE	chaîne de caractères	non nul
adresse	chaîne de caractères	non nulle
noAllee	entier	entre 1 et 999, non nulle
noCage	entier	entre 1 et 999, non nul
fonction	chaîne de caractères	non nulle
nomA	chaîne de caractères	
nomM	chaîne de caractères	
sexe	chaîne de caractères	femelle ou male ou hermaphrodite, valeur par défaut « male »
type	chaîne de caractères	non nul
pays	chaîne de caractères	valeur par défaut « France »
anNais	entier	≥ 1900

5.2.3 Contraintes d'intégrité

LesResponsables[nomE] $\subseteq$ LesEmployes[nomE]
LesCages[noAllee] $\subseteq$ LesResponsables[noAllee]
LesGardiens[nomE] $\subseteq$ LesEmployes[nomE]
LesGardiens[noCage] $\subseteq$ LesCages[noCage]
LesAnimaux[noCage] $\subseteq$ LesCages[noCage]
LesMaladies[nomA] $\subseteq$ LesAnimaux[nomA]

Lorsqu'une ligne est supprimée dans une table, toutes les lignes qui y font référence (dans les autres tables) doivent être supprimées.

5.3 Création

Implanter en SQL le schéma relationnel décrit dans la section précédente. Vous devrez respecter les domaines et contraintes données, notamment les clés primaires et étrangères.

5.4 Remplir la base

Une base *zoo* existe déjà sur le serveur. En créant votre base, vous n'écrasez pas l'existant. Lorsque vous faites une requête sur une de vos tables, par exemple *SELECT * FROM LesAnimaux*, La table *LesAnimaux* est en fait la table *foo.LesAnimaux* où *foo* est votre login. On peut alors accéder aux données de la base pré-existante en utilisant le login de son créateur, ici *zoo*. Par exemple : *SELECT * FROM zoo.LesAnimaux*. Ainsi, vous pouvez remplir vos tables à partir de la base de *zoo*. Par exemple :

```
INSERT INTO LesEmployes(nomE,adresse) SELECT nomE, adresse FROM zoo.LesEmployes ;
```

Répéter l'opération pour chacune de vos tables.

5.5 Déclencheurs (triggers)

Certaines contraintes de l'énoncé n'ont pas été traduites par le schéma relationnel. Nous proposons donc de les créer à partir de déclencheurs.

— *Un employé est soit un gardien soit un responsable mais jamais les deux.* Créer deux déclencheurs qui empêchent les insertions dans les tables LesResponsables et LesGardiens qui violent la contrainte d'exclusion.
— *Une cage vide n'est pas gardée.* Créer un déclencheur qui lorsqu'une cage devient vide (DELETE dans LesAnimaux), enlève les gardiens affectés à cette cage.
— *Une cage non vide doit être gardée par au moins un gardien.* Créer un déclencheur qui lorsqu'un animal est ajouté, vérifie si la cage occupée par l'animal est gardée et le cas échéant, affecte *Scholl* comme gardien.

Chapitre 6

JDBC/Servlet

L'objectif de ce TD est de réaliser une application internet faisant des accès à la base de données *Zoo*. Cette application sera basée sur les *servlets* et *JDBC* (Java DataBase Connectivity). On utilisera de plus pour la compilation et la gestion des répertoires l'outil `ant` du projet `Apache`.

6.1 Mise en place

Tout d'abord, vous devez mettre en place la base de donnée. Pour cela connectez-vous à la machine *im2ag-oracle* :

```
ssh im2ag-oracle.e.ujf-grenoble.fr
```

Ensuite téléchargez le script de création de la base *Zoo* :

```
wget www-verimag.imag.fr/~devismes/BD/create_zoo_eleve.sql
```

Connectez-vous à `sqlplus`. Puis exécutez le script :

```
start create_zoo_eleve;
```

Puis, quittez `sqlplus` (`exit`) et quittez `im2ag-oracle` (`exit`).

Vous allez maintenant installer et configurer un squelette d'application utilisant l'outil `ant`. Connectez-vous en mode graphique à la machine `im2ag-tomcat` :

```
ssh -X im2ag-tomcat.univ-grenoble-alpes.fr
```

Ensuite téléchargez le squelette *Zoo* :

```
wget www-verimag.imag.fr/~devismes/BD/appliEleve2.tgz
```

Décompressez le fichier :

```
tar -xvzf appliEleve2.tgz
```

Modifiez les droits d'accès à votre *home* et au répertoire `appliEleve2` pour autoriser l'accès en lecture et exécution à tous les utilisateurs :

```
cd
chmod ugo+rx .
chmod ugo+rx appliEleve2
```

Positionnez-vous dans le répertoire `appliEleve2`. Editez le fichier de configuration `build.properties`. Ce fichier contient un certain nombre d'informations permettant à `ant` de déployer l'application : location du server `Apache`, le login et le mot de passe permettant d'y accéder, l'url du manager qui va déployer votre site et enfin le point d'entrée du site (`app.path`). Ce point d'entrée doit être différent pour chaque étudiant, aussi nommez ce point d'entrée avec votre **propre** login :

```
app.path=/login
```

Puis mettez à jour la configuration de `ant` en tapant :

```
ant clean
```

`ant` est maintenant prêt à être utiliser.

6.2 Utilisation de `ant`

Les sources des applications sont stockées dans le répertoire `src`. Le squelette de l'application *Zoo* est donc dans `src/zoo`.

Noter qu'à chaque fichier source correspond deux entrées `servlet` et `servlet-mapping` dans le fichier de configuration `appliEleve2/web/WEB-INF/web.xml`. Ainsi, si vous souhaitez ajouter des fichiers sources (ce qui n'est pas nécessaire lors de ce TD), il est obligatoire de modifier ce fichier de configuration en ajoutant les entrées appropriées. Noter enfin, que `appliEleve2/web/WEB-INF/lib` contient les bibliothèques supplémentaires nécessaires, ici `ojdbc14.jar` (le pilote permettant de communiquer avec Oracle).

Pour déployer le squelette d'application, placez-vous dans le répertoire `appliEleve2` et tapez :

```
ant install
```

Par la suite, pour redéployer l'application après des modifications tapez :

```
ant remove install
```

En cas d'échec, par exemple suite à une erreur de compilation, votre site a été supprimé du serveur, donc corrigez votre code puis tapez :

```
ant install
```

Si vous avez lancé `firefox` depuis `im2ag-tomcat`, votre application est accessible depuis `firefox` par l'adresse :

```
localhost:8080/login
```

Sinon, votre application est accessible depuis `http://im2ag-tomcat.univ-grenoble-alpes.fr:8080/login/`, mais à condition que l'adresse `im2ag-tomcat.univ-grenoble-alpes.fr` soit listée parmi les exceptions proxy de votre browser internet (Preferences/Advanced/Network/Settings).

Toutefois, **pour éviter de surcharger le serveur** `im2ag-tomcat`, nous vous conseillons de créer un **tunnel ssh** afin d'accéder aux pages web via le `localhost` de votre machine de TP. Pour cela :

1. Lancez un terminal depuis votre machine de TP.
2. Dans ce terminal, tapez `ssh -L 28080:localhost:8080 im2ag-tomcat.univ-grenoble-alpes.fr`
3. Laissez ce terminal ouvert jusqu'à la fin du TP.
4. Lancez `firefox` (en local, depuis un autre terminal, par exemple), puis tapez `http://localhost:28080/login` dans la barre d'adresse.

6.3 Questions

Le schéma relationnel de la base *Zoo* est rappelé dans la section 6.4. Vous pouvez regarder la solution quasi-complète à l'adresse :

`localhost:8080/solution/`

ou, si vous avez mis en place le tunnel ssh, tapez

`http://localhost:28080/solution/`

1. Considérez les fichiers `AnimalDetails.java` et `AnimalDetailsAction.java`. Modifiez-les pour :
   (a) Afficher le sexe et l'année de naissance d'un animal sélectionné.
   (b) Afficher tous les détails des animaux venant d'un pays choisi dans une liste.
2. Complétez l'application en ajoutant les fonctions suivantes :
   (a) Afficher les cages inoccupées.
   (b) Afficher le nom des gardiens et du responsable d'un animal donné (donné par son nom).
   (c) Enregistrer une maladie donnée pour un animal donné.
   (d) Modifier l'affectation d'un gardien ou d'un responsable.
   (e) Enregistrer un animal et le cas échéant, créer une affectation d'un employé à une cage (rôle de gardien).

6.4 La base de donnée *ZOO*

6.4.1 Les relations

LesEmployes(nomE,adresse)
LesResponsables(noAllee,nomE)
LesCages(noCage,fonction,noAllee)
LesGardiens(noCage,nomE)
LesAnimaux(nomA,sexe,type,pays,anNais,noCage)
LesMaladies(nomA,nomM)

6.4.2 Domaines

Attribut	Domaine	Contraintes
nomE	chaîne de caractères	non nul
adresse	chaîne de caractères	non nulle
noAllee	entier	entre 1 et 999, non nulle
noCage	entier	entre 1 et 999, non nul
fonction	chaîne de caractères	non nulle
nomA	chaîne de caractères	
nomM	chaîne de caractères	
sexe	chaîne de caractères	femelle ou male ou hermaphrodite, valeur par défaut « male »
type	chaîne de caractères	non nul
pays	chaîne de caractères	valeur par défaut « France »
anNais	entier	≥ 1900

6.4.3 Contraintes d'intégrité

LesResponsables(nomE) $\subseteq$ LesEmployes(nomE)
LesCages(noAllee) $\subseteq$ LesResponsables(noAllee)
LesGardiens(nomE) $\subseteq$ LesEmployes(nomE)
LesGardiens(noCage) $\subseteq$ LesCages(noCage)
LesAnimaux(noCage) $\subseteq$ LesCages(noCage)
LesMaladies(nomA) $\subseteq$ LesAnimaux(nomA)

Chapitre 7

Normalisation

7.1 Exercice 1

Le programme d'un conférence scientifique est représenté par la relation suivante :

Programme(Session,Date_Session,Titre_Exposé,Id_Intervenant,Nom_Intervenant)

La conférence est divisée en sessions. Chaque session consiste en une série d'exposés.
Dans la suite, nous utiliserons les abréviations suivantes :
— S : Session,
— D : Date_Session,
— T : Titre_Exposé,
— I : Id_Intervenant et
— N : Nom_Intervenant

Questions :

1. Traduisez chacune des hypothèses suivantes en dépendance fonctionnelle :
   (a) On identifie chaque intervenant par un identifiant unique, en particulier chaque identifiant détermine un nom d'intervenant.
   (b) Chaque session se déroule à une date précise.
   (c) Un exposé ne peut pas être présenté par deux intervenants différents lors de la même session.
   (d) Un intervenant ne présente pas plus d'une fois le même exposé lors de la conférence.
2. Justifiez que IT et ST sont des clés de la relation.

7.2 Exercice 2

Soit la relation $R(A, B, C, D, E, G)$. Soit $F = \{AB \rightarrow C, B \rightarrow D, CD \rightarrow E, CE \rightarrow G, G \rightarrow A\}$ un ensemble de dépendances fonctionnelles de R.

Question :

1. Calculer les *clés candidates* en utilisant l'algorithme et les astuces vues en cours.

7.3 Exercice 3

Soit la relation $R(A, B, C, D, E)$. Soit $F = \{A \rightarrow B, BC \rightarrow E, ED \rightarrow A\}$ un ensemble de dépendances fonctionnelles de R.

Questions :

1. Calculer les *clés candidates*.
2. Justifier si R est une $3NF$ ou non. Si R n'est pas une $3NF$, appliquer l'algorithme de synthèse pour transformer R en $3NF$.
3. Justifier pourquoi R n'est pas en $BCNF$.
4. Mettre R en $BCNF$ en utilisant l'algorithme récursif.

7.4 Exercice 4

Soit la relation $R(A, B, C, D, E)$. Soit $F = \{A \rightarrow C, C \rightarrow B, C \rightarrow D, CE \rightarrow A, CE \rightarrow B\}$ un ensemble de dépendances fonctionnelles de R.

Questions :

1. Calculer les *clés candidates*.
2. Justifier si R est une $3NF$ ou non. Si R n'est pas une $3NF$, appliquer l'algorithme de synthèse pour transformer R en $3NF$.
3. Justifier pourquoi R n'est pas en $BCNF$.
4. Mettre R en $BCNF$ en utilisant l'algorithme récursif.

7.5 Exercice 5

Soit la relation $R(G, R, A, S, P)$. Soit $F = \{G \rightarrow AS, R \rightarrow S, AP \rightarrow R, R \rightarrow P, RSP \rightarrow A\}$ un ensemble de dépendances fonctionnelles de R.

Questions :

1. Calculer les *clés candidates*.
2. Justifier si R est une $3NF$ ou non. Si R n'est pas une $3NF$, appliquer l'algorithme de synthèse pour transformer R en $3NF$.
3. Justifier pourquoi R n'est pas en $BCNF$.
4. Mettre R en $BCNF$ en utilisant l'algorithme récursif.

Annexe A

Découvrir Oracle

L'accès au Système de Gestion de Bases de Données (*SGBD*) **Oracle** n'est possible que sur le serveur `im2ag-oracle`.

A.1 Connexion à Oracle

Se connecter à la machine `im2ag-oracle`. Pour cela, par exemple, dans un terminal (lorsque vous êtes sur une machine de la salle de TP) taper :

```
ssh -X <login>@im2ag-oracle.e.ujf-grenoble.fr
```

`<login>` correspond à votre *nom d'utilisateur AGALAN*. Vous devez ensuite fournir votre *votre mot de passe AGALAN*. Ce sont les login et mot de passe que vous utilisez habituellement sur le réseau de l'UFR IMAG. Vous êtes maintenant sous Unix, il s'affiche :

```
>
```

Ensuite, taper la commande suivante :

```
> source /oracle/TPTOMCAT/.cshrc
```

Pour éviter de retaper cette commande au début de chaque TP, vous pouvez copier ce fichier à la racine de votre compte :

```
> cp /oracle/TPTOMCAT/.cshrc ~
```

Ensuite, appeler le logiciel **Oracle** en utilisant :

```
> sqlplus
```

Vous devez fournir votre *nom d'utilisateur AGALAN* et *votre mot de passe AGALAN* (ceux que vous utilisez habituellement sur le réseau de l'UFR IMAG). Vous êtes sous **Oracle**, il s'affiche :

```
SQL>
```

Pour sortir d'**Oracle**, taper `exit`, qui ramène sous `Unix`. Ensuite, pour sortir d'`Unix`, utiliser à nouveau `exit`.

Votre compte est monté à la fois sur les machines de la salle de TP et sur le `im2ag-oracle`. Donc, lorsque vous êtes sur une machine de la salle de TP, vous pouvez directment taper vos requêtes dans un éditeurs (*cf.*, Exécution d'une requête à partir d'un fichier) et les sauvegarder sur votre compte. Vous pourrez ainsi les exécuter (à partir de votre compte) sur le serveur `im2ag-oracle`.

A.2 Travailler depuis chez soi

Vous pouvez vous connecter à `Oracle` toujours en tapant :

```
ssh <login>@im2ag-oracle.e.ujf-grenoble.fr
```

Vous pouvez aussi monter votre `home` en local. Pour cela, créer un répertoire vide `sql`. Puis, taper :

```
sshfs <login>@im2ag-mandelbrot.e.ujf-grenoble.fr: sql/
```

Le contenu de votre `home` apparaît alors dans le répertoire `sql`. Votre pouvez alors lire et copier dans votre `home` en passant par le répertoire local `sql`.

Pour démonter le `home`, taper :

```
umount sql
```

Pour plus d'informations, consultez :

`https://im2ag-wiki.ujf-grenoble.fr/doku.php?id=environnements:oracle`

A.3 Exécution d'une requête en ligne

Attention, contrairement aux TDs, le nom des tables dans les requêtes doit **toujours** être préfixé par le nom de la base à laquelle la table appartient. Par exemple, dans l'environnement **Oracle**, taper la ligne suivante et analyser le résultat :

```
SQL> select nomA from zoo.LesAnimaux;
```

Ensuite, taper la ligne suivante et analyser le résultat :

```
SQL> select nomA from zoo.LesAnimaux
```

Pour la présentation des jeux d'essai, les formats d'affichage peuvent être re-définis en utilisant les instructions column, set linesize, set pagesize, etc. Taper help nom_commande pour plus de détails.

Pour connaître les noms des bases correspondant aux tables, taper (utiliser la clause WHERE pour chercher le nom de la base d'une table particulière) :

```
SQL> select owner, table_name from all_tables;
```

Noter que nous utilisons principalements les bases : `zoo`, `repas`, `agence`, et `batiments`.

Pour avoir une description des colonnes de la table `zoo.LesAnimaux`, taper :

```
SQL> desc zoo.LesAnimaux;
```

A.4 Exécution d'une requête à partir d'un fichier

La plupart du temps, on prépare les requêtes dans un fichier qu'on demande ensuite à **Oracle** d'exécuter, en utilisant la commande `start` (ou le raccourci `@`). Dans un fichier de suffixe `.sql`, par exemple `Req1.sql`, taper le texte suivant :

```
-- le nom des animaux
select nomA from zoo.LesAnimaux;
```

La ligne commençant par le double tiret est une ligne de commentaires, les autres lignes forment une requête SQL. Il ne faut pas placer de commentaires à l'intérieur d'une requête. Pour exécuter cette requête, taper sous **Oracle** :

```
SQL> start Req1
```

Le système cherche alors le fichier `Req1.sql` et l'exécute. Le résultat doit être le même que lorsqu'on tape la requête en ligne. Essayer différentes variantes dans le fichier `Req1.sql`, les faire exécuter et analyser le résultat. Si **Oracle** indique des erreurs, les corriger. Par exemple (à vous de compléter avec des commentaires) :

```
select nomA, noCage from zoo.LesAnimaux where pays='Kenya';
```

Sur l'exemple ci-dessus, essayer diverses combinaisons de majuscules et minuscules. Ensuite, essayer en remplaçant la ligne :

```
where pays='Kenya';
```

par une des lignes suivantes :

```
where anNais<1992 and sexe='male';
```

Essayer aussi d'écrire plusieurs requêtes l'une après l'autre dans le même fichier.

A.5 Stockage des résultats des requêtes dans un fichier

Pour stocker automatiquement les résultats des requêtes dans un fichier, utiliser `spool`, en insérant dans le fichier `Req1.sql` une ligne en haut et une ligne en bas :

```
spool ResReq1
... (ici vos requêtes) ...
spool off;
```

Comme précédemment, taper sous **Oracle** :

```
SQL> start Req1
```

Un fichier est créé automatiquement, de nom `ResReq1.lst` (le nom que vous avez donné, suivi du suffixe `.lst`). Ouvrir le fichier `ResReq1.lst` pour voir son contenu.

Remarque 3 *On peut remplacer* `spool ResReq1` *par* `spool Req1`*, le système ne confond pas les fichiers* `Req1.lst` *et* `Req1.sql` *puisqu'ils n'ont pas le même suffixe.*

A.6 Requêtes paramétrées

Dans un fichier de nom, par exemple, `Req2.sql`, taper le texte suivant (où `accept` et `prompt` sont des mots-clés d'**Oracle**) :

```
-- le nom et le numéro de cage des animaux mâles
-- né avec une date donnée et
-- provenant d'un pays donné
accept lepays prompt 'Donner le pays : '
accept lannee prompt 'Donner l''année : '
select noma, nocage from zoo.lesanimaux where pays='&lepays' and anNais<&lannee
and sexe='male';
```

Remarque 4 *Attention à l'utilisation des quotes ! Il faut les doubler pour* `l''année` *(dans un* `prompt`*), il faut en mettre pour* `'&lepays'` *(chaîne de caractères) mais pas pour* `&lannee` *(nombre).*

Taper `start Req2`. Le système imprime le texte `Donner le pays :` (sans quotes). Répondre en donnant un pays, par exemple *Kenya* (sans quotes). Puis le système imprime le texte `Donner l'année :` (sans quotes). Répondre en donnant un entier, par exemple 1992. Alors le système exécute la requête.