PROJET DSB 2021-2022 PARTIE 1

Données structurées et bases de données
Rapport de projet

Groupe 1
MOULHERAT Hadrien – hadrien.moulherat@etudiant.univ-rennes1.fr
ZHILE Zhang – zhile.zhang@etudiant.univ-rennes1.fr

Rapport de projet DSB
Partie 1

1 Introduction

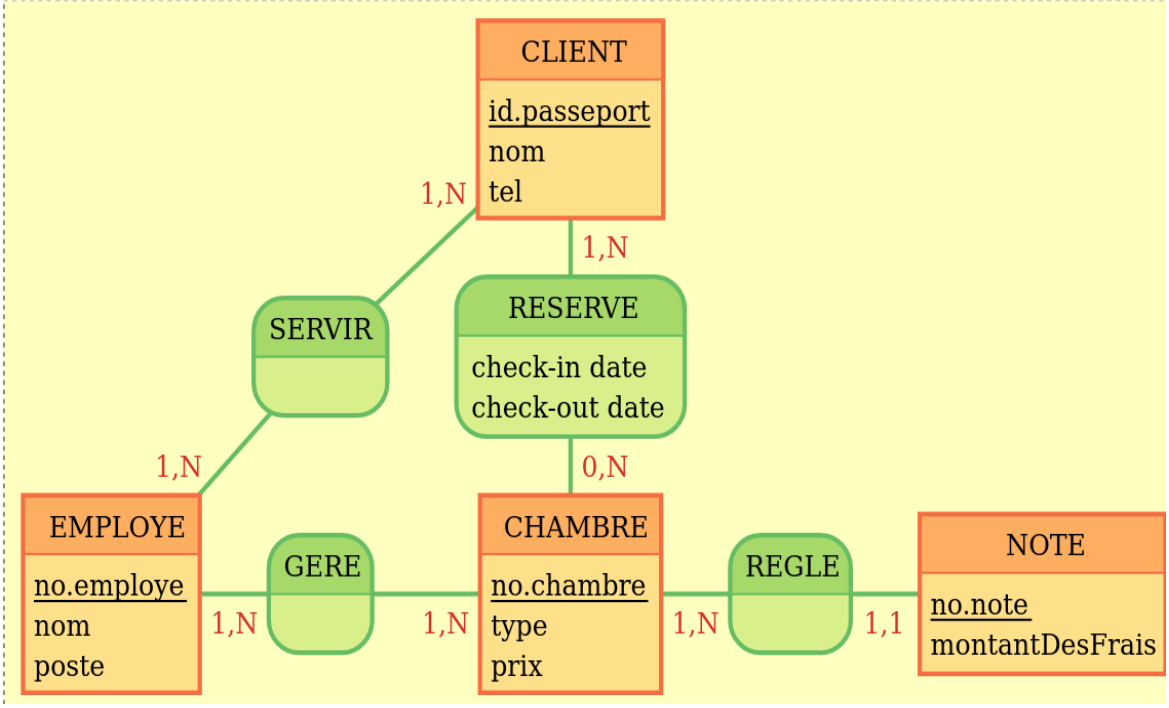
Nous avons choisi l'hôtellerie comme domaine d'application. Plus précisément l'organisation d'un l'hôtel.

La base de données créée stocke les durées de réservation passée ou actuelle d'une chambre. Ainsi on crée un registre des réservations de l'hôtel. On peut alors interroger la base de données pour connaître la chambre ou se trouve un client. (information très utile surtout quand les grands hôtel font des cadeaux aux clients les plus fidèles, ainsi les employés savent dans quelle chambre poser les cadeaux). Elle nous permet aussi d'avoir l'identité des clients ayant passé une nuit ou plus dans une chambre en particulier (l'historique des clients de la chambre).

La base de données stocke également le montant du séjour d'un client de l'hôtel. Lors du check-out c'est ainsi que le concierge connaît le montant à régler et peut informer le client afin qu'il règle sa note.

Enfin la gestion des employés est essentielle pour géré efficacement un hôtel. Étant donné que les bases de données sont essentiellement utilisé dans la gestion, il nous a paru évidant que notre base de données puisse gérer l'organisation des employés. De ce fait la base de données stocke l'identité de chaque employés et les associes à une/des chambre(s) ou un/des client(s) en fonction du métier. Par exemple une gouvernante d'hôtel gérera plusieurs chambre, le room service sera aussi affecté à une chambre ou à un client, mais on pourra connaître sa chambre grâce à la base de données et enfin un valet sera associer a une/des chambre(s) et un/des client(s).

2 Modèle conceptuel (MCD)



ENTITÉ

- **CLIENT** : L'entité CLIENT stocke identité du client a travers l'attribut id.passeport. Les attributs nom et tel stockent respectivement le nom et le numéro de téléphone du client, le numéro de téléphone n'est pas un attribut obligatoire. L'attribut id.passeport est la clé primaire de l'entité CLIENT.

- **EMPLOYE** : L'entité EMPLOYE stocke les informations des employés de l'hôtel. L'attribut no.employe enregistre le numéro de l'employé, enfin les attributs nom et poste enregistrent respectivement le nom et le poste de l'employé. La clé primaire de l'entité EMPLOYE est no.employe.

- **CHAMBRE** : L’entité CHAMBRE stocke les informations lié a une chambre de l’hôtel. L’attribut no.chambre enregistre le numéro de la chambre, l’attribut type stocke le type de chambre (suite, chambre double, etc), enfin l’attribut prix enregistre le prix de la chambre. L’entité CHAMBRE a pour clé primaire no.chambre.

- **NOTE**: L’entité NOTE stocke les informations des frais lié a une chambre. L’attribut no.note stocke le numéro de la note, et l’attribut montantDesFrais enregistre le montant final de la note. La clé primaire de l’attribut NOTE est no.note.

ASSOCIATION

- **SERVIR** :Contraintes pour l'association entre CLIENT et EMPLOYE
  Un CLIENT est servi par plusieurs EMPLOYE
  Un EMPLOYE peut servir plusieurs CLIENT

- **RESERVE** :Contraintes pour l'association entre CLIENT et CHAMBRE
  Un CLIENT peut réserver une ou plusieurs CHAMBRE
  Une CHAMBRE est réservé par 0 ou plusieurs CLIENT
  Les deux attributs check-in date et check-out date enregistrent la date d’arrivé et la date de départ du CLIENT

- **GERE** :Contraintes pour l'association entre EMPLOYE et CHAMBRE
  Un EMPLOYE peut gérer une ou plusieurs CHAMBRE
  Une CHAMBRE est géré par une ou plusieurs EMPLOYE

- **REGLE** :Contraintes pour l'association entre NOTE et CHAMBRE
  Le client d’une CHAMBRE régle une ou plusieurs NOTE
  Une NOTE est réglé par un client d’une CHAMBRE

3 Schéma relationnel (MLD)

Application de la règle 1

- CLIENT (id.passeport, nom*, tel)
- EMPLOYE (no.employe, nom*, poste*)
- CHAMBRE (no.chambre, type*, prix*)
- NOTE (no.note, montantDesFrais*)

Application de la règle 2 ou 2 bis

- NOTE (no.note, montantDesFrais*, no.chambre*)

  On ajoute no.chambre qui fait référence a la relation CHAMBRE

Application de la règle 3

- SERVIR (id.passeport*, no.employe*)

  On crée une nouvelle relation SERVIR avec no.employe qui fait référence a la relation EMPLOYE, et id.passeport qui fait référence a la relation CLIENT, la paire no.employe et id.passeport est la clé primaire.

- RESERVE (no.chambre*, id.passeport*, check-in date*, check-out date*)

  On crée une nouvelle relation RESERVE avec no.chambre qui fait référence a la relation CHAMBRE, et id.passeport qui fait référence a CLIENT, la paire no.chambre et id.passeport est la clé primaire.

- GERE (no.chambre*, no.employe*)

  On crée une nouvelle relation GERE avec no.chambre qui fait référence a la relation CHAMBRE et no.employe qui fait référence a la relation EMPLOYE, la paire no.chambre et no.employe est la clé primaire.

4 Schéma physique (MPD)

- create table CLIENT (IDpasseport varchar(42) not null, Cnom varchar(42) not null, tel varchar(42), primary key (`IDpasseport`))

- create table EMPLOYE (NOemploye varchar(42) not null, Enom varchar(42) not null, poste varchar(42) not null, primary key (`NOemploye`))

- create table CHAMBRE (NOchambre varchar(42) not null, type varchar(42) not null, prix varchar(42) not null, primary key (NOchambre))

- create table NOTE (NOnote varchar(42) not null, montantDesFrais varchar(42) not null, NOchambre varchar(42) not null, foreign key(`NOchambre`) references CHAMBRE(`NOchambre`), primary key (NOnote))

- create table SERVIR (IDpasseport varchar(42) not null, NOemploye varchar(42) not null, foreign key(`IDpasseport`) references CLIENT(`IDpasseport`), foreign key(`NOemploye`) references EMPLOYE(`NOemploye`), primary key (`IDpasseport`))

- create table RESERVE (NOchambre varchar(42) not null, IDpasseport varchar(42) not null, check_in_date date not null, check_out_date date not null, foreign key(`NOchambre`) references CHAMBRE(`NOchambre`), foreign key(`IDpasseport`) references CLIENT(`IDpasseport`), primary key (`NOchambre`, `IDpasseport`))

- create table GERE (NOchambre varchar(42) not null, NOemploye varchar(42) not null, foreign key(`NOchambre`) references CHAMBRE(`NOchambre`), foreign key(`NOemploye`) references EMPLOYE(`NOemploye`), primary key (`NOchambre`, `NOemploye`))

5 Peuplement des tables

Client : Brad Dourif dont le l'identifiant de passeport est 12546A54X dont le numéro de téléphone est 02 99 59 37 74.

CLIENT('12546A54X', 'Brad Dourif', '02 99 59 37 74')

Employe : Hadrien Lopes dont l'identifiant est 27S dont le poste est Directeur générale

EMPLOYE('27S', 'Hadrien Lopes', 'Directeur générale')

Chambre : La chambre 8020 qui est une suite dont le prix a la nuit est de 200 euros.

CHAMBRE('8020', 'Suite', '200')

Note : La note numéro 1337 dont le montant est de 600 euros règle la note de la chambre 8020.

NOTE('1337', '600', '8020')

Servir : Le client dont l'identifiant de passeport est 12546A54X se fait servir par l'employer dont l'identifiant est 27S.

SERVIR('12546A54X', '27S')

Reserve : Le client dont l'identifiant de passeport est 12546A54X réserve du 23/02/2022 jusqu'au 26/02/2022 la chambre 8020

RESERVE('12546A54X', '8020', 2022-02-23, 2022-02-26)

Gere : L'employer dont l'identifiant est 27S gère la chambre 8020.

GERE('27S', '8020')

6 Requêtes

a une sélection avec projection

- Français : l'IDpasseport des clients dont le nom est Brad
- Algèbre : $\pi_{IDpasseport}(\sigma_{Cnom='Brad'}(CLIENT))$
- SQL : select IDpasseport
  from CLIENT
  where Cnom = 'Brad'

b une jointure

- Français : le montant des frais et le type de la chambre 8020 (NOchambre ='8020')
- Algèbre : $\pi_{montantDesFrais,type}(\sigma_{NOchambre='8020'}(CHAMBRE \bowtie NOTE))$
- SQL : select montantDesFrais, type
  from CHAMBRE natural join NOTE
  where NOchambre = '8020'

c une moyenne sur l'intégralité d'un attribut

- Français : la moyenne prix des chambres
- Algèbre :
- SQL : select avg(prix) as moyenPrix
  from CHAMBRE

d un regroupement avec calcul

- Français : Nombre de chambres gérés par chaque employé
- Algèbre :
- SQL : select NOemploye, count(NOchambre)
  from GERE
- group by NOemploye

e une différence

- Français : les chambres libres
- Algèbre : $\pi_{NOchambre}(CHAMBRE) - \pi_{NOchambre}(RESERVE)$
- SQL : select NOchambre
  from CHAMBRE
  where NOchambre not in (select NOchambre form RESERVE)

f une division

- Français : le(s) employé(s) qui gère(nt) toutes les chambres
- Algèbre : $\pi_{NOemploye,NOchambre}(GERE) \div \pi_{NOchambre}(CHAMBRE)$
- SQL : create view RF as(
  select NOemploye, count(NOchambre) as NBchambre
  from GERE
  group by NOemploye
  having NBchambre = (select count(*) form CHAMBRE)
)
  select NOemploye from R1