Mini-Projet : Entrepôts de Données

AirBNB

Groupe: Leonie Berwanger, Tahani Qattan, Leila Rekkab, Khady Ajna Thiam

Rendu le: 23 octobre 2018

I Considérations préliminaires

Airbnb est une plateforme communautaire qui permet de réserver , en ligne des logements pour des particuliers dans le but d'un séjour pour une période bien déterminée, et ce, à travers le monde entier. La plateforme comporte deux types d'utilisateurs :

- Hôte : c'est un utilisateur qui offre son logement sur plateforme et sera rémunéré lorsque qu'il reçoit une demande de réservation.
- Client : utilisateur qui effectue une réservation d'un logement pour une periode donnée.

Notre interêt repose sur l'analyse du fonctionnement de plateforme, et ainsi parvenir à l'augmentation du chiffre d'affaire de la plateforme. Pour ce faire, nous nous sommes intéréssées à l'analyse des différentes opérations qui peuvent être réaliser par les differents utilisateurs d'Airbnb (offre d'un logementpar un hôte ou réservaation d'un logement par un Client).
Une fois l'analyse effectuée, les anomalies sont detectées, et ainsi des améliorations peuvent être effectuées pour atteindre notre principal objectif qu'est "l'Augmentation du Chiffre d'affaires".

I.1 Quelles sont les actions/opérations importantes pour la plateforme de locations de vacances AirBNB ?

- Réservation d'un logement
- Offre d'un logement par un hôte
- Annulation d'une réservation

I.2 Pour chaque opération : trois traitements possibles permettant d'aider à la prise de décision sur le sujet.

Réservation d'un logement

- Le prix moyen (/nuit) des differents logements réservés par type
- Le prix moyen (/nuit) des differents logements réservés par ville
- Nombre de reservations par ville et par type de logement

Offre d'un logement par un hôte

- Le nombre de logements offerts par des « Superhosts » par ville le 2 juillet 2018
- Le type de logement le plus offert le 2 juillet 2018
- Le nombre de logements disponibles le weekend par ville

Annulation d'une réservation

- Les raisons d'annulations les plus fréquentes
- Les hôtes les plus exposés aux annulations

I.3 L'ordre d'importance

1. Réservation d'un logement par un client : Permet d'augmenter la rentabilité de la plateforme
2. Offre d'un logement par un Hôte : Permet d'augementer le nombre de logememnt offerts et par conséquent l'augmentation du nombre de réservations
3. Annulation d'une réservation par un client : Permet de retirer de l'espace de stockage de la plateform les logements qui ne sont pas fiables

II Conception de l'entrepôt

II.1 Les deux actions les plus importantes à analyser

1. Réservation d'un logement par un client
2. Offre d'un logement par un hôte

II.2 Réservation d'un logement

La table des faits

Pour enregistrer les réservations, on choisit des faits transactionnels, parce qu'on s'intéresse aux caractéristiques des réservations spécifiques et il ne sert à rien de les regrouper à l'avance. La table des faits suit le modèle relationnel suivant :
Reservations(
idLogement ChaineCar,
idLocalisation ChaineCar,
idHote ChaineCar,
idClient ChaineCar,
idPeriode ChaineCar,
nbVoyageurs Integer,

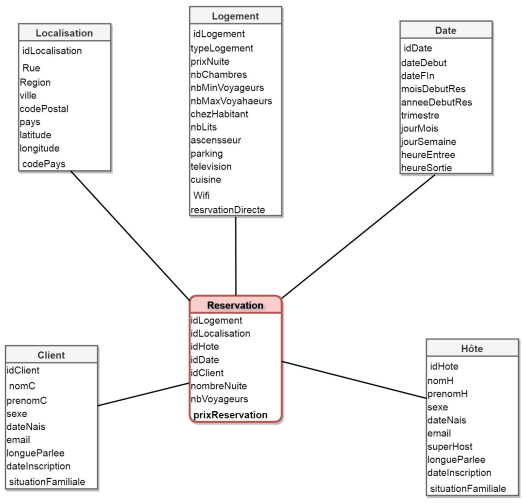

FIGURE 1 – Modèle en étoile des réservations

duree Nombre(10),
prixTotal Nombre(10,2)
fraisDuTotal Nombre(3,2))

Les mesures

La table des faits contient les mesures *nbVoyageurs*, *duree* et *prixTotal*. Ces mesures sont additives, parce qu'elles sont exprimées en nombres et liées à une certaine transaction. La mesure fraisDuTotal n'est pas additive parce qu'elle représente un pourcentage.

II.3 Offre d'un logement par un hôte

Pour les logements offerts on crée une table de snapshots (une période raisonnable serait 1 jour). À la fin d'une période fixe on enregistre le nombre de logements par localisation,type de logement et disponibilité, ça veut dire les créneaux de temps pour lequel l'hôte les ouvre aux clients.

La table des faits

Un fait **(idType, idLocalisation, idDisponibiltite, idHote, j, nombreOffres,pourcentageLoue)** existe pour chaque combinaison de localisation idLocalisation, type de logement idType, disponibilité idDisponibilite, hôte idHote et jour j. La mesure nombreOffres représente le nombre de logements avec ces paramètres offerts et le pourcentage loué mesure l'occupation.

Les mesures

La mesure nombreOffres est semi-additive, parce qu'on peut p.e. prendre le somme de logements dans toutes les localisations et de tous les types, mais on ne peut pas additionner à travers de la dimension du temps. Le pourcentage n'est bien sûr pas additif.

II.4 Les dimensions

Les tables des dimensions suivent le modèle relationnel suivant :
typesLogement(idType,taille,nbChambres,chezHabitant,nbLits)
Localisation(rf.II.2, 9 attributs)
Disponibilite(rf. Implementation) Hôte(9 attributs)
Jour(rf.Dates)

III Implémentation

Dans le fichier *tableCreation.sql* est décrite la création des tables de l'entrepöt. On initialise d'abord les dimensions et après les tables des faits qui en prennent leurs clés primaires.

```
CREATE TABLE Localisation (
        idLocalisation NUMBER(10) PRIMARY KEY,
        Rue VARCHAR2(100),
        Region VARCHAR2(100),
        Ville VARCHAR2(100),
        codePostal VARCHAR2(10),
        pays VARCHAR2(100),
        latitude NUMBER(10,7),
        CONSTRAINT lat CHECK(latitude BETWEEN 0.0 AND 90.0),
        longitude NUMBER(10,7),
        CONSTRAINT lon CHECK(longitude BETWEEN -280.0 AND 280.0),
        codePays VARCHAR2(100)
);

-- create shared dimension table Dates as specified
CREATE TABLE Dates (
        idDate NUMBER(10) PRIMARY KEY,
        dateEntier DATE,
        description VARCHAR2(20),
        semaine NUMBER(2),
        mois VARCHAR2(10),
        trimestre NUMBER(1),
        an NUMBER(4),
        jourSemaine VARCHAR2(20),
        jourMois NUMBER(2),
```

```
        weekend VARCHAR2(20),
        feries VARCHAR2(22),
        saison VARCHAR2(20)
);
-- add constraints on values
ALTER TABLE Dates ADD (
        CONSTRAINT months CHECK(mois in
        ('January','February','March','April','May','June',
        'July','August','September','October','November','December')),
        CONSTRAINT trimester CHECK(trimestre BETWEEN 1 AND 4),
        CONSTRAINT weekDay CHECK(jourSemaine in
        ('Monday','Tuesday','Wednesday','Thursday','Friday',
        'Saturday','Sunday')),
        CONSTRAINT monthDay CHECK(jourMois BETWEEN 1 AND 31),
        CONSTRAINT weekend CHECK(weekend in ('Weekend','Weekday')),
        CONSTRAINT holiday CHECK(feries in ('Holiday','Non-Holiday')),
        CONSTRAINT season CHECK(saison in
        ('Spring','Summer','Fall','Winter'))
);


CREATE TABLE typesLogement (
        idType NUMBER(10) PRIMARY KEY,
        taille VARCHAR2(10),
        CONSTRAINT sizing CHECK(taille IN ('house','appartment','room')),
        nbChambres NUMBER(3),
        CONSTRAINT numberRooms CHECK (nbChambres > 0),
        chezHabitant VARCHAR2(20),
        CONSTRAINT withHost CHECK (chezHabitant IN ('independent','with_host')),
        nbLits NUMBER(10),
        CONSTRAINT numberBeds CHECK (nbLits > 0)
);


CREATE TABLE Logement (
        idLogement NUMBER(10) PRIMARY KEY,
        prixNuite NUMBER(10,2),
        typeLog NUMBER(10),
        CONSTRAINT tL FOREIGN KEY (typeLog) REFERENCES typesLogement(idType),
        nbMinVoyageurs NUMBER(10),
        CONSTRAINT minTravellers CHECK (nbMinVoyageurs > 0),
        nbMaxVoyageurs NUMBER(10),
        CONSTRAINT maxTravellers CHECK (nbMaxVoyageurs > 0),
        ascenseur VARCHAR2(3),
        CONSTRAINT lift CHECK (ascenseur IN ('yes','no')),
        parking VARCHAR2(3),
        CONSTRAINT park CHECK (parking IN ('yes','no')),
```

```
        television VARCHAR2(3),
        CONSTRAINT tele CHECK (television IN ('yes','no')),
        cuisine VARCHAR2(3),
        CONSTRAINT kitchen CHECK (cuisine IN ('yes','no')),
        wifi VARCHAR2(3),
        CONSTRAINT wlan CHECK (wifi IN ('yes','no')),
        reservationDirecte VARCHAR2(3),
        CONSTRAINT res CHECK (reservationDirecte IN ('yes','no'))
);

/*create shared dimension Utilisateur represented by
        the virtual views Hote and Client*/
CREATE TABLE Utilisateur (
        idUtilisateur NUMBER(10) PRIMARY KEY,
        nom VARCHAR2(50),
        prenom VARCHAR2(50),
        sexe VARCHAR2(10),
        CONSTRAINT sex CHECK (sexe IN ('male','female','other')),
        dateNaissance DATE,
        villeOrigine VARCHAR(50),
        email VARCHAR2(100),
        langueParlee VARCHAR2(50),
        dateInscription DATE,
        hote VARCHAR2(10),
        CONSTRAINT hosting CHECK (hote in ('Host','Non-Host')),
        superUser VARCHAR2(10),
        CONSTRAINT isSuperUser CHECK (superUser in ('SuperUser','Basic'))
);
```

Nous utilisons des nombres comme SurrogateKeys qui suivent l'ordre des entrées :

```
CREATE TABLE Disponibilite (
        idDisponibilite NUMBER(10) PRIMARY KEY,
        description VARCHAR2(100),
        weekends VARCHAR2(50),
        CONSTRAINT dispoWeek CHECK(weekends IN ('Weekends','Only Weekdays')),
        feries VARCHAR2(50),
        CONSTRAINT dispoHoliday CHECK(feries IN ('Holidays','No Holidays')),
        seuleSaison VARCHAR2(10),
        CONSTRAINT dispoSaison CHECK(
                seuleSaison IN ('','Spring','Summer','Fall','Winter')),
        dureeMax NUMBER(10),
        dureeMin NUMBER(10)
);
```

Après les tables des faits sont ajoutées et chargées avec des contraintes concernant les clés primaires referencées des tables des dimensions.

```
ALTER TABLE Reservation ADD (
CONSTRAINT loge FOREIGN KEY (idLogement) REFERENCES Logement(idLogement),
CONSTRAINT loco FOREIGN KEY (idLocalisation) REFERENCES Localisation(idLocalisatio
CONSTRAINT h FOREIGN KEY (idHote) REFERENCES Utilisateur(idUtilisateur),
CONSTRAINT c FOREIGN KEY (idClient) REFERENCES Utilisateur(idUtilisateur),
CONSTRAINT dD FOREIGN KEY (idDateDebut) REFERENCES Dates(idDate),
CONSTRAINT dF FOREIGN KEY (idDateFin) REFERENCES Dates(idDate)
);
```

```
CREATE TABLE Offre (
idType NUMBER(10),
CONSTRAINT otL FOREIGN KEY(idType) REFERENCES typesLogement(idType),
idLocalisation NUMBER(10),
CONSTRAINT oL FOREIGN KEY(idLocalisation) REFERENCES Localisation(idLocalisation),
idDisponibilite NUMBER(10),
CONSTRAINT oD FOREIGN KEY(idDisponibilite)REFERENCES Disponibilite(idDisponibilite
idHote NUMBER(10),
CONSTRAINT oU FOREIGN KEY(idHote) REFERENCES Utilisateur(idUtilisateur),
idJour NUMBER(10),
CONSTRAINT oDa FOREIGN KEY(idJour) REFERENCES Dates(idDate),
nombreOffres NUMBER(10),
pourcentageLoue NUMBER(3,2),
CONSTRAINT rentedOut CHECK(pourcentageLoue BETWEEN 0.00 AND 1.00)
```

Le fichier *insertion.sql* contient un script pour remplir les tables. On a fait attention de respecter l'esprit des entrepôts en entrant beaucoup de valeurs dans les tables des faits pendant que les dimensions restent rélativement plates.

III.1 Les requêtes

Le code pour les requêtes est contenu dans le fichier *requetes.sql*.
On a d'abord crée les vues virtuelles suivantes pour les dimensions partagées. Pour parler des hôtes par exemple on ne s'intéresse plus à la colonne *hote* de *Utilisateur*, alors elle ne doit pas être intégrée à chaque jointure.

```
-- create virtual views for dateDebut et dateFin of shared dimension Dates
CREATE OR REPLACE VIEW date_Debut AS SELECT * FROM Dates;
CREATE OR REPLACE VIEW date_Fin AS SELECT * FROM Dates;
CREATE OR REPLACE VIEW jour AS SELECT * FROM Dates;

CREATE OR REPLACE VIEW Hote AS SELECT idUtilisateur as idHote,nom,prenom,sexe,
dateNaissance,villeOrigine,email,langueParlee,dateInscription,superUser
FROM Utilisateur WHERE hote = 'Host';
```

```
CREATE OR REPLACE VIEW Client AS SELECT idUtilisateur as IdClient,nom,prenom,sexe,
dateNaissance,villeOrigine,email,langueParlee,dateInscription
FROM Utilisateur WHERE hote = 'Non-Host';

CREATE OR REPLACE VIEW typesReservation AS SELECT * FROM typesLogement;
CREATE OR REPLACE VIEW typesOffre AS SELECT * FROM typesLogement;
```

Requêtes analytiques : Reservations

Le prix moyen (/nuit) des differents logements réservés par type de logement reservé

```
SELECT taille,nbChambres,chezHabitant,nbLits,ROUND(AVG(prixNuit),2) as Prix_Moyen_
        SELECT t.taille,t.nbChambres,t.chezHabitant,t.nbLits,(r.prixTotal/duree) a
        FROM Logement l,Reservation r, typesReservation t
        WHERE l.idLogement = r.idLogement AND l.typeLog = t.idType) rl
GROUP BY ROLLUP(taille,nbChambres,chezHabitant,nbLits);
```

Résultat :

TAILLE	NBCHAMBRES	CHEZHABITANT	NBLITS	PRIX_MOYEN_NUIT
room	1	with host	1	901.5
room	1	with host	3	1120
room	1	with host		967.05
room	1	independent	2	258.73
room	1	independent	3	440
room	1	independent	4	2195.18
room	1	independent		963.27
room	1			964.84
room				964.84
house	1	with host	3	1574.66
house	1	with host		1574.66

TAILLE	NBCHAMBRES	CHEZHABITANT	NBLITS	PRIX_MOYEN_NUIT
house	1	independent	4	13528.1
house	1	independent		13528.1
house	1			9045.56
house				9045.56
appartment	1	with host	3	226.07
appartment	1	with host		226.07
appartment	1			226.07
appartment				226.07
				3836.13

```
20 rows selected.
```

Le prix moyen (/nuit) des differents logements réservés par ville

```
SELECT ville ,ROUND(AVG(prixNuit),2) as prix_moyen_nuit FROM (
        SELECT lo.ville ,(r.prixTotal/duree) as prixNuit
        FROM Localisation lo, Reservation r
        WHERE lo.idLocalisation = r.idLocalisation) rlo
GROUP BY ville;
```

Résultat :

```
/* Prix moyen par ville */

VILLE    PRIX_MOYEN_NUIT
_____________________________

Steinkjer
             26

Montpellier
          7130.74

New York City
           794.58
```

Nombre de logements réservés par ville par type

```
SELECT lo.ville ,t.taille ,t.nbChambres,t.chezHabitant,t.nbLits,COUNT(*) FROM
        Localisation lo, Logement l, Reservation r, typesReservation t
        WHERE lo.idLocalisation = r.idLocalisation AND l.idLogement = r.idLogement
                AND l.typeLog = t.idType
        GROUP BY CUBE(lo.ville ,t.taille),t.nbChambres,t.chezHabitant,t.nbLits;
```

Résultat :

TAILLE	NBCHAMBRES	CHEZHABITANT	NBLITS	VILLE	COUNT(*)
	1	with host	1		7
room	1	with host	1		7
	1	with host	1	Montpellier	6
room	1	with host	1	Montpellier	6
	1	with host	1	New York City	1
room	1	with host	1	New York City	1
	1	with host	3		13
room	1	with host	3		3
house	1	with host	3		6

appartment	1	with host	3		4
	1	with host	3	Montpellier	5

TAILLE	NBCHAMBRES	CHEZHABITANT	NBLITS	VILLE	COUNT(*)
house	1	with host	3	Montpellier	5
	1	with host	3	New York City	8
room	1	with host	3	New York City	3
house	1	with host	3	New York City	1
appartment	1	with host	3	New York City	4
	1	independent	2		8
room	1	independent	2		8
	1	independent	2	New York City	8
room	1	independent	2	New York City	8
	1	independent	3		3
room	1	independent	3		3

TAILLE	NBCHAMBRES	CHEZHABITANT	NBLITS	VILLE	COUNT(*)
	1	independent	3	Steinkjer	1
room	1	independent	3	Steinkjer	1
	1	independent	3	Montpellier	1
room	1	independent	3	Montpellier	1
	1	independent	3	New York City	1
room	1	independent	3	New York City	1
	1	independent	4		15
room	1	independent	4		5
house	1	independent	4		10
	1	independent	4	Montpellier	9
house	1	independent	4	Montpellier	9

TAILLE	NBCHAMBRES	CHEZHABITANT	NBLITS	VILLE	COUNT(*)
	1	independent	4	New York City	6
room	1	independent	4	New York City	5
house	1	independent	4	New York City	1

Requêtes analytiques : Offres

Nombre de logements offerts par des SuperHosts par ville le 2 juillet 2018

```
SELECT l.ville ,SUM(o.idType) as nombreLogements FROM Offre o,Localisation l,Hote h
WHERE o.idLocalisation = l.idLocalisation AND o.idHote = h.idHote AND
                o.idJour = j.idDate AND j.dateEntier='02-july-2018' AND
                h.superUser = 'SuperUser'
GROUP BY l.ville;
```

Résultat :

VILLE	NOMBRELOGEMENTS
Steinkjer	7
Montpellier	11
New York City	1500

Le type de logement le plus souvent offert le 2 juillet 2018

```
SELECT t.taille as type,SUM(o.nombreOffres) as nombre_offres
FROM Offre o, typesLogement t, jour j
WHERE o.idType = t.idType AND o.idJour = j.idDate AND
                j.dateEntier = '02-july-2018'
GROUP BY (t.taille)
HAVING SUM(o.nombreOffres) =
(SELECT MAX(SUM(o.nombreOffres))
FROM Offre o, typesLogement t, jour j
WHERE o.idType = t.idType AND o.idJour = j.idDate AND
                j.dateEntier = '02-july-2018'
GROUP BY (t.taille));
```

Résultat :

TYPE	NOMBRE_OFFRES
room	235

Le nombre de logements offerts disponibles le weekend par ville le 2 juillet 2018

```
SELECT lo.ville as ville, SUM(o.nombreOffres) as nombre_offres
FROM Offre o, Localisation lo, Disponibilite d, jour j
WHERE o.idLocalisation = lo.idLocalisation AND o.idDisponibilite = d.idDisponibili
        AND j.idDate = o.idJour AND j.dateEntier = '02-july-2018'
        AND d.weekends = 'Weekends'
GROUP BY lo.ville;
```

Résultat :

VILLE	NOMBRE_OFFRES
Steinkjer	10
Montpellier	69
New York City	418