NF17 : Modélisation d'une base de donnée pour une agence touristique

Quentin Leprat

Juin 2020

1 Test sur la BDD

L'objectif de ce document est d'expliquer les différents tests effectués sur la BDD notemment les vues demandées par le client.

1.1 Statistiques sur la BDD

Dans sa BDD, le client a besoin de pouvoir connaître le nombre de réservation par mois ou encore connaître les circuits qui ont eu le plus de réservation.

Pour cela, deux vues ont été mise en place : **vCircuitFamous** ainsi que **vNombreReservation**.

1.1.1 Test des fonctionnalités de vNombreReservation

La question a laquelle nous devons répondre est la suivante : quel est le nombre de réservation payée au total par mois pour tous les circuits de l'agence ?

On dispose de la table suivante :

RESERVATION

id:integer	client:text	circuittouristique:integer	status:text	date_emission:date	nombre_personne:integer
1	598072452892409	2	Reservé	2019-12-09	2
2	598072452892409	2	Reservé	2019-12-07	4
3	598072452892409	2	Payé	2019-11-11	7
4	598072452892409	2	Payé	2019-11-11	9
5	598072452892409	2	Payé	2019-12-07	3
11	198074722520893	3	Reservé	2018-12-09	10
21	598072452892409	2	Reservé	2019-12-07	4
31	198074722520893	2	Payé	2017-11-11	7
41	198074722520893	3	Payé	2017-11-11	15
51	598072452892409	2	Payé	2019-12-07	50

FIGURE 1 – Table SQL des réservations

On remarque tout d'abord qu'il y a 6 réservations qui ont le status "payé", d'autre part, ces réservations ont été effectué les mois de Décembre et Novembre, respectivement 2 en Décembre et 4 en Novembre. Voici le code SQL de la vue qui permet d'automatiser ce calcul :

```
CREATE VIEW vNombreReservation
AS SELECT COUNT(*) AS NombreReservation, to_char(date_emission,'Mon') as Mon
FROM Reservation
WHERE status = 'Payé'
GROUP BY Mon;
```

On obtient bien le résultat escompté :

VNOMBRERESERVATION

nombrereservation	mon
2	Dec
4	Nov

FIGURE 2 – Table SQL de la vue vNombreReservation

1.1.2 Test des fonctionnalités de vCircuitFamous

La question a laquelle nous devons répondre est la suivante : quelle est le circuit qui a reçu le plus de personne à l'année N ?

On dispose de la table SQL des réservations (Figure 1), dans cette table figure les réservations et le nombre de participants inscrits (nombre personne). Il suffit alors de compter pour chaques réservations (qui ont le status payé) associées à chaque circuit, combien il y a eu de personne en tout à une année donnée.

Dans notre cas, il y a eu 4 réservations pour le circuit 2 en 2019, pour un total de $7+9+3+50 = 69$ et 1 réservation en 2017 pour un total de 7 personnes, et il y a eu 1 réservation pour le circuit 3 en 2017 pour un total de 15 personnes.

En résumé, on veut une table SQL qui nous donne, par année, le nombre total de personne qui ont participé à chaque circuit. Voici le code SQL en vue d'automatiser ce calcul :

```
CREATE VIEW vCircuitFamous
AS SELECT  circuittouristique,
SUM(nombre_personne) AS NbPersonne,
extract(year from date_emission) AS yyyy
FROM Reservation
WHERE status = 'Payé'
GROUP BY circuittouristique,yyyy
ORDER BY NbPersonne DESC;
```

On obtient bien le résultat escompté :

VCIRCUITFAMOUS

circuittouristique	nbpersonne	yyyy
2	69	2019
3	15	2017
2	7	2017

FIGURE 3 – Table SQL de la vue vCircuitFamous

1.2 Vues et attributs JSON

1.2.1 Vue SQL sur les antécédents et adresses

On dispose cette fois-ci des tables Clients et RessourcesHumaines.

nss:text	nom:text	prenom:text	date_naissance:date
198072722924031	Leprat	Quentin	1998-07-01
198074722520893	Durand	Antoine	2007-09-01
598072452892409	Renard	Vincent	1998-08-05

adresse:json
[{"Adresse" : "18 rue de l abbaye", "Ville" : "Ivry la Bataille", "Code postal" : "27540"}]
[{"Adresse" : "44 Rue Henry Litolff","Ville" : "Colombes", "Code postal" : "92270"}]
[{"Adresse" : " 5 Rue des Rosiers","Ville" : " Verneuil-l Étang", "Code postal" : "77390"}]

FIGURE 4 – Table SQL de RessourceHumaine

nss:text	num_tel:integer	adresse:text
198074722520893	689784565	Boulevard de la république
598072452892409	658785369	Rue des belles femmes

antecedent:json
[{"Maladie" : "asthme", "Allergie" : "Produit laitiers"}]
[{"Maladie" : "", "Allergie" : "Acarien", "Régime alimentaire" : "viande sans porc"}]

FIGURE 5 – Table SQL de Client

Un client dispose d'un attribut JSON, qui regroupe ses antécédents (maladie, régime alimentaire etc) et une RessourceHumaine (dont hérite un client), possède un attribut JSON permettant de stocker son adresse.

L'objectif est donc de récupérer, pour chaque client, son nom, adresse et ses antécédents et de la stocker une table SQL.
Par exemple, prenons le client qui a pour NSS : 598072452892409, on va rechercher dans la table RessourcesHumaines, le nom de la personne associée à ce NSS, et on tombe sur Vincent Renard, il suffit alors de récupérer ses antécédents et son adresse. Voici alors le code SQL permettant de le faire (pour tous les clients) :

```
CREATE VIEW vAntecedent_adresse AS SELECT
    r.nom,
    ad ->> 'Adresse' AS adresse,
    ad ->> 'Ville' AS ville,
    ad ->> 'Code postal' AS cp,
    a  ->> 'Maladie' AS maladie,
    a  ->> 'Allergie' AS allergie,
    a  ->> 'Régime alimentaire' AS régime_alimentaire
    FROM RessourceHumaine r,
         Client c ,
         JSON_ARRAY_ELEMENTS(c.antecedent) a,
         JSON_ARRAY_ELEMENTS(r.adresse) ad
    WHERE c.NSS = r.NSS;
```

On obtient la vue SQL suivante :

VANTECEDENT_ADRESSE

nom	adresse	ville	cp	maladie	allergie	régime_alimentaire
Renard	5 Rue des Rosiers	Verneuil-l Étang	77390		Acarien	viande sans porc
Durand	44 Rue Henry Litolff	Colombes	92270	asthme	Produit laitiers	

1.2.2 Vue sur les circuits touristiques

On dispose de la table CircuitTouristique :

CIRCUITTOURISTIQUE

id:integer	type_circuit:text	date_depart:date	duree:text	nb_max:integer	difficulte:integer	description:json
1	Aquatique	2020-05-30	7 jours	25	5	[{"Région" : "PACA", "Nom" : "Circuit DELTA", "Activités" : " [Plongée sous marine, Aquaponey, parc aquatique]"}]
2	Montagne	2020-02-12	15 jours	18	3	[{"Région" : "Haute-savoie", "Nom" : "Circuit TANGO", "Activités" : "[Ski, randonnée pedestre, Initiation aux remontées mécaniques]"}]
3	Aviation	2020-07-30	7 jours	10	1	

FIGURE 7 – Table SQL de CircuitTouristique

On souhaiterait afficher pour chaque circuit, le type de circuit (aquatique, montagne, etc) ainsi que sa description qui est au format JSON, pour cela on dispose de la vue vCircuitTouristique dont le code SQL est le suivant :

```
CREATE VIEW vCircuitTouristique AS SELECT
    c.type_circuit,
    d->>'Région' AS Région,
    d->>'Nom' AS Nom,
    d->>'Activités' AS Activites
    FROM CircuitTouristique c, JSON_ARRAY_ELEMENTS(c.description) d;
```

Le résultat de ce code SQL est contenu dans la vue suivante :

VCIRCUITTOURISTIQUE

type_circuit	région	nom	activites
Aquatique	PACA	Circuit DELTA	[Plongée sous marine, Aquaponey, parc aquatique]
Montagne	Haute-savoie	Circuit TANGO	[Ski, randonnée pedestre, Initiation aux remontées mécaniques]

FIGURE 8 – Table SQL de vCircuitTouristique

On obtient bien une table SQL créée à partir de l'attribut JSON description ! D'autre part, on remarque pour le dernier circuit de la Figure 7, qu'il ne figure pas sur le résultat de la vue SQL vCircuitTouristique, tout simplement car il ne possède pas de description.

2 Vues particulières : héritage par classe mère.

Les tables Transport, Equipement et Logement héritent par la classe mère Société, donc ces tables n'existent pas vraiment en tant que tel dans la BDD, il y a juste un attribut t dans la table Société pour les différencier :

SOCIETE

siren:varchar(14)	nom:text	reputation:text	t:USER-DEFINED
48170829500037	ParapenteLOC	Bonne	Equipement
32212091600208	AutoBUS	Moyenne	Transport
30264000800017	HAUTel	Bonne	Hebergement

FIGURE 9 – Table SQL de Société

Pour pouvoir les visualiser, on dispose des vues vHebergeur, vEquipementier et vTransporteur.

VHEBERGEUR

siren	nom	reputation	t
30264000800017	HAUTel	Bonne	Hebergement

[vue] SELECT societe.siren, societe.nom, societe.reputation, societe.t FROM societe WHERE (societe.t = 'Equipement'::type_societe);

VEQUIPEMENTIER

siren	nom	reputation	t
48170829500037	ParapenteLOC	Bonne	Equipement

[vue] SELECT societe.siren, societe.nom, societe.reputation, societe.t FROM societe WHERE (societe.t = 'Transport'::type_societe);

VTRANSPORTEUR

siren	nom	reputation	t
32212091600208	AutoBUS	Moyenne	Transport

FIGURE 10 – Tables SQL des classes héritées par Société