BestMarket

RetailInsight360

Simon

Doussin

01/06/2022

1) Contexte et expression du besoin

Bestmarket est une entreprise de grande distribution. Elle souhaite utiliser les données des retours et avis de ses clients pour améliorer la qualité de son réseau de magasins et améliorer son service client.

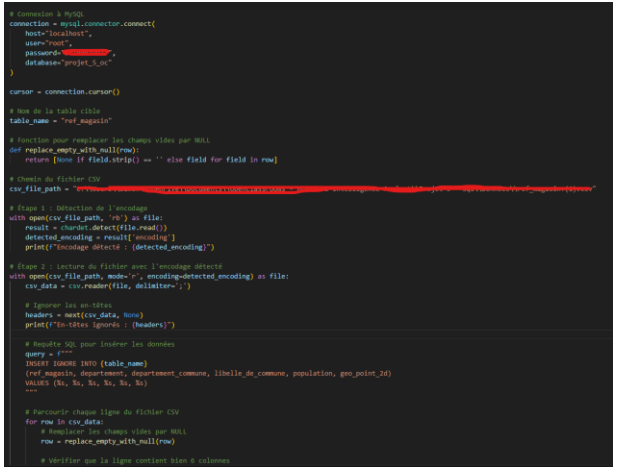
Expression des besoins par :

Client / Note / Magasin / Produit / NPS

2) Sauvegarde et stockage de la BDD

Utilisation de VSCode

Utilisation de python pour importer la 3e table :

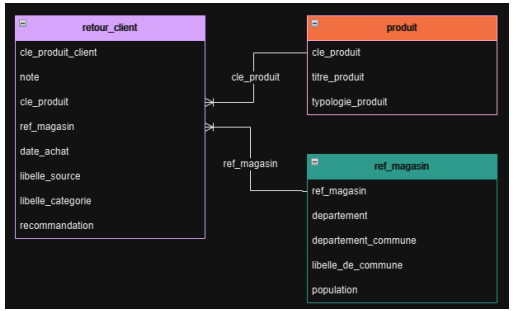


3) Méthodologie suivie

1) Création de la base de données
2) Importation des 2 tables (*retour_client* et *produit*)
3) Importation de la 3e table (*ref_magasin*) via python
4) Création du schéma et du dictionnaire de donnée
5) Vérification de la qualité des données (doublon, valeur extrême)
6) Requête SQL

4) Requêtes SQL et Analyses

Création du schéma de donnée :

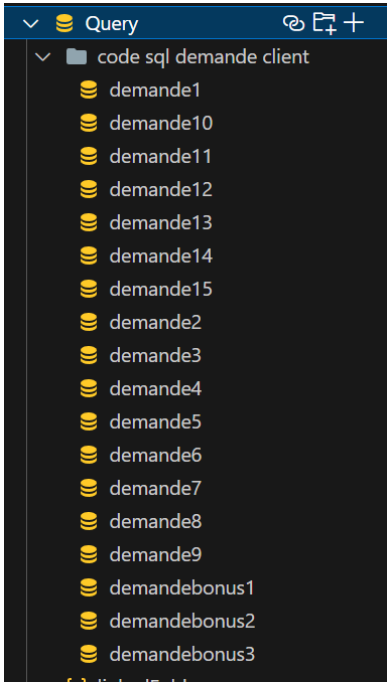


Création du dictionnaire de donnée :

	Nom du champs	Type de données	Taille	Contrainte	Description
Table Retour client	cle_retour_client	INT		Clé primaire	ID unique pour les retours clients
	note	INT			Note donnée par le client, comprise entre 0 et 10, la note est la réponse à la question : "Sur une échelle de 0 à 10 quelle est la probabilité que vous recommandiez notre entreprise à votre entourage ?"
	Clé_produit	INT			ID des produits
	ref_magasin	INT			ID des magasins
	date_achat	DATE			Date à laquelle l'achat du client a eu lieu
	libelle_source	CHAR	50		Libellé de la source d'où provient le retour client (Réseaux sociaux, téléphone, email)
	libelle_categorie	CHAR	50		Libellé de la catégorie du retour client (Drive, service après-vente, qualité produit, expérience en magasin, livraison)
	recommandation	CHAR			Recommandation laissée par le client à la question 'Recommandez vous l'entreprise?' True / False
Table Produit	cle_produit	INT		Clé primaire	ID unique pour les produits
	titre_produit	CHAR	50		Libellé des produits
	typologie_produit	INT			Typologie des produits (Alimentaire, High-tech etc...)
Table Ref_magasin	ref_magasin	INT		Clé primaire	ID unique pour les magasins
	departement	INT			Numéro du département
	departement_commune	INT			Code postal de la commune
	libelle_de_commune	CHAR	50		Nom de la commune
	population	INT			Nombre d'ahabitant

4) Requêtes SQL et Analyses

Traitement de la demande du client :



4) Requêtes SQL et Analyses : Client

Retour client :

```sql
---- Demande n°10 : Sur quel mois a-t-on le plus de retour sur le service après-vente ? ----
SELECT MONTH(date_achat) as mois, COUNT(*) as nb_retour
FROM retour_client
WHERE libelle_categorie = 'service après-vente'
GROUP BY MONTH(date_achat)
ORDER BY nb_retour DESC  4ms
;
```

retour_client

Search Results | Export | Cost: 7ms | 1 | Total 12

mois	nb_retour
10	55
9	53
6	53
8	52
3	52
11	52
1	52
5	52
4	52
7	48
2	44
12	38

Les mois d'octobre, septembre et juin sont les mois ayant le plus de retour SAV.

```sql
---- Demande bonus n°1 : Quel est le nombre de retour clients par source ? ----
SELECT libelle_source, COUNT(*) AS nb_retour
FROM retour_client
GROUP BY libelle_source
ORDER BY nb_retour DESC  6ms
;
```

retour_client

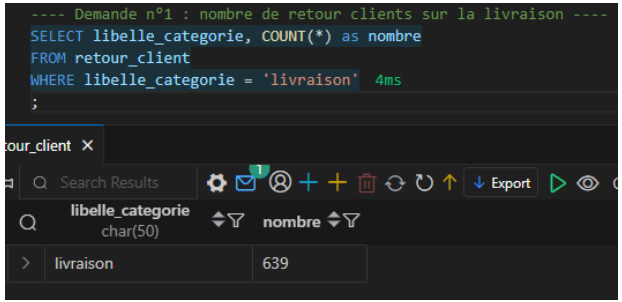
Search Results | Export | Cost: 9ms | 1

libelle_source char(50)	nb_retour
email	1032
réseaux sociaux	998
téléphone	970

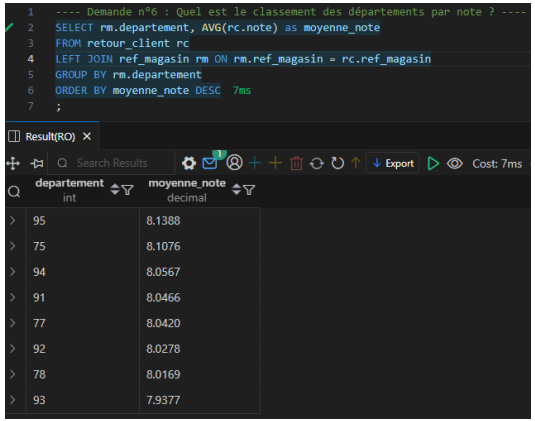
On compte 1032 retours clients par email, 998 par les réseaux sociaux et 970 par téléphones.

4) Requêtes SQL et Analyses : Client

Retour client :



Le nombre de retour client sur la livraison est de 639.
(drive : 611, service A-V : 603)



Les départements n°95, 75 et 94 ont, en moyenne, les meilleurs notes.

4) Requêtes SQL et Analyses : Client

Expérience client :

```sql
---- Demande n°9 : Quel est le classement des jours de la semaine où l'expérience client est la meilleure expérience en magasin ? ----
SELECT DAY(date_achat) as jour, AVG(note) as moyenne_note
FROM retour_client
WHERE libelle_categorie = 'expérience en magasin'
GROUP BY DAY(date_achat)
ORDER BY moyenne_note DESC  6ms
;
```

retour_client

Search Results | Export | Cost: 9ms | 1 | Total 31

jour	moyenne_note
31	9.5000
10	8.5238
7	8.4737
13	8.4444
11	8.4286
2	8.4286
24	8.4000
6	8.2632
21	8.2381
3	8.2308
5	8.2000
18	8.1852
9	8.1333
16	8.1304
19	8.1176

Il semblerait que le 31, le 10 et le 7 de chaque mois les clients ont, en moyenne, une meilleure expérience en magasin.

Recommandation client :

```sql
CREATE VIEW retour_client_reco_1 AS
SELECT *
FROM retour_client
WHERE recommandation = '1'
;

SELECT ROUND((COUNT(rcr1.cle_retour_client) * 100) / COUNT(rc.cle_retour_client),2) as pourcentage_recommandation
FROM retour_client rc
LEFT JOIN retour_client_reco_1 rcr1 ON rc.cle_retour_client = rcr1.cle_retour_client
ORDER BY pourcentage_recommandation DESC  8ms
;
# % recommandation client = 70,5
```

Result(RO)

Search Results | Export | Cost: 8ms | 1 | Total 1

pourcentage_recomm decimal
70.50

70,5% des clients recommande BestMarket.

4) Requêtes SQL et Analyses : Note

```sql
---- Demande n°2 : Quelle est la liste des notes des clients sur les réseaux sociaux sur les TV ? --
SELECT rc.note, COUNT(*) as nombre
FROM retour_client rc
LEFT JOIN produit p ON p.cle_produit = rc.cle_produit
WHERE rc.libelle_source = 'réseaux sociaux'
AND p.titre_produit = 'TV'
GROUP BY rc.note  8ms
;
```

Result(RO)

Search Results | Export | Cost: 8ms | 1 | Total 3

note int	nombre bigint
10	2
9	1
8	1

Sur les réseaux sociaux, les clients ont mis une note de 10 à 2 reprises, et une note de 9 et 8 à 1 reprise.

4) Requêtes SQL et Analyses : Note

Note selon le produit :

```sql
---- Demande n°3 : Quelle est la note moyenne pour chaque catégorie de produit ?
SELECT p.typologie_produit, AVG(rc.note) as moyenne_note
FROM retour_client rc
LEFT JOIN produit p ON p.cle_produit = rc.cle_produit
GROUP BY p.typologie_produit
ORDER BY moyenne_note DESC  12ms
;
```

Result(RO)

Search Results | Export | Cost: 12ms | 1

typologie_produit (string)	moyenne_note (decimal)
High-Tech	8.1607
Loisirs	8.0904
Alimentaire	8.0418
Maison	7.8507

Les produits high-techs ont une note moyenne de 8.2, les loisirs : 8.1, les produits alimentaires : 8 et les produits de maison : 7.85

```sql
SELECT p.titre_produit, AVG(rc.note) as moyenne_note
FROM retour_client rc
LEFT JOIN produit p ON p.cle_produit = rc.cle_produit
GROUP BY p.titre_produit
ORDER BY moyenne_note DESC  11ms
;
```

Result(RO)

Search Results | Export

titre_produit (string)	moyenne_note (decimal)
Plantes aromatiques surgelé	9.2000
Sodas	9.0000
Boissons alcoolisées	8.7895
Sauces au soja	8.7727
Aliments à base de plantes	8.7143
TV	8.7143
Aliments à base de plantes	8.6842
Petit-déjeuners	8.6000
Concentré de tomate	8.6000
Plats au bœuf	8.5714

Les plantes aromatiques surgelé ont la meilleure note moyenne (9.2) suivi des sodas (9) et des boissons alcoolisées (8.8)

4) Requêtes SQL et Analyses : Note

```sql
---- Demande n°8 : Quelle est la note moyenne sur l'ensemble des boissons ? ----
SELECT p.titre_produit, AVG(rc.note) as moyenne_note
FROM retour_client rc
LEFT JOIN produit p ON p.cle_produit = rc.cle_produit
WHERE p.titre_produit LIKE ('Boissons%')
GROUP BY p.titre_produit
ORDER BY moyenne_note DESC
;

SELECT AVG(rc.note) as moyenne_note
FROM retour_client rc
LEFT JOIN produit p ON p.cle_produit = rc.cle_produit
WHERE p.titre_produit LIKE ('Boissons%')
ORDER BY moyenne_note DESC  7ms
;
```

Result(RO) ×

Search Results | Export | Cost: 7ms | 1

moyenne_note decimal
8.3208

Parmi les boissons, celles alcoolisées ont les meilleurs notes en moyenne (8.8).

```sql
-- Liste des produit ayant des notes inférieure ou égale à 6 --
SELECT rc.ref_magasin, rc.libelle_categorie, rc.cle_produit, p.typologie_produit, p.titre_produit,  rc.n
FROM retour_client rc
LEFT JOIN produit p oN p.cle_produit = rc.cle_produit
GROUP BY rc.ref_magasin, rc.libelle_categorie, rc.cle_produit, p.typologie_produit, p.titre_produit, rc.
HAVING rc.note <= 6
ORDER BY p.typologie_produit ASC, rc.note ASC  17ms
;
```

Result(RO) ×

Search Results | Export | Cost: 17ms | 1 2 3 | Total 271

ref_magasin int	libelle_categorie string	cle_produit int	typologie_produit string	titre_produit string	note int
5	drive	115	Alimentaire	Viandes	0
57	service après-vente	63	Alimentaire	Légumineuses	0
20	drive	119	Alimentaire	Vinaigres d'alcools	0
58	livraison	36	Alimentaire	Exhausteurs de goût	0
36	livraison	80	Alimentaire	Pickles d'origine végétale	0
32	drive	47	Alimentaire	Fruits tropicaux	0
51	service après-vente	108	Alimentaire	Sucres	0
64	service après-vente	79	Alimentaire	Pickles	0
14	livraison	104	Alimentaire	Snacks salés	0

Voici une liste des produits qui ont une note inférieure à 6.

4) Requêtes SQL et Analyses : Magasin

```sql
---- Demande n°4 : Quels sont les 5 magasins avec les meilleures notes moyennes ? ----
SELECT ref_magasin, AVG(note) as moyenne_note
FROM retour_client
GROUP BY ref_magasin
ORDER BY moyenne_note DESC
LIMIT 5  6ms
;
```

retour_client

Search Results | Export | Cost: 10ms | 1 2 3

ref_magasin int	moyenne_note
75	8.7273
78	8.5484
62	8.5000
23	8.4839
19	8.4524

Le magasin n°75, 78 et 62 ont les meilleurs note moyenne.

```sql
---- Demande n°5 : Quels sont les magasins qui ont plus de 12 feedbacks sur le drive ? ----
SELECT ref_magasin, COUNT(*) as nb_feedback
FROM retour_client
WHERE libelle_categorie = 'drive'
GROUP BY ref_magasin
HAVING nb_feedback > 12
ORDER BY nb_feedback DESC  5ms
;
```

retour_client

Search Results | Export | Cost: 9ms | 1 | Total 3

ref_magasin int	nb_feedback
67	14
63	13
45	13

Au niveau du drive : les magasins n°67, 63 et 45 ont plus de 12 feedbacks.

4) Requêtes SQL et Analyses : Magasin

```sql
--- Liste magasin dynamique selon note moyenne ---
CREATE VIEW note_moyenne AS
SELECT AVG(note) as moyenne_note
FROM retour_client
;
SELECT rc.ref_magasin, AVG(rc.note) as moyenne_note_magasin
FROM retour_client rc
GROUP BY rc.ref_magasin
HAVING AVG(note) < (SELECT moyenne_note FROM note_moyenne)
ORDER BY moyenne_note_magasin DESC  8ms
;
```

Result(RO)

Export

ref_magasin int	moyenne_note_maga... decimal
50	8.0513
36	8.0500
3	8.0357
47	8.0286

Voici la liste des magasins ayant une note inférieure à la moyenne.

```sql
---- Demande bonus n°2 : Quels sont les 5 magasins avec le plus de feedbacks ? ----
SELECT ref_magasin, COUNT(*) AS nb_retour
FROM retour_client
GROUP BY ref_magasin
ORDER BY nb_retour DESC
LIMIT 5  4ms
;
```

retour_client

Search Results

Export

Cost: 6ms

1 2

ref_magasin int	nb_retour
29	55
6	49
80	47
5	45
63	44

Les magasins n°29, 6, 80, 5 et 63 ont eu le plus de feedbacks.

4) Requêtes SQL et Analyses : Produit

```sql
---- Demande n°7 : Quelle est la typologie de produit qui apporte le meilleur service après-vente ? ----
SELECT p.typologie_produit, rc.libelle_categorie, AVG(rc.note) as moyenne_note
FROM retour_client rc
LEFT JOIN produit p ON p.cle_produit = rc.cle_produit
WHERE rc.libelle_categorie = 'service après-vente'
GROUP BY p.typologie_produit
ORDER BY moyenne_note DESC  7ms
;
```

Result(RO) ×

Search Results | Export | Cost: 7ms | 1 | Total 4

typologie_produit (string)	libelle_categorie (string)	moyenne_note (decimal)
Loisirs	service après-vente	8.5135
High-Tech	service après-vente	8.1231
Alimentaire	service après-vente	8.0289
Maison	service après-vente	7.8824

Les produits de type loisirs apporte le meilleur service après-vente avec une note moyenne de 8.5.

```sql
---- Demande n°13 : Quelles sont les typologies produits qui ont amélioré leur moyenne entre le 1er et l
SELECT np2t.typologie_produit, np1t.moyenne_note_1er_tri, np2t.moyenne_note_2e_tri
FROM note_produit_1er_trimestre np1t
LEFT JOIN note_produit_2e_trimestre np2t ON np2t.typologie_produit = np1t.typologie_produit
HAVING np2t.moyenne_note_2e_tri > np1t.moyenne_note_1er_tri  13ms
;
```

Result(RO) ×

Search Results | Export | Cost: 13ms | 1 | Total 2

typologie_produit (string)	moyenne_note_1er_tri (decimal)	moyenne_note_2e_tri (decimal)
Alimentaire	7.9932	8.0380
Loisirs	8.0000	8.1753

Les produits de type alimentaire et loisirs ont amélioré leur note moyenne entre le 1er et le 2e trimestre.

4) Requêtes SQL et Analyses : NPS

```sql
-- Score NPS --
SELECT *,
    CASE
        WHEN nps = 'promoteurs' THEN pourcentage_tg - (
            SELECT pourcentage_tg
            FROM nps_nb_pourcent
            WHERE nps = 'détracteurs'
        )
        ELSE ''
    END AS score_nps
FROM nps_nb_pourcent  39ms
;
```

Result(RO)

Search Results | Export

nps varchar	nombre bigint	pourcentage_tg decimal	score_nps varchar
passifs	1529	50.97	
promoteurs	1200	40.00	30.97
détracteurs	271	9.03	

Le score NPS est de 30,97, les clients sont donc en général satisfaits des magasins mais la fidélisation des clients reste à améliorer.

```sql
-- Score NPS par source --
SELECT *,
    CASE
        WHEN nps = 'promoteurs' AND libelle_source = 'email' THEN pourcentage_tg - (
            SELECT pourcentage_tg
            FROM nps_source_nb_pourcent
            WHERE nps = 'détracteurs' AND libelle_source = 'email'
        )
        WHEN nps = 'promoteurs' AND libelle_source = 'réseaux sociaux' THEN pourcentage_tg - (
            SELECT pourcentage_tg
            FROM nps_source_nb_pourcent
            WHERE nps = 'détracteurs' AND libelle_source = 'réseaux sociaux'
        )
        WHEN nps = 'promoteurs' AND libelle_source = 'téléphone' THEN pourcentage_tg - (
            SELECT pourcentage_tg
            FROM nps_source_nb_pourcent
            WHERE nps = 'détracteurs' AND libelle_source = 'téléphone'
        )
        ELSE ''
    END AS score_nps_source
FROM nps_source_nb_pourcent  19ms
;
```

Result(RO)

Search Results | Export | Cost: 19ms | 1 | Total 9

nps varchar	libelle_source string	nombre bigint	pourcentage_tg decimal	score_nps_source varchar
promoteurs	email	387	37.50	29.65
passifs	email	564	54.65	
détracteurs	email	81	7.85	
promoteurs	réseaux sociaux	410	41.08	29.56
passifs	réseaux sociaux	473	47.39	
détracteurs	réseaux sociaux	115	11.52	
passifs	téléphone	492	50.72	
promoteurs	téléphone	403	41.55	33.82
détracteurs	téléphone	75	7.73	

5) Cohérence des données

Aucun doublon ou valeurs nulles dans les données, ni de valeurs aberrantes.

```sql
SELECT *
FROM produit
WHERE titre_produit IS NULL
; #aucune valeure nulle
```

Exemple de requête pour les valeurs nulles.

```sql
--- Rechercher les doublons ---
SELECT COUNT(*) as nb_doublon, typologie_produit, titre_produit
FROM produit
GROUP BY typologie_produit, titre_produit
HAVING nb_doublon > 1
; #aucun doublon
```

Exemple de requête pour les doublons.

```sql
--- Rechercher valeur aberrant ---
SELECT DISTINCT (note)
FROM retour_client
; #aucune valeure aberrante
SELECT MIN(note)
FROM retour_client
; #aucune valeure aberrante
SELECT MAX(note)
FROM retour_client
; #aucune valeure aberrante
```

Exemple de requête pour les valeurs aberrantes.