Managing Databases : SQL & NoSQL

Etude d'une plateforme de streaming : HQ-Streaming

Pierre DUMONTEL
Loris BULLIARD
William ROUET
Victor PION

Table des matières

Part 1 : Model 3
Scénario 3
Hypothèses et limites du modèle 4
Dictionnaire des tables 5
Conceptual Data Model 7
Logical Data Model 8
Part 2 : Queries 9
Basic queries 9
WHERE Clause Queries 11
ORDER BY Queries 12
Multi-Table Queries 14
Queries with Numeric expressions and functions 16
Group By Queries 17
Nested Queries 18
Part 3 : mongoDB 20
Séries 20
Caractéristiques globales 20
Caractéristiques spécifiques 23
Caractéristiques selon le nombre d'épisodes 26
Caractéristiques selon les catégories 30
Caractéristiques selon le nombre de saisons 34
Bilan 39
Musiques 40
Visualisation globale des données 40
Information et valeurs extrêmes des likes 51
Information et valeurs extrêmes des dislikes 52
Informations sur la durée des musiques 55
Effet du « The » 57
Rapport likes-dislikes 58
Bilan 61
Clients 62
Caractéristiques générales des clients 62
Application du tarif et du règlement de restriction 75
Eléments de profilage des clients 89
Bilan 91
Conclusion 92

Part 1 : Model

Scénario

Nous allons considérer dans notre étude la plateforme de streaming HQ streaming. Sur cette plateforme sont disponibles des films, des séries ainsi que les musiques de ces dernières. Les clients sont abonnés à la plateforme et peuvent bénéficier d'une réduction selon leur âge. Ils peuvent aussi se voir bloquer l'accès à certains contenus en fonction de leur âge. Les clients sont répertoriés selon leur pays et peuvent bénéficier d'un mois gratuit à l'abonnement. La plateforme promeut ses produits via des publicités qui lui coûtent de l'argent.

Les abonnés peuvent juger les films, les séries et les musiques de manière binaire (like/dislike). Les films peuvent même recevoir des distinctions classées par catégories qui sont distribuées par HQ streaming. Les films de la plateforme sont supprimés s'ils ne respectent pas un certain nombre de vues. Tandis que les séries sont-elles supprimées si leur coût de stockage est trop élevé. Il existe également un système de recommandation qui permet à un utilisateur de se voir conseiller un film similaire de celui qu'il vient de regarder, de même pour les séries. Sachant que la plateforme associe des films entre eux ainsi que des séries entre elles pour les recommander aux utilisateurs.

La plateforme HQ streaming fait face à l'émergence de nouveaux concurrents sur le marché du streaming de films et de séries ainsi que de la musique. Le siège social de HQ streaming qui se situe dans les Alpes-Maritimes souhaite donc analyser les données de sa plateforme afin de maximiser son fonctionnement et d'optimiser ses parts de marché. Des ingénieurs informaticiens employés par la plateforme récupèrent les données concernant les films, les séries, les musiques, les vues des films, le stockage des séries, les suppressions de contenus, les clients, les publicités ou encore différents types d'abonnements.

Les ingénieurs informaticiens en traitant ces données vont notamment chercher à savoir quels sont les films et séries qui sont les plus regardés, les musiques les plus écoutées, les profils les plus récurrents chez les abonnés ou bien encore les contenus les plus appréciés de la plateforme. Cela aura pour but final de conseiller les dirigeants de HQ streaming afin qu'ils adoptent les meilleures stratégies et prennent les meilleures décisions pour développer la plateforme.

Hypothèses et limites du modèle

On a travaillé avec un modèle simplifié d'entreprise et une base de données restrictive. Nous n'avions pas les pré-requis nécessaires à la création complète d'une plateforme de streaming ou d'une création d'entreprise en générale. De plus une telle entreprise n'aurait pas tenu dans un maximum de 30 tables. C'est pourquoi l'hypothèse directrice de notre devoir est basée sur la relation entre clients et produits proposés par la plateforme.

Puis les autres hypothèse étaient de proposer une version honnête mais simplifiée de notre hypothèse directrice :
-classification clientèle
-interaction entre les actions d'un client et ce que la plateforme peut lui proposer comme contenu.
-gestion des produits présents sur la plateforme.

En ce qui concerne la quantité des données : nous avons fait en sorte d'avoir suffisamment d'éléments pour couvrir toutes nos requêtes. Mais elle reste bien évidemment très limitée et en deçà de ce qu'elle pourrait être dans un cas pratique.

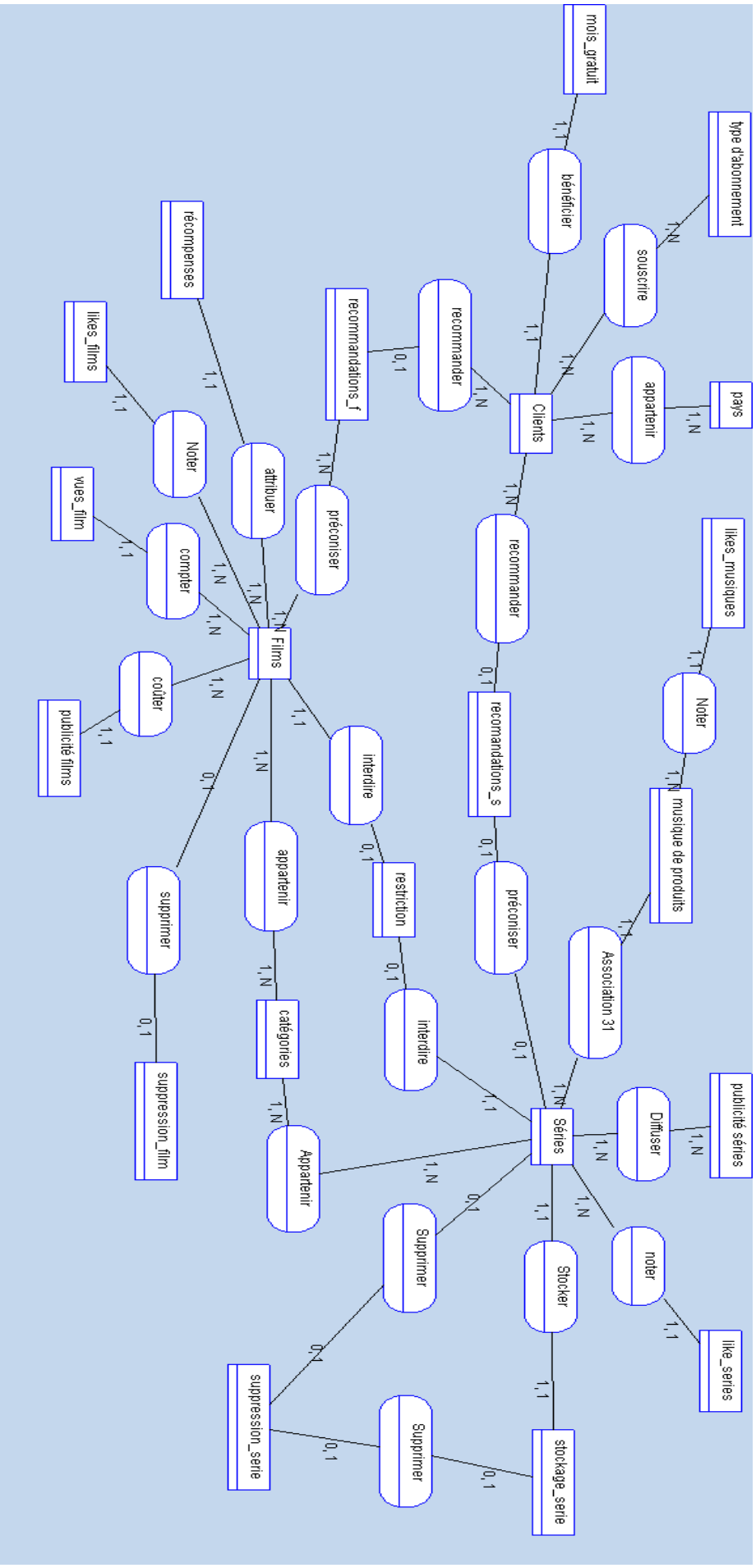
Concernant la normalisation, nous avons fait en sorte que la base de données ne comprenne pas de doublons tout en garantissant les relations logiques entre les tables de données.

Dictionnaire des tables

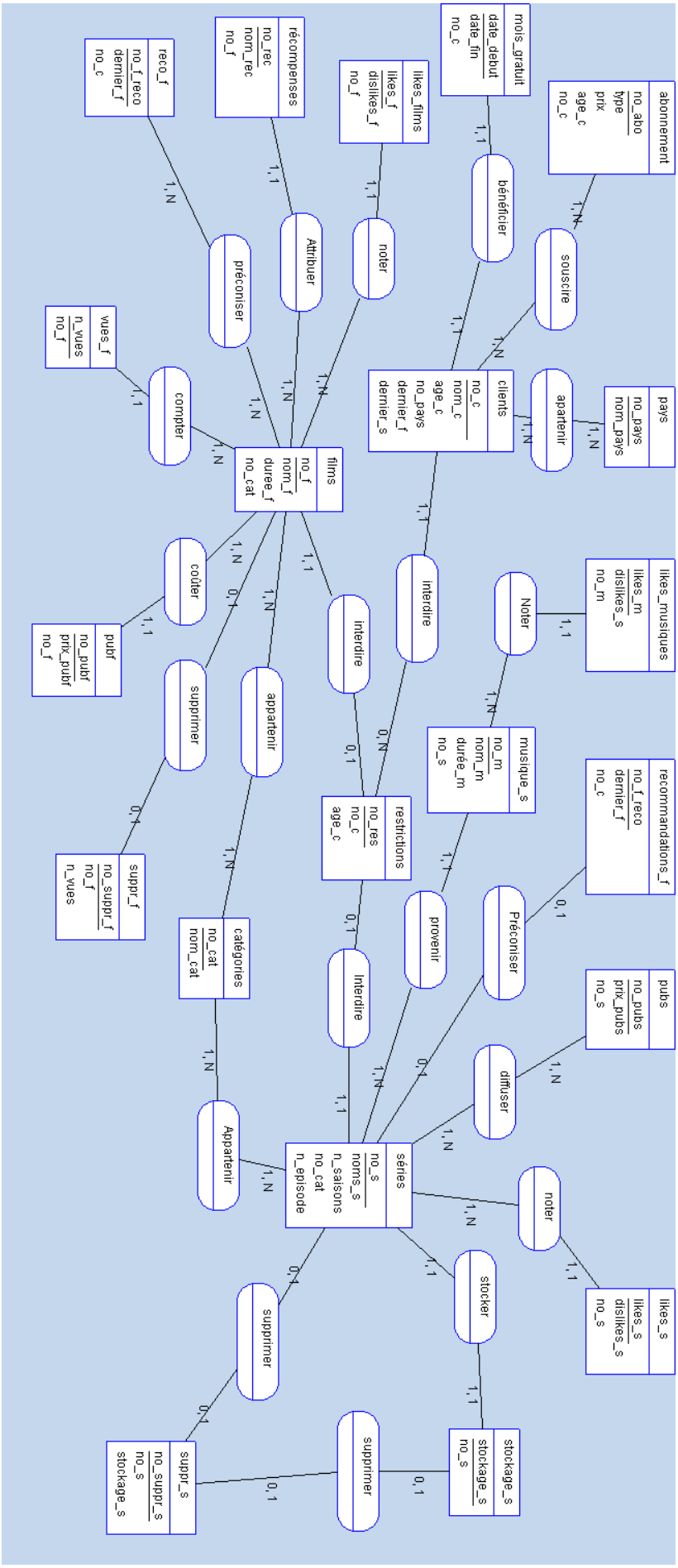
Table	Clés	Description
series	**no_s** nom_s n_saisons no_cat n_episode	**identifiant de série** nom de la série nombre de saisons numéro de catégorie nombre d'ep par saisons
films	**no_f** nom_f duree_f no_cat	**identifiant de film** nom du film durée du film en minutes numéro de catégorie
clients	**no_c** nom_c age_c no_pays dernier_f dernier_s	**identifiant de client** nom du client âge du client identifiant du pays dernier film visionné dernière série visionnée
restrictions	**no_res** no_c age_c	**identifiant de restriction** identifiant du client âge du client
pubf	**no_pubf** prix_pubf no_f	**identifiant de la pub (f)** prix de la pub (f) identifiant du film
pubs	**no_pubs** prix_pubs no_s	**identifiant de la pub (s)** prix de la pub (s) identifiant de la série
suppr_f	**no_suppr_f** no_f n_vues	**identifiant de suppression** identifiant du film nombre de vues
vues_f	**n_vues** no_f	**nombre de vues** identifiant du film
stockage_s	**stockage_s** no_s	**poid en giga série** identifiant de série
suppr_s	stockage_s	poid en giga série

	no_s **no_suppr_s**	identifiant de la série **identifiant de suppression**
reco_f	**no_f_reco** dernier_f no_c	**film recommandé** dernier film visionné identifiant du client
reco_s	**no_s_reco** dernier_s no_c	**série recommandé** dernière série visionné identifiant du client
categories	**no_cat** nom_cat	**identifiant catégorie** nom catégorie
musiques_s	**no_m** nom_m duree_m no_s	**identifiant musique** nom de la musique durée de la musique identifiant de la série
mois_gratuit	**date_debut** date_fin no_c	**date de début** date de fin identifiant client
abonnements	**no_abo** type prix no_c age_c	**identifiant abonnement** type d'abonnement prix de l'abonnement identifiant du client âge du client
pays	**no_pays** nom_pays	**identifiant du pays** nom du pays
recompenses	**no_rec** nom_rec no_f	**identifiant récompense** nom de la récompense identifiant du film
likes_f	**likes_f** dislikes_f no_f	**nombres de likes (f)** nombre de dislikes (f) identifiant du film
likes_s	**likes_s** dislikes_s no_s	**nombre de likes (s)** nombre de dislikes (s) identifiant de la série
likes_m	**likes_m** disklikes_m no_m	**nombre de likes (m)** nombre de dislikes (m) identifiant de la musique

Conceptual Data Model



Logical Data Model



Part 2 : Queries

Basic queries

1. Afficher toutes les caractéristiques des films

SELECT * FROM films;

no_f	nom_f	duree_f	no_cat
1	Loris et la chocolaterie	120	5
2	The Godefather	150	4
3	22B Rue Ledru Rolin	5	3
4	La Ptite Annick	140	2
5	Retour vers le futal	120	6
6	Juraciste Park	120	6
7	OSS 117 3 : Alerte rouge en Afrique noire	110	2
8	Star Wars III : La revenge des SIF	140	1
9	2 Gaulle	120	5
10	Le diner de cons	90	2

2. Afficher toutes les caractéristiques des séries

SELECT * FROM series;

no_s	nom_s	n_saisons	no_cat	n_episode
1	Game of Holes (c'est du golf)	8	1	10
2	How I Met Your Step-daughter	9	2	15
3	The Sweating Bed	7	3	12
4	House of Darts	6	4	10
5	Being Bad	7	1	13
6	The Large Bang Theory	12	2	16
7	Twins Peak	3	4	8
8	European Horror Story	5	3	10
9	Camelote	30	2	25
10	Cherloque	4	4	3

3. Afficher le numéro et le nom des clients classés par âge

SELECT no_c, nom_, age_c FROM clients GROUP BY age_c;

no_c	nom_c	age_c
4	Donald Trump	2
7	Carlos Ghosn	4
3	Victor Pion	5
10	Laura T	9
5	Pedro Pierdo el Telephono	18
8	Luc Marty	35
9	Maurice Johnson	50
1	Jean-Jean	56
6	Loris à Seattle avec Shalom peps	69
2	Thierry Morvan	89

4. Afficher la durée en minutes des musiques des séries

SELECT nom_m, (duree_m/60) AS "duree_M_min" FROM musiques_s;

nom_m	duree_M_min
The Putt	3.3333
HIMYSD Main Theme	2.1667
Zom-bi	4.0000
The Sting	3.0000
Meth Lab	6.0667
Walter White Theme	4.4167
Fire Walk With Me	2.7333
Jumpscare	3.7333
Alexandre Astier Remix	9.6333
The Final Problem	5.5500

5. Afficher le nombre total d'épisodes d'une séries toutes saisons confondues

SELECT no_s, (n_saisons*n_episode) AS "episod_tot" FROM series;

no_s	"episod_tot"
1	80
2	135
3	84
4	60
5	91
6	192
7	24
8	50
9	750
10	12

WHERE Clause Queries

1. Afficher les musiques qui ont eu au moins 20 000 likes et moins de 500 dislikes

SELECT * FROM likes_m WHERE likes_m > 20000 AND dislikes_m < 500;

no_m	likes_m	dislikes_m
10	25896	464
2	45698	324

2. Afficher les films supprimés qui avaient entre 50 et 100 vues

SELECT * FROM suppr_f WHERE n_vues BETWEEN 50 AND 100;

no_suppr_f	no_f	n_vues
1	1	69

3. Afficher les films recommandés pour un client qui a déjà regardé un des films 6,7,8,9,10

SELECT * FROM reco_f WHERE dernier_f > 5;

no_c	dernier_f	no_f_reco
9	6	1
1	7	3
8	10	5
10	9	8
6	8	9

4. Afficher les films qui n'ont pas de vues

SELECT * FROM vues_f WHERE n_vues =0;

n_vues	no_f
0	3

5. Afficher les abonnés qui ont moins de 12 ans et qui habitent dans le pays numéro 1

SELECT * FROM clients WHERE age_c <12 AND no_pays = 1;

no_c	nom_c	age_c	no_pays	dernier_f	dernier_s
3	Victor Pion	5	1	1	5

ORDER BY Queries

1. Classer les films par stockage dans l'ordre croissant

SELECT * FROM stockage_s ORDER BY stockage_s ASC;

no_s	stockage_s ▲ 1
1	57
9	68
3	97
7	125
2	180
5	197
10	220
8	340
4	380
6	400

2. Ordonner les films par catégories

SELECT * FROM films ORDER BY no_cat ASC;

no_f	nom_f	duree_f	no_cat ▲ 1
8	Star Wars III : La revenge des SIF	140	1
4	La Ptite Annick	140	2
7	OSS 117 3 : Alerte rouge en Afrique noire	110	2
10	Le diner de cons	90	2
3	22B Rue Ledru Rolin	5	3
2	The Godefather	150	4
1	Loris et la chocolaterie	120	5
9	2 Gaulle	120	5
5	Retour vers le futal	120	6
6	Juraciste Park	120	6

3. Ordonner les films par nombre de likes, ordre décroissant

SELECT * FROM likes_f ORDER BY likes_f DESC;

no_f	likes_f ▼ 1	dislikes_f
5	98652	526
2	47895	6058
8	36647	1548
7	12978	9875
6	8426	326
1	5970	302
4	1489	30
3	779	23
10	354	65
9	0	36948

4. Ordonner les publicités films par le prix de publicité, ordre décroissant

SELECT * FROM pubf ORDER BY prix_pubf DESC;

no_pubf	prix_pubf ▼ 1	no_f
10	100820	10
7	78954	7
2	58000	2
8	47698	8
1	34000	1
6	28700	6
5	10500	5
4	6080	4
9	4500	9
3	740	3

5. Afficher les types d'abonnement par prix, ordre croissant

SELECT * FROM abonnements ORDER BY prix ASC;

no_abo	no_c	age_c	prix ▲ 1	type
3	3	5	5	Enfant
4	4	2	5	Enfant
7	7	4	5	Enfant
10	10	9	5	Enfant
2	2	89	8	Senior
6	6	69	8	Senior
1	1	56	10	Normal
5	5	18	10	Normal
8	8	35	10	Normal
9	9	50	10	Normal

Multi-Table Queries

1. Afficher les séries dont le nom commence par "THE" et donner leur cout en publicité

SELECT * FROM series, pubs WHERE series.no_s = pubs.no_s AND nom_s LIKE "THE%";

no_s	nom_s	n_saisons	no_cat	n_episode	no_pubs	prix_pubs	no_s
3	The Sweating Bed	7	3	12	3	45360	3
6	The Large Bang Theory	12	2	16	6	38705	6

2. Afficher les films par catégories pour les films d'une durée supérieure ou égale à 120 mins

SELECT nom_cat, nom_f FROM films, categories CAT WHERE films.no_cat = CAT.no_cat AND duree_f >=120;

nom_cat	nom_f
Famille	Loris et la chocolaterie
Drame	The Godefather
Humour	La Ptite Annick
Aventure	Retour vers le futal
Aventure	Juraciste Park
Action	Star Wars III : La revenge des SIF
Famille	2 Gaulle

3. Afficher pour les séries de catégorie 1 les séries avec leur séries recommandées

SELECT * FROM series, reco_s WHERE series.no_s = reco_s.dernier_s AND no_cat = 2;

no_s	nom_s	n_saisons	no_cat	n_episode	no_c	dernier_s	no_s_reco
6	The Large Bang Theory	12	2	16	5	6	2
9	Camelote	30	2	25	2	9	5
2	How I Met Your Step-daughter	9	2	15	7	2	6

4. Afficher les noms des clients qui ont bénéficié du mois gratuit en 2015

SELECT nom_c FROM clients, mois_gratuit WHERE clients.no_c = mois_gratuit.no_c AND date_debut LIKE "2015%";

nom_c
Victor Pion

5. Afficher le nom et l'abonnement des clients.

SELECT nom_c, type FROM clients, abonnements WHERE clients.no_c = abonnements.no_c;

nom_c	type
Jean-Jean	Normal
Thierry Morvan	Senior
Victor Pion	Enfant
Donald Trump	Enfant
Pedro Pierdo el Telephono	Normal
Loris à Seattle avec Shalom peps	Senior
Carlos Ghosn	Enfant
Luc Marty	Normal
Maurice Johnson	Normal
Laura T	Enfant

Queries with Numeric expressions and functions

1. Afficher l'âge moyen des abonnés

SELECT AVG(age_c) AS «âge_moyen_abonnés» FROM clients ;

«âge_moyen_abonnés»
33.7000

2. Afficher le nombre de séries disponibles sur la plateforme

SELECT COUNT(no_s) AS «nombre_de_séries_disponibles» FROM series ;

«nombre_de_séries_disponibles»
10

3. Afficher le nombre de vues total sur les films

SELECT SUM(n_vues) AS «nombre_vues_total_films» FROM vues_f ;

«nombre_vues_total_films»
28354272

4. Afficher la moyenne de likes, sur les musiques, qui devrait augmenter de 20% à la suite de la sortie d'un nouveau blockbuster dont la bande originale a beaucoup de succès

SELECT AVG(likes_m) * 1.2 AS "nouvelle_moyenne_likes_m" FROM likes_m ;

nouvelle_moyenne_likes_m
12813.20000

5. Afficher le numéro du client ayant bénéficier du mois gratuit en premier parmi les abonnés actuels

SELECT MIN (date_debut), no_c FROM mois_gratuit;

MIN(date_debut)	no_c
2014-09-15 00:00:00	1

Group By Queries

1. Afficher les types d'abonnement selon leur prix

SELECT prix, type FROM abonnements GROUP BY prix;

prix	type
5	Enfant
8	Senior
10	Normal

2. Afficher le nombre de séries selon leur nombre de saisons

SELECT COUNT(no_s), n_saisons FROM series GROUP BY n_saisons;

COUNT(no_s)	n_saisons
1	3
1	4
1	5
1	6
2	7
1	8
1	9
1	12
1	30

3. Afficher le nombre de films selon leur nombre de likes

SELECT likes_f, films.no_f, nom_f FROM likes_f, films WHERE films.no_f = likes_f.no_f GROUP BY likes_f;

likes_f	no_f	nom_f
0	9	2 Gaulle
354	10	Le diner de cons
779	3	22B Rue Ledru Rolin
1489	4	La Ptite Annick
5970	1	Loris et la chocolaterie
8426	6	Juraciste Park
12978	7	OSS 117 3 : Alerte rouge en Afrique noire
36647	8	Star Wars III : La revenge des SIF
47895	2	The Godefather
98652	5	Retour vers le futal

4. Afficher le nombre de clients par pays

SELECT COUNT(no_c), clients.no_pays, nom_pays FROM clients, pays WHERE clients.no_pays = pays.no_pays GROUP BY no_pays;

COUNT(no_c)	no_pays	nom_pays
3	1	France
3	2	USA
1	3	Angleterre
1	4	Japon
2	5	Australie

5. Afficher le nombre de films par durée

SELECT COUNT(no_f), duree_f FROM films GROUP BY duree_f;

COUNT(no_f)	duree_f
1	5
1	90
1	110
4	120
2	140
1	150

Nested Queries

1. Afficher la série dont la durée de la musique associée est la plus longue

SELECT series.no_s, nom_s, duree_m FROM series, musiques_s WHERE musiques_s.no_s = series.no_s AND musiques_s.duree_m = (SELECT MAX(duree_m) FROM musiques_s);

no_s	nom_s	duree_m
9	Camelote	578

2. Afficher les films avec plus de vues que le film numéro 4

SELECT films.no_f, films.nom_f, vues_f.n_vues FROM films, vues_f WHERE films.no_f = vues_f.no_f AND n_vues > (SELECT n_vues FROM vues_f WHERE no_f =4) ;

no_f	nom_f	n_vues
2	The Godefather	4806809
7	OSS 117 3 : Alerte rouge en Afrique noire	23354780

3. Afficher le film associé à la publicité la plus chère

SELECT pubf.no_f, pubf.prix_pubf, films.nom_f FROM pubf, films WHERE pubf.no_f = films.no_f AND pubf.prix_pubf = (SELECT MAX(prix_pubf) FROM pubf) ;

no_f	prix_pubf	nom_f
10	100820	Le diner de cons

4. Afficher le film le moins regardé

SELECT films.no_f, films.nom_f, vues_f.n_vues FROM films, vues_f WHERE vues_f.no_f = films.no_f AND vues_f.n_vues = (SELECT MIN(n_vues) FROM vues_f) ;

no_f	nom_f	n_vues
3	22B Rue Ledru Rolin	0

5. **Afficher la série la plus appréciée**

SELECT series.no_s, nom_s, likes_s.likes_s FROM series, likes_s WHERE likes_s.no_s = series.no_s AND likes_s.likes_s = (SELECT MAX(likes_s) FROM likes_s);

no_s	nom_s	likes_s
1	Game of Holes (c'est du golf)	4598798

Part 3 : mongoDB

Pour cette partie, nous avons choisi les tables clients, séries et musiques, qui à nos yeux sont représentatives de la base de données. Toutefois, afin d'obtenir des résultats plus variés, nous avons fusionné la table « likes_m » avec musiques, pour avoir le nombre de likes et dislikes pour chaque musique.

On limite notre étude à 25 requêtes MongoDB par table, en utilisant de façon équitable des requêtes : basics, where, group by, with numeric expressions and functions et order by.

Séries

Le streaming de séries est l'une des activités majeures de HQ-Streaming. Le but de cette analyse est donc de permettre aux dirigeants de la plateforme de streaming d'orienter leurs choix en ce qui concerne le panel de séries mis à disposition des abonnés HQ-Streaming. Nous allons donc analyser les caractéristiques des séries de la plateforme. Nous allons faire une analyse générale, spécifique, par catégorie puis par nombre de saisons et d'épisodes.

Caractéristiques globales

1. Afficher les caractéristiques des séries (Basic)

db.series.find({}).pretty()

{"_id" : ObjectId("5e8f0d9f9d995da935a1e72f"),

"no_s" : 1,

"nom_s" : "Game of Holes (c'est du golf)",

"n_saisons" : 8,

"no_cat" : 1,

"n_episode" : 10

}

{

"_id" : ObjectId("5e8f0d9f9d995da935a1e730"),

"no_s" : 2,

"nom_s" : "How I Met Your Step-daughter",

"n_saisons" : 9,

"no_cat" : 2,

```
        "n_episode" : 15
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e731"),
        "no_s" : 3,
        "nom_s" : "The Sweating Bed",
        "n_saisons" : 7,
        "no_cat" : 3,
        "n_episode" : 12
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e732"),
        "no_s" : 4,
        "nom_s" : "House of Darts",
        "n_saisons" : 6,
        "no_cat" : 4,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e733"),
        "no_s" : 5,
        "nom_s" : "Being Bad",
        "n_saisons" : 7,
        "no_cat" : 1,
        "n_episode" : 13
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e734"),
        "no_s" : 6,
        "nom_s" : "The Large Bang Theory",
        "n_saisons" : 12,
        "no_cat" : 2,
        "n_episode" : 16
}
```

```
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e735"),
        "no_s" : 7,
        "nom_s" : "Twins Peak",
        "n_saisons" : 3,
        "no_cat" : 4,
        "n_episode" : 8
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e736"),
        "no_s" : 8,
        "nom_s" : "European Horror Story",
        "n_saisons" : 5,
        "no_cat" : 3,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e737"),
        "no_s" : 9,
        "nom_s" : "Camelote",
        "n_saisons" : 30,
        "no_cat" : 2,
        "n_episode" : 25
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e738"),
        "no_s" : 10,
        "nom_s" : "Cherloque",
        "n_saisons" : 4,
        "no_cat" : 4,
        "n_episode" : 3
}
```

2. Afficher le nom des séries (Basic)

db.series.distinct("nom_s")

```
[
        "Game of Holes (c'est du golf)",
        "How I Met Your Step-daughter",
        "The Sweating Bed",
        "House of Darts",
        "Being Bad",
        "The Large Bang Theory",
        "Twins Peak",
        "European Horror Story",
        "Camelote",
        "Cherloque"
]
```

3. Compter le nombre de séries (Numeric expressions and functions)

db.series.count()

10

Caractéristiques spécifiques

1. Afficher les caractéristiques de la série « Camelote » (Where)

db.series.find({nom_s:"Camelote"}).pretty()

```
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e737"),
        "no_s" : 9,
        "nom_s" : "Camelote",
        "n_saisons" : 30,
        "no_cat" : 2,
        "n_episode" : 25
}
```

2. Afficher les caractéristiques de la série numéro 4 (Where)

db.series.find({no_s:4}).pretty()


```
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e732"),
        "no_s" : 4,
        "nom_s" : "House of Darts",
        "n_saisons" : 6,
        "no_cat" : 4,
        "n_episode" : 10
}
```


3. Trier les films dans l'ordre anti-alphabétique (Order by)

db.series.find().sort({"nom_s":-1}).pretty()


```
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e735"),
        "no_s" : 7,
        "nom_s" : "Twins Peak",
        "n_saisons" : 3,
        "no_cat" : 4,
        "n_episode" : 8
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e731"),
        "no_s" : 3,
        "nom_s" : "The Sweating Bed",
        "n_saisons" : 7,
        "no_cat" : 3,
        "n_episode" : 12
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e734"),
        "no_s" : 6,
        "nom_s" : "The Large Bang Theory",
        "n_saisons" : 12,
```

```
        "no_cat" : 2,
        "n_episode" : 16
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e730"),
        "no_s" : 2,
        "nom_s" : "How I Met Your Step-daughter",
        "n_saisons" : 9,
        "no_cat" : 2,
        "n_episode" : 15
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e732"),
        "no_s" : 4,
        "nom_s" : "House of Darts",
        "n_saisons" : 6,
        "no_cat" : 4,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e72f"),
        "no_s" : 1,
        "nom_s" : "Game of Holes (c\\'est du golf)",
        "n_saisons" : 8,
        "no_cat" : 1,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e736"),
        "no_s" : 8,
        "nom_s" : "European Horror Story",
        "n_saisons" : 5,
        "no_cat" : 3,
        "n_episode" : 10
```

```
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e738"),
        "no_s" : 10,
        "nom_s" : "Cherloque",
        "n_saisons" : 4,
        "no_cat" : 4,
        "n_episode" : 3
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e737"),
        "no_s" : 9,
        "nom_s" : "Camelote",
        "n_saisons" : 30,
        "no_cat" : 2,
        "n_episode" : 25
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e733"),
        "no_s" : 5,
        "nom_s" : "Being Bad",
        "n_saisons" : 7,
        "no_cat" : 1,
        "n_episode" : 13
}
```


Caractéristiques selon le nombre d’épisodes

1. Afficher les caractéristiques des séries qui sont dans la catégorie 3 et qui ont plus de 10 épisodes (Where)

db.series.find({"$and" : [{"no_cat":3},{"n_episode":{$gt:10}}]}).pretty()


```
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e731"),
        "no_s" : 3,
```

```
        "nom_s" : "The Sweating Bed",
        "n_saisons" : 7,
        "no_cat" : 3,
        "n_episode" : 12
}
```


2. Trier les séries selon leur catégorie et leur nombre d’épisodes (order by)

db.series.find().sort({"no_cat":1, "n_episode":1}).pretty()


```
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e72f"),
        "no_s" : 1,
        "nom_s" : "Game of Holes (c\\'est du golf)",
        "n_saisons" : 8,
        "no_cat" : 1,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e733"),
        "no_s" : 5,
        "nom_s" : "Being Bad",
        "n_saisons" : 7,
        "no_cat" : 1,
        "n_episode" : 13
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e730"),
        "no_s" : 2,
        "nom_s" : "How I Met Your Step-daughter",
        "n_saisons" : 9,
        "no_cat" : 2,
        "n_episode" : 15
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e734"),
```

```
        "no_s" : 6,
        "nom_s" : "The Large Bang Theory",
        "n_saisons" : 12,
        "no_cat" : 2,
        "n_episode" : 16
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e737"),
        "no_s" : 9,
        "nom_s" : "Camelote",
        "n_saisons" : 30,
        "no_cat" : 2,
        "n_episode" : 25
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e736"),
        "no_s" : 8,
        "nom_s" : "European Horror Story",
        "n_saisons" : 5,
        "no_cat" : 3,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e731"),
        "no_s" : 3,
        "nom_s" : "The Sweating Bed",
        "n_saisons" : 7,
        "no_cat" : 3,
        "n_episode" : 12
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e738"),
        "no_s" : 10,
        "nom_s" : "Cherloque",
```

```
        "n_saisons" : 4,
        "no_cat" : 4,
        "n_episode" : 3
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e735"),
        "no_s" : 7,
        "nom_s" : "Twins Peak",
        "n_saisons" : 3,
        "no_cat" : 4,
        "n_episode" : 8
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e732"),
        "no_s" : 4,
        "nom_s" : "House of Darts",
        "n_saisons" : 6,
        "no_cat" : 4,
        "n_episode" : 10
}
```


3. Afficher le nombre minimum d'épisodes sur une série toutes séries confondues (numeric expressions and functions)

db.series.aggregate([{$group:{"_id":null, min_episode:{$min:"$n_episode"}}}])


```
{ "_id" : null, "min_episode" : 3 }
```


4. Afficher le nombre maximum d'épisodes sur une série toutes séries confondues (numeric expressions and functions)

db.series.aggregate([{$group:{"_id":null, max_episode:{$max:"$n_episode"}}}])


```
{ "_id" : null, "max_episode" : 25 }
```

5. Afficher le nombre de séries par nombre d'épisodes (group by)

db.series.aggregate([{$group:{"_id":"$n_episode", total:{$sum:1}}}])

{ "_id" : 3, "total" : 1 }

{ "_id" : 8, "total" : 1 }

{ "_id" : 16, "total" : 1 }

{ "_id" : 13, "total" : 1 }

{ "_id" : 12, "total" : 1 }

{ "_id" : 25, "total" : 1 }

{ "_id" : 15, "total" : 1 }

{ "_id" : 10, "total" : 3 }

6. Afficher le numéro des séries qui ont moins de 10 épisodes (group by)

db.series.aggregate([{$match:{"n_episode":{$lt:10}}},{$group:{_id:"$n_episode",total:{$sum:1}}}])

{ "_id" : 3, "total" : 1 }

Caractéristiques selon les catégories

1. Afficher les numéros de catégorie des séries (basic)

db.series.distinct("no_cat")

[1, 2, 3, 4]

2. Afficher les séries de la catégorie 1 (basic)

db.series.distinct("nom_s", {"no_cat":1})

["Game of Holes (c'est du golf)", "Being Bad"]

3. Afficher les caractéristiques des séries de la catégorie 3 ou 4 (where)

db.series.find({"$or" : [{"no_cat":3},{"no_cat":4}]}).pretty()

{

"_id" : ObjectId("5e8f0d9f9d995da935a1e731"),

"no_s" : 3,

"nom_s" : "The Sweating Bed",

```
        "n_saisons" : 7,
        "no_cat" : 3,
        "n_episode" : 12
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e732"),
        "no_s" : 4,
        "nom_s" : "House of Darts",
        "n_saisons" : 6,
        "no_cat" : 4,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e735"),
        "no_s" : 7,
        "nom_s" : "Twins Peak",
        "n_saisons" : 3,
        "no_cat" : 4,
        "n_episode" : 8
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e736"),
        "no_s" : 8,
        "nom_s" : "European Horror Story",
        "n_saisons" : 5,
        "no_cat" : 3,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e738"),
        "no_s" : 10,
        "nom_s" : "Cherloque",
        "n_saisons" : 4,
        "no_cat" : 4,
```

"n_episode" : 3}

4. Trier les films selon leur numéro de catégorie dans l'ordre décroissant (order by)

db.series.find().sort({"no_cat":-1}).pretty()

{

"_id" : ObjectId("5e8f0d9f9d995da935a1e732"),

"no_s" : 4,

"nom_s" : "House of Darts",

"n_saisons" : 6,

"no_cat" : 4,

"n_episode" : 10

}

{

"_id" : ObjectId("5e8f0d9f9d995da935a1e735"),

"no_s" : 7,

"nom_s" : "Twins Peak",

"n_saisons" : 3,

"no_cat" : 4,

"n_episode" : 8

}

{

"_id" : ObjectId("5e8f0d9f9d995da935a1e738"),

"no_s" : 10,

"nom_s" : "Cherloque",

"n_saisons" : 4,

"no_cat" : 4,

"n_episode" : 3

}

{

"_id" : ObjectId("5e8f0d9f9d995da935a1e731"),

"no_s" : 3,

"nom_s" : "The Sweating Bed",

"n_saisons" : 7,

"no_cat" : 3,

```
        "n_episode" : 12
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e736"),
        "no_s" : 8,
        "nom_s" : "European Horror Story",
        "n_saisons" : 5,
        "no_cat" : 3,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e730"),
        "no_s" : 2,
        "nom_s" : "How I Met Your Step-daughter",
        "n_saisons" : 9,
        "no_cat" : 2,
        "n_episode" : 15
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e734"),
        "no_s" : 6,
        "nom_s" : "The Large Bang Theory",
        "n_saisons" : 12,
        "no_cat" : 2,
        "n_episode" : 16
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e737"),
        "no_s" : 9,
        "nom_s" : "Camelote",
        "n_saisons" : 30,
        "no_cat" : 2,
        "n_episode" : 25
}
```

{
"_id" : ObjectId("5e8f0d9f9d995da935a1e72f"),
"no_s" : 1,
"nom_s" : "Game of Holes (c\\'est du golf)",
"n_saisons" : 8,
"no_cat" : 1,
"n_episode" : 10
}
{
"_id" : ObjectId("5e8f0d9f9d995da935a1e733"),
"no_s" : 5,
"nom_s" : "Being Bad",
"n_saisons" : 7,
"no_cat" : 1,
"n_episode" : 13
}

5. Compter le nombre de séries par catégorie (numeric expressions and functions)

db.series.aggregate([{$group:{"_id":"$no_cat", total:{$sum:1}}}])

{ "_id" : 4, "total" : 3 }
{ "_id" : 3, "total" : 2 }
{ "_id" : 2, "total" : 3 }
{ "_id" : 1, "total" : 2 }

Caractéristiques selon le nombre de saisons

1. Afficher le nombre de saisons de la série « being bad » (basic)

db.series.distinct("n_saisons", {"nom_s":"Being Bad"})

[7]

2. Afficher les caractéristiques des séries avec plus de 10 saisons (where)

db.series.find({n_saisons : {$gt : 10}}).pretty()

{
"_id" : ObjectId("5e8f0d9f9d995da935a1e734"),

```
        "no_s" : 6,
        "nom_s" : "The Large Bang Theory",
        "n_saisons" : 12,
        "no_cat" : 2,
        "n_episode" : 16
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e737"),
        "no_s" : 9,
        "nom_s" : "Camelote",
        "n_saisons" : 30,
        "no_cat" : 2,
        "n_episode" : 25}
```


3. Trier les séries par nombre de saisons (order by)

db.series.find().sort({"n_saisons":1}).pretty()


```
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e735"),
        "no_s" : 7,
        "nom_s" : "Twins Peak",
        "n_saisons" : 3,
        "no_cat" : 4,
        "n_episode" : 8
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e738"),
        "no_s" : 10,
        "nom_s" : "Cherloque",
        "n_saisons" : 4,
        "no_cat" : 4,
        "n_episode" : 3
}
{
```

```
        "_id" : ObjectId("5e8f0d9f9d995da935a1e736"),
        "no_s" : 8,
        "nom_s" : "European Horror Story",
        "n_saisons" : 5,
        "no_cat" : 3,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e732"),
        "no_s" : 4,
        "nom_s" : "House of Darts",
        "n_saisons" : 6,
        "no_cat" : 4,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e731"),
        "no_s" : 3,
        "nom_s" : "The Sweating Bed",
        "n_saisons" : 7,
        "no_cat" : 3,
        "n_episode" : 12
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e733"),
        "no_s" : 5,
        "nom_s" : "Being Bad",
        "n_saisons" : 7,
        "no_cat" : 1,
        "n_episode" : 13
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e72f"),
        "no_s" : 1,
```

```
        "nom_s" : "Game of Holes (c\\'est du golf)",
        "n_saisons" : 8,
        "no_cat" : 1,
        "n_episode" : 10
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e730"),
        "no_s" : 2,
        "nom_s" : "How I Met Your Step-daughter",
        "n_saisons" : 9,
        "no_cat" : 2,
        "n_episode" : 15
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e734"),
        "no_s" : 6,
        "nom_s" : "The Large Bang Theory",
        "n_saisons" : 12,
        "no_cat" : 2,
        "n_episode" : 16
}
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e737"),
        "no_s" : 9,
        "nom_s" : "Camelote",
        "n_saisons" : 30,
        "no_cat" : 2,
        "n_episode" : 25
}
```


4. Trier les séries de moins de 5 saisons selon leur catégorie (order by)

db.series.aggregate([{"$match":{"n_saisons":{$lt:5}}},{"$sort":{"no_cat":1}}]).pretty()


```
{
        "_id" : ObjectId("5e8f0d9f9d995da935a1e735"),
```

"no_s" : 7,

"nom_s" : "Twins Peak",

"n_saisons" : 3,

"no_cat" : 4,

"n_episode" : 8

}

{

"_id" : ObjectId("5e8f0d9f9d995da935a1e738"),

"no_s" : 10,

"nom_s" : "Cherloque",

"n_saisons" : 4,

"no_cat" : 4,

"n_episode" : 3

}

5. Afficher le nombre moyen de saisons par catégorie (numeric expressions and functions)

db.series.aggregate([{$group:{"_id":"$no_cat", moyenne_saisons:{$avg:"$n_saisons"}}}])

{ "_id" : 4, "moyenne_saisons" : 4.333333333333333 }

{ "_id" : 3, "moyenne_saisons" : 6 }

{ "_id" : 2, "moyenne_saisons" : 17 }

{ "_id" : 1, "moyenne_saisons" : 7.5 }

6. Afficher le nombre de séries par nombre de saisons (group by)

db.series.aggregate([{$group:{"_id":"$n_saisons", total:{$sum:1}}}])

{ "_id" : 4, "total" : 1 }

{ "_id" : 5, "total" : 1 }

{ "_id" : 3, "total" : 1 }

{ "_id" : 30, "total" : 1 }

{ "_id" : 12, "total" : 1 }

{ "_id" : 6, "total" : 1 }

{ "_id" : 7, "total" : 2 }

{ "_id" : 9, "total" : 1 }

{ "_id" : 8, "total" : 1 }

7. Afficher le nombre minimum de saisons pour une série (group by)

db.series.aggregate([{$group:{"_id":null, min_saisons:{$min:"$n_saisons"}}}])

{ "_id" : null, "min_saisons" : 3 }

8. Afficher le nombre maximum de saisons pour une série (group by)

db.series.aggregate([{$group:{"_id":null, max_saisons:{$max:"$n_saisons"}}}])

{ "_id" : null, "max_saisons" : 30 }

Bilan

À la suite de l'analyse des séries présentes sur la plateforme hq streaming, les dirigeants ont désormais en leur possession les caractéristiques des séries dans leur ensemble, par catégorie ainsi que par nombre d'épisodes et de saisons. Ils peuvent ainsi remarquer que les séries que peuvent visionner les abonnés hq streaming sont assez diversifiées puisque les quatre catégories de série disponibles représentent chacune entre 20 et 30% des séries disponibles. Les formats des séries sont aussi divers puisqu'on peut trouver sur la plateforme de streaming des séries qui ont entre 4 et 30 saisons et entre 3 et 25 épisodes. Ils peuvent donc se satisfaire de ce constat et persévérer dans ce sens en ce qui concerne le fond et la forme des séries. Cependant, ils ont certainement remarqué le faible nombre de séries disponibles (10) et vont sûrement chercher à proposer plus de séries aux abonnés au fur et à mesure de leur développement en vue de lutter contre de nouveaux concurrents.

Musiques

Les musiques ont été mises à disposition sur la plateforme car HQ Streaming a obtenu leurs droits de diffusion en même temps que le droit d'utilisation des séries correspondantes. L'idée est désormais de voir s'il est utile de garder les musiques sur la plateforme, ou d'en supprimer certaines. Il y a une limite rencontrée à notre analyse : on ne dispose pas du nombre d'écoutes de chaque musique. On va donc faire nos sélections arbitrairement selon le nombre de likes et dislikes et la durée de la musique.

Objectifs :

•Chercher la musique qui a le meilleur rapport temps-likes-dislikes et la soumettre pour l'utiliser comme la musique d'entrée sur la plateforme pour les clients.

•Éliminer les musiques de la plateforme, selon :

- celles qui ont un mauvais rapport likes-dislikes.

-Définir s'il y a une corrélation entre la durée des musiques et leur nombre de likes/dislikes.

•Voir si le nom « THE » qui est le mot le plus présent sur la plateforme est un biais cognitif ou non.

Visualisation globale des données

1. (Basic) Afficher toutes les informations de la table musique des séries de la plateforme

db.musiques.find({}).pretty()


```
{
	"_id" : ObjectId("5e902bb5bed3f3027d440849"),
	"no_m" : 1,
	"nom_m" : "The Putt",
	"duree_m" : 200,
	"no_s" : 1,
	"likes_m" : 123,
	"dislikes_m" : 5
}
{
	"_id" : ObjectId("5e902bb5bed3f3027d44084a"),
	"no_m" : 2,
	"nom_m" : "HIMYSD Main Theme",
	"duree_m" : 130,
	"no_s" : 2,
	"likes_m" : 45698,
	"dislikes_m" : 324
}
{
	"_id" : ObjectId("5e902bb5bed3f3027d44084b"),
	"no_m" : 3,
	"nom_m" : "Zom-bi",
	"duree_m" : 240,
	"no_s" : 3,
	"likes_m" : 4002,
	"dislikes_m" : 6587
}
{
	"_id" : ObjectId("5e902bb5bed3f3027d44084c"),
	"no_m" : 4,
```

```
        "nom_m" : "The Sting",
        "duree_m" : 180,
        "no_s" : 4,
        "likes_m" : 23,
        "dislikes_m" : 0
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084d"),
        "no_m" : 5,
        "nom_m" : "Meth Lab",
        "duree_m" : 364,
        "no_s" : 5,
        "likes_m" : 7896,
        "dislikes_m" : 632
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084e"),
        "no_m" : 6,
        "nom_m" : "Walter White Theme",
        "duree_m" : 265,
        "no_s" : 6,
        "likes_m" : 4567,
        "dislikes_m" : 3658
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084f"),
        "no_m" : 7,
        "nom_m" : "Fire Walk With Me",
        "duree_m" : 164,
        "no_s" : 7,
        "likes_m" : 3800,
        "dislikes_m" : 987
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440850"),
        "no_m" : 8,
        "nom_m" : "Jumpscare",
        "duree_m" : 224,
        "no_s" : 8,
        "likes_m" : 0,
        "dislikes_m" : 9886
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440851"),
        "no_m" : 9,
        "nom_m" : "Alexandre Astier Remix",
        "duree_m" : 578,
        "no_s" : 9,
        "likes_m" : 7894,
        "dislikes_m" : 5896
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440852"),
        "no_m" : 10,
        "nom_m" : "The Final Problem",
        "duree_m" : 333,
        "no_s" : 10,
        "likes_m" : 25896,
        "dislikes_m" : 464
}
```

2. (with numeric expressions and functions) Compter le nombre de musiques

db.musiques.count( )

Output : 10

3. (Order by) trier les musiques par durée.

db.musiques.find( ).sort({'duree_m' : 1}).pretty( )


```
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084a"),
        "no_m" : 2,
        "nom_m" : "HIMYSD Main Theme",
        "duree_m" : 130,
        "no_s" : 2,
        "likes_m" : 45698,
        "dislikes_m" : 324
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084f"),
        "no_m" : 7,
        "nom_m" : "Fire Walk With Me",
        "duree_m" : 164,
        "no_s" : 7,
        "likes_m" : 3800,
        "dislikes_m" : 987
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084c"),
        "no_m" : 4,
        "nom_m" : "The Sting",
        "duree_m" : 180,
        "no_s" : 4,
        "likes_m" : 23,
        "dislikes_m" : 0
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440849"),
        "no_m" : 1,
        "nom_m" : "The Putt",
        "duree_m" : 200,
        "no_s" : 1,
        "likes_m" : 123,
```

```
        "dislikes_m" : 5
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440850"),
        "no_m" : 8,
        "nom_m" : "Jumpscare",
        "duree_m" : 224,
        "no_s" : 8,
        "likes_m" : 0,
        "dislikes_m" : 9886
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084b"),
        "no_m" : 3,
        "nom_m" : "Zom-bi",
        "duree_m" : 240,
        "no_s" : 3,
        "likes_m" : 4002,
        "dislikes_m" : 6587
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084e"),
        "no_m" : 6,
        "nom_m" : "Walter White Theme",
        "duree_m" : 265,
        "no_s" : 6,
        "likes_m" : 4567,
        "dislikes_m" : 3658
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440852"),
        "no_m" : 10,
        "nom_m" : "The Final Problem",
        "duree_m" : 333,
        "no_s" : 10,
        "likes_m" : 25896,
        "dislikes_m" : 464
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084d"),
        "no_m" : 5,
```

```
        "nom_m" : "Meth Lab",
        "duree_m" : 364,
        "no_s" : 5,
        "likes_m" : 7896,
        "dislikes_m" : 632
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440851"),
        "no_m" : 9,
        "nom_m" : "Alexandre Astier Remix",
        "duree_m" : 578,
        "no_s" : 9,
        "likes_m" : 7894,
        "dislikes_m" : 5896
}
```


4. (Order by) Lister la table en inversant l'ordre des numéros des musiques

db.musiques.find( ).sort({'no_m' : -1}).pretty( )


```
{
        "_id" : ObjectId("5e902bb5bed3f3027d440852"),
        "no_m" : 10,
        "nom_m" : "The Final Problem",
        "duree_m" : 333,
        "no_s" : 10,
        "likes_m" : 25896,
        "dislikes_m" : 464
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440851"),
        "no_m" : 9,
        "nom_m" : "Alexandre Astier Remix",
        "duree_m" : 578,
        "no_s" : 9,
        "likes_m" : 7894,
        "dislikes_m" : 5896
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440850"),
        "no_m" : 8,
```

```
        "nom_m" : "Jumpscare",
        "duree_m" : 224,
        "no_s" : 8,
        "likes_m" : 0,
        "dislikes_m" : 9886
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084f"),
        "no_m" : 7,
        "nom_m" : "Fire Walk With Me",
        "duree_m" : 164,
        "no_s" : 7,
        "likes_m" : 3800,
        "dislikes_m" : 987
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084e"),
        "no_m" : 6,
        "nom_m" : "Walter White Theme",
        "duree_m" : 265,
        "no_s" : 6,
        "likes_m" : 4567,
        "dislikes_m" : 3658
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084d"),
        "no_m" : 5,
        "nom_m" : "Meth Lab",
        "duree_m" : 364,
        "no_s" : 5,
        "likes_m" : 7896,
        "dislikes_m" : 632
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084c"),
        "no_m" : 4,
        "nom_m" : "The Sting",
        "duree_m" : 180,
        "no_s" : 4,
        "likes_m" : 23,
        "dislikes_m" : 0
```

```
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084b"),
        "no_m" : 3,
        "nom_m" : "Zom-bi",
        "duree_m" : 240,
        "no_s" : 3,
        "likes_m" : 4002,
        "dislikes_m" : 6587
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084a"),
        "no_m" : 2,
        "nom_m" : "HIMYSD Main Theme",
        "duree_m" : 130,
        "no_s" : 2,
        "likes_m" : 45698,
        "dislikes_m" : 324
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440849"),
        "no_m" : 1,
        "nom_m" : "The Putt",
        "duree_m" : 200,
        "no_s" : 1,
        "likes_m" : 123,
        "dislikes_m" : 5
}
```


5. (Order by) Trier la liste par les noms (ordre anti-alphabétique) et par durée de musique (ordre croissant).

db.musiques.find( ).sort({“nom_m” : 1, “duree_m” : 1}).pretty( )


```
{
        "_id" : ObjectId("5e902bb5bed3f3027d440851"),
        "no_m" : 9,
        "nom_m" : "Alexandre Astier Remix",
        "duree_m" : 578,
        "no_s" : 9,
        "likes_m" : 7894,
```

```
        "dislikes_m" : 5896
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084f"),
        "no_m" : 7,
        "nom_m" : "Fire Walk With Me",
        "duree_m" : 164,
        "no_s" : 7,
        "likes_m" : 3800,
        "dislikes_m" : 987
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084a"),
        "no_m" : 2,
        "nom_m" : "HIMYSD Main Theme",
        "duree_m" : 130,
        "no_s" : 2,
        "likes_m" : 45698,
        "dislikes_m" : 324
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440850"),
        "no_m" : 8,
        "nom_m" : "Jumpscare",
        "duree_m" : 224,
        "no_s" : 8,
        "likes_m" : 0,
        "dislikes_m" : 9886
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084d"),
        "no_m" : 5,
        "nom_m" : "Meth Lab",
        "duree_m" : 364,
        "no_s" : 5,
        "likes_m" : 7896,
        "dislikes_m" : 632
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440852"),
        "no_m" : 10,
```

```
        "nom_m" : "The Final Problem",
        "duree_m" : 333,
        "no_s" : 10,
        "likes_m" : 25896,
        "dislikes_m" : 464
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440849"),
        "no_m" : 1,
        "nom_m" : "The Putt",
        "duree_m" : 200,
        "no_s" : 1,
        "likes_m" : 123,
        "dislikes_m" : 5
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084c"),
        "no_m" : 4,
        "nom_m" : "The Sting",
        "duree_m" : 180,
        "no_s" : 4,
        "likes_m" : 23,
        "dislikes_m" : 0
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084e"),
        "no_m" : 6,
        "nom_m" : "Walter White Theme",
        "duree_m" : 265,
        "no_s" : 6,
        "likes_m" : 4567,
        "dislikes_m" : 3658
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084b"),
        "no_m" : 3,
        "nom_m" : "Zom-bi",
        "duree_m" : 240,
        "no_s" : 3,
        "likes_m" : 4002,
        "dislikes_m" : 6587}
```

6. (Order by) Trier les musiques par nom, ordre anti-alphabétique

db.musiques.find( ).sort({“nom_m” : -1}).pretty( )


```
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084b"),
        "no_m" : 3,
        "nom_m" : "Zom-bi",
        "duree_m" : 240,
        "no_s" : 3,
        "likes_m" : 4002,
        "dislikes_m" : 6587
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084e"),
        "no_m" : 6,
        "nom_m" : "Walter White Theme",
        "duree_m" : 265,
        "no_s" : 6,
        "likes_m" : 4567,
        "dislikes_m" : 3658
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084c"),
        "no_m" : 4,
        "nom_m" : "The Sting",
        "duree_m" : 180,
        "no_s" : 4,
        "likes_m" : 23,
        "dislikes_m" : 0
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440849"),
        "no_m" : 1,
        "nom_m" : "The Putt",
        "duree_m" : 200,
        "no_s" : 1,
        "likes_m" : 123,
        "dislikes_m" : 5
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440852"),
        "no_m" : 10,
```

```
        "nom_m" : "The Final Problem",
        "duree_m" : 333,
        "no_s" : 10,
        "likes_m" : 25896,
        "dislikes_m" : 464
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084d"),
        "no_m" : 5,
        "nom_m" : "Meth Lab",
        "duree_m" : 364,
        "no_s" : 5,
        "likes_m" : 7896,
        "dislikes_m" : 632
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440850"),
        "no_m" : 8,
        "nom_m" : "Jumpscare",
        "duree_m" : 224,
        "no_s" : 8,
        "likes_m" : 0,
        "dislikes_m" : 9886
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084a"),
        "no_m" : 2,
        "nom_m" : "HIMYSD Main Theme",
        "duree_m" : 130,
        "no_s" : 2,
        "likes_m" : 45698,
        "dislikes_m" : 324
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084f"),
        "no_m" : 7,
        "nom_m" : "Fire Walk With Me",
        "duree_m" : 164,
        "no_s" : 7,
        "likes_m" : 3800,
        "dislikes_m" : 987
```

}
{
"_id" : ObjectId("5e902bb5bed3f3027d440851"),
"no_m" : 9,
"nom_m" : "Alexandre Astier Remix",
"duree_m" : 578,
"no_s" : 9,
"likes_m" : 7894,
"dislikes_m" : 5896
}

Information et valeurs extrêmes des likes

7. (basic) Afficher les likes des musiques

db.musiques.distinct(“likes_m”)

[123, 45698, 4002, 23, 7896, 4567, 3800, 0, 7894, 25896]

8. (basic) Afficher le nom de la musique qui a le plus de likes

db.musiques.distinct(“nom_m”, {“likes_m” : 45698})

["HIMYSD Main Theme"]

9. (Where) Lister toutes les musiques qui ont moins de 500 likes

db.musiques.find({likes_m : {$lt : 500}}).pretty( )

{
"_id" : ObjectId("5e902bb5bed3f3027d440849"),
"no_m" : 1,
"nom_m" : "The Putt",
"duree_m" : 200,
"no_s" : 1,
"likes_m" : 123,
"dislikes_m" : 5
}
{
"_id" : ObjectId("5e902bb5bed3f3027d44084c"),
"no_m" : 4,

```
        "nom_m" : "The Sting",
        "duree_m" : 180,
        "no_s" : 4,
        "likes_m" : 23,
        "dislikes_m" : 0
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440850"),
        "no_m" : 8,
        "nom_m" : "Jumpscare",
        "duree_m" : 224,
        "no_s" : 8,
        "likes_m" : 0,
        "dislikes_m" : 9886
}
```


10. (Group by) Grouper le nombre de musiques selon le nombre de likes qu'elles ont reçu

db.musiques.aggregate([{$group : {_id : "$likes_m", total :{$sum :1}}}])


```
{ "_id" : 0, "total" : 1 }
{ "_id" : 3800, "total" : 1 }
{ "_id" : 4567, "total" : 1 }
{ "_id" : 7896, "total" : 1 }
{ "_id" : 23, "total" : 1 }
{ "_id" : 25896, "total" : 1 }
{ "_id" : 4002, "total" : 1 }
{ "_id" : 7894, "total" : 1 }
{ "_id" : 45698, "total" : 1 }
{ "_id" : 123, "total" : 1 }
```


Remarque : toutes les informations sur les likes sont très hétérogènes, le nombre de likes allant de 0 à 45 698.

Information et valeurs extrêmes des dislikes

11. (basic) Afficher les dislikes des musiques

db.musiques.distinct("dislikes_m")


```
[ 5, 324, 6587, 0, 632, 3658, 987, 9886, 5896, 464 ]
```


12. (basic) Afficher le nom de la musique qui a le plus de dislikes

db.musiques.distinct(“nom_m”, {“dislikes_m” : 9886})

["Jumpscare"]

13. (Where) Lister toutes les musiques qui ont plus de 1000 dislikes

db.musiques.find({dislikes_m : {$gt : 100}}).pretty( )


```
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084a"),
        "no_m" : 2,
        "nom_m" : "HIMYSD Main Theme",
        "duree_m" : 130,
        "no_s" : 2,
        "likes_m" : 45698,
        "dislikes_m" : 324
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084b"),
        "no_m" : 3,
        "nom_m" : "Zom-bi",
        "duree_m" : 240,
        "no_s" : 3,
        "likes_m" : 4002,
        "dislikes_m" : 6587
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084d"),
        "no_m" : 5,
        "nom_m" : "Meth Lab",
        "duree_m" : 364,
        "no_s" : 5,
        "likes_m" : 7896,
        "dislikes_m" : 632
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084e"),
        "no_m" : 6,
        "nom_m" : "Walter White Theme",
        "duree_m" : 265,
        "no_s" : 6,
        "likes_m" : 4567,
```

```
        "dislikes_m" : 3658
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084f"),
        "no_m" : 7,
        "nom_m" : "Fire Walk With Me",
        "duree_m" : 164,
        "no_s" : 7,
        "likes_m" : 3800,
        "dislikes_m" : 987
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440850"),
        "no_m" : 8,
        "nom_m" : "Jumpscare",
        "duree_m" : 224,
        "no_s" : 8,
        "likes_m" : 0,
        "dislikes_m" : 9886
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440851"),
        "no_m" : 9,
        "nom_m" : "Alexandre Astier Remix",
        "duree_m" : 578,
        "no_s" : 9,
        "likes_m" : 7894,
        "dislikes_m" : 5896
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440852"),
        "no_m" : 10,
        "nom_m" : "The Final Problem",
        "duree_m" : 333,
        "no_s" : 10,
        "likes_m" : 25896,
        "dislikes_m" : 464
}
```

14. (Group by) Grouper le nombre de musiques selon le nombre de dislikes qu'elles ont reçu

db.musiques.aggregate([{$group : {_id : "$dislikes_m", total :{$sum : 1}}}])

```
{ "_id" : 464, "total" : 1 }
{ "_id" : 9886, "total" : 1 }
{ "_id" : 987, "total" : 1 }
{ "_id" : 3658, "total" : 1 }
{ "_id" : 632, "total" : 1 }
{ "_id" : 5896, "total" : 1 }
{ "_id" : 0, "total" : 1 }
{ "_id" : 6587, "total" : 1 }
{ "_id" : 324, "total" : 1 }
{ "_id" : 5, "total" : 1 }
```

Informations sur la durée des musiques

15. (Where) Lister toutes les musiques qui durent plus de 4 minutes (240 secondes)

db.musiques.find({duree_m : {$gt : 240}}).pretty( )

```
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084d"),
        "no_m" : 5,
        "nom_m" : "Meth Lab",
        "duree_m" : 364,
        "no_s" : 5,
        "likes_m" : 7896,
        "dislikes_m" : 632
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084e"),
        "no_m" : 6,
        "nom_m" : "Walter White Theme",
        "duree_m" : 265,
        "no_s" : 6,
        "likes_m" : 4567,
        "dislikes_m" : 3658
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440851"),
        "no_m" : 9,
        "nom_m" : "Alexandre Astier Remix",
        "duree_m" : 578,
        "no_s" : 9,
        "likes_m" : 7894,
        "dislikes_m" : 5896
```

```
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440852"),
        "no_m" : 10,
        "nom_m" : "The Final Problem",
        "duree_m" : 333,
        "no_s" : 10,
        "likes_m" : 25896,
        "dislikes_m" : 464
}
```


16. (Where) Lister toutes les musiques qui durent moins de 3 minutes (180 secondes).

db.musiques.find({duree_m : {$lt : 180}}).pretty( )


```
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084a"),
        "no_m" : 2,
        "nom_m" : "HIMYSD Main Theme",
        "duree_m" : 130,
        "no_s" : 2,
        "likes_m" : 45698,
        "dislikes_m" : 324
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084f"),
        "no_m" : 7,
        "nom_m" : "Fire Walk With Me",
        "duree_m" : 164,
        "no_s" : 7,
        "likes_m" : 3800,
        "dislikes_m" : 987
}
```


17. (with numeric expressions and functions) Donner la durée maximale d'une musique

db.musiques.aggregate([{$group:{"_id":"duree_m", max_tps:{$max : "$duree_m"}}}])


```
{ "_id" : "duree_m", "max_tps" : 578 }
```

18. (with numeric expressions and functions) Donner la durée totale de musique sur la plateforme

db.musiques.aggregate([{$group:{“_id”:“duree_m”, duree_tot:{$sum : “$duree_m”}}}])

```
{ "_id" : "duree_m", "duree_tot" : 2678 }
```

19. (Group by) Grouper la durée maximum des musiques, de la plus longue à la moins longue

db.musiques.aggregate([{ $group : {_id : "$duree_m" , total : {$sum:1} } }, {$sort: {_id : -1}}])

```
{ "_id" : 578, "total" : 1 }
{ "_id" : 364, "total" : 1 }
{ "_id" : 333, "total" : 1 }
{ "_id" : 265, "total" : 1 }
{ "_id" : 240, "total" : 1 }
{ "_id" : 224, "total" : 1 }
{ "_id" : 200, "total" : 1 }
{ "_id" : 180, "total" : 1 }
{ "_id" : 164, "total" : 1 }
{ "_id" : 130, "total" : 1 }
```

Effet du « The »

20. (Where) Afficher les informations des musiques dont le titre commence par “THE”

db.musiques.find({“$or”:[{“nom_m” : “The Putt”},{“nom_m” : “The Sting”},{“nom_m” : “The Final Problem”}]}).pretty( )

```
{
        "_id" : ObjectId("5e902bb5bed3f3027d440849"),
        "no_m" : 1,
        "nom_m" : "The Putt",
        "duree_m" : 200,
        "no_s" : 1,
        "likes_m" : 123,
        "dislikes_m" : 5
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084c"),
        "no_m" : 4,
        "nom_m" : "The Sting",
        "duree_m" : 180,
        "no_s" : 4,
        "likes_m" : 23,
        "dislikes_m" : 0
}
```

{
"_id" : ObjectId("5e902bb5bed3f3027d440852"),
"no_m" : 10,
"nom_m" : "The Final Problem",
"duree_m" : 333,
"no_s" : 10,
"likes_m" : 25896,
"dislikes_m" : 464
}

Remarque : Toutes les informations des musiques commençant par « THE » sont très différentes, il faut aller regarder du côté des séries si on veut en savoir plus.

Rapport likes-dislikes

21. (Order by) Trier les musiques selon leur likes (ordre croissant) et de leur dislikes (ordre décroissant)

db.musiques.find( ).sort({“likes_m” : 1, “dislikes_m” : -1}).pretty( )

{
"_id" : ObjectId("5e902bb5bed3f3027d440850"),
"no_m" : 8,
"nom_m" : "Jumpscare",
"duree_m" : 224,
"no_s" : 8,
"likes_m" : 0,
"dislikes_m" : 9886
}
{
"_id" : ObjectId("5e902bb5bed3f3027d44084c"),
"no_m" : 4,
"nom_m" : "The Sting",
"duree_m" : 180,
"no_s" : 4,
"likes_m" : 23,
"dislikes_m" : 0
}
{
"_id" : ObjectId("5e902bb5bed3f3027d440849"),
"no_m" : 1,
"nom_m" : "The Putt",
"duree_m" : 200,

```
        "no_s" : 1,
        "likes_m" : 123,
        "dislikes_m" : 5
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084f"),
        "no_m" : 7,
        "nom_m" : "Fire Walk With Me",
        "duree_m" : 164,
        "no_s" : 7,
        "likes_m" : 3800,
        "dislikes_m" : 987
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084b"),
        "no_m" : 3,
        "nom_m" : "Zom-bi",
        "duree_m" : 240,
        "no_s" : 3,
        "likes_m" : 4002,
        "dislikes_m" : 6587
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d44084e"),
        "no_m" : 6,
        "nom_m" : "Walter White Theme",
        "duree_m" : 265,
        "no_s" : 6,
        "likes_m" : 4567,
        "dislikes_m" : 3658
}
{
        "_id" : ObjectId("5e902bb5bed3f3027d440851"),
        "no_m" : 9,
        "nom_m" : "Alexandre Astier Remix",
        "duree_m" : 578,
        "no_s" : 9,
        "likes_m" : 7894,
        "dislikes_m" : 5896
}
{
```

"_id" : ObjectId("5e902bb5bed3f3027d44084d"),
"no_m" : 5,
"nom_m" : "Meth Lab",
"duree_m" : 364,
"no_s" : 5,
"likes_m" : 7896,
"dislikes_m" : 632
}
{
"_id" : ObjectId("5e902bb5bed3f3027d440852"),
"no_m" : 10,
"nom_m" : "The Final Problem",
"duree_m" : 333,
"no_s" : 10,
"likes_m" : 25896,
"dislikes_m" : 464
}
{
"_id" : ObjectId("5e902bb5bed3f3027d44084a"),
"no_m" : 2,
"nom_m" : "HIMYSD Main Theme",
"duree_m" : 130,
"no_s" : 2,
"likes_m" : 45698,
"dislikes_m" : 324
}

22. (with numeric expressions and functions) Donner la moyenne de likes (i) et de dislikes (ii)

i. **db.musiques.aggregate([{$group:{"_id":"likes_m", moy_likes:{$avg : "$likes_m"}}}])**

{ "_id" : "likes_m", "moy_likes" : 9989.9 }

ii. **db.musiques.aggregate([{$group:{"_id":"dislikes_m",moy_dislikes:{$avg : "$dislikes_m"}}}])**

{ "_id" : "dislikes_m", "moy_dislikes" : 2843.9 }

23. (with numeric expressions and functions) Donner la somme des (i) dislikes et (ii) des likes de la plateforme

i. **db.musiques.aggregate([{$group:{"_id":"likes_m", sum_likes:{$sum :"$likes_m"}}}])**

{ "_id" : "likes_m", "sum_likes" : 99899 }

ii. **db.musiques.aggregate([{$group:{"_id":"dislikes_m",sum_dislikes:{$sum:"$dislikes_m"}}}])**

{ "_id" : "dislikes_m", "sum_dislikes" : 28439 }

24. (Group by) Grouper les musiques qui ont reçu entre 0 et 500 likes.

db.musiques.aggregate([{$match:{ "likes_m": {$lt : 500}}} , {$group : {_id : null, total:{$sum:1}}}])

{ "_id" : null, "total" : 3 }

25. (Group by) Compter le nombre de musiques qui ont reçu entre 5000 et 10000 de dislikes.

db.musiques.aggregate ([{$match:{ "dislikes_m" : {$gt : 5000, $lt : 10000}}}, {$group : {_id : null, count : {$sum :1}}}])

{ "_id" : null, "count" : 3 }

Bilan

Conseils à soumettre : supprimer la musique « Jumpscare » qui a reçu le maximum de dislikes et le minimum de likes de la plateforme. A noter que 2 autres films ont plus de 5000 dislikes, sachant qu'il y a 10 musiques et que la moyenne de dislikes est de 2844, ces musiques peuvent être également retirées.

Les musiques "The Final Problem" et "HIMYSD Main Theme" recueillent les meilleurs résultats selon le rapport entre Likes et Dislikes. Si la direction tient à ajouter une musique d'entrée sur la plateforme pour les clients, un extrait d'une de ses musiques serait apprécié au vu des retours que la plateforme a dessus.

Remarques : on ne relève pas de tendances dans les musiques dont le titre commence par « The », pas besoin de retirer les « The » du titre ou de les ajouter à chaque titre. Et enfin la durée des musiques n'influence pas le nombre de likes ou de dislikes.

Clients

HQ-streaming dispose et récolte des données sur ces clients depuis le début de son activité. Les dirigeants de la plateforme aimeraient utiliser ces données afin de prendre de meilleures décisions et de développer l'activité.

Tout d'abord, les requêtes permettent d'afficher les caractéristiques générales des clients. On peut ensuite appliquer le règlement de restriction de contenu selon le pays et l'âge du client ainsi que le prix d'abonnement selon l'âge.

On pourra alors entraîner des modèles en mettant en relation les données du profil des clients et leur comportement quant à la consommation et l'appréciation de séries, de musiques et de films que la plateforme a regroupé par catégorie. On pourra ainsi essayer d'établir des profils types et d'affiner les recommandations de façon à fidéliser les clients. On pourra aussi se servir de ses résultats pour mieux choisir les projets de publicité selon leur cible.

Caractéristiques générales des clients

1. Les caractéristiques des clients (Basic)

db.clients.find({}).pretty()


```
{
        "_id"
: ObjectId("5e8eebe2dcc8b02b9f336b32"),
        "no_c"
: 1,
        "nom_c"
: "Jean-Jean",
        "age_c"
: 56,
        "no_pays"
: 1,
        "dernier_f"
: 7,
        "dernier_s"
: 8
}
```

```
{
	"_id"
: ObjectId("5e8eebe2dcc8b02b9f336b33"),
	"no_c"
: 2,
	"nom_c"
: "Thierry Morvan",
	"age_c"
: 89,
	"no_pays"
: 1,
	"dernier_f"
: 4,
	"dernier_s"
: 9
}
{
	"_id"
: ObjectId("5e8eebe2dcc8b02b9f336b34"),
	"no_c"
: 3,
	"nom_c"
: "Victor Pion",
	"age_c"
: 5,
	"no_pays"
: 1,
	"dernier_f"
: 1,
	"dernier_s"
: 5
}
{
	"_id"
```

: ObjectId("5e8eebe2dcc8b02b9f336b35"),
"no_c"
: 4,
"nom_c"
: "Donald Trump",
"age_c"
: 2,
"no_pays"
: 2,
"dernier_f"
: 3,
"dernier_s"
: 4
}
{
"_id"
: ObjectId("5e8eebe2dcc8b02b9f336b36"),
"no_c"
: 5,
"nom_c"
: "Pedro Pierdo el Telephono",
"age_c"
: 18,
"no_pays"
: 2,
"dernier_f"
: 2,
"dernier_s"
: 6
}
{
"_id"
: ObjectId("5e8eebe2dcc8b02b9f336b37"),
"no_c"

: 6,
"nom_c"
: "Loris à Seattle avec Shalom peps",
"age_c"
: 69,
"no_pays"
: 2,
"dernier_f"
: 8,
"dernier_s"
: 1
}
{
"_id"
: ObjectId("5e8eebe2dcc8b02b9f336b38"),
"no_c"
: 7,
"nom_c"
: "Carlos Ghosn",
"age_c"
: 4,
"no_pays"
: 4,
"dernier_f"
: 5,
"dernier_s"
: 2
}
{
"_id"
: ObjectId("5e8eebe2dcc8b02b9f336b39"),
"no_c"
: 8,
"nom_c"

: "Luc Marty",
"age_c"
: 35,
"no_pays"
: 5,
"dernier_f"
: 10,
"dernier_s"
: 3
}
{
"_id"
: ObjectId("5e8eebe2dcc8b02b9f336b3a"),
"no_c"
: 9,
"nom_c"
: "Maurice Johnson",
"age_c"
: 50,
"no_pays"
: 3,
"dernier_f"
: 6,
"dernier_s"
: 7
}
{
"_id"
: ObjectId("5e8eebe2dcc8b02b9f336b3b"),
"no_c"
: 10,
"nom_c"
: "Laura T",
"age_c"

: 9,

"no_pays"

: 5,

"dernier_f"

: 9,

"dernier_s"

: 10

}

2. Les noms des clients (Basic)

db.clients.distinct("nom_c")

[

"Jean-Jean",

"Thierry Morvan",

"Victor Pion",

"Donald Trump",

"Pedro Pierdo el Telephono",

"Loris à Seattle avec Shalom peps",

"Carlos Ghosn",

"Luc Marty",

"Maurice Johnson",

"Laura T"

]

3. Trier les clients selon le nom dans l'ordre alphabétique et l'âge croissant (Order by)

db.clients.find().sort({"nom_c": 1, "age_c" : 1}).pretty()

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b38"),

"no_c" : 7,

"nom_c" : "Carlos Ghosn",

"age_c" : 4,

"no_pays" : 4,

```
        "dernier_f" : 5,
        "dernier_s" : 2
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b35"),
        "no_c" : 4,
        "nom_c" : "Donald Trump",
        "age_c" : 2,
        "no_pays" : 2,
        "dernier_f" : 3,
        "dernier_s" : 4
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b32"),
        "no_c" : 1,
        "nom_c" : "Jean-Jean",
        "age_c" : 56,
        "no_pays" : 1,
        "dernier_f" : 7,
        "dernier_s" : 8
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3b"),
        "no_c" : 10,
        "nom_c" : "Laura T",
        "age_c" : 9,
        "no_pays" : 5,
        "dernier_f" : 9,
        "dernier_s" : 10
}
```

```
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),
        "no_c" : 6,
        "nom_c" : "Loris à Seattle avec Shalom peps",
        "age_c" : 69,
        "no_pays" : 2,
        "dernier_f" : 8,
        "dernier_s" : 1
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b39"),
        "no_c" : 8,
        "nom_c" : "Luc Marty",
        "age_c" : 35,
        "no_pays" : 5,
        "dernier_f" : 10,
        "dernier_s" : 3
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3a"),
        "no_c" : 9,
        "nom_c" : "Maurice Johnson",
        "age_c" : 50,
        "no_pays" : 3,
        "dernier_f" : 6,
        "dernier_s" : 7
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),
        "no_c" : 5,
```

```
        "nom_c" : "Pedro Pierdo el Telephono",
        "age_c" : 18,
        "no_pays" : 2,
        "dernier_f" : 2,
        "dernier_s" : 6
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b33"),
        "no_c" : 2,
        "nom_c" : "Thierry Morvan",
        "age_c" : 89,
        "no_pays" : 1,
        "dernier_f" : 4,
        "dernier_s" : 9
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b34"),
        "no_c" : 3,
        "nom_c" : "Victor Pion",
        "age_c" : 5,
        "no_pays" : 1,
        "dernier_f" : 1,
        "dernier_s" : 5
}
```


4. Trier les clients selon le nom anti-alphabétique (Order by)

db.clients.find().sort({"nom_client" : -1}).pretty()


```
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b32"),
        "no_c" : 1,
        "nom_c" : "Jean-Jean",
```

```
        "age_c" : 56,
        "no_pays" : 1,
        "dernier_f" : 7,
        "dernier_s" : 8
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b33"),
        "no_c" : 2,
        "nom_c" : "Thierry Morvan",
        "age_c" : 89,
        "no_pays" : 1,
        "dernier_f" : 4,
        "dernier_s" : 9
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b34"),
        "no_c" : 3,
        "nom_c" : "Victor Pion",
        "age_c" : 5,
        "no_pays" : 1,
        "dernier_f" : 1,
        "dernier_s" : 5
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b35"),
        "no_c" : 4,
        "nom_c" : "Donald Trump",
        "age_c" : 2,
        "no_pays" : 2,
        "dernier_f" : 3,
```

```
        "dernier_s" : 4
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),
        "no_c" : 5,
        "nom_c" : "Pedro Pierdo el Telephono",
        "age_c" : 18,
        "no_pays" : 2,
        "dernier_f" : 2,
        "dernier_s" : 6
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),
        "no_c" : 6,
        "nom_c" : "Loris à Seattle avec Shalom peps",
        "age_c" : 69,
        "no_pays" : 2,
        "dernier_f" : 8,
        "dernier_s" : 1
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b38"),
        "no_c" : 7,
        "nom_c" : "Carlos Ghosn",
        "age_c" : 4,
        "no_pays" : 4,
        "dernier_f" : 5,
        "dernier_s" : 2
}
{
```

```
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b39"),
        "no_c" : 8,
        "nom_c" : "Luc Marty",
        "age_c" : 35,
        "no_pays" : 5,
        "dernier_f" : 10,
        "dernier_s" : 3
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3a"),
        "no_c" : 9,
        "nom_c" : "Maurice Johnson",
        "age_c" : 50,
        "no_pays" : 3,
        "dernier_f" : 6,
        "dernier_s" : 7
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3b"),
        "no_c" : 10,
        "nom_c" : "Laura T",
        "age_c" : 9,
        "no_pays" : 5,
        "dernier_f" : 9,
        "dernier_s" : 10
}
```

5. Classer les clients dont le nom commence par L par numéro de pays (queries with numeric expressions and functions)

db.clients.aggregate([{$match :{ nom_c : /^L/ }}, {$sort : {no_pays :1}}]).pretty()


```
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),
        "no_c" : 6,
        "nom_c" : "Loris à Seattle avec Shalom peps",
        "age_c" : 69,
        "no_pays" : 2,
        "dernier_f" : 8,
        "dernier_s" : 1
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b39"),
        "no_c" : 8,
        "nom_c" : "Luc Marty",
        "age_c" : 35,
        "no_pays" : 5,
        "dernier_f" : 10,
        "dernier_s" : 3
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3b"),
        "no_c" : 10,
        "nom_c" : "Laura T",
        "age_c" : 9,
        "no_pays" : 5,
        "dernier_f" : 9,
        "dernier_s" : 10
}
```

6. Compter le nombre de client (queries with numeric expressions and functions)

db.clients.count()

10

7. L'age du client le plus âgé (queries with numeric expressions and functions)

db.clients.aggregate([{$group:{"_id": null , age :{ $max : "$age_c"}}}])

{ "_id" : null, "age" : 89 }

Application du tarif et du règlement de restriction

8. Le nom des clients venant du pays 2 (Basic)

db.clients.distinct("nom_c",{"no_pays": 2})

[

"Donald Trump",

"Pedro Pierdo el Telephono",

"Loris à Seattle avec Shalom peps"

]

9. Clients dont le numéro du pays est 2 (Where).

db.clients.find({"no_pays": 2}).pretty()

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b35"),

"no_c" : 4,

"nom_c" : "Donald Trump",

"age_c" : 2,

"no_pays" : 2,

"dernier_f" : 3,

"dernier_s" : 4

}

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),

```
        "no_c" : 5,
        "nom_c" : "Pedro Pierdo el Telephono",
        "age_c" : 18,
        "no_pays" : 2,
        "dernier_f" : 2,
        "dernier_s" : 6
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),
        "no_c" : 6,
        "nom_c" : "Loris à Seattle avec Shalom peps",
        "age_c" : 69,
        "no_pays" : 2,
        "dernier_f" : 8,
        "dernier_s" : 1
}
```


10. Clients dont l'âge est supérieur ou égal à 12 (donc pas de restriction)(Where)

db.clients.find({"age_c":{$gte : 12}}).pretty()


```
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b32"),
        "no_c" : 1,
        "nom_c" : "Jean-Jean",
        "age_c" : 56,
        "no_pays" : 1,
        "dernier_f" : 7,
        "dernier_s" : 8
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b33"),
        "no_c" : 2,
```

```
        "nom_c" : "Thierry Morvan",
        "age_c" : 89,
        "no_pays" : 1,
        "dernier_f" : 4,
        "dernier_s" : 9
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),
        "no_c" : 5,
        "nom_c" : "Pedro Pierdo el Telephono",
        "age_c" : 18,
        "no_pays" : 2,
        "dernier_f" : 2,
        "dernier_s" : 6
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),
        "no_c" : 6,
        "nom_c" : "Loris à Seattle avec Shalom peps",
        "age_c" : 69,
        "no_pays" : 2,
        "dernier_f" : 8,
        "dernier_s" : 1
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b39"),
        "no_c" : 8,
        "nom_c" : "Luc Marty",
        "age_c" : 35,
        "no_pays" : 5,
```

```
        "dernier_f" : 10,
        "dernier_s" : 3
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3a"),
        "no_c" : 9,
        "nom_c" : "Maurice Johnson",
        "age_c" : 50,
        "no_pays" : 3,
        "dernier_f" : 6,
        "dernier_s" : 7
}
```


11. Clients qui ont plus de 12 ans et qui sont dans le pays 1 (Where)

db.clients.find({"$and":[{"age_c" :{$gte :12}}, {"no_pays" :1 }]}).pretty()


```
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b32"),
        "no_c" : 1,
        "nom_c" : "Jean-Jean",
        "age_c" : 56,
        "no_pays" : 1,
        "dernier_f" : 7,
        "dernier_s" : 8
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b33"),
        "no_c" : 2,
        "nom_c" : "Thierry Morvan",
        "age_c" : 89,
        "no_pays" : 1,
        "dernier_f" : 4,
```

"dernier_s" : 9
}

12. Clients qui ont plus de 12 et moins de 60 ans (tarif normal) (Where)

db.clients.find({"$and":[{"age_c" :{$gte :12}}, { "age_c" :{$lt : 60}}]}).pretty()

{
"_id" : ObjectId("5e8eebe2dcc8b02b9f336b32"),
"no_c" : 1,
"nom_c" : "Jean-Jean",
"age_c" : 56,
"no_pays" : 1,
"dernier_f" : 7,
"dernier_s" : 8
}
{
"_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),
"no_c" : 5,
"nom_c" : "Pedro Pierdo el Telephono",
"age_c" : 18,
"no_pays" : 2,
"dernier_f" : 2,
"dernier_s" : 6
}
{
"_id" : ObjectId("5e8eebe2dcc8b02b9f336b39"),
"no_c" : 8,
"nom_c" : "Luc Marty",
"age_c" : 35,
"no_pays" : 5,
"dernier_f" : 10,
"dernier_s" : 3

}

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b3a"),

"no_c" : 9,

"nom_c" : "Maurice Johnson",

"age_c" : 50,

"no_pays" : 3,

"dernier_f" : 6,

"dernier_s" : 7

}

13. Trier les clients selon le pays (croissant) (Order by)

db.clients.find().sort({"no_pays": 1}).pretty()

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b32"),

"no_c" : 1,

"nom_c" : "Jean-Jean",

"age_c" : 56,

"no_pays" : 1,

"dernier_f" : 7,

"dernier_s" : 8

}

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b33"),

"no_c" : 2,

"nom_c" : "Thierry Morvan",

"age_c" : 89,

"no_pays" : 1,

"dernier_f" : 4,

"dernier_s" : 9

}

```
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b34"),
        "no_c" : 3,
        "nom_c" : "Victor Pion",
        "age_c" : 5,
        "no_pays" : 1,
        "dernier_f" : 1,
        "dernier_s" : 5
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b35"),
        "no_c" : 4,
        "nom_c" : "Donald Trump",
        "age_c" : 2,
        "no_pays" : 2,
        "dernier_f" : 3,
        "dernier_s" : 4
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),
        "no_c" : 5,
        "nom_c" : "Pedro Pierdo el Telephono",
        "age_c" : 18,
        "no_pays" : 2,
        "dernier_f" : 2,
        "dernier_s" : 6
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),
        "no_c" : 6,
```

```
        "nom_c" : "Loris à Seattle avec Shalom peps",
        "age_c" : 69,
        "no_pays" : 2,
        "dernier_f" : 8,
        "dernier_s" : 1
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3a"),
        "no_c" : 9,
        "nom_c" : "Maurice Johnson",
        "age_c" : 50,
        "no_pays" : 3,
        "dernier_f" : 6,
        "dernier_s" : 7
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b38"),
        "no_c" : 7,
        "nom_c" : "Carlos Ghosn",
        "age_c" : 4,
        "no_pays" : 4,
        "dernier_f" : 5,
        "dernier_s" : 2
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b39"),
        "no_c" : 8,
        "nom_c" : "Luc Marty",
        "age_c" : 35,
        "no_pays" : 5,
```

```
        "dernier_f" : 10,
        "dernier_s" : 3
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3b"),
        "no_c" : 10,
        "nom_c" : "Laura T",
        "age_c" : 9,
        "no_pays" : 5,
        "dernier_f" : 9,
        "dernier_s" : 10
}
```


14. Trier les clients selon l'âge et le pays croissant (Order by)

db.clients.find().sort({"no_pays": 1, "age_c" : 1}).pretty()


```
        {
            "_id" : ObjectId("5e8eebe2dcc8b02b9f336b34"),
            "no_c" : 3,
            "nom_c" : "Victor Pion",
            "age_c" : 5,
            "no_pays" : 1,
            "dernier_f" : 1,
            "dernier_s" : 5
        }
        {
            "_id" : ObjectId("5e8eebe2dcc8b02b9f336b32"),
            "no_c" : 1,
            "nom_c" : "Jean-Jean",
            "age_c" : 56,
            "no_pays" : 1,
            "dernier_f" : 7,
```

```
    "dernier_s" : 8
}
{
    "_id" : ObjectId("5e8eebe2dcc8b02b9f336b33"),
    "no_c" : 2,
    "nom_c" : "Thierry Morvan",
    "age_c" : 89,
    "no_pays" : 1,
    "dernier_f" : 4,
    "dernier_s" : 9
}
{
    "_id" : ObjectId("5e8eebe2dcc8b02b9f336b35"),
    "no_c" : 4,
    "nom_c" : "Donald Trump",
    "age_c" : 2,
    "no_pays" : 2,
    "dernier_f" : 3,
    "dernier_s" : 4
}
{
    "_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),
    "no_c" : 5,
    "nom_c" : "Pedro Pierdo el Telephono",
    "age_c" : 18,
    "no_pays" : 2,
    "dernier_f" : 2,
    "dernier_s" : 6
}
{
```

```
    "_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),
    "no_c" : 6,
    "nom_c" : "Loris à Seattle avec Shalom peps",
    "age_c" : 69,
    "no_pays" : 2,
    "dernier_f" : 8,
    "dernier_s" : 1
}
{
    "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3a"),
    "no_c" : 9,
    "nom_c" : "Maurice Johnson",
    "age_c" : 50,
    "no_pays" : 3,
    "dernier_f" : 6,
    "dernier_s" : 7
}
{
    "_id" : ObjectId("5e8eebe2dcc8b02b9f336b38"),
    "no_c" : 7,
    "nom_c" : "Carlos Ghosn",
    "age_c" : 4,
    "no_pays" : 4,
    "dernier_f" : 5,
    "dernier_s" : 2
}
{
    "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3b"),
    "no_c" : 10,
    "nom_c" : "Laura T",
```

```
        "age_c" : 9,
        "no_pays" : 5,
        "dernier_f" : 9,
        "dernier_s" : 10
    }
    {
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b39"),
        "no_c" : 8,
        "nom_c" : "Luc Marty",
        "age_c" : 35,
        "no_pays" : 5,
        "dernier_f" : 10,
        "dernier_s" : 3
    }
```


15. Trier les clients plus âgés que 12 ans et par pays (Order by)

db.clients.aggregate([{"$match" : { "age_c" : {$gte : 12 }}}, {"$sort":{"no_pays": 1}}]).pretty()


```
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b32"),
        "no_c" : 1,
        "nom_c" : "Jean-Jean",
        "age_c" : 56,
        "no_pays" : 1,
        "dernier_f" : 7,
        "dernier_s" : 8
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b33"),
        "no_c" : 2,
        "nom_c" : "Thierry Morvan",
        "age_c" : 89,
```

```
        "no_pays" : 1,
        "dernier_f" : 4,
        "dernier_s" : 9
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),
        "no_c" : 5,
        "nom_c" : "Pedro Pierdo el Telephono",
        "age_c" : 18,
        "no_pays" : 2,
        "dernier_f" : 2,
        "dernier_s" : 6
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),
        "no_c" : 6,
        "nom_c" : "Loris à Seattle avec Shalom peps",
        "age_c" : 69,
        "no_pays" : 2,
        "dernier_f" : 8,
        "dernier_s" : 1
}
{
        "_id" : ObjectId("5e8eebe2dcc8b02b9f336b3a"),
        "no_c" : 9,
        "nom_c" : "Maurice Johnson",
        "age_c" : 50,
        "no_pays" : 3,
        "dernier_f" : 6,
        "dernier_s" : 7
```

}

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b39"),

"no_c" : 8,

"nom_c" : "Luc Marty",

"age_c" : 35,

"no_pays" : 5,

"dernier_f" : 10,

"dernier_s" : 3

}

16. Grouper les clients par pays (Group by)

db.clients.aggregate([{ $group : {_id : "$no_pays" , total : {$sum:1} } }])

{ "_id" : 3, "total" : 1 }

{ "_id" : 5, "total" : 2 }

{ "_id" : 4, "total" : 1 }

{ "_id" : 2, "total" : 3 }

{ "_id" : 1, "total" : 3 }

17. Compter le nombre de clients appartenant au groupe des plus de 12 ans (Group by)

db.clients.aggregate([{$match : { age_c : {$gte :12}}}, {$group :{_id :null , somme : {$sum :1}}}])

{ "_id" : null, "somme" : 6 }

18. Grouper les clients qui ont plus de 12 ans, par pays(Group by).

db.clients.aggregate([{$match : { age_c : {$gte :12}}}, { $group : {_id : "$no_pays" , total : {$sum:1}}}])

{ "_id" : 3, "total" : 1 }

{ "_id" : 5, "total" : 1 }

{ "_id" : 2, "total" : 2 }

{ "_id" : 1, "total" : 2 }

Eléments de profilage des clients

19. L'âge des clients (Basic)

db.clients.distinct("age_c")

[56, 89, 5, 2, 18, 69, 4, 35, 50, 9]

20. L'identifiant du dernier film regardé (Basic)

db.clients.distinct("dernier_s")

[8, 9, 5, 4, 6, 1, 2, 3, 7, 10]

21. Clients pour lequel le dernier film et la dernière série regardés était le numéro 5 et 2 respectivement (Where)

db.clients.find({"$and":[{"dernier_f" : 5}, {"dernier_s" :2 }]}).pretty()

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b38"),

"no_c" : 7,

"nom_c" : "Carlos Ghosn",

"age_c" : 4,

"no_pays" : 4,

"dernier_f" : 5,

"dernier_s" : 2

}

22. Classer les clients du pays numéro 2 par âge décroissant (queries with numeric expressions and functions)

db.clients.aggregate([{$match :{ no_pays : 2 }}, {$sort : { age_c : -1} }]).pretty()

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b37"),

"no_c" : 6,

"nom_c" : "Loris à Seattle avec Shalom peps",

"age_c" : 69,

"no_pays" : 2,

"dernier_f" : 8,

"dernier_s" : 1

}

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b36"),

"no_c" : 5,

"nom_c" : "Pedro Pierdo el Telephono",

"age_c" : 18,

"no_pays" : 2,

"dernier_f" : 2,

"dernier_s" : 6

}

{

"_id" : ObjectId("5e8eebe2dcc8b02b9f336b35"),

"no_c" : 4,

"nom_c" : "Donald Trump",

"age_c" : 2,

"no_pays" : 2,

"dernier_f" : 3,

"dernier_s" : 4

}

23. Age moyen des clients (queries with numeric expressions and functions).

db.clients.aggregate([{$group:{"_id":"no_c", age_moyen :{ $avg : "$age_c"}}}])

{ "_id" : "no_c", "age_moyen" : 33.7 }

24. Grouper les clients par dernier film regarder (Group by)

db.clients.aggregate([{ $group : {_id : "$dernier_f" , total : {$sum:1} } }])

{ "_id" : 9, "total" : 1 }

{ "_id" : 6, "total" : 1 }

{ "_id" : 10, "total" : 1 }

{ "_id" : 5, "total" : 1 }

{ "_id" : 8, "total" : 1 }

{ "_id" : 2, "total" : 1 }

{ "_id" : 3, "total" : 1 }

{ "_id" : 4, "total" : 1 }

{ "_id" : 1, "total" : 1 }

{ "_id" : 7, "total" : 1 }

25. Grouper les clients par âge dans l'ordre décroissant (Group by).

```
db.clients.aggregate( [ { $group : {_id : "$age_c", total : {$sum:1} } }, {$sort: {_id: -1}} ]
).pretty()
```

{ "_id" : 89, "total" : 1 }

{ "_id" : 69, "total" : 1 }

{ "_id" : 56, "total" : 1 }

{ "_id" : 50, "total" : 1 }

{ "_id" : 35, "total" : 1 }

{ "_id" : 18, "total" : 1 }

{ "_id" : 9, "total" : 1 }

{ "_id" : 5, "total" : 1 }

{ "_id" : 4, "total" : 1 }

{ "_id" : 2, "total" : 1 }

Bilan

Les requêtes ont permis d'avoir une vue d'ensemble sur les clients. On peut alors appliquer les restrictions selon l'âge et le pays de chaque individu. L'âge moyen est de 33 ans mais cache des disparités élevées donc on pourra essayer de classifier plusieurs profils de client afin de personnaliser les recommandations de contenu sur la plateforme.

Conclusion

L’étude des ingénieurs informaticiens a permis aux dirigeants de HQ-Streaming d’avoir une vue d’ensemble chiffrée du fonctionnement de leur plateforme. Désormais, ils ont également à leur disposition une étude précise sur leurs clients et les programmes auxquels leur plateforme donne accès pour affiner leur prise de décision. Ils vont notamment pouvoir envisager de donner accès à plus de séries toujours aussi diversifiées à leurs clients, de supprimer la série “Jumpscare” peu appréciée ou encore de personnaliser les contenus selon l’âge des clients au vu de la forte disparité d’âge entre les abonnés. Ainsi, grâce à cette étude les dirigeants de HQ-Streaming peuvent aborder la concurrence avec les autres plateformes de streaming mieux armés.