Jobintech

MAROC ynov CAMPUS

Rapport de projet

SQL & NoSQL

GLITCH

- *Mohamed AIT MHAMED*
- *Ossama BENZEROUALA*
- *Islam CHLIH*
- *Maryam AYAD*

Encadré par :

- *M. Oussama ETTALALI*

Date : 14-10-2025

Table des matières

Introduction générale

Contexte

Dans le cadre de notre formation et afin de **mettre en pratique l'ensemble des connaissances acquises** tout au long du cours de **bases de données**, deux projets de synthèse ont été réalisés. Ces projets représentent une opportunité concrète d'application des concepts de modélisation, de manipulation et d'analyse de données, aussi bien dans un environnement relationnel (MySQL) que non relationnel (MongoDB).

Le premier projet, intitulé **"Advanced E-commerce Platform (Gadget World)"**, a été développé sous **MySQL**. Il vise à concevoir une base de données complète pour la gestion d'une entreprise e-commerce, intégrant les aspects de catalogage des produits, gestion des commandes, paiements, fournisseurs et entrepôts multiples. Ce projet met l'accent sur la modélisation relationnelle, les contraintes d'intégrité, les transactions et la création de requêtes analytiques complexes.

Le deuxième projet **"ConnectSphere"**, réalisé avec **MongoDB**, explore le paradigme **NoSQL** à travers la création et la manipulation de collections de documents. Il met en avant la flexibilité du modèle de données, l'utilisation d'opérations CRUD et l'exploitation des **Aggregation** Pipelines pour l'analyse et la visualisation des données.

Ensemble, ces deux projets démontrent la maîtrise des outils et des techniques nécessaires à la conception, la gestion et l'exploitation efficace de bases de données modernes, aussi bien relationnelles que non relationnelles.

Objectifs

L'objectif principal de ce travail est de mettre en œuvre les compétences acquises en conception, gestion et exploitation de bases de données à travers la réalisation de deux projets complémentaires. Il s'agit de :

- **Concevoir des structures de données performantes et cohérentes**, adaptées aux besoins spécifiques d'un système d'information.
- **Appliquer les principes du modèle relationnel et du modèle NoSQL**, afin de comparer leurs approches et leurs avantages respectifs.
- **Manipuler et analyser efficacement les données** à l'aide de requêtes SQL et d'opérations MongoDB avancées.
- **Développer une compréhension pratique** des enjeux liés à la modélisation, à l'intégrité et à la performance des bases de données dans des contextes réels.

Projet 1:

e-commerce company

Gadget World

Phase 1: schema.sql (DDL language de definition des données)

Cette phase consiste à créer toutes les tables nécessaires pour la plateforme de commerce électronique **Gadget World**, en définissant leurs colonnes**,** types de données**,** clés primaires**,** clés étrangères**,** et autres contraintes**.**

Creation de base de données et de tables

Command 1: Create Database gadget_world

```
CREATE DATABASE IF NOT EXISTS gadget_world;
```

Explication:

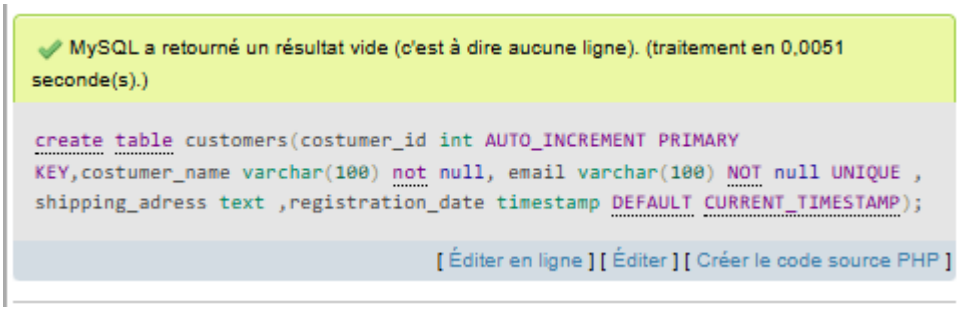
Cette instruction SQL crée une nouvelle base de données nommée **gadget_world**.
La clause **IF NOT EXISTS** empêche l'apparition d'une erreur si une base de données portant le même nom existe déjà.
Cela garantit que le script peut être exécuté plusieurs fois sans provoquer de problèmes.

Command 2 : use database gadget world

```
USE gadget_world;
```

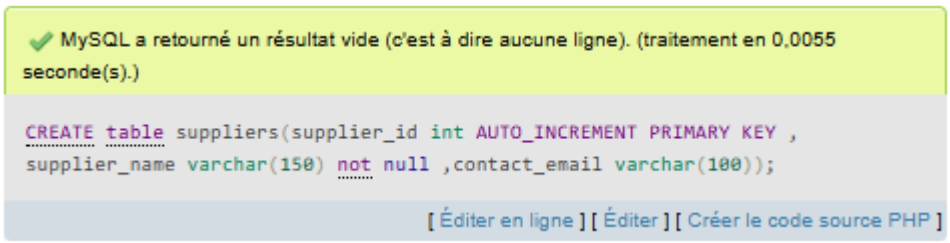
Explication :
Cette commande sélectionne la base de données gadget_world comme base de données active pour les opérations suivantes.
Ainsi, toutes les instructions **CREATE TABLE**, **INSERT**, **UPDATE** et **DELETE** seront désormais appliquées à cette base de données**.**

Command 3 : create table customers



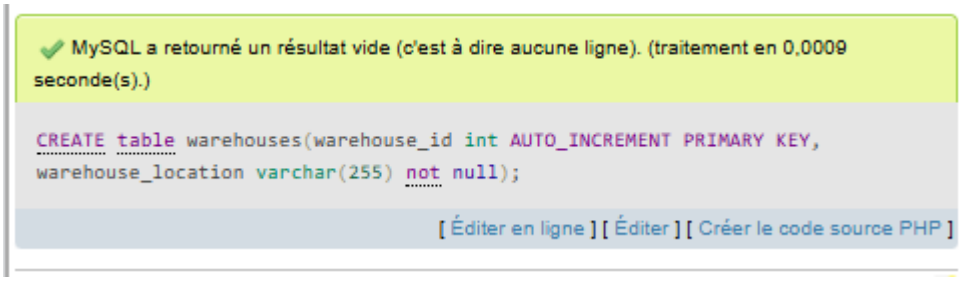
Explication :
Cette instruction crée la table Customers afin de stocker les informations des clients**.**

Command 4: create table suppliers



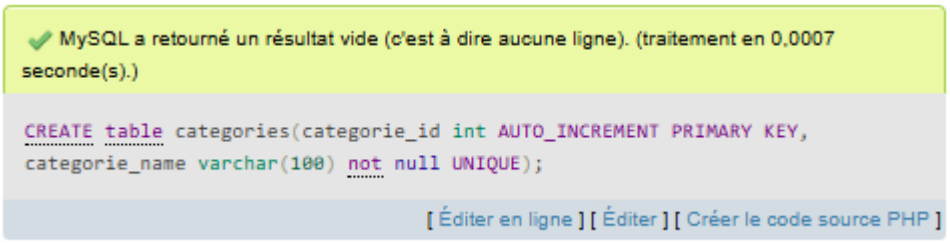
Explication :
Cette instruction crée la table **Suppliers** afin de gérer les informations concernant les fournisseurs de produits**.**

Command 5: create table warehouses



Explication :
Cette instruction crée la table Warehouses afin de répertorier les emplacements physiques où les stocks sont conservés**.**

Command 6: create table categories



Explication :
Cette instruction crée la table **Categories** afin d'organiser les produits en groupes logiques.

Command 7: create table products

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0022 seconde(s).)

```
CREATE table products(product_id int AUTO_INCREMENT PRIMARY KEY, product_name
varchar(200) not null , description text , price decimal(10,2) not null
CHECK(price>0), categorie_id int , supplier_id int , FOREIGN KEY
(supplier_id) REFERENCES Suppliers(supplier_id), FOREIGN KEY (categorie_id)
REFERENCES Categories(categorie_id));
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Explication :
Cette instruction crée la table **Products** afin de gérer le catalogue des produits**.**

Command 8: create table inventory

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0014 seconde(s).)

```
CREATE table inventory(product_id int , warehouse_id int,quantity int not
null DEFAULT 0 ,PRIMARY KEY (product_id, warehouse_id), FOREIGN KEY
(product_id) REFERENCES Products(product_id) ON DELETE CASCADE, FOREIGN KEY
(warehouse_id) REFERENCES Warehouses(warehouse_id));
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Explication :
Cette instruction crée la table **Inventory**, une table de jonction pour représenter la relation many-to-many entre Products et Warehouses, afin de suivre le stock de chaque produit dans chaque entrepôt**.**

Command 9: create table reviews

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0021 seconde(s).)

```
CREATE TABLE Reviews ( review_id INT AUTO_INCREMENT PRIMARY KEY, product_id
INT, customer_id INT, rating INT NOT NULL CHECK (rating BETWEEN 1 AND 5),
comment_text TEXT, review_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP, FOREIGN
KEY (product_id) REFERENCES Products(product_id), FOREIGN KEY (customer_id)
REFERENCES Customers(costumer_id) );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Explication :
Cette instruction crée la table **Reviews** afin de stocker les avis des clients sur les produits.

Command 10 : create table orders

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0030 seconde(s).)

```
CREATE TABLE Orders ( order_id INT AUTO_INCREMENT PRIMARY KEY, costumer_id INT, order_date
TIMESTAMP DEFAULT CURRENT_TIMESTAMP, status
ENUM('Pending','Processing','Shipped','Delivered','Canceled') DEFAULT 'Pending', FOREIGN KEY
(costumer_id) REFERENCES customers(costumer_id) );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Explication :
Cette instruction crée la table **Orders** afin de suivre les commandes des clients**.**

Command 11: create table order_item

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0011 seconde(s).)

```
CREATE TABLE Order_items ( order_item_id INT AUTO_INCREMENT PRIMARY KEY,
order_id INT, product_id int, quantity int not null check(quantity>0) ,
unit_price decimal(10,2) not null, FOREIGN KEY (order_id) REFERENCES
orders(order_id), FOREIGN KEY (product_id) REFERENCES products(product_id) );
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Explication :
Cette instruction crée la table **Order_Items**, qui détaille les produits inclus dans chaque commande**.**
Cette table représente une relation many-to-many entre Orders et Products**.**

Command 12: create table payments

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0025 seconde(s).)

```
CREATE TABLE Payments ( payment_id INT AUTO_INCREMENT PRIMARY KEY, order_id
INT, payment_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP, amount DECIMAL(12,2)
NOT NULL, payment_method ENUM('Credit Card','PayPal','Bank Transfer'),
FOREIGN KEY (order_id) REFERENCES Orders(order_id) ON DELETE CASCADE );
```

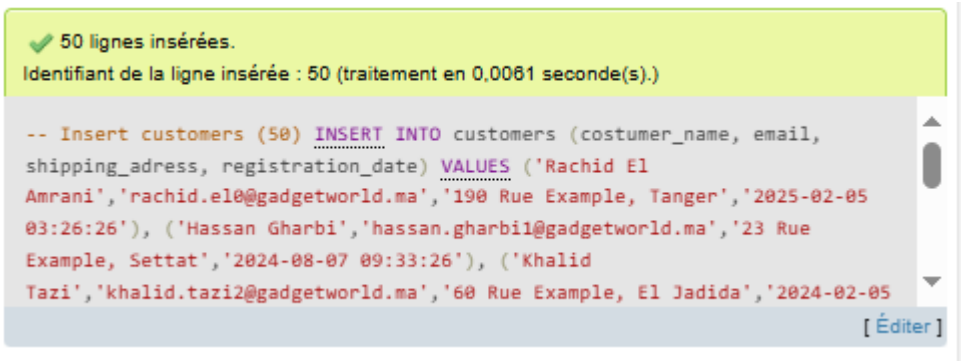
[Éditer en ligne] [Éditer] [Créer le code source PHP]

Explication :
Cette instruction crée la table **Payments** afin de consigner les transactions de paiement associées aux commandes**.**

Phase 2: data.sql

Insertion des données

Command 1 : insert data into customers table

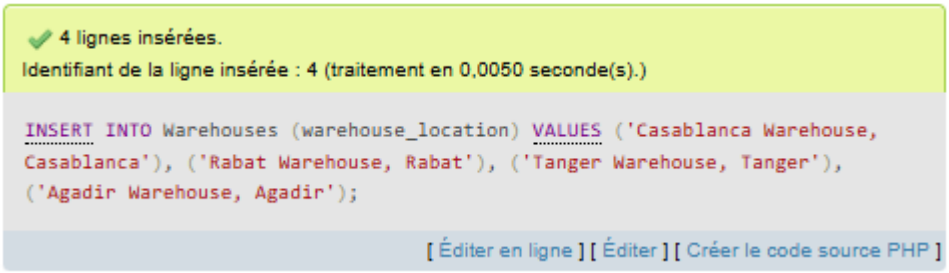


Explication :
Cette instruction INSERT INTO remplit la table Customers.
Nous spécifions les colonnes customer_name, email et shipping_address.
La colonne customer_id est générée automatiquement grâce à l'AUTO_INCREMENT, et registration_date prend par défaut la date et l'heure actuelles**.**
Nous avons inséré **50 enregistrements distincts** de clients, comme requis.

Command 2: insert data info suppliers table

Explication :
Cette instruction remplit la table Suppliers avec **10 enregistrements**.
Chaque enregistrement inclut le supplier_name et le contact_email.
La colonne supplier_id est générée automatiquement grâce à l'AUTO_INCREMENT.

Command 3: insert data into warehouses table



Explication :
Cette instruction remplit la table Warehouses avec 4 enregistrements, chacun précisant un warehouse_location. La colonne warehouse_id est générée automatiquement.

Command 4: insert data into categories table

Explication :
Cette instruction remplit la table Categories avec 20 enregistrements distincts, chacun précisant un category_name. La colonne category_id est générée automatiquement.

Command 5: insert data into products table

Explication :
Cette instruction INSERT INTO remplit la table Products avec 100 produits différents. Chaque produit possède un product_name, une description, un price, et surtout un category_id et un supplier_id pour les lier à leurs catégories et fournisseurs respectifs. La colonne product_id est auto-incrémentée. Nous avons veillé à ce que le prix soit positif, conformément à la contrainte CHECK de la table.

Command 6: insert data into inventory table

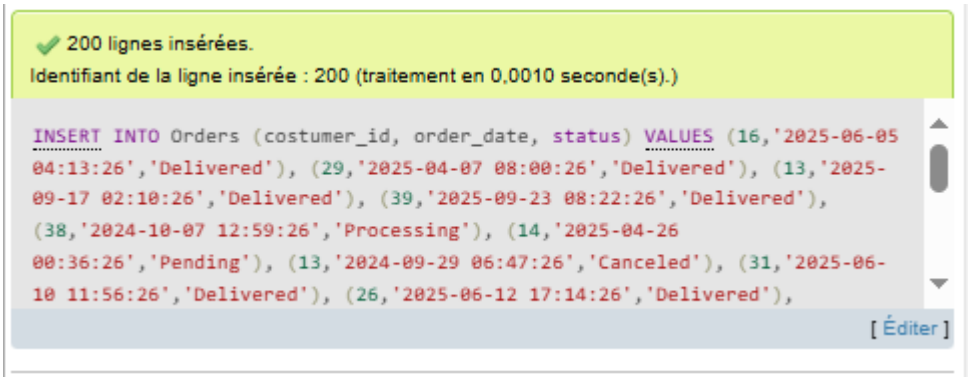
Explanation:

This INSERT INTO statement populates the Inventory table. It establishes the stock quantity of specific product_id in specific warehouse_id locations. Since it's a many-to-many relationship, a single product can appear in multiple warehouses, and a single warehouse can hold multiple products. We aim for at least 160 records. I've provided a comprehensive list that meets this requirement by assigning multiple products to various warehouses.

Command 7: insert data into reviews table

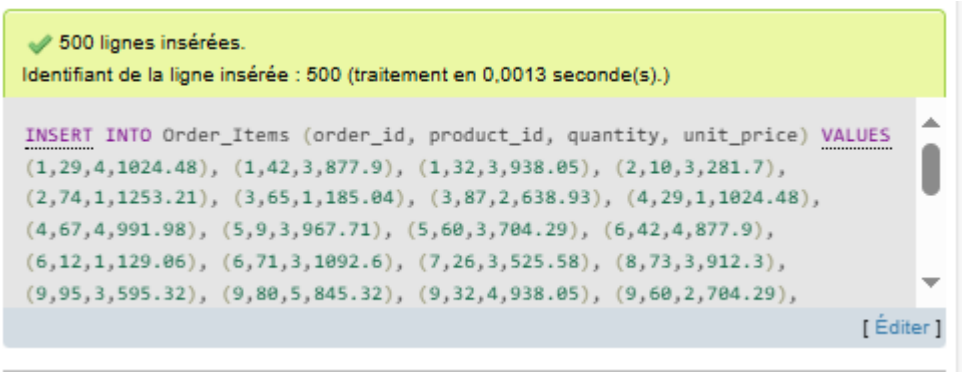
Explication :
Cette instruction INSERT INTO permet de remplir la table Reviews avec 150 enregistrements. Chaque avis associe un product_id à un customer_id, inclut une rating (note entre 1 et 5) et un commentaire facultatif. Les champs review_id et review_date sont générés automatiquement. Nous avons veillé à inclure une diversité de notes et de commentaires.

Command 8:insert data into orders table



Explication :
Cette instruction INSERT INTO remplit la table Orders avec 200 enregistrements. Chaque commande est associée à un customer_id, possède une order_date et un status choisi parmi les valeurs définies dans l'ENUM. Les commandes sont réparties entre différents clients et statuts, incluant certaines commandes « Pending » datant de plus de 3 jours, ce qui sera utile pour une requête ultérieure. Le champ order_id est généré automatiquement.

Command 9: insert data into order_item table



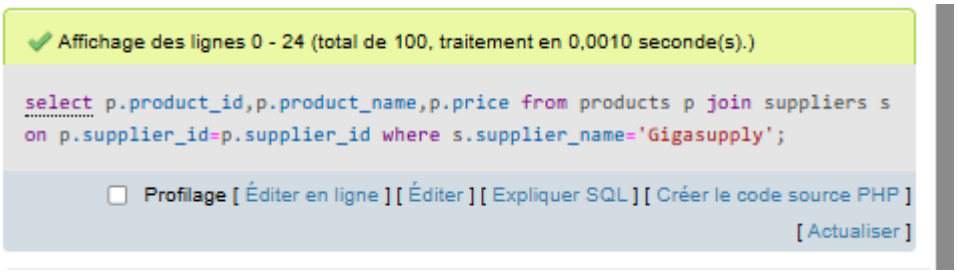
Explication :
Cette instruction INSERT INTO remplit la table Order_Items avec 500 enregistrements. Pour chaque enregistrement, nous associons un order_id à un product_id, précisons la quantity de ce produit dans la commande, et enregistrons le unit_price au moment de l'achat. Cela reflète fidèlement ce qui a été acheté dans chaque commande. Nous avons veillé à ce que la quantité soit supérieure à zéro, conformément à la contrainte CHECK.

Command 10: insert into data into payments table

Phase 3: queries.sql(DQL)

Filtration des données

1.Listez tous les produits fournis par ‘supplier x’ avec leur prix actuel



Affichage

Explication:
Cette requête vise à récupérer le nom du produit et son prix pour tous les produits dont le fournisseur est 'Gigasupply’.

2. Trouvez l’historique complet des commandes d’un client , y compris la date et le statut de la commande

Affichage:

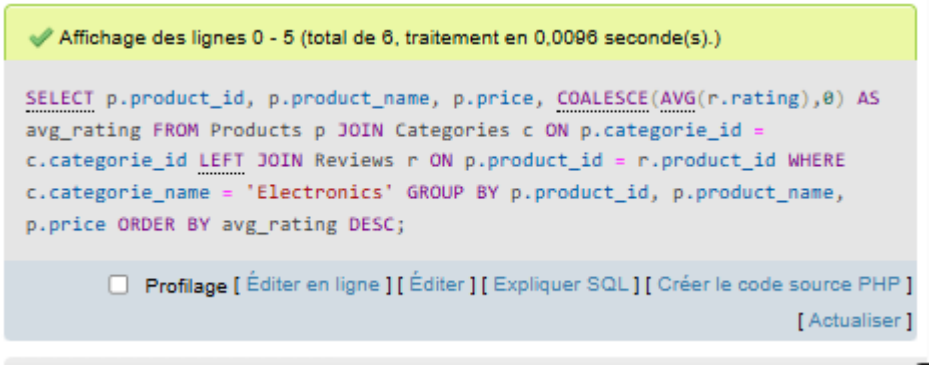
order_id	order_date 1	status	order_item_id	product_id	quantity	unit_price
24	2025-06-23 18:05:26	Processing	55	14	1	579.96
24	2025-06-23 18:05:26	Processing	56	20	2	1147.02
178	2025-06-01 04:32:26	Delivered	440	22	3	1025.70
178	2025-06-01 04:32:26	Delivered	441	16	1	761.43
175	2025-04-12 00:26:26	Processing	432	63	2	1275.12
175	2025-04-12 00:26:26	Processing	433	46	4	897.02
175	2025-04-12 00:26:26	Processing	434	97	1	1236.62
175	2025-04-12 00:26:26	Processing	435	25	2	1125.01

Explication:

Cette requête récupère tous les détails des articles commandés par un client spécifique, ainsi que les informations générales sur la commande.

product_id	product_name	price
1	Module 1 Plus	415.29
2	Widget 2 Plus	842.69
3	Gadget 3 Pro	601.18
4	Gadget 4 X	284.96
5	Device 5 Pro	1144.25
6	Widget 6 Lite	1085.52
7	Module 7 X	20.08
8	Module 8 Pro	1221.21
9	Gadget 9 Pro	967.71
10	Device 10 X	281.70
11	Module 11 Plus	1110.25
12	Device 12 Pro	129.06
13	Gadget 13 Pro	440.28
14	Widget 14 Plus	579.96
15	Gadget 15 Pro	107.05
16	Gadget 16 S	761.43
17	Widget 17 Pro	407.64
18	Widget 18 Plus	331.88
19	Gear 19 Lite	1266.24
20	Widget 20 Max	1147.02
21	Module 21 S	1019.27
22	Device 22 Max	1025.70
23	Gadget 23 Lite	296.93
24	Device 24 Max	297.56
25	Device 25 S	1125.01

3.Listez tous les produits de la catégorie ‘électronique ‘ avec leur évaluation moyenne

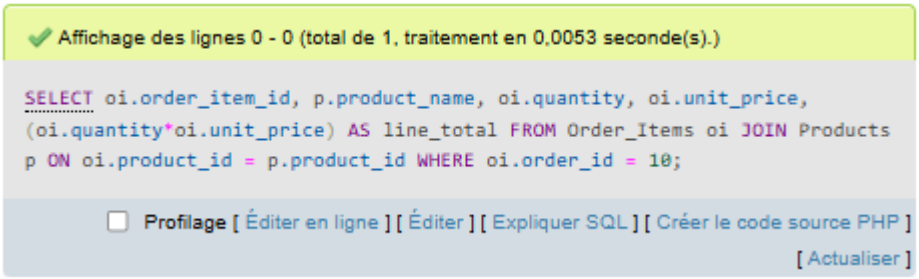


Affichage :

product_id	product_name	price	avg_rating ▾ 1
32	Gear 32 Plus	938.05	4.5000
87	Device 87 Max	638.93	4.0000
21	Module 21 S	1019.27	3.0000
92	Device 92 Max	531.67	2.3333
65	Module 65 Lite	185.04	1.0000
81	Gizmo 81 Pro	569.24	0.0000

Explication:
Cette requête liste les noms des produits de la catégorie 'Électronique' et calcule leur évaluation moyenne à partir des avis.

4.Afficher tous les articles pour une commande spécifique,en montrant le nom,quantité,prix totale de la ligne :

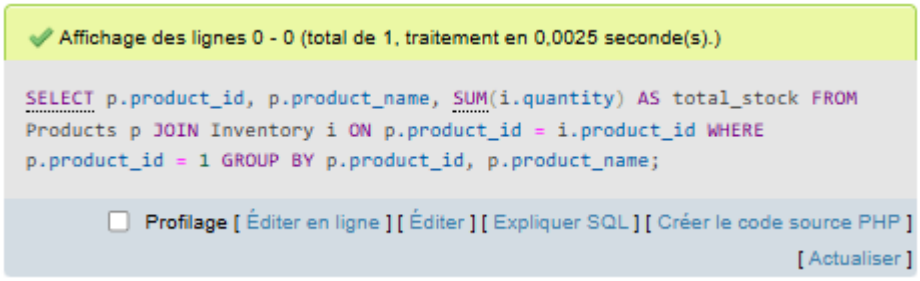


Affichage:

order_item_id	product_name	quantity	unit_price	line_total
21	Device 94 Lite	5	1295.64	6478.20

Explication:
Cette requête fournit les détails des articles d'une commande spécifique, calculant le coût total pour chaque ligne d'article.

5.Trouvez la quantité totale de stock pour un produit spécifique dans tous les entrepots.



Affichage :

product_id	product_name	total_stock
1	Module 1 Plus	661

Explication:

Cette requête calcule la somme de la quantité en stock d'un produit donné, en tenant compte de tous les entrepôts où il est stocké.

6.Identifies les clients qui ont acheté desproduits de plus de 3 catégories différentes:

costumer_id	costumer_name	category_count
1	Rachid El Amrani	5
2	Hassan Gharbi	5
3	Khalid Tazi	8
5	Mohamed Rifi	9
6	Sofia Bennis	13
8	Youness Ait Haddou	4
9	Nabil Azzouzi	10
10	Meryem Ait Haddou	7
11	Nabil Rifi	5
12	Omar Laamrani	14
13	Fatima Bouziane	11
14	Amina Bennis	9
16	Alaa Tazi	15
17	Mehdi Haddad	7
18	Ismail Zerouali	7
19	Walid Baraka	6
20	Fatima Zerouali	4
21	Fadoua Kabbaj	7
22	Rachid Gharbi	12
23	Meryem Zerouali	7
24	Nora Kabbaj	7
25	Kenza Kabbaj	8
26	Fatima Gharbi	8
27	Anas El Amrani	7
29	Marwa Bouazza	11

Affichage:

costumer_id	costumer_name	category_count
1	Rachid El Amrani	5
2	Hassan Gharbi	5
3	Khalid Tazi	8
5	Mohamed Rifi	9
6	Sofia Bennis	13
8	Youness Ait Haddou	4
9	Nabil Azzouzi	10
10	Meryem Ait Haddou	7
11	Nabil Rifi	5
12	Omar Laamrani	14
13	Fatima Bouziane	11
14	Amina Bennis	9
16	Alaa Tazi	15
17	Mehdi Haddad	7
18	Ismail Zerouali	7
19	Walid Baraka	6
20	Fatima Zerouali	4
21	Fadoua Kabbaj	7
22	Rachid Gharbi	12
23	Meryem Zerouali	7
24	Nora Kabbaj	7
25	Kenza Kabbaj	8
26	Fatima Gharbi	8
27	Anas El Amrani	7
29	Marwa Bouazza	11

Explication:

Cette requête identifie les clients ayant une diversité d'achats en termes de catégories de produits.

7.Identifies les produits qui on tune évaluation moyenne inférieur à 2 étoiles et qui ont été vendus au moins une fois

Affichage:

product_id	product_name	avg_rating	total_sold
8	Module 8 Pro	1.5000	64
16	Gadget 16 S	1.0000	14
20	Widget 20 Max	1.0000	9
40	Gear 40 Lite	1.0000	15
52	Device 52 Max	1.0000	14
54	Module 54 S	1.0000	26
60	Gizmo 60 Lite	1.0000	21
63	Gizmo 63 Max	1.0000	17
65	Module 65 Lite	1.0000	32
66	Gadget 66 S	1.0000	15
73	Widget 73 Lite	1.0000	22
88	Device 88 S	1.0000	9
89	Gadget 89 X	1.0000	14
95	Device 95 Plus	1.0000	19

Explication:
Cette requête trouve les produits qui sont mal notés et qui ont eu au moins une vente.

8.Calculer le revenue total généré par chaque catégorie de produits

Affichage des lignes 0 - 19 (total de 20, traitement en 0,0068 seconde(s).)

```
SELECT cat.categorie_id, cat.categorie_name, SUM(oi.quantity *
oi.unit_price) AS revenue FROM Categories cat JOIN Products p ON
cat.categorie_id = p.categorie_id JOIN Order_Items oi ON p.product_id =
oi.product_id GROUP BY cat.categorie_id, cat.categorie_name ORDER BY
revenue DESC;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP]
[Actualiser]

Affichage:

categorie_id	categorie_name	revenue 1
18	Car Electronics	115237.49
12	Networking	88184.06
9	Wearables	80951.34
4	Accessories	76312.58
8	Office	64689.60
14	Printers	63540.70
13	Storage	61260.46
20	Health	55257.10
15	Smart Home	50610.96
1	Electronics	48681.97
5	Audio	46445.40
7	Gaming	45729.89
11	TV & Video	41496.90
19	Beauty Tech	40734.79
17	Drones	34944.28
16	Fitness	26971.94
10	Cameras	26236.63
6	Home Appliances	24760.60
2	Computers	24216.16
3	Smartphones	10592.66

Explication:
Cette requête calcule le chiffre d'affaires total pour chaque catégorie de produits.

Trouver l’entrepot qui contient le plus grand nombre de produits uniques

La requête SQL a été exécutée avec succès.

```
SELECT i.warehouse_id, w.warehouse_location, COUNT(DISTINCT i.product_id)
AS unique_products FROM Inventory i JOIN Warehouses w ON i.warehouse_id =
w.warehouse_id GROUP BY i.warehouse_id, w.warehouse_location ORDER BY
unique_products DESC LIMIT 1;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP]
[Actualiser]

Affichage :

warehouse_id	warehouse_location	unique_products
1	Casablanca Warehouse, Casablanca	40

Explication:

Cette requête identifie quel entrepôt stocke la plus grande variété de produits.

10. lister les clients qui ont dépensé plus de 500$ et ont laisse au moins un avis 5 étoiles

Affichage des lignes 0 - 24 (total de 25, traitement en 0,0165 seconde(s).)

```
SELECT c.costumer_id, c.costumer_name, SUM(oi.quantity * oi.unit_price) AS
total_spent FROM Customers c JOIN Orders o ON c.costumer_id = o.costumer_id
JOIN Order_Items oi ON o.order_id = oi.order_id WHERE c.costumer_id IN (
SELECT customer_id FROM Reviews WHERE rating = 5 ) GROUP BY c.costumer_id,
c.costumer_name HAVING total_spent > 500;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP]
[Actualiser]

Affichage:

costumer_id	costumer_name	total_spent
1	Rachid El Amrani	16337.49
4	Imane Gharbi	634.61
5	Mohamed Rifi	24650.67
6	Sofia Bennis	27053.71
9	Nabil Azzouzi	32703.77
11	Nabil Rifi	11541.03
14	Amina Bennis	29218.88
15	Alaa Mansouri	4131.54
21	Fadoua Kabbaj	16219.08
22	Rachid Gharbi	51990.74
24	Nora Kabbaj	17814.51
26	Fatima Gharbi	29726.70
27	Anas El Amrani	20748.09
29	Marwa Bouazza	29640.66
31	Imane Chafai	9247.60
32	Rachid Oulhaj	21732.36
33	Ghita Oulhaj	5068.06
36	Yassine Azzouzi	9071.83
38	Fadoua Benali	20906.73
40	Soufiane Zerouali	14895.62
41	Omar Bouziane	44552.53
44	Youness Haddad	14216.37
47	Reda Kabbaj	7447.66
49	Rachid Azzouzi	18978.18
50	Khalid Zerouali	12339.28

Explication:

Cette requête identifie les clients les plus dépensiers qui sont aussi très satisfaits (ayant laissé un avis 5 étoiles).

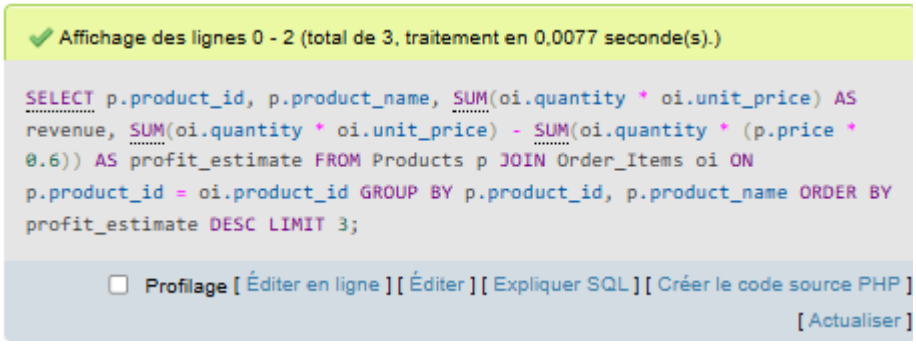
11.Calculez le revenue mensuel pour l'année en cours

Affichage :

month ▲ 1	revenue
2025-01	46212.63
2025-02	77511.83
2025-03	84691.85
2025-04	85998.60
2025-05	23337.19
2025-06	84119.77
2025-07	21255.47
2025-08	90465.68
2025-09	77763.25
2025-10	38162.87

Explication:
Cette requête fournit une vue d'ensemble des performances de vente mois par mois pour l'année en cours.

12.Trouver les 3 produits les plus rentables(revue total-cout total, en supposant que le cout représente 60% du prix)



Affichage:

product_id	product_name	revenue	profit_estimate ▼ 1
8	Module 8 Pro	39078.72	15631.488
29	Device 29 Plus	31758.88	12703.552
34	Device 34 Plus	29806.11	11922.444

Explication:
Cette requête identifie les produits qui génèrent le plus grand profit, en utilisant une formule de coût simple.

13.Trouver le client qui a passé le plus de commandes

La requête SQL a été exécutée avec succès.

```
SELECT c.costumer_id, c.costumer_name, COUNT(o.order_id) AS orders_count
FROM Customers c JOIN Orders o ON c.costumer_id = o.costumer_id GROUP BY
c.costumer_id, c.costumer_name ORDER BY orders_count DESC LIMIT 1;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP]
[Actualiser]

Affichage:

costumer_id	costumer_name	orders_count
45	Ghita Baraka	11

Explication:
Cette requête identifie le client le plus fidèle en termes de nombre de commandes passées.

14.Pour chaque fournisseur, listez le nombre de produits qu'il fournit et l'évaluation moyenne de ces produits

supplier_id	supplier_name	product_count ▾ 1	avg_rating
5	NorthStar Supplies	31	3.0000
4	GlobalParts	19	3.2353
6	PrimeTech	19	3.3333
2	TechSource	18	2.6250
7	ElectronixCo	16	3.3333
8	SmartLine	16	2.9333
10	Zenith Traders	16	3.0000
3	AlphaElectro	15	2.9333
1	GigaSupply	10	3.6250
9	Omega Supplies	10	2.8333

Affichage :

supplier_id	supplier_name	product_count ▾ 1	avg_rating
5	NorthStar Supplies	31	3.0000
4	GlobalParts	19	3.2353
6	PrimeTech	19	3.3333
2	TechSource	18	2.6250
7	ElectronixCo	16	3.3333
8	SmartLine	16	2.9333
10	Zenith Traders	16	3.0000
3	AlphaElectro	15	2.9333
1	GigaSupply	10	3.6250
9	Omega Supplies	10	2.8333

Explication:
Cette requête évalue chaque fournisseur en fonction du nombre de produits qu'il propose et de la satisfaction moyenne des clients pour ces produits.

15.Générer un rapport de toutes les commandes 'en attente' de plus de 3 jours , y compris les informations de cotat du client

Affichage des lignes 0 - 22 (total de 23, traitement en 0,0073 seconde(s).)

```
SELECT o.order_id, o.order_date, o.status, c.costumer_name, c.email,
c.shipping_adress FROM Orders o JOIN Customers c ON o.costumer_id =
c.costumer_id WHERE o.status = 'Pending' AND o.order_date < NOW() -
INTERVAL 3 DAY;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP]
[Actualiser]

Affichage:

order_id	order_date	status	costumer_name	email	shipping_adress
22	2024-11-05 08:46:26	Pending	Mohamed Rifi	mohamed.rifi4@gadgetworld.ma	179 Rue Example, Oujda
132	2025-06-30 03:43:26	Pending	Nabil Azzouzi	nabil.azzouzi8@gadgetworld.ma	50 Rue Example, Rabat
91	2025-04-20 10:04:26	Pending	Meryem Ait Haddou	meryem.ait9@gadgetworld.ma	60 Rue Example, Rabat
162	2025-09-07 17:58:26	Pending	Meryem Ait Haddou	meryem.ait9@gadgetworld.ma	60 Rue Example, Rabat
30	2025-08-20 07:01:26	Pending	Omar Laamrani	omar.laamrani11@gadgetworld.ma	163 Rue Example, Marrakech
116	2025-05-04 19:55:26	Pending	Omar Laamrani	omar.laamrani11@gadgetworld.ma	163 Rue Example, Marrakech
6	2025-04-26 00:36:26	Pending	Amina Bennis	amina.bennis13@gadgetworld.ma	197 Rue Example, Casablanca
13	2025-04-07 00:08:26	Pending	Amina Bennis	amina.bennis13@gadgetworld.ma	197 Rue Example, Casablanca
96	2025-09-20 12:27:26	Pending	Fatima Zerouali	fatima.zerouali19@gadgetworld.ma	153 Rue Example, Rabat
87	2025-03-07 07:34:26	Pending	Rachid Gharbi	rachid.gharbi21@gadgetworld.ma	193 Rue Example, Tanger
148	2024-10-29 09:32:26	Pending	Rachid Gharbi	rachid.gharbi21@gadgetworld.ma	193 Rue Example, Tanger
158	2025-08-12 20:50:26	Pending	Rachid Gharbi	rachid.gharbi21@gadgetworld.ma	193 Rue Example, Tanger
167	2025-03-07 23:44:26	Pending	Meryem Zerouali	meryem.zerouali22@gadgetworld.ma	41 Rue Example, Kenitra
153	2024-11-06 14:24:26	Pending	Nora Kabbaj	nora.kabbaj23@gadgetworld.ma	196 Rue Example, Marrakech
155	2024-11-12 07:49:26	Pending	Salma Azzouzi	salma.azzouzi27@gadgetworld.ma	21 Rue Example, Rabat
56	2025-04-12 02:28:26	Pending	Marwa Bouazza	marwa.bouazza28@gadgetworld.ma	33 Rue Example, Kenitra
26	2024-11-09 19:33:26	Pending	Imane Chafai	imane.chafai30@gadgetworld.ma	103 Rue Example, Agadir
120	2025-07-20 07:58:26	Pending	Rachid Oulhaj	rachid.oulhaj31@gadgetworld.ma	58 Rue Example, Rabat
157	2024-10-29 20:34:26	Pending	Ibrahim Oulhaj	ibrahim.oulhaj33@gadgetworld.ma	18 Rue Example, Casablanca
21	2024-11-19 09:01:26	Pending	Salma Haddad	salma.haddad34@gadgetworld.ma	172 Rue Example, Kenitra
38	2025-08-25 17:36:26	Pending	Hassan El Idrissi	hassan.el36@gadgetworld.ma	169 Rue Example, Oujda
110	2024-12-15 07:51:26	Pending	Soufiane Zerouali	soufiane.zerouali39@gadgetworld.ma	47 Rue Example, Tanger
184	2024-11-14 12:53:26	Pending	Rachid Azzouzi	rachid.azzouzi48@gadgetworld.ma	64 Rue Example, Settat

Explication:
Cette requête est utile pour identifier les commandes "en attente" qui nécessitent une action car elles n'ont pas été traitées rapidement.

Phase 4:modifications.sql(DML)

Update:une livraison de 50 unités du ‘produit y’ arrive à ‘ l’entrepot z’,mettez à jour l’inventaire

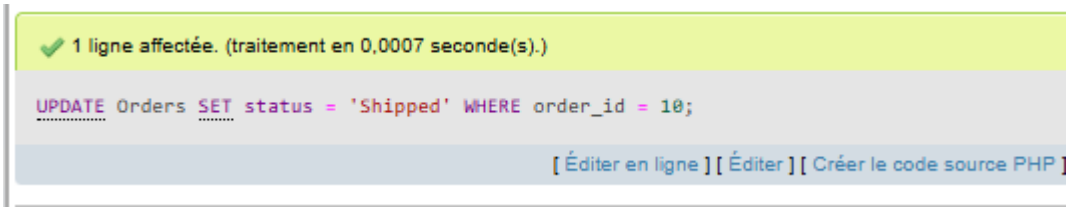
1 ligne affectée. (traitement en 0,0082 seconde(s).)

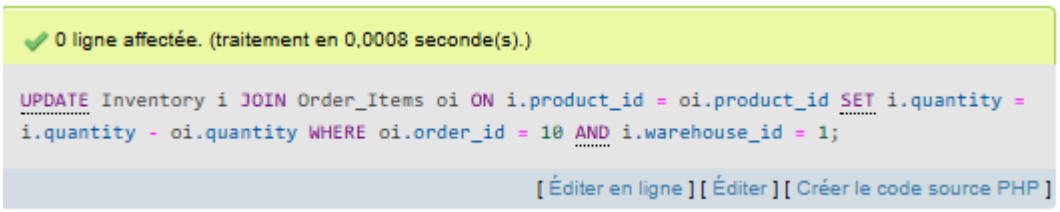
```
UPDATE Inventory SET quantity = quantity + 50 WHERE product_id = 5 AND warehouse_id =
2;
```

[Éditer en ligne] [Éditer] [Créer le code source PHP

Explication:
Cette requête `UPDATE` est utilisée pour refléter l'arrivée d'une nouvelle livraison en augmentant la quantité de stock d'un produit spécifique dans un entrepôt donné.

Update:une commande a été expédiée. Mettez à jour le statut de la commande et diminuez la quantité de stock pour chaque article de la commande depuis l'entrepot approprié

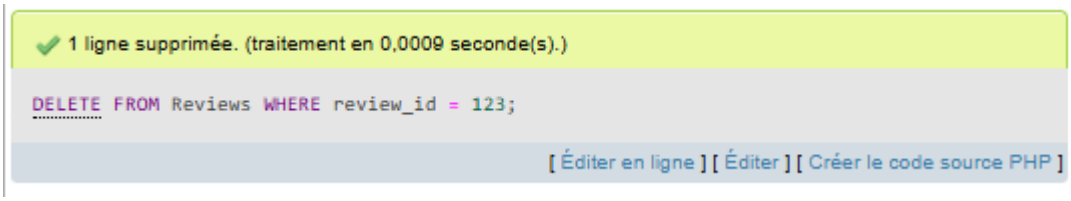




Explication:
Cette tâche nécessite deux étapes logiques :

1. Mettre à jour le statut de la commande à 'Shipped' (Expédiée).
2. Pour chaque article de cette commande, réduire le stock dans l'entrepôt concerné.

3. delete:l'avis d'un client a été signal comme spam et doit étre supprimé



Explication:
Cette requête DELETE est utilisée pour supprimer un enregistrement spécifique de la table Reviews sur la base de son identifiant.

Projet 2:

social media startup

ConnectSphere

Phase 1:Initialisation de la base de données

Création des Collection :

```
jobintech> db.createCollection("users");
           db.createCollection("groups");
           db.createCollection("posts");
           db.createCollection("comments");
           db.createCollection("notifications");
```

```
< switched to db jobintech
> db.users.drop()
  db.groups.drop()
  db.posts.drop()
  db.comments.drop()
  db.notifications.drop()
< true
```

Creation seccées :

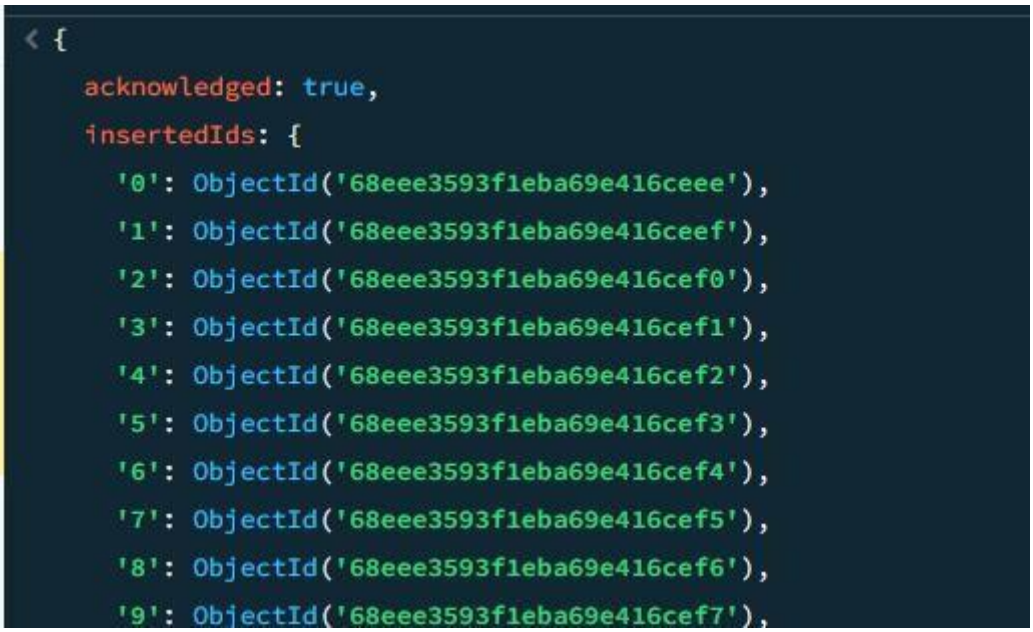
```
< { ok: 1 }
```

Phase 2 : Création des Ce fichier doit contenir des commandes `insertMany()` supplémentaires pour créer un ensemble de données riche et interconnecté.

Création de 50 utilisateurs initiaux dans la Collection Users :

Création avec succées:

```
jobintech> db.users.insertMany([
  {
    username: "Benzerouala Ossama",
    email: "benzeroualaossama@gmail.com",
    join_date: ISODate("2024-01-15T10:30:00Z"),
    profile: {
      bio: "Développeur passionné de Casablanca MA",
      location: "Casablanca, Maroc"
    },
    followers: [],
    following: [],
    groups: []
  },
  {
    username: "mohammedaitmed",
    email: "mohammedaitmed@gmail.com",
    join_date: ISODate("2024-02-10T14:20:00Z"),
    profile: {
      bio: "Designer UI/UX | Amoureuse de l'art marocain",
      location: "Rabat, Maroc"
    },
    followers: [],
    following: [],
    groups: []
```



Création du 10 groupes de base :

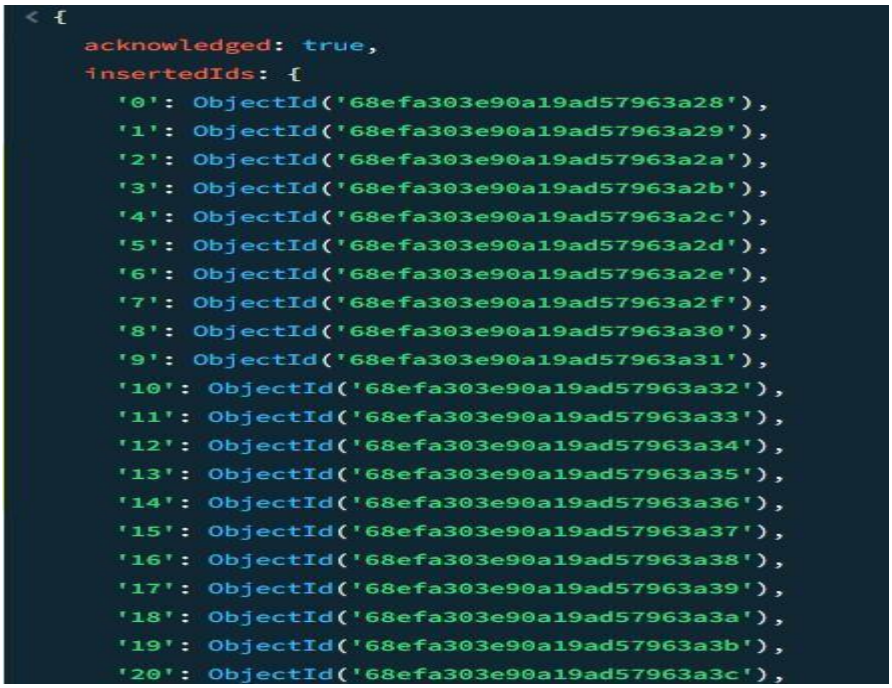
```
jobintech> db.groups.insertMany([
  {
    group_name: "Développeurs Digital",
    description: "Communauté des développeurs marocains - Partage d'expériences et opportunités",
    created_by: ObjectId("507f1f77bcf86cd799439011"),
    members: []
  },
  {
    group_name: "Cuisine Marocaine",
    description: "Recettes traditionnelles et modernes de la gastronomie marocaine",
    created_by: ObjectId("507f1f77bcf86cd799439012"),
    members: []
  },
  {
    group_name: "Gaming Maroc",
    description: "Communauté des gamers marocains - tournois, nouveautés et partages d'expériences",
    created_by: ObjectId("507f1f77bcf86cd799439026"),
    members: []
  },
  {
    group_name: "Photographie Maroc",
    description: "Espace pour les passionnés de photographie au Maroc - partage de techniques et d'œuvres",
    created_by: ObjectId("507f1f77bcf86cd799439013"),
    members: []
```

Création des groups avec sucées :

Création 200 posts initiaux :

```
jobintech> db.posts.insertMany([
            {
              author_id: ObjectId("507f1f77bcf86cd799439011"),
              content: "Premier projet avec MongoDB réussi! #mongodb #nosql",
              post_type: "text",
              image_url: null,
              group_id: null,
              timestamp: ISODate("2024-10-14T18:30:00Z"),
              likes: [],
              tags: ["Casablanca", "Maroc"]
            },
            {
              author_id: ObjectId("507f1f77bcf86cd799439012"),
              content: "Nouveau projet MongoDB en cours! Les bases NoSQL sont vraiment puissantes 💪 #mongodb #nosql",
              post_type: "text",
              image_url: null,
              group_id: null,
              timestamp: ISODate("2024-10-15T09:15:00Z"),
              likes: [],
              tags: ["mongodb", "nosql"]
            },
            {
              author_id: ObjectId("507f1f77bcf86cd799439013"),
              content: "Performance optimisation avec MongoDB indexes #mongodb #performance",
```

Creation des posts avec succées



Creation des 500 Comments :

```
jobintech> db.comments.insertMany([
  {
    post_id: ObjectId("507f1f77bcf86cd799439021"),
    author_id: ObjectId("507f1f77bcf86cd799439012"),
    parent_comment_id: null,
    text: "Magnifique photo! J'adore Casablanca 😍",
    timestamp: ISODate("2024-10-14T19:00:00Z")
  },
  {
    post_id: ObjectId("507f1f77bcf86cd799439022"),
    author_id: ObjectId("507f1f77bcf86cd799439013"),
    parent_comment_id: null,
    text: "MongoDB est vraiment génial pour les projets scalables!",
    timestamp: ISODate("2024-10-15T09:30:00Z")
  },
  {
    post_id: ObjectId("507f1f77bcf86cd799439023"),
    author_id: ObjectId("507f1f77bcf86cd799439014"),
    parent_comment_id: null,
    text: "La médina de Fès est un trésor national 🇲🇦",
    timestamp: ISODate("2024-10-13T15:00:00Z")
  },
  {
    post_id: ObjectId("507f1f77bcf86cd799439024"),
```

Creation des comments avec succées :

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('68eeeb913f1eba69e416cfa7'),
    '1': ObjectId('68eeeb913f1eba69e416cfa8'),
    '2': ObjectId('68eeeb913f1eba69e416cfa9'),
    '3': ObjectId('68eeeb913f1eba69e416cfaa'),
    '4': ObjectId('68eeeb913f1eba69e416cfab'),
    '5': ObjectId('68eeeb913f1eba69e416cfac'),
    '6': ObjectId('68eeeb913f1eba69e416cfad'),
    '7': ObjectId('68eeeb913f1eba69e416cfae'),
    '8': ObjectId('68eeeb913f1eba69e416cfaf'),
    '9': ObjectId('68eeeb913f1eba69e416cfb0')
```

Phase 3: Requêtes NoSQL avancées

1. Find the complete profile of a user, and using `$lookup`,also retrieve the full document of the users they are following .

```
jobintech> db.users.aggregate([
            {
              $match: { username: "Benzerouala ossama" }
            },
            {
              $lookup: {
                from: "users",localField: "following",foreignField: "_id",
                as: "following_details"
              }
            },
            {
              $project: {
                username: 1,email: 1,profile: 1,
                join_date: 1,followers: 1,following: 1,
                groups: 1,
                following_details: {
                  username: 1,
                  email: 1,
                  "profile.bio": 1,
                  "profile.location": 1
                }
              }
            }
          ])
```

L'affichage des résultats

2. List all posts made within a specific group:

```
jobintech> db.posts.find({
             group_id: ObjectId("68eee4653f1eba69e416cf1e")
           }).sort({ timestamp: -1 })
```

The result :

```
{
  _id: ObjectId('68eeeaff3f1eba69e416cf9d'),
  author_id: ObjectId('507f1f77bcf86cd799439011'),
  content: 'Magnifique coucher de soleil sur la corniche de Casablanca 🌅 #Casablanca #Maroc',
  post_type: 'text',
  image_url: null,
  group_id: ObjectId('68eee4653f1eba69e416cf1e'),
  timestamp: 2024-10-14T18:30:00.000Z,
  likes: [],
  tags: [
    'Casablanca',
    'Maroc'
  ]
}
{
  _id: ObjectId('68eeeaff3f1eba69e416cfa2'),
  author_id: ObjectId('507f1f77bcf86cd799439016'),
  content: 'Session de surf incroyable à Taghazout ce matin 🏄 #Surf #Agadir',
  post_type: 'image',
  image_url: 'https://example.com/surf_agadir.jpg',
  group_id: ObjectId('68eee4653f1eba69e416cf1e'),
  timestamp: 2024-10-14T10:30:00.000Z,
  likes: [],
  tags: [
```

3. For a specific post, retrieve all its parent comments. Then, for a specific parent comment, retrieve all of its replies:

- *All parent comments of a post :*

```
jobintech> db.comments.find({
             post_id: ObjectId("507f1f77bcf86cd799439021"),
             parent_comment_id: null
           }).sort({ timestamp: -1 })
```

Result :

```
{
  _id: ObjectId('68eeed583f1eba69e416d1c9'),
  post_id: ObjectId('507f1f77bcf86cd799439022'),
  author_id: ObjectId('68eeed583f1eba69e416d141'),
  parent_comment_id: null,
  text: "J'ai commencé MongoDB il y a 2 mois, c'est génial",
  timestamp: 2024-10-15T11:00:00.000Z
}
{
  _id: ObjectId('68eeeb913f1eba69e416cfa8'),
  post_id: ObjectId('507f1f77bcf86cd799439022'),
  author_id: ObjectId('507f1f77bcf86cd799439013'),
  parent_comment_id: null,
  text: 'MongoDB est vraiment génial pour les projets scalables!',
  timestamp: 2024-10-15T09:30:00.000Z,
  group_id: ObjectId('507f1f77bcf86cd799439022')
}
{
  _id: ObjectId('68eeeb913f1eba69e416cfa7'),
  post_id: ObjectId('507f1f77bcf86cd799439022'),
  author_id: ObjectId('507f1f77bcf86cd799439012'),
  parent_comment_id: null,
  text: "Magnifique photo! J'adore Casablanca 😍",
  timestamp: 2024-10-14T19:00:00.000Z,
```

➢ *All replies from a parent comment:*

```
{
  _id: ObjectId('68eeed583f1eba69e416d1e1'),
  post_id: ObjectId('68eeed583f1eba69e416d175'),
  author_id: ObjectId('68eeed583f1eba69e416d176'),
  parent_comment_id: ObjectId('507f1f77bcf86cd799439031'),
  text: 'Dima Raja! 🌍',
  timestamp: 2024-10-14T18:30:00.000Z
}
{
  _id: ObjectId('68eeeb913f1eba69e416cfa8'),
  post_id: ObjectId('507f1f77bcf86cd799439022'),
  author_id: ObjectId('507f1f77bcf86cd799439013'),
  parent_comment_id: ObjectId('507f1f77bcf86cd799439031'),
  text: 'MongoDB est vraiment génial pour les projets scalables!',
  timestamp: 2024-10-15T09:30:00.000Z,
  group_id: ObjectId('507f1f77bcf86cd799439022')
}
```

4. Find all posts from the last 24 hours that contain the tag '#mongodb'.

```
jobintech> db.comments.find({
             parent_comment_id: ObjectId("507f1f77bcf86cd799439031")
           }).sort({ timestamp: 1 })
jobintech> db.posts.find({
             tags: "mongodb",
             timestamp: {
               $gte: ISODate("2024-10-14T15:00:00Z")
             }
           }).sort({ timestamp: -1 })
```

```
{
  _id: ObjectId('68eeecc43f1eba69e416d115'),
  author_id: ObjectId('68eeecc43f1eba69e416d02a'),
  content: 'Tutoriel: Comment utiliser $lookup dans MongoDB #mongodb #tutorial',
  post_type: 'text',
  image_url: null,
  group_id: null,
  timestamp: 2024-10-15T11:30:00.000Z,
  likes: [],
  tags: [
    'mongodb',
    'tutorial'
  ]
}
{
  _id: ObjectId('68eeecc43f1eba69e416d114'),
  author_id: ObjectId('68eeecc43f1eba69e416d029'),
  content: 'Apprentissage de MongoDB! Les agrégations sont puissantes 💪 #mongodb #database',
  post_type: 'text',
  image_url: null,
  group_id: null,
  timestamp: 2024-10-15T10:00:00.000Z,
  likes: [],
```

```
  author_id: ObjectId('68eeecc43f1eba69e416d029'),
  content: 'Apprentissage de MongoDB! Les agrégations sont puissantes 💪 #mongodb #database
  post_type: 'text',
  image_url: null,
  group_id: null,
  timestamp: 2024-10-15T10:00:00.000Z,
  likes: [],
  tags: [
    'mongodb',
    'database'
  ]
}
{
  _id: ObjectId('68eeecc43f1eba69e416d116'),
  author_id: ObjectId('68eeecc43f1eba69e416d02b'),
  content: 'MongoDB Atlas est incroyable pour le cloud! #mongodb #cloud',
  post_type: 'text',
  image_url: null,
  group_id: null,
  timestamp: 2024-10-15T09:45:00.000Z,
  likes: [],
  tags: [
    'mongodb',
    'cloud'
```

5. Generate an Advanced User Feed: For a given user, find all posts from users they `follow` AND all posts from `groups` they are a member of, sort them all by the most recent `timestamp`, and limit to 20.

```
jobintech> db.users.aggregate([
            {
              $match: { username: "Benzerouala Ossama" }
            },
            {
              $lookup: {
                from: "posts",
                let: { following: "$following", groups: "$groups" },
                pipeline: [
                  {
                    $match: {
                      $expr: {
                        $or: [
                          { $in: ["$author_id", "$$following"] },
                          { $in: ["$group_id", "$$groups"] }
                        ]}},
                  { $sort: { timestamp: -1 } },
                  { $limit: 20 }
                ],
                as: "feed"
              }},
            {
              $project: {username: 1,feed: 1}}])
```

```
{
  _id: ObjectId('68eee3593f1eba69e416ceee'),
  username: 'Benzerouala Ossama',
  feed: [
    {
      _id: ObjectId('68eeeaff3f1eba69e416cf9d'),
      author_id: ObjectId('507f1f77bcf86cd799439011'),
      content: 'Magnifique coucher de soleil sur la corniche de Casablanca 🌅 #Casablanca #Maroc',
      post_type: 'text',
      image_url: null,
      group_id: ObjectId('68eee4653f1eba69e416cf1e'),
      timestamp: 2024-10-14T18:30:00.000Z,
      likes: [],
      tags: [
        'Casablanca',
        'Maroc'
      ]
    },
    {
      _id: ObjectId('68eeeaff3f1eba69e416cfa2'),
      author_id: ObjectId('507f1f77bcf86cd799439016'),
      content: 'Session de surf incroyable à Taghazout ce matin 🏄 #Surf #Agadir',
      post_type: 'image',
      image_url: 'https://example.com/surf_agadir.jpg',
```

6. Identify users who are members of a group but have never posted in it.

```
jobintech> db.groups.aggregate([
            { $match: { group_name: "Développement Personnel Maroc" } },{
              $lookup: {
                from: "posts",
                let: { group_id: "$_id", members: "$members" },
                pipeline: [
                  {$match: {
                      $expr: {
                        $and: [
                          { $eq: ["$group_id", "$$group_id"] },
                          { $in: ["$author_id", "$$members"] }] } } }
                ],
                as: "group_posts"}},{
              $project: {
                inactive_members: {
                  $setDifference: ["$members", "$group_posts.author_id"]}}},{
              $lookup: {
                from: "users",
                localField: "inactive_members",
                foreignField: "_id",
                as: "inactive_users"
              }},{
              $project: {
                "inactive_users.username": 1,"inactive_users.email": 1 }}])
```

```
< {
    _id: ObjectId('68eee4653f1eba69e416cf1e'),
    inactive_users: [
      {
        username: 'Benzerouala Ossama',
        email: 'benzeroualaossama@gmail.com'
      }
    ]
  }
  {
    _id: ObjectId('68eee5323f1eba69e416cf2a'),
    inactive_users: []
  }
```

7: Posts avec plus de 10 likes mais zéro commentaires :

```
jobintech> db.posts.aggregate([
            {
              $match: {
                $expr: { $gte: [{ $size: "$likes" }, 10] }
              }
            },
            {
              $lookup: {
                from: "comments",
                localField: "_id",
                foreignField: "post_id",
                as: "comments"
              }}, {
              $match: {
                comments: { $size: 0 }
              }},
            {
              $project: {
                content: 1,
                author_id: 1,
                likes_count: { $size: "$likes" },
                timestamp: 1
              }}])
```

```
  _id: ObjectId('68eeecc43f1eba69e416d119'),
  author_id: ObjectId('68eeecc43f1eba69e416d02e'),
  content: 'Magnifique lever de soleil sur le désert de Merzouga 🌅',
  timestamp: 2024-10-10T06:30:00.000Z,
  likes_count: 12
}
{
  _id: ObjectId('68eeecc43f1eba69e416d11e'),
  author_id: ObjectId('68eeecc43f1eba69e416d052'),
  content: 'Les couleurs de Chefchaouen sont magiques',
  timestamp: 2024-10-09T10:45:00.000Z,
  likes_count: 11
}
{
  _id: ObjectId('68eeecc43f1eba69e416d12b'),
  author_id: ObjectId('68eeecc43f1eba69e416d0a7'),
  content: 'Architecture mauresque de la mosquée Hassan II',
  timestamp: 2024-10-08T15:30:00.000Z,
  likes_count: 10
}
```

8. Count the number of members in each group:

```
jobintech> db.groups.aggregate([
            {
              $project: {
                group_name: 1,
                members_count: { $size: "$members" }
              }
            },
            {
              $sort: { members_count: -1 }
            }
          ])
```

```
< {
    _id: ObjectId('68eee4653f1eba69e416cf1e'),
    group_name: 'Développement Personnel Maroc',
    members_count: 5
  }
  {
    _id: ObjectId('68eee4653f1eba69e416cf1b'),
    group_name: 'Gaming Maroc',
    members_count: 3
  }
  {
    _id: ObjectId('68eee4653f1eba69e416cf20'),
    group_name: 'Mode & Beauté Maroc',
    members_count: 2
  }
  {
    _id: ObjectId('68eee4653f1eba69e416cf1c'),
    group_name: 'Cinéma & Séries Maroc',
    members_count: 0
  }
```

9. Generate a list of unread notifications for a user:

```
jobintech> db.notifications.find({
             recipient_id: ObjectId("68eee3593f1eba69e416cef0"),
             is_read: false
           }).sort({ timestamp: -1 })
```

10. List the top 10 most influential users, defined as those with the most followers:

```
< {
    _id: ObjectId('68eee3593f1eba69e416cef0'),
    username: 'Maryemayad',
    email: 'Maryemayad.com',
    profile: {
      location: 'Marrakech, Maroc'
    },
    followers_count: 3
  }
  {
    _id: ObjectId('68eee3593f1eba69e416ceee'),
    username: 'Benzerouala Ossama',
    email: 'benzeroualaossama@gmail.com',
    profile: {
      location: 'Casablanca, Maroc'
    },
    followers_count: 2
  }
  {
    _id: ObjectId('68eee3593f1eba69e416cef6'),
    username: 'rachid_meknes',
    email: 'rachid.lahlou@gmail.com',
    profile: {
      location: 'Meknès, Maroc'
```

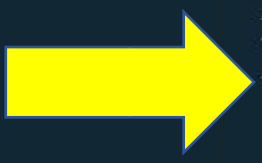
11. Calculate the average number of likes per post for each user

```
jobintech> db.posts.aggregate([
            {$group: {
                _id: "$author_id",
                total_posts: { $sum: 1 },
                total_likes: { $sum: { $size: "$likes" } }}},{
              $project: {
                total_posts: 1,total_likes: 1,average_likes: {
                  $cond: {
                    if: { $eq: ["$total_posts", 0] },
                    then: 0,
                    else: { $divide: ["$total_likes", "$total_posts"] } } }}
            },{ $lookup: {
                from: "users",
                localField: "_id",
                foreignField: "_id",
                as: "user"}},{
              $unwind: "$user"
            },{
              $project: {
                username: "$user.username",
                total_posts: 1,
                total_likes: 1,
                average_likes: { $round: ["$average_likes", 2] } }},
            {$sort: { average_likes: -1 }}])
```

```
< {
    _id: ObjectId('68eee3593f1eba69e416cef0'),
    total_posts: 3,
    total_likes: 10,
    username: 'Maryemayad',
    average_likes: 3.33
  }
  {
    _id: ObjectId('68eee3593f1eba69e416ceef'),
    total_posts: 2,
    total_likes: 6,
    username: 'mohammedaitmed',
    average_likes: 3
  }
```

12. Find the 'trending' tags of the day by counting tag occurrences in posts from the last 24 hours:

```
jobintech> db.posts.aggregate([
  {
    $match: {
      timestamp: {
        $gte: ISODate("2024-10-14T15:00:00Z")
      }
    }
  },{
    $unwind: "$tags"
  },{
    $group: {
      _id: "$tags",
      count: { $sum: 1 }
    }
  }, {
    $sort: { count: -1 }
  },{
    $limit: 10
  },{
    $project: {
      tag: "$_id",
      occurrences: "$count",
      _id: 0 }}])
```



```
{
  tag: 'mongodb',
  occurrences: 6
}
{
  tag: 'thé',
  occurrences: 2
}
{
  tag: 'nosql',
  occurrences: 2
}
{
  tag: 'test',
  occurrences: 2
}
{
  tag: 'StartupMaroc',
  occurrences: 1
}
{
  tag: 'tradition',
  occurrences: 1
}
```

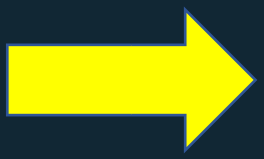
13. Find the most active user in a specific group (most posts + comments):

```
jobintech> db.groups.aggregate([
        {$match: { group_name: "Développement Personnel Maroc" }},
        {$lookup: {
            from: "posts",let: { group_id: "$_id" },pipeline: [
              {$match: {$expr: { $eq: ["$group_id", "$$group_id"] }}
              },{$group: {
                  _id: "$author_id",
                  posts_count: { $sum: 1 }}],
            as: "posts_by_user"
          }
        },{
          $lookup: {
            from: "posts",
            let: { group_id: "$_id" },
            pipeline: [
              {
                $match: {
                  $expr: { $eq: ["$group_id", "$$group_id"] }
                }},{
                $lookup: {
                  from: "comments",
                  localField: "_id",
                  foreignField: "post_id",
                  $unwind: "$comments"
                },{
                  $group: {
                    _id: "$comments.author_id",
                    comments_count: { $sum: 1 } }}],
              as: "comments_by_user"}},{
            $project: {
              activity: {
                $concatArrays: ["$posts_by_user", "$comments_by_user"]}}},{
            $unwind: "$activity"},{
            $group: {
              _id: "$activity._id",
              total_posts: { $sum: { $ifNull: ["$activity.posts_count", 0] } },
              total_comments: { $sum: { $ifNull: ["$activity.comments_count", 0] } } } },{
            $project: {
              total_posts: 1,
              total_comments: 1,
              total_activity: { $add: ["$total_posts", "$total_comments"] }}},{
            $sort: { total_activity: -1 }},{
            $limit: 1},{
            $lookup: {
              from: "users",localField: "_id",foreignField: "_id",as: "user"}},{
            $unwind: "$user"},{
            $project: {username: "$user.username",email: "$user.email",total_posts: 1,total_comments: 1,total_activity: 1}
```

```
< {
    _id: ObjectId('68eee3593f1eba69e416cef0'),
    total_posts: 2,
    total_comments: 0,
    total_activity: 2,
    username: 'Maryemayad',
    email: 'Maryemayad.com'
  }
```

14. Generate a report showing the daily user registration count for the last 30 days.

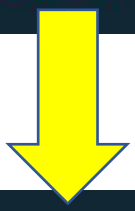
```
jobintech> db.users.aggregate([
            {
              $match: {
                join_date: {
                  $gte: ISODate("2024-09-15T00:00:00Z")
                }}},{
              $project: {
                date: {
                  $dateToString: {
                    format: "%Y-%m-%d",
                    date: "$join_date"
                  } }},{
              $group: {
                _id: "$date",
                registrations: { $sum: 1 }
              },{
              $sort: { _id: 1 }
            },
            {
              $project: {
                date: "$_id",
                registrations: 1,
                _id: 0
              }])
```



```
[
  {
    registrations: 3,
    date: '2024-09-15'
  }
  {
    registrations: 3,
    date: '2024-09-18'
  }
  {
    registrations: 3,
    date: '2024-09-20'
  }
  {
    registrations: 3,
    date: '2024-09-22'
  }
  {
    registrations: 3,
    date: '2024-09-25'
  }
  {
    registrations: 3,
    date: '2024-09-28'
  }
```

15. For a user's profile, aggregate their total post count, total likes received across all their posts, and their follower count.

```
jobintech> db.posts.aggregate([{ $group: {
                _id: "$author_id",
                total_posts: { $sum: 1 },
                total_likes: {
                  $sum: {
                    $size: {
                      $ifNull: ["$likes", []]   } } }},{
              $project: {
                total_posts: 1, total_likes: 1,
                average_likes: {
                  $cond: {
                    if: { $eq: ["$total_posts", 0] },
                    then: 0,
                    else: { $divide: ["$total_likes", "$total_posts"] }}}}}{
              $lookup: {
                from: "users",localField: "_id",
                foreignField: "_id",
                as: "user"}},
             { $unwind: "$user" },{
                $project: {
                  username: "$user.username", total_posts: 1,total_likes: 1,
                  average_likes: { $round: ["$average_likes", 2] }
                }},{ $sort: { average_likes: -1 } }]);
```



```
< {
    _id: ObjectId('68eee3593f1eba69e416cef0'),
    total_posts: 3,
    total_likes: 10,
    username: 'Maryemayad',
    average_likes: 3.33
  }
  {
    _id: ObjectId('68eee3593f1eba69e416ceef'),
    total_posts: 2,
    total_likes: 6,
    username: 'mohammedaitmed',
    average_likes: 3
  }
  {
    _id: ObjectId('68eee3593f1eba69e416ceee'),
    total_posts: 2,
    total_likes: 0,
    username: 'Benzerouala Ossama',
    average_likes: 0
  }
```

Phase 4: `modifications.js` (Updates)

1. A user follows another: use `$addToSet` on both users' documents to update `following` and `followers` arrays. Then, create a `notification` document.

Step 1a: Add in following of the following user:

```
jobintech> db.users.updateOne(
            { username: "Benzerouala Ossama" },
            {
              $addToSet: {
                following: ObjectId("68eee3593f1eba69e416cef0")
              }
            }
          )
```

```
{
  following: [
    ObjectId('68eee3593f1eba69e416cef0')
  ]
}
```

```
jobintech> db.users.find(
            { username: "Benzerouala Ossama" },
            { following: 1, _id: 0 }
          );
```

```
jobintech> db.users.aggregate([
            {
              $match: { username: "Benzerouala Ossama" }
            },
            {
              $lookup: {
                from: "users",
                localField: "following",
                foreignField: "_id",
                as: "following_users"
              }
            },
            {
              $project: {
                _id: 0,
                username: 1,
                "following_users.username": 1,
                "following_users.email": 1
              }
            }
          ]);
```

```
{
  username: 'Benzerouala Ossama',
  following_users: [
    {
      username: 'Maryemayad',
      email: 'Maryemayad.com'
    }
  ]
}
```

1b: Add to the followers of the followed user :

```
jobintech> db.users.updateOne(
            { _id: ObjectId("68eee3593f1eba69e416ceee") },
            {
              $addToSet: {
                followers: ObjectId("68eee3593f1eba69e416ceef")
              }
            }
          )
```

```
{
  followers: [
    ObjectId('68eee3593f1eba69e416ceef'),
    ObjectId('68eee3593f1eba69e416cef0')
  ]
}
```

```
jobintech> db.users.find(
             { _id: ObjectId("68eee3593f1eba69e416ceee") },
             { followers: 1, _id: 0 }
           );
```

```
jobintech> db.users.aggregate([
            {
              $match: { _id: ObjectId("68eee3593f1eba69e416ceee") }
            },
            {
              $lookup: {
                from: "users",
                localField: "followers",
                foreignField: "_id",
                as: "followers_info"
              }
            },
            {
              $project: {
                _id: 0,
                username: 1,
                "followers_info.username": 1,
                "followers_info.email": 1
              }
            }
          ]);
```

```
{
  username: 'Benzerouala Ossama',
  followers_info: [
    {
      username: 'mohammedaitmed',
      email: 'mohammedaitmed@gmail.com'
    },
    {
      username: 'Maryemayad',
      email: 'Maryemayad.com'
    }
  ]
}
```

1c: Create a notification for tracking

```
jobintech> db.notifications.insertOne({
             recipient_id: ObjectId("68eee3593f1eba69e416ceee"),
             sender_id: ObjectId("68eee3593f1eba69e416ceef"),
             type: "follow",
             post_id: null,
             is_read: false,
             timestamp: ISODate("2024-10-15T14:30:00Z")
           })
```

```
< {
    acknowledged: true,
    insertedId: ObjectId('68ef2bbd3f1eba69e416d214')
  }
```

```
jobintech> db.comments.insertOne({
             _id: ObjectId("507f1f77bcf86cd799439999"),
             post_id: ObjectId("68eeeaff3f1eba69e416cf9e"),
             author_id: ObjectId("68eee3593f1eba69e416ceee"),
             parent_comment_id: ObjectId("507f1f77bcf86cd799439031"),
             text: "Mongodb",
             timestamp: ISODate("2024-10-15T15:00:00Z")
           })
```

```
< {
    acknowledged: true,
    insertedId: ObjectId('507f1f77bcf86cd799439999')
  }
```

2. A user posts a reply to a comment: create a new `comment` document with the `parent_comment_id` set correctly. Then, create a `notification` for the author of the parent comment.:

Step 2a: Create the comment reply:

2b: Create a notification for the parent comment author:

```
jobintech> db.notifications.insertOne({
            _id: ObjectId("507f1f77bcf86cd799440001"),
            recipient_id: ObjectId("68eee3593f1eba69e416cef0"),
            sender_id: ObjectId("68eee3593f1eba69e416ceee"),
            type: "comment",
            post_id: ObjectId("507f1f77bcf86cd799439021"),
            comment_id: ObjectId("507f1f77bcf86cd799439999"),
            is_read: false,
            timestamp: ISODate("2024-10-15T15:00:00Z")
          })
```

Step 3a: Add the user to the group members: