P\_DB 106

Abiram Muthulingam

ETML

CID2A

Table des matières

[**Importer les données et le schéma de base de données** 2](#_Toc178597192)

[Charger le dump MySQL de la DB dans MySQL 2](#_Toc178597193)

[**Requêtes de sélection** 2](#_Toc178597194)

[Requêtes n°1 2](#_Toc178597195)

[Requêtes n°2 3](#_Toc178597196)

[Requêtes n°3 3](#_Toc178597197)

[Requêtes n°4 4](#_Toc178597198)

[Requêtes n°5 4](#_Toc178597199)

[Requêtes n°6 5](#_Toc178597200)

[Requêtes n°7 5](#_Toc178597201)

[Requêtes n°8 6](#_Toc178597202)

[Requêtes n°9 6](#_Toc178597203)

[Requêtes n°10 7](#_Toc178597204)

[**Gestions des utilisateurs** 7](#_Toc178597205)

[Administrateur du jeu 7](#_Toc178597206)

[Joueur 7](#_Toc178597207)

[Gestionnaire de la boutique 8](#_Toc178597208)

[**Création des index** 8](#_Toc178597209)

[**Backup / Restore** 8](#_Toc178597210)

# **Importer les données et le schéma de base de données**

## Charger le dump MySQL de la DB dans MySQL

Commande :

docker exec -i db mysql -u root -proot db\_space\_invaders < db\_space\_invaders.sql

Lance le container db dans docker, avec l’utilisateur root dans la base de données db\_space\_invaders.

# **Requêtes de sélection**

## Requêtes n°1

La première requête que l’on vous demande de réaliser est de sélectionner les 5 joueurs qui ont le meilleur score c’est-à-dire qui ont le nombre de points le plus élevé. Les joueurs doivent être classés dans l’ordre décroissant.

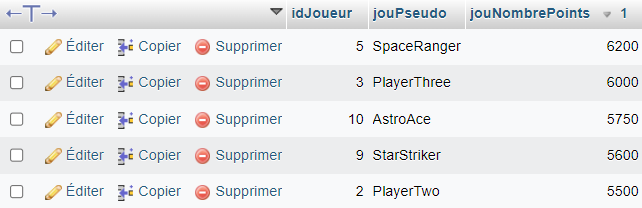
Commande :

SELECT \*

from t\_joueur

ORDER BY jouNombrePoints DESC

LIMIT 5;



Sélectionne toutes les colonnes de la tables t\_joueur, et affiche seulement les 5 qui ont le plus grand score dans jouNombrePoints.

## Requêtes n°2

Trouver le prix maximum, minimum et moyen des armes. Les colonnes doivent avoir pour nom « PrixMaximum », « PrixMinimum » et « PrixMoyen »

Commande :

SELECT MAX(armprix) AS PrixMaximum, MIN(armprix) AS PrixMinimum, AVG(armprix) AS PrixMoyen from t\_arme;



Sélectionne le prix maximum des armes en la renomment PrixMaximum, le prix minimum des armes en renommant PrixMinimum ainsi que la moyenne des prix en renommant PrixMoyen de la table t\_arme.

## Requêtes n°3

Trouver le nombre total de commandes par joueur et trier du plus grand nombre au plus petit. La 1ère colonne aura pour nom "IdJoueur", la 2ème colonne aura pour nom "NombreCommandes"

Commande :

SELECT t\_commande.fkJoueur AS idJoueur, COUNT(t\_detail\_commande.detQuantiteCommande) AS NombreCommandes

FROM t\_commande

JOIN t\_detail\_commande

ON t\_commande.idCommande = t\_detail\_commande.fkCommande

GROUP BY t\_commande.fkJoueur

ORDER BY NombreCommandes DESC;

Sélectionne l’id du joueur et compte le nombre de commande fait par joueur, JOIN… ON… permet de lier la table commande avec détail\_commande, puis trie du plus grand au plus petit le nombre de commandes.

## Requêtes n°4

Trouver les joueurs qui ont passé plus de 2 commandes. La 1ère colonne aura pour nom "IdJoueur", la 2ème colonne aura pour nom "NombreCommandes"

Commande :

SELECT fkJoueur AS IdJoueur, COUNT(\*) AS NombreCommandes

FROM t\_commande

GROUP BY fkJoueur

HAVING COUNT(\*) > 2;

Affiche l’id du joueur et le nombre de commandes par joueur qui sont supérieur à 2.

## Requêtes n°5

Trouver le pseudo du joueur et le nom de l'arme pour chaque commande.

Commande :

SELECT DISTINCT jouPseudo, armNom

FROM t\_arme

JOIN t\_detail\_commande

ON t\_detail\_commande.fkArme = t\_arme.idArme

JOIN t\_commande

ON t\_commande.idCommande = t\_detail\_commande.fkCommande

JOIN t\_joueur

ON t\_commande.fkJoueur = t\_joueur.idJoueur;

## Requêtes n°6

Trouver le total dépensé par chaque joueur en ordonnant par le montant le plus élevé en premier, et limiter aux 10 premiers joueurs. La 1ère colonne doit avoir pour nom "IdJoueur" et la 2ème colonne "TotalDepense"

Commande :

SELECT idJoueur AS IdJoueur, SUM(t\_arme.armPrix) AS TotalDepense

FROM t\_joueur

JOIN t\_commande

ON t\_commande.fkJoueur = t\_joueur.idJoueur

JOIN t\_detail\_commande

ON t\_commande.idCommande = t\_detail\_commande.fkCommande

JOIN t\_arme

ON t\_detail\_commande.fkArme = t\_arme.idArme

GROUP BY idJoueur

ORDER BY TotalDepense DESC LIMIT 10;

Sélectionne l’id du joueur et la somme des prix des armes achetées par chaque joueur dans l’ordre décroissant.

## Requêtes n°7

Récupérez tous les joueurs et leurs commandes, même s'ils n'ont pas passé de commande. Dans cet exemple, même si un joueur n'a jamais passé de commande, il sera quand même listé, avec des valeurs `NULL` pour les champs de la table `t\_commande`.

Commande :

SELECT idJoueur, idCommande

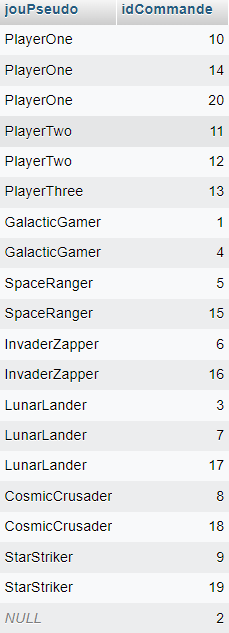
From t\_joueur

LEFT JOIN t\_commande

ON t\_joueur.idJoueur = t\_commande.fkJoueur;

Sélectionne l’id du joueur et l’id de la commande faites par le joueur, LEFT JOIN permet de lister tous les résultats de la table de gauche même s’il n’y a pas de correspondance dans la deuxième tables donc (NULL).

## Requêtes n°8

Récupérer toutes les commandes et afficher le pseudo du joueur s’il existe, sinon afficher `NULL` pour le pseudo.

Commande :

SELECT jouPseudo, idCommande

From t\_joueur

RIGHT JOIN t\_commande

ON t\_joueur.idJoueur = t\_commande.fkJoueur;

## Requêtes n°9

Trouver le nombre total d'armes achetées par chaque joueur (même si ce joueur n'a acheté aucune Arme).

Commande :

SELECT jouPseudo, SUM(detQuantiteCommande)

From t\_joueur

LEFT JOIN t\_commande

ON t\_joueur.idJoueur = t\_commande.fkJoueur

LEFT JOIN t\_detail\_commande

ON t\_detail\_commande.fkCommande = t\_commande.idCommande

GROUP BY jouPseudo;

## Requêtes n°10

Trouver les joueurs qui ont acheté plus de 3 types d'armes différentes

Commande :

SELECT jouPseudo, COUNT(t\_detail\_commande.fkArme)

FROM t\_joueur

JOIN t\_commande

ON t\_joueur.idJoueur = t\_commande.fkJoueur

JOIN t\_detail\_commande

ON t\_detail\_commande.fkCommande = t\_commande.idCommande

JOIN t\_arme

ON t\_detail\_commande.fkArme = t\_arme.idArme

GROUP BY t\_joueur.jouPseudo

Having COUNT(DISTINCT t\_detail\_commande.fkArme) > 3;



# **Gestions des utilisateurs**

## Administrateur du jeu

Création du rôle :

CREATE ROLE 'Administrateur du jeu';

Privilèges du rôle :

GRANT ALL PRIVILEGES ON \*.\* to 'Administrateur du jeu' with grant option;

Droit à tous les privilèges.

## Joueur

CREATE ROLE 'Joueur';

Privilèges du rôle :

GRANT SELECT ON db\_space\_invaders.t\_arme to 'Joueur';

Droit de lecture des informations sur la table t\_arme.

GRANT CREATE ON db\_space\_invaders.t\_commande to 'Joueur';

Droit de créer une commande.

GRANT SELECT ON db\_space\_invaders.t\_commande to 'Joueur';

Droit de lecture dans la table commande.

## Gestionnaire de la boutique

CREATE ROLE 'Gestionnaire de la boutique';

Privilèges du rôle :

GRANT SELECT ON db\_space\_invaders.t\_joueur to 'Gestionnaire de

la boutique’ ;

Droit de lecture dans la table t\_joueur.

GRANT UPDATE, SELECT, DELETE ON db\_space\_invaders.t\_arme to 'Gestionnaire de la boutique';

Droit de mise à jour, lecture, et suppression des armes.

GRANT SELECT ON db\_space\_invaders.t\_commande to ‘Gestionnaire de la boutique’;

Droit de lecture dans la table t\_commande.

# **Création des index**

1. Pourtant certains index existent déjà. Pourquoi ?
2. Quels sont les avantages et les inconvénients des index ?
3. Sur quel champ (de quelle table), cela pourrait être pertinent d’ajouter un index ?

# **Backup / Restore**

* Backup de la base de données db\_space\_invaders

docker exec -i db mysqldump -u root -proot --databases db\_space\_invaders > db\_space\_invaders.sql

Créer un fichier db\_space\_invaders.sql qui contient toute la base de données de db\_space\_invaders.

* Restore de la base de données db\_space\_invaders

docker exec -i db mysql -u root -proot < db\_space\_invaders.sql

Restauration de la base de données de db\_space\_invaders.