# P\_Web 295



**Auteur : Senna Cruz**

**Classe : CID2A**

**Date de dernière mise à jour : 11.02.2025**

# Sommaire

[P\_DB 106 1](#_Toc181090817)

[Sommaire 2](#_Toc181090818)

[Introduction 3](#_Toc181090819)

[Mise en place de la DB 4](#_Toc181090820)

[Création du conteneur MySQL sur Docker 4](#_Toc181090821)

[Importation de la DB sur le conteneur SQL 4](#_Toc181090822)

[Connexion à MySQL 4](#_Toc181090823)

[Gestion des utilisateurs 6](#_Toc181090824)

[Création des rôles et attribution des privilèges 6](#_Toc181090825)

[Création des utilisateurs et attribution aux rôles 6](#_Toc181090826)

[Requêtes de sélection SQL 7](#_Toc181090827)

[Requête n°1 : Les 5 joueurs avec le meilleur score 7](#_Toc181090828)

[Requête n°2 : Le prix maximum, minimum et moyen des armes 7](#_Toc181090829)

[Requête n°3 : Nombre total de commandes par joueur, trié par nombre de commandes 8](#_Toc181090830)

[Requête n°4 : Trouver les joueurs ayant passé plus de 2 commandes 8](#_Toc181090831)

[Requête n°5 : Trouver le pseudo du joueur et le nom de l'arme pour chaque commande 9](#_Toc181090832)

[Requête n°6 : Total dépensé par chaque joueur, ordonné par montant et limité aux 10 premiers 9](#_Toc181090833)

[Requête n°7 : Récupérer tous les joueurs et leurs commandes, même s'ils n'ont pas passé de commande 10](#_Toc181090834)

[Requête n°8 : Récupérer toutes les commandes et afficher le pseudo du joueur s’il existe sinon NULL 10](#_Toc181090835)

[Requête n°9 : Nombre total d'armes achetées par chaque joueur (même s'ils n'ont acheté aucune arme) 11](#_Toc181090836)

[Requête n°10 : Trouver les joueurs ayant acheté plus de 3 types d'armes différentes 11](#_Toc181090837)

[Index 12](#_Toc181090838)

[Backup et Restore 13](#_Toc181090839)

# Introduction

# Mise en place de la DB

Premièrement est nécessaire mettre en place un conteneur *SQL* dans l’application Docker pour pouvoir exécuter les requêtes en ligne de commande sur le terminal Windows. Pour la réussite de ce travail, trois tâches sont nécessaire la création d’un conteneur *SQL*, l’importation de la base de données et la connexion à la base de données (*MySQL*).

## Création du conteneur MySQL sur Docker

Un conteneur Docker est un conteneur exécutable autonome, qui comprend tous les éléments nécessaires pour exécuter une application, notamment les bibliothèques, les outils système, le code et le runtime.

Un fois donnée la définition de conteneur Docker, c’est le moment de créer ce conteneur dans lequel est présent *MySQL*.

Le téléchargement du paquet des commandes Docker et MySQL est requis, donc télécharger le dossier *ZIP* depuis le support de cours sur Teams et le dézipper.

Exécuter la commande ci-dessous sur l’emplacement du dossier *Docker\_MySQL* pour créer le container *SQL*.



## Importation de la DB sur le conteneur SQL

L’importation de la base de données, avec un fichier *SQL*, se fait grâce à la ligne de commande ci-dessous exécutée sur l’emplacement du dossier *Docker\_MySQL*.



La commande importe le contenu du fichier SQL spécifié dans une base de données *MySQL* en cours d'exécution dans un conteneur Docker nommé "*db*" qui a été créé précédemment.

## Connexion à MySQL

Maintenant pour l’utilisation de la base de données importée est nécessaire se connecter sur le conteneur de docker, puis sur *MySQL*.

Dans le terminal Windows (*CMD*), une fois placé sur le dossier *Docker\_MySQL*, exécuter la commande ci-dessous pour se connecter au container *SQL* du Docker.



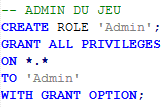
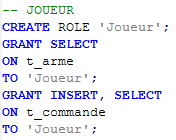
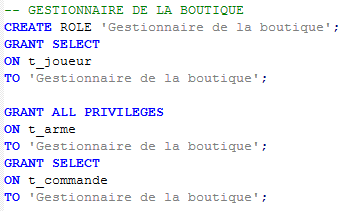
Une fois connecté au conteneur, exécuter la deuxième (ci-dessous) commande qui permet la connexion à *MySQL* et l’utilisation de la base de données.



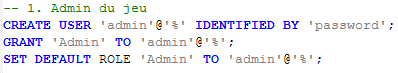
Enfin le terminal est prêt pour exécuter les Requêtes *SQL* sur la base de données.

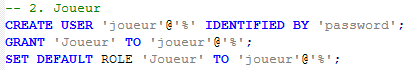
# Gestion des utilisateurs

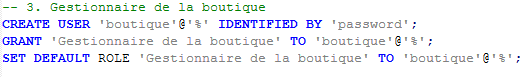
## Création des rôles et attribution des privilèges

## Création des utilisateurs et attribution aux rôles

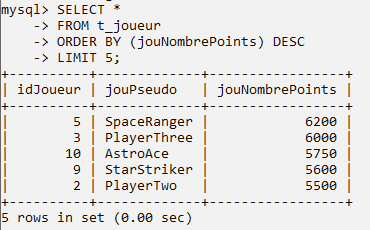




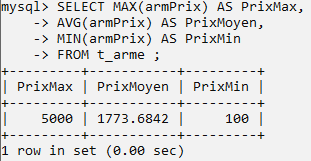


# Requêtes de sélection SQL

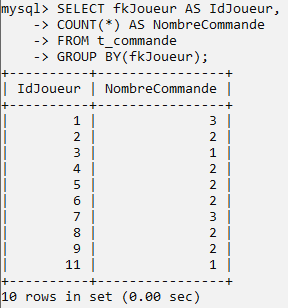
## Requête n°1 : Les 5 joueurs avec le meilleur score



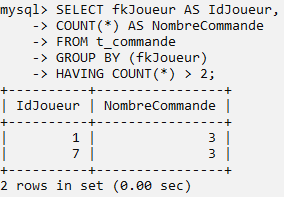
## Requête n°2 : Le prix maximum, minimum et moyen des armes



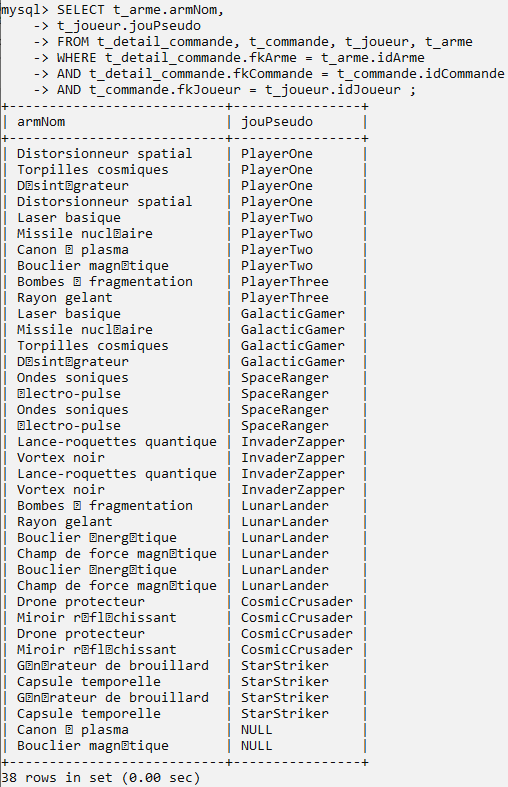
## Requête n°3 : Nombre total de commandes par joueur, trié par nombre de commandes



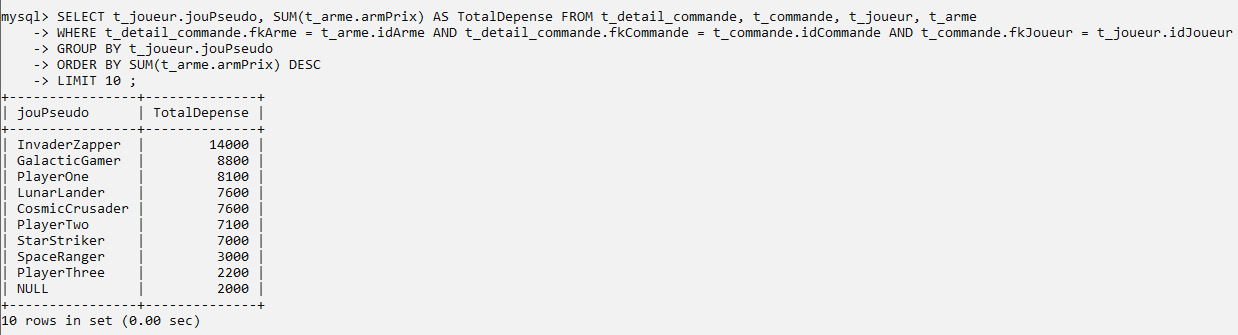
## Requête n°4 : Trouver les joueurs ayant passé plus de 2 commandes



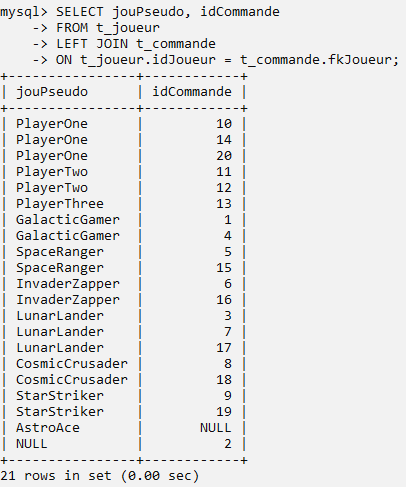
## Requête n°5 : Trouver le pseudo du joueur et le nom de l'arme pour chaque commande



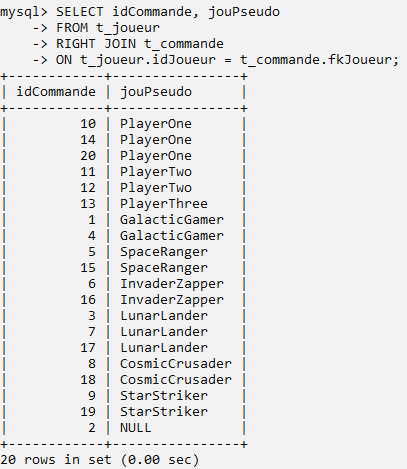
## Requête n°6 : Total dépensé par chaque joueur, ordonné par montant et limité aux 10 premiers



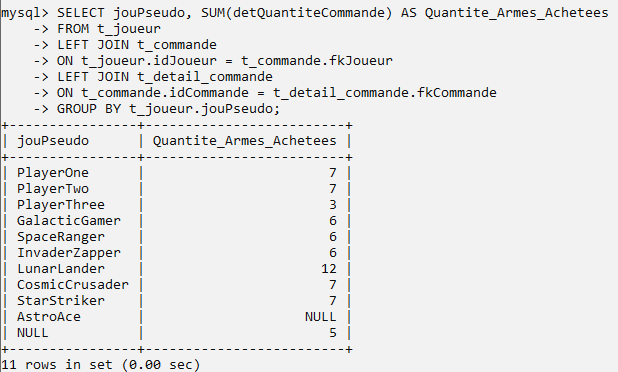
## Requête n°7 : Récupérer tous les joueurs et leurs commandes, même s'ils n'ont pas passé de commande



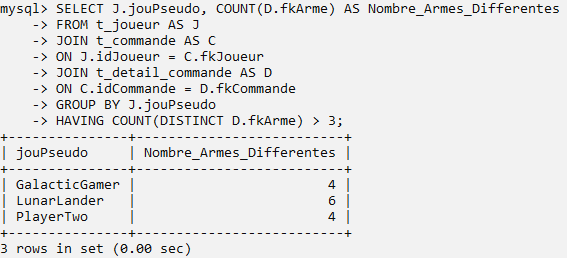
## Requête n°8 : Récupérer toutes les commandes et afficher le pseudo du joueur s’il existe sinon NULL



## Requête n°9 : Nombre total d'armes achetées par chaque joueur (même s'ils n'ont acheté aucune arme)



## Requête n°10 : Trouver les joueurs ayant acheté plus de 3 types d'armes différentes



# Index

1. **Pourtant certains index existent déjà. Pourquoi ?**

Parce que c'est des PRIMARY KEY ou des UNIQUE

1. **Quels sont les avantages et les inconvénients des index ?**

Pour identifier un élément unique de la table

1. **Sur quel champ (de quelle table), cela pourrait être pertinent d’ajouter un index ? Justifier votre réponse**.

Imaginons une course avec 100 participants, chaque coureur aura un numéro unique sur son maillot pour pouvoir l'identifier. Ce numéro sera l'index.

# Backup et Restore

