Morgado-Samagaio Jonathan

**IBD / TP10 Oracle**

Table des matières

[Table des matières 2](#_Toc58591317)

[1. Introduction 3](#_Toc58591318)

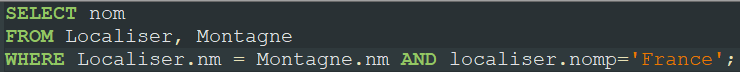
[2. Requêtes 3](#_Toc58591319)

[3. Conclusion 8](#_Toc58591320)

# Introduction

Dans ce TP, nous allons à nouveau faire des requêtes multi-table. Nous allons voir comment écrire les jointures en SQL.

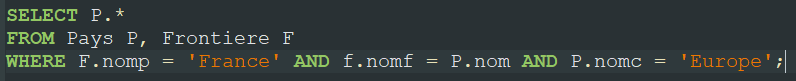
# Requêtes

1. Pour commencer le TP, nous allons en premier faire une jointure des tables Localiser et Montagne pour obtenir les montagnes localisé en France. Pour cela, nous allons utiliser la commande suivante : 

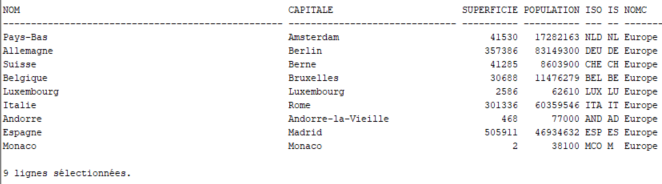
Nous pouvons voir que le FROM prend deux tables, c’est donc bien une requête multi-table. Il y a aussi la condition de jointure dans le WHERE avec l’égalité des deux colonnes. Nous obtenons donc :



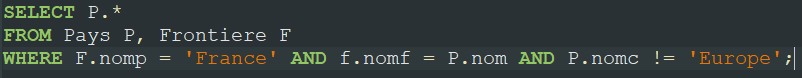
1. Pour cette requête, nous allons ajouter plus de conditions dans le WHERE pour obtenir les pays européen frontaliers à la France. Nous utilisons donc la commande suivante :



La condition de jointure est toujours présente avec l’égalité entre deux tables et nous avons ajouté une nouvelle condition. Nous obtenons donc :



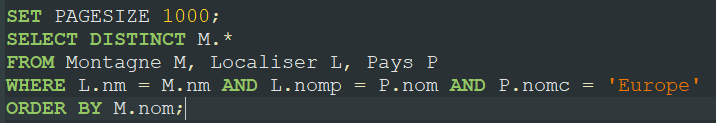
1. Cette fois-ci, l’information demandée est presque similaire à la précédente. En effet, nous devons obtenir les pays non européen frontaliers à la France. Pour cela, nous allons juste prendre l’inverse d’une des conditions de la commande précédente :



Nous pouvons voir qu’il y a un ‘ !’ pour la condition du continent qui nous permet de prendre tous les pays qui n’appartiennent pas au continent Europe. Nous obtenons donc :



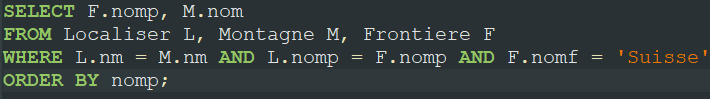
1. Nous allons commencer à joindre trois tables dans cette requête. Nous cherchons les montagnes localisées en Europe. Le résultat des requêtes commençant à être long, nous allons utiliser la commande SET PAGESIZE pour avoir un meilleur affichage. La commande est donc :



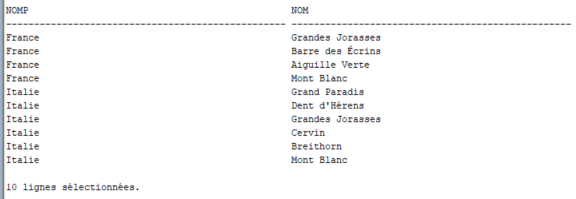
Nous pouvons que, cette fois-ci, il y a deux conditions de jointure pour joindre les trois tables ensemble. Il y a aussi l’ajout du ORDER BY Pour trié le résultat. Nous obtenons :



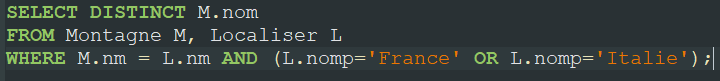
1. Nous restons dans le même principe de requête avec l’interrogation de trois tables. Nous cherchons les montagnes localisées dans les pays frontaliers de la Suisse. Nous utilisons donc :



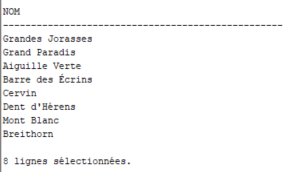
La commande est similaire à la précédente. Nous obtenons :



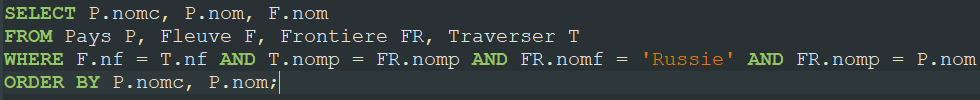
1. Pour cette requête, nous allons utiliser le OR dans les conditions. Le OR n’étant pas prioritaire, il faut faire attention lors de son utilisation a ce que les conditions testées soient celles que l’on veut. Nous utilisons la commande :



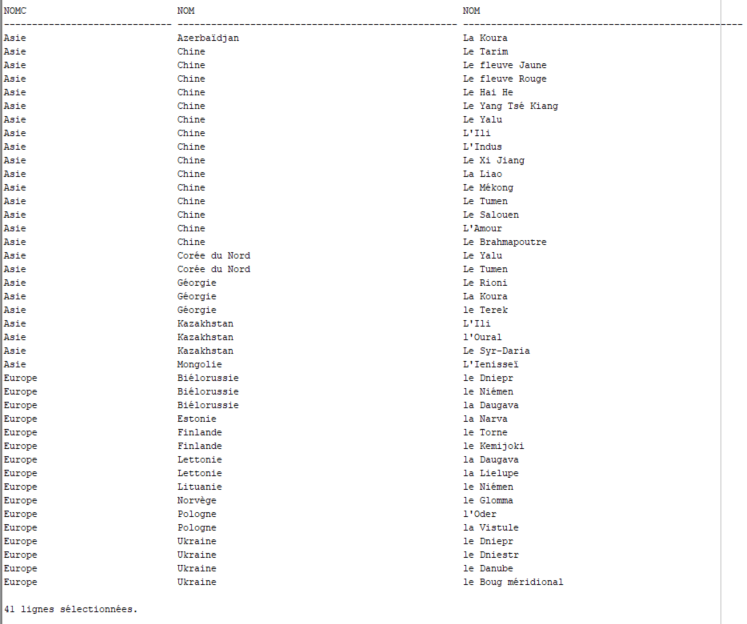
Nous pouvons donc voir les parenthèses pour le OR pour que la condition correcte soit respectée. Nous obtenons :



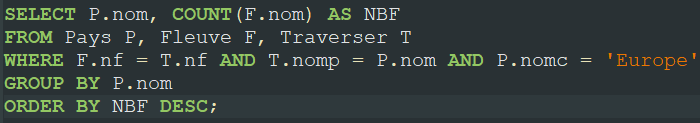
1. Nous compliquons encore plus les choses avec une requête utilisant quartes tables. Nous cherchons les fleuves qui traversent les pays frontaliers à la Russie mais en affichant les continents, le nom du pays et le nom du fleuve. Nous utilisons la commande suivante :



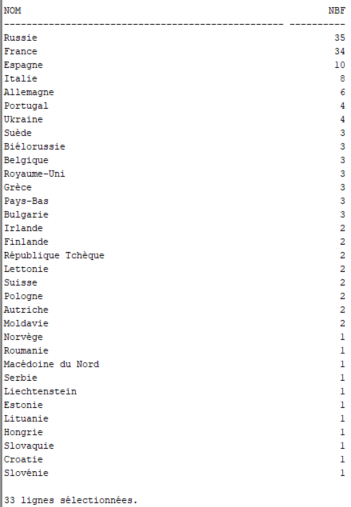
La commande ressemble à la question 5 car c’est le même principe sur des tables différentes. Il y a juste une condition de jointure de plus. Nous obtenons :



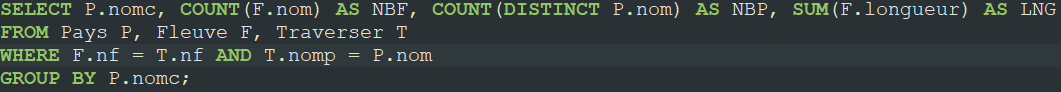
1. Nous allons pour la première fois utiliser COUNT dans des requêtes multi-table. Nous cherchons le nombre de fleuve qui traversent chaque pays européen. Nous utilisons donc la commande :



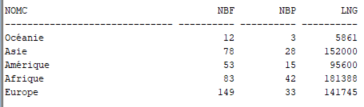
Nous pouvons voir que son utilisation ne change pas. Nous obtenons donc :



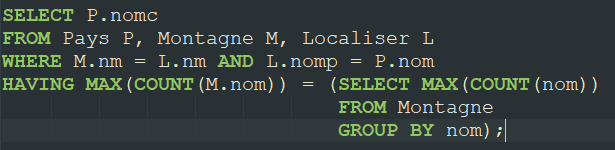
1. Nous allons utiliser plusieurs COUNT et un SUM dans le SELECT. Nous cherchons le nombre de fleuve, le nombre de pays et la longueur cumulée des fleuves par continent. Nous utilisons la commande :



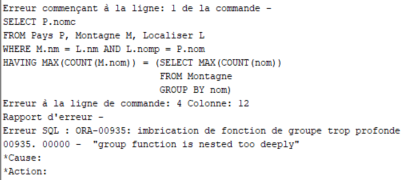
Nous pouvons voir l’utilisation du DISTINCT dans le COUNT pour ne pas compter plusieurs fois les pays. Nous obtenons donc :



1. Cette question nous demande de récupérer le pays le plus montagneux. Cependant, j’ai quelques difficultés à y répondre. Ma commande est :



SQL Developpeur renvoie l’erreur :



Je n’arrive pas à comprendre d’où vient mon erreur.

# Conclusion

Dans ce TP, nous avons pu voir comment écrire les opérations de jointure. Nous avons pu interroger plusieurs tables en même temps avec des conditions de jointure dans le WHERE. Nous avons aussi vu l’utilisation du AND et du OR pour les opérations logiques. Je n’ai malheureusement pas pu finir la dernière question.