

TP 2 : Réplication de base de données

Frédéric Fort - Université de Lille

2025

Introduction

Dans ce TP, nous allons mettre en pratique les compétences acquises précédemment sur un cas d'étude concret et courant. Le but sera de mettre en place une base de données sur plusieurs instance OpenStack et de les configurer afin qu'elles offrent des garanties de cohérence et disponibilité.

Le but sera de mettre en place différentes architectures, les caractériser et de les contraster. Vous êtes libres de sélectionner le SGBD de votre choix tant que celui-ci est open source. Munissez-vous également d'un jeu de données permettant de peupler vos instances.

1 Prise en main

Configurer deux instances avec le SGBD de votre choix en suivant les procédures du TP1. Configurer une troisième instance afin que celle-ci puisse opérer en intermédiaire si nécessaire. Validez que celles-ci fonctionnent correctement et répondent aux requêtes.

2 Réplication stand-by lecture seule

Pour le premier cas d'étude, le choix est fait qu'un seul serveur, dit le serveur primaire, doit répondre aux requêtes initialement. Il doit pouvoir servir des requêtes en lecture et en écriture. Si celui-ci ne répond plus aux requêtes, le serveur secondaire doit continuer de répondre aux requêtes en lecture seule. L'état du serveur secondaire doit respecter les requêtes en écriture les plus récentes possibles, mais refuser les futures écritures. Une remise en route du serveur principale doit rétablir la possibilité d'effectuer des écritures.

Mettez en place l'architecture ci-dessus. Construisez un scénario caractérisant votre solution.

3 Réplication stand-by lecture/écriture

Est-il possible de retravailler votre solution de manière à ce que le serveur secondaire puisse accepter des requêtes en écriture ?

Si oui, mettez cette architecture en place, caractérisez la et comparez-la à la solution précédente. Observez-vous des différences ?

Si non, expliquez pourquoi en utilisant la documentation de votre SGBD et le cahier de charges.

4 Répartition de charge

Dans les configurations précédentes, un seul serveur répondait aux requêtes à un instant T. Ceci permettait d'améliorer la disponibilité, mais multipliait aussi le coût par 2 pour un service à priori identique l'écrasante majorité du temps.

Dans cette exercice, il vous sera demandé de mettre en place une architecture où plusieurs serveurs peuvent répondre aux requêtes à n'importe quel moment. Si non utilisée jusqu'ici, votre troisième instance intermédiaire pourrait se révéler utile.

4.1 Priorisation de la cohérence

Mettez en place l'architecture décrite ci-dessus. Si plusieurs choix s'offrent à vous, privilégiez la cohérence.

4.2 Priorisation de la disponibilité

Mettez en place l'architecture décrite ci-dessus. Si plusieurs choix s'offrent à vous, privilégiez la disponibilité.

4.3 Évaluation

Décrivez les choix différenciant vos solutions.

Utilisez les scénarios utilisés précédemment et étendez-les si besoin afin révéler les spécificités de chaque solution. Dans quel cas privilégieriez-vous quelle solution ?

Comment se comparent vos solutions vis-à-vis de celles de l'exercice précédent ? Quelles avantages voyez-vous ? Quels inconvénients voyez-vous ?