# Rapport de Configuration : Déploiement du Socle de Services (Partie 3)

Ce document résume le déploiement de l'infrastructure de données de base pour l'environnement staging, conformément à la Partie 3 du guide. L'objectif était de provisionner une base de données PostgreSQL (Cloud SQL) et un cache en mémoire Redis (Memorystore) à l'aide de Terraform.

## 1. Objectif

L'objectif était de créer un "socle de services" managés, sécurisés et prêts à être utilisés par les futures applications. Ces services devaient être déployés à l'intérieur du réseau privé (VPC) créé précédemment pour garantir qu'ils ne soient pas exposés à l'internet public.

## 2. Processus de Déploiement

Le déploiement a été réalisé en plusieurs étapes clés :

1. **Gestion Sécurisée du Mot de Passe :**
   * Un secret nommé **postgres-password-staging** a été créé manuellement dans **Google Secret Manager**.
   * Cette approche garantit que le mot de passe de la base de données n'est jamais stocké en clair dans le code source, respectant ainsi les meilleures pratiques de sécurité.
2. **Définition de l'Infrastructure en tant que Code :**
   * Un fichier **database.tf** a été créé pour définir l'instance Cloud SQL (PostgreSQL), la base de données elle-même (skillforge\_db) et l'utilisateur applicatif (skillforge\_user). Le code a été configuré pour lire le mot de passe directement depuis Secret Manager.
   * Un fichier **cache.tf** a été créé pour définir l'instance Memorystore (Redis). Ce code a également géré la configuration réseau complexe requise, incluant la réservation d'une plage d'adresses IP et la création d'une connexion de peering VPC.
3. **Application de la Configuration :**
   * La commande **terraform apply** a été exécutée depuis le dossier terraform/environments/staging/ pour créer ces ressources sur Google Cloud.

## 3. Problèmes Rencontrés et Solutions Appliquées

Cette phase a été riche en enseignements et a nécessité plusieurs itérations pour résoudre une cascade de problèmes techniques, illustrant un scénario de débogage d'infrastructure très réaliste.

### Problème 1 : Dépendance Réseau et Changements d'Offre GCP

* **Symptômes (en chaîne) :**
  1. La création de la base de données a d'abord échoué car la connexion réseau (VPC peering) n'était pas encore prête (une "race condition").
  2. Après correction, elle a échoué à nouveau car le type de machine (db-n1-standard-1) n'était pas compatible avec la nouvelle édition par défaut de Cloud SQL ("Enterprise Plus").
  3. Après avoir forcé l'édition "Enterprise", une dernière erreur a indiqué que le type de machine n'était de toute façon pas valide pour PostgreSQL, qui requiert un type shared-core ou custom.
* **Solution Appliquée :** Le fichier **database.tf** a été profondément modifié pour :
  1. Ajouter une dépendance explicite (depends\_on) pour s'assurer que la base de données attende la création de la connexion réseau.
  2. Spécifier explicitement l'édition à utiliser : **edition = "ENTERPRISE"**.
  3. Changer le type de machine pour un type compatible et économique : **tier = "db-g1-small"**.

### Problème 2 : Timeout et Désynchronisation de l'État Terraform

* **Symptôme :** Lors d'une tentative d'application, la création de la base de données (une opération très longue) a dépassé le délai d'attente de Terraform, qui a affiché une erreur de timeout. Cependant, la création s'est poursuivie avec succès en arrière-plan sur Google Cloud. Au terraform apply suivant, une erreur 409: The Cloud SQL instance already exists est apparue.
* **Cause Racine :** L'état de Terraform (sa "mémoire" de ce qui existe) était désynchronisé de la réalité sur le cloud. Terraform pensait que l'instance n'existait pas et a essayé de la recréer.
* **Solution Appliquée :** La commande **terraform import** a été utilisée pour forcer la synchronisation. Elle a permis de "dire" à Terraform que l'instance existait déjà sur Google Cloud et de l'adopter dans son état. Un dernier terraform apply a ensuite pu créer les sous-ressources (base de données et utilisateur) sans conflit.

## 4. Résultat Final

Malgré les défis, la **Partie 3** a été complétée avec succès. L'infrastructure de l'environnement staging dispose maintenant :

* D'une instance **Cloud SQL (PostgreSQL)** nommée skillforge-pg-instance-staging.
* D'une instance **Memorystore (Redis)** nommée skillforge-redis-instance-staging.

Ces deux services sont correctement connectés au VPC skillforge-vpc-staging et sont gérés par Terraform, prêts pour la prochaine phase du projet.