Réglage des SQL paramètres Postgre dans Amazon RDS et Amazon Aurora

AWS Directives prescriptives

AWS Directives prescriptives: Réglage des SQL paramètres Postgre dans Amazon RDS et Amazon Aurora

Table of Contents

Réglage des paramètres PostgreSQL dans Amazon RDS et Amazon Aurora

Sumana Yanamandra, Ramu Jagini et Rohit Kapoor, Amazon Web Services ()AWS

Février 2024 (historique du document)

Amazon Aurora PostgreSQL Compatible Edition et Amazon Relational Database Service (Amazon RDS) pour PostgreSQL sont des services de base de données relationnelle open source sophistiqués qui offrent une gamme complète de fonctionnalités. Vous pouvez utiliser ces services pour configurer votre base de données PostgreSQL sur diverses plateformes et applications.

Aurora et Amazon RDS proposent un moyen simplifié de gérer et d'exploiter vos bases de données PostgreSQL. Ils sont conçus pour gérer l'infrastructure de base de données et fournir une disponibilité, une durabilité et une évolutivité élevées tout en vous concentrant sur le développement d'applications. Cependant, les configurations par défaut de ces services peuvent ne pas être optimales pour toutes les charges de travail. Par défaut, ces services sont configurés pour fonctionner partout avec le moins de ressources possible et sans introduire de vulnérabilités. Le réglage des paramètres peut vous aider à améliorer les performances, à réduire les temps d'arrêt et à améliorer l'efficacité globale de la base de données. En optimisant les paramètres pour votre charge de travail spécifique, vous pouvez tirer pleinement parti des fonctionnalités fournies par Amazon RDS et Aurora et optimiser leurs avantages.

Par exemple, vous pouvez améliorer les performances en optimisant Aurora et Amazon RDS pour PostgreSQL et en configurant leurs paramètres. Vous devez également tenir compte des performances lorsque vous créez des requêtes de base de données. Même lorsque vous optimisez les paramètres de base de données, le système peut mal fonctionner si vos requêtes analysent des tables complètes, utilisent un index ou exécutent des opérations de jointure ou d'agrégation coûteuses.

Ce guide s'adresse aux développeurs de bases de données, aux ingénieurs de bases de données et aux administrateurs qui souhaitent ajuster les paramètres de mémoire, d'autovacuum, de journalisation et de réplication logique pour leurs bases de données PostgreSQL. Le guide couvre également les paramètres spécifiques à Amazon RDS pour PostgreSQL et à la compatibilité avec Aurora PostgreSQL. Le réglage de ces paramètres peut vous aider à optimiser les performances de la base de données et à réduire l'utilisation des ressources pour votre charge de travail spécifique, ce qui se traduit par de meilleures performances et des économies de coûts.

Utilisation de groupes de paramètres de base de données et de clusters de bases de

Amazon RDS et Aurora peuvent déterminer automatiquement les valeurs de paramètres les plus appropriées pour certains paramètres en fonction de la taille de votre instance de base de données. Ils prennent également en charge la personnalisation des paramètres pour le réglage des performances via des groupes de paramètres pour les instances de base de données et les clusters.

Vous pouvez utiliser les groupes de paramètres de base de données et de clusters de bases de données pour modifier les paramètres qui contrôlent divers aspects du comportement du moteur de base de données, tels que l'utilisation de la mémoire, les E/S du disque, la mise en réseau et le verrouillage. En ajustant ces paramètres, vous pouvez optimiser le moteur de base de données en fonction de votre charge de travail spécifique et améliorer les performances.

Vous pouvez créer et configurer des groupes de paramètres de base de données et de clusters de bases de données à l' AWS Management Console aide de l'API, de la AWS Command Line Interface (AWS CLI) ou de l'API Amazon RDS. Ce guide part du principe que vous utilisez le AWS CLI. Pour les instructions relatives à la console et à l'API, consultez les sections Utilisation des groupes de paramètres de base de données et Utilisation des groupes de paramètres de clusters de bases de données dans la documentation Amazon RDS.

> ⚠ Important
>
> Pour utiliser les AWS CLI commandes fournies dans ce guide, vous devez d'abord installer et configurer le AWS CLI.

Pour créer et configurer un groupe de paramètres de base de données :

```
# Create a new DB parameter group
aws rds create-db-parameter-group \
  --db-parameter-group-name mydbparamgroup \
  --db-parameter-group-family postgres13 \
  --description "My DB Parameter Group"

# Modify a parameter on the DB parameter group
aws rds modify-db-parameter-group \
```

```
  --db-parameter-group-name <param group name> \
  --parameters "ParameterName=max_connections<parameter-
name>,ParameterValue=<value>,ApplyMethod=immediate"

# Verify DB parameters
aws rds describe-db-parameters \
  --db-parameter-group-name aurora-instance-1
```

Pour créer et configurer un groupe de paramètres de cluster de base de données :

```
# Create a new DB cluster parameter group
aws rds create-db-cluster-parameter-group \
  --db-cluster-parameter-group-name myparametergroup \
  --db-parameter-group-family postgres12 \
  --description "My new parameter group"

# Modify a parameter on the DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name aws-guide-cluster \
  --parameters "ParameterName=<parameter-name>,ParameterValue=,ApplyMethod=immediate"

# Allocate the new DB cluster parameter to your cluster
aws rds modify-db-cluster  \
  --db-cluster-identifier  \
  --db-cluster-parameter-group-name=-cluster

# Verify cluster parameters
aws rds describe-db-cluster-parameters \
  --db-cluster-parameter-group-name=-cluster
```

> **Note**
>
> Aurora et Amazon RDS fournissent un groupe de paramètres par défaut avec des valeurs préconfigurées qui ne peuvent pas être modifiées.

Les groupes de paramètres peuvent être définis comme statiques ou dynamiques. Les paramètres dynamiques sont appliqués immédiatement, que l'`ApplyMethod=immediate`option soit activée ou non. Les paramètres statiques nécessitent un redémarrage manuel pour être pris en compte.

Réglage des paramètres de mémoire

Le réglage des paramètres de mémoire est une tâche essentielle pour optimiser les performances des bases de données compatibles avec Amazon RDS et Aurora PostgreSQL. L'allocation correcte de mémoire pour diverses opérations de base de données telles que l'exécution de requêtes, le tri, l'indexation et la mise en cache peut améliorer considérablement les performances de la base de données. Cette section couvre certains des paramètres de mémoire les plus critiques compatibles avec Amazon RDS et Aurora PostgreSQL, notamment leurs valeurs par défaut, les formules de calcul des valeurs appropriées et la manière de les modifier. Pour obtenir la liste complète des paramètres, consultez la section Utilisation des paramètres de votre instance de base de données RDS pour PostgreSQL dans la documentation Amazon RDS et les paramètres Amazon Aurora PostgreSQL dans la documentation Aurora.

L'optimisation de ces paramètres nécessite une connaissance approfondie de la charge de travail de votre base de données, ainsi que des ressources disponibles sur votre instance de base de données Aurora ou Amazon RDS. Les performances du système sont influencées par deux grandes catégories de paramètres : les paramètres vitaux et les paramètres conditionnels.

Les paramètres vitaux sont des paramètres indispensables qui exercent une influence directe et significative sur les performances du système et sont essentiels pour obtenir des résultats optimaux.

- buffers partagés
- temp_buffers
- taille_cache_effective
- work_mem
- maintenance_work_mem

Les paramètres conditionnels sont des paramètres spécifiques au scénario et à l'entreprise. Ils dépendent d'autres facteurs et jouent un rôle indirect mais essentiel dans le soutien des paramètres vitaux et dans l'optimisation des performances globales du système.

- coût d'une page aléatoire
- seq_page_cost
- track_activity_query_size
- délai d'expiration de la session d'in_in_transaction

- timeout du relevé
- chemin_de recherche
- max_connexions

Ces paramètres sont décrits plus en détail dans les sections suivantes.

shared_buffers

Le `shared_buffers` paramètre contrôle la quantité de mémoire utilisée par PostgreSQL pour mettre en cache les données en mémoire. La définition d'une valeur appropriée à ce paramètre peut contribuer à améliorer les performances des requêtes.

Pour Amazon RDS, la valeur par défaut `shared_buffers` est définie sur `{DBInstanceClassMemory/32768}` octets, en fonction de la mémoire disponible pour l'instance de base de données. Pour Aurora, la valeur par défaut est définie sur`{DBInstanceClassMemory/12038,-50003}`, en fonction de la mémoire disponible pour l'instance de base de données. La valeur optimale pour ce paramètre dépend de plusieurs facteurs, notamment de la taille de la base de données, du nombre de connexions simultanées et de la mémoire d'instance disponible.

AWS CLI syntaxe

La commande suivante change `shared_buffers` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify shared_buffers on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=shared_buffers,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify shared_buffers on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters "ParameterName=shared_buffers,ParameterValue=<new-
value>,ApplyMethod=immediate"
```

Type : statique (l'application des modifications nécessite un redémarrage)

Valeur par défaut : `{DBInstanceClassMemory/32768}` octets dans Amazon RDS pour PostgreSQL, dans Aurora `{DBInstanceClassMemory/12038,-50003}` PostgreSQL compatible. Dans la plupart des cas, cette équation représente environ 25 % de la mémoire de votre système. Conformément à cette directive, le `shared_buffers` réglage du groupe de paramètres est configuré en utilisant les unités par défaut de PostgreSQL, à savoir 8 000 tampons, au lieu d'octets ou de kilo-octets.

Le réglage des `shared_buffers` paramètres peut avoir un impact significatif sur les performances. Nous vous recommandons donc de tester vos modifications de manière approfondie afin de vous assurer que la valeur est adaptée à votre charge de travail.

Exemple

Supposons que vous disposiez d'une application de services financiers qui exécute une base de données PostgreSQL sur Amazon RDS ou Aurora. Cette base de données est utilisée pour stocker les données des transactions des clients. Il comporte un grand nombre de tables et est accessible par de multiples applications sur un grand nombre de serveurs. L'application connaît des performances de requête lentes et une utilisation élevée du processeur. Vous déterminez que le réglage du `shared_buffers` paramètre peut contribuer à améliorer les performances.

Dans Amazon RDS for PostgreSQL, la `shared_buffers` valeur par défaut de est `{DBInstanceClassMemory/32768}` définie sur les octets de `db.r5.xlarge` mémoire disponible (par exemple, 3 Go). Pour déterminer la valeur appropriée pour`shared_buffers`, vous devez exécuter une série de tests avec différentes valeurs de`shared_buffers`, en commençant par la valeur par défaut de la mémoire disponible et en augmentant progressivement la valeur. Pour chaque test, vous mesurez les performances des requêtes et l'utilisation du processeur de la base de données.

Sur la base des résultats du test, vous déterminez que la définition d'une valeur de `shared_buffers` 8 Go permet d'obtenir les meilleures performances globales de requête et d'utiliser le processeur pour votre charge de travail. La valeur est déterminée par une combinaison de tests et d'analyses des caractéristiques de la charge de travail, notamment la taille de la base de données, le nombre et la complexité des requêtes, le nombre d'utilisateurs simultanés et les ressources système disponibles. Une fois la modification apportée, vos systèmes de surveillance vérifient les performances de la base de données pour s'assurer que la nouvelle valeur est adaptée à votre charge de travail. Vous pouvez ensuite affiner les paramètres supplémentaires si nécessaire pour améliorer encore les performances.

temp_buffers

`temp_buffers`est un paramètre de configuration clé dans Aurora PostgreSQL compatible et Amazon RDS for PostgreSQL qui peut avoir un impact significatif sur les performances des charges de travail impliquant des tris, des hachages et des opérations d'agrégation sur des tables temporaires. Ce paramètre détermine la quantité de mémoire allouée aux tampons temporaires, ce qui, à son tour, affecte l'efficacité et la rapidité de ces opérations. Si la mémoire allouée est insuffisante`temp_buffers`, le système devra peut-être utiliser des méthodes plus lentes et moins efficaces pour les opérations de tri, de hachage et d'agrégation sur les tables temporaires, ce qui se traduira par des performances sous-optimales.

AWS CLI syntaxe

La commande suivante change `temp_buffers` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify temp_buffers on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=temp_buffers,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify temp_buffers on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=temp_buffers,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 8 Mo

Pour plus d'informations sur ce paramètre, consultez la section Consommation de ressources dans la documentation de PostgreSQL.

Exemple

Si votre charge de travail implique de nombreuses opérations de tri, de hachage et d'agrégation sur des tables temporaires, vous `temp_buffers` risquez de ne pas allouer suffisamment de

mémoire. Dans ce cas, le système peut être amené à effectuer des opérations de tri sur des tables temporaires, ce qui entraîne des méthodes sur disque plus lentes au lieu d'opérations de tri, de hachage et d'agrégation en mémoire. Cela peut entraîner un ralentissement significatif des performances des requêtes, en particulier pour les requêtes impliquant de grands ensembles de données. L'augmentation de la valeur de `temp_buffers` peut garantir qu'il y a suffisamment de mémoire disponible pour effectuer de telles opérations en mémoire, ce qui se traduit par une amélioration significative des performances.

Pour trouver la valeur optimale de`temp_buffers`, surveillez les performances de votre système et identifiez les domaines dans lesquels les performances ne sont pas optimales. Si vous remarquez des temps de réponse aux requêtes lents ou une utilisation élevée du processeur, pensez à effectuer des ajustements`temp_buffers`. Par exemple, si votre charge de travail implique un grand nombre de tables temporaires, l'augmentation de la valeur de `temp_buffers` peut contribuer à garantir le stockage de ces tables en mémoire. Cela peut être beaucoup plus rapide que d'utiliser les E/S en lecture/écriture depuis le stockage.

Testez différentes valeurs par petits `temp_buffers` incréments et surveillez attentivement les performances du système après chaque modification. Analysez l'impact de différentes valeurs sur les performances et ajustez le réglage en fonction des caractéristiques spécifiques de votre charge de travail.

effective_cache_size

Le `effective_cache_size` paramètre indique la quantité de mémoire que PostgreSQL doit supposer être disponible pour la mise en cache des données. La définition correcte de ce paramètre peut améliorer les performances en permettant à PostgreSQL de mieux utiliser la mémoire disponible.

AWS CLI syntaxe

La commande suivante change `effective_cache_size` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify effective_cache_size on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=effective_cache_size,ParameterValue=<new_value>,ApplyMethod=immediate"
```

```
# Modify effective_cache_size on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=effective_cache_size,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `SUM(DBInstanceClassMemory/12038,-50003)` Ko

Exemple

Une plateforme d'apprentissage en ligne possède une vaste base de données de supports de cours, de données sur les étudiants et d'autres contenus fréquemment consultés par les utilisateurs. L'application s'exécute sur une instance Amazon RDS `db.r5.xlarge` for PostgreSQL dotée de 32 Go de mémoire. Les performances de l'application sont ralenties lorsque les utilisateurs essaient de lire le contenu fréquemment consulté. Après avoir analysé l'utilisation des ressources du serveur de base de données, vous déterminez que PostgreSQL n'utilise pas de manière optimale la mémoire disponible.

Le `effective_cache_size` paramètre d'Amazon RDS for PostgreSQL contrôle la quantité de mémoire utilisée par le serveur pour la mise en cache du disque. La valeur par défaut est définie sur `SUM({DBInstanceClassMemory/12038},-50003)` Ko pour la classe d'instance`db.r5.xlarge`, mais cette valeur par défaut peut ne pas convenir à toutes les charges de travail. Dans cet exemple, le serveur de base de données peut stocker de grandes quantités de supports de cours fréquemment consultés et de données sur les étudiants. L'augmentation de la valeur du `effective_cache_size` paramètre peut entraîner la mise en cache d'un plus grand nombre de données en mémoire, ce qui réduit le nombre de lectures de disque requises et améliore les performances des requêtes.

Lorsque vous exécutez une requête, Amazon RDS for PostgreSQL vérifie d'abord si les données requises par la requête se trouvent déjà dans le cache. Dans ce cas, les données peuvent être lues depuis la mémoire au lieu d'être lues depuis le disque. Si les données ne se trouvent pas dans le cache, elles doivent être lues sur le disque, ce qui peut être une opération lente.

Pour la plateforme d'apprentissage en ligne, vous pouvez décider de passer `effective_cache_size` à 16 Go (la moitié de la mémoire disponible) après les tests et les analyses. Cette valeur permet à PostgreSQL de mieux utiliser la mémoire disponible, ce qui réduit le nombre de lectures de disque requises et améliore les performances des requêtes.

work_mem

Le `work_mem` paramètre contrôle la quantité de mémoire utilisée par les requêtes pour les opérations de tri et de hachage. Sa valeur par défaut est de 4 Mo. Si une requête inclut plusieurs opérations, elle peut utiliser jusqu'à 4 Mo pour chaque opération. L'augmentation de la valeur de `work_mem` peut améliorer les performances des requêtes qui nécessitent un tri ou un hachage, car ces opérations nécessitent plus de mémoire. Toutefois, une valeur trop élevée de ce paramètre peut entraîner une utilisation excessive de la mémoire et une dégradation des performances.

Pour calculer la valeur optimale pour`work_mem`, vous pouvez utiliser la formule suivante :

```
work_mem = (available_memory / (max_connections * work_mem_fraction))
```

où `available_memory` est la quantité totale de mémoire disponible sur le serveur, max_connections est le nombre maximum de connexions autorisées et work_mem_fraction est une fraction qui détermine la quantité de mémoire disponible qui doit être allouée à chaque connexion.

AWS CLI syntaxe

La commande suivante change `work_mem` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify work_mem on a DB parameter group
aws rds modify-db-parameter-group \
   --db-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=work_mem,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify work_mem on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
   --db-cluster-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=work_mem,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 4 Mo

Exemple

Un outil d'analyse des réseaux sociaux traite une grande quantité de données, et les requêtes impliquant des opérations complexes de tri et de jointure entraînent un volume d'E/S élevé sur le disque et se répercutent sur le disque. Si vous augmentez la valeur `work_mem` de 4 Mo à 16 Mo, PostgreSQL peut utiliser davantage de mémoire pour ces opérations. Cela réduit le volume d'E/S et améliore les performances des requêtes.

maintenance_work_mem

Le `maintenance_work_mem` paramètre contrôle la quantité de mémoire utilisée par les opérations de maintenance telles que`VACUUM`,`ANALYZE`, et la création d'index. La valeur par défaut de ce paramètre dans Amazon RDS et Aurora est de 64 Mo.

Pour calculer la valeur appropriée pour ce paramètre, vous pouvez utiliser cette formule :

```
maintenance_work_mem = (total_memory - shared_buffers) / (max_connections * 5)
```

Aurora PostgreSQL Compatible Edition et Amazon RDS for PostgreSQL appliquent la formule suivante pour définir la valeur optimale :

```
GREATEST({DBInstanceClassMemory/63963136*1024},65536)
```

AWS CLI syntaxe

La commande suivante change `maintenance_work_mem` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify maintenance_work_mem on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=maintenance_work_mem,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify maintenance_work_mem on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
```

```
  --parameters
"ParameterName=maintenance_work_mem,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 64 Mo

Exemple

Votre application à grande échelle utilise une base de données PostgreSQL hébergée sur Aurora ou Amazon RDS. Vous remarquez que la base de données est lente et ne répond pas pendant les activités de maintenance telles que l'aspiration et l'indexation. Vous pouvez surveiller des indicateurs tels que l'utilisation de la mémoire, les durées des opérations de maintenance et l'utilisation du processeur afin de déterminer si la valeur actuelle de `maintenance_work_mem` est à l'origine de problèmes.

Pour déterminer la valeur optimale pour`maintenance_work_mem`, vous pouvez ajuster le paramètre et surveiller son impact. Si l'utilisation de la mémoire est constamment élevée ou si les durées de fonctionnement sont plus longues que prévu pendant les opérations de maintenance, une augmentation `maintenance_work_mem` peut être utile. À l'inverse, si l'utilisation du processeur est constamment élevée pendant les opérations de maintenance, une diminution `maintenance_work_mem` peut être utile. En effectuant des ajustements et des tests, vous pouvez trouver la valeur optimale `maintenance_work_mem` qui offre le meilleur équilibre entre l'utilisation de la mémoire, les durées des opérations de maintenance et l'utilisation du processeur.

Au cours de votre enquête, supposons que vous déterminez que la valeur par défaut de 64 Mo pour `maintenance_work_mem` est trop faible pour la taille de votre base de données. Par conséquent, les opérations de maintenance prennent plus de temps, entraînent des temps d'arrêt excessifs et ralentissent les performances de votre application. Pour résoudre ce problème, vous pouvez décider de régler le `maintenance_work_mem` paramètre en le faisant passer de 64 Mo à 512 Mo (valeur que vous identifiez comme étant la valeur optimale). L'application de cette modification peut améliorer les délais de vos opérations de maintenance des deux tiers. Par exemple, une opération d'aspiration qui prenait auparavant 30 minutes peut désormais ne prendre que 10 minutes. Grâce à cette optimisation, votre base de données peut désormais gérer les activités de maintenance de manière plus efficace.

random_page_cost

Le `random_page_cost` paramètre permet de déterminer le coût d'un accès aléatoire aux pages. Le planificateur de requêtes d'Amazon RDS et d'Aurora utilise ce paramètre, ainsi que d'autres statistiques relatives à la table, pour déterminer le plan le plus efficace pour exécuter une requête.

Les `random_page_cost` paramètres `seq_page_cost` et sont étroitement liés et sont généralement utilisés conjointement par le planificateur pour comparer les coûts des différentes méthodes d'accès et décider laquelle est la plus efficace. Par conséquent, si vous modifiez l'un de ces paramètres, vous devez également déterminer si l'autre paramètre doit être ajusté.

En général, le planificateur de requêtes essaie de minimiser le coût d'exécution d'une requête. Le coût est déterminé en combinant le nombre de pages lues sur le disque et la valeur de`random_page_cost`. Une valeur élevée de `random_page_cost` tend à favoriser les analyses séquentielles, tandis qu'une valeur inférieure tend à favoriser les analyses d'index. Une valeur inférieure tend également à favoriser les jointures par boucle imbriquée au lieu des jointures par hachage.

Le `random_page_cost` paramètre utilise la valeur par défaut du moteur PostgreSQL (4) sauf si une valeur est définie dans le groupe de paramètres ou dans la session locale. Vous pouvez ajuster cette valeur en fonction des caractéristiques spécifiques de votre serveur et de votre charge de travail. Si la plupart des index utilisés dans votre charge de travail se trouvent dans la mémoire ou dans le cache hiérarchisé Aurora, il `seq_page_cost` est approprié de `random_page_cost` remplacer la valeur de par une valeur proche de.

AWS CLI syntaxe

La commande suivante change `random_page_cost` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify random_page_cost on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=random_page_cost,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify random_page_cost on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
```

```
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=random_page_cost,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 4

Exemple

Supposons que vous disposiez d'une base de données qui stocke une grande quantité de données dans une table fréquemment interrogée avec des filtres sur des colonnes non indexées. Les requêtes sont longues à exécuter et le planificateur de requêtes ne sélectionne pas le plan le plus efficace pour accéder aux données.

Une façon d'améliorer les performances serait de diminuer le `random_page_cost` paramètre. Si vous le définissez sur 1, le coût de l'accès aléatoire aux pages sera quatre fois moins élevé que la valeur par défaut. Si vous mainteniez `random_page_cost` la valeur par défaut de 4, l'accès aléatoire aux pages serait quatre fois plus coûteux que l'accès séquentiel aux pages (tel que déterminé par le `seq_page_cost` paramètre, qui est de 1,0 par défaut). Cependant, dans ce cas précis, l'accès aléatoire aux pages peut en fait être beaucoup plus coûteux en fonction du type de stockage.

La diminution de la valeur du `random_page_cost` paramètre peut inciter le planificateur de requêtes à sélectionner un plan basé sur un index ou à utiliser une autre méthode d'accès mieux adaptée aux caractéristiques spécifiques de la table.

Nous vous recommandons de surveiller les performances des requêtes après avoir modifié le paramètre et d'apporter les ajustements nécessaires. Vous devez également vérifier le planificateur de requêtes avec une `EXPLAIN` instruction pour vérifier s'il sélectionne un plan efficace.

Il ne s'agit là que d'un exemple. Le réglage optimal dépend des caractéristiques spécifiques de votre charge de travail. Il ne s'agit également que d'un aspect du réglage des performances ; vous devez également prendre en compte d'autres paramètres et options de configuration susceptibles d'affecter les performances de vos requêtes.

seq_page_cost

Le `seq_page_cost` paramètre permet de déterminer le coût de l'accès séquentiel aux pages. Le planificateur de requêtes d'Amazon RDS et d'Aurora utilise ce paramètre, ainsi que d'autres statistiques relatives à la table, pour déterminer le plan le plus efficace pour exécuter une requête.

Les `random_page_cost` paramètres `seq_page_cost` et sont étroitement liés et sont généralement utilisés conjointement par le planificateur pour comparer les coûts des différentes méthodes d'accès et décider laquelle est la plus efficace. Par conséquent, si vous modifiez l'un de ces paramètres, vous devez également déterminer si l'autre paramètre doit être ajusté.

Lorsque l'on accède à une table de manière séquentielle, PostgreSQL peut utiliser le cache du système de fichiers du système d'exploitation pour accélérer l'accès. Par défaut, `seq_page_cost` il est défini sur 1.0, ce qui suppose que l'accès séquentiel aux pages est aussi économique que la lecture d'un seul bloc de disque. Si vous avez une table à laquelle vous accédez principalement de manière séquentielle, mais que l'accès au disque est lent car il est limité par un nombre réduit d'IOPS, vous souhaiterez peut-être augmenter la valeur de `seq_page_cost` pour refléter le coût supplémentaire lié à l'accès au disque.

La modification de la valeur de ce paramètre affecte toutes les requêtes exécutées dans le système. Nous vous recommandons donc de tester vos requêtes avec différentes valeurs afin de déterminer la valeur optimale pour votre cas d'utilisation spécifique.

AWS CLI syntaxe

La commande suivante change `seq_page_cost` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify seq_page_cost on a DB parameter group
aws rds modify-db-parameter-group \
   --db-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=seq_page_cost,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify seq_page_cost on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
   --db-cluster-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=seq_page_cost,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissezApplyMethod=immediate)

Valeur par défaut : 1.0

Exemple

Supposons que vous disposiez d'une base de données qui stocke une grande quantité de données dans une table à laquelle on accède principalement de manière séquentielle. La table est principalement utilisée pour les rapports, les requêtes s'exécutent très lentement et le planificateur de requêtes n'est pas en mesure de sélectionner le plan le plus efficace pour accéder aux données.

Une façon d'améliorer les performances serait de diminuer le `seq_page_cost` paramètre. La valeur par défaut est 1,0, ce qui suppose qu'un accès séquentiel à une page est aussi économique que la lecture d'un seul bloc de disque. Cependant, dans ce cas précis, l'accès séquentiel aux pages peut en fait être plus coûteux en raison du type de stockage (en fonction des E/S). Si vous définissez cette valeur `seq_page_cost` sur 0,5, le coût de l'accès séquentiel aux pages est deux fois moins élevé que la valeur par défaut. Cette modification peut rendre le planificateur de requêtes plus susceptible de sélectionner un plan utilisant une méthode d'accès séquentiel mieux adaptée aux caractéristiques spécifiques de la table.

Nous vous recommandons de surveiller les performances des requêtes après avoir modifié le paramètre et d'apporter les ajustements nécessaires. Vous devez également vérifier le plan de requête à l'aide d'une `EXPLAIN` instruction pour vérifier s'il sélectionne un plan efficace.

Il ne s'agit là que d'un exemple. Le réglage optimal dépend des caractéristiques spécifiques de votre charge de travail. Il ne s'agit également que d'un aspect du réglage des performances ; vous devez également prendre en compte les autres paramètres et options de configuration qui affectent les performances de vos requêtes.

track_activity_query_size

Le `track_activity_query_size` paramètre contrôle la taille de la chaîne de requête enregistrée pour chaque session active dans la `pg_stat_activity` vue. Par défaut, seuls les 1 024 premiers octets de la chaîne de requête sont enregistrés dans Amazon RDS pour PostgreSQL, et 4 096 octets sont enregistrés dans la version compatible avec Aurora PostgreSQL. Si vous souhaitez enregistrer des requêtes plus longues, vous pouvez définir une valeur plus élevée pour ce paramètre.

AWS CLI syntaxe

La commande suivante change `track_activity_query_size` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify track_activity_query_size on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=track_activity_query_size,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify track_activity_query_size on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=track_activity_query_size,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : statique (l'application des modifications nécessite un redémarrage)

Valeur par défaut : 1 024 octets (Amazon RDS pour PostgreSQL), 4 096 octets (compatible avec Aurora PostgreSQL)

Exemple

Les performances de votre base de données Amazon RDS for PostgreSQL sont lentes et vous pensez que le problème est peut-être lié à des requêtes de longue durée. Vous pouvez approfondir vos recherches en enregistrant les requêtes plus longues dans la `pg_stat_activity` vue.

L'augmentation de la valeur du `track_activity_query_size` paramètre peut entraîner une augmentation de la journalisation, ce qui peut avoir un impact sur les performances de la base de données. Nous vous recommandons de rétablir la valeur par défaut de 1 024 du paramètre une fois le problème résolu.

idle_in_transaction_session_timeout

Le `idle_in_transaction_session_timeout` paramètre contrôle la durée pendant laquelle une transaction inactive attend avant d'être arrêtée.

La valeur par défaut de ce paramètre dans Amazon RDS for PostgreSQL et Aurora PostgreSQL compatible est de 86 400 000 millisecondes.

AWS CLI syntaxe

La commande suivante change `idle_in_transaction_session_timeout` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify idle_in_transaction_session_timeout on a DB parameter group
aws rds modify-db-parameter-group \
   --db-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=idle_in_transaction_session_timeout,ParameterValue=<new_value>,ApplyMethod=imme

# Modify idle_in_transaction_session_timeout on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
   --db-cluster-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=idle_in_transaction_session_timeout,ParameterValue=<new_value>,ApplyMethod=imme
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 86 400 000 millisecondes (compatible avec Aurora PostgreSQL)

Exemple

Vous disposez d'une application de commerce électronique qui traite les commandes en ligne. L'application utilise une base de données PostgreSQL hébergée sur Amazon RDS ou Aurora. Chaque fois qu'un client passe une commande, l'application lance une nouvelle transaction pour mettre à jour l'inventaire et les enregistrements de commandes.

Si une transaction reste inactive pendant une longue période, cela peut empêcher d'autres transactions d'accéder aux mêmes enregistrements, ce qui peut entraîner des problèmes de performances et même une interruption de l'application. En outre, les transactions inactives qui ne sont pas correctement arrêtées peuvent consommer de précieuses ressources système telles que la mémoire et le processeur.

Pour éviter de tels problèmes, vous pouvez définir le `idle_in_transaction_session_timeout` paramètre sur une valeur adaptée à votre application. Par exemple, vous pouvez le définir sur 5 minutes (300 secondes) afin que toute transaction restée inactive pendant plus de 5 minutes soit automatiquement arrêtée. Cela permet de garantir que les ressources du système sont utilisées

efficacement et que l'application peut traiter un grand nombre de commandes sans ralentir. En définissant la valeur appropriée pour`idle_in_transaction_session_timeout`, vous pouvez garantir que votre application fonctionne de manière optimale sur Amazon RDS ou Aurora.

statement_timeout

Le `statement_timeout` paramètre définit la durée maximale pendant laquelle une requête peut être exécutée avant son arrêt.

AWS CLI syntaxe

La commande suivante change `statement_timeout` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify statement_timeout on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=statement_timeout,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify statement_timeout on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=statement_timeout,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 0 millisecondes (aucun délai d'attente)

Exemple

Votre application Web permet aux utilisateurs de faire des recherches dans une vaste base de données de produits. Les requêtes de recherche peuvent parfois prendre beaucoup de temps, ce qui ralentit les temps de réponse des utilisateurs. Pour résoudre ce problème, vous pouvez définir le `statement_timeout` paramètre sur une valeur faible, telle que 10 secondes, ce qui forcerait l'arrêt de toute requête de plus de 10 secondes.

Cela peut sembler une mesure drastique, mais elle peut en fait être très efficace pour améliorer les performances. Dans de nombreux cas, les requêtes de longue durée sont dues à des requêtes SQL mal optimisées ou à des index inefficaces. En définissant une `statement_timeout` valeur faible, vous pouvez identifier ces requêtes problématiques et prendre des mesures pour les optimiser.

Supposons, par exemple, que vous découvriez qu'une requête de recherche précise expire régulièrement. Vous pouvez utiliser des outils tels que `EXPLAIN` et `EXPLAIN ANALYZE` pour analyser la requête et identifier les éventuels goulots d'étranglement liés aux performances. Une fois le problème identifié, vous pouvez prendre des mesures pour optimiser la requête en ajoutant de nouveaux index, en réécrivant la requête ou en utilisant un algorithme de recherche différent. En analysant et en optimisant en permanence vos requêtes SQL de cette manière, vous pouvez améliorer considérablement les performances de votre application.

search_path

Le `search_path` paramètre détermine l'ordre dans lequel les objets sont recherchés dans les schémas dans les instructions SQL. La valeur par défaut est`$user, public`, ce qui signifie que PostgreSQL recherche les objets d'abord dans le schéma correspondant au nom de l'utilisateur, puis dans le schéma public.

Si vous disposez d'un grand nombre de schémas ou si vous devez accéder à des objets dans un schéma spécifique, la modification du `search_path` paramètre peut contribuer à améliorer les performances. Lorsque vous définissez `search_path` un schéma spécifique, PostgreSQL peut trouver les objets plus rapidement sans avoir à effectuer de recherche dans plusieurs schémas.

Pour modifier le `search_path` paramètre dans Amazon RDS et Aurora, vous pouvez utiliser la commande suivante pour le `ROLE` niveau :

```
ALTER ROLE <username> SET search_path = <schema>;
```

AWS CLI syntaxe

La commande suivante change `search_path` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify search_path on a DB parameter group
aws rds modify-db-parameter-group \
```

```
   --db-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=search_path,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify search_path on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
   --db-cluster-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=search_path,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `$user, public`

Exemple

Vous disposez d'une application mutualisée avec des schémas distincts pour chaque locataire qui utilise une base de données compatible Amazon RDS for PostgreSQL ou Aurora PostgreSQL, et vous devez exécuter une requête qui implique de joindre des données provenant de plusieurs schémas.

Par défaut, Amazon RDS et Aurora utilisent le chemin de recherche pour déterminer le schéma à utiliser pour une table donnée. Le chemin de recherche est une liste de noms de schéma que PostgreSQL recherche dans l'ordre lorsque vous faites référence à une table sans qualifier le nom du schéma. Par défaut, Amazon RDS et Aurora recherchent d'abord la table dans le schéma portant le même nom que l'utilisateur actuel, puis dans le schéma public.

Supposons que vous souhaitiez exécuter une requête qui implique de joindre des tables provenant de plusieurs schémas, nommés `tenant1``tenant2`, et`tenant3`. Pour utiliser les schémas de locataires, vous pouvez inclure les noms des schémas dans la requête :

```
SELECT *
FROM tenant1.table1
JOIN tenant2.table2 ON tenant1.table1.id = tenant2.table2.id
JOIN tenant3.table3 ON tenant2.table2.id = tenant3.table3.id;
```

Cependant, une méthode plus efficace consiste à modifier le `search_path` paramètre pour inclure les schémas de locataires à l'aide des commandes de la section de AWS CLI syntaxe. Vous pouvez également utiliser la `SET` commande dans une session PostgreSQL :

```
SET search_path = tenant1, tenant2, tenant3, public;
```

Vous pouvez ensuite écrire la requête sans qualifier les noms des schémas :

```
SELECT *
FROM table1
JOIN table2 ON table1.id = table2.id
JOIN table3 ON table2.id = table3.id;
```

Cela peut rendre votre requête plus concise et plus facile à lire, et cela peut également simplifier le code de votre application si vous avez de nombreuses requêtes impliquant la jonction de tables provenant de plusieurs schémas.

max_connections

Le `max_connections` paramètre définit le nombre maximal de connexions simultanées pour votre base de données PostgreSQL.

AWS CLI syntaxe

La commande suivante change `max_connections` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify max_connections on a DB parameter group
aws rds modify-db-parameter-group \
--db-parameter-group-name <parameter_group_name> \
--parameters
 "ParameterName=max_connections,ParameterValue=<new_value>,ApplyMethod=pending-reboot"

# Modify max_connections on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
--db-cluster-parameter-group-name <parameter_group_name> \
--parameters
 "ParameterName=max_connections,ParameterValue=<new_value>,ApplyMethod=pending-reboot"
```

Type : statique (l'application des modifications nécessite un redémarrage)

Valeur par défaut : `LEAST(DBInstanceClassMemory/9531392,5000)` connexions

Pour optimiser l'utilisation d'`max_connections`Amazon RDS ou Aurora et minimiser son impact sur les performances, prenez en compte les meilleures pratiques suivantes :

- Définissez la valeur du paramètre en fonction des ressources système disponibles.
- Surveillez l'utilisation de la connexion pour éviter d'atteindre rapidement la limite.
- Utilisez le regroupement de connexions pour réduire le nombre de connexions requises.
- Utilisez le proxy Amazon RDS pour le regroupement des connexions.

Lorsque vous activez Amazon RDS for PostgreSQL ou Amazon Aurora PostgreSQL compatible, tenez compte des types d'instances disponibles et des ressources qui leur sont allouées, et concentrez-vous sur la mémoire et la capacité du processeur. `max_connections` Les informations relatives au stockage et aux E/S sont gérées par AWS, ce qui vous permet de surveiller les caractéristiques générales de la charge de travail et les indicateurs du système, par exemple `FreeableMemory` via Amazon CloudWatch ou la console Amazon RDS pour vérifier que la mémoire est suffisante pour les connexions. Surveillez `CPUUtilization` les valeurs élevées, qui peuvent indiquer que des ajustements sont nécessaires. Évitez de définir un paramètre `max_connections` trop élevé, car cela peut avoir un impact sur l'utilisation de la mémoire et potentiellement influencer les E/S de manière indirecte. N'oubliez pas que chaque connexion consomme de la mémoire. Pour trouver le bon équilibre, augmentez lentement `max_connections` et observez comment cela affecte votre système. Surveillez les signes d'un ralentissement des performances ou d'une utilisation accrue du processeur. Vérifiez si votre application fonctionne toujours correctement. Utilisez des fonctionnalités telles que les répliques de lecture dans Aurora pour répartir le trafic de lecture et réduire la charge sur l'instance principale. Passez en revue et ajustez `max_connections` régulièrement en fonction des modèles d'utilisation observés afin de garantir des performances de base de données optimales dans le cadre des contraintes de ressources données.

Réglage des paramètres d'autovacuum

Les bases de données Amazon RDS for PostgreSQL et la compatibilité avec Aurora PostgreSQL nécessitent une maintenance périodique connue sous le nom de mise sous vide. Autovacuum est un utilitaire PostgreSQL intégré qui supprime les données obsolètes ou inutiles afin de libérer de l'espace dans la base de données. Le processus d'autovacuum exécute la `VACUUM` commande en arrière-plan à intervalles réguliers.

Le réglage des paramètres d'autovacuum est une étape cruciale pour maintenir les performances, la stabilité et la disponibilité de votre système de base de données compatible avec Amazon RDS for PostgreSQL ou Aurora PostgreSQL. En ajustant les paramètres d'autovacuum en fonction de votre charge de travail et de la taille de votre base de données, vous pouvez optimiser les performances du processus d'aspiration automatique et réduire son impact sur les ressources du système, améliorant ainsi la santé globale de votre base de données.

Outre le réglage des paramètres d'autovacuum, il est important de surveiller les performances de votre base de données et de ses composants à l'aide des outils et des métriques disponibles dans Amazon RDS et Aurora. En surveillant les indicateurs de performance tels que le surchargement, l'espace libre et le temps d'exécution des requêtes, vous pouvez identifier les problèmes potentiels avant qu'ils ne deviennent graves et prendre les mesures appropriées pour les résoudre.

Cette section aborde les sujets et paramètres suivants relatifs à l'aspirateur automatique :

- Commandes VACUUM et ANALYZE
- Vérification de la présence de ballonnements
- aspirateur automatique
- autovacuum_work_mem
- autovacuum_naptime
- autovacuum_max_workers
- autovacuum_vacuum_scale_factor
- seuil d'aspiration automatique
- autovacuum_analyze_scale_factor
- seuil d'analyse automatique
- autovacuum_vacuum_cost_limit

Pour plus d'informations sur Autovacuum, consultez les liens suivants :

- Comprendre l'autovacuum dans les environnements Amazon RDS for PostgreSQL (article de blog)
- Utilisation de l'aspirateur automatique PostgreSQL sur Amazon RDS pour PostgreSQL (documentation Amazon RDS)
- Aspiration parallèle dans Amazon RDS pour PostgreSQL et Amazon Aurora PostgreSQL (article de blog)

Commandes VACUUM et ANALYZE

`VACUUM`collecte et analyse éventuellement une base de données. Pour la plupart des applications, il suffit de laisser le démon autovacuum effectuer l'aspiration. Cependant, certains administrateurs souhaiteront peut-être modifier les paramètres de base de données pour Autovacuum, ou compléter ou remplacer les activités du démon en utilisant des `VACUUM` commandes gérées manuellement qui peuvent être exécutées selon un planificateur.

`VACUUM`récupère le stockage occupé par des tuples morts. Dans les opérations PostgreSQL standard, lorsque des tuples sont supprimés ou rendus obsolètes par une mise à jour, ils ne sont pas physiquement supprimés des tables tant `VACUUM` qu'une opération n'est pas effectuée. Par conséquent, nous vous recommandons de les exécuter `VACUUM` régulièrement, en particulier sur les tables fréquemment mises à jour.

Le réglage `VACUUM` des paramètres est particulièrement important dans la compatibilité avec Amazon RDS for PostgreSQL et Aurora PostgreSQL, car ces services de base de données gérés présentent des caractéristiques différentes de celles des bases de données PostgreSQL autogérées. Ces différences peuvent affecter les performances des opérations d'aspiration. Le réglage `VACUUM` des paramètres est essentiel pour optimiser l'utilisation des ressources et garantir que les opérations sous vide n'affectent pas négativement les performances et la disponibilité de votre système de base de données.

Voici certains des paramètres que vous pouvez utiliser avec la `VACUUM` commande dans Aurora PostgreSQL compatible et Amazon RDS for PostgreSQL :

- `FULL`
- `FREEZE`
- `VERBOSE`
- `ANALYZE`

- `DISABLE_PAGE_SKIPPING`
- `table_name`
- `column_name`

`VACUUM ANALYZE`effectue une `VACUUM` opération suivie d'une `ANALYZE` opération pour chaque table sélectionnée. Il constitue un moyen efficace d'effectuer la maintenance de routine.

L'utilisation de la `VACUUM` commande sans l'`FULL`option permet de récupérer de l'espace à réutiliser. Elle ne nécessite pas de verrou exclusif sur la table. Vous pouvez donc exécuter cette commande lors d'opérations de lecture et d'écriture standard. Cependant, dans la plupart des cas, la commande ne restitue pas d'espace supplémentaire au système d'exploitation, mais le maintient disponible pour être réutilisé dans la même table. `VACUUM FULL`réécrit l'intégralité du contenu de la table dans un nouveau fichier disque sans espace supplémentaire, et autorise le retour de l'espace inutilisé au système d'exploitation. Ce formulaire est beaucoup plus lent et nécessite un `ACCESS EXCLUSIVE` verrou sur chaque table.

Pour obtenir des informations complètes sur ces paramètres, consultez la documentation de PostgreSQL.

Dans Aurora et Amazon RDS, autovacuum est un processus daemon (utilitaire d'arrière-plan) qui exécute régulièrement `ANALYZE` les commandes `VACUUM` and pour nettoyer les données redondantes dans la base de données et le serveur. Même si vous utilisez l'aspirateur automatique, nous vous recommandons de consulter et de régler les paramètres d'aspiration automatique décrits dans les sections suivantes afin de garantir des performances optimales.

Vérification de la présence de ballonnements

La requête SQL suivante examine chaque table du schéma XML et identifie les lignes mortes (tuples) qui gaspillent de l'espace disque :

```
SELECT schemaname || '.' || relname as tuplename,
    n_dead_tup,
    (n_dead_tup::float / n_live_tup::float) * 100 as pfrag
FROM pg_stat_user_tables
WHERE schemaname = 'xml' and n_dead_tup > 0 and n_live_tup > 0 order by pfrag desc;
```

Si cette requête renvoie un pourcentage élevé (pfrag) de tuples morts, vous pouvez utiliser la `VACUUM` commande pour récupérer de l'espace.

Pour surveiller la taille des données avant et après les transactions, exécutez la requête suivante dans le shell après vous être connecté à une base de données spécifique :

```
SELECT pg_size_pretty(pg_relation_size('table_name'));
```

autovacuum

Vous pouvez définir l'autovacuum globalement à l'aide du paramètre de `autovacuum` configuration, ou vous pouvez le modifier par table en définissant la `autovacuum_enabled` colonne de la `pg_class` table sur `true` ou `false` pour une table spécifique.

Lorsque vous activez Autovacuum sur une table, le serveur de base de données analyse régulièrement la table à la recherche de lignes et de tuples morts et les supprime en arrière-plan, sans aucune intervention de l'administrateur de base de données. Cela permet de réduire la taille de la table, d'améliorer les performances des requêtes et de réduire la taille des sauvegardes.

AWS CLI syntaxe

La commande suivante active un groupe `autovacuum` de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters "ParameterName=autovacuum,ParameterValue=true,ApplyMethod=immediate"

# Modify autovacuum on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters "ParameterName=autovacuum,ParameterValue=true,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : Activé

Vous pouvez également désactiver ou activer l'autovacuum sur une table spécifique en utilisant psql :

```
ALTER TABLE <table_name> SET (autovacuum_enabled = true);
```

Une utilisation excessive de l'aspirateur peut affecter les performances. Il est donc important de surveiller les performances du processus d'aspiration automatique ainsi que les performances de votre base de données, et d'ajuster les paramètres selon les besoins.

Exemple

Votre base de données PostgreSQL possède une table qui reçoit un grand nombre d'opérations d'écriture et de suppression. Sans autovacuum, cette table finirait par être remplie de lignes mortes (c'est-à-dire de lignes marquées pour suppression mais qui n'ont pas encore été physiquement supprimées de la table). Ces lignes mortes occuperaient de l'espace sur le disque, ralentiraient les requêtes et augmenteraient la taille des sauvegardes. Vous pouvez activer l'aspirateur automatique sur la table pour rechercher automatiquement les lignes mortes et les supprimer afin d'atténuer ces problèmes.

autovacuum_work_mem

`autovacuum_work_mem`est un paramètre de configuration PostgreSQL qui contrôle la quantité de mémoire utilisée par le processus d'aspiration automatique lorsqu'il exécute des tâches de maintenance de tables telles que l'aspiration ou l'analyse.

Dans Aurora et Amazon RDS, vous pouvez ajuster la valeur de `autovacuum_work_mem` pour optimiser les performances.

AWS CLI syntaxe

La commande suivante active un groupe `autovacuum_work_mem` de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum_work_mem on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_work_mem,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify autovacuum_work_mem on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_work_mem,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissezApplyMethod=immediate)

Valeur par défaut : GREATEST({DBInstanceClassMemory/32768},131072) Ko dans la version compatible avec Aurora PostgreSQL, 64 Mo dans Amazon RDS for PostgreSQL. Toutefois, la valeur par défaut peut varier en fonction de la version spécifique d'Amazon RDS ou d'Aurora que vous utilisez.

Exemple

Votre base de données Amazon RDS for PostgreSQL possède une grande table fréquemment mise à jour. Au fil du temps, vous remarquez que la base de données devient plus lente et vous pensez que l'autovacuum prend trop de temps à terminer.

Dans le cadre de votre enquête, vous consultez les journaux du système, vous utilisez la pg_stat_activity vue pour voir les requêtes et les processus en cours d'exécution, vous consultez la pg_stat_user_tables vue pour voir les statistiques de chaque table, vous utilisez la pg_settings vue pour comparer la valeur de autovacuum_work_mem à la mémoire disponible sur le système et vous surveillez l'utilisation de la mémoire pour détecter les pics d'utilisation. Après avoir rassemblé ces informations, vous pouvez autovacuum_work_mem définir la valeur optimale dont votre charge de travail a besoin. Pour trouver le juste équilibre entre l'utilisation de la mémoire et les performances, vous pouvez décider de le régler sur un quart de la mémoire disponible sur le système. Après avoir modifié la valeur, vous surveillez les performances de la base de données et vous constaterez peut-être qu'Autovacuum s'exécute beaucoup plus rapidement qu'auparavant et que votre base de données fonctionne globalement plus rapidement.

autovacuum_naptime

Le autovacuum_naptime paramètre contrôle l'intervalle de temps entre les cycles successifs du processus d'aspiration automatique. La valeur par défaut est de 15 secondes pour Amazon RDS pour PostgreSQL et de 5 secondes pour Aurora PostgreSQL compatible.

Supposons, par exemple, que votre base de données Amazon RDS for PostgreSQL possède une table qui reçoit un grand nombre d'opérations d'écriture et de suppression. Si vous conservez le paramètre par défaut, les analyses fréquentes par aspiration automatique perturberont cette table hautement transactive. Si vous définissez ce paramètre sur une valeur élevée, l'intervalle entre les scans successifs sera plus long et les lignes mortes seront supprimées moins fréquemment.

Vous pouvez l'utiliser `autovacuum_naptime` pour gérer la charge causée par le processus de mise sous vide, en particulier si vous avez un serveur occupé qui a déjà une charge de processeur ou d'E/S élevée. Plus la durée de la sieste est longue, moins l'aspirateur automatique s'exécute fréquemment, ce qui réduit la charge sur le serveur. Toutefois, la définition `autovacuum_naptime` d'une valeur très élevée peut entraîner la croissance de vos tables PostgreSQL et l'accumulation de lignes mortes, ce qui entraîne une baisse des performances. Nous vous recommandons de surveiller les performances du processus d'aspiration automatique et d'ajuster le `autovacuum_naptime` réglage selon les besoins.

AWS CLI syntaxe

La commande suivante change `autovacuum_naptime` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum_naptime on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_naptime,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify autovacuum_naptime on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_naptime,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 15 secondes (Amazon RDS pour PostgreSQL), 5 secondes (compatible avec Aurora PostgreSQL)

autovacuum_max_workers

Le `autovacuum_max_workers` paramètre contrôle le nombre maximum de processus de travail que le processus d'aspiration automatique peut créer. Chaque processus de travail est chargé de passer l'aspirateur ou d'analyser une seule table.

Supposons, par exemple, que vous disposiez d'une base de données volumineuse contenant de nombreuses tables fréquemment mises à jour et supprimées. Si vous définissez une valeur faible telle que 1, une seule table peut être aspirée à la fois et le nettoyage de toutes les tables prend plus de temps. `autovacuum_max_workers` Si vous définissez `autovacuum_max_workers` une valeur élevée telle que 8, vous pouvez aspirer jusqu'à huit tables simultanément. Cela peut accélérer le processus de nettoyage pour les bases de données contenant de nombreuses tables.

AWS CLI syntaxe

La commande suivante change `autovacuum_max_workers` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum_max_workers on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_max_workers,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify autovacuum_max_workers on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_max_workers,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : statique (l'application des modifications nécessite un redémarrage)

Valeur par défaut : `GREATEST(DBInstanceClassMemory/64371566592,3)` travailleurs

L'augmentation du `autovacuum_max_workers` paramètre peut augmenter la charge sur le serveur, ce qui peut avoir un impact sur les performances si vous ne disposez pas de suffisamment de ressources. Le paramètre optimal dépend des exigences spécifiques de votre base de données, de sa taille et du nombre de tables qu'elle contient. Nous vous recommandons de tester différentes valeurs et de surveiller les performances afin de trouver le réglage optimal pour votre cas d'utilisation.

autovacuum_vacuum_scale_factor

Le paramètre `autovacuum_vacuum_scale_factor` de configuration contrôle le degré d'agressivité du processus d'aspiration automatique lors de l'aspiration d'une table.

Le facteur d'échelle de vide est une fraction du nombre total de tuples d'une table qui doit être modifiée avant que Autovacuum ne nettoie la table. La valeur par défaut est 0,1 (c'est-à-dire que 10 % des tuples doivent être modifiés). Par exemple, si une table contient 1 000 000 tuples et que 100 000 de ces tuples sont marqués comme morts ou supprimés, Autovacuum vide la table en fonction de la valeur de comme facteur de `autovacuum_vacuum_threshold` contrôle.

Le `autovacuum_vacuum_scale_factor` paramètre vous permet de contrôler la fréquence d'exécution du processus de mise sous vide. Si une table fait l'objet de nombreuses opérations d'écriture, vous pouvez réduire le facteur d'échelle du vide afin que l'autovacuum s'exécute plus fréquemment, tout en réduisant la taille de la table. À l'inverse, si une table reçoit peu d'opérations d'écriture, vous souhaiterez peut-être augmenter le facteur d'échelle du vide afin de réduire la fréquence d'exécution de l'autovacuum et d'économiser des ressources.

AWS CLI syntaxe

La commande suivante change `autovacuum_vacuum_scale_factor` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum_vacuum_scale_factor on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_vacuum_scale_factor,ParameterValue=<new_value>,ApplyMethod=immediate

# Modify autovacuum_vacuum_scale_factor on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_vacuum_scale_factor,ParameterValue=<new_value>,ApplyMethod=immediate
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 0.1

Le `autovacuum_vacuum_scale_factor` paramètre fonctionne conjointement avec `autovacuum_vacuum_threshold` les `autovacuum_vacuum_cost_limit, and autovacuum_naptime` paramètres. Pour plus d'informations sur ce paramètre, consultez le billet de AWS blog Understanding autovacuum in Amazon RDS for PostgreSQL environments.

autovacuum_vacuum_threshold

Le `autovacuum_vacuum_threshold` paramètre contrôle le nombre minimum d'opérations de mise à jour ou de suppression de tuples qui doivent avoir lieu sur une table avant qu'Autovacuum ne l'aspire. Ce paramètre peut être utile pour éviter de passer inutilement l'aspirateur sur les tables où le taux d'opérations de ce type n'est pas élevé. La valeur par défaut est 50, qui est la valeur par défaut du moteur PostgreSQL, à la fois pour Amazon RDS for PostgreSQL et pour Aurora PostgreSQL compatible.

Supposons, par exemple, que vous disposiez d'un tableau de 100 000 lignes défini sur 50. `autovacuum_vacuum_threshold` Si la table ne reçoit que 49 mises à jour ou suppressions, Autovacuum ne la vide pas. Si la table reçoit 50 mises à jour ou suppressions ou plus, Autovacuum la vide en fonction de la valeur `autovacuum_vacuum_scale_factor` multipliée par le nombre de lignes de la table comme facteur de contrôle.

Si ce paramètre est trop élevé, la table s'agrandit et les lignes mortes s'accumulent, ce qui peut affecter les performances.

AWS CLI syntaxe

La commande suivante change `autovacuum_vacuum_threshold` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum_vacuum_threshold on a DB parameter group
aws rds modify-db-parameter-group \
   --db-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=autovacuum_vacuum_threshold,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify autovacuum_vacuum_threshold on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
   --db-cluster-parameter-group-name <parameter_group_name> \
   --parameters
 "ParameterName=autovacuum_vacuum_threshold,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 50 opérations

Le `autovacuum_vacuum_threshold` paramètre fonctionne conjointement avec `autovacuum_vacuum_scale_factor` les `autovacuum_vacuum_cost_limit, and autovacuum_naptime` paramètres. Les paramètres optimaux dépendent des exigences spécifiques de votre base de données et de la taille de votre table.

Pour plus d'informations sur ce paramètre, consultez le billet de AWS blog Understanding autovacuum in Amazon RDS for PostgreSQL environments.

autovacuum_analyze_scale_factor

Le `autovacuum_analyze_scale_factor` paramètre contrôle le degré d'agressivité du processus d'aspiration automatique lors de l'analyse (collecte de statistiques) sur la distribution des données dans un tableau.

Le processus d'aspiration automatique utilise ce paramètre pour calculer un seuil basé sur le nombre de tuples dans une table. Si le nombre d'insertions, de mises à jour ou de suppressions de tuples dépasse ce seuil, Autovacuum analyse le tableau. La valeur par défaut est 0,05 (c'est-à-dire que 5 % des tuples doivent être modifiés) à la fois pour Amazon RDS for PostgreSQL et pour Aurora PostgreSQL compatible.

Supposons, par exemple, que votre table comporte 1 000 000 tuples et que vous conserviez la `autovacuum_analyze_scale_factor` valeur par défaut à 0,05. Si la table reçoit 50 000 mises à jour ou suppressions ou plus, Autovacuum l'aspire en fonction de la `autovacuum_analyze_threshold` valeur et en ajoutant le nombre de lignes de la table comme facteur de contrôle.

AWS CLI syntaxe

La commande suivante change `autovacuum_analyze_scale_factor` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum_analyze_scale_factor on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_analyze_scale_factor,ParameterValue=<new_value>,ApplyMethod=immediat

# Modify autovacuum_analyze_scale_factor on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
```

```
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
"ParameterName=autovacuum_analyze_scale_factor,ParameterValue=<new_value>,ApplyMethod=immediat
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 0,05 (5 %)

Il est essentiel que le planificateur de requêtes collecte des statistiques afin de prendre des décisions éclairées, telles que la manière d'accéder aux données et de les organiser. Nous vous recommandons donc de surveiller les performances du processus d'aspiration automatique et d'ajuster les paramètres selon les besoins pour garantir que les statistiques sont à jour.

Le `autovacuum_analyze_scale_factor` paramètre fonctionne conjointement avec `autovacuum_analyze_threshold` les `autovacuum_analyze_cost_limit, and autovacuum_naptime` paramètres. Le réglage optimal dépend des exigences spécifiques de votre base de données et de la taille de votre table, ainsi que de la fréquence des mises à jour. Pour plus d'informations sur ce paramètre, consultez le billet de AWS blog Understanding autovacuum in Amazon RDS for PostgreSQL environments.

autovacuum_analyze_threshold

Le `autovacuum_analyze_threshold` paramètre est similaire à`autovacuum_vacuum_threshold`. Il contrôle le nombre minimum d'insertions, de mises à jour ou de suppressions de tuples qui doivent avoir lieu sur une table avant qu'Autovacuum ne l'analyse. Ce paramètre peut être utile pour éviter de passer inutilement l'aspirateur sur les tables où le taux d'opérations de ce type n'est pas élevé. La valeur par défaut est 50, qui est la valeur par défaut du moteur PostgreSQL, à la fois pour Amazon RDS for PostgreSQL et pour Aurora PostgreSQL compatible.

Supposons, par exemple, que vous ayez une table de 100 000 lignes et que vous conserviez la `autovacuum_analyze_threshold` valeur par défaut à 50. Si la table ne reçoit que 49 insertions, mises à jour ou suppressions, Autovacuum ne l'analysera pas. Si la table reçoit 50 insertions, mises à jour ou suppressions ou plus, Autovacuum l'analysera en conservant la valeur `autovacuum_analyze_scale_factor` multipliée par le nombre de lignes du tableau comme facteur de contrôle.

AWS CLI syntaxe

La commande suivante change `autovacuum_analyze_threshold` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum_analyze_threshold on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_analyze_threshold,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify autovacuum_analyze_threshold on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_analyze_threshold,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 50 opérations

Ce paramètre fonctionne conjointement avec le `autovacuum_analyze_scale_factor` paramètre, prenez donc en compte les deux paramètres lorsque vous configurez l'aspirateur automatique.

Il est essentiel que le planificateur de requêtes collecte des statistiques afin de prendre des décisions éclairées, par exemple sur la manière d'accéder aux données et de les organiser. Une `autovacuum_analyze_threshold` valeur trop élevée peut rendre les statistiques obsolètes, ce qui peut entraîner de mauvaises performances. Nous vous recommandons de surveiller les performances du processus d'aspiration automatique et d'ajuster les paramètres selon les besoins.

Pour plus d'informations sur ce paramètre, consultez le billet de AWS blog Understanding autovacuum in Amazon RDS for PostgreSQL environments.

autovacuum_vacuum_cost_limit

Le `autovacuum_vacuum_cost_limit` paramètre contrôle la quantité de ressources CPU et d'E/S qu'un aspirateur automatique peut consommer.

La limitation de l'utilisation des ressources par les processus d'autovacuum peut aider à éviter qu'ils ne consomment trop d'E/S sur le processeur ou sur le disque, ce qui pourrait avoir un impact sur les

performances des autres requêtes exécutées sur le même système. Le paramètre spécifie une limite de coût, qui est une unité de travail que le travailleur est autorisé à effectuer avant de devoir faire une pause et vérifier si elle est toujours inférieure à la limite. Par exemple, si le paramètre est défini sur 2 000, un collaborateur est autorisé à traiter 2 000 unités de travail avant de faire une pause.

Vous pouvez définir le `autovacuum_vacuum_cost_limit` paramètre à l'aide de la `SET` commande dans une session PostgreSQL ou à l'aide d'une commande. AWS CLI

AWS CLI syntaxe

La commande suivante change `autovacuum_vacuum_cost_limit` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify autovacuum_vacuum_cost_limit on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_vacuum_cost_limit,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify autovacuum_vacuum_cost_limit on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=autovacuum_vacuum_cost_limit,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `GREATEST({log(DBInstanceClassMemory/21474836480)*600},200)` unités de travail

Si vous définissez une valeur `autovacuum_vacuum_cost_limit` trop élevée, le processus d'autovacuum risque de consommer trop de ressources et de ralentir les autres requêtes. Si vous le réglez trop bas, le processus d'aspiration automatique risque de ne pas récupérer suffisamment d'espace, ce qui agrandira la table au fil du temps. Il est essentiel de trouver le bon équilibre qui convient à votre système.

Ce paramètre affecte uniquement le processus d'aspiration automatique, et non les `VACUUM` commandes manuelles. De plus, cela ne s'applique qu'aux processus d'aspiration automatique pour`VACUUM`, mais pas pour`ANALYZE`.

Réglage des paramètres de journalisation

Le réglage des paramètres de journalisation dans PostgreSQL permet de s'assurer que vous collectez les bonnes informations sans générer de gros journaux qui surchargent votre système.

L'optimisation des paramètres de journalisation est essentielle pour équilibrer les détails des journaux avec les performances du système et l'utilisation du disque. Vous pouvez personnaliser les paramètres de journalisation suivants afin de capturer le niveau de détail approprié dans les journaux, de diagnostiquer les problèmes et d'enquêter efficacement sur les incidents tout en minimisant l'impact sur les performances du système et l'utilisation du disque.

- rds.force_autovacuum_logging
- rds.force_admin_logging_level
- durée du journal
- log_min_duration_statement
- log_error_verbosité
- log_statement
- log_statement_stats
- log_min_error_statement
- log_min_messages
- log_temp_files
- log_connections
- log_déconnexions

Ces paramètres sont décrits plus en détail dans les sections suivantes.

> **Warning**
>
> Les meilleurs paramètres pour ces paramètres dépendent des politiques et des exigences de conformité de votre organisation. Cependant, l'activation des paramètres de journalisation peut générer un grand nombre de journaux et de messages, ce qui peut utiliser l'espace de stockage et affecter les performances, en particulier pour une base de données très chargée. Nous vous recommandons d'utiliser ces paramètres avec précaution. Par exemple, vous

pouvez décider de les activer temporairement pour limiter un problème lié à une instruction SQL peu performante, et de les désactiver une fois la période de surveillance terminée.

rds.force_autovacuum_logging

Le `rds.force_autovacuum_logging` paramètre (disponible uniquement dans Amazon RDS for PostgreSQL) contrôle si les actions d'autovacuum sont enregistrées dans le journal du serveur. Ses valeurs sont`disabled`,`debug5`,`debug4`,`debug3`,`debug2`,`debug1`,`info`,`notice`,`warning`,`error`,`log`,`fatal`,`pan`
La valeur par défaut est `warning`.

Lorsque vous l'activez`rds.force_autovacuum_logging`, toutes les actions du processus d'aspiration automatique, telles que le début du processus, sa fin et le nombre de rangées aspirées, sont enregistrées. Cela est utile pour le débogage ou le dépannage des problèmes de performance de l'aspirateur automatique.

AWS CLI syntaxe

La commande suivante change `rds.force_autovacuum_logging` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify rds.force_autovacuum_logging on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=rds.force_autovacuum_logging,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify rds.force_autovacuum_logging on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=rds.force_autovacuum_logging,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `warning`

Exemple

Vous pouvez utiliser `rds.force_autovacuum_logging` ce paramètre pour analyser les performances de l'autovacuum sur une table dont le taux d'écriture est très élevé. Par exemple, si votre table reçoit un grand nombre d'opérations d'écriture et de suppression par seconde et que les performances sont lentes, vous pouvez activer le paramètre pour enregistrer les heures de début et de fin de chaque cycle d'aspiration automatique et pour déterminer le nombre de lignes aspirées. Cela peut fournir des informations précieuses sur la fréquence de fonctionnement de l'aspirateur automatique, sa durée de fonctionnement et le nombre de rangées qu'il aspire. Vous pouvez ensuite utiliser ces informations pour affiner les paramètres d'aspiration automatique tels que `autovacuum_vacuum_scale_factorautovacuum_vacuum_threshold`, et `autovacuum_naptime` pour optimiser les performances.

rds.force_admin_logging_level

Le `rds.force_admin_logging_level` paramètre (disponible uniquement dans Amazon RDS for PostgreSQL) contrôle le niveau de détail des journaux produits par des opérations administratives telles que la mise sous vide, l'analyse et la réindexation. Il accepte les valeurs`debug5`,`debug4`,`debug3`,`debug2`,`debug1`,`log`,,`info`,`notice`,`warning`,`error`, `logfatal`, et `off` (par défaut). Le réglage optimal dépend de votre cas d'utilisation. Par exemple, si vous résolvez un problème, vous souhaiterez peut-être définir le paramètre sur un niveau de débogage. Sinon, vous pouvez utiliser le `warning` paramètre `loginfo`, ou.

Si vous définissez cette `rds.force_admin_logging_level` option`debug1`, vous pouvez enregistrer des informations détaillées relatives à une opération de réindexation, telles que les heures de début et de fin, le nombre de lignes traitées et les erreurs ou avertissements éventuels survenant au cours du processus. Cela peut fournir des informations précieuses sur le fonctionnement du processus de réindexation et vous aider à résoudre les problèmes éventuels.

AWS CLI syntaxe

La commande suivante change `rds.force_admin_logging_level` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify rds.force_admin_logging_level on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
```

```
  --parameters
 "ParameterName=rds.force_admin_logging_level,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify rds.force_admin_logging_level on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=rds.force_admin_logging_level,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `off`

Exemple

Vous pouvez l'utiliser `rds.force_admin_logging_level` pour surveiller et analyser les performances des opérations administratives sur plusieurs tables d'une base de données volumineuse. Supposons, par exemple, que vous disposiez d'une base de données volumineuse comportant de nombreuses tables et que vous souhaitiez optimiser les performances de ces tables en effectuant régulièrement des opérations de nettoyage et d'analyse sur celles-ci. En définissant le `rds.force_admin_logging_level` paramètre sur `info` ou`log`, vous pouvez enregistrer les heures de début et de fin de chaque opération ainsi que les tables concernées. Vous pouvez utiliser ces informations pour suivre les performances des opérations administratives sur différentes tables et identifier les tables susceptibles de nécessiter une maintenance plus fréquente ou plus agressive.

Certains niveaux de journalisation génèrent un grand nombre de fichiers journaux et de messages susceptibles d'occuper rapidement de l'espace disque, en particulier si votre base de données est occupée. Nous vous recommandons d'utiliser ce paramètre avec précaution et de le désactiver une fois la période de surveillance terminée.

log_duration

Le `log_duration` paramètre contrôle si la durée de chaque requête (c'est-à-dire le temps nécessaire à son exécution) est enregistrée avec la requête. Lorsque vous définissez ce paramètre suron, le temps nécessaire pour exécuter chaque requête est inclus dans la sortie du journal avec le texte de la requête. Le temps est mesuré en millisecondes.

Le principal cas d'utilisation du `log_duration` paramètre est de faciliter le réglage des performances et le dépannage. En enregistrant la durée de chaque requête, vous pouvez identifier

les requêtes dont l'exécution est la plus longue, puis concentrer vos efforts sur l'optimisation de ces requêtes. Cela peut vous aider à identifier et à résoudre les problèmes de performance et à améliorer les performances globales de votre base de données.

AWS CLI syntaxe

La commande suivante change `log_duration` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_duration on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_duration,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_duration on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_duration,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `off`

Exemple

Vous pouvez utiliser ce paramètre si vous pensez qu'une requête ou un ensemble de requêtes spécifique est à l'origine de problèmes de performances. En activant le `log_duration` paramètre et en examinant la sortie du journal, vous pouvez identifier les requêtes dont l'exécution est la plus longue, puis prendre les mesures appropriées, telles que l'optimisation des index, l'ajout de nouveaux index ou la réécriture de la requête.

L'activation `log_duration` peut augmenter le volume de sortie du journal. Nous vous recommandons de ne l'utiliser qu'en cas de besoin et de le désactiver pendant les opérations standard pour éviter de remplir le stockage ou de rendre les journaux difficiles à lire.

log_min_duration_statement

Le `log_min_duration_statement` paramètre contrôle la durée minimale, en millisecondes, pendant laquelle une instruction SQL est exécutée avant d'être enregistrée.

Ce paramètre vous aide à identifier les requêtes de longue durée susceptibles d'entraîner des problèmes de performances. Vous pouvez le définir sur une valeur seuil (un temps d'exécution considéré comme trop long pour une charge de travail spécifique) afin de capturer les requêtes qui dépassent ce seuil et d'identifier les problèmes de performances potentiels. Pour un exemple de cas d'utilisation, consultez la section Utilisation de paramètres de journalisation pour capturer des variables de liaison plus loin dans ce guide.

AWS CLI syntaxe

La commande suivante change `log_min_duration_statement` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_min_duration_statement on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_min_duration_statement,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_ min_duration_statement on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_min_duration_statement,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : 1 (désactivé, qui est le moteur PostgreSQL par défaut)

Exemple

La commande suivante enregistre toute instruction dont l'exécution prend plus de 100 millisecondes :

```
aws rds modify-db-parameter-group \
   --db-parameter-group-name <parameter_group_name> \
```

```
   --parameters
 "ParameterName=log_min_duration_statement,ParameterValue=100,ApplyMethod=immediate"
```

log_error_verbosity

Le `log_error_verbosity` paramètre contrôle le niveau de détail inclus dans la sortie du journal pour les erreurs et les messages enregistrés au niveau d'erreur ou supérieur. Ce paramètre peut prendre l'une des trois valeurs suivantes : `terse`, `default` ou `verbose`.

- `terse` inclut uniquement le texte du message, le niveau d'erreur, ainsi que le numéro de fichier et de ligne où l'erreur s'est produite.
- `default` inclut le texte du message, le niveau d'erreur, le numéro de fichier et de ligne, ainsi que le contexte de l'erreur.
- `verbose` inclut le texte du message, le niveau d'erreur, le numéro de fichier et de ligne, le contexte de l'erreur et le message d'erreur complet.

Définissez le paramètre sur pour `verbose` obtenir les informations les plus détaillées sur le dépannage et le débogage dans un environnement hors production. Dans un environnement de production, vous pouvez le configurer de `terse` manière à `default` ce qu'il ne fournisse que les informations essentielles et qu'il ne remplisse pas le stockage des journaux avec trop de détails.

AWS CLI syntaxe

La commande suivante change `log_error_verbosity` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_error_verbosity on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_error_verbosity,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_error_verbosity on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_error_verbosity,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissezApplyMethod=immediate)

Valeur par défaut : default

log_statement

Le log_statement paramètre contrôle les instructions SQL enregistrées dans le journal du serveur. Le paramètre peut prendre l'une des valeurs suivantes :

- none(par défaut) n'enregistre aucune instruction
- ddlenregistre uniquement les instructions du langage de définition des données (DDL) telles que et CREATE TABLE ALTER TABLE
- modenregistre uniquement les déclarations modifiant les données telles que INSERTUPDATE, et DELETE
- allenregistre toutes les instructions SQL

Vous pouvez utiliser le log_statement paramètre pour contrôler la quantité d'informations écrites dans le journal en documentant uniquement les types spécifiques d'instructions correspondant à votre cas d'utilisation.

AWS CLI syntaxe

La commande suivante change log_statement pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_statement on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_statement,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_statement on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_statement,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissezApplyMethod=immediate)

Valeur par défaut : none

Exemple

Dans un environnement de production, vous souhaiterez peut-être configurer log_statement ddl pour enregistrer uniquement les instructions DDL et suivre les modifications apportées au schéma de base de données. Dans un environnement de développement, vous souhaiterez peut-être définir le paramètre de manière all à consigner toutes les instructions afin de faciliter le débogage et le dépannage. Pour un autre exemple d'utilisation, consultez la section Utilisation de paramètres de journalisation pour capturer des variables de liaison plus loin dans ce guide.

L'activation log_statement peut augmenter le volume de sortie des journaux. Par conséquent, utilisez-le uniquement lorsque cela est nécessaire et désactivez-le pour éviter de surcharger l'espace de stockage ou de rendre les journaux difficiles à lire.

Nous vous recommandons de surveiller votre système et d'ajuster la valeur de ce paramètre afin de trouver le juste équilibre entre la quantité d'informations enregistrées et le stockage et les performances du système.

log_statement_stats

Le log_statement_stats paramètre contrôle si les statistiques associées à l'exécution d'une instruction SQL sont enregistrées avec l'instruction. Lorsque vous activez ce paramètre, les statistiques telles que le nombre de lignes affectées, le nombre de blocs de disque lus et écrits et le temps nécessaire à l'exécution de l'instruction sont incluses dans la sortie du journal.

Vous pouvez utiliser le log_statement_stats paramètre pour recueillir des informations supplémentaires sur les performances des instructions individuelles et sur la charge de travail globale. En enregistrant les statistiques des instructions, vous pouvez identifier les modèles de performances des requêtes et d'utilisation des ressources, et utiliser ces informations pour optimiser votre base de données et améliorer les performances globales.

AWS CLI syntaxe

La commande suivante change log_statement_stats pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_statement_stats on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_statement_stats,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_statement_stas on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_statement_stats,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `off` (valeur par défaut du moteur PostgreSQL) ; utilisez 0 ou 1 (booléen) pour définir des groupes de paramètres

Exemple

Vous pouvez l'utiliser `log_statement_stats` pour analyser le comportement d'une requête spécifique, voir comment elle utilise des ressources telles que le processeur, la mémoire et les E/S de disque, et déterminer si la requête peut être optimisée. Vous pouvez également utiliser ce paramètre pour voir si une table spécifique est lue fréquemment (ce qui peut indiquer la nécessité de créer un index sur une colonne spécifique) ou si une table est analysée trop souvent.

L'activation `log_statement_stats` peut augmenter le volume de sortie des journaux. Par conséquent, utilisez-le uniquement lorsque cela est nécessaire et désactivez-le pour éviter de surcharger l'espace de stockage ou de rendre les journaux difficiles à lire.

log_min_error_statement

Le `log_min_error_statement` paramètre contrôle les instructions SQL à l'origine d'une erreur qui seront enregistrées. Ses valeurs sont`debug5`,`debug4`,`debug3`,`debug2`,`debug1`,`info`,`notice`,`warning`,`error`,`log`,`fatal`, et`panic`. Ces paramètres contrôlent la quantité d'informations écrites dans le journal afin que vous puissiez filtrer les messages moins graves. Vous pouvez attribuer à ce paramètre un niveau de gravité plus élevé afin de réduire le volume de données générées par le journal et de retrouver plus facilement les messages importants.

AWS CLI syntaxe

La commande suivante change `log_min_error_statement` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_min_error_statement on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_min_error_statement,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_min_error_statement on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_min_error_statement,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `error` (valeur par défaut du moteur PostgreSQL)

Exemple

Vous pouvez envisager de l'utiliser `log_min_error_statement` lorsque vous résolvez un problème spécifique et que vous souhaitez voir les messages d'erreur provenant d'instructions SQL à l'origine d'erreurs.

log_min_messages

Le `log_min_messages` paramètre contrôle le niveau de gravité inscrit dans le journal. Vous pouvez définir le paramètre sur`debug5`,`debug4`, `debug3``debug2`,`debug1`,`info`,`notice`,`warning`,`error`,`log`,`fatal`, ou`panic`. Ces paramètres contrôlent la quantité d'informations écrites dans le journal afin que vous puissiez filtrer les messages moins graves. Vous pouvez attribuer à ce paramètre un niveau de gravité plus élevé afin de réduire le volume de données générées par le journal et de retrouver plus facilement les messages importants.

AWS CLI syntaxe

La commande suivante change `log_min_messages` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_min_messages on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_min_messages,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_min_messages on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_min_messages,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `notice`

Exemple

Si vous résolvez un problème spécifique et que vous souhaitez voir tous les messages d'erreur, vous pouvez définir ce paramètre sur `error` pour enregistrer uniquement les erreurs et les problèmes plus graves. Si vous souhaitez surveiller les performances du système, vous pouvez définir ce paramètre `info` pour afficher des informations plus détaillées, telles que la durée et les statistiques de chaque instruction.

Le réglage `log_min_messages` d'un niveau de gravité plus élevé réduit le volume des journaux. Nous vous recommandons de régler ce paramètre en fonction de votre cas d'utilisation spécifique, de la taille du journal que vous souhaitez consulter et de l'espace disque dont vous disposez.

log_temp_files

Le `log_temp_files` paramètre contrôle la journalisation des noms et des tailles de fichiers temporaires. Elle s'applique aux fichiers temporaires créés à des fins telles que le tri, le hachage et les résultats de requêtes temporaires. Lorsque ce paramètre est activé, une entrée de journal est générée pour chaque fichier temporaire lors de la suppression, y compris sa taille de fichier, en

octets. Vous pouvez définir ce paramètre sur 0 (zéro) pour une journalisation complète de toutes les informations relatives aux fichiers temporaires, ou sur une valeur positive pour les fichiers journaux dont la taille dépasse cette taille (en kilo-octets, si aucune unité n'est spécifiée). Cela peut être utile pour identifier et résoudre les problèmes de performance ou autres problèmes liés au stockage temporaire. Par défaut, la journalisation des fichiers temporaires est désactivée.

AWS CLI syntaxe

La commande suivante change `log_temp_files` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_temp_files on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_temp_files,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_temp_files on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_temp_files,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : -1 (valeur par défaut du moteur PostgreSQL)

Exemple

Vous pouvez activer ce paramètre si vous pensez que le système utilise trop d'espace de stockage temporaire ou que les fichiers temporaires ne sont pas correctement supprimés. Lorsque vous examinez la sortie du journal, vous pouvez voir les requêtes ou les opérations qui génèrent des fichiers temporaires et la manière dont ces fichiers sont utilisés.

Certaines requêtes ou opérations créent un grand nombre de fichiers temporaires. Leur activation `log_temp_files` peut donc avoir un impact sur les performances globales de votre système.

log_connections

Le `log_connections` paramètre contrôle si les connexions à la base de données sont enregistrées. Lorsque vous définissez ce paramètre suron, le journal contient des informations sur chaque connexion réussie à la base de données, telles que l'adresse IP du client, le nom d'utilisateur, le nom de la base de données, ainsi que la date et l'heure de la connexion.

Vous pouvez utiliser le `log_connections` paramètre pour surveiller et résoudre les problèmes de connexion à la base de données. Vous pouvez voir les utilisateurs, les applications, les terminaux et les robots qui se connectent à la base de données, d'où ils se connectent et à quelle fréquence. Ces informations peuvent être utiles pour identifier et résoudre les problèmes liés à la connexion ou pour suivre les habitudes d'utilisation.

AWS CLI syntaxe

La commande suivante change `log_connections` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_connections on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_connections,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_connections on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_connections,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `off` (valeur par défaut du moteur PostgreSQL)

Exemple

Vous pouvez utiliser ce paramètre si vous pensez qu'un trop grand nombre de connexions à la base de données, ou qu'un utilisateur ou une adresse IP spécifique se connectant trop fréquemment

affectent les performances. En activant le `log_connections` paramètre et en examinant la sortie du journal, vous pouvez voir le nombre et les détails de toutes les connexions.

Avant d'activer ce paramètre, vérifiez les politiques de votre organisation et considérez les implications en termes de sécurité de la journalisation des adresses IP et des noms d'utilisateur.

log_disconnections

Le `log_disconnections` paramètre contrôle l'enregistrement des déconnexions de la base de données. Lorsque vous définissez ce paramètre suron, il enregistre les informations relatives à la fin de chaque session, telles que l'adresse IP du client, le nom d'utilisateur, le nom de la base de données, ainsi que la date et l'heure de la déconnexion.

Vous pouvez utiliser le `log_disconnections` paramètre pour surveiller et résoudre les problèmes d'interruption de session de base de données. Vous pouvez voir les utilisateurs, les applications, les terminaux et les robots qui se déconnectent de la base de données, quand et pourquoi. Par exemple, vous pouvez examiner les interruptions inattendues, telles qu'un crash ou des déconnexions initiées par l'administrateur. Ces informations peuvent être utiles pour identifier et résoudre les problèmes liés aux déconnexions ou pour suivre les habitudes d'utilisation.

AWS CLI syntaxe

La commande suivante change `log_disconnections` pour un groupe de paramètres de base de données spécifique. Cette modification s'applique à toutes les instances ou à tous les clusters qui utilisent le groupe de paramètres.

```
# Modify log_disconnections on a DB parameter group
aws rds modify-db-parameter-group \
  --db-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_disconnections,ParameterValue=<new_value>,ApplyMethod=immediate"

# Modify log_disconnections on a DB cluster parameter group
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name <parameter_group_name> \
  --parameters
 "ParameterName=log_disconnections,ParameterValue=<new_value>,ApplyMethod=immediate"
```

Type : Dynamique (les modifications sont appliquées immédiatement si vous définissez`ApplyMethod=immediate`)

Valeur par défaut : `off` (valeur par défaut du moteur PostgreSQL)

Exemple

Vous pouvez l'utiliser `log_disconnections` si vous pensez qu'un trop grand nombre d'utilisateurs se déconnectent de la base de données ou qu'un utilisateur ou une adresse IP spécifique se déconnectent trop fréquemment. En activant le `log_disconnections` paramètre et en examinant la sortie du journal, vous pouvez voir le nombre et les détails de toutes les déconnexions, y compris qui, quand et si des erreurs ont été signalées avant la déconnexion.

Avant d'activer ce paramètre, vérifiez les politiques de votre organisation et considérez les implications en termes de sécurité de la journalisation des adresses IP et des noms d'utilisateur.

Utilisation de paramètres de journalisation pour capturer des variables de liaison

Un cas d'utilisation typique pour capturer des variables de liaison dans PostgreSQL consiste à déboguer et à optimiser les performances des requêtes SQL. Une variable de liaison vous permet de transmettre des données à une requête lorsque vous l'exécutez. En capturant les variables de liaison, vous pouvez voir les données d'entrée qui ont été transmises à une requête, ce qui peut vous aider à identifier tout problème lié aux données ou aux performances de la requête. La capture des variables de liaison peut également vous aider à auditer les données d'entrée et à détecter les risques de sécurité potentiels ou les activités malveillantes.

Il existe plusieurs méthodes pour capturer des variables de liaison pour PostgreSQL. L'une des méthodes consiste à activer les `debug_print_rewritten` paramètres `debug_print_parse` et. Cela oblige PostgreSQL à envoyer les versions analysées et réécrites des instructions SQL, ainsi que les variables liées, au journal du serveur.

- `debug_print_parse`: Lorsque vous activez ce paramètre, l'arbre d'analyse des requêtes entrantes est imprimé dans le journal du serveur. Cela peut être utile pour comprendre la structure d'une requête et les valeurs de tous les paramètres liés.
- `debug_print_rewritten`: Lorsque vous activez ce paramètre, les formulaires réécrits des requêtes entrantes sont imprimés dans le journal du serveur. Cela peut être utile pour comprendre comment le planificateur de requêtes interprète une requête et les valeurs de tous les paramètres liés.

Vous pouvez utiliser deux paramètres supplémentaires dans Amazon RDS et Aurora pour capturer les variables de liaison dans vos bases de données PostgreSQL :

- `log_min_duration_statement`: Ce paramètre définit la durée minimale d'une instruction avant qu'elle ne soit enregistrée, en millisecondes. Lorsqu'une instruction prend plus de temps que la durée spécifiée, ses valeurs de liaison sont incluses dans la sortie du journal.
- `log_statement`: Ce paramètre contrôle les instructions SQL qui sont enregistrées. Définissez ce paramètre sur `all` ou liez pour inclure les valeurs liées dans le journal. L'augmentation du niveau de journalisation affecte les performances. Nous vous recommandons donc d'annuler les modifications après le dépannage.

Vous pouvez également utiliser l'`pg_stat_statements`extension, qui fournit des statistiques de performance pour toutes les instructions SQL exécutées par un serveur, y compris le texte de la requête et les valeurs liées. Cette extension vous permet d'utiliser pgAdmin ou des outils similaires pour surveiller et analyser les performances des requêtes.

Une autre option consiste à utiliser la `pg_bind_parameter_status()` fonction pour obtenir les valeurs des paramètres liés à partir d'une instruction préparée ou à utiliser la `pg_get_parameter_status (paramname)` fonction pour récupérer le statut ou la valeur d'un paramètre d'exécution spécifique.

En outre, vous pouvez utiliser des outils tiers tels que pgBadger pour analyser les journaux PostgreSQL et extraire les variables de liaison ainsi que d'autres informations pour une analyse plus approfondie.

Réglage des paramètres de réplication

Dans PostgreSQL, vous pouvez répliquer les modifications de données d'une base de données PostgreSQL à une autre en utilisant la réplication logique plutôt que la réplication physique basée sur des fichiers. La réplication logique utilise le journal d'écriture anticipée (WAL) pour capturer les modifications et prend en charge la réplication de tables sélectionnées ou de bases de données complètes.

Amazon RDS for PostgreSQL et Aurora PostgreSQL compatibles prennent tous deux en charge la réplication logique, ce qui vous permet de configurer une architecture de base de données hautement disponible et évolutive capable de gérer le trafic de lecture et d'écriture provenant de plusieurs sources. Ces services utilisent pglogical, une extension PostgreSQL open source, pour implémenter la réplication logique.

Le réglage de la réplication logique dans Aurora et Amazon RDS est important pour obtenir des performances, une évolutivité et une disponibilité optimales. Vous pouvez ajuster les paramètres de l'extension pglogical pour gérer les performances de la réplication logique. Par exemple, vous pouvez :

- Améliorez les performances de réplication en augmentant le nombre de processus de travail ou en ajustant leur allocation de mémoire.
- Réduisez le risque de retard de réplication en ajustant la fréquence de synchronisation entre les bases de données source et répliquée.
- Optimisez l'utilisation des ressources en ajustant la mémoire et l'allocation du processeur des processus de travail.
- Assurez-vous que le processus de réplication n'a pas d'impact indu sur les performances de la base de données source.

Vous pouvez utiliser les paramètres suivants dans Aurora et Amazon RDS pour contrôler et configurer la réplication logique :

- `max_replication_slots`définit le nombre maximal de slots de réplication pouvant être créés sur le serveur. Un slot de réplication est une réservation permanente nommée pour une connexion de réplication afin d'envoyer des données WAL à une réplique.

- `max_wal_senders`définit le nombre maximum de processus d'expéditeur WAL connectés simultanément. Les processus d'expéditeur du WAL sont utilisés pour diffuser le WAL du serveur principal vers la réplique.
- `wal_sender_timeout`définit le temps maximum, en millisecondes, pendant lequel un expéditeur WAL attend une réponse de la réplique avant d'abandonner et de se reconnecter.
- `wal_receiver_timeout`définit le temps maximum, en millisecondes, pendant lequel une réplique attend les données WAL de la base de données principale avant d'expirer.
- `log_replication_commands`, lorsqu'il est défini suron, exécute les instructions SQL relatives à la réplication.

Lorsque vous activez le `rds.logical_replication` paramètre (en le définissant sur 1), le `wal_level` paramètre est défini sur`logical`, ce qui signifie que toutes les modifications apportées à la base de données sont écrites dans le WAL dans un format qui peut être lu et appliqué à une réplique. Ce paramètre est nécessaire pour activer la réplication logique. Ce paramètre permet également la réplication des `SELECT` instructions.

Le réglage `wal_level` sur `logical` peut augmenter la quantité de données écrites sur le WAL, et donc sur le disque, ce qui peut affecter les performances du système. Nous vous recommandons de prendre en compte l'espace disque disponible et les performances du système lorsque vous activez la réplication logique.

Exemple

Vous souhaitez répliquer les données de votre base de données principale vers une base de données secondaire à des fins de sauvegarde et de reprise après sinistre. Cependant, la base de données secondaire comporte un volume élevé d'opérations de lecture. Vous devez donc vous assurer que le processus de réplication est aussi rapide et efficace que possible sans compromettre l'intégrité des données.

Les valeurs par défaut pour la réplication logique dans Amazon RDS et Aurora privilégient la cohérence par rapport aux performances. Elles peuvent donc ne pas être optimales pour ce cas d'utilisation. Pour optimiser vos paramètres de réplication logique en termes de rapidité et d'efficacité, vous pouvez personnaliser les paramètres comme suit :

- Passez `max_replication_slots` de 10 (par défaut pour Amazon RDS) ou 20 (par défaut pour Aurora) à 30 pour répondre aux besoins potentiels de croissance et de réplication futurs.

- Passez `max_wal_senders` de 10 (par défaut) à 20 pour garantir qu'il existe suffisamment de processus d'expéditeur WAL pour répondre à la demande de réplication.
- Passez `wal_sender_timeout` de 30 secondes (par défaut) à 15 secondes pour garantir que les processus d'expéditeur WAL inactifs sont interrompus plus rapidement, ce qui libère des ressources pour une réplication active.
- Passez `wal_receiver_timeout` de 30 secondes (par défaut) à 15 secondes pour que les processus de réception WAL inactifs soient interrompus plus rapidement, ce qui libère des ressources pour une réplication active.
- Passez `max_logical_replication_workers` de 4 (par défaut) à 8 pour garantir qu'il existe suffisamment de processus de réplication logiques pour répondre à la demande de réplication.

Ces optimisations permettent une réplication des données plus rapide et plus efficace tout en préservant l'intégrité et la sécurité des données.

Par exemple, si un sinistre devait se produire et que la base de données principale devenait indisponible, la base de données secondaire disposerait déjà des dernières données disponibles grâce au processus de réplication optimisé. Cela permettrait à vos activités commerciales de continuer à fournir des services essentiels sans interruption.

Bonnes pratiques

L'optimisation de la réplication logique avec des charges de travail importantes peut s'avérer une tâche complexe qui dépend de nombreux facteurs, notamment de la taille du jeu de données, du nombre de tables répliquées, du nombre de répliques et des ressources disponibles. Voici quelques conseils généraux pour optimiser la réplication logique avec des charges de travail importantes :

- Surveillez le décalage de réplication. Le délai de réplication est le décalage horaire entre le serveur principal et les serveurs de secours. La surveillance du décalage de réplication peut vous aider à identifier les goulets d'étranglement potentiels et à prendre des mesures pour améliorer les performances de réplication. Vous pouvez utiliser cette `pg_current_wal_lsn()` fonction pour vérifier le délai de réplication actuel.
- Réglez les paramètres WAL. L'`pg_logical`extension utilise WAL pour transmettre les modifications du serveur principal au serveur de secours. Si les paramètres WAL ne sont pas correctement réglés, la réplication peut devenir lente et peu fiable. Assurez-vous de définir les `max_replication_slots` paramètres `max_wal_senders` et sur des valeurs appropriées, en fonction de vos charges de travail.

- Adoptez une stratégie d'indexation. Le fait de disposer d'index appropriés sur le serveur principal peut contribuer à améliorer les performances de la réplication logique, à réduire les E/S sur le serveur principal et à réduire la charge sur le système.
- Utilisez la réplication parallèle. L'utilisation de la réplication parallèle peut contribuer à augmenter la vitesse de réplication en permettant à plusieurs processus de travail parallèle de répliquer les données. Cette fonctionnalité est disponible sur PostgreSQL 12 et versions ultérieures.

Étapes suivantes

Après avoir optimisé les paramètres de mémoire, de réplication, d'autovacuum et de journalisation pour votre base de données compatible Amazon RDS for PostgreSQL ou Aurora PostgreSQL, envisagez les étapes suivantes pour améliorer encore les performances de votre base de données :

- Surveillez votre base de données. Suivez les performances de votre base de données au fil du temps à l'aide d'outils de surveillance intégrés ou de solutions tierces. Surveillez les indicateurs de performance clés tels que l'utilisation du processeur, les E/S du disque, l'utilisation de la mémoire et les temps d'exécution des requêtes afin d'identifier les goulets d'étranglement potentiels et les domaines à améliorer.
- Réglez les paramètres en continu. À mesure que votre charge de travail évolue, continuez à surveiller et à ajuster les paramètres de votre base de données pour garantir des performances optimales. Consultez régulièrement les journaux du système, les messages d'erreur et les indicateurs de performance pour identifier de nouvelles opportunités de réglage.
- Implémentez la mise en cache. Utilisez la mise en cache pour réduire le nombre de requêtes qui atteignent la base de données. Vous pouvez implémenter la mise en cache au niveau de l'application à l'aide d'outils tels que Memcached ou Redis, ou vous pouvez utiliser Amazon ElastiCache pour fournir un cache en mémoire pour votre base de données.
- Optimisez vos requêtes. Des requêtes mal conçues peuvent avoir un impact significatif sur les performances de la base de données. Utilisez `EXPLAIN` d'autres outils de réglage des requêtes pour identifier les requêtes lentes, les optimiser et éliminer les requêtes inutiles.

En suivant ces directives, vous pouvez optimiser les performances de votre base de données Aurora ou Amazon RDS for PostgreSQL et vous assurer qu'elle répond aux besoins de votre application en améliorant les performances de la base de données, en augmentant la fiabilité, en réduisant les temps d'arrêt, en renforçant la sécurité et en réduisant les coûts. En optimisant les paramètres de configuration en fonction de votre charge de travail, vous pouvez vous assurer que votre base de données fonctionne efficacement et utilise les ressources de manière efficace, ce qui se traduit par de meilleures performances et une application plus réactive. En outre, des paramètres correctement configurés peuvent réduire le risque d'erreurs et de vulnérabilités, ce qui se traduit par une fiabilité accrue et une meilleure sécurité. Cela peut se traduire par des économies en termes de réduction de la maintenance et des temps d'arrêt, ainsi que par une amélioration de l'expérience et de la satisfaction générales des utilisateurs.

Ressources

- Paramètres Amazon Aurora PostgreSQL, partie 1 : gestion AWS de la mémoire et du plan de requêtes (article de blog)
- Paramètres Amazon Aurora PostgreSQL, partie 2 : réplication, sécurité et journalisation AWS (article de blog)
- Paramètres Amazon Aurora PostgreSQL, partie 3 : paramètres de l'optimiseur (article de blog)AWS
- Paramètres Amazon Aurora PostgreSQL, partie 4 : options de compatibilité ANSI (article de blog)AWS
- Utilisation d'Amazon Aurora AWS PostgreSQL (documentation)
- Utilisation d'Amazon RDS pour PostgreSQL (documentation)AWS
- Surveillance de la charge de base de données avec Performance Insights sur Amazon RDS (AWS documentation)
- Utilisation CloudWatch des métriques Amazon (AWS documentation)
- pg_stats_statements (documentation PostgreSQL)

Historique du document

Le tableau suivant décrit les modifications importantes apportées à ce guide. Pour être averti des mises à jour à venir, abonnez-vous à un fil RSS.

Modification	Description	Date
Informations mises à jour sur les paramètres de mémoire et d'autovacuum	Mise à jour de la description du paramètre random_page_cost ; ajout d'unités manquantes aux valeurs par défaut pour les paramètres de mémoire et d'autovacuum ; mise à jour de la AWS CLI syntaxe du paramètre max_connections.	27 février 2024
Informations mises à jour sur `autovacuum`	Correction du réglage par défaut de l'aspirateur automatique (activé).	27 décembre 2023
Informations mises à jour sur `max_connections`	Mise à jour de la section max_connections avec de nouvelles instructions sur le réglage de ce paramètre.	15 novembre 2023
Publication initiale	—	31 octobre 2023

AWS Glossaire des directives prescriptives

Les termes suivants sont couramment utilisés dans les stratégies, les guides et les modèles fournis par les directives AWS prescriptives. Pour suggérer des entrées, veuillez utiliser le lien Faire un commentaire à la fin du glossaire.

Nombres

7 R

Sept politiques de migration courantes pour transférer des applications vers le cloud. Ces politiques s'appuient sur les 5 R identifiés par Gartner en 2011 et sont les suivantes :

- Refactorisation/réarchitecture : transférez une application et modifiez son architecture en tirant pleinement parti des fonctionnalités natives cloud pour améliorer l'agilité, les performances et la capacité de mise à l'échelle. Cela implique généralement le transfert du système d'exploitation et de la base de données. Exemple : migrez votre base de données Oracle sur site vers l'édition compatible avec Amazon Aurora PostgreSQL.
- Replateformer (déplacer et remodeler) : transférez une application vers le cloud et introduisez un certain niveau d'optimisation pour tirer parti des fonctionnalités du cloud. Exemple : migrez votre base de données Oracle sur site vers Amazon Relational Database Service (Amazon RDS) pour Oracle dans le. AWS Cloud
- Racheter (rachat) : optez pour un autre produit, généralement en passant d'une licence traditionnelle à un modèle SaaS. Exemple : migrez votre système de gestion de la relation client (CRM) vers Salesforce.com.
- Réhéberger (lift and shift) : transférez une application vers le cloud sans apporter de modifications pour tirer parti des fonctionnalités du cloud. Exemple : migrez votre base de données Oracle sur site vers Oracle sur une instance EC2 dans le. AWS Cloud
- Relocaliser (lift and shift au niveau de l'hyperviseur) : transférez l'infrastructure vers le cloud sans acheter de nouveau matériel, réécrire des applications ou modifier vos opérations existantes. Vous migrez des serveurs d'une plateforme sur site vers un service cloud pour la même plateforme. Exemple : migrer une Microsoft Hyper-V application vers AWS.
- Retenir : conservez les applications dans votre environnement source. Il peut s'agir d'applications nécessitant une refactorisation majeure, que vous souhaitez retarder, et d'applications existantes que vous souhaitez retenir, car rien ne justifie leur migration sur le plan commercial.

- Retirer : mettez hors service ou supprimez les applications dont vous n'avez plus besoin dans votre environnement source.

A

ABAC

Voir contrôle d'accès basé sur les attributs.

services abstraits

Consultez la section Services gérés.

ACIDE

Voir atomicité, consistance, isolation, durabilité.

migration active-active

Méthode de migration de base de données dans laquelle la synchronisation des bases de données source et cible est maintenue (à l'aide d'un outil de réplication bidirectionnelle ou d'opérations d'écriture double), tandis que les deux bases de données gèrent les transactions provenant de la connexion d'applications pendant la migration. Cette méthode prend en charge la migration par petits lots contrôlés au lieu d'exiger un basculement ponctuel. Elle est plus flexible mais demande plus de travail qu'une migration active-passive.

migration active-passive

Méthode de migration de base de données dans laquelle la synchronisation des bases de données source et cible est maintenue, mais seule la base de données source gère les transactions provenant de la connexion d'applications pendant que les données sont répliquées vers la base de données cible. La base de données cible n'accepte aucune transaction pendant la migration.

fonction d'agrégation

Fonction SQL qui agit sur un groupe de lignes et calcule une valeur de retour unique pour le groupe. Des exemples de fonctions d'agrégation incluent `SUM` et`MAX`.

AI

Voir intelligence artificielle.

AIOps

Voir les opérations d'intelligence artificielle.

anonymisation

Processus de suppression définitive d'informations personnelles dans un ensemble de données. L'anonymisation peut contribuer à protéger la vie privée. Les données anonymisées ne sont plus considérées comme des données personnelles.

anti-motif

Solution fréquemment utilisée pour un problème récurrent lorsque la solution est contre-productive, inefficace ou moins efficace qu'une solution alternative.

contrôle des applications

Une approche de sécurité qui permet d'utiliser uniquement des applications approuvées afin de protéger un système contre les logiciels malveillants.

portefeuille d'applications

Ensemble d'informations détaillées sur chaque application utilisée par une organisation, y compris le coût de génération et de maintenance de l'application, ainsi que sa valeur métier. Ces informations sont essentielles pour le processus de découverte et d'analyse du portefeuille et permettent d'identifier et de prioriser les applications à migrer, à moderniser et à optimiser.

intelligence artificielle (IA)

Domaine de l'informatique consacré à l'utilisation des technologies de calcul pour exécuter des fonctions cognitives généralement associées aux humains, telles que l'apprentissage, la résolution de problèmes et la reconnaissance de modèles. Pour plus d'informations, veuillez consulter Qu'est-ce que l'intelligence artificielle ?

opérations d'intelligence artificielle (AIOps)

Processus consistant à utiliser des techniques de machine learning pour résoudre les problèmes opérationnels, réduire les incidents opérationnels et les interventions humaines, mais aussi améliorer la qualité du service. Pour plus d'informations sur la façon dont les AIOps sont utilisées dans la stratégie de migration AWS , veuillez consulter le guide d'intégration des opérations.

chiffrement asymétrique

Algorithme de chiffrement qui utilise une paire de clés, une clé publique pour le chiffrement et une clé privée pour le déchiffrement. Vous pouvez partager la clé publique, car elle n'est pas utilisée pour le déchiffrement, mais l'accès à la clé privée doit être très restreint.

atomicité, cohérence, isolement, durabilité (ACID)

Ensemble de propriétés logicielles garantissant la validité des données et la fiabilité opérationnelle d'une base de données, même en cas d'erreur, de panne de courant ou d'autres problèmes.

contrôle d'accès par attributs (ABAC)

Pratique qui consiste à créer des autorisations détaillées en fonction des attributs de l'utilisateur, tels que le service, le poste et le nom de l'équipe. Pour plus d'informations, consultez ABAC pour AWS dans la documentation AWS Identity and Access Management (IAM).

source de données faisant autorité

Emplacement où vous stockez la version principale des données, considérée comme la source d'information la plus fiable. Vous pouvez copier les données de la source de données officielle vers d'autres emplacements à des fins de traitement ou de modification des données, par exemple en les anonymisant, en les expurgant ou en les pseudonymisant.

Zone de disponibilité

Un emplacement distinct au sein d'un Région AWS réseau isolé des défaillances dans d'autres zones de disponibilité et fournissant une connectivité réseau peu coûteuse et à faible latence aux autres zones de disponibilité de la même région.

AWS Cadre d'adoption du cloud (AWS CAF)

Un cadre de directives et de meilleures pratiques visant AWS à aider les entreprises à élaborer un plan efficace pour réussir leur migration vers le cloud. AWS La CAF organise ses conseils en six domaines prioritaires appelés perspectives : les affaires, les personnes, la gouvernance, les plateformes, la sécurité et les opérations. Les perspectives d'entreprise, de personnes et de gouvernance mettent l'accent sur les compétences et les processus métier, tandis que les perspectives relatives à la plateforme, à la sécurité et aux opérations se concentrent sur les compétences et les processus techniques. Par exemple, la perspective liée aux personnes cible les parties prenantes qui s'occupent des ressources humaines (RH), des fonctions de dotation en personnel et de la gestion des personnes. Dans cette perspective, la AWS CAF fournit des conseils pour le développement du personnel, la formation et les communications afin de préparer

l'organisation à une adoption réussie du cloud. Pour plus d'informations, veuillez consulter le site Web AWS CAF et le livre blanc AWS CAF.

AWS Cadre de qualification de la charge de travail (AWS WQF)

Outil qui évalue les charges de travail liées à la migration des bases de données, recommande des stratégies de migration et fournit des estimations de travail. AWS Le WQF est inclus avec AWS Schema Conversion Tool (AWS SCT). Il analyse les schémas de base de données et les objets de code, le code d'application, les dépendances et les caractéristiques de performance, et fournit des rapports d'évaluation.

B

mauvais bot

Un bot destiné à perturber ou à nuire à des individus ou à des organisations.

BCP

Consultez la section Planification de la continuité des activités.

graphique de comportement

Vue unifiée et interactive des comportements des ressources et des interactions au fil du temps. Vous pouvez utiliser un graphique de comportement avec Amazon Detective pour examiner les tentatives de connexion infructueuses, les appels d'API suspects et les actions similaires. Pour plus d'informations, veuillez consulter Data in a behavior graph dans la documentation Detective.

système de poids fort

Système qui stocke d'abord l'octet le plus significatif. Voir aussi endianité.

classification binaire

Processus qui prédit un résultat binaire (l'une des deux classes possibles). Par exemple, votre modèle de machine learning peut avoir besoin de prévoir des problèmes tels que « Cet e-mail est-il du spam ou non ? » ou « Ce produit est-il un livre ou une voiture ? ».

filtre de Bloom

Structure de données probabiliste et efficace en termes de mémoire qui est utilisée pour tester si un élément fait partie d'un ensemble.

déploiement bleu/vert

Stratégie de déploiement dans laquelle vous créez deux environnements distincts mais identiques. Vous exécutez la version actuelle de l'application dans un environnement (bleu) et la nouvelle version de l'application dans l'autre environnement (vert). Cette stratégie vous permet de revenir rapidement en arrière avec un impact minimal.

bot

Application logicielle qui exécute des tâches automatisées sur Internet et simule l'activité ou l'interaction humaine. Certains robots sont utiles ou bénéfiques, comme les robots d'exploration Web qui indexent des informations sur Internet. D'autres robots, connus sous le nom de mauvais robots, sont destinés à perturber ou à nuire à des individus ou à des organisations.

botnet

Réseaux de robots infectés par des logiciels malveillants et contrôlés par une seule entité, connue sous le nom d'herder ou d'opérateur de bots. Les botnets sont le mécanisme le plus connu pour faire évoluer les bots et leur impact.

branche

Zone contenue d'un référentiel de code. La première branche créée dans un référentiel est la branche principale. Vous pouvez créer une branche à partir d'une branche existante, puis développer des fonctionnalités ou corriger des bogues dans la nouvelle branche. Une branche que vous créez pour générer une fonctionnalité est communément appelée branche de fonctionnalités. Lorsque la fonctionnalité est prête à être publiée, vous fusionnez à nouveau la branche de fonctionnalités dans la branche principale. Pour plus d'informations, consultez À propos des branches (GitHub documentation).

accès par brise-vitre

Dans des circonstances exceptionnelles et par le biais d'un processus approuvé, c'est un moyen rapide pour un utilisateur d'accéder à un accès auquel Compte AWS il n'est généralement pas autorisé. Pour plus d'informations, consultez l'indicateur Implementation break-glass procedures dans le guide Well-Architected AWS .

stratégie existante (brownfield)

L'infrastructure existante de votre environnement. Lorsque vous adoptez une stratégie existante pour une architecture système, vous concevez l'architecture en fonction des contraintes des systèmes et de l'infrastructure actuels. Si vous étendez l'infrastructure existante, vous pouvez combiner des politiques brownfield (existantes) et greenfield (inédites).

cache de tampon

Zone de mémoire dans laquelle sont stockées les données les plus fréquemment consultées.

capacité métier

Ce que fait une entreprise pour générer de la valeur (par exemple, les ventes, le service client ou le marketing). Les architectures de microservices et les décisions de développement peuvent être dictées par les capacités métier. Pour plus d'informations, veuillez consulter la section Organisation en fonction des capacités métier du livre blanc Exécution de microservices conteneurisés sur AWS.

planification de la continuité des activités (BCP)

Plan qui tient compte de l'impact potentiel d'un événement perturbateur, tel qu'une migration à grande échelle, sur les opérations, et qui permet à une entreprise de reprendre ses activités rapidement.

C

CAF

Voir le cadre d'adoption du AWS cloud.

déploiement de Canary

Diffusion lente et progressive d'une version pour les utilisateurs finaux. Lorsque vous êtes sûr, vous déployez la nouvelle version et remplacez la version actuelle dans son intégralité.

CCoE

Voir le Centre d'excellence du cloud.

CDC

Consultez la section Capture des données de modification.

capture des données de modification (CDC)

Processus de suivi des modifications apportées à une source de données, telle qu'une table de base de données, et d'enregistrement des métadonnées relatives à ces modifications. Vous pouvez utiliser la CDC à diverses fins, telles que l'audit ou la réplication des modifications dans un système cible afin de maintenir la synchronisation.

ingénierie du chaos

Introduire intentionnellement des défaillances ou des événements perturbateurs pour tester la résilience d'un système. Vous pouvez utiliser AWS Fault Injection Service (AWS FIS) pour effectuer des expériences qui stressent vos AWS charges de travail et évaluer leur réponse.

CI/CD

Découvrez l'intégration continue et la livraison continue.

classification

Processus de catégorisation qui permet de générer des prédictions. Les modèles de ML pour les problèmes de classification prédisent une valeur discrète. Les valeurs discrètes se distinguent toujours les unes des autres. Par exemple, un modèle peut avoir besoin d'évaluer la présence ou non d'une voiture sur une image.

chiffrement côté client

Chiffrement des données localement, avant que la cible ne les AWS service reçoive.

Centre d'excellence cloud (CCoE)

Une équipe multidisciplinaire qui dirige les efforts d'adoption du cloud au sein d'une organisation, notamment en développant les bonnes pratiques en matière de cloud, en mobilisant des ressources, en établissant des délais de migration et en guidant l'organisation dans le cadre de transformations à grande échelle. Pour plus d'informations, consultez les articles du CCoE sur le blog de stratégie AWS Cloud d'entreprise.

cloud computing

Technologie cloud généralement utilisée pour le stockage de données à distance et la gestion des appareils IoT. Le cloud computing est généralement associé à la technologie informatique de pointe.

modèle d'exploitation du cloud

Dans une organisation informatique, modèle d'exploitation utilisé pour créer, faire évoluer et optimiser un ou plusieurs environnements cloud. Pour plus d'informations, consultez la section Création de votre modèle d'exploitation cloud.

étapes d'adoption du cloud

Les quatre phases que les entreprises traversent généralement lorsqu'elles migrent vers AWS Cloud :

- Projet : exécution de quelques projets liés au cloud à des fins de preuve de concept et d'apprentissage
- Base : réaliser des investissements fondamentaux pour mettre à l'échelle l'adoption du cloud (par exemple, en créant une zone de destination, en définissant un CCoE ou en établissant un modèle opérationnel)
- Migration : migration d'applications individuelles
- Réinvention : optimisation des produits et services et innovation dans le cloud

Ces étapes ont été définies par Stephen Orban dans le billet de blog The Journey Toward Cloud-First & the Stages of Adoption publié sur le blog AWS Cloud Enterprise Strategy. Pour plus d'informations sur leur lien avec la stratégie de AWS migration, consultez le guide de préparation à la migration.

CMDB

Voir base de données de gestion de configuration.

référentiel de code

Emplacement où le code source et d'autres ressources, comme la documentation, les exemples et les scripts, sont stockés et mis à jour par le biais de processus de contrôle de version. Les référentiels cloud courants incluent GitHub ou AWS CodeCommit. Chaque version du code est appelée branche. Dans une structure de microservice, chaque référentiel est consacré à une seule fonctionnalité. Un seul pipeline CI/CD peut utiliser plusieurs référentiels.

cache passif

Cache tampon vide, mal rempli ou contenant des données obsolètes ou non pertinentes. Cela affecte les performances, car l'instance de base de données doit lire à partir de la mémoire principale ou du disque, ce qui est plus lent que la lecture à partir du cache tampon.

données gelées

Données rarement consultées et généralement historiques. Lorsque vous interrogez ce type de données, les requêtes lentes sont généralement acceptables. Le transfert de ces données vers des niveaux ou classes de stockage moins performants et moins coûteux peut réduire les coûts.

vision par ordinateur (CV)

Domaine de l'IA qui utilise l'apprentissage automatique pour analyser et extraire des informations à partir de formats visuels tels que des images numériques et des vidéos. Par exemple, AWS

Panorama propose des appareils qui ajoutent des CV aux réseaux de caméras locaux, et Amazon SageMaker fournit des algorithmes de traitement d'image pour les CV.

dérive de configuration

Pour une charge de travail, une modification de configuration par rapport à l'état attendu. Cela peut entraîner une non-conformité de la charge de travail, et cela est généralement progressif et involontaire.

base de données de gestion des configurations (CMDB)

Référentiel qui stocke et gère les informations relatives à une base de données et à son environnement informatique, y compris les composants matériels et logiciels ainsi que leurs configurations. Vous utilisez généralement les données d'une CMDB lors de la phase de découverte et d'analyse du portefeuille de la migration.

pack de conformité

Ensemble de AWS Config règles et d'actions correctives que vous pouvez assembler pour personnaliser vos contrôles de conformité et de sécurité. Vous pouvez déployer un pack de conformité en tant qu'entité unique dans une région Compte AWS et, ou au sein d'une organisation, à l'aide d'un modèle YAML. Pour plus d'informations, consultez la section Packs de conformité dans la AWS Config documentation.

intégration continue et livraison continue (CI/CD)

Processus d'automatisation des étapes source, de génération, de test, intermédiaire et de production du processus de publication du logiciel. CI/CD est communément décrit comme un pipeline. CI/CD peut vous aider à automatiser les processus, à améliorer la productivité, à améliorer la qualité du code et à accélérer les livraisons. Pour plus d'informations, veuillez consulter Avantages de la livraison continue. CD peut également signifier déploiement continu. Pour plus d'informations, veuillez consulter Livraison continue et déploiement continu.

CV

Voir vision par ordinateur.

D

données au repos

Données stationnaires dans votre réseau, telles que les données stockées.

classification des données

Processus permettant d'identifier et de catégoriser les données de votre réseau en fonction de leur sévérité et de leur sensibilité. Il s'agit d'un élément essentiel de toute stratégie de gestion des risques de cybersécurité, car il vous aide à déterminer les contrôles de protection et de conservation appropriés pour les données. La classification des données est une composante du pilier de sécurité du AWS Well-Architected Framework. Pour plus d'informations, veuillez consulter Classification des données.

dérive des données

Une variation significative entre les données de production et les données utilisées pour entraîner un modèle ML, ou une modification significative des données d'entrée au fil du temps. La dérive des données peut réduire la qualité, la précision et l'équité globales des prédictions des modèles ML.

données en transit

Données qui circulent activement sur votre réseau, par exemple entre les ressources du réseau.

maillage de données

Un cadre architectural qui fournit une propriété des données distribuée et décentralisée avec une gestion et une gouvernance centralisées.

minimisation des données

Le principe de collecte et de traitement des seules données strictement nécessaires. La pratique de la minimisation des données AWS Cloud peut réduire les risques liés à la confidentialité, les coûts et l'empreinte carbone de vos analyses.

périmètre de données

Ensemble de garde-fous préventifs dans votre AWS environnement qui permettent de garantir que seules les identités fiables accèdent aux ressources fiables des réseaux attendus. Pour plus d'informations, voir Création d'un périmètre de données sur AWS.

prétraitement des données

Pour transformer les données brutes en un format facile à analyser par votre modèle de ML. Le prétraitement des données peut impliquer la suppression de certaines colonnes ou lignes et le traitement des valeurs manquantes, incohérentes ou en double.

provenance des données

Le processus de suivi de l'origine et de l'historique des données tout au long de leur cycle de vie, par exemple la manière dont les données ont été générées, transmises et stockées.

sujet des données

Personne dont les données sont collectées et traitées.

entrepôt des données

Un système de gestion des données qui prend en charge les informations commerciales, telles que les analyses. Les entrepôts de données contiennent généralement de grandes quantités de données historiques et sont généralement utilisés pour les requêtes et les analyses.

langage de définition de base de données (DDL)

Instructions ou commandes permettant de créer ou de modifier la structure des tables et des objets dans une base de données.

langage de manipulation de base de données (DML)

Instructions ou commandes permettant de modifier (insérer, mettre à jour et supprimer) des informations dans une base de données.

DDL

Voir langage de définition de base de données.

ensemble profond

Sert à combiner plusieurs modèles de deep learning à des fins de prédiction. Vous pouvez utiliser des ensembles profonds pour obtenir une prévision plus précise ou pour estimer l'incertitude des prédictions.

deep learning

Un sous-champ de ML qui utilise plusieurs couches de réseaux neuronaux artificiels pour identifier le mappage entre les données d'entrée et les variables cibles d'intérêt.

defense-in-depth

Approche de la sécurité de l'information dans laquelle une série de mécanismes et de contrôles de sécurité sont judicieusement répartis sur l'ensemble d'un réseau informatique afin de protéger la confidentialité, l'intégrité et la disponibilité du réseau et des données qu'il contient. Lorsque vous adoptez cette stratégie AWS, vous ajoutez plusieurs contrôles à différentes couches de

la AWS Organizations structure afin de sécuriser les ressources. Par exemple, une defense-in-depth approche peut combiner l'authentification multifactorielle, la segmentation du réseau et le chiffrement.

administrateur délégué

Dans AWS Organizations, un service compatible peut enregistrer un compte AWS membre pour administrer les comptes de l'organisation et gérer les autorisations pour ce service. Ce compte est appelé administrateur délégué pour ce service. Pour plus d'informations et une liste des services compatibles, veuillez consulter la rubrique Services qui fonctionnent avec AWS Organizations dans la documentation AWS Organizations .

déploiement

Processus de mise à disposition d'une application, de nouvelles fonctionnalités ou de corrections de code dans l'environnement cible. Le déploiement implique la mise en œuvre de modifications dans une base de code, puis la génération et l'exécution de cette base de code dans les environnements de l'application.

environnement de développement

Voir environnement.

contrôle de détection

Contrôle de sécurité conçu pour détecter, journaliser et alerter après la survenue d'un événement. Ces contrôles constituent une deuxième ligne de défense et vous alertent en cas d'événements de sécurité qui ont contourné les contrôles préventifs en place. Pour plus d'informations, veuillez consulter la rubrique Contrôles de détection dans Implementing security controls on AWS.

cartographie de la chaîne de valeur du développement (DVSM)

Processus utilisé pour identifier et hiérarchiser les contraintes qui nuisent à la rapidité et à la qualité du cycle de vie du développement logiciel. DVSM étend le processus de cartographie de la chaîne de valeur initialement conçu pour les pratiques de production allégée. Il met l'accent sur les étapes et les équipes nécessaires pour créer et transférer de la valeur tout au long du processus de développement logiciel.

jumeau numérique

Représentation virtuelle d'un système réel, tel qu'un bâtiment, une usine, un équipement industriel ou une ligne de production. Les jumeaux numériques prennent en charge la maintenance prédictive, la surveillance à distance et l'optimisation de la production.

tableau des dimensions

Dans un schéma en étoile, table plus petite contenant les attributs de données relatifs aux données quantitatives d'une table de faits. Les attributs des tables de dimensions sont généralement des champs de texte ou des nombres discrets qui se comportent comme du texte. Ces attributs sont couramment utilisés pour la contrainte des requêtes, le filtrage et l'étiquetage des ensembles de résultats.

catastrophe

Un événement qui empêche une charge de travail ou un système d'atteindre ses objectifs commerciaux sur son site de déploiement principal. Ces événements peuvent être des catastrophes naturelles, des défaillances techniques ou le résultat d'actions humaines, telles qu'une mauvaise configuration involontaire ou une attaque de logiciel malveillant.

reprise après sinistre (DR)

La stratégie et le processus que vous utilisez pour minimiser les temps d'arrêt et les pertes de données causés par un sinistre. Pour plus d'informations, consultez Disaster Recovery of Workloads on AWS : Recovery in the Cloud in the AWS Well-Architected Framework.

DML

Voir langage de manipulation de base de données.

conception axée sur le domaine

Approche visant à développer un système logiciel complexe en connectant ses composants à des domaines évolutifs, ou objectifs métier essentiels, que sert chaque composant. Ce concept a été introduit par Eric Evans dans son ouvrage Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston : Addison-Wesley Professional, 2003). Pour plus d'informations sur l'utilisation du design piloté par domaine avec le modèle de figuier étrangleur, veuillez consulter Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

Consultez la section Reprise après sinistre.

détection de dérive

Suivi des écarts par rapport à une configuration de référence. Par exemple, vous pouvez l'utiliser AWS CloudFormation pour détecter la dérive des ressources du système ou AWS Control Tower

pour détecter les modifications de votre zone d'atterrissage susceptibles d'affecter le respect des exigences de gouvernance.

DVSM

Voir la cartographie de la chaîne de valeur du développement.

E

EDA

Voir analyse exploratoire des données.

informatique de périphérie

Technologie qui augmente la puissance de calcul des appareils intelligents en périphérie d'un réseau IoT. Comparé au cloud computing, l'informatique de pointe peut réduire la latence des communications et améliorer le temps de réponse.

chiffrement

Processus informatique qui transforme des données en texte clair, lisibles par l'homme, en texte chiffré.

clé de chiffrement

Chaîne cryptographique de bits aléatoires générée par un algorithme cryptographique. La longueur des clés peut varier, et chaque clé est conçue pour être imprévisible et unique.

endianisme

Ordre selon lequel les octets sont stockés dans la mémoire de l'ordinateur. Les systèmes de poids fort stockent d'abord l'octet le plus significatif. Les systèmes de poids faible stockent d'abord l'octet le moins significatif.

point de terminaison

Voir point de terminaison de service.

service de point de terminaison

Service que vous pouvez héberger sur un cloud privé virtuel (VPC) pour le partager avec d'autres utilisateurs. Vous pouvez créer un service de point de terminaison avec AWS PrivateLink et accorder des autorisations à d'autres principaux Comptes AWS ou à AWS Identity and Access Management (IAM) principaux. Ces comptes ou principaux peuvent se connecter à votre

service de point de terminaison de manière privée en créant des points de terminaison d'un VPC d'interface. Pour plus d'informations, veuillez consulter Création d'un service de point de terminaison dans la documentation Amazon Virtual Private Cloud (Amazon VPC).

planification des ressources d'entreprise (ERP)

Système qui automatise et gère les principaux processus métier (tels que la comptabilité, le MES et la gestion de projet) pour une entreprise.

chiffrement d'enveloppe

Processus de chiffrement d'une clé de chiffrement à l'aide d'une autre clé de chiffrement. Pour plus d'informations, consultez la section Chiffrement des enveloppes dans la documentation AWS Key Management Service (AWS KMS).

environment

Instance d'une application en cours d'exécution. Les types d'environnement les plus courants dans le cloud computing sont les suivants :

- Environnement de développement : instance d'une application en cours d'exécution à laquelle seule l'équipe principale chargée de la maintenance de l'application peut accéder. Les environnements de développement sont utilisés pour tester les modifications avant de les promouvoir dans les environnements supérieurs. Ce type d'environnement est parfois appelé environnement de test.
- Environnements inférieurs : tous les environnements de développement d'une application, tels que ceux utilisés pour les générations et les tests initiaux.
- Environnement de production : instance d'une application en cours d'exécution à laquelle les utilisateurs finaux peuvent accéder. Dans un pipeline CI/CD, l'environnement de production est le dernier environnement de déploiement.
- Environnements supérieurs : tous les environnements accessibles aux utilisateurs autres que l'équipe de développement principale. Ils peuvent inclure un environnement de production, des environnements de préproduction et des environnements pour les tests d'acceptation par les utilisateurs.

épopée

Dans les méthodologies agiles, catégories fonctionnelles qui aident à organiser et à prioriser votre travail. Les épopées fournissent une description détaillée des exigences et des tâches d'implémentation. Par exemple, les points forts de la AWS CAF en matière de sécurité incluent la gestion des identités et des accès, les contrôles de détection, la sécurité des infrastructures,

la protection des données et la réponse aux incidents. Pour plus d'informations sur les épopées dans la stratégie de migration AWS , veuillez consulter le guide d'implémentation du programme.

ERP

Voir Planification des ressources d'entreprise.

analyse exploratoire des données (EDA)

Processus d'analyse d'un jeu de données pour comprendre ses principales caractéristiques. Vous collectez ou agrégez des données, puis vous effectuez des enquêtes initiales pour trouver des modèles, détecter des anomalies et vérifier les hypothèses. L'EDA est réalisée en calculant des statistiques récapitulatives et en créant des visualisations de données.

F

tableau des faits

La table centrale dans un schéma en étoile. Il stocke des données quantitatives sur les opérations commerciales. Généralement, une table de faits contient deux types de colonnes : celles qui contiennent des mesures et celles qui contiennent une clé étrangère pour une table de dimensions.

échouer rapidement

Une philosophie qui utilise des tests fréquents et progressifs pour réduire le cycle de vie du développement. C'est un élément essentiel d'une approche agile.

limite d'isolation des défauts

Dans le AWS Cloud, une limite telle qu'une zone de disponibilité Région AWS, un plan de contrôle ou un plan de données qui limite l'effet d'une panne et contribue à améliorer la résilience des charges de travail. Pour plus d'informations, consultez la section Limites d'isolation des AWS pannes.

branche de fonctionnalités

Voir la succursale.

fonctionnalités

Les données d'entrée que vous utilisez pour faire une prédiction. Par exemple, dans un contexte de fabrication, les fonctionnalités peuvent être des images capturées périodiquement à partir de la ligne de fabrication.

importance des fonctionnalités

Le niveau d'importance d'une fonctionnalité pour les prédictions d'un modèle. Il s'exprime généralement sous la forme d'un score numérique qui peut être calculé à l'aide de différentes techniques, telles que la méthode Shapley Additive Explanations (SHAP) et les gradients intégrés. Pour plus d'informations, voir Interprétabilité du modèle d'apprentissage automatique avec :AWS.

transformation de fonctionnalité

Optimiser les données pour le processus de ML, notamment en enrichissant les données avec des sources supplémentaires, en mettant à l'échelle les valeurs ou en extrayant plusieurs ensembles d'informations à partir d'un seul champ de données. Cela permet au modèle de ML de tirer parti des données. Par exemple, si vous décomposez la date « 2021-05-27 00:15:37 » en « 2021 », « mai », « jeudi » et « 15 », vous pouvez aider l'algorithme d'apprentissage à apprendre des modèles nuancés associés à différents composants de données.

FGAC

Découvrez le contrôle d'accès détaillé.

contrôle d'accès détaillé (FGAC)

Utilisation de plusieurs conditions pour autoriser ou refuser une demande d'accès.

migration instantanée (flash-cut)

Méthode de migration de base de données qui utilise la réplication continue des données via la capture des données de modification afin de migrer les données dans les plus brefs délais, au lieu d'utiliser une approche progressive. L'objectif est de réduire au maximum les temps d'arrêt.

G

blocage géographique

Voir les restrictions géographiques.

restrictions géographiques (blocage géographique)

Sur Amazon CloudFront, option permettant d'empêcher les utilisateurs de certains pays d'accéder aux distributions de contenu. Vous pouvez utiliser une liste d'autorisation ou une liste de blocage pour spécifier les pays approuvés et interdits. Pour plus d'informations, consultez la section Restreindre la distribution géographique de votre contenu dans la CloudFront documentation.

Flux de travail Gitflow

Approche dans laquelle les environnements inférieurs et supérieurs utilisent différentes branches dans un référentiel de code source. Le flux de travail Gitflow est considéré comme existant, et le flux de travail basé sur les troncs est l'approche moderne préférée.

stratégie inédite

L'absence d'infrastructures existantes dans un nouvel environnement. Lorsque vous adoptez une stratégie inédite pour une architecture système, vous pouvez sélectionner toutes les nouvelles technologies sans restriction de compatibilité avec l'infrastructure existante, également appelée brownfield. Si vous étendez l'infrastructure existante, vous pouvez combiner des politiques brownfield (existantes) et greenfield (inédites).

barrière de protection

Règle de haut niveau qui permet de régir les ressources, les politiques et la conformité au sein des unités d'organisation (UO). Les barrières de protection préventives appliquent des politiques pour garantir l'alignement sur les normes de conformité. Elles sont mises en œuvre à l'aide de politiques de contrôle des services et de limites des autorisations IAM. Les barrières de protection de détection détectent les violations des politiques et les problèmes de conformité, et génèrent des alertes pour y remédier. Ils sont implémentés à l'aide d'Amazon AWS Config AWS Security Hub GuardDuty AWS Trusted Advisor, d'Amazon Inspector et de AWS Lambda contrôles personnalisés.

H

HA

Découvrez la haute disponibilité.

migration de base de données hétérogène

Migration de votre base de données source vers une base de données cible qui utilise un moteur de base de données différent (par exemple, Oracle vers Amazon Aurora). La migration hétérogène fait généralement partie d'un effort de réarchitecture, et la conversion du schéma peut s'avérer une tâche complexe. AWS propose AWS SCT qui facilite les conversions de schémas.

haute disponibilité (HA)

Capacité d'une charge de travail à fonctionner en continu, sans intervention, en cas de difficultés ou de catastrophes. Les systèmes HA sont conçus pour basculer automatiquement, fournir

constamment des performances de haute qualité et gérer différentes charges et défaillances avec un impact minimal sur les performances.

modernisation de l'historien

Approche utilisée pour moderniser et mettre à niveau les systèmes de technologie opérationnelle (OT) afin de mieux répondre aux besoins de l'industrie manufacturière. Un historien est un type de base de données utilisé pour collecter et stocker des données provenant de diverses sources dans une usine.

migration de base de données homogène

Migration de votre base de données source vers une base de données cible qui partage le même moteur de base de données (par exemple, Microsoft SQL Server vers Amazon RDS for SQL Server). La migration homogène s'inscrit généralement dans le cadre d'un effort de réhébergement ou de replateforme. Vous pouvez utiliser les utilitaires de base de données natifs pour migrer le schéma.

données chaudes

Données fréquemment consultées, telles que les données en temps réel ou les données translationnelles récentes. Ces données nécessitent généralement un niveau ou une classe de stockage à hautes performances pour fournir des réponses rapides aux requêtes.

correctif

Solution d'urgence à un problème critique dans un environnement de production. En raison de son urgence, un correctif est généralement créé en dehors du flux de travail de DevOps publication habituel.

période de soins intensifs

Immédiatement après le basculement, période pendant laquelle une équipe de migration gère et surveille les applications migrées dans le cloud afin de résoudre les problèmes éventuels. En règle générale, cette période dure de 1 à 4 jours. À la fin de la période de soins intensifs, l'équipe de migration transfère généralement la responsabilité des applications à l'équipe des opérations cloud.

I

IaC

Considérez l'infrastructure comme un code.

politique basée sur l'identité

Politique attachée à un ou plusieurs principaux IAM qui définit leurs autorisations au sein de l' AWS Cloud environnement.

application inactive

Application dont l'utilisation moyenne du processeur et de la mémoire se situe entre 5 et 20 % sur une période de 90 jours. Dans un projet de migration, il est courant de retirer ces applications ou de les retenir sur site.

IIoT

Voir Internet industriel des objets.

infrastructure immuable

Modèle qui déploie une nouvelle infrastructure pour les charges de travail de production au lieu de mettre à jour, d'appliquer des correctifs ou de modifier l'infrastructure existante. Les infrastructures immuables sont intrinsèquement plus cohérentes, fiables et prévisibles que les infrastructures mutables. Pour plus d'informations, consultez les meilleures pratiques de déploiement à l'aide d'une infrastructure immuable dans le AWS Well-Architected Framework.

VPC entrant (d'entrée)

Dans une architecture AWS multi-comptes, un VPC qui accepte, inspecte et achemine les connexions réseau depuis l'extérieur d'une application. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

migration incrémentielle

Stratégie de basculement dans le cadre de laquelle vous migrez votre application par petites parties au lieu d'effectuer un basculement complet unique. Par exemple, il se peut que vous ne transfériez que quelques microservices ou utilisateurs vers le nouveau système dans un

premier temps. Après avoir vérifié que tout fonctionne correctement, vous pouvez transférer progressivement des microservices ou des utilisateurs supplémentaires jusqu'à ce que vous puissiez mettre hors service votre système hérité. Cette stratégie réduit les risques associés aux migrations de grande ampleur.

Industry 4.0

Terme introduit par Klaus Schwab en 2016 pour désigner la modernisation des processus de fabrication grâce aux avancées en matière de connectivité, de données en temps réel, d'automatisation, d'analyse et d'IA/ML.

infrastructure

Ensemble des ressources et des actifs contenus dans l'environnement d'une application.

infrastructure en tant que code (IaC)

Processus de mise en service et de gestion de l'infrastructure d'une application via un ensemble de fichiers de configuration. IaC est conçue pour vous aider à centraliser la gestion de l'infrastructure, à normaliser les ressources et à mettre à l'échelle rapidement afin que les nouveaux environnements soient reproductibles, fiables et cohérents.

internet industriel des objets (IIoT)

L'utilisation de capteurs et d'appareils connectés à Internet dans les secteurs industriels tels que la fabrication, l'énergie, l'automobile, les soins de santé, les sciences de la vie et l'agriculture. Pour plus d'informations, veuillez consulter Building an industrial Internet of Things (IIoT) digital transformation strategy.

VPC d'inspection

Dans une architecture AWS multi-comptes, un VPC centralisé qui gère les inspections du trafic réseau entre les VPC (identiques ou Régions AWS différents), Internet et les réseaux sur site. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

Internet des objets (IoT)

Réseau d'objets physiques connectés dotés de capteurs ou de processeurs intégrés qui communiquent avec d'autres appareils et systèmes via Internet ou via un réseau de communication local. Pour plus d'informations, veuillez consulter la section Qu'est-ce que l'IoT ?.

interprétabilité

Caractéristique d'un modèle de machine learning qui décrit dans quelle mesure un être humain peut comprendre comment les prédictions du modèle dépendent de ses entrées. Pour plus d'informations, veuillez consulter Machine learning model interpretability with AWS.

IoT

Voir Internet des objets.

Bibliothèque d'informations informatiques (ITIL)

Ensemble de bonnes pratiques pour proposer des services informatiques et les aligner sur les exigences métier. L'ITIL constitue la base de l'ITSM.

gestion des services informatiques (ITSM)

Activités associées à la conception, à la mise en œuvre, à la gestion et à la prise en charge de services informatiques d'une organisation. Pour plus d'informations sur l'intégration des opérations cloud aux outils ITSM, veuillez consulter le guide d'intégration des opérations.

ITIL

Consultez la bibliothèque d'informations informatiques.

ITSM

Consultez la section Gestion des services informatiques.

L

contrôle d'accès basé sur des étiquettes (LBAC)

Une implémentation du contrôle d'accès obligatoire (MAC) dans laquelle une valeur d'étiquette de sécurité est explicitement attribuée aux utilisateurs et aux données elles-mêmes. L'intersection entre l'étiquette de sécurité utilisateur et l'étiquette de sécurité des données détermine les lignes et les colonnes visibles par l'utilisateur.

zone de destination

Une zone d'atterrissage est un AWS environnement multi-comptes bien conçu, évolutif et sécurisé. Il s'agit d'un point de départ à partir duquel vos entreprises peuvent rapidement lancer et déployer des charges de travail et des applications en toute confiance dans leur environnement

de sécurité et d'infrastructure. Pour plus d'informations sur les zones de destination, veuillez consulter Setting up a secure and scalable multi-account AWS environment.

migration de grande envergure

Migration de 300 serveurs ou plus.

LBAC

Voir contrôle d'accès basé sur des étiquettes.

principe de moindre privilège

Bonne pratique de sécurité qui consiste à accorder les autorisations minimales nécessaires à l'exécution d'une tâche. Pour plus d'informations, veuillez consulter la rubrique Accorder les autorisations de moindre privilège dans la documentation IAM.

lift and shift

Voir 7 Rs.

système de poids faible

Système qui stocke d'abord l'octet le moins significatif. Voir aussi endianité.

environnements inférieurs

Voir environnement.

M

machine learning (ML)

Type d'intelligence artificielle qui utilise des algorithmes et des techniques pour la reconnaissance et l'apprentissage de modèles. Le ML analyse et apprend à partir de données enregistrées, telles que les données de l'Internet des objets (IoT), pour générer un modèle statistique basé sur des modèles. Pour plus d'informations, veuillez consulter Machine Learning.

branche principale

Voir la succursale.

malware

Logiciel conçu pour compromettre la sécurité ou la confidentialité de l'ordinateur. Les logiciels malveillants peuvent perturber les systèmes informatiques, divulguer des informations sensibles

ou obtenir un accès non autorisé. Parmi les malwares, on peut citer les virus, les vers, les rançongiciels, les chevaux de Troie, les logiciels espions et les enregistreurs de frappe.

services gérés

AWS services qui AWS gère la couche d'infrastructure, le système d'exploitation et les plateformes, et vous accédez aux points de terminaison pour stocker et récupérer des données. Amazon Simple Storage Service (Amazon S3) et Amazon DynamoDB sont des exemples de services gérés. Ils sont également connus sous le nom de services abstraits.

système d'exécution de la fabrication (MES)

Un système logiciel pour le suivi, la surveillance, la documentation et le contrôle des processus de production qui convertissent les matières premières en produits finis dans l'atelier.

MAP

Voir Migration Acceleration Program.

mécanisme

Processus complet au cours duquel vous créez un outil, favorisez son adoption, puis inspectez les résultats afin de procéder aux ajustements nécessaires. Un mécanisme est un cycle qui se renforce et s'améliore lorsqu'il fonctionne. Pour plus d'informations, voir Création de mécanismes dans le cadre AWS Well-Architected.

compte membre

Tous, à l' Comptes AWS exception du compte de gestion, qui font partie d'une organisation dans AWS Organizations. Un compte ne peut être membre que d'une seule organisation à la fois.

MAILLES

Voir le système d'exécution de la fabrication.

Transport télémétrique en file d'attente de messages (MQTT)

Protocole de communication léger machine-to-machine (M2M), basé sur le modèle de publication/d'abonnement, pour les appareils IoT aux ressources limitées.

microservice

Petit service indépendant qui communique via des API bien définies et qui est généralement détenu par de petites équipes autonomes. Par exemple, un système d'assurance peut inclure des microservices qui mappent à des capacités métier, telles que les ventes ou le marketing, ou

à des sous-domaines, tels que les achats, les réclamations ou l'analytique. Les avantages des microservices incluent l'agilité, la flexibilité de la mise à l'échelle, la facilité de déploiement, la réutilisation du code et la résilience. Pour plus d'informations, consultez la section Intégration de microservices à l'aide de services AWS sans serveur.

architecture de microservices

Approche de création d'une application avec des composants indépendants qui exécutent chaque processus d'application en tant que microservice. Ces microservices communiquent via une interface bien définie à l'aide d'API légères. Chaque microservice de cette architecture peut être mis à jour, déployé et mis à l'échelle pour répondre à la demande de fonctions spécifiques d'une application. Pour plus d'informations, consultez la section Implémentation de microservices sur AWS.

Programme d'accélération des migrations (MAP)

Un AWS programme qui fournit un support de conseil, des formations et des services pour aider les entreprises à établir une base opérationnelle solide pour passer au cloud, et pour aider à compenser le coût initial des migrations. MAP inclut une méthodologie de migration pour exécuter les migrations héritées de manière méthodique, ainsi qu'un ensemble d'outils pour automatiser et accélérer les scénarios de migration courants.

migration à grande échelle

Processus consistant à transférer la majeure partie du portefeuille d'applications vers le cloud par vagues, un plus grand nombre d'applications étant déplacées plus rapidement à chaque vague. Cette phase utilise les bonnes pratiques et les enseignements tirés des phases précédentes pour implémenter une usine de migration d'équipes, d'outils et de processus en vue de rationaliser la migration des charges de travail grâce à l'automatisation et à la livraison agile. Il s'agit de la troisième phase de la stratégie de migration AWS.

usine de migration

Équipes interfonctionnelles qui rationalisent la migration des charges de travail grâce à des approches automatisées et agiles. Les équipes de Migration Factory comprennent généralement les opérations, les analystes commerciaux et les propriétaires, les ingénieurs de migration, les développeurs et les DevOps professionnels travaillant dans le cadre de sprints. Entre 20 et 50 % du portefeuille d'applications d'entreprise est constitué de modèles répétés qui peuvent être optimisés par une approche d'usine. Pour plus d'informations, veuillez consulter la rubrique discussion of migration factories et le guide Cloud Migration Factory dans cet ensemble de contenus.

métadonnées de migration

Informations relatives à l'application et au serveur nécessaires pour finaliser la migration. Chaque modèle de migration nécessite un ensemble de métadonnées de migration différent. Les exemples de métadonnées de migration incluent le sous-réseau cible, le groupe de sécurité et le AWS compte.

modèle de migration

Tâche de migration reproductible qui détaille la stratégie de migration, la destination de la migration et l'application ou le service de migration utilisé. Exemple : réorganisez la migration vers Amazon EC2 AWS avec le service de migration d'applications.

Évaluation du portefeuille de migration (MPA)

Outil en ligne qui fournit des informations pour valider l'analyse de rentabilisation en faveur de la migration vers le. AWS Cloud La MPA propose une évaluation détaillée du portefeuille (dimensionnement approprié des serveurs, tarification, comparaison du coût total de possession, analyse des coûts de migration), ainsi que la planification de la migration (analyse et collecte des données d'applications, regroupement des applications, priorisation des migrations et planification des vagues). L'outil MPA (connexion requise) est disponible gratuitement pour tous les AWS consultants et consultants APN Partner.

Évaluation de la préparation à la migration (MRA)

Processus qui consiste à obtenir des informations sur l'état de préparation d'une organisation au cloud, à identifier les forces et les faiblesses et à élaborer un plan d'action pour combler les lacunes identifiées, à l'aide du AWS CAF. Pour plus d'informations, veuillez consulter le guide de préparation à la migration. La MRA est la première phase de la stratégie de migration AWS.

stratégie de migration

L'approche utilisée pour migrer une charge de travail vers le AWS Cloud. Pour plus d'informations, reportez-vous aux 7 R de ce glossaire et à Mobiliser votre organisation pour accélérer les migrations à grande échelle.

ML

Voir apprentissage automatique.

modernisation

Transformation d'une application obsolète (héritée ou monolithique) et de son infrastructure en un système agile, élastique et hautement disponible dans le cloud afin de réduire les coûts, de

gagner en efficacité et de tirer parti des innovations. Pour plus d'informations, consultez la section Stratégie de modernisation des applications dans le AWS Cloud.

évaluation de la préparation à la modernisation

Évaluation qui permet de déterminer si les applications d'une organisation sont prêtes à être modernisées, d'identifier les avantages, les risques et les dépendances, et qui détermine dans quelle mesure l'organisation peut prendre en charge l'état futur de ces applications. Le résultat de l'évaluation est un plan de l'architecture cible, une feuille de route détaillant les phases de développement et les étapes du processus de modernisation, ainsi qu'un plan d'action pour combler les lacunes identifiées. Pour plus d'informations, consultez la section Évaluation de l'état de préparation à la modernisation des applications dans le AWS Cloud.

applications monolithiques (monolithes)

Applications qui s'exécutent en tant que service unique avec des processus étroitement couplés. Les applications monolithiques ont plusieurs inconvénients. Si une fonctionnalité de l'application connaît un pic de demande, l'architecture entière doit être mise à l'échelle. L'ajout ou l'amélioration des fonctionnalités d'une application monolithique devient également plus complexe lorsque la base de code s'élargit. Pour résoudre ces problèmes, vous pouvez utiliser une architecture de microservices. Pour plus d'informations, veuillez consulter Decomposing monoliths into microservices.

MPA

Voir Évaluation du portefeuille de migration.

MQTT

Voir Message Queuing Telemetry Transport.

classification multi-classes

Processus qui permet de générer des prédictions pour plusieurs classes (prédiction d'un résultat parmi plus de deux). Par exemple, un modèle de ML peut demander « Ce produit est-il un livre, une voiture ou un téléphone ? » ou « Quelle catégorie de produits intéresse le plus ce client ? ».

infrastructure mutable

Modèle qui met à jour et modifie l'infrastructure existante pour les charges de travail de production. Pour améliorer la cohérence, la fiabilité et la prévisibilité, le AWS Well-Architected Framework recommande l'utilisation d'une infrastructure immuable comme meilleure pratique.

O

OAC

Voir Contrôle d'accès à l'origine.

OAI

Voir l'identité d'accès à l'origine.

OCM

Voir gestion du changement organisationnel.

migration hors ligne

Méthode de migration dans laquelle la charge de travail source est supprimée au cours du processus de migration. Cette méthode implique un temps d'arrêt prolongé et est généralement utilisée pour de petites charges de travail non critiques.

OI

Consultez la section Intégration des opérations.

OLA

Voir l'accord au niveau opérationnel.

migration en ligne

Méthode de migration dans laquelle la charge de travail source est copiée sur le système cible sans être mise hors ligne. Les applications connectées à la charge de travail peuvent continuer à fonctionner pendant la migration. Cette méthode implique un temps d'arrêt nul ou minimal et est généralement utilisée pour les charges de travail de production critiques.

OPC-UA

Voir Open Process Communications - Architecture unifiée.

Communications par processus ouvert - Architecture unifiée (OPC-UA)

Un protocole de communication machine-to-machine (M2M) pour l'automatisation industrielle. L'OPC-UA fournit une norme d'interopérabilité avec des schémas de cryptage, d'authentification et d'autorisation des données.

accord au niveau opérationnel (OLA)

Accord qui précise ce que les groupes informatiques fonctionnels s'engagent à fournir les uns aux autres, afin de prendre en charge un contrat de niveau de service (SLA).

examen de l'état de préparation opérationnelle (ORR)

Une liste de questions et de bonnes pratiques associées qui vous aident à comprendre, évaluer, prévenir ou réduire l'ampleur des incidents et des défaillances possibles. Pour plus d'informations, voir Operational Readiness Reviews (ORR) dans le AWS Well-Architected Framework.

technologie opérationnelle (OT)

Systèmes matériels et logiciels qui fonctionnent avec l'environnement physique pour contrôler les opérations, les équipements et les infrastructures industriels. Dans le secteur manufacturier, l'intégration des systèmes OT et des technologies de l'information (IT) est au cœur des transformations de l'industrie 4.0.

intégration des opérations (OI)

Processus de modernisation des opérations dans le cloud, qui implique la planification de la préparation, l'automatisation et l'intégration. Pour en savoir plus, veuillez consulter le guide d'intégration des opérations.

journal de suivi d'organisation

Un parcours créé par AWS CloudTrail qui enregistre tous les événements pour tous les membres Comptes AWS d'une organisation dans AWS Organizations. Ce journal de suivi est créé dans chaque Compte AWS qui fait partie de l'organisation et suit l'activité de chaque compte. Pour plus d'informations, consultez la section Création d'un suivi pour une organisation dans la CloudTrail documentation.

gestion du changement organisationnel (OCM)

Cadre pour gérer les transformations métier majeures et perturbatrices du point de vue des personnes, de la culture et du leadership. L'OCM aide les organisations à se préparer et à effectuer la transition vers de nouveaux systèmes et de nouvelles politiques en accélérant l'adoption des changements, en abordant les problèmes de transition et en favorisant des changements culturels et organisationnels. Dans la stratégie de AWS migration, ce cadre est appelé accélération du personnel, en raison de la rapidité du changement requise dans les projets d'adoption du cloud. Pour plus d'informations, veuillez consulter le guide OCM.

contrôle d'accès d'origine (OAC)

Dans CloudFront, une option améliorée pour restreindre l'accès afin de sécuriser votre contenu Amazon Simple Storage Service (Amazon S3). L'OAC prend en charge tous les compartiments S3 dans leur ensemble Régions AWS, le chiffrement côté serveur avec AWS KMS (SSE-KMS) et les requêtes dynamiques PUT adressées au compartiment S3. DELETE

identité d'accès d'origine (OAI)

Dans CloudFront, une option permettant de restreindre l'accès afin de sécuriser votre contenu Amazon S3. Lorsque vous utilisez OAI, il CloudFront crée un principal auprès duquel Amazon S3 peut s'authentifier. Les principaux authentifiés ne peuvent accéder au contenu d'un compartiment S3 que par le biais d'une distribution spécifique CloudFront . Voir également OAC, qui fournit un contrôle d'accès plus précis et amélioré.

OU

Voir l'examen de l'état de préparation opérationnelle.

DE

Voir technologie opérationnelle.

VPC sortant (de sortie)

Dans une architecture AWS multi-comptes, un VPC qui gère les connexions réseau initiées depuis une application. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

P

limite des autorisations

Politique de gestion IAM attachée aux principaux IAM pour définir les autorisations maximales que peut avoir l'utilisateur ou le rôle. Pour plus d'informations, veuillez consulter la rubrique Limites des autorisations dans la documentation IAM.

informations personnelles identifiables (PII)

Informations qui, lorsqu'elles sont consultées directement ou associées à d'autres données connexes, peuvent être utilisées pour déduire raisonnablement l'identité d'une personne. Les

exemples d'informations personnelles incluent les noms, les adresses et les informations de contact.

PII

Voir les informations personnelles identifiables.

manuel stratégique

Ensemble d'étapes prédéfinies qui capturent le travail associé aux migrations, comme la fourniture de fonctions d'opérations de base dans le cloud. Un manuel stratégique peut revêtir la forme de scripts, de runbooks automatisés ou d'un résumé des processus ou des étapes nécessaires au fonctionnement de votre environnement modernisé.

PLC

Voir contrôleur logique programmable.

PLM

Consultez la section Gestion du cycle de vie des produits.

politique

Objet capable de définir les autorisations (voir la politique basée sur l'identité), de spécifier les conditions d'accès (voir la politique basée sur les ressources) ou de définir les autorisations maximales pour tous les comptes d'une organisation dans AWS Organizations (voir la politique de contrôle des services).

persistance polyglotte

Choix indépendant de la technologie de stockage de données d'un microservice en fonction des modèles d'accès aux données et d'autres exigences. Si vos microservices utilisent la même technologie de stockage de données, ils peuvent rencontrer des difficultés d'implémentation ou présenter des performances médiocres. Les microservices sont plus faciles à mettre en œuvre, atteignent de meilleures performances, ainsi qu'une meilleure capacité de mise à l'échelle s'ils utilisent l'entrepôt de données le mieux adapté à leurs besoins. Pour plus d'informations, veuillez consulter Enabling data persistence in microservices.

évaluation du portefeuille

Processus de découverte, d'analyse et de priorisation du portefeuille d'applications afin de planifier la migration. Pour plus d'informations, veuillez consulter Evaluating migration readiness.

predicate

Une condition de requête qui renvoie `true` ou`false`, généralement située dans une `WHERE` clause.

prédicat pushdown

Technique d'optimisation des requêtes de base de données qui filtre les données de la requête avant le transfert. Cela réduit la quantité de données qui doivent être extraites et traitées à partir de la base de données relationnelle et améliore les performances des requêtes.

contrôle préventif

Contrôle de sécurité conçu pour empêcher qu'un événement ne se produise. Ces contrôles constituent une première ligne de défense pour empêcher tout accès non autorisé ou toute modification indésirable de votre réseau. Pour plus d'informations, veuillez consulter Preventative controls dans Implementing security controls on AWS.

principal

Entité AWS capable d'effectuer des actions et d'accéder aux ressources. Cette entité est généralement un utilisateur root pour un Compte AWS rôle IAM ou un utilisateur. Pour plus d'informations, veuillez consulter la rubrique Principal dans Termes et concepts relatifs aux rôles, dans la documentation IAM.

Confidentialité dès la conception

Une approche de l'ingénierie des systèmes qui prend en compte la confidentialité tout au long du processus d'ingénierie.

zones hébergées privées

Conteneur qui contient des informations concernant la façon dont vous souhaitez qu'Amazon Route 53 réponde aux requêtes DNS pour un domaine et ses sous-domaines dans un ou plusieurs VPC. Pour plus d'informations, veuillez consulter Working with private hosted zones dans la documentation Route 53.

contrôle proactif

Contrôle de sécurité conçu pour empêcher le déploiement de ressources non conformes. Ces contrôles analysent les ressources avant qu'elles ne soient provisionnées. Si la ressource n'est pas conforme au contrôle, elle n'est pas provisionnée. Pour plus d'informations, consultez le guide de référence sur les contrôles dans la AWS Control Tower documentation et consultez la section Contrôles proactifs dans Implémentation des contrôles de sécurité sur AWS.

gestion du cycle de vie des produits (PLM)

Gestion des données et des processus d'un produit tout au long de son cycle de vie, depuis la conception, le développement et le lancement, en passant par la croissance et la maturité, jusqu'au déclin et au retrait.

environnement de production

Voir environnement.

contrôleur logique programmable (PLC)

Dans le secteur manufacturier, un ordinateur hautement fiable et adaptable qui surveille les machines et automatise les processus de fabrication.

pseudonymisation

Processus de remplacement des identifiants personnels dans un ensemble de données par des valeurs fictives. La pseudonymisation peut contribuer à protéger la vie privée. Les données pseudonymisées sont toujours considérées comme des données personnelles.

publier/souscrire (pub/sub)

Modèle qui permet des communications asynchrones entre les microservices afin d'améliorer l'évolutivité et la réactivité. Par exemple, dans un MES basé sur des microservices, un microservice peut publier des messages d'événements sur un canal auquel d'autres microservices peuvent s'abonner. Le système peut ajouter de nouveaux microservices sans modifier le service de publication.

Q

plan de requête

Série d'étapes, telles que des instructions, utilisées pour accéder aux données d'un système de base de données relationnelle SQL.

régression du plan de requêtes

Le cas où un optimiseur de service de base de données choisit un plan moins optimal qu'avant une modification donnée de l'environnement de base de données. Cela peut être dû à des changements en termes de statistiques, de contraintes, de paramètres d'environnement, de liaisons de paramètres de requêtes et de mises à jour du moteur de base de données.

R

Matrice RACI

Voir responsable, responsable, consulté, informé (RACI).

rançongiciel

Logiciel malveillant conçu pour bloquer l'accès à un système informatique ou à des données jusqu'à ce qu'un paiement soit effectué.

Matrice RASCI

Voir responsable, responsable, consulté, informé (RACI).

RCAC

Voir contrôle d'accès aux lignes et aux colonnes.

réplica en lecture

Copie d'une base de données utilisée en lecture seule. Vous pouvez acheminer les requêtes vers le réplica de lecture pour réduire la charge sur votre base de données principale.

réarchitecte

Voir 7 Rs.

objectif de point de récupération (RPO)

Durée maximale acceptable depuis le dernier point de récupération des données. Cela permet de déterminer ce qui est considéré comme une perte de données acceptable entre le dernier point de restauration et l'interruption du service.

objectif de temps de récupération (RTO)

Le délai maximum acceptable entre l'interruption du service et le rétablissement du service.

refactoriser

Voir 7 Rs.

Région

Un ensemble de AWS ressources dans une zone géographique. Chacun Région AWS est isolé et indépendant des autres pour garantir tolérance aux pannes, stabilité et résilience. Pour plus d'informations, voir Spécifier ce que Régions AWS votre compte peut utiliser.

régression

Technique de ML qui prédit une valeur numérique. Par exemple, pour résoudre le problème « Quel sera le prix de vente de cette maison ? », un modèle de ML pourrait utiliser un modèle de régression linéaire pour prédire le prix de vente d'une maison sur la base de faits connus à son sujet (par exemple, la superficie en mètres carrés).

réhéberger

Voir 7 Rs.

version

Dans un processus de déploiement, action visant à promouvoir les modifications apportées à un environnement de production.

déplacer

Voir 7 Rs.

replateforme

Voir 7 Rs.

rachat

Voir 7 Rs.

résilience

La capacité d'une application à résister aux perturbations ou à s'en remettre. La haute disponibilité et la reprise après sinistre sont des considérations courantes lors de la planification de la résilience dans le AWS Cloud. Pour plus d'informations, consultez AWS Cloud Résilience.

politique basée sur les ressources

Politique attachée à une ressource, comme un compartiment Amazon S3, un point de terminaison ou une clé de chiffrement. Ce type de politique précise les principaux auxquels l'accès est autorisé, les actions prises en charge et toutes les autres conditions qui doivent être remplies.

matrice responsable, redevable, consulté et informé (RACI)

Une matrice qui définit les rôles et les responsabilités de toutes les parties impliquées dans les activités de migration et les opérations cloud. Le nom de la matrice est dérivé des types de responsabilité définis dans la matrice : responsable (R), responsable (A), consulté (C) et informé (I). Le type de support (S) est facultatif. Si vous incluez le support, la matrice est appelée matrice RASCI, et si vous l'excluez, elle est appelée matrice RACI.

contrôle réactif

Contrôle de sécurité conçu pour permettre de remédier aux événements indésirables ou aux écarts par rapport à votre référence de sécurité. Pour plus d'informations, veuillez consulter la rubrique Responsive controls dans Implementing security controls on AWS.

retain

Voir 7 Rs.

se retirer

Voir 7 Rs.

rotation

Processus de mise à jour périodique d'un secret pour empêcher un attaquant d'accéder aux informations d'identification.

contrôle d'accès aux lignes et aux colonnes (RCAC)

Utilisation d'expressions SQL simples et flexibles dotées de règles d'accès définies. Le RCAC comprend des autorisations de ligne et des masques de colonnes.

RPO

Voir l'objectif du point de récupération.

RTO

Voir l'objectif en matière de temps de rétablissement.

runbook

Ensemble de procédures manuelles ou automatisées nécessaires à l'exécution d'une tâche spécifique. Elles visent généralement à rationaliser les opérations ou les procédures répétitives présentant des taux d'erreur élevés.

S

SAML 2.0

Un standard ouvert utilisé par de nombreux fournisseurs d'identité (IdPs). Cette fonctionnalité permet l'authentification unique fédérée (SSO), afin que les utilisateurs puissent se connecter AWS Management Console ou appeler les opérations d' AWS API sans que vous ayez à créer

un utilisateur dans IAM pour tous les membres de votre organisation. Pour plus d'informations sur la fédération SAML 2.0, veuillez consulter À propos de la fédération SAML 2.0 dans la documentation IAM.

SCADA

Voir Contrôle de supervision et acquisition de données.

SCP

Voir la politique de contrôle des services.

secret

Dans AWS Secrets Manager des informations confidentielles ou restreintes, telles qu'un mot de passe ou des informations d'identification utilisateur, que vous stockez sous forme cryptée. Il comprend la valeur secrète et ses métadonnées. La valeur secrète peut être binaire, une chaîne unique ou plusieurs chaînes. Pour plus d'informations, voir Que contient le secret d'un Secrets Manager ? dans la documentation de Secrets Manager.

contrôle de sécurité

Barrière de protection technique ou administrative qui empêche, détecte ou réduit la capacité d'un assaillant d'exploiter une vulnérabilité de sécurité. Il existe quatre principaux types de contrôles de sécurité : préventifs, détectifs, réactifs et proactifs.

renforcement de la sécurité

Processus qui consiste à réduire la surface d'attaque pour la rendre plus résistante aux attaques. Cela peut inclure des actions telles que la suppression de ressources qui ne sont plus requises, la mise en œuvre des bonnes pratiques de sécurité consistant à accorder le moindre privilège ou la désactivation de fonctionnalités inutiles dans les fichiers de configuration.

système de gestion des informations et des événements de sécurité (SIEM)

Outils et services qui associent les systèmes de gestion des informations de sécurité (SIM) et de gestion des événements de sécurité (SEM). Un système SIEM collecte, surveille et analyse les données provenant de serveurs, de réseaux, d'appareils et d'autres sources afin de détecter les menaces et les failles de sécurité, mais aussi de générer des alertes.

automatisation des réponses de sécurité

Action prédéfinie et programmée conçue pour répondre automatiquement à un événement de sécurité ou y remédier. Ces automatisations servent de contrôles de sécurité détectifs ou réactifs

qui vous aident à mettre en œuvre les meilleures pratiques AWS de sécurité. Parmi les actions de réponse automatique, citons la modification d'un groupe de sécurité VPC, l'application de correctifs à une instance Amazon EC2 ou la rotation des informations d'identification.

chiffrement côté serveur

Chiffrement des données à destination, par celui AWS service qui les reçoit.

Politique de contrôle des services (SCP)

Politique qui propose un contrôle centralisé des autorisations pour tous les comptes d'une organisation dans AWS Organizations. Les SCP définissent des barrières de protection ou des limites aux actions qu'un administrateur peut déléguer à des utilisateurs ou à des rôles. Vous pouvez utiliser les SCP comme listes d'autorisation ou de refus, pour indiquer les services ou les actions autorisés ou interdits. Pour plus d'informations, consultez la section Politiques de contrôle des services dans la AWS Organizations documentation.

point de terminaison du service

URL du point d'entrée pour un AWS service. Pour vous connecter par programmation au service cible, vous pouvez utiliser un point de terminaison. Pour plus d'informations, veuillez consulter la rubrique AWS service endpoints dans Références générales AWS.

contrat de niveau de service (SLA)

Accord qui précise ce qu'une équipe informatique promet de fournir à ses clients, comme le temps de disponibilité et les performances des services.

indicateur de niveau de service (SLI)

Mesure d'un aspect des performances d'un service, tel que son taux d'erreur, sa disponibilité ou son débit.

objectif de niveau de service (SLO)

Mesure cible qui représente l'état d'un service, tel que mesuré par un indicateur de niveau de service.

modèle de responsabilité partagée

Un modèle décrivant la responsabilité que vous partagez en matière AWS de sécurité et de conformité dans le cloud. AWS est responsable de la sécurité du cloud, alors que vous êtes responsable de la sécurité dans le cloud. Pour de plus amples informations, veuillez consulter Modèle de responsabilité partagée.

SIEM

Consultez les informations de sécurité et le système de gestion des événements.

point de défaillance unique (SPOF)

Défaillance d'un seul composant critique d'une application susceptible de perturber le système.

SLA

Voir le contrat de niveau de service.

SLI

Voir l'indicateur de niveau de service.

SLO

Voir l'objectif de niveau de service.

split-and-seed modèle

Modèle permettant de mettre à l'échelle et d'accélérer les projets de modernisation. Au fur et à mesure que les nouvelles fonctionnalités et les nouvelles versions de produits sont définies, l'équipe principale se divise pour créer des équipes de produit. Cela permet de mettre à l'échelle les capacités et les services de votre organisation, d'améliorer la productivité des développeurs et de favoriser une innovation rapide. Pour plus d'informations, consultez la section Approche progressive de la modernisation des applications dans le. AWS Cloud

SPOF

Voir point de défaillance unique.

schéma en étoile

Structure organisationnelle de base de données qui utilise une grande table de faits pour stocker les données transactionnelles ou mesurées et utilise une ou plusieurs tables dimensionnelles plus petites pour stocker les attributs des données. Cette structure est conçue pour être utilisée dans un entrepôt de données ou à des fins de business intelligence.

modèle de figuier étrangleur

Approche de modernisation des systèmes monolithiques en réécrivant et en remplaçant progressivement les fonctionnalités du système jusqu'à ce que le système hérité puisse être mis hors service. Ce modèle utilise l'analogie d'un figuier de vigne qui se développe dans un arbre existant et qui finit par supplanter son hôte. Le schéma a été présenté par Martin Fowler comme

un moyen de gérer les risques lors de la réécriture de systèmes monolithiques. Pour obtenir un exemple d'application de ce modèle, veuillez consulter Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

sous-réseau

Plage d'adresses IP dans votre VPC. Un sous-réseau doit se trouver dans une seule zone de disponibilité.

contrôle de supervision et acquisition de données (SCADA)

Dans le secteur manufacturier, un système qui utilise du matériel et des logiciels pour surveiller les actifs physiques et les opérations de production.

chiffrement symétrique

Algorithme de chiffrement qui utilise la même clé pour chiffrer et déchiffrer les données.

tests synthétiques

Tester un système de manière à simuler les interactions des utilisateurs afin de détecter les problèmes potentiels ou de surveiller les performances. Vous pouvez utiliser Amazon CloudWatch Synthetics pour créer ces tests.

T

balises

Des paires clé-valeur qui agissent comme des métadonnées pour organiser vos AWS ressources. Les balises peuvent vous aider à gérer, identifier, organiser, rechercher et filtrer des ressources. Pour plus d'informations, veuillez consulter la rubrique Balisage de vos AWS ressources.

variable cible

La valeur que vous essayez de prédire dans le cadre du ML supervisé. Elle est également qualifiée de variable de résultat. Par exemple, dans un environnement de fabrication, la variable cible peut être un défaut du produit.

liste de tâches

Outil utilisé pour suivre les progrès dans un runbook. Liste de tâches qui contient une vue d'ensemble du runbook et une liste des tâches générales à effectuer. Pour chaque tâche générale, elle inclut le temps estimé nécessaire, le propriétaire et l'avancement.

environnement de test

Voir environnement.

entraînement

Pour fournir des données à partir desquelles votre modèle de ML peut apprendre. Les données d'entraînement doivent contenir la bonne réponse. L'algorithme d'apprentissage identifie des modèles dans les données d'entraînement, qui mettent en correspondance les attributs des données d'entrée avec la cible (la réponse que vous souhaitez prédire). Il fournit un modèle de ML qui capture ces modèles. Vous pouvez alors utiliser le modèle de ML pour obtenir des prédictions sur de nouvelles données pour lesquelles vous ne connaissez pas la cible.

passerelle de transit

Hub de transit de réseau que vous pouvez utiliser pour relier vos VPC et vos réseaux sur site. Pour plus d'informations, voir Qu'est-ce qu'une passerelle de transit dans la AWS Transit Gateway documentation.

flux de travail basé sur jonction

Approche selon laquelle les développeurs génèrent et testent des fonctionnalités localement dans une branche de fonctionnalités, puis fusionnent ces modifications dans la branche principale. La branche principale est ensuite intégrée aux environnements de développement, de préproduction et de production, de manière séquentielle.

accès sécurisé

Accorder des autorisations à un service que vous spécifiez pour effectuer des tâches au sein de votre organisation AWS Organizations et dans ses comptes en votre nom. Le service de confiance crée un rôle lié au service dans chaque compte, lorsque ce rôle est nécessaire, pour effectuer des tâches de gestion à votre place. Pour plus d'informations, consultez la section Utilisation AWS Organizations avec d'autres AWS services dans la AWS Organizations documentation.

réglage

Pour modifier certains aspects de votre processus d'entraînement afin d'améliorer la précision du modèle de ML. Par exemple, vous pouvez entraîner le modèle de ML en générant un ensemble d'étiquetage, en ajoutant des étiquettes, puis en répétant ces étapes plusieurs fois avec différents paramètres pour optimiser le modèle.

équipe de deux pizzas

Une petite DevOps équipe que vous pouvez nourrir avec deux pizzas. Une équipe de deux pizzas garantit les meilleures opportunités de collaboration possible dans le développement de logiciels.

U

incertitude

Un concept qui fait référence à des informations imprécises, incomplètes ou inconnues susceptibles de compromettre la fiabilité des modèles de ML prédictifs. Il existe deux types d'incertitude : l'incertitude épistémique est causée par des données limitées et incomplètes, alors que l'incertitude aléatoire est causée par le bruit et le caractère aléatoire inhérents aux données. Pour plus d'informations, veuillez consulter le guide Quantifying uncertainty in deep learning systems.

tâches indifférenciées

Également connu sous le nom de « levage de charges lourdes », ce travail est nécessaire pour créer et exploiter une application, mais qui n'apporte pas de valeur directe à l'utilisateur final ni d'avantage concurrentiel. Les exemples de tâches indifférenciées incluent l'approvisionnement, la maintenance et la planification des capacités.

environnements supérieurs

Voir environnement.

V

mise à vide

Opération de maintenance de base de données qui implique un nettoyage après des mises à jour incrémentielles afin de récupérer de l'espace de stockage et d'améliorer les performances.

contrôle de version

Processus et outils permettant de suivre les modifications, telles que les modifications apportées au code source dans un référentiel.

Appairage de VPC

Connexion entre deux VPC qui vous permet d'acheminer le trafic à l'aide d'adresses IP privées. Pour plus d'informations, veuillez consulter la rubrique Qu'est-ce que l'appairage de VPC ? dans la documentation Amazon VPC.

vulnérabilités

Défaut logiciel ou matériel qui compromet la sécurité du système.

W

cache actif

Cache tampon qui contient les données actuelles et pertinentes fréquemment consultées. L'instance de base de données peut lire à partir du cache tampon, ce qui est plus rapide que la lecture à partir de la mémoire principale ou du disque.

données chaudes

Données rarement consultées. Lorsque vous interrogez ce type de données, des requêtes modérément lentes sont généralement acceptables.

fonction de fenêtre

Fonction SQL qui effectue un calcul sur un groupe de lignes liées d'une manière ou d'une autre à l'enregistrement en cours. Les fonctions de fenêtre sont utiles pour traiter des tâches, telles que le calcul d'une moyenne mobile ou l'accès à la valeur des lignes en fonction de la position relative de la ligne en cours.

charge de travail

Ensemble de ressources et de code qui fournit une valeur métier, par exemple une application destinée au client ou un processus de backend.

flux de travail

Groupes fonctionnels d'un projet de migration chargés d'un ensemble de tâches spécifique. Chaque flux de travail est indépendant, mais prend en charge les autres flux de travail du projet. Par exemple, le flux de travail du portefeuille est chargé de prioriser les applications, de planifier les vagues et de collecter les métadonnées de migration. Le flux de travail du portefeuille fournit ces actifs au flux de travail de migration, qui migre ensuite les serveurs et les applications.

VER

Voir écrire une fois, lire plusieurs.

WQF

Consultez le cadre de qualification des charges de travail AWS.

écrire une fois, lire plusieurs (WORM)

Modèle de stockage qui écrit les données une seule fois et empêche leur suppression ou leur modification. Les utilisateurs autorisés peuvent lire les données autant de fois que nécessaire, mais ils ne peuvent pas les modifier. Cette infrastructure de stockage de données est considérée comme immuable.

Z

exploit Zero-Day

Une attaque, généralement un logiciel malveillant, qui tire parti d'une vulnérabilité de type « jour zéro ».

vulnérabilité de type « jour zéro »

Une faille ou une vulnérabilité non atténuée dans un système de production. Les acteurs malveillants peuvent utiliser ce type de vulnérabilité pour attaquer le système. Les développeurs prennent souvent conscience de la vulnérabilité à la suite de l'attaque.

application zombie

Application dont l'utilisation moyenne du processeur et de la mémoire est inférieure à 5 %. Dans un projet de migration, il est courant de retirer ces applications.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.