Data Warehouse Service

FAQs

Edição 01

Data 2025-11-14

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Marcas registadas e permissões

e outras marcas registadas da Huawei são marcas registadas da Huawei Technologies Co., Ltd. Todos as outras marcas registadas e os nomes registados mencionados neste documento são propriedade dos seus respectivos detentores.

Aviso

Os produtos, os serviços e as funcionalidades adquiridos são estipulados pelo contrato estabelecido entre a Huawei Cloud e o cliente. Os produtos, os serviços e as funcionalidades descritos neste documento, no todo ou em parte, podem não estar dentro do âmbito de aquisição ou do âmbito de uso. Salvo especificação em contrário no contrato, todas as declarações, informações e recomendações neste documento são fornecidas "TAL COMO ESTÃO" sem garantias ou representações de qualquer tipo, sejam expressas ou implícitas.

As informações contidas neste documento estão sujeitas a alterações sem aviso prévio. Foram feitos todos os esforços na preparação deste documento para assegurar a exatidão do conteúdo, mas todas as declarações, informações e recomendações contidas neste documento não constituem uma garantia de qualquer tipo, expressa ou implícita.

Índice

1 Perguntas frequentes principais

Você pode encontrar respostas para suas perguntas nesta página.

Utilização da sintaxe

Tabela 1-1 Utilização da sintaxe

Classificação	Pergunta frequente	URL da página
1	● Como consultar o número de bits em uma cadeia de consulta? ● Como truncar uma subcadeia? ● Como substituir algumas cadeias no resultado retornado? ● Como retornar os primeiros caracteres de uma cadeia? ● Como filtrar o cabeçalho e a cauda de uma cadeia? ● Como obtenho o número de bytes de uma cadeia especificada?	**Funções de processamento de caracteres e operadores**
2	● Como criar uma tabela de partição? ● Quais são os tipos de partição de tabela suportados?	**CREATE TABLE PARTITION**
3	● Como visualizar todos os esquemas? ● Como exibir todas as tabelas em um esquema?	**Esquema**

Classificação	Pergunta frequente	URL da página
4	● Como adicionar uma coluna de tabela? ● Como alterar um tipo de dados? ● Como adicionar uma restrição NOT NULL a uma coluna? ● Como definir a chave primária? ● Como alterar os atributos da tabela?	**ALTER TABLE**
5	● Como usar as funções de data? ● Como usar pg_sleep()? ● Como fazer a subtração do mês? ● Como utilizar funções para alterar os tipos de data?	**Funções de processamento de data e hora e operadores**
6	● Como alterar as colunas de distribuição? ● Como alterar a coluna de distribuição para outra coluna? ● Por que os dados na coluna de distribuição não podem ser atualizados e a mensagem "Distributed key column can't be updated in current version" é exibida?	**Como alterar colunas de distribuição**
7	● Como gerenciar partições? ● Como adicionar ou excluir uma partição? ● Como renomear uma partição? ● Como consultar dados em uma partição?	**Criação e gerenciamento de tablas particionadas**
8	● Como consultar o número de linhas em uma partição? ● Como mesclar duas partições?	**ALTER TABLE PARTITION**
9	Como chamar o procedimento armazenado?	**CALL**
10	● Como criar uma tabela? ● CREATE TABLE LIKE	**CREATE TABLE**

Classificação	Pergunta frequente	URL da página
11	• Qual é o comando para autorização? • Como usar a sintaxe da concessão? • Como conceder os direitos de um usuário a outros usuários? • Como conceder permissões de tabela a um usuário? • Como conceder permissões de banco de dados a um usuário? • Quais são as permissões de tabela estrangeira?	**GRANT**
12	• Como usar a função REPLACE? • Como usar to_timestamp? • Como cobrir um carimbo de data/hora para um formato especificado? • Como usar current_timestamp? • Como usar to_number?	**Funções de conversão de tipos**

Gerenciamento de banco de dados

Tabela 1-2 Gerenciamento de banco de dados

Classificação	Pergunta frequente	URL da página
1	• Como visualizar definições de tabela? • Como visualizar o DDL? • Como visualizar a estrutura das visualizações? • Como consultar o tamanho de um banco de dados ou uma tabela?	**Consulta da tabela e o tamanho do banco de dados**
2	• Como limpar os tablespaces? • Como usar a sintaxe VACUUM FULL? • Como melhorar o desempenho da consulta após um grande número de operações de adição, exclusão e modificação?	**VACUUM**

Classificação	Pergunta frequente	URL da página
3	● Como verificar se um usuário tem permissões na tabela atual? ● Como verificar se um usuário tem uma determinada permissão em uma tabela?	**Como visualizar as permissões da tabela de um usuário?**
4	● Como consultar e finalizar declarações bloqueadas? ● Como consultar declarações ativas? ● Como encerrar uma sessão? ● O que fazer ao bloquear o tempo de espera?	**Análise de instruções SQL que estão sendo executadas**
5	● Existe uma descrição detalhada dos planos de execução SQL? ● Como visualizar o plano de execução? ● Quais são as diferenças entre Nested Loop, Hash Join e Merge Join?	**Mergulho profundo no plano de execução de SQL**
6	● Como cancelar o status somente leitura? ● O que fazer quando o banco de dados entra no estado somente leitura?	**Um cluster de GaussDB(DWS) torna-se somente leitura e é bloqueado e os dados não podem ser gravados**
7	● Como coletar estatísticas?	**ANALYZE | ANALYSE**
8	● Como configurar o otimizador? ● Como ativar ou desativar nestloop? ● Como ativar ou desativar mergejoin? ● Quais são os parâmetros que afetam o plano de execução?	**Configuração do método do otimizador**
9	● Como alterar o fuso horário do banco de dados? ● Como alterar o fuso horário?	**Como alterar o fuso horário padrão de um banco de dados quando o horário do banco de dados é diferente do horário do sistema?**
10	● Como consultar definições de função? ● Como usar as funções de pesquisa de privilégios de acesso? ● Como consultar definições de exibição?	**Funções de informações de sistema**

Classificação	Pergunta frequente	URL da página
11	● Quais são as especificações técnicas? ● Quais são os tamanhos de tabela de partição suportados? ● Qual é o volume máximo de dados em uma única tabela? ● Qual é o número máximo de colunas suportadas por uma tabela?	**Especificações técnicas**
12	● Quais são as operações do desenvolvedor? ● Como especificar se habilitar o uso de um framework distribuído pelo otimizador?	**Opções do desenvolvedor**

Gerenciamento de cluster

Tabela 1-3 Gerenciamento de cluster

Classificação	Pergunta frequente	URL da página
1	● O que fazer quando o uso do disco é muito alto? ● O que fazer se o disco estiver cheio? ● O que fazer se o cluster se tornar somente leitura? ● O que fazer se um alarme de disco for gerado? ● Como definir o limite de disco? ● Como ativar a assinatura de alarme de disco?	**Um cluster de GaussDB(DWS) torna-se somente leitura e é bloqueado e os dados não podem ser gravados**
2	● Como exibir informações básicas do cluster? ● Como visualizar o EIP de um cluster? ● Como visualizar o endereço de conexão do banco de dados? ● Quais são os flavors de cluster? ● Qual é a informação básica do disco?	**Exibição de detalhes do cluster**

Classificação	Pergunta frequente	URL da página
3	Como comprar um cluster?	**Criação de um cluster**
4	● O que fazer se o cluster estiver desbalanceado? ● Como realizar a alternância de cluster primário/em espera?	**Realização de uma alternância de cluster primário/em espera**

2 Problemas gerais

2.1 Por que os armazéns de dados são necessários?

Status quo e requisitos

Muitos dados (pedidos, estoques, materiais e pagamentos) são gerados nos sistemas de operação de negócios e no banco de dados de fundo (transacional) das empresas.

Os tomadores de decisão categorizam e analisam os dados para a tomada de decisões de negócios.

Dificuldades

A categorização e análise de dados envolvem o acesso simultâneo aos dados em várias tabelas de banco de dados. Ou seja, várias tabelas sendo atualizadas por diferentes transações podem ser bloqueadas simultaneamente, o que pode causar complicações aos sistemas de banco de dados durante as horas de pico.

- Bloquear várias tabelas aumenta a latência de consultas complexas.
- As transações que estão atualizando as tabelas do banco de dados são bloqueadas, causando atrasos ou interrupções.

Solução

Os armazéns de dados se destacam em agregação e associação de dados, para que os usuários extraiam mais dados, obtenham mais informações e tomem melhores decisões. A mineração requer consultas complexas que envolvem dados em várias tabelas.

O processo ETL copia dados em bancos de dados de operações de negócios para armazéns de dados para análise e computação. Os dados podem ser agregados de vários sistemas operacionais de negócios em um armazém de dados para melhor associação, análise e percepções acionáveis.

Os armazéns de dados e bancos de dados orientados a transações padrão, como Oracle,SQL Server e MySQL, usam diferentes modos de design. Os armazéns de dados são otimizados em termos de agregação e associação de dados, mas a transação ou adição de dados e exclusão de funções ou desempenho podem não ser garantidos. Portanto, armazéns de dados e bancos de

dados se aplicam a diferentes cenários. Os bancos de dados transacionais são dedicados ao processamento de transações (operação comercial das empresas), enquanto os armazéns de dados se destacam na análise de dados complexa. Em conclusão, os bancos de dados se aplicam a atualizações de dados, enquanto os armazéns de dados se aplicam à análise de dados.

2.2 Por que devo usar GaussDB(DWS) da nuvem pública?

Armazéns de dados convencionais não são práticos para empresas menores devido ao alto custo, à demorada seleção e aquisição de dispositivos e sistemas e à expansão complexa.

GaussDB(DWS) na nuvem pública é a melhor escolha:

- Este serviço de armazenamento de dados MPP distribuído em nuvem é muito aberto, eficiente, compatível, escalável e fácil de O&M.
- Desenvolvido no kernel de armazém de dados FusionInsight LibrA, ele capacita as empresas de nuvem pública com uma experiência melhor e consistente dentro e fora da nuvem.

  FusionInsight LibrA é um sistema de armazenamento de dados distribuído de última geração com direitos de propriedade intelectual independentes. Atualmente, é amplamente utilizado em governo, finanças e operadoras. FusionInsight LibrA é compatível com bancos de dados Postgres de código aberto, especialmente em instruções SQL de Oracle e Teradata. Os engenheiros do GaussDB(DWS) projetaram um núcleo de armazenamentos híbridos de linhas e colunas não apenas para análise mais rápida, mas também para processamento de dados, como adicionar, excluir e modificar dados. FusionInsight LibrA possui o otimizador de custo e tecnologias de armazém, incluindo computação vetorial de código de máquina e inter/intra-paralelismo para operadores e nós. Ele usa o LLVM para otimizar o código local em planos de consulta de compilação. A consulta e análise de dados mais poderosas abordam os pontos problemáticos do serviço e melhoram a experiência do usuário.
- GaussDB(DWS) pode ser usado imediatamente.

  A aplicação do GaussDB (DWS) na nuvem pública leva apenas alguns minutos, liberando você do processo demorado de pesquisa e compra de armazéns de dados. Isso não apenas simplifica a aquisição, mas também reduz o custo e os requisitos para o uso de armazéns de dados. Empresas de pequeno e médio porte com acesso ao GaussDB(DWS) podem facilmente extrair valores de dados para seu desenvolvimento e formar insights acionáveis.

2.3 Como escolher o GaussDB (DWS) ou o RDS na nuvem pública?

Ambos permitem executar bancos de dados relacionais convencionais na nuvem e transferir cargas de gerenciamento de banco de dados. Os bancos de dados RDS são úteis para OLTP, relatórios e análises, mas são menos capazes de lidar com operações de leitura de uma grande quantidade de dados (consultas complexas somente leitura). O GaussDB (DWS) é útil para OLAP, reduzindo as cargas de trabalho de análise e relatório de grandes conjuntos de dados em uma ordem de magnitude, graças à sua escala e recursos de vários nós e algoritmos otimizados (**armazenamento em coluna, executores vetorizados, e estruturas distribuídas**).

Você pode escalar um cluster do GaussDB (DWS) para lidar com dados e consultas complexas, ou para lidar com análises esmagadoras e relatar cargas de trabalho que afetam o desempenho do OLTP.

A tabela a seguir mostra a comparação entre OLTP e OLAP.

Tabela 2-1 Comparação de recursos entre OLTP e OLAP

Característica	RDS for OLTP	GaussDB(DWS) for OLAP
Usuário	Operações e gerenciamento de baixo nível	Tomadores de decisão e alta administração
Função	Processamento diário da operação	Análise e tomada de decisão
Projetar	Por aplicação	Por tema
Dados	Últimos, detalhados, bidimensionais, discretas	Históricos, integrados, multidimensionais, unificados
Acesso	Dezenas de registros de leitura e gravação	Milhões de registros lidos
Cobertura	Operações simples de leitura/escrita	Consultas complexas
Tamanho do banco de dados	Centenas de GB	TB ou PB

2.4 O que é a cota de usuário?

Para HUAWEI CLOUDserviços, as cotas limitam o número de recursos disponíveis para os usuários. Se precisar de mais, envie um tíquete de serviço para aumentar suas cotas. Uma vez aprovado, atualizaremos sua cota de recursos de acordo e enviaremos uma notificação. Para obter detalhes sobre operações de cotas, consulte **Cotas**.

2.5 Quais são as diferenças entre usuários e funções?

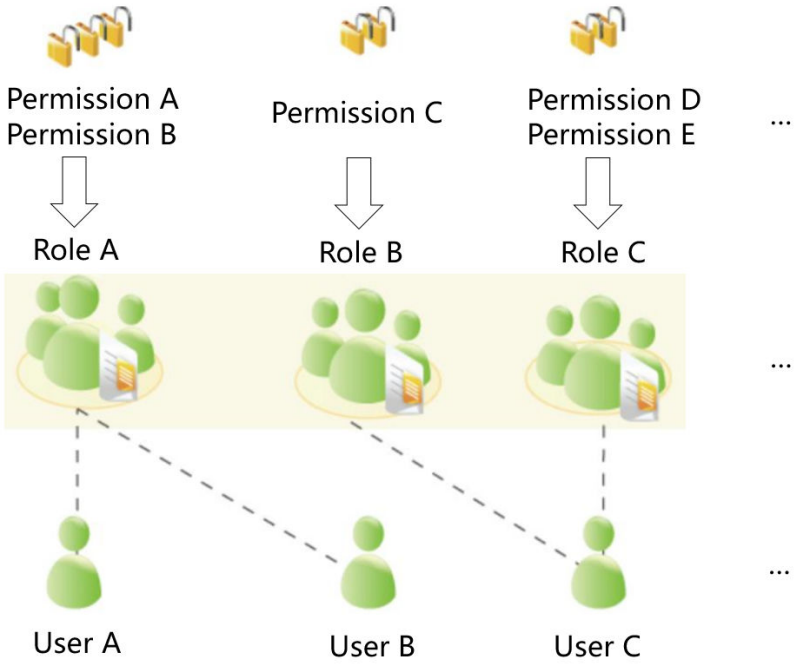
Os usuários e as funções são compartilhados dentro de todo o cluster, mas seus dados não são compartilhados. Ou seja, um usuário pode se conectar a qualquer banco de dados, mas após a conexão ser bem-sucedida, qualquer usuário pode acessar apenas o banco de dados declarado na solicitação de conexão.

- Uma função é um conjunto de permissões. Geralmente, as funções são usadas para classificar permissões. Os usuários são usados para gerenciar permissões e executar operações.
- Uma função pode herdar permissões de outras funções. Todos os usuários em um grupo de usuários herdam automaticamente as permissões de operação da função do grupo.
- Em um banco de dados, as permissões dos usuários vêm de funções.
- Um grupo de usuários é um grupo de usuários que têm a mesma permissão.

- Um usuário pode ser considerado como uma função com a permissão de login.
- Uma função pode ser considerada como um usuário sem a permissão de login.

As permissões fornecidas pelo Gauss (DWS) incluem as permissões de O&M para componentes no plano de gerenciamento. Você pode atribuir permissões diferentes aos usuários conforme necessário. O plano de gerenciamento usa funções para melhor gerenciamento de permissões. Você pode selecionar permissões especificadas e atribuí-las a funções de maneira unificada. Desta forma, as permissões podem ser visualizadas e gerenciadas de forma centralizada.

A figura a seguir mostra as relações entre permissões, funções e usuários no gerenciamento unificado de permissões.



O GaussDB(DWS) fornece várias permissões. Selecione e atribua permissões a usuários diferentes com base em cenários de serviço. Uma função pode ser atribuída a uma ou mais permissões.

Depois que uma função for concedida a um usuário por meio de **GRANT**, o usuário terá todas as permissões da função. Recomenda-se que as funções sejam usadas para conceder permissões de forma eficiente. Um usuário tem permissões apenas para suas próprias tabelas, mas não tem permissões para tabelas de outros usuários em seus esquemas.

- A função A recebe as permissões de operação A e B. Depois que a função A é alocada ao usuário A e ao usuário B, o usuário A e o usuário B podem obter permissões de operação A e B.
- A função B é atribuída a permissão de operação C. Depois que a função B é alocada ao usuário C, o usuário C pode obter permissões de operação C.
- A função C recebe as permissões de operação D e E. Depois que a função C é alocada ao usuário C, o usuário C pode obter as permissões de operação D e F.

2.6 Quando usar o GaussDB (DWS) e o MRS?

O MRS funciona melhor com estruturas de processamento de Big data como Apache Spark, Hadoop e HBase, para processar e analisar conjuntos de dados ultra-grandes por meio de código personalizado. Ele permite que você controle configurações de cluster e software instalado no cluster.

O GaussDB (DWS) funciona melhor com consultas complexas de uma grande quantidade de dados estruturados. O objetivo é reunir dados de diferentes fontes, como inventário, finanças e sistema de varejo. Para garantir a consistência e a precisão dos relatórios corporativos, o GaussDB (DWS) armazena dados de maneira altamente estruturada. Essa estrutura pode criar diretamente a regra de consistência de dados na tabela do banco de dados. Além disso, o GaussDB (DWS) é altamente compatível com instruções SQL padrão e com a sintaxe de bancos de dados convencionais suportados por transações.

O GaussDB(DWS) é preferido quando se deseja realizar consultas complexas de uma grande quantidade de dados estruturados com alto desempenho.

2.7 Como verificar o tempo de criação de um usuário de banco de dados?

Método 1:

Quando você cria um usuário de banco de dados GaussDB(DWS), se a hora em que o usuário entra em vigor (**VALID BEGIN**) é a mesma que a hora de criação do usuário, e a hora em que o usuário entra em vigor não foi alterada, você pode verificar a coluna **valbegin** na visão **PG_USER** para verificar o horário de criação do usuário.

O seguinte é um exemplo:

Crie o usuário **jerry** e defina sua hora de início de validade para sua hora de criação atual.

```
CREATE USER jerry PASSWORD 'password' VALID BEGIN '2022-05-19 10:31:56';
```

Exiba usuários na visão **PG_USER**. A coluna **valbegin** indica a hora em que **jerry** entrou em vigor, ou seja, a hora em que o jerry foi criado.

```
SELECT * FROM PG_USER;
 usename | usesysid | usecreatedb | usesuper | usecatupd | userepl |  passwd
|        valbegin        | valuntil |   respool    | parent | spacelimit |
useconfig | nodegroup | tempspacelimit |
 spillspacelimit
---------+----------+-------------+----------+-----------+---------+----------
+-------------------------+----------+--------------+--------+------------
+-----------+-----------+----------------+
-----------------
 Ruby    |       10 | t           | t        | t         | t       | ********
|                         |          | default_pool |      0 |
|           |           |                |

 dbadmin |    16393 | f           | f        | f         | f       | ********
|                         |          | default_pool |      0 |
|           |           |                |

 jack    |   451897 | f           | f        | f         | f       | ********
|                         |          | default_pool |      0 |
|           |           |                |

 emma    |   451910 | f           | f        | f         | f       | ********
```



```
|                          |           | default_pool |      0 |
|           |           |                |

 jerry    |   457386 | f           | f         | f          | f        | ******** |
2022-05-19 10:31:56+08 |           | default_pool |      0 |
|           |           |                |
(5 rows)
```

Método 2:

Verifique a coluna **passwordtime** no catálogo do sistema **PG_AUTH_HISTORY**. Essa coluna indica a hora em que a senha inicial do usuário foi criada. Somente usuários com permissões de administrador do sistema podem acessar o catálogo.

```
select roloid, min(passwordtime) as create_time from pg_auth_history group by
roloid order by roloid;
```

O seguinte é um exemplo:

Consulte a visão **PG_USER** para obter o OID do usuário **jerry**, que é **457386**. Consulte a coluna **passwordtime** para obter o horário de criação do usuário **jerry**, que é **2022-05-19 10:31:56**.

```
select roloid, min(passwordtime) as create_time from pg_auth_history group by
roloid order by roloid;
 roloid |          create_time
--------+-------------------------------
     10 | 2022-02-25 09:53:38.711785+08
  16393 | 2022-02-25 09:55:17.992932+08
 451897 | 2022-05-18 09:42:26.897855+08
 451910 | 2022-05-18 09:46:33.152354+08
 457386 | 2022-05-19 10:31:56.037706+08
(5 rows)
```

2.8 Regiões e AZs

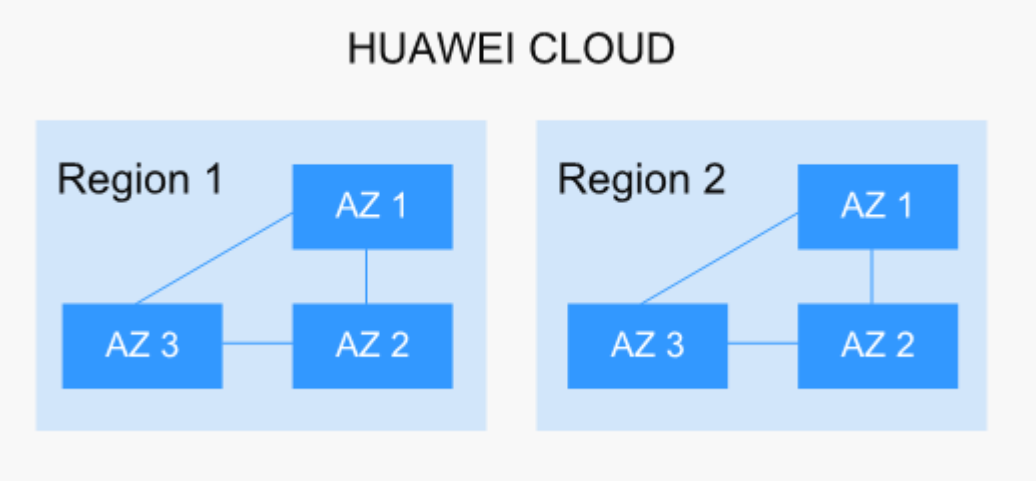
Conceitos

Uma região e uma zona de disponibilidade (AZ) identificam a localização de um data center. Você pode criar recursos em regiões e AZs.

- **Regiões** são definidas em termos de localização geográfica e latência da rede. Cada região tem seus próprios serviços públicos compartilhados (ECS, EVS, OBS, VPC, EIP e IMS). As regiões são comuns ou dedicadas. Uma região comum fornece serviços de nuvem comuns disponíveis para todos os locatários. Uma região dedicada fornece serviços de um tipo específico ou apenas para locatários específicos.
- Uma **AZ** contém um ou mais data centers físicos. Cada AZ tem instalações independentes de resfriamento, extinção de incêndio, antiumidade e eletricidade. A computação, rede, armazenamento e outros recursos em uma AZ são logicamente divididos em vários clusters. As AZs em uma região são interconectadas por meio de fibra óptica de alta velocidade, para que os sistemas implementados nas AZs possam alcançar maior disponibilidade.

Figura 2-1 mostra a relação entre as regiões e as AZs.

Figura 2-1 Regiões e AZs



A HUAWEI CLOUD fornece serviços em todo o mundo. Você pode selecionar uma região e AZ conforme necessário. Para obter mais informações, consulte **Produtos globais e serviços**.

Como selecionar uma região?

Ao selecionar uma região, considere o seguinte:

- Localização

  Selecione uma região mais próxima de seus usuários-alvo para menor latência de rede e acesso rápido. As regiões da China continental fornecem a mesma infraestrutura, qualidade de rede BGP e operações e configurações de recursos. Portanto, se seus usuários-alvo estiverem na China continental, não será necessário considerar as diferenças de latência de rede ao selecionar uma região.

  Países e regiões fora da China continental, como Bangkok e Hong Kong, fornecem serviços para usuários fora da China continental. Se você ou seus usuários-alvo estiverem na China continental, essas regiões não são recomendadas devido à alta latência de acesso.

  – Se você ou seus usuários-alvo estiverem na região Ásia Asiático (excluindo a China continental), selecione a região **AP-Bangkok** ou **AP-Singapore**.
  – Se você ou seus usuários de destino estiverem na África, selecione a região **AF-Johannesburg**.
  – Se você ou seus usuários-alvo estiverem na Europa, selecione a região **EU-Paris**.

- Preço dos recursos

  Isso varia de acordo com a região. Para obter detalhes, consulte **Detalhes de preços**.

Como escolher uma AZ?

Considere seus requisitos de DR e latência de rede ao selecionar uma AZ:

- Implemente recursos em diferentes AZs na mesma região para fins de DR.
- Implemente recursos na mesma AZ para uma latência mínima.

Regiões e pontos de extremidade

Ao usar recursos com chamadas de API, você deve especificar o ponto de extremidade regional. Para obter detalhes sobre regiões e pontos de extremidade da Huawei Cloud, consulte "Regions and Endpoints".

2.9 Meus dados estão seguros no GaussDB(DWS)?

Sim. Na era do Big Data, os dados tornaram-se um ativo fundamental. A nuvem pública aderirá ao compromisso assumido ao longo dos anos de não tocar em suas aplicações ou dados, ajudando você a proteger seus ativos principais. Este é o nosso compromisso com os usuários e a sociedade, estabelecendo as bases para o sucesso comercial da nuvem pública e seus parceiros.

GaussDB(DWS) é um sistema de armazenamento de dados com segurança de classe de telecomunicações para proteger seus dados e privacidade. Além disso, a nuvem pública do GaussDB(DWS) oferece qualidade de classe de operadora, que pode satisfazer os requisitos de segurança e privacidade de dados de governos, organizações financeiras e operadoras. Portanto, é amplamente utilizado por várias indústrias. GaussDB(DWS) da nuvem pública ganhou a seguinte autenticação de segurança:

- Laboratório interno de segurança cibernética (ICSL) em conformidade com os padrões de segurança cibernética emitidos pelas autoridades do Reino Unido.
- Certificação de avaliação de privacidade e segurança (PSA) para atender aos requisitos da UE de segurança e privacidade de dados.

Segurança de dados de serviço

O GaussDB(DWS) é construído na infraestrutura de software da nuvem pública, incluindo ECS e OBS. Tanto o ECS quanto o OBS ganharam a Certificação de nuvem confiável emitida pela China Data Center Alliance em 2017.

Os dados de serviço dos usuários do GaussDB(DWS) são armazenados nos ECSs do cluster. Nem os usuários nem os administradores de O&M da nuvem pública podem fazer logon nos ECSs.

O sistema operacional dos ECSs é reforçado para segurança, incluindo endurecimento do kernel, instalação do patch mais recente, controle de permissão, gerenciamento de portas e anti-ataque de protocolo e porta.

O GaussDB(DWS) fornece medidas de segurança completas, como políticas de senha, autenticação, gerenciamento de sessão, gerenciamento de permissões de usuário e auditoria de banco de dados.

Segurança de dados de snapshot

Os backups do GaussDB(DWS) são snapshots armazenados no OBS. O OBS passou pela autenticação de segurança de nuvem confiável da China Data Center Alliance. O OBS suporta controle de permissão de acesso, acesso a chaves e recursos de criptografia de dados. Os dados de snapshot do GaussDB(DWS) podem ser usados apenas para backup e restauração de dados e não podem ser acessados por nenhum usuário. Os administradores do GaussDB(DWS) podem visualizar o espaço do OBS ocupado pelos dados de snapshots no console do GaussDB(DWS) e nas contas de nuvem pública.

Segurança de acesso à rede

O GaussDB(DWS) é totalmente isolado entre as redes de camada 2 e camada 3 para atender aos requisitos de segurança dos usuários governamentais e financeiros.

- O GaussDB(DWS) é implementado no ambiente do ECS dedicado ao locatário, que não é compartilhado com outros locatários. Portanto, o vazamento de dados devido ao compartilhamento de recursos de computação é impossível fisicamente.
- Os ECSs em um cluster do GaussDB(DWS) são isolados por meio de VPCs, impedindo que os ECSs sejam descobertos e invadidos por outros locatários.
- A rede é dividida em plano de serviço e plano de gerenciamento. Os dois aviões estão fisicamente isolados, garantindo a segurança da rede.
- Os locatários podem personalizar de forma flexível o grupo de segurança e as regras de acesso.
- Software de aplicação externa acessa GaussDB(DWS) por SSL.
- Os dados importados do OBS são criptografados.

2.10 Como o GaussDB (DWS) é protegido?

O GaussDB (DWS) usa o IAM e a VPC para controlar o acesso do usuário e isolar a rede do cluster. O acesso ao cluster é feito por SSL e conjunto de cifras. Além disso, o GaussDB (DWS) suporta autenticação de certificado digital bidirecional.

Os SOs de nó em cada cluster são reforçados para permitir acesso válido apenas aos arquivos do SO.

2.11 Can I Modify the Security Group of a GaussDB(DWS) Cluster?

After a GaussDB(DWS) cluster is created, you can change the security group. You can also add, delete, or modify security group rules in the current security group.

- Change the security group to another one.
  a. Log in to the GaussDB(DWS) management console.
  b. In the navigation pane on the left, choose Cluster > Dedicated Cluster.
  c. In the cluster list, find the target cluster and click the cluster name. The **Basic Information** page is displayed.
  d. On the cluster details page, locate the Security Group parameter, click Modify on the right of the security group name, and select the name of the security group to be changed.
  e. Click OK. The security group is modified.
- Modifying an existing security group rule:
  a. Log in to the GaussDB(DWS) management console.
  b. In the navigation pane on the left, choose Cluster > Dedicated Cluster.
  c. In the cluster list, find the target cluster and click the cluster name. The **Basic Information** page is displayed.

d. Locate the **Security Group** parameter and click the security group name to switch to the **Security Groups** page on the VPC console, on which you can set the security group.

2.12 O que é um banco de dados/armazém de dados/data lake/lakehouse?

A Internet e a IoT em evolução produzem enormes volumes de dados. Esses dados precisam ser gerenciados, usando conceitos como banco de dados, armazém de dados, data lake e lakehouse. Quais são esses conceitos? Quais são as relações deles? Quais são os produtos e soluções específicos? Este documento ajuda você a compreendê-los por meio de comparação.

Banco de dados

Um banco de dados é onde os dados são organizados, armazenados e gerenciados pela estrutura de dados.

Bancos de dados têm sido usados em computadores desde a década de 1960, com os dois modelos de dados predominantes (hierárquico e de rede), e dados e aplicações eram muito interdependentes. Este limitou as aplicações de banco de dados.

Um banco de dados geralmente se refere a um banco de dados relacional. Um banco de dados relacional organiza dados com um modelo relacional, ou seja, os dados são armazenados em linhas e colunas. Portanto, os dados do banco de dados são bem estruturados e independentes, com baixa redundância. Em 1970, os bancos de dados relacionais nasceram para separar completamente os dados das aplicações de software e se tornaram uma parte indispensável dos sistemas de computadores convencionais. Bancos de dados relacionais são a base de produtos de banco de dados de todos os fornecedores, com suporte a API relacional, mesmo que um banco de dados não seja relacional.

Bancos de dados relacionais processam transações básicas e rotineiras usando OLTP, como transações bancárias.

Armazém de dados

O crescimento do banco de dados facilitou o crescimento dos dados. OLAP explora a relação entre dados e minera mais valor de dados. No entanto, é difícil compartilhar dados entre diferentes bancos de dados, e a integração e análise de dados também enfrentam grandes desafios.

Para superar esses desafios para as empresas, Bill Inmon, propôs a ideia de data warehousing em 1990. O armazém de dados é executado em uma arquitetura de armazenamento exclusiva para executar OLAP em uma grande quantidade de dados OLTP acumulados ao longo dos anos. Desta forma, as empresas podem obter informações valiosas a partir de dados massivos de forma rápida e eficaz para tomar decisões informadas. Graças aos armazéns de dados, a indústria da informação evoluiu de sistemas operacionais baseados em bancos de dados relacionais para sistemas de suporte à decisão.

Ao contrário de um banco de dados, um armazém de dados tem os seguintes recursos:

- Um armazém de dados usa temas. Ele é construído para suportar vários serviços, com dados provenientes de dados operacionais dispersos. Portanto, os dados necessários precisam ser extraídos de múltiplas fontes de dados heterogêneas, processados e integrados, e reorganizados por tema.

- Um armazém de dados oferece suporte principalmente à análise de decisões corporativas e as operações envolvidas são focadas na consulta de dados. Portanto, ele melhora a velocidade de consulta e reduz o custo total de propriedade (TCO) otimizando estruturas de tabela e modos de armazenamento.

Tabela 2-2 Comparação entre armazéns de dados e bancos de dados

Dimensão	Armazém de dados	Banco de dados
Cenário de aplicação	OLAP	OLTP
Fonte de dados	Múltiplas	Única
Normalização de dados	Esquemas desnormalizados	Esquemas estáticos altamente normalizados
Acesso a dados	Operações de leitura otimizadas	Operações de gravação otimizadas

Data lake

Os dados são um ativo importante para as empresas. Os dados de produção e operações são salvos e destilados em políticas de gerenciamento eficazes.

O data lake faz isso. É um grande armazém de dados que armazena centralmente dados estruturados e não estruturados. Ele pode armazenar dados brutos de várias fontes e tipos de dados, o que significa que os dados podem ser acessados, processados, analisados e transmitidos sem serem estruturados primeiro. O data lake ajuda as empresas a concluir rapidamente a análise federada de fontes de dados heterogêneas e explorar o valor dos dados.

Um data lake é, em essência, uma solução que consiste em uma arquitetura de armazenamento de dados e ferramentas de processamento de dados.

- A **arquitetura de armazenamento** deve ser escalável e confiável o suficiente para armazenar dados massivos de qualquer tipo (dados estruturados, semi-estruturados, não estruturados).
- Os dois tipos de **ferramentas de processamento** têm funções separadas:
  - O primeiro tipo: migra dados para o lago, incluindo definição de fontes, formulação de políticas de sincronização, movimentação de dados e compilação de catálogos.
  - O segundo tipo então usa esses dados, incluindo análise, mineração e uso. O data lake deve estar equipado com recursos abrangentes, como gerenciamento abrangente de dados e ciclo de vida de dados, análise de dados diversificados e aquisição e liberação segura de dados. Essas ferramentas de governança de dados ajudam a garantir a qualidade dos dados, que pode ser comprometida pela falta de metadados e transformar o data lake em um pântano de dados.

Agora, com Big Data e IA, o data lake é ainda mais valioso e desempenha novos papéis. Representa mais capacidades empresariais. Por exemplo, o data lake pode centralizar o gerenciamento de dados, ajudando as empresas a criar modelos de operação mais otimizados. Ele também fornece outros recursos empresariais, como análise de previsão e modelos de recomendação. Esses modelos podem estimular um maior crescimento.

Assim como qualquer outro armazém e lago, um armazena bens, ou dados, de uma fonte, enquanto o outro armazena água, ou dados, de muitas fontes.

Tabela 2-3 Comparação entre data lakes e armazém de dados

Dimensão	Data lake	Armazém de dados
Cenário de aplicação	Análise exploratória (aprendizado de máquina, descoberta de dados, criação de perfis, previsão)	Análise de dados (com base em dados estruturados históricos)
Custo	Baixo custo inicial, alto custo subsequente	Alto custo inicial, baixo custo subsequente
Qualidade dos dados	Dados brutos maciços a serem limpos e normalizados antes do uso	Dados de alta qualidade que podem ser usados como base de fatos
Usuário-alvo	Cientistas de dados e desenvolvedores de dados	Analistas de negócios

Lakehouse

Embora os cenários e arquiteturas de aplicações de um armazém de dados e um data lake sejam diferentes, eles podem cooperar para resolver problemas. Um armazém de dados armazena dados estruturados e é ideal para suporte rápido de BI e tomada de decisões, enquanto um data lake armazena dados em qualquer formato e pode gerar maior valor ao minerar dados. Portanto, sua convergência pode trazer mais benefícios para as empresas em alguns cenários.

Um lakehouse, a convergência de um armazém de dados e um data lake, visa permitir a mobilidade de dados e agilizar a construção. A chave da arquitetura do lakehouse é permitir o fluxo livre de dados/metadados entre o armazém de dados e o data lake. Os dados de valor explícito no lago podem fluir ou até mesmo ser usados diretamente pelo armazém. Os dados de valor implícito no armazém também podem fluir para o lago para armazenamento de longo prazo a baixo custo e para a mineração de dados futura.

Solução inteligente de dados

DataArts Studio é uma plataforma de capacitação de dados que ajuda grandes agências governamentais e empresas a personalizar soluções inteligentes de gerenciamento de recursos de dados. Essa solução pode importar dados de todos os domínios para o data lake, eliminando silos de dados, liberando o valor dos dados e capacitando a transformação digital orientada por dados.

DataArts Studio apresenta o data lake inteligente FusionInsight como seu núcleo. Em torno dele estão os mecanismos de computação, como o banco de dados, o armazém de dados, o data lake e a plataforma de dados. Ele fornece capacitação de dados abrangente, abrangendo coleta de dados, agregação, computação, gerenciamento de ativos e abertura de dados.

Os mecanismos de lago, armazém e banco de dados permitem a construção ágil de data lake, migração rápida de bancos de dados GaussDB e análise em tempo real do armazém de dados. Para mais informações, acesse:

- Banco de dados

  - Os bancos de dados relacionais incluem: **Relational Database Service (RDS)** , **GaussDB(for MySQL)**, **GaussDB** , **RDS for PostgreSQL** .
  - Banco de dados não relacional: **Document Database Service (DDS)**, GaussDB NoSQL (includindo **Influx**, **Redis**, **Cassandra**)
- Armazém de dados: **DWS**
- Integração com data lake e armazém de dados: **MapReduce Service (MRS)**, **Data Lake Insight (DLI)** .
- Centro de governança de dados: **DataArts Studio**.

2.13 How Are Dirty Pages Generated in GaussDB(DWS)?

Causes

By using the versioning concurrency control (MVCC) mechanism, GaussDB(DWS) can achieve consistency and concurrency for multiple transactions that access the database simultaneously. This mechanism has the benefit of avoiding read-write conflicts, but the drawback of causing disk bloat and dirty pages.

The scenarios are as follows:

- When the DELETE operation is performed on a table, data is logically deleted but not physically deleted from the disk.
- When the UPDATE operation is performed on a table, GaussDB(DWS) logically marks the data to be updated as delete and inserts new data.

For the DELETE and UPDATE operations in a table, the data marked as deleted is called discarded tuples. The proportion of discarded tuples in the entire table is the dirty page rate. Therefore, when the dirty page rate of a table is high, the proportion of data marked as deleted in the table is high.

Solution:

GaussDB(DWS) provides a system view for querying the dirty page rate. For details, see section **PGXC_STAT_TABLE_DIRTY**.

To solve the problem of disk space bloat caused by high dirty page rate, GaussDB(DWS) provides the VACUUM function to clear the data logically marked as deleted. For details, see **VACUUM**.

VACUUM does not release the allocated space. To completely reclaim the cleared space, run **VACUUM FULL**.

NOTA

- **VACUUM FULL** clears and releases the space of deleted data, improving database performance and efficiency. However, running **VACUUM FULL** consumes more time and resources, and may cause some tables to be locked. Therefore, run **VACUUM FULL** only when the database load is light.
- To reduce the impact of disk bloat on database performance, you are advised to do **VACUUM FULL** on non-system catalogs whose dirty page rate exceeds 80%. You can determine whether to do **VACUUM FULL** based on service scenarios.

3 Uso do banco de dados

3.1 Como alterar as colunas de distribuição?

Em um banco de dados de armazém de dados, você precisa escolher cuidadosamente colunas de distribuição para tabelas grandes, pois elas podem afetar o desempenho do banco de dados e da consulta. Se uma chave de distribuição imprópria for usada, a distorção de dados poderá ocorrer após a importação dos dados. Como resultado, o uso de alguns discos será muito maior do que o de outros discos, e o cluster pode até se tornar somente leitura. Se a política de distribuição de hash for usada e ocorrer distorção de dados, o desempenho de I/O de alguns DNs será ruim, afetando o desempenho geral da consulta. A seleção adequada e o ajuste das colunas de distribuição são essenciais para o desempenho da consulta de tabela.

Se a política de distribuição de hash for usada, você precisará verificar as tabelas para garantir que seus dados sejam distribuídos uniformemente em cada DN. Geralmente, mais de 5% de diferença entre a quantidade de dados em diferentes DNs é considerada como distorção de dados. Se a diferença for superior a 10%, terá de escolher outra coluna de distribuição.

Para tabelas que não são distribuídas uniformemente, ajuste suas colunas de distribuição para reduzir a distorção de dados e evitar problemas de desempenho do banco de dados.

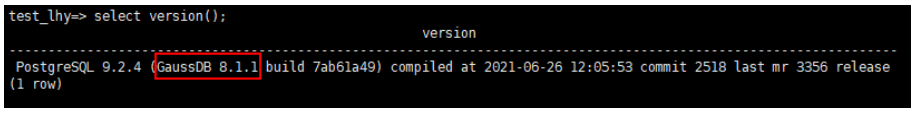
Escolher uma coluna de distribuição apropriada

A coluna de distribuição em uma tabela de hash deve atender aos seguintes requisitos, que são classificados por prioridade em ordem decrescente:

- Os valores da chave de distribuição devem ser discretos para que os dados possam ser distribuídos uniformemente em cada DN. Você pode selecionar a chave primária da tabela como a chave de distribuição. Por exemplo, para uma tabela de informações da pessoa, escolha a coluna número do cartão de identificação como a chave de distribuição.
- Não selecione a coluna onde existe um filtro constante.
- Selecione a condição de junção como a coluna de distribuição, para que as tarefas de junção possam ser enviadas para os DNs para serem executadas, reduzindo a quantidade de dados transferidos entre os DNs.
- Várias colunas de distribuição podem ser selecionadas para distribuir uniformemente os dados.

Procedimento

Execute a instrução **select version();** para consultar a versão atual do banco de dados. O desempenho necessário varia de acordo com a versão.



- **Para 8.0.x e versões anteriores, especifique a coluna de distribuição ao recriar uma tabela.**

Passo 1 Use Data Studio ou gsql no Linux para acessar o banco de dados.

Passo 2 Crie uma tabela.

NOTA

Nas instruções a seguir, **table1** é o nome da tabela original e **table1_new** é o nome da nova tabela. **column1** e **column2** são nomes de colunas de distribuição.

```
CREATE TABLE IF NOT EXISTS table1_new
( LIKE table1 INCLUDING ALL EXCLUDING DISTRIBUTION)
DISTRIBUTE BY
HASH (column1, column2);
```

Passo 3 Migre dados para a nova tabela.

```
START TRANSACTION;
LOCK TABLE table1 IN ACCESS EXCLUSIVE MODE;
INSERT INTO table1_new SELECT * FROM table1;
COMMIT;
```

Passo 4 Verifique se os dados da tabela foram migrados. Exclua a tabela original.

```
SELECT COUNT(*) FROM table1_new;
DROP TABLE table1;
```

Passo 5 Substitua a tabela original.

```
ALTER TABLE table1_new RENAME TO table1;
```

----Fim

3.2 Como exibir e definir a codificação de caracteres do banco de dados?

Exibir a codificação de caracteres do banco de dados

Use o parâmetro **server_encoding** para verificar a codificação do conjunto de caracteres do banco de dados atual. Por exemplo, a codificação de caracteres do banco de dados **music** é UTF8.

```
music=> SHOW server_encoding;
 server_encoding
-----------------
 UTF8
(1 row)
```


Definir a codificação de caracteres do banco de dados

NOTA

GaussDB(DWS) não suporta a modificação do formato de codificação de caracteres de um banco de dados criado.

Se você precisar especificar o formato de codificação de caracteres de um banco de dados, use **template0** e a sintaxe **CREATE DATABASE** para criar um banco de dados. Para tornar seu banco de dados compatível com a maioria dos caracteres, é aconselhável usar a codificação UTF8 ao criar um banco de dados.

Sintaxe CREATE DATABASE

```
CREATE DATABASE database_name
    [ [ WITH ] { [ OWNER [=] user_name ] |
                 [ TEMPLATE [=] template ] |
                 [ ENCODING [=] encoding ] |
                 [ LC_COLLATE [=] lc_collate ] |
                 [ LC_CTYPE [=] lc_ctype ] |
                 [ DBCOMPATIBILITY [=] compatibility_type ] |
                 [ CONNECTION LIMIT [=] connlimit ]}[...] ];
```

- **TEMPLATE [=] template**

  Indica o nome do modelo, ou seja, o nome do modelo a ser usado para criar o banco de dados. GaussDB(DWS) cria um banco de dados copiando um modelo de banco de dados. GaussDB(DWS) tem dois bancos de dados de modelos iniciais **template0** e **template1** e um banco de dados de usuário padrão **gaussdb**.

  Intervalo de valores: um nome de banco de dados existente. Se isso não for especificado, o sistema copia **template1** por padrão. Seu valor não pode ser **gaussdb**.

  AVISO

  Atualmente, os modelos de banco de dados não podem conter sequências. Se houver sequências na biblioteca de modelos, a criação do banco de dados falhará.

- **ENCODING [=] encoding**

  Codificação de caracteres usada pelo banco de dados. O valor pode ser uma cadeia de caracteres (por exemplo, **SQL_ASCII'**) ou um número inteiro.

  Se este parâmetro não for especificado, a codificação do banco de dados de modelo é usada por padrão. A codificação dos bancos de dados de modelo **template0** e **template1** depende do sistema operacional por padrão. A codificação de caracteres de **template1** não pode ser alterada. Para alterar a codificação, use **template0** para criar um banco de dados.

  Intervalo de valores: **GBK**, **UTF8** e **Latin1**

  AVISO

  A codificação do conjunto de caracteres do novo banco de dados deve ser compatível com as configurações locais (**LC_COLLATE** e **LC_CTYPE**).

Exemplos

Crie banco de dados **music** usando UTF8 (o tipo de codificação local também é UTF8).

```
CREATE DATABASE music ENCODING 'UTF8' template = template0;
```

3.3 O que devo fazer se o tipo de data for convertido automaticamente para o tipo de carimbo de data/hora durante a criação da tabela?

Ao criar um banco de dados, você pode definir o parâmetro **DBCOMPATIBILITY** para o tipo de banco de dados compatível. O valor de **DBCOMPATIBILITY** pode ser **ORA**, **TD** e **MySQL**, indicando os bancos de dados Oracle, Teradata e MySQL, respectivamente. Se este parâmetro não for especificado durante a criação do banco de dados, o valor padrão **ORA** será usado. No modo de compatibilidade ORA, o tipo de data é automaticamente convertido em carimbo de data/hora(0). O tipo de data só é suportado no modo de compatibilidade do MySQL.

Para resolver o problema, você precisa alterar o modo de compatibilidade para o MySQL. O modo de compatibilidade de um banco de dados existente não pode ser alterado. Ele só pode ser especificado durante a criação do banco de dados. O GaussDB(DWS) suporta o modo de compatibilidade do MySQL no cluster versão 8.1.1 e posterior. Para configurar esse modo, execute os seguintes comandos:

```
gaussdb=> CREATE DATABASE mydatabase DBCOMPATIBILITY='mysql';
CREATE DATABASE
gaussdb=> \c mydatabase
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "mydatabase" as user "dbadmin".
mydatabase=> create table t1(c1 int, c2 date);
NOTICE: The 'DISTRIBUTE BY' clause is not specified. Using round-robin as the
distribution mode by default.
HINT: Please use 'DISTRIBUTE BY' clause to specify suitable data distribution
column.
CREATE TABLE
```

Se o problema não puder ser resolvido alterando a compatibilidade, você pode tentar alterar o tipo de coluna. Por exemplo, insira dados do tipo data como cadeias em uma tabela. Exemplo:

```
gaussdb=> CREATE TABLE mytable (a date,b int);
CREATE TABLE
gaussdb=> INSERT INTO mytable VALUES(date '12-08-2023',01);
INSERT 0 1
gaussdb=> SELECT * FROM mytable;
          a          | b
---------------------+---
 2023-12-08 00:00:00 | 1
(1 row)
gaussdb=> ALTER TABLE mytable MODIFY a VARCHAR(20);
ALTER TABLE
gaussdb=> INSERT INTO mytable VALUES('2023-12-10',02);
INSERT 0 1
gaussdb=> SELECT * FROM mytable;
          a          | b
---------------------+---
 2023-12-08 00:00:00 | 1
 2023-12-10          | 2
(2 rows)
```


3.4 Preciso executar VACUUM FULL e ANALYZE em tabelas comuns periodicamente?

Sim.

Para tabelas que são frequentemente adicionadas, excluídas ou modificadas, é necessário realizar periodicamente o **VACUUM FULL** e **ANALYZE** para recuperar o espaço em disco ocupado por dados atualizados ou excluídos, evitando a deterioração do desempenho causada pelo inchaço dos dados e estatísticas imprecisas.

- Geralmente, você é aconselhado a executar **ANALYZE** depois que um grande número de operações de **adição ou modificação** são executadas em uma tabela.
- Depois que uma tabela é excluída, é aconselhável executar **VACUUM** em vez de **VACUUM FULL**. No entanto, você pode executar **VACUUM FULL** em alguns casos específicos, como quando você deseja restringir fisicamente uma tabela para diminuir o espaço ocupado em disco após excluir a maioria das linhas da tabela. Para obter detalhes sobre as diferenças entre **VACUUM** e **VACUUM FULL**, consulte **VACUUM e VACUUM FULL**.

Sintaxe

Realize **ANALYZE** em uma tabela.

```
ANALYZE table_name;
```

Realize **ANALYZE** em todas as tabelas (tabelas não estrangeiras) no banco de dados.

```
ANALYZE;
```

Execute **VACUUM** em uma tabela.

```
VACUUM table_name;
```

Execute **VACUUM FULL** em uma tabela.

```
VACUUM FULL table_name;
```

Para mais detalhes, veja **VACUUM** e **ANALYZE | ANALYSE**.

NOTA

- Se o uso do espaço físico não diminuir após a execução do comando **VACUUM FULL**, verifique se havia outras transações ativas (iniciadas antes de excluir transações de dados e não terminadas antes de executar **VACUUM FULL**). Se sim, execute este comando novamente quando as transações tiverem terminado.
- Na versão 8.1.3 ou posterior, **VACUUM/VACUUM FULL** pode ser chamado no plano de gerenciamento. Para obter detalhes, consulte **O&M inteligente**.

VACUUM e VACUUM FULL

No GaussDB(DWS), a operação **VACUUM** é como um aspirador de pó usado para absorver poeira. Aqui, "poeira" significa dados antigos. Se os dados não forem limpos em tempo hábil, o espaço do banco de dados ficará inchado, causando deterioração do desempenho ou até avaria do sistema.

Finalidades de VACUUM:

- Resolver inchaço do espaço: limpar tuplas obsoletas e índices correspondentes, que incluem a tupla (e índice) de uma transação **DELETE** comprometida, a versão antiga (e índice) de uma transação **UPDATE**, a tupla inserida (e índice) de uma transação **INSERT** revertida, a nova versão (e índice) de uma transação **UPDATE** e a tupla (e índice) de uma transação **COPY**.
- VACUUM FREEZE: impedir a quebra do sistema causada pelo envolvimento do ID da transação. Ele converte IDs de transação menores que OldestXmin para congelar xids, atualizar relfrozenxids em uma tabela e atualizar relfrozenxids e truncar clogs em um banco de dados
- Atualizar as estatísticas: **VACUUM ANALYZE** atualiza as estatísticas, permitindo que o otimizador selecione uma maneira melhor de executar instruções SQL.

A instrução VACUUM inclui **VACUUM** e **VACUUM FULL**. Atualmente, **VACUUM** só pode funcionar em tabelas de armazenamento de linha. **VACUUM FULL** pode ser usado para liberar espaço de tabelas de armazenamento de colunas. Para obter detalhes, consulte a tabela a seguir.

Tabela 3-1 VACUUM e VACUUM FULL

Item	VACUUM	VACUUM FULL
Limpar espaço	Se o registro excluído estiver no final de uma tabela, o espaço ocupado pelo registro excluído será liberado fisicamente e devolvido ao sistema operacional. Se os dados não estiverem no final de uma tabela, o espaço ocupado por tuplas mortas na tabela ou índice é definido para estar disponível para reutilização.	Apesar da posição dos dados excluídos, o espaço ocupado pelos dados é liberado fisicamente e devolvido ao sistema operacional. Quando os dados são inseridos, uma nova página de disco é alocada.
Tipo de bloqueio	Bloqueio compartilhado. A operação **VACUUM** pode ser realizada em paralelo com outras operações.	Bloqueio exclusivo. Todas as operações baseadas na tabela são suspensas durante a execução.
Espaço físico	Não liberado	Liberado
ID da transação	Não recuperado	Recuperado
Sobrecarga de execução	A sobrecarga é baixa e a operação pode ser executada periodicamente.	A sobrecarga é alta. É aconselhável executá-lo quando o espaço de página do disco ocupado pelo banco de dados estiver próximo do limite e as operações de dados forem poucas.
Efeito	Melhora a eficiência das operações na tabela.	Melhora muito a eficiência das operações na tabela.

3.5 É necessário definir uma chave de distribuição após definir uma chave primária?

Não, você só precisa definir a chave primária. Por padrão, a primeira coluna da chave primária é selecionada como a chave de distribuição. Se ambas estiverem definidas, a chave primária deve conter a chave de distribuição.

3.6 O GaussDB(DWS) é compatível com os procedimentos armazenados do PostgreSQL?

Sim.

GaussDB(DWS) é compatível com os procedimentos armazenados do PostgreSQL. Para obter detalhes, consulte **Procedimentos armazenados**.

3.7 Quais são tabelas particionadas, partições e chaves de partição?

Tabela particionada: o particionamento refere-se a dividir o que é logicamente uma grande tabela em pedaços físicos menores com base em esquemas específicos. A tabela baseada na lógica é chamada de tabela particionada, e uma parte física é chamada de partição. Os dados são armazenados nessas partes físicas menores, ou seja, partições, em vez da tabela particionada lógica maior.

Partição: no sistema distribuído do GaussDB(DWS), particionamento de dados é particionar horizontalmente dados da tabela dentro de um nó com base em uma política especificada. A tabela é dividida em partições que não se sobrepõem dentro de um intervalo específico.

Chave de partição: uma chave de partição é um conjunto ordenado de uma ou mais colunas de tabela. Os valores nas chaves de partição da tabela são usados para determinar a partição de dados à qual uma linha pertence.

3.8 Como exportar a estrutura da tabela?

Recomendamos que você use o cliente gráfico do Data Studio para exportar dados da tabela. Você pode exportar dados de:

- Uma tabela específica
- Todas as tabelas em um esquema
- Todas as tabelas em um banco de dados

Alternativamente, use **gs_dump** e **gs_dumpall** para exportar dados. Você pode:

- Exportar um único banco de dados:
  - Exportação em nível de banco de dados
  - Exportação em nível de modo

  - Exportação em nível de tabela
- Exportar todos os bancos de dados:
  - Exportação em nível de banco de dados
  - Exportação de objetos globais de cada banco de dados

Para obter detalhes, consulte **gs_dump** e **gs_dumpall**.

3.9 Como excluir dados da tabela de forma eficiente?

Sim. **TRUNCATE** é mais eficiente que **DELETE** para excluir dados em massa.

Para obter detalhes, consulte **TRUNCATE**.

Função

TRUNCATE remove rapidamente todas as linhas de uma tabela de banco de dados.

Tem o mesmo efeito que **DELETE** incondicional, mas **TRUNCATE** é mais rápido, especialmente para tabelas grandes, porque não verifica tabelas.

Funções

- **TRUNCATE TABLE** funciona como uma instrução **DELETE** sem cláusula de **WHERE**, ou seja, esvaziando uma tabela.
- **TRUNCATE TABLE** usa menos recursos do sistema e do log de transações.
  - **DELETE** exclui uma linha a cada vez e registra cada exclusão no log de transações.
  - **TRUNCATE TABLE** exclui todas as linhas em uma tabela liberando a página de dados e registra apenas cada liberação da página de dados no log de transações.
- **TRUNCATE**, **DELETE** e **DROP** são diferentes em que:
  - **TRUNCATE TABLE** exclui conteúdo, libera espaço, mas não exclui definições.
  - **DELETE TABLE** exclui conteúdo, mas não exclui definições ou libera espaço.
  - **DROP TABLE** exclui conteúdo e definições e libera espaço.

Exemplos

- Crie uma tabela.

```
CREATE TABLE tpcds.reason_t1 AS TABLE tpcds.reason;
```

  Trunque a tabela.

```
TRUNCATE TABLE tpcds.reason_t1;
```

  Exclua a tabela.

```
DROP TABLE tpcds.reason_t1;
```

- Crie uma tabela particionada.

```
CREATE TABLE tpcds.reason_p
(
  r_reason_sk integer,
  r_reason_id character(16),
  r_reason_desc character(100)
)PARTITION BY RANGE (r_reason_sk)
(
```



```
  partition p_05_before values less than (05),
  partition p_15 values less than (15),
  partition p_25 values less than (25),
  partition p_35 values less than (35),
  partition p_45_after values less than (MAXVALUE)
);
```

Insira dados.

```
INSERT INTO tpcds.reason_p SELECT * FROM tpcds.reason;
```

Trunque a partição **p_05_before**.

```
ALTER TABLE tpcds.reason_p TRUNCATE PARTITION p_05_before;
```

Trunque a partição **p_15**.

```
ALTER TABLE tpcds.reason_p TRUNCATE PARTITION for (13);
```

Trunque a tabela particionada.

```
TRUNCATE TABLE tpcds.reason_p;
```

Exclua a tabela.

```
DROP TABLE tpcds.reason_p;
```

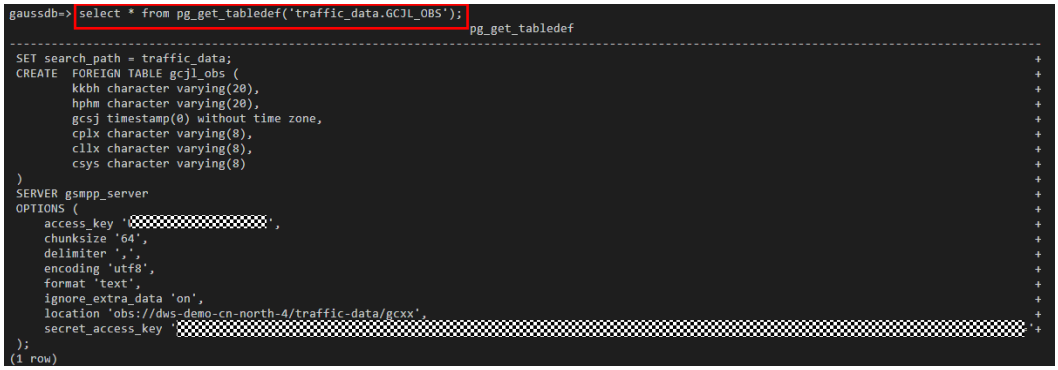
3.10 Como exibir informações de tabela estrangeira?

Para consultar informações sobre tabelas estrangeiras do OBS/GDS, como caminhos do OBS, execute a seguinte instrução:

```
SELECT * FROM pg_get_tabledef ('foreign_table_name')
```

A seguir usa a tabela **traffic_data.GCJL_OBS** como um exemplo:

```
SELECT * FROM pg_get_tabledef('traffic_data.GCJL_OBS');
```



3.11 Se nenhuma coluna de distribuição for especificada, como os dados serão armazenados?

> **NOTA**
>
> Para clusters 8.1.2 ou posterior, você pode usar o parâmetro GUC **default_distribution_mode** para consultar e definir o modo de distribuição de tabela padrão.

Se nenhuma coluna de distribuição for especificada durante a criação da tabela, os dados serão armazenados da seguinte forma:

- Cenário 1

Se a chave primária ou restrição exclusiva for incluída durante a criação da tabela, a distribuição de hash será selecionada. A coluna de distribuição é a coluna correspondente à chave primária ou restrição exclusiva.

```
CREATE TABLE warehouse1
(
    W_WAREHOUSE_SK            INTEGER            PRIMARY KEY,
    W_WAREHOUSE_ID            CHAR(16)              NOT NULL,
    W_WAREHOUSE_NAME          VARCHAR(20)
);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index
"warehouse1_pkey" for table "warehouse1"
CREATE TABLE

SELECT getdistributekey('warehouse1');
 getdistributekey
------------------
 w_warehouse_sk
(1 row)
```

- Cenário 2

  Se a chave primária ou restrição exclusiva não for incluída durante a criação da tabela, mas houver colunas cujos tipos de dados podem ser usados como colunas de distribuição, a distribuição de hash será selecionada. A coluna de distribuição é a primeira coluna cujo tipo de dados pode ser usado como uma coluna de distribuição.

```
CREATE TABLE warehouse2
(
    W_WAREHOUSE_SK            INTEGER                       ,
    W_WAREHOUSE_ID            CHAR(16)              NOT NULL,
    W_WAREHOUSE_NAME          VARCHAR(20)
);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'w_warehouse_sk'
as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data
distribution column.
CREATE TABLE

SELECT getdistributekey('warehouse2');
 getdistributekey
------------------
 w_warehouse_sk
(1 row)
```

- Cenário 3

  Se a chave primária ou restrição exclusiva não for incluída durante a criação da tabela e não existir nenhuma coluna cujo tipo de dados possa ser usado como uma coluna de distribuição, a distribuição round-robin será selecionada.

```
CREATE TABLE warehouse3
(
    W_WAREHOUSE_ID            CHAR(16)              NOT NULL,
    W_WAREHOUSE_NAME          VARCHAR(20)
);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'w_warehouse_id'
as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data
distribution column.
CREATE TABLE

SELECT getdistributekey('warehouse3');
 getdistributekey
------------------
 w_warehouse_id
(1 row)
```


3.12 Como substituir o resultado nulo por 0?

Quando OUTER JOIN (LEFT JOIN, RIGHT JOIN e FULL JOIN) é executada, a falha de correspondência na outer join gera um grande número de valores NULL. Você pode substituir esses valores nulos por 0.

Você pode usar a função **COALESCE** para fazer isso. Esta função retorna o primeiro valor de parâmetro não nulo na lista de parâmetros. Por exemplo:

```
SELECT coalesce(NULL,'hello');
 coalesce
----------
 hello
(1 row)
```

Use left join para ingressar nas tabelas **course1** e **course2**.

```
SELECT * FROM course1;
  stu_id  |  stu_name  |     cour_name
----------+------------+--------------------
 20110103 | ALLEN      | Math
 20110102 | JACK       | Programming Design
 20110101 | MAX        | Science
(3 rows)

SELECT * FROM course2;
 cour_id |     cour_name      | teacher_name
---------+--------------------+--------------
    1002 | Programming Design | Mark
    1001 | Science            | Anne
(2 rows)

SELECT course1.stu_name,course2.cour_id,course2.cour_name,course2.teacher_name
FROM course1 LEFT JOIN course2 ON course1.cour_name = course2.cour_name ORDER BY
1;
  stu_name  | cour_id |     cour_name      | teacher_name
------------+---------+--------------------+--------------
 ALLEN      |         |                    |
 JACK       |    1002 | Programming Design | Mark
 MAX        |    1001 | Science            | Anne
(3 rows)
```

Use a função **COALESCE** para substituir valores nulos no resultado da consulta por 0 ou outros valores diferentes de zero:

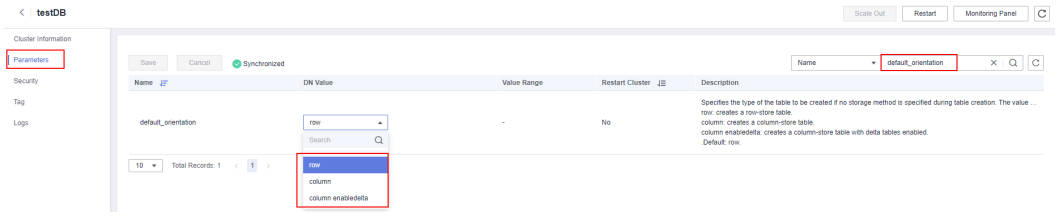
```
SELECT course1.stu_name,
 coalesce(course2.cour_id,0) AS cour_id,
 coalesce(course2.cour_name,'NA') AS cour_name,
 coalesce(course2.teacher_name,'NA') AS teacher_name
 FROM course1
 LEFT JOIN course2 ON course1.cour_name = course2.cour_name
 ORDER BY 1;
  stu_name  | cour_id |     cour_name      | teacher_name
------------+---------+--------------------+--------------
 ALLEN      |       0 | NA                 | NA
 JACK       |    1002 | Programming Design | Mark
 MAX        |    1001 | Science            | Anne
(3 rows)
```


3.13 Como verificar se uma tabela é armazenada em linha ou em coluna?

O modo de armazenamento de uma tabela é controlado pelo parâmetro ORIENTATION na instrução de criação de tabela. **row** indica armazenamento de linha e **column** indica armazenamento de coluna.

Em 8.1.2 e versões anteriores, se **ORIENTATION** não for especificado, o armazenamento de linha é usado por padrão.

Em versões posteriores a 8.1.3, o parâmetro **default_orientation** pode ser usado para controlar o modo de armazenamento. Se o parâmetro **ORIENTATION** não for especificado durante a criação da tabela, o modo de armazenamento da tabela é baseado no valor de **default_orientation**. **row** indica uma tabela de armazenamento de linhas, **column** indica uma tabela de armazenamento de colunas e **column enabledelta** indica que uma tabela de armazenamento de colunas com delta ativado é criado. O parâmetro GUC pode ser configurado no console do GaussDB(DWS), conforme mostrado na figura a seguir.



Você pode usar a função de definição de tabela **PG_GET_TABLEDEF** para verificar se a tabela criada é de armazenamento de linha ou de coluna.

Por exemplo, **orientation=column** indica uma tabela de armazenamento de colunas.

Atualmente, não é possível executar a instrução **ALTER TABLE** para modificar o parâmetro **ORIENTATION**.

```
SELECT * FROM PG_GET_TABLEDEF('customer_t1');

pg_get_tabledef
--------------------------------------------------------------------------------
-
 SET search_path = tpchobs;
+
 CREATE  TABLE customer_t1
(                                                                +
          c_customer_sk integer,
+
          c_customer_id character(5),
+
          c_first_name character(6),
+
          c_last_name character(8)
+
 )
+
 WITH (orientation=column, compression=middle, colversion=2.0, enable_delta=false)
+
 DISTRIBUTE BY HASH(c_last_name)
+
 TO GROUP group_version1;
(1 row)
```


3.14 Como consultar as informações sobre tabelas de armazenamento de colunas do GaussDB(DWS)?

As instruções SQL a seguir são usadas para consultar informações comuns sobre tabelas de armazenamento de colunas:

Crie uma tabela particionada de armazenamento de colunas **my_table**.

```
CREATE TABLE my_table
(
    product_id INT,
    product_name VARCHAR2(40),
    product_quantity INT
 )
WITH (ORIENTATION = COLUMN)
PARTITION BY range(product_quantity)
(
partition my_table_p1 values less than(600),
partition my_table_p2 values less than(800),
partition my_table_p3 values less than(950),
partition my_table_p4 values less than(1000));

INSERT INTO my_table VALUES(1011, 'tents', 720);
INSERT INTO my_table VALUES(1012, 'hammock', 890);
INSERT INTO my_table VALUES(1013, 'compass', 210);
INSERT INTO my_table VALUES(1014, 'telescope', 490);
INSERT INTO my_table VALUES(1015, 'flashlight', 990);
INSERT INTO my_table VALUES(1016, 'ropes', 890);
```

Execute o seguinte comando para exibir a tabela particionada de armazenamento de colunas criada:

```
SELECT * FROM my_table;
 product_id | product_name | product_quantity
------------+--------------+------------------
       1013 | compass      |              210
       1014 | telescope    |              490
       1011 | tents        |              720
       1015 | flashlight   |              990
       1012 | hammock      |              890
       1016 | ropes        |              890
(6 rows)
```

Consultar o limite de uma partição

```
SELECT relname, partstrategy, boundaries FROM pg_partition where parentid=(select
parentid from pg_partition where relname='my_table');
   relname   | partstrategy | boundaries
-------------+--------------+------------
 my_table    | r            |
 my_table_p1 | r            | {600}
 my_table_p2 | r            | {800}
 my_table_p3 | r            | {950}
 my_table_p4 | r            | {1000}
(5 rows)
```

Consultar o número de colunas em uma tabela de armazenamento de colunas

```
SELECT count(*) FROM ALL_TAB_COLUMNS where table_name='my_table';
 count
-------
     3
(1 row)
```


Consultar a distribuição de dados em DNs

```
SELECT table_skewness('my_table');
           table_skewness
-------------------------------------
 ("dn_6007_6008        ",3,50.000%)
 ("dn_6009_6010        ",2,33.333%)
 ("dn_6003_6004        ",1,16.667%)
 ("dn_6001_6002        ",0,0.000%)
 ("dn_6005_6006        ",0,0.000%)
 ("dn_6011_6012        ",0,0.000%)
(6 rows)
```

Consultar os nomes das tabelas Cudesc e Delta na partição P1 em um DN

```
EXECUTE DIRECT ON (dn_6003_6004) 'select a.relname from pg_class a, pg_partition
b where (a.oid=b.reldeltarelid or a.oid=b.relcudescrelid) and
b.relname=''my_table_p1''';
       relname
----------------------
 pg_delta_part_60317
 pg_cudesc_part_60317
(2 rows)
```

3.15 Por que às vezes os índices de consulta do GaussDB(DWS) se tornam inválidos?

A criação de índices para tabelas pode melhorar o desempenho da consulta do banco de dados. No entanto, às vezes os índices não podem ser usados em um plano de consulta. Esta seção descreve vários motivos comuns e métodos de otimização.

Razão 1: os conjuntos de resultados retornados são grandes.

O seguinte usa Seq Scan e Index Scan em uma tabela de armazenamento de linha como um exemplo:

- Seq Scan: pesquisa registros de tabelas em sequência. Todos os registros são recuperados durante cada varredura. Este é o método de digitalização de tabela mais simples e básico, e seu custo é alto.
- Index Scan: pesquisa o índice primeiro, encontra o local de destino (ponteiro) no índice e, em seguida, recupera dados na página de destino.

A varredura de índice é mais rápida do que a varredura de sequência na maioria dos casos. No entanto, se os conjuntos de resultados obtidos representarem uma grande proporção (mais de 70%) de todos os dados, Index Scan precisa verificar os índices antes de ler os dados da tabela. Isso torna a varredura de tabela mais lenta.

Razão 2: ANALYZE não é realizada em tempo hábil.

ANALYZE é usada para atualizar as estatísticas da tabela. Se **ANALYZE** não for executada em uma tabela ou uma grande quantidade de dados for adicionada ou excluída de uma tabela após a execução de **ANALYZE**, as estatísticas podem ser imprecisas, o que pode fazer com que uma consulta pule o índice.

Método da optimização: execute a instrução **ANALYZE** na tabela para atualizar as estatísticas.

Razão 3: condições de filtragem contém funções ou conversão de tipo de dados implícita

Se cálculo, função ou conversão implícita de tipo de dados estiver contida nos critérios de filtro, os índices podem falhar ao serem selecionados.

Por exemplo, quando uma tabela é criada, os índices são criados nas colunas **a**, **b** e **c**.

```
create table test(a int, b text, c date);
```

- Execute o cálculo nas colunas indexadas.

  A saída do comando a seguir indica que ambos **where a = 101** e **where a = 102 - 1** usam o índice na coluna a, mas **where a + 1 = 102** não usa o índice.

```
explain verbose select * from test where a  = 101;
                                                 QUERY PLAN
-------------------------------------------------------------------------------------
-----------------------------
  id |                   operation                    | E-rows | E-distinct |
E-memory | E-width | E-costs
 ----+------------------------------------------------+--------+------------
+----------+---------+---------
   1 | ->  Streaming (type: GATHER)                   |      1 |
|          |      44 | 16.27
   2 |    ->  Index Scan using index_a on public.test |      1 |            |
1MB      |      44 | 8.27

 Predicate Information (identified by plan id)
 -----------------------------------------
   2 --Index Scan using index_a on public.test
         Index Cond: (test.a = 101)

 Targetlist Information (identified by plan id)
 -----------------------------------------
   1 --Streaming (type: GATHER)
         Output: a, b, c
         Node/s: dn_6005_6006
   2 --Index Scan using index_a on public.test
         Output: a, b, c
         Distribute Key: a

   ====== Query Summary =====
 ------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(24 rows)
 explain verbose select * from test where a  = 102 - 1;
                                                 QUERY PLAN
-------------------------------------------------------------------------------------
-----------------------------
  id |                   operation                    | E-rows | E-distinct |
E-memory | E-width | E-costs
 ----+------------------------------------------------+--------+------------
+----------+---------+---------
   1 | ->  Streaming (type: GATHER)                   |      1 |
|          |      44 | 16.27
   2 |    ->  Index Scan using index_a on public.test |      1 |            |
1MB      |      44 | 8.27

 Predicate Information (identified by plan id)
 -----------------------------------------
   2 --Index Scan using index_a on public.test
         Index Cond: (test.a = 101)

 Targetlist Information (identified by plan id)
 -----------------------------------------
   1 --Streaming (type: GATHER)
```



```
        Output: a, b, c
        Node/s: dn_6005_6006
   2 --Index Scan using index_a on public.test
        Output: a, b, c
        Distribute Key: a

   ====== Query Summary =====
 -------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(24 rows)
explain verbose select * from test where a + 1 = 102;
                                        QUERY PLAN
-------------------------------------------------------------------------------
--------------
  id |           operation            | E-rows | E-distinct | E-memory | E-
width | E-costs
 ----+--------------------------------+--------+------------+----------
+---------+---------
   1 | ->  Streaming (type: GATHER)   |      1 |            |          |
44 | 22.21
   2 |    ->  Seq Scan on public.test |      1 |            | 1MB      |
44 | 14.21

 Predicate Information (identified by plan id)
 -----------------------------------------
   2 --Seq Scan on public.test
        Filter: ((test.a + 1) = 102)

 Targetlist Information (identified by plan id)
 ------------------------------------------
   1 --Streaming (type: GATHER)
        Output: a, b, c
        Node/s: All datanodes
   2 --Seq Scan on public.test
        Output: a, b, c
        Distribute Key: a

   ====== Query Summary =====
 -------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(24 rows)
```

Método da optimização: use constantes em vez de expressões ou coloque cálculo constante à direita do sinal de igual (=).

- Use funções em colunas indexadas.

  De acordo com o seguinte resultado de execução, se uma função for usada em uma coluna indexada, o índice não será selecionado.

```
explain verbose select * from test where to_char(c, 'yyyyMMdd') =
to_char(CURRENT_DATE,'yyyyMMdd');
                                                                        QUERY PLAN
-------------------------------------------------------------------------------
-------------------------------------------------------------
  id |           operation            | E-rows | E-distinct | E-memory | E-
width | E-costs
 ----+--------------------------------+--------+------------+----------
+---------+---------
   1 | ->  Streaming (type: GATHER)   |      1 |            |          |
44 | 22.28
   2 |    ->  Seq Scan on public.test |      1 |            | 1MB      |
44 | 14.28

                                             Predicate Information
(identified by plan id)
```



```
------------------------------------------------------------------------------
-------------------------------------------------------
   2 --Seq Scan on public.test
         Filter: (to_char(test.c, 'yyyyMMdd'::text) =
to_char(('2022-11-30'::pg_catalog.date)::timestamp with time zone,
'yyyyMMdd'::text))

 Targetlist Information (identified by plan id)
 -----------------------------------------------
   1 --Streaming (type: GATHER)
         Output: a, b, c
         Node/s: All datanodes
   2 --Seq Scan on public.test
         Output: a, b, c
         Distribute Key: a

   ====== Query Summary =====
 -------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(24 rows)
explain verbose select * from test where c = current_date;
                                                  QUERY PLAN
------------------------------------------------------------------------------
-------------------------------
  id |                   operation                   | E-rows | E-distinct |
E-memory | E-width | E-costs
 ----+-----------------------------------------------+--------+------------
+----------+---------+---------
   1 | ->  Streaming (type: GATHER)                  |      1 |
|          |      44 | 16.27
   2 |    ->  Index Scan using index_c on public.test |      1 |            |
1MB      |      44 | 8.27

        Predicate Information (identified by plan id)
 ------------------------------------------------------------
   2 --Index Scan using index_c on public.test
         Index Cond: (test.c = '2022-11-30'::pg_catalog.date)

 Targetlist Information (identified by plan id)
 -----------------------------------------------
   1 --Streaming (type: GATHER)
         Output: a, b, c
         Node/s: All datanodes
   2 --Index Scan using index_c on public.test
         Output: a, b, c
         Distribute Key: a

   ====== Query Summary =====
 -------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(24 rows)
```

Método da optimização: não use funções desnecessárias em colunas indexadas.

- Conversão implícita de tipos de dados.

  Esse cenário é comum. Por exemplo, o tipo de coluna **b** é Text e a condição de filtragem é **where b = 2**. Durante a geração do plano, o tipo Text é implicitamente convertido para o tipo Bigint e a condição de filtragem real muda para **where b::bigint = 2**. Como resultado, o índice na coluna **b** se torna inválido.

```
explain verbose select * from test where b = 2;
                                        QUERY PLAN
------------------------------------------------------------------------------
--------------
  id |           operation            | E-rows | E-distinct | E-memory | E-
```



```
width | E-costs
 ----+--------------------------------+--------+------------+----------
+---------+---------
   1 | ->  Streaming (type: GATHER)   |      1 |            |          |
44 | 22.21
   2 |    ->  Seq Scan on public.test |      1 |            | 1MB      |
44 | 14.21

 Predicate Information (identified by plan id)
 -----------------------------------------------
   2 --Seq Scan on public.test
         Filter: ((test.b)::bigint = 2)

 Targetlist Information (identified by plan id)
 -----------------------------------------------
   1 --Streaming (type: GATHER)
         Output: a, b, c
         Node/s: All datanodes
   2 --Seq Scan on public.test
         Output: a, b, c
         Distribute Key: a

   ====== Query Summary =====
 -------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(24 rows)
explain verbose select * from test where b = '2';
                                                 QUERY PLAN
-------------------------------------------------------------------------------------
-------------------------------
  id |                   operation                   | E-rows | E-distinct |
E-memory | E-width | E-costs
 ----+-----------------------------------------------+--------+------------
+----------+---------+---------
   1 | ->  Streaming (type: GATHER)                  |      1 |
|          |      44 | 16.27
   2 |    ->  Index Scan using index_b on public.test |      1 |            |
1MB      |      44 | 8.27

 Predicate Information (identified by plan id)
 -----------------------------------------------
   2 --Index Scan using index_b on public.test
         Index Cond: (test.b = '2'::text)

 Targetlist Information (identified by plan id)
 -----------------------------------------------
   1 --Streaming (type: GATHER)
         Output: a, b, c
         Node/s: All datanodes
   2 --Index Scan using index_b on public.test
         Output: a, b, c
         Distribute Key: a

   ====== Query Summary =====
 -------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(24 rows)
```

Método da optimização: use constantes do mesmo tipo que a coluna indexada para evitar a conversão de tipo implícita.

Cenário 4: Hashjoin é substituído por Nestloop + Indexscan.

Quando duas tabelas são juntadas, o número de linhas no conjunto de resultados filtradas pela condição WHERE em uma tabela é pequeno, portanto, o número de linhas no conjunto de resultados finais também é pequeno. Neste caso, o efeito de nestloop+indexscan é melhor do que o de hashjoin. O melhor plano de execução é o seguinte:

Você pode ver que Index Cond: (t1.b = t2.b) na camada 5 empurrou a condição de junção para baixo para a varredura da tabela base.

```
explain verbose select t1.a,t1.b from t1,t2 where t1.b=t2.b and t2.a=4;
 id |                    operation                     | E-rows | E-distinct | E-
memory | E-width | E-costs
----+--------------------------------------------------+--------+------------
+----------+---------+---------
  1 | ->  Streaming (type: GATHER)                     |     26 |
|          |       8 | 17.97
  2 |    ->  Nested Loop (3,5)                         |     26 |            |
1MB      |       8 | 11.97
  3 |       ->  Streaming(type: BROADCAST)             |      2 |            |
2MB      |       4 | 2.78
  4 |          ->  Seq Scan on public.t2               |      1 |            |
1MB      |       4 | 2.62
  5 |       ->  Index Scan using t1_b_idx on public.t1 |     26 |            |
1MB      |       8 | 9.05
(5 rows)

 Predicate Information (identified by plan id)
-----------------------------------------------
   4 --Seq Scan on public.t2
         Filter: (t2.a = 4)
   5 --Index Scan using t1_b_idx on public.t1
         Index Cond: (t1.b = t2.b)
(4 rows)

 Targetlist Information (identified by plan id)
------------------------------------------------
   1 --Streaming (type: GATHER)
         Output: t1.a, t1.b
         Node/s: All datanodes
   2 --Nested Loop (3,5)
         Output: t1.a, t1.b
   3 --Streaming(type: BROADCAST)
         Output: t2.b
         Spawn on: datanode2
         Consumer Nodes: All datanodes
   4 --Seq Scan on public.t2
         Output: t2.b
         Distribute Key: t2.a
   5 --Index Scan using t1_b_idx on public.t1
         Output: t1.a, t1.b
         Distribute Key: t1.a
(15 rows)

   ====== Query Summary =====
---------------------------------
 System available mem: 9262694KB
 Query Max mem: 9471590KB
 Query estimated mem: 5144KB
(3 rows)
```

Se o otimizador não selecionar tal plano de execução, você poderá otimizá-lo da seguinte maneira:

```
set enable_index_nestloop = on;
set enable_hashjoin = off;
set enable_seqscan = off;
```


Razão 5: o método de verificação é especificado incorretamente por dicas.

As dicas de plano do GaussDB(DWS) podem especificar três métodos de varredura: tablescan, indexscan e indexonlyscan.

- Table Scan: varredura de tabela completa, como Seq Scan de tabelas de armazenamento de linha e CStore Scan de tabelas de armazenamento de coluna.
- Index Scan: verifica índices e, em seguida, obtém registros de tabela com base nos índices.
- Index-Only Scan: verifica os índices, que cobrem todos os resultados necessários. Comparada com index scan, a index-only scan cobre todas as colunas consultadas. Dessa forma, apenas os índices são recuperados e os registros de dados não precisam ser recuperados.

Em cenários de Index-Only Scan, a Index Scan especificada por uma dica será inválida.

```
explain verbose select/*+ indexscan(test)*/ b from test where b = '1';
WARNING:  unused hint: IndexScan(test)
                                                  QUERY PLAN
------------------------------------------------------------------------------------------
------------------------------
  id |                       operation                       | E-rows | E-distinct
| E-memory | E-width | E-costs
 ----+-------------------------------------------------------+--------+------------
+----------+---------+---------
   1 | ->  Streaming (type: GATHER)                          |      1 |
|          |      32 | 16.27
   2 |    ->  Index Only Scan using index_b on public.test |      1 |
| 1MB      |      32 | 8.27

   Predicate Information (identified by plan id)
 -----------------------------------------------
   2 --Index Only Scan using index_b on public.test
         Index Cond: (test.b = '1'::text)

   Targetlist Information (identified by plan id)
 -----------------------------------------------
   1 --Streaming (type: GATHER)
         Output: b
         Node/s: All datanodes
   2 --Index Only Scan using index_b on public.test
         Output: b
         Distribute Key: a

   ====== Query Summary =====
 ------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(24 rows)
explain verbose select/*+ indexonlyscan(test)*/ b from test where b = '1';
                                                  QUERY PLAN
------------------------------------------------------------------------------------------
------------------------------
  id |                       operation                       | E-rows | E-distinct
| E-memory | E-width | E-costs
 ----+-------------------------------------------------------+--------+------------
+----------+---------+---------
   1 | ->  Streaming (type: GATHER)                          |      1 |
|          |      32 | 16.27
   2 |    ->  Index Only Scan using index_b on public.test |      1 |
| 1MB      |      32 | 8.27

   Predicate Information (identified by plan id)
 -----------------------------------------------
   2 --Index Only Scan using index_b on public.test
```



```
        Index Cond: (test.b = '1'::text)

  Targetlist Information (identified by plan id)
------------------------------------------------
  1 --Streaming (type: GATHER)
        Output: b
        Node/s: All datanodes
  2 --Index Only Scan using index_b on public.test
        Output: b
        Distribute Key: a

  ====== Query Summary =====
------------------------------
System available mem: 3358720KB
Query Max mem: 3358720KB
Query estimated mem: 1024KB
(24 rows)
```

Método da optimização: especifique corretamente Index scan e Index-Only Scan.

Razão 6: uso Incorreto de índice GIN na recuperação de texto completo

Para acelerar a pesquisa de texto, você pode criar um índice GIN para pesquisa de texto completo.

```
CREATE INDEX idxb ON test using gin(to_tsvector('english',b));
```

Ao criar o índice GIN, você deve usar a versão de 2 argumentos de to_tsvector. Somente quando a consulta também usa a versão de 2 argumentos e os argumentos são os mesmos que no índice GIN, o índice GIN pode ser chamado.

NOTA

A função to_tsvector() aceita um ou dois argumentos. Se a versão de um argumento do índice for usada, o sistema usará a configuração especificada por **default_text_search_config** por padrão. Para criar um índice, a versão de dois argumentos deve ser usada, ou o conteúdo do índice pode ser inconsistente.

```
explain verbose select  * from test where to_tsvector(b) @@ to_tsquery('cat')
order by 1;
                                          QUERY PLAN
------------------------------------------------------------------------------
-------------
  id |             operation             | E-rows | E-distinct | E-memory | E-
width | E-costs
 ----+-----------------------------------+--------+------------+----------
+---------+---------
   1 | ->  Streaming (type: GATHER)      |      2 |            |          |
44 | 22.23
   2 |    ->  Sort                       |      2 |            | 16MB     |
44 | 14.23
   3 |       ->  Seq Scan on public.test |      1 |            | 1MB      |
44 | 14.21

        Predicate Information (identified by plan id)
 -------------------------------------------------------------
   3 --Seq Scan on public.test
         Filter: (to_tsvector(test.b) @@ '''cat'''::tsquery)

 Targetlist Information (identified by plan id)
 ------------------------------------------------
   1 --Streaming (type: GATHER)
         Output: a, b, c
         Merge Sort Key: test.a
         Node/s: All datanodes
   2 --Sort
         Output: a, b, c
         Sort Key: test.a
```



```
   3 --Seq Scan on public.test
         Output: a, b, c
         Distribute Key: a

   ====== Query Summary =====
 -------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 1024KB
(29 rows)
explain verbose select  * from test where to_tsvector('english',b) @@
to_tsquery('cat') order by 1;
                                                QUERY PLAN
------------------------------------------------------------------------------------------
--------------------
  id |                operation                | E-rows | E-distinct | E-memory
| E-width | E-costs
 ----+-----------------------------------------+--------+------------+----------
+---------+---------
   1 | ->  Streaming (type: GATHER)            |      2 |            |
|      44 | 20.03
   2 |    ->  Sort                             |      2 |            | 16MB
|      44 | 12.03
   3 |       ->  Bitmap Heap Scan on public.test |      1 |            | 1MB
|      44 | 12.02
   4 |          ->  Bitmap Index Scan          |      1 |            | 1MB
|       0 | 8.00

                     Predicate Information (identified by plan id)

------------------------------------------------------------------------------------------
-----
   3 --Bitmap Heap Scan on public.test
         Recheck Cond: (to_tsvector('english'::regconfig, test.b) @@
'''cat'''::tsquery)
   4 --Bitmap Index Scan
         Index Cond: (to_tsvector('english'::regconfig, test.b) @@
'''cat'''::tsquery)

 Targetlist Information (identified by plan id)
 ------------------------------------------------
   1 --Streaming (type: GATHER)
         Output: a, b, c
         Merge Sort Key: test.a
         Node/s: All datanodes
   2 --Sort
         Output: a, b, c
         Sort Key: test.a
   3 --Bitmap Heap Scan on public.test
         Output: a, b, c
         Distribute Key: a

   ====== Query Summary =====
 -------------------------------
 System available mem: 3358720KB
 Query Max mem: 3358720KB
 Query estimated mem: 2048KB
(32 rows)
```

Método da optimização: use a versão de dois argumentos de to_tsvector para a consulta e certifique-se de que os valores do argumento sejam os mesmos do índice.

3.16 Como usar uma função definida pelo usuário para reescrever a função CRC32()?

Atualmente, o GaussDB(DWS) não tem uma função **CRC32()** embutida. Em vez disso, você pode usar a função definida pelo usuário de GaussDB(DWS) para reescrever a função **CRC32()**.

- CRC32(expr)
- Descrição: calcula a redundância cíclica. O parâmetro de entrada **expr** é uma cadeia. Se o parâmetro for NULL, NULL é retornado. Caso contrário, um valor não assinado de 32 bits é retornado após o cálculo de redundância.

Exemplo de reescrever a função **CRC32()** usando a declaração de função definida pelo usuário de GaussDB(DWS):

```
CREATE OR REPLACE FUNCTION crc32(text_string text) RETURNS bigint AS $$
DECLARE
    val bigint;
    i int;
    j int;
    byte_length int;
    binary_string bytea;
BEGIN
    IF text_string is null THEN
        RETURN null;
    ELSIF text_string = '' THEN
        RETURN 0;
    END IF;

    i = 0;
    val = 4294967295;
    byte_length = bit_length(text_string) / 8;
    binary_string = decode(replace(text_string, E'\\', E'\\\\'), 'escape');
    LOOP
        val = (val # get_byte(binary_string, i))::bigint;
        i = i + 1;
        j = 0;
        LOOP
            val = ((val >> 1) # (3988292384 * (val & 1)))::bigint;
            j = j + 1;
            IF j >= 8 THEN
                EXIT;
            END IF;
        END LOOP;
        IF i >= byte_length THEN
            EXIT;
        END IF;
    END LOOP;
    RETURN (val # 4294967295);
END
$$ IMMUTABLE LANGUAGE plpgsql;
```

Verifique o resultado da reescrita.

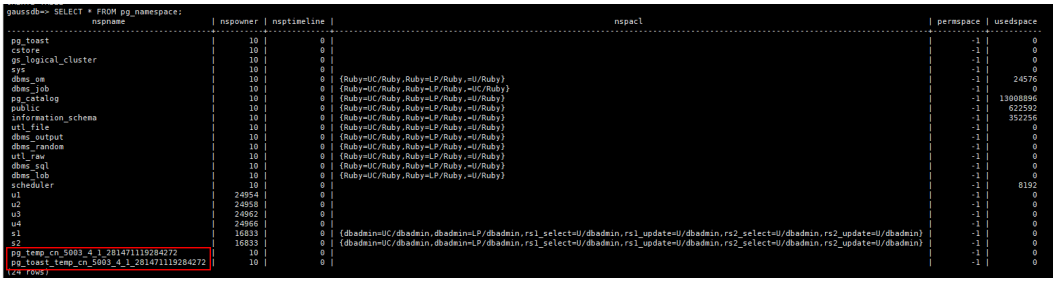
```
select crc32(null),crc32(''),crc32('1');
 crc32 | crc32 |   crc32
-------+-------+------------
       |     0 | 2212294583
(1 row)
```

Para obter detalhes sobre como usar funções definidas pelo usuário, consulte **CREATE FUNCTION**.

3.17 Quais são os esquemas começando com pg_toast_temp* ou pg_temp*?

Quando você consulta a lista de esquemas, o resultado da consulta pode conter esquemas começando com **pg_temp*** ou **pg_toast_temp***, conforme mostrado na figura a seguir.

```
SELECT * FROM pg_namespace;
```



Esses esquemas são criados quando tabelas temporárias são criadas. Cada sessão tem um esquema independente começando com **pg_temp** para garantir que as tabelas temporárias sejam visíveis apenas para a sessão atual. Portanto, não é recomendável excluir manualmente esquemas que começam com **pg_temp** ou **pg_toast_temp** durante operações de rotina.

As tabelas temporárias são visíveis apenas na sessão atual e são automaticamente excluídas após o término da sessão. Os esquemas correspondentes também são excluídos.

3.18 Soluções para resultados de consultas inconsistentes do GaussDB(DWS)

No GaussDB(DWS), às vezes uma consulta SQL pode obter resultados diferentes. Esse problema é provavelmente causado por sintaxe ou uso inadequado. Para evitar esse problema, use a sintaxe corretamente. A seguir estão alguns exemplos de inconsistência de resultados de consulta junto com as soluções.

Os resultados da função de janela são classificados incompletamente

Cenário:

Na função de janela **row_number()**, a coluna **c** da tabela **t3** é consultada após a ordenação. Os dois resultados da consulta são diferentes.

```
select * from t3 order by 1,2,3;
 a | b | c
---+---+---
 1 | 2 | 1
 1 | 2 | 2
 1 | 2 | 3
(3 rows)

select c,rn from (select c,row_number() over(order by a,b) as rn from t3) where
rn = 1;
 c | rn
---+----
1  |  1
(1 row)
select c,rn from (select c,row_number() over(order by a,b) as rn from t3) where
```



```
rn = 1;
 c | rn
---+----
 3 |  1
(1 row)
```

Análise:

Como mostrado acima, execute s**select c,rn from (select c,row_number() over(order by a,b) as rn from t3) where rn = 1;** duas vezes, os resultados são diferentes. Isso ocorre porque os valores duplicados **1** e **2** existem nas colunas de classificação **a** e **b** da função de janela, enquanto seus valores na coluna **c** são diferentes. Como resultado, quando o primeiro registro é obtido com base no resultado de classificação nas colunas **a** e **b**, os dados obtidos na coluna **c** são aleatórios, como resultado, os conjuntos de resultados são inconsistentes.

Solução:

Os valores na coluna **c** precisam ser adicionados à classificação.

```
select c,rn from (select c,row_number() over(order by a,b,c) as rn from t3) where
rn = 1;
 c | rn
---+----
 1 |  1
(1 row)
```

Usar a classificação em subvisões/subconsultas

Cenário

Depois que a tabela **test** e a visão **v** são criadas, os resultados da consulta são inconsistentes quando a classificação é usada para consultar a tabela **test** em uma subconsulta.

```
CREATE TABLE test(a serial ,b int);
INSERT INTO test(b) VALUES(1);
INSERT INTO test(b) SELECT b FROM test;
...
INSERT INTO test(b) SELECT b FROM test;
CREATE VIEW v as SELECT * FROM test ORDER BY a;
```

SQL de problema:

```
select * from v limit 1;
 a | b
---+---
 3 | 1
(1 row)

select * from (select * from test order by a) limit 10;
 a  | b
----+---
 14 | 1
(1 row)

select * from test order by a limit 10;
 a | b
---+---
 1 | 1
(1 row)
```

Análise:

ORDER BY é inválida para subvisões e subconsultas.

Solução:

Não é aconselhável usar **ORDER BY** em subvisões e subconsultas. Para garantir que os resultados estejam em ordem, use **ORDER BY** na consulta mais externa.

LIMIT em subconsultas

Cenário: quando **LIMIT** é usada em uma subconsulta, os dois resultados da consulta são inconsistentes.

```
select * from (select a from test limit 1 ) order by 1;
 a
---
 5
(1 row)

select * from (select a from test limit 1 ) order by 1;
 a
---
 1
(1 row)
```

Análise:

A LIMIT na subconsulta faz com que resultados aleatórios sejam obtidos.

Solução:

Para garantir a estabilidade do resultado final da consulta, não use **LIMIT** em subconsultas.

Usar String_agg

Cenário: quando **string_agg** é usada para consultar a tabela **employee**, os resultados da consulta são inconsistentes.

```
select * from employee;
 empno | ename  |   job   | mgr  |      hiredate       |  sal  | comm | deptno
-------+--------+---------+------+---------------------+-------+------+--------
  7654 | MARTIN | SALEMAN | 7698 | 2022-11-08 00:00:00 | 12000 | 1400 |     30
  7566 | JONES  | MANAGER | 7839 | 2022-11-08 00:00:00 | 32000 |    0 |     20
  7499 | ALLEN  | SALEMAN | 7698 | 2022-11-08 00:00:00 | 16000 |  300 |     30
(3 rows)

select count(*) from (select deptno, string_agg(ename, ',') from employee group
by deptno) t1, (select deptno, string_agg(ename, ',') from employee group by
deptno) t2 where t1.string_agg = t2.string_agg;
 count
-------
     2
(1 row)

select count(*) from (select deptno, string_agg(ename, ',') from employee group
by deptno) t1, (select deptno, string_agg(ename, ',') from employee group by
deptno) t2 where t1.string_agg = t2.string_agg;
 count
-------
     1
(1 row)
```

Análise:

A função **string_agg** é usada para concatenar dados em um grupo em uma linha. No entanto, se você usar **string_agg(ename, ',')**, a ordem dos resultados concatenados precisa ser especificada. Por exemplo, na instrução anterior, **select deptno, string_agg(ename, ',') from employee group by deptno;**

pode produzir uma das seguintes:

```
30 | ALLEN,MARTIN
```

Ou:


```
30 |MARTIN,ALLEN
```

No cenário anterior, o resultado da subconsulta **t1** pode ser diferente da subconsulta **t2** quando deptno é **30**.

Solução:

Adicione **ORDER BY** a **String_agg** para garantir que os dados sejam concatenados em sequência.

```
select count(*) from (select deptno, string_agg(ename, ',' order by ename desc)
from employee group by deptno) t1 ,(select deptno, string_agg(ename, ',' order by
ename desc) from employee group by deptno) t2 where t1.string_agg = t2.string_agg;
```

Modo de compatibilidade de banco de dados

Cenário: os resultados da consulta de cadeias de caracteres vazias no banco de dados são inconsistentes.

banco de dados1 (compatível com TD):

```
td=# select '' is null;
 isnull
--------
 f
(1 row)
```

banco de dados2 (compatível com ORA):

```
ora=# select '' is null;
 isnull
--------
 t
(1 row)
```

Análise:

Os resultados da consulta de cadeia de caracteres vazia são diferentes porque a sintaxe da cadeia de caracteres vazia é diferente da da cadeia de caracteres nula em compatibilidade de banco de dados diferente.

Atualmente, o GaussDB(DWS) suporta três tipos de compatibilidade de banco de dados: Oracle, TD e MySQL. A sintaxe e o comportamento variam dependendo da compatibilidade. Para obter detalhes sobre as diferenças de compatibilidade, consulte **Diferenças de compatibilidade de sintaxe entre Oracle, Teradata e MySQL**

Bancos de dados em diferentes modos de compatibilidade têm diferentes problemas de compatibilidade. Você pode executar **select datname, datcompatibility from pg_database;** para verificar a compatibilidade do banco de dados.

Solução:

O problema é resolvido quando os modos de compatibilidade dos bancos de dados nos dois ambientes são definidos para o mesmo. O atributo **DBCOMPATIBILITY** de um banco de dados não oferece suporte a **ALTER**. Você só pode especificar o mesmo atributo **DBCOMPATIBILITY** ao criar um banco de dados.

O item de configuração behavior_compat_options para comportamentos de compatibilidade de banco de dados é configurado de forma inconsistente.

Cenário: os resultados do cálculo da função **add_months** são inconsistentes.

banco de dados1:

```
select add_months('2018-02-28',3) from dual;
add_months
---------------------
2018-05-28 00:00:00
(1 row)
```

banco de dados2:

```
select add_months('2018-02-28',3) from dual;
add_months
---------------------
2018-05-31 00:00:00
(1 row)
```

Análise:

Alguns comportamentos variam de acordo com o item de configuração de compatibilidade de banco de dados **behavior_compat_options**. Para obter detalhes sobre as opções de parâmetro, consulte **behavior_compat_options**.

O **end_month_calculate** em **behavior_compat_options** controla a lógica de cálculo da função **add_months**. Se este parâmetro for especificado e o **Day** de **param1** indicar o último dia de um mês menor que **result**, o **Day** no resultado do cálculo será igual ao **result**.

Solução:

O parâmetro **behavior_compat_options** deve ser configurado consistentemente. Este parâmetro é do tipo **USERSET** e pode ser definido no nível da sessão ou modificado no nível do cluster.

Os atributos da função definida pelo usuário não estão configurados corretamente.

Cenário: quando a função personalizada **get_count()** é invocada, os resultados são inconsistentes.

```
CREATE FUNCTION get_count() returns int
SHIPPABLE
as $$
declare
    result int;
begin
result = (select count(*) from test); --test table is a hash table.
    return result;
end;
$$
language plpgsql;
```

Chame essa função.

```
SELECT get_count();
 get_count
-----------
      2106
(1 row)

SELECT get_count() FROM t_src;
 get_count
-----------
      1032
(1 row)
```

Análise:

Esta função especifica o atributo **SHIPPABLE**. Quando um plano é gerado, a função o empurra para DNs para execução. A tabela de teste definida na função é uma tabela de hash. Portanto, cada DN tem apenas parte dos dados da tabela, o resultado retornado por **select count(*) from test;** não é o resultado de dados completos na tabela de teste. O resultado esperado muda depois que **from** é adicionado.

Solução:

Use um dos seguintes métodos (o primeiro método é recomendado):

1. Altere a função para não empurrar para baixo: **ALTER FUNCTION get_count() not shippable;**
2. Altere a tabela usada na função para uma tabela de replicação. Desta forma, os dados completos da tabela são armazenados em cada DN. Mesmo que o plano seja empurrado para DNs para execução, o conjunto de resultados será o esperado.

Usar a tabela não registrada

Cenário:

Depois que uma tabela não registrada é usada e o cluster é reiniciado, o conjunto de resultados de consulta associado é anormal e alguns dados estão ausentes na tabela não registrada.

Análise:

Se **max_query_retry_times** for definido como **0** e a palavra-chave **UNLOGGED** for especificada durante a criação da tabela, a tabela criada será uma tabela não registrada. Os dados gravados em tabelas não registradas não são gravados no log de gravação antecipada, o que os torna consideravelmente mais rápidos do que as tabelas comuns. No entanto, uma tabela não registrada é automaticamente truncada após uma falha ou desligamento impuro, incorrendo em riscos de perda de dados. O conteúdo de uma tabela não registrada também não é replicado para servidores em espera. Quaisquer índices criados em uma tabela não registrada também não são registrados automaticamente. Se o cluster for reiniciado inesperadamente (reinicialização do processo, falha do nó ou reinicialização do cluster), alguns dados na memória não serão liberados nos discos em tempo hábil e alguns dados serão perdidos, fazendo com que o conjunto de resultados seja anormal.

Solução:

A segurança das tabelas não registradas não pode ser assegurada se o cluster for defeituoso. Na maioria dos casos, as tabelas não registradas são usadas apenas como tabelas temporárias. Se um cluster estiver com defeito, você precisará reconstruir a tabela não registrada ou fazer backup dos dados e importá-los para o banco de dados novamente para garantir que os dados estejam normais.

3.19 Em quais catálogos do sistema a operação VACUUM FULL não pode ser executada?

VACUUM FULL pode ser executada em todos os catálogos do sistema de GaussDB(DWS). No entanto, durante o processo, os bloqueios de nível 8 serão impostos aos catálogos do sistema e os serviços que envolvem esses catálogos do sistema serão bloqueados.

As sugestões são baseadas em versões do banco de dados:

8.1.3 e versões posteriores

- Para clusters da versão 8.1.3 ou posterior, o **AUTO VACUUM** é ativada por padrão (controlado pelo parâmetro **autovacuum**). Depois de definir o parâmetro, o sistema executa automaticamente **VACUUM FULL** em todos os catálogos do sistema e tabelas de armazenamento de linha.
  - Se o valor de **autovacuum_max_workers** for **0**, nem nos catálogos do sistema nem nas tabelas ordinárias **VACUUM FULL** será executada automaticamente.
  - Se **autovacuum** estiver definido como **off**, **VACUUM FULL** será executada automaticamente em tabelas comuns, mas não em catálogos do sistema.
- Isso se aplica somente a tabelas de armazenamento de linha. Para acionar automaticamente **VACUUM** para tabelas de armazenamento de colunas, é necessário configurar tarefas de agendamento inteligente no console de gerenciamento. Para obter detalhes, consulte **Plano de O&M**.

8.1.1 e versões anteriores

1. Reformar **VACUUM FULL** nos seguintes catálogos do sistema afeta todos os serviços. Execute essa operação em uma janela de tempo ocioso ou quando os serviços forem interrompidos.
   - **pg_statistic** (Informação estatística. É aconselhável não limpá-lo porque afeta o desempenho da consulta de serviço.)
   - pg_attribute
   - pgxc_class
   - pg_type
   - pg_depend
   - pg_class
   - pg_index
   - pg_proc
   - pg_partition
   - pg_object
   - pg_shdepend
2. Os seguintes catálogos do sistema afetam o monitoramento de recursos e as interfaces de consulta de tamanho de tabela, mas não afetam outros serviços.
   - gs_wlm_user_resource_history
   - gs_wlm_session_info
   - gs_wlm_instance_history
   - gs_respool_resource_history
   - pg_relfilenode_size
3. Outros catálogos do sistema não ocupam espaço e não precisam ser limpos.
4. Durante a O&M de rotina, é aconselhável monitorar os tamanhos desses catálogos do sistema e coletar estatísticas todas as semanas. Se o espaço precisar ser recuperado, limpe o espaço com base nos tamanhos das tabelas do sistema.

   A instrução é a seguinte:

   ```
   SELECT c.oid,c.relname, c.relkind, pg_relation_size(c.oid) AS size  FROM
   pg_class c WHERE c.relkind IN ('r') AND c.oid <16385 ORDER BY size DESC;
   ```


3.20 Em quais cenários uma instrução fica "idle in transaction"?

Quando as informações SQL do usuário são consultadas na exibição **PGXC_STAT_ACTIVITY**, a coluna **state** no resultado da consulta às vezes exibe **idle in transaction**. **idle in transaction** indica que o back-end está em uma transação, mas nenhuma instrução está sendo executada. Esse status indica que uma instrução foi executada. Portanto, o valor de query_id é 0, mas a transação não foi confirmada ou revertida. As instruções nesse estado foram executadas e não ocupam recursos de CPU e I/O, mas ocupam recursos de conexão, como conexões e conexões simultâneas.

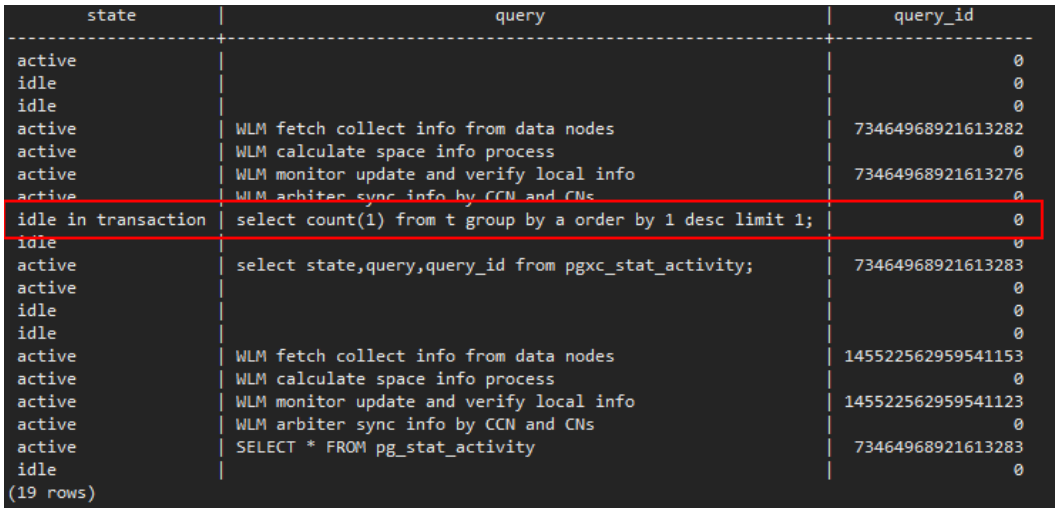
Se uma instrução estiver no estado **idle in transaction**, corrija a falha referindo-se aos seguintes cenários e soluções comuns:

Cenário 1: uma transação é iniciada, mas não comprometida, e a instrução está no estado "idle in transaction"

BEGIN/START TRANSACTION é executada manualmente para iniciar uma transação. Depois que as instruções são executadas, **COMMIT/ROLLBACK** não é executada. Exiba o **PGXC_STAT_ACTIVITY**:

```
SELECT state, query, query_id FROM pgxc_stat_activity;
```

O resultado mostra que a instrução está no estado **idle in transaction**.



state	query	query_id
active		0
idle		0
idle		0
active	WLM fetch collect info from data nodes	73464968921613282
active	WLM calculate space info process	0
active	WLM monitor update and verify local info	73464968921613276
active	WLM arbiter sync info by CCN and CNs	0
idle in transaction	select count(1) from t group by a order by 1 desc limit 1;	0
idle		0
active	select state,query,query_id from pgxc_stat_activity;	73464968921613283
active		0
idle		0
idle		0
active	WLM fetch collect info from data nodes	145522562959541153
active	WLM calculate space info process	0
active	WLM monitor update and verify local info	145522562959541123
active	WLM arbiter sync info by CCN and CNs	0
active	SELECT * FROM pg_stat_activity	73464968921613283
idle		0

(19 rows)

Solução: execute manualmente **COMMIT/ROLLBACK** na transação iniciada.

Cenário 2: após uma instrução DDL em um procedimento armazenado é executado, outros nós do procedimento armazenado está no estado "idle in transaction"

Crie um procedimento armazenado:

```
CREATE OR REPLACE FUNCTION public.test_sleep()
RETURNS void
LANGUAGE plpgsql
AS $$
```



```
BEGIN
    truncate t1;
    truncate t2;
    EXECUTE IMMEDIATE 'select pg_sleep(6)';
    RETURN;
END$$;
```

Exiba a visão **PGXC_STAT_ACTIVITY**:

```
SELECT coorname,pid,query_id,state,query,usename FROM pgxc_stat_activity WHERE
usename='jack';
```

O resultado mostra que **truncate t2** está no estado **idle in transaction** e **coorname** é c**coordinator2**. Isso indica que a instrução foi executada em **cn2** e o procedimento armazenado está executando a próxima instrução.

```
   coorname    |       pid       |     query_id      |        state        |        query         | usename
---------------+-----------------+-------------------+---------------------+----------------------+---------
 coordinator1 | 139767124588288 | 73464968921614213 | active              | select test_sleep(); | jack
 coordinator2 | 140055318353664 |                 0 | idle in transaction | truncate t2          | jack
(2 rows)
```

Solução: este problema é causado pela execução lenta do procedimento armazenado. Aguarde até que a execução do procedimento armazenado seja concluída. Você também pode otimizar as instruções que são executadas lentamente no procedimento armazenado.

Cenário 3: um grande número de instruções SAVEPOINT/RELEASE estão no estado "idle in transaction" (versões de cluster anteriores a 8.1.0)

Exiba a visão **PGXC_STAT_ACTIVITY**:

```
SELECT coorname,pid,query_id,state,query,usename FROM pgxc_stat_activity WHERE
usename='jack';
```

O resultado mostra que a instrução SAVEPOINT/RELEASE está no estado **idle in transaction**.

```
   coorname    |       pid       |     query_id      |        state        |        query         | usename
---------------+-----------------+-------------------+---------------------+----------------------+---------
 coordinator1 | 140127877723904 | 77687093572141691 | active              | select test_sleep1(); | jack
 coordinator2 | 139773127153408 |                 0 | idle in transaction | release s1           | jack
 coordinator3 | 140193352906496 |                 0 | idle in transaction | release s1           | jack
(3 rows)
```

Solução:

As instruções **SAVEPOINT** e **RELEASE** são geradas automaticamente pelo sistema quando um procedimento armazenado com **EXCEPTION** é executado. Em versões posteriores à 8.1.0, o **SAVEPOINT** não é entregue aos CNs. Os procedimentos armazenados do GaussDB(DWS) com **EXCEPTION** são implementados com base em subtransações, o mapeamento é o seguinte:

```
begin
    (Savepoint s1)
    DDL/DML
exception
    (Rollback to s1)
    (Release s1)
...
end
```

Se houver **EXCEPTION** em um procedimento armazenado quando ele for iniciado, uma subtransação será iniciada. Se houver uma exceção durante a execução, a transação atual é revertida e a exceção é tratada; se não houver uma exceção, a subtransação é confirmada.

Esse problema pode ocorrer quando há muitos desses procedimentos armazenados e os procedimentos armazenados são aninhados. Semelhante ao cenário 2, você só tem que esperar depois que todo o procedimento armazenado é executado. Se houver um grande número de mensagens **RELEAS**, o procedimento armazenado aciona várias exceções. Neste caso, você tem que analisar se a lógica do procedimento armazenado é adequada.

3.21 Como o GaussDB(DWS) implementa a conversão de linha para coluna e de coluna para linha?

Esta seção descreve como usar instruções SQL para converter linhas em colunas e converter colunas em linhas no GaussDB(DWS).

Cenário

Use uma tabela de pontuação do aluno como exemplo:

Os professores registram a pontuação de cada matéria de cada aluno em uma tabela, mas os alunos se importam apenas com suas próprias pontuações. Um aluno precisa usar a conversão de linha em coluna para visualizar suas pontuações de todas as disciplinas. Se o professor de um assunto quer ver as feridas de todos os alunos desse assunto, o professor precisa usar a conversão de coluna para linha.

A figura a seguir mostra a conversão de linha para coluna e de coluna para linha.

Figura 3-1 Diagrama

name	subject	score
matu	math	75
matu	physics	90
matu	literature	85
lily	math	95
lily	physics	80
lily	literature	92
jack	math	90
jack	physics	95
jack	literature	95

row to column

column to row

name	math	physics	literature
matu	75	90	85
lily	95	80	92
jack	90	95	95

- Conversão de linhas para coluna

  Converta várias linhas de dados em uma linha ou converta uma coluna de dados em várias colunas.
- Conversão de coluna para linha

  Converta uma linha de dados em várias linhas ou converta várias colunas de dados em uma coluna.

Exemplo

- Crie uma tabela de armazenamento de linha **students_info** e insira dados na tabela.

```
CREATE TABLE students_info(name varchar(20),subject varchar(100),score
bigint) distribute by hash(name);
INSERT INTO students_info VALUES('lily','math',95);
INSERT INTO students_info VALUES('lily','physics',80);
```



```
INSERT INTO students_info VALUES('lily','literature',92);
INSERT INTO students_info VALUES('matu','math',75);
INSERT INTO students_info VALUES('matu','physics',90);
INSERT INTO students_info VALUES('matu','literature',85);
INSERT INTO students_info VALUES('jack','math',90);
INSERT INTO students_info VALUES('jack','physics',95);
INSERT INTO students_info VALUES('jack','literature',95);
```

Exiba informações sobre a tabela **students_info**.

```
SELECT * FROM students_info;
 name |  subject   | score
------+------------+-------
 matu | math       |    75
 matu | physics    |    90
 matu | literature |    85
 lily | math       |    95
 lily | physics    |    80
 lily | literature |    92
 jack | math       |    90
 jack | physics    |    95
 jack | literature |    95
```

- Crie uma tabela de armazenamento de colunas **students_info1** e insira dados na tabela.

```
CREATE TABLE students_info1(name varchar(20), math bigint, physics bigint,
literature bigint) with (orientation = column) distribute by hash(name);
INSERT INTO students_info1 VALUES('lily',95,80,92);
INSERT INTO students_info1 VALUES('matu',75,90,85);
INSERT INTO students_info1 VALUES('jack',90,95,95);
```

Exiba informações sobre a tabela **students_info1**.

```
SELECT * FROM students_info1;
 name | math | physics | literature
------+------+---------+------------
 matu |   75 |      90 |         85
 lily |   95 |      80 |         92
 jack |   90 |      95 |         95
(3 rows)
```

Conversão estática de linha para coluna

A conversão estática de linha para coluna exige que você especifique manualmente os nomes das colunas usando os valores fornecidos. Se nenhum valor for dado a uma coluna, o valor padrão **0** será atribuído à coluna.

```
SELECT name,
sum(case when subject='math' then score else 0 end) as math,
sum(case when subject='physics' then score else 0 end) as physics,
sum(case when subject='literature' then score else 0 end) as literature FROM
students_info GROUP BY name;
 name | math | physics | literature
------+------+---------+------------
 matu |   75 |      90 |         85
 lily |   95 |      80 |         92
 jack |   90 |      95 |         95
(3 rows)
```

Conversão dinâmica de linha para coluna

Para clusters de 8.1.2 ou posterior, você pode usar **GROUP_CONCAT** para gerar instruções de armazenamento de colunas.

```
SELECT group_concat(concat('sum(IF(subject = ''', subject, ''', score, 0)) AS "',
name, '"'))FROM students_info;
      group_concat

--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
```



```
-----------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-----------------------------------------
------------------------
 sum(IF(subject = 'literature', score, 0)) AS "jack",sum(IF(subject =
'literature', score, 0)) AS "lily",sum(IF(subject = 'literature', score, 0)) AS
"matu",sum(IF(subject = 'math', score, 0)) AS "jack",sum(IF
(subject = 'math', score, 0)) AS "lily",sum(IF(subject = 'math', score, 0)) AS
"matu",sum(IF(subject = 'physics', score, 0)) AS "jack",sum(IF(subject =
'physics', score, 0)) AS "lily",sum(IF(subject = 'physics
', score, 0)) AS "matu"
(1 row)
```

Na 8.1.1 e versões anteriores, você pode usar **LISTAGG** para gerar instruções de armazenamento de colunas.

```
SELECT listagg(concat('sum(case when subject = ''', subject, ''' then score else
0 end) AS "', subject, '"'),',') within GROUP(ORDER BY 1)FROM (select distinct
subject from students_info);

                   listagg

-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-----------------------------------------
--
 sum(case when subject = 'literature' then score else 0 end) AS
"literature",sum(case when subject = 'physics' then score else 0 end) AS
"physics",sum(case when subject = 'math' then score else 0 end) AS "math
"
(1 row)
```

Reconstrua dinamicamente a visualização:

```
CREATE OR REPLACE FUNCTION build_view()
RETURNS VOID
LANGUAGE plpgsql
AS $$ DECLARE
sql text;
rec record;
BEGIN
sql := 'select LISTAGG(
    CONCAT( ''sum(case when subject = ''''''', subject, ''''''' then score else 0
end) AS "'', subject, ''"'' )
    ,'','' ) within group(order by 1) from (select distinct subject from
students_info);';
    EXECUTE sql INTO rec;
    sql := 'drop view if exists get_score';
    EXECUTE sql;
    sql := 'create view get_score as select name, ' || rec.LISTAGG || ' from
students_info group by name';
    EXECUTE sql;
END$$;
```

Reconstrua o banco de dados:

```
CALL build_view();
```

Visualização de consulta:

```
SELECT * FROM get_score;
 name | literature | physics | math
------+------------+---------+------
 matu |         85 |      90 |   75
 lily |         92 |      80 |   95
 jack |         95 |      95 |   90
(3 rows)
```


Conversão de coluna para linha

Use **UNION ALL** para mesclar assuntos (matemática, física e literatura) em uma coluna. O seguinte é um exemplo:

```
SELECT * FROM
(
SELECT name, 'math' AS subject, math AS score FROM students_info1
union all
SELECT name, 'physics' AS subject, physics AS score FROM students_info1
union all
SELECT name, 'literature' AS subject, literature AS score FROM students_info1
)
order by name;
 name |  subject   | score
------+------------+-------
 jack | math       |    90
 jack | physics    |    95
 jack | literature |    95
 lily | math       |    95
 lily | physics    |    80
 lily | literature |    92
 matu | math       |    75
 matu | physics    |    90
 matu | literature |    85
(9 rows)
```

3.22 Quais são as diferenças entre restrições únicas e índices únicos?

- Os conceitos de uma restrição única e um índice único são diferentes.

  Uma restrição única especifica que os valores em uma coluna ou um grupo de colunas são todos exclusivos. Se **DISTRIBUTE BY REPLICATION** não for especificado, a tabela de colunas que contém apenas valores exclusivos deve conter colunas de distribuição.

  Um índice único é usado para garantir a exclusividade de um valor de campo ou a combinação de valores de vários campos. **CREATE UNIQUE INDEX** cria um índice único.

- As funções de uma restrição única e de um índice único são diferentes.

  Restrições são usadas para garantir a integridade dos dados e índices são usados para facilitar a consulta.

- Os usos de uma restrição única e um índice único são diferentes.

  a. Ambas as restrições únicas e índices únicos podem ser usados para garantir a exclusividade dos valores da coluna que podem ser NULL.

  b. Quando uma restrição única é criada, um índice único com o mesmo nome é criado automaticamente. O índice não pode ser excluído separadamente. Quando a restrição é excluída, o índice é automaticamente excluído. Uma restrição única usa um índice único para garantir a exclusividade dos dados. As tabelas de armazenamento de linha do GaussDB(DWS) suportam restrições únicas, mas as tabelas de armazenamento de coluna não.

  c. Um índice único criado é independente e pode ser excluído separadamente. Atualmente no GaussDB(DWS), índices únicos só podem ser criados usando B-Tree.

d. Se você quiser ter uma restrição única e um índice único em uma coluna, e eles podem ser excluídos separadamente, você pode criar um índice único e, em seguida, uma restrição única com o mesmo nome.

e. Se um campo em uma tabela deve ser usado como uma chave estrangeira de outra tabela, o campo deve ter uma restrição única (ou é uma chave primária). Se o campo tiver apenas um índice único, um erro será relatado.

Exemplo: crie um índice composto para duas colunas, que não precisa ser um índice único.

```
CREATE TABLE t (n1 number,n2 number);
CREATE INDEX t_idx ON t(n1,n2);
```

Você pode usar o índice **t_idx** criado no exemplo anterior para criar uma restrição única **t_uk**, que é exclusiva apenas na coluna **n1**. Uma restrição única é mais rigorosa do que um índice único.

```
ALTER TABLE t ADD CONSTRAINT t_uk UNIQUE (n1) USING INDEX t_idx;
```

3.23 What Are the Differences Between Functions and Stored Procedures?

Functions and stored procedures are two common objects in database management systems. They have similarities and differences in implementing specific functions. Understanding their characteristics and application scenarios is important for properly designing the database structure and improving database performance.

Tabela 3-2 Differences between functions and stored procedures

Function	Stored procedures
Both can be used to implement specific functions. Both functions and stored procedures can encapsulate a series of SQL statements to complete certain specific operations.	
Both can receive input parameters and perform corresponding operations based on the parameters.	
The identifier of a function is **FUNCTION**.	The identifier of the stored procedure is **PROCEDURE**.
A function must return a specific value of the specified numeric type.	A stored procedure can have no return value, one return value, or multiple return values. You can use output parameters to return results or directly use the SELECT statement in a stored procedure to return result sets.
Functions are used to return single values, for example, a number calculation result, a string processing result, or a table.	Stored procedures are used for DML operations, for example, inserting, updating, and deleting data in batches.

- **Creating and Invoking a Function**

  Create the **emp** table and insert data into the table. The table data is as follows:


```
SELECT * FROM emp;
 empno | ename |   job    | mgr  |      hiredate       |   sal   |  comm  |
deptno
-------+-------+----------+------+---------------------+---------+--------
+--------
  7369 | SMITH | CLERK    | 7902 | 1980-12-17 00:00:00 |  800.00 |
|     20
  7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 00:00:00 | 1600.00 | 300.00
|     30
  7566 | JONES | MANAGER  | 7839 | 1981-04-02 00:00:00 | 2975.00 |
|     20
  7521 | WARD  | SALESMAN | 7698 | 1981-02-22 00:00:00 | 1250.00 | 500.00
|     30
(4 rows)
```

Create the **emp_comp** function to accept two numbers as input and return the calculated value.

```
CREATE OR REPLACE FUNCTION emp_comp (
    p_sal           NUMBER,
    p_comm          NUMBER
) RETURN NUMBER
IS
BEGIN
    RETURN (p_sal + NVL(p_comm, 0)) * 24;
END;
/
```

Run the **SELECT** command to invoke the function:

```
SELECT ename "Name", sal "Salary", comm "Commission", emp_comp(sal, comm)
"Total Compensation" FROM emp;
 Name  | Salary  | Commission | Total Compensation
-------+---------+------------+--------------------
 SMITH |  800.00 |            |           19200.00
 ALLEN | 1600.00 |     300.00 |           45600.00
 JONES | 2975.00 |            |           71400.00
 WARD  | 1250.00 |     500.00 |           42000.00
(4 rows)
```

- **Creating and Invoking a Stored Procedure**

  Create the **MATCHES** table and insert data into the table. The table data is as follows:

  ```
  SELECT * FROM MATCHES;
   matchno | teamno | playerno | won | lost
  ---------+--------+----------+-----+------
         1 |      1 |        6 |   3 |    1
         7 |      1 |       57 |   3 |    0
         8 |      1 |        8 |   0 |    3
         9 |      2 |       27 |   3 |    2
        11 |      2 |      112 |   2 |    3
  (5 rows)
  ```

  Create the stored procedure **delete_matches** to delete all matches that a specified player participates in.

  ```
  CREATE PROCEDURE delete_matches(IN p_playerno INTEGER)
  AS
  BEGIN
     DELETE FROM MATCHES WHERE playerno = p_playerno;
  END;
  /
  ```

  Invoke the stored procedure **delete_matches**.

  ```
  CALL delete_matches(57);
  ```

  Query the **MATCHES** table again. The returned result indicates that the data of the player whose **playerno** is **57** has been deleted.

  ```
  SELECT * FROM MATCHES;
   matchno | teamno | playerno | won | lost
  ---------+--------+----------+-----+------
  ```



```
      11 |      2 |      112 |   2 |    3
       8 |      1 |        8 |   0 |    3
       1 |      1 |        6 |   3 |    1
       9 |      2 |       27 |   3 |    2
(4 rows)
```


4 Gerenciamento de clusters

4.1 O que fazer se a criação de um cluster do GaussDB(DWS) falhar?

Solução de problemas

Verifique se você tem cota suficiente para criar o cluster.

Suporte técnico

Se a falha não puder ser identificada, envie um tíquete de serviço para relatar o problema: faça login no console de gerenciamento e escolha **Service Tickets** > **Create Service Ticket**.

4.2 Como limpar e recuperar o espaço de armazenamento?

Depois de excluir dados armazenados em armazéns de dados de GaussDB(DWS), dados sujos podem ser gerados possivelmente porque o espaço em disco não é liberado. Isso resulta em desperdício de espaço em disco e deteriora o desempenho de criação e restauração de napshot. O seguinte descreve o impacto sobre o sistema e operação subsequente para limpar o espaço em disco:

Pontos que valem a pena mencionar durante a limpeza e recuperação de espaço de armazenamento:

- Dados desnecessários precisam ser excluídos para liberar o espaço de armazenamento.
- Operações frequentes de leitura e gravação podem afetar o uso adequado do banco de dados. Portanto, é uma boa prática limpar e recuperar o espaço de armazenamento quando não estiver em horários de pico.
- O tempo de limpeza de dados depende dos dados armazenados no banco de dados.

Execute as seguintes etapas para limpar e recuperar o espaço de armazenamento:

1. Conecte-se ao banco de dados. Para obter detalhes, consulte **Métodos de conexão a um cluster**.
2. Execute o seguinte comando para limpar e recuperar o espaço de armazenamento:

VACUUM FULL;

Por padrão, as tabelas nas quais o usuário atual tem a permissão são excluídas. Outras tabelas são puladas.

As seguintes informações são exibidas quando o espaço é limpo:

```
VACUUM
```

NOTA

- **VACUUM FULL** recupera todo o espaço de linha expirado, no entanto, requer um bloqueio exclusivo em cada tabela que está sendo processada e pode levar muito tempo para ser concluído em tabelas de banco de dados grandes e distribuídas. É aconselhável fazer **VACUUM FULL** para tabelas especificadas. Se você quiser fazer **VACUUM FULL** para todo o banco de dados, é aconselhável fazê-lo durante a manutenção do banco de dados.
- A informação estatística perder-se-á se utilizar o parâmetro **FULL**. Para coletar as estatísticas, adicione a palavra-chave **ANALYZE**, por exemplo, **VACUUM FULL ANALYZE;**.

  Para obter mais informações sobre **VACUUM**, consulte **VACUUM** na *Referência de sintaxe SQL*.

4.3 É possível alternar uns nós de cluster para outra região após a compra?

Não. Os nós de cluster não podem ser usados entre regiões.

Em vez disso, cancele o pedido na região original, coloque um novo pedido na região desejada e crie um cluster.

4.4 Por que o armazenamento usado encolheu após a expansão?

Possíveis causas

Se você não executar o **VACUUM** para limpar e recuperar o espaço de armazenamento antes da ampliação, os dados excluídos do GaussDB (DWS) podem não liberar o espaço em disco ocupado.

Durante o dimensionamento, o sistema redistribui os dados porque o volume de dados de serviço nos nós originais é significativamente maior do que nos nós recém-adicionados. Quando a redistribuição é iniciada, o sistema executa automaticamente o **VACUUM** para liberar o espaço de armazenamento. Isso causa uma grande queda na capacidade.

Procedimento de tratamento

É aconselhável limpar e recuperar periodicamente o espaço de armazenamento executando **VACUUM FULL** para evitar a expansão de dados.

Se o espaço de armazenamento usado ainda for grande após executar o **VACUUM FULL**, analise se o flavor de cluster existente atende aos requisitos de serviço. Se não, expanda o cluster.

4.5 Como exibir métricas de nós (uso de CPU, memória e disco)?

Você pode exibir a capacidade usada de uma CPU de cluster, memória e discos no console de gerenciamento do Cloud Eye. Execute as seguintes etapas para exibir as informações:

Passo 1 Faça login no console do GaussDB(DWS) e clique em **Viewing Metric** ao lado de um cluster.

Passo 2 Clique em  para retornar à página **Cloud Service Monitoring**, alterne para a página **Data Warehouse Node** e clique em **View Metric** à direita do nó correspondente para exibir o uso do disco.

----Fim

4.6 O GaussDB (DWS) suporta um nó único para um ambiente de aprendizagem?

Sim. No GaussDB(DWS), você pode criar um cluster de armazém de dados híbrido no modo autônomo. Se o nome do nó selecionado contiver **h1** (por exemplo, **dwsx2.h1.xlarge.2.c6**), o armazém de dados híbrido suportará somente a implementação autônoma, que não fornece recursos de HA. O custo de armazenamento pode ser reduzido pela metade. Um armazém de dados autônomo pode ser restaurado pela reconstrução automática do ECS, e sua confiabilidade de dados é garantida pelo mecanismo multi-cópia do EVS. É menos caro do que outros flavors e é uma boa escolha para serviços leves.

4.7 O GaussDB (DWS) suporta o BMS?

Sim. Você pode enviar um tíquete de serviço para solicitar flavors do BMS. O GaussDB (DWS) usa ECSs apenas na HUAWEI CLOUD por padrão.

4.8 Como é calculado o espaço em disco ou a capacidade do GaussDB(DWS)?

1. Capacidade total do disco do GaussDB(DWS): para um cluster de três nós, se cada nó tiver 320 GB, a capacidade total será de 960 GB. Quando 1 GB de dados são armazenados, o GaussDB(DWS) armazena 1 GB de dados em dois nós devido à duplicação, um mecanismo de segurança, ocupando assim um total de 2 GB de espaço. Como resultado, mais de 2 GB de espaço é ocupado se metadados e índices forem calculados. Portanto, um cluster de três nós com capacidade total de 960 GB pode armazenar dados de 480 GB. Esse mecanismo garante a segurança dos dados.

   Quando você compra nós no console, você é cobrado pela capacidade disponível de um nó. Por exemplo, o espaço atual de **dws.m3.xlarge** é 320 GB e o espaço disponível exibido é 160 GB, o espaço pelo qual você será cobrado.

2. Verifique o uso do disco de um único nó.

   Da mesma forma, se a capacidade total for de 960 GB e houver três nós de dados, a capacidade de disco de cada nó será de 320 GB.

Faça logon no console do Gauss(DWS) e escolha **Monitoring** > **Node Monitoring** > **Overview** para exibir o uso de discos e outros recursos em cada nó.

NOTA

- O espaço em disco exibido na página **Node Management** é a capacidade total de todos os discos, ou seja, discos do sistema e discos de dados, no cluster de armazém de dados. O espaço em disco exibido na página **Overview** é apenas o espaço disponível para armazenar dados de tabela no cluster. Além disso, as tabelas no cluster de armazém de dados têm cópias de backup, essas cópias também ocupam o armazenamento em disco.
- Se o cluster for somente leitura e um alarme de espaço em disco insuficiente for gerado, expanda a capacidade do cluster seguindo as instruções fornecidas em **Expansão de um cluster**.

4.9 Quais são os bancos de dados gaussdb e postgres do GaussDB (DWS)?

Os bancos de dados **gaussdb** e **postgres** são bancos de dados embutidos do GaussDB(DWS). Você pode criar esquemas e tabelas neles. No entanto, é aconselhável recriar um banco de dados e criar esquemas e tabelas no novo banco de dados.

4.10 Como configurar o número máximo de sessões ao adicionar uma regra de alarme no Cloud Eye?

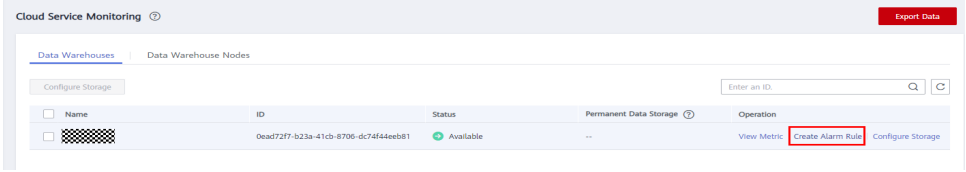
Depois de se conectar a um banco de dados, execute a seguinte instrução SQL para verificar o número máximo de sessões concorrentes globalmente:

```
show max_active_statements;
```

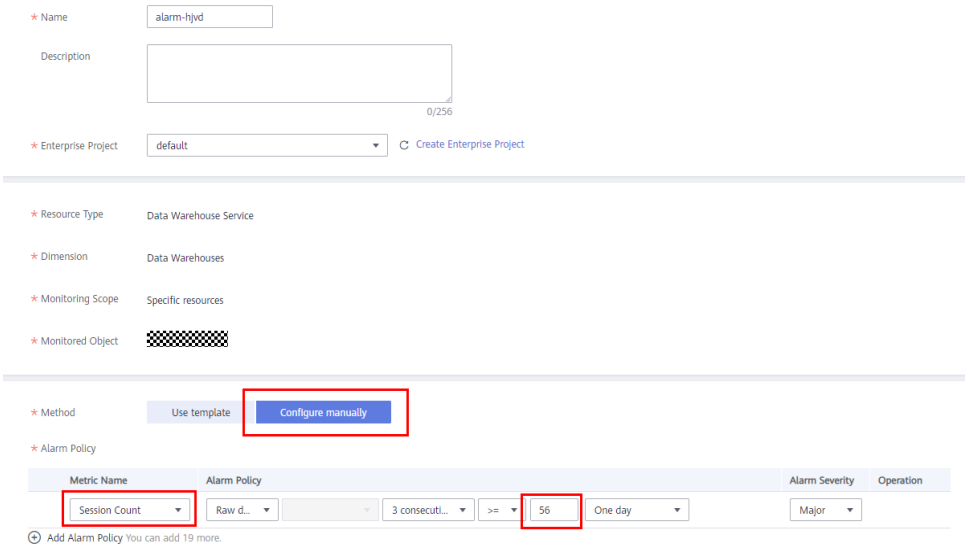
Acesse o console do Cloud Eye e defina o limite para 70% a 80% do valor obtido. Por exemplo, se o valor de **max_active_statements** for **80**, defina o limite como **56** (80 x 70%).

Procedimento:

1. No console de gerenciamento do GaussDB(DWS), escolha **Clusters** > **Dedicated Clusters**.
2. Clique em **View Metric** na coluna **Operation** do cluster de destino para acessar o console do Cloud Eye.
3. Clique em no canto superior esquerdo na página exibida e clique em **Create Alarm Rule** do cluster de destino.



4. Defina **Method** para **Configure manually**, **Metric Name** para **Session Count**, **Alarm Policy** para **56** e **Alarm Severity** para **Major**. Em seguida, clique em **Create**.



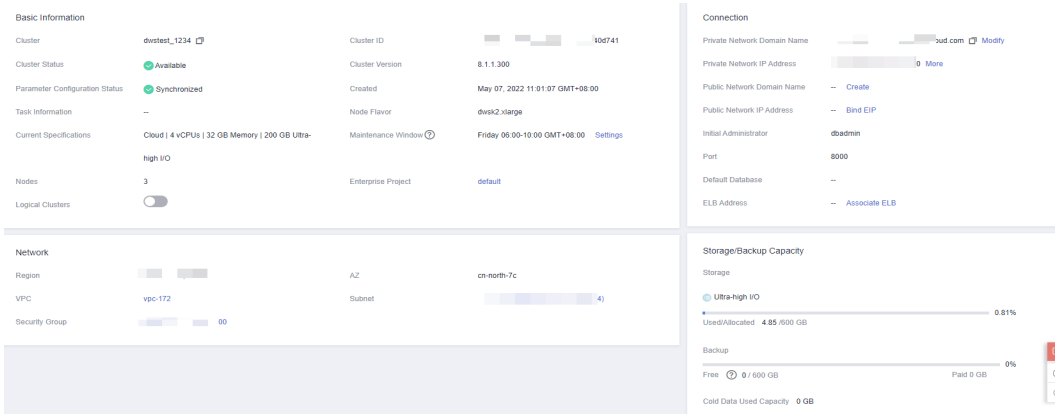
4.11 Como determinar se um cluster usa a arquitetura x86 ou ARM?

Procedimento

Passo 1 Faça logon no console de gerenciamento do GaussDB(DWS).

Passo 2 Escolha **Clusters** > **Dedicated Cluster**. Todos os clusters serão exibidos por padrão.

Passo 3 Na lista de clusters, clique no nome de um cluster para acessar a página **Cluster Information**. Na área **Basic Information**, verifique as especificações do nó do cluster.



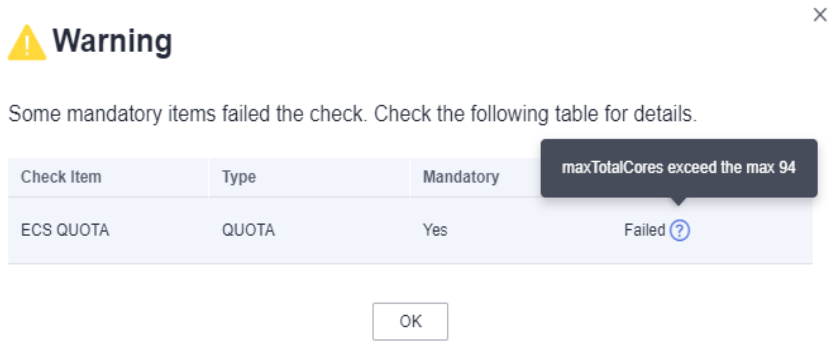
Passo 4 Determine a arquitetura do cluster com base nas especificações do nó.

----Fim

4.12 O que fazer se a verificação de expansão falhar?

Sintoma

Depois de clicar em **OK** durante a expansão ou a adição de nós ociosos, uma mensagem de aviso é exibida e você não pode ir para a próxima etapa.



Análise de causa

Antes de enviar uma tarefa de expansão, o sistema verifica itens como cotas de recursos e permissões do IAM. Itens anormais impedirão que as tarefas de expansão sejam enviadas para evitar falhas de expansão.

Solução

- Se a verificação da cota falhar, verifique se a cota do recurso é suficiente. Se os nós disponíveis forem insuficientes, clique em **Increase Quota** para enviar uma ordem de serviço para aumentar a cota de nó.
- Permissões do IAM: esse problema ocorre quando um usuário tem apenas uma conta do IAM. Nesse caso, permissões como VPC e EVC/BMS precisam ser atribuídas à conta do IAM. Ou, o usuário pode usar a conta principal para executar a expansão.
- Se não houver itens anormais, mas a caixa de diálogo ainda existir, entre em contato com o suporte técnico.

4.13 Quando devo adicionar CNs ou expandir um cluster?

Introdução à simultaneidade do CN

CN é a abreviação de Nó de Coordenador. Um CN é um componente importante do GaussDB(DWS) e está intimamente relacionado aos usuários. Ele fornece interfaces para aplicativos externos, otimiza planos de execução globais, distribui planos de execução para DataNodes e resume e processa resultados de execução. Um CN é uma interface para aplicações externas. A capacidade de simultaneidade do CN determina a simultaneidade do serviço.

A simultaneidade do CN é determinada pelos seguintes parâmetros:

- **max_connections**: especifica o número máximo de conexões concorrentes ao banco de dados. Este parâmetro afeta a capacidade de processamento simultâneo do cluster. O valor padrão depende das especificações do cluster. Para obter detalhes, consulte **Gerenciamento de conexões de banco de dados**.
- **max_active_statements**: especifica o número máximo de jobs concorrentes. Esse parâmetro se aplica a todos os trabalhos em um CN. O valor padrão é **60**, o que indica que um máximo de 60 trabalhos podem ser executados ao mesmo tempo. Outros trabalhos serão colocados na fila.

Adicionar CNs ou dimensionar um cluster?

- Conexões insuficientes: quando um cluster é criado pela primeira vez, o número padrão de CNs no cluster é 3, que pode atender aos requisitos básicos de conexão do cliente. Se o cluster tiver um grande número de solicitações simultâneas e o número de conexões para cada CN for grande ou o uso da CPU de um CN exceder sua capacidade, é aconselhável adicionar CNs. Para obter detalhes, consulte **CNs**.
- Capacidade de armazenamento e desempenho insuficientes: se sua empresa crescer e você tiver requisitos mais altos de capacidade e desempenho de armazenamento, ou se a CPU do cluster for insuficiente, é aconselhável dimensionar o cluster. Para obter detalhes, consulte **Expansão de um cluster**.

Com a expansão dos nós de cluster, mais CNs são necessários para atender aos requisitos de distribuição do GaussDB(DWS). Em suma, adicionar CNs não requer necessariamente escalabilidade de cluster. No entanto, após a expansão do cluster, os CNs podem precisar ser adicionados.

4.14 Quais são os cenários de redimensionar um cluster, alterar o flavor do nó, expandir e reduzir?

Redimensionar um cluster tem um grande impacto em suas cargas de trabalho. É semelhante à migração de um cluster antigo para um novo, com alterações em nós e especificações. Você é aconselhado a realizar operações leves, como expansão, redução e mudança de flavor. A tabela a seguir lista os cenários de aplicação das opções de modificação do cluster.

Tabela 4-1 Opções de modificação de cluster

Opções	Cenário aplicável	Observações
Expansão	Se sua empresa crescer e você tiver requisitos mais altos de capacidade e desempenho de armazenamento, ou se a CPU do cluster for insuficiente, é aconselhável dimensionar o cluster.	Os nós não podem ser adicionados a um armazém de dados híbrido (autônomo).

Opções	Cenário aplicável	Observações
Redução	Durante os horários fora de pico, quando uma grande quantidade de capacidade de cluster está ociosa, você pode reduzir o número de nós para reduzir os custos.	Um armazém de dados híbrido (modo de cluster) não pode ser dimensionado em um cluster autônomo.
Alteração do flavor do nó	Essa opção altera os tipos de cluster (incluindo CPU, memória e outros) para atender aos requisitos de serviço. Não altera o número de nós.	Atualmente, você só pode modificar os flavors de clusters de armazém de dados em nuvem e clusters de armazém de dados em fluxo que usam apenas ECS e recursos do EVS para computação e armazenamento.
Alterar todas as especificações	Você pode redimensionar o cluster quando: ● Seus clusters são clusters de BMS ou não suportam alteração de flavor. ● Você deseja alterar a topologia do cluster em vez de expandir ou reduzir que simplesmente adiciona nós ou exclui nós anel por anel. ● Seu cluster está envelhecido e você deseja alterá-lo para um novo cluster sem migrar dados.	Atualmente, apenas clusters de armazém de dados padrão são suportados.

4.15 How Should I Select from a Small-Flavor Many-Node Cluster and a Large-Flavor Three-Node Cluster with Same CPU Cores and Memory?

- Small-flavor many-node:

  If your data volume is small and you have requirement for node scaling, but you have limited budget, you can select a small-flavor many-node cluster.

  For example, a small-flavor cluster (dwsx2.h.2xlarge.4.c6) with 8 cores and 32 GB memory can provide strong computing capabilities. The cluster has a large number of

nodes and can process high concurrent requests. In this case, you only need to ensure that the network speed between nodes is normal to avoid cluster performance limitation.

- Large-flavor three-node:

  If you have a large amount of data to be processed, have high requirement on computing, and have a high budget, you can select a large-flavor three-node cluster.

  For example, a large-flavor cluster (dws2.m6.8xlarge.8) with 32 cores and 256 GB memory has faster CPU processing capability and larger memory, and can process data more quickly. However, the cluster has limited nodes, which may cause low performance in high-concurrency scenarios.

4.16 Quais são as diferenças entre SSDs em nuvem e SSDs locais?

SSDs em nuvem oferece suporte à expensão. Portanto, você é aconselhado a usá-los. As diferenças entre os SSDs na nuvem e os SSDs locais são as seguintes:

- **SSDs em nuvem**
  - Os SSDs em nuvem usam o EVS como mídia de armazenamento e podem ser expandidos à medida que os dados crescem. Isto é mais flexível.
  - Os SSDs em nuvem não estão vinculados ao flavor do ECS. Portanto, você pode ajustar os flavors dos SSDs em nuvem quando necessário.
- **SSDs locais**
  - Os SSDs locais usam discos locais do ECS como mídia de armazenamento. Eles têm capacidade fixa e desempenho mais alto, mas não podem ser expandidos.
  - Se a capacidade for insuficiente, você terá que adicionar mais nós para aumentar a capacidade. Os flavors de SSDs locais não podem ser ajustados.

4.17 Quais são as diferenças entre armazenamento de dados a quente e armazenamento de dados a frio?

A maior diferença entre armazenamento de dados a quente e armazenamento de dados a frio está na mídia de armazenamento.

- Os dados quentes são frequentemente consultados ou atualizados e têm altos requisitos de tempo de resposta de acesso. Eles são armazenados em **discos de dados DN**.
- Os dados frios não são atualizados e são ocasionalmente consultados e não têm altos requisitos de tempo de resposta de acesso. Eles são armazenados no **OBS**.

Mídias de armazenamento diferentes determinam o custo, o desempenho e os cenários de aplicativos dos dois modos de armazenamento, conforme mostrado na **Tabela 4-2**.

Tabela 4-2 Diferenças entre armazenamento de dados a quente e a frio

Armazenamento	Leitura e gravação	Custo	Capacidade	Cenários de aplicação
Armazenamento a quente	Rápida	Alto	Fixa e restrita	Esse modo é aplicável a cenários em que o volume de dados é limitado e precisa ser lido e atualizado com frequência.
Armazenagem a frio	Lenta	Baixo	Grande e ilimitada	Esse modo é aplicável a cenários como arquivamento de dados. Possui baixo custo e grande capacidade.

4.18 O que devo fazer se o botão Scale-in estiver esmaecido?

Sintoma

Quando um usuário executa uma operação de ajuste de escala, o botão **Scale In** em não está disponível e o usuário não pode prosseguir para a próxima operação de redução.

Possíveis causas

O sistema verifica a elegibilidade do cluster para dimensionamento antes de cada operação. O botão **Scale In** em ficará esmaecido se o cluster não se qualificar.

Solução

Verifique as informações de configuração do cluster e verifique se o dimensionamento atende às seguintes condições:

- O conjunto consiste em anéis de quatro ou cinco hosts cada, com DNs primários, em espera e secundários implementados neles. Um anel de cluster é a menor unidade para dimensionamento, o que requer pelo menos dois anéis. O sistema remove os nós do último anel para o primeiro ao reduzir.
- Os nós removidos não podem conter o componente GTM, CM Server ou CN.
- O status do cluster é **Normal** e nenhuma outra informação de tarefa é exibida.
- A conta de locatário do cluster não pode estar no estado somente leitura, congelado ou restrito.
- O cluster não está no modo de cluster lógico.
- Um cluster anual/mensal a ser reduzido não pode estar no período de carência.

5 Conexões de banco de dados

5.1 Como as aplicações se comunicam com o GaussDB(DWS)?

Para que as aplicações se comuniquem com o GaussDB (DWS), certifique-se de que as redes entre eles estejam conectadas. A tabela a seguir lista cenários comuns de conexão.

Tabela 5-1 Comunicação entre aplicações e GaussDB(DWS)

Cenário		Descrição	Tipo de conexão suportado
Nuvem	**A aplicação de serviço e o GaussDB(DWS) estão na mesma VPC na mesma região**	Dois endereços IP privados na mesma VPC podem se comunicar diretamente entre si.	● gsql ● Data Studio ● JDBC/ODBC Para obter mais modos de conexão, consulte **Métodos de conexão a um cluster**.
	As aplicações de serviço e o GaussDB(DWS) estão em VPCs diferentes na mesma região	Depois que uma conexão de **emparelhamento de VPC** é criada entre duas VPCs, os dois endereços IP privados podem se comunicar diretamente entre si.	
	Aplicações de serviço e GaussDB(DWS) estão em regiões diferentes	Depois que **conexão de nuvem** (CC) é estabelecida entre duas regiões, as duas regiões se comunicam entre si por meio de endereços IP privados.	

Cenário		Descrição	Tipo de conexão suportado
No local e na nuvem	**As aplicações de serviço são implementadas em data centers locais e precisam se comunicar com o GaussDB(DWS).**	• Use o **endereço IP público ou nome de domínio** de GaussDB(DWS) para comunicação. • Use **Direct Connect** (DC) é para comunicação.	

A aplicação de serviço e o GaussDB(DWS) estão na mesma VPC na mesma região

Para garantir baixa latência de serviço, recomenda-se implementar aplicações de serviço e GaussDB(DWS) na mesma região. Por exemplo, se uma aplicação de serviço for implementada em um ECS, recomendamos implementar o cluster de armazém de dados na mesma VPC que o ECS. Desta forma, a aplicação pode se comunicar diretamente com o GaussDB(DWS) através de um endereço IP de intranet. Nesse caso, implemente o cluster de data armazém de dados na mesma região e VPC em que o ECS reside.

Por exemplo, se o ECS for implementado na **CN-Hong Kong**, selecione **CN-Hong Kong** para o cluster de GaussDB(DWS) e assegure-se de que o cluster de GaussDB(DWS) e o ECS estejam ambos em **VPC1**. O endereço IP privado do ECS é **192.168.120.1**, o endereço IP privado do GaussDB(DWS) é **192.168.120.2**. Portanto, eles podem se comunicar uns com os outros através de endereços IP privados.

Os pontos-chave na verificação de comunicação são a regra de saída do ECS e a regra de entrada do GaussDB(DWS). O procedimento de verificação é o seguinte:

Passo 1 Verifique as regras de saída do ECS:

Certifique-se de que a regra de saída do grupo de segurança do ECS permita acesso. Se o acesso não for permitido, consulte as **Regras de configuração do grupo de segurança**.

Action	Protocol & Port	Type	Destination
Allow	All	IPv4	0.0.0.0/0

Passo 2 Verifique as regras de entrada do GaussDB(DWS):

Se nenhum grupo de segurança for configurado quando o GaussDB(DWS) for criado, a regra de entrada padrão permitirá o acesso TCP de todos os endereços IPv4 e da porta 8000. Para garantir a segurança, você também pode permitir apenas um endereço IP. Para obter detalhes, consulte **Como configurar uma lista branca para proteger clusters disponíveis por meio de um endereço IP público?**

Allow	TCP : 8000	IPv4	0.0.0.0/0

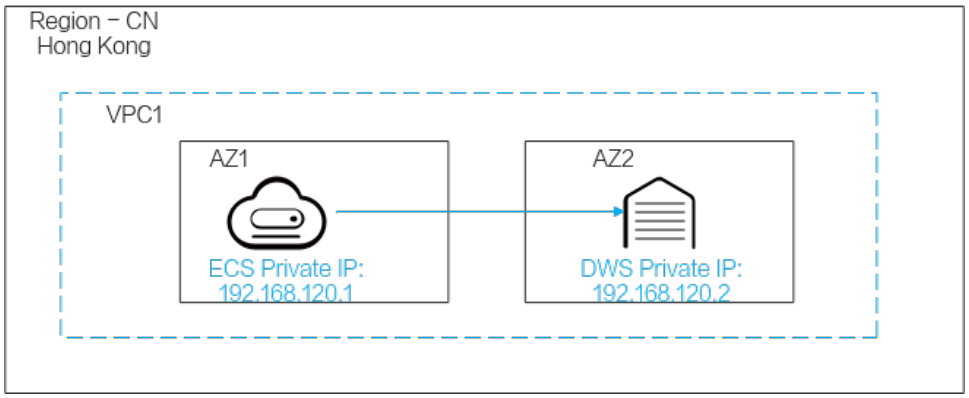
Passo 3 Efetue logon no ECS. Se o endereço IP interno do GaussDB(DWS) pode ser pingado, a conexão de rede é normal. Se o endereço IP não puder ser pingado, verifique a configuração anterior. Se o ECS tiver um firewall, verifique a configuração do firewall.

----Fim

Exemplo de uso do gsql para conexão:

gsql -d gaussdb -h *192.168.120.2* **-p 8000 -U dbadmin -W** *password* **-r**

Figura 5-1 Acesso via endereços IP privados



As aplicações de serviço e o GaussDB(DWS) estão em VPCs diferentes na mesma região

Para garantir baixa latência de serviço, recomenda-se implementar aplicações de serviço e GaussDB(DWS) na mesma região. Por exemplo, se as aplicações de serviço forem implementadas em um ECS, é recomendável implementar o cluster de armazém de dados na mesma VPC que o ECS. Se uma VPC diferente for selecionada para o cluster de armazém de dados, o ECS não poderá se conectar diretamente ao GaussDB(DWS).

Por exemplo, tanto o ECS quanto o GaussDB(DWS) são implementados em **CN-Hong Kong**, mas o ECS está em VPC1 e o GaussDB(DWS) está em VPC2. Nesse caso, você precisa criar uma **conexão de emparelhamento de VPC** entre VPC1 e VPC2 para que o ECS possa acessar o GaussDB(DWS) usando o endereço IP privado de GaussDB(DWS).

Os principais pontos para verificar a comunicação são as regras de saída do ECS, as regras de entrada do GaussDB(DWS) e a conexão de emparelhamento de VPC. O procedimento de verificação é o seguinte:

Passo 1 **Verifique as regras de saída do ECS:**

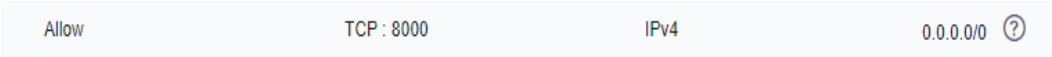
Certifique-se de que a regra de saída do grupo de segurança do ECS permita acesso. Se o acesso não for permitido, consulte as **Regras de configuração do grupo de segurança**.

Action	Protocol & Port	Type	Destination
Allow	All	IPv4	0.0.0.0/0

Passo 2 **Verifique as regras de entrada do GaussDB(DWS):**

Se nenhum grupo de segurança for configurado quando o GaussDB(DWS) for criado, a regra de entrada padrão permitirá o acesso TCP de todos os endereços IPv4 e da porta 8000. Para

garantir a segurança, você também pode permitir apenas um endereço IP. Para obter detalhes, consulte **Como configurar uma lista branca para proteger clusters disponíveis por meio de um endereço IP público?**



Passo 3 Crie uma **conexão de emparelhamento VPC** entre VPC1 onde está o ECS e a VPC2 onde o GaussDB(DWS) está.

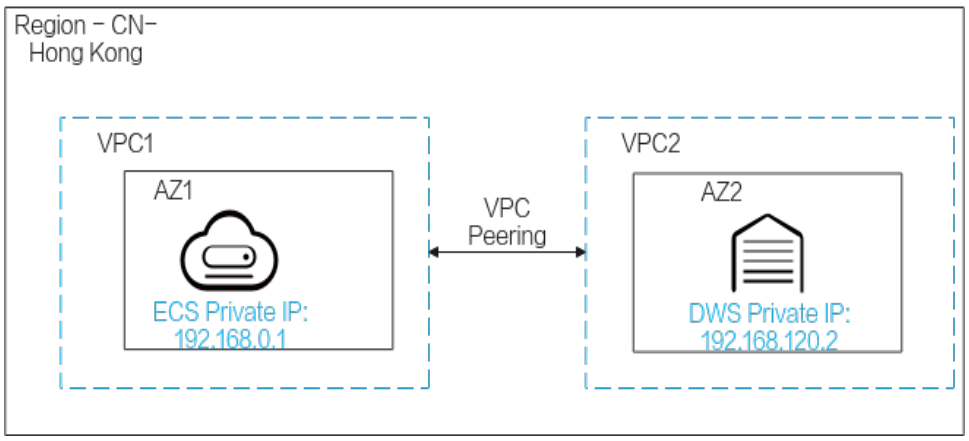
Passo 4 Efetue logon no ECS. Se o endereço IP interno do GaussDB(DWS) pode ser pingado, a conexão de rede é normal. Se o endereço IP não puder ser pingado, verifique a configuração anterior. Se o ECS tiver um firewall, verifique a configuração do firewall.

----Fim

Exemplo de uso do gsql para conexão:

gsql -d gaussdb -h *192.168.120.2* **-p 8000 -U dbadmin -W** *password* **-r**

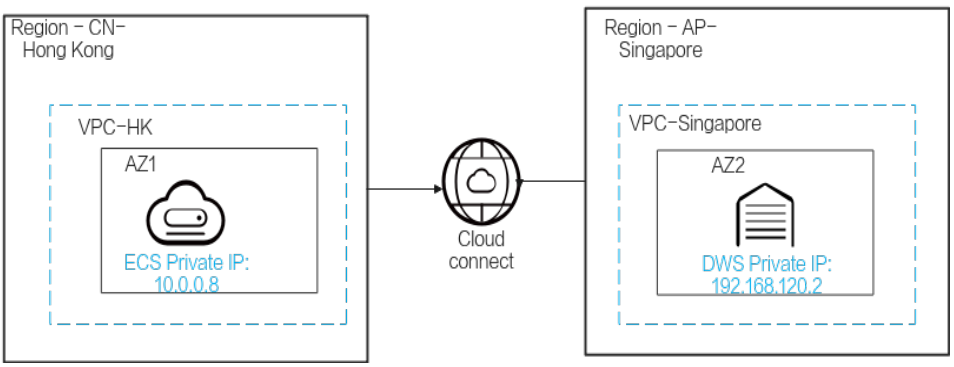
Figura 5-2 Acesso via emparelhamento de VPC



Aplicações de serviço e GaussDB(DWS) estão em regiões diferentes

Se a aplicação de serviço e o GaussDB(DWS) estiverem em regiões diferentes, por exemplo, ECS está em **CN-Hong Kong** e GaussDB (DWS) está em CN **AP-Singapore**, você precisa estabelecer um **Cloud Connect** entre as duas regiões para comunicação.

Figura 5-3 Acesso via Cloud Connect



As aplicações de serviço são implementadas em data centers locais e precisam se comunicar com o GaussDB(DWS).

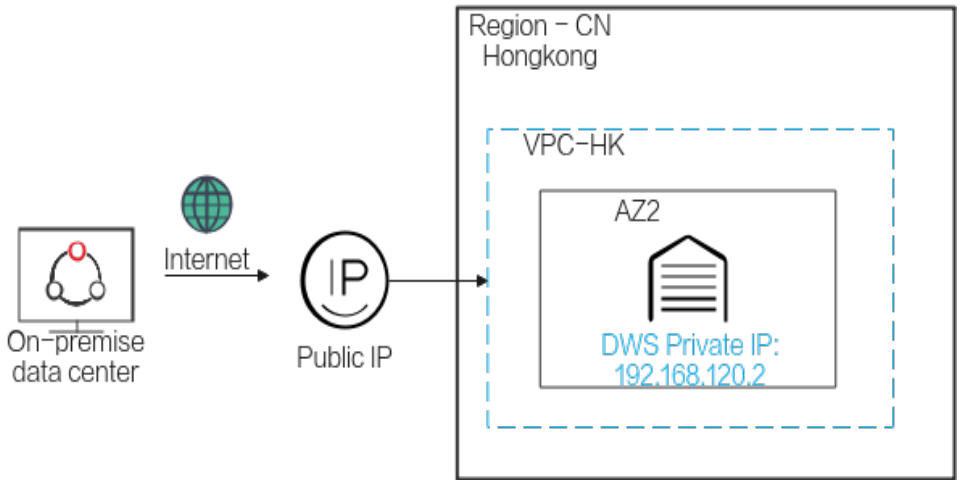
Se as aplicações de serviço não estiverem na nuvem, mas no data center local, elas precisarão se comunicar com o GaussDB(DWS) na nuvem.

- **Cenário 1**: as aplicações de serviço locais se comunicam com o GaussDB(DWS) por meio de endereços IP públicos do GaussDB (DWS).

  Exemplo de usar gsql para a conexão:

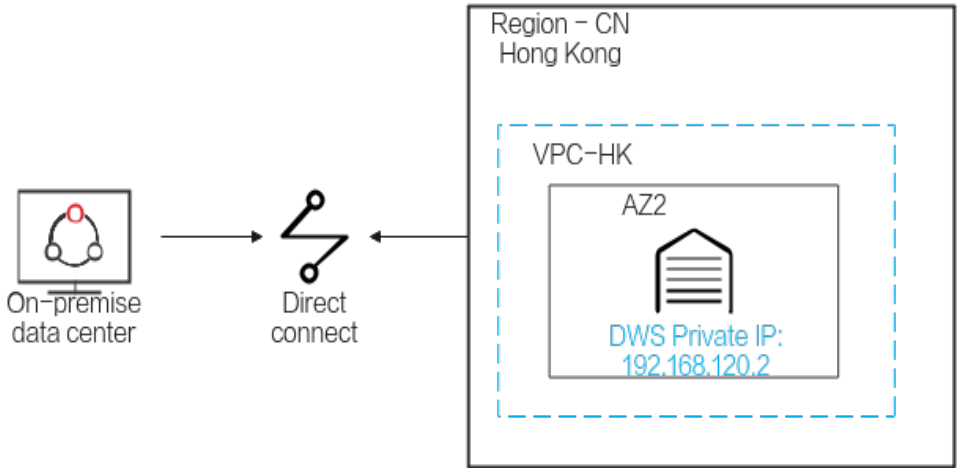
  gsql -d gaussdb -h *public_IP_address* **-p 8000 -U dbadmin -W** *password* **-r**

  Figura 5-4 Acesso via endereços IP públicos



- **Cenário 2**: os serviços locais não podem acessar a rede externa. Nesse caso, o **Direct Connect** é necessário para a comunicação.

  Figura 5-5 Acesso via Direct Connect



5.2 O GaussDB(DWS) oferece suporte a clientes de terceiros e drivers JDBC e ODBC?

Sim, mas os clientes e drivers do GaussDB(DWS) são recomendados. Ao contrário dos clientes e drivers do PostgreSQL de código aberto, os clientes e drivers do GaussDB(DWS) têm duas vantagens principais:

- **Fortalecimento de segurança**: os drivers do PostgreSQL suportam apenas a autenticação MD5, mas os drivers do GaussDB(DWS) suportam SHA256 e MD5.
- **Aprimoramento do tipo de dados**: os drivers do GaussDB(DWS) suportam novos tipos de dados smalldatetime e tinyint.

GaussDB(DWS) suporta clientes do PostgreSQL de código aberto e drivers JDBC e ODBC.

As versões de driver e cliente compatíveis são:

- PostgreSQL psql 9.2.4 ou mais recente
- PostgreSQL JDBC Driver 9.3-1103 ou mais recente
- PSQL ODBC 09.01.0200 ou mais recente

Para obter detalhes sobre como usar JDBC/ODBC para conectar-se ao GaussDB(DWS), consulte **Guia: desenvolvimento baseado em JDBC ou ODBC**.

5.3 É possível conectar-se aos nós de um cluster do GaussDB (DWS) via SSH?

Não, o acesso direto não é suportado. As VMs na camada inferior do GaussDB (DWS) servem como nós de computação para análise de dados. Acesse bancos de dados de cluster usando o endereço de acesso de rede privada ou pública em vez disso.

5.4 O que fazer se não conseguir se conectar a um cluster de armazém de dados?

Solução de problemas

Verificar:

- Se o status do cluster é normal.
- Se o comando de conexão, nome de usuário, senha, endereço IP e porta estão corretos.
- Se o tipo de sistema operacional e a versão do cliente estão corretos.
- Se o cliente está instalado corretamente.

Se a conexão de cluster falhou na nuvem pública, verifique os seguintes itens:

- Se os ECSs estão na mesma AZ, VPC, sub-rede e grupo de segurança que o cluster.
- Se as regras de entrada e saída do grupo de segurança estão corretas.

Se a conexão do cluster falhou através da Internet, confirme os seguintes itens:

- Se a sua rede está conectada à Internet.
- Se o firewall bloqueou o acesso.
- Se você precisa acessar a Internet através de um proxy.

Suporte técnico

Se a falha não puder ser identificada, envie um tíquete de serviço para relatar o problema: faça login no console de gerenciamento e selecione **Service Tickets** > **Create Service Ticket**.:

5.5 Por que não for notificado de falha na desvinculação do EIP quando o GaussDB (DWS) está conectado à Internet?

Depois que o EIP é desacoplado, a rede pode ser desconectada. No entanto, a camada TCP não detecta uma conexão física defeituosa a tempo devido às configurações de manutenção de atividade. Como resultado, os clientes gsql, ODBC e JDBC também não conseguem identificar a falha de rede a tempo.

A duração quando o banco de dados envia a mensagem de desconexão para o cliente depende das configurações de manutenção de atividade. O algoritmo específico para calcular a duração é:

keepalive_time + **keepalive_probes** x **keepalive_intvl**

Os valores de Keepalive afetam a estabilidade da comunicação de rede. Ajuste-os à pressão de serviço e às condições da rede.

No Linux, execute o comando **sysctl** para modificar os seguintes parâmetros:

- net.ipv4.tcp_keepalive_time
- net.ipv4.tcp_keeaplive_probes
- net.ipv4.tcp_keepalive_intvl

Por exemplo, se você quiser alterar o valor de **net.ipv4.tcp_keepalive_time**, execute o seguinte comando para alterá-lo para **120**.

sysctl net.ipv4.tcp_keepalive_time=120

No Windows, modifique as seguintes informações de configuração no registro **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Tcpip\Parameters**:

- KeepAliveTime
- KeepAliveInterval
- TcpMaxDataRetransmissions (equivalente a **tcp_keepalive_probes**)

NOTA

Se não conseguir localizar os parâmetros anteriores no registo**HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Tcpip\Parameters**, adicione estes parâmetros. Abra **Registry Editor**, clique com o botão direito do mouse na área em branco à direita e escolha **Create** > **DWORD (32-bit) Value** para adicionar esses parâmetros.

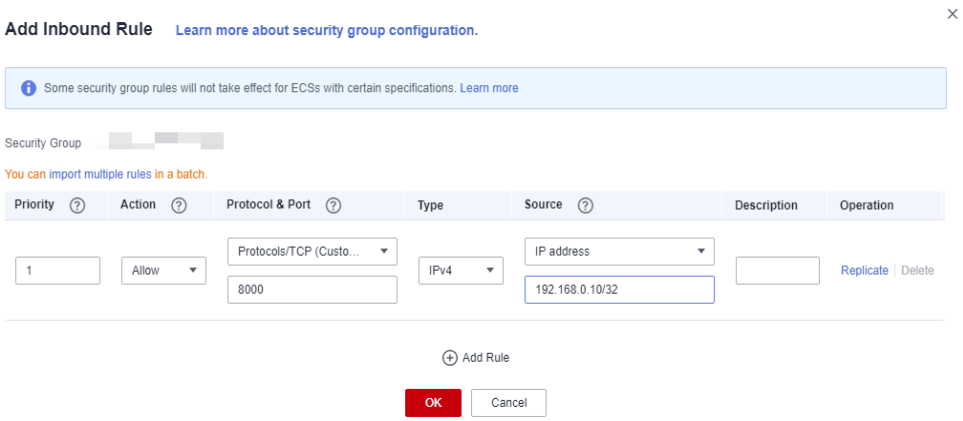
5.6 Como configurar uma lista branca para proteger clusters disponíveis por meio de um endereço IP público?

Você também pode fazer logon no **console de gerenciamento da VPC** para criar manualmente um grupo de segurança. Em seguida, volte para a página de criação de clusters de armazém de dados, clique no botão  ao lado da lista suspensa de **Security Group** para atualizar a página e selecione o novo grupo de segurança.

Para habilitar o cliente de GaussDB(DWS) para se conectar ao cluster, você precisa adicionar uma regra de entrada ao novo grupo de segurança para conceder a permissão de acesso à porta do banco de dados do cluster do GaussDB(DWS).

- **Protocol**: **TCP**
- **Port**: **8000** Use a porta do banco de dados definida ao criar o cluster do GaussDB(DWS). Esta porta é usada para receber conexões de clientes para GaussDB(DWS).
- **Source**: selecione **IP address** e use o endereço IP do host do host do cliente, por exemplo, **192.168.0.10/32**.

Figura 5-6 Adicionar uma regra de entrada



A lista branca será adicionada.

6 Importação e exportação de dados

6.1 Quais são as diferenças entre os formatos de dados suportados pelas tabelas estrangeiras do OBS e GDS?

Os formatos de arquivo suportados pelas tabelas estrangeiras do OBS e GDS são os seguintes:

O OBS suporta formatos de arquivo ORC, TEXT, JSON, CSV, CARBONDATA e PARQUET para importação de dados e formatos de arquivo ORC, CSV e TEXT para exportação de dados. O formato padrão é TEXT.

O GDS suporta os seguintes formatos de arquivo: TEXTO, CSV e FIXED. O formato padrão é TEXT.

6.2 Como importar dados incrementais usando uma tabela estrangeira do OBS?

Quando você usa uma tabela estrangeira do OBS para importar dados, **INSERT** importa os dados para uma tabela física local. Quando os dados do OBS são atualizados, você não precisa executar a instrução **INSERT** novamente. Você pode usar a instrução **MERGE INTO**.

6.3 Como importar dados para o GaussDB(DWS)?

GaussDB (DWS) suporta a importação eficiente de dados de várias fontes de dados. A seguir, lista os modos típicos de importação de dados. Para obter detalhes, consulte **Importação de dados**.

- Importar dados do OBS

  Carregue dados para o OBS e, em seguida, exporte-os para clusters do GaussDB(DWS). Formatos de dados como CSV e TEXT são suportados.

- Inserir dados com instruções **INSERT**

  Use a ferramenta de cliente gsql fornecida pelo GaussDB(DWS) ou o driver JDBC/ODBC para gravar dados no GaussDB(DWS) de aplicações de camada superior. O GaussDB(DWS) suporta operações CRUD completas no nível de transação do banco de

dados. Este é o método mais simples e é aplicável a cenários com pequeno volume de dados e baixa simultaneidade.

- Importar dados do MRS com o MRS como ETL.
- Importar de dados com o comando **COPY FROM STDIN**

  Execute o comando **COPY FROM STDIN** para gravar dados em uma tabela.
- Importar dados de um servidor remoto para GaussDB(DWS) usando GDS

  Use a função de importação de dados GDS fornecida pelo GaussDB(DWS) para importar arquivos de dados de um sistema de arquivos comum (por exemplo, um ECS).
- Migrar dados para GaussDB(DWS) usando CDM

6.4 Quantos dados de serviço um armazém de dados pode armazenar?

Cada nó em um cluster de armazém de dados tem uma capacidade de armazenamento padrão de 1,49 TB, 2,98 TB, 4,47 TB, 160 GB, 1,68 TB ou 13,41 TB. Um cluster pode abrigar de 3 a 256 nós e a capacidade total de armazenamento do cluster se expande proporcionalmente à medida que a escala do cluster cresce.

Para aumentar a confiabilidade, cada nó tem uma cópia, que ocupa metade do espaço de armazenamento.

O sistema do GaussDB (DWS) faz backup de dados e gera índices, arquivos de cache temporários e logs de execução, que ocupam espaço de armazenamento. Portanto, o espaço de armazenamento real de cada nó é cerca de metade da capacidade total de armazenamento.

6.5 Como usar \Copy para importar e exportar dados?

GaussDB(DWS) é um serviço totalmente gerenciado na nuvem. Os usuários não podem fazer logon em segundo plano para importar ou exportar dados usando **COPY**, portanto, a sintaxe **COPY** está desabilitada. É aconselhável armazenar arquivos de dados no OBS e usar tabelas estrangeiras do OBS para importar dados. Se você quiser usar **COPY** para importar e exportar dados, execute as seguintes operações:

1. Coloque o arquivo de dados no cliente.
2. Use o gsql para se conectar ao cluster de destino.
3. Execute o seguinte comando para importar dados. Digite o nome do diretório e o nome do arquivo de dados no cliente e especifique a opção de importação em **with**. O comando é quase o mesmo que o comando **COPY** comum. Você só precisa adicionar uma barra invertida (\) antes do comando. Quando os dados são importados com êxito, nenhuma notificação será exibida.
   ```
   \copy tb_name from '/directory_name/file_name' with(...);
   ```
4. Execute o seguinte comando para exportar dados para um arquivo local. Mantenha as configurações padrão dos parâmetros.
   ```
   \copy table_name to '/directory_name/file_name';
   ```
5. Especifique o parâmetro **copy_option** para exportar dados para um arquivo CSV.
   ```
   \copy table_name to '/directory_name/file_name' CSV;
   ```
6. Use **with** para especificar parâmetros, exportando dados como arquivos CSV que usam barras verticais (|) como delimitadores.
   ```
   \copy table_name to '/directory_name/file_name' with(format 'csv',delimiter
   '|') ;
   ```


6.6 Como implementar a importação de tolerância a falhas entre diferentes bibliotecas de codificação

Para importar dados do banco de dados A (UTF8) para o banco de dados B (GBK), pode haver um erro de incompatibilidade de conjunto de caracteres que faz com que a importação de dados falhe.

Para importar uma pequena quantidade de dados, execute o comando **\COPY**. O procedimento é o seguinte:

Passo 1 Crie bancos de dados A e B. O formato de codificação do banco de dados A é UTF8 e o do banco de dados B é GBK.

```
postgres=> CREATE DATABASE A ENCODING 'UTF8' template = template0;
postgres=> CREATE DATABASE B ENCODING 'GBK' template = template0;
```

Passo 2 Exiba a lista do banco de dados. Você pode visualizar os bancos de dados A e B criados.

```
postgres=> \l
                             List of databases
   Name    |  Owner  | Encoding  | Collate | Ctype | Access privileges
-----------+---------+-----------+---------+-------+-------------------
 a         | dbadmin | UTF8      | C       | C     |
 b         | dbadmin | GBK       | C       | C     |
 gaussdb   | Ruby    | SQL_ASCII | C       | C     |
 postgres  | Ruby    | SQL_ASCII | C       | C     |
 template0 | Ruby    | SQL_ASCII | C       | C     | =c/Ruby          +
           |         |           |         |       | Ruby=CTc/Ruby
 template1 | Ruby    | SQL_ASCII | C       | C     | =c/Ruby          +
           |         |           |         |       | Ruby=CTc/Ruby
 xiaodi    | dbadmin | UTF8      | C       | C     |
(7 rows)
```

Passo 3 Alterne para o banco de dados A e digite a senha do usuário. Crie uma tabela chamada **test01** e insira os dados na tabela.

```
postgres=> \c a
Password for user dbadmin:
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_128_GCM_SHA256, bits: 128)
You are now connected to database "a" as user "dbadmin".

a=> CREATE TABLE test01
 (
     c_customer_sk             integer,
     c_customer_id             char(5),
     c_first_name              char(6),
     c_last_name               char(8)
 )
 with (orientation = column,compression=middle)
 distribute by hash (c_last_name);
CREATE TABLE
a=> INSERT INTO test01(c_customer_sk, c_customer_id, c_first_name) VALUES (3769,
'hello', 'Grace');
INSERT 0 1
a=> INSERT INTO test01 VALUES (456, 'good');
INSERT 0 1
```

Passo 4 Execute o comando **\COPY** para exportar dados da biblioteca UTF8 no formato Unicode para o arquivo **test01.dat**.

```
\copy test01 to '/opt/test01.dat' with (ENCODING 'Unicode');
```

Passo 5 Mude para o banco de dados B e crie uma tabela com o mesmo nome **test01**.

```
a=> \c b
Password for user dbadmin:
```



```
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_128_GCM_SHA256, bits: 128)
You are now connected to database "b" as user "dbadmin".

b=> CREATE TABLE test01
 (
     c_customer_sk             integer,
     c_customer_id             char(5),
     c_first_name              char(6),
     c_last_name               char(8)
 )
 with (orientation = column,compression=middle)
 distribute by hash (c_last_name);
```

Passo 6 Execute o comando **\COPY** para importar o arquivo **test01.dat** para o banco de dados B.

```
\copy test01 from '/opt/test01.dat' with (ENCODING
'Unicode' ,COMPATIBLE_ILLEGAL_CHARS 'true');
```

NOTA

- O parâmetro de tolerância de erro **COMPATIBLE_ILLEGAL_CHARS** especifica que caracteres inválidos são tolerados durante a importação de dados. Caracteres inválidos são convertidos e, em seguida, importados para o banco de dados. Nenhuma mensagem de erro é exibida. A importação não é interrompida.
- O formato BINARY não é suportado. Quando os dados desse formato são importados, o erro "cannot specify bulkload compatibility options in BINARY mode" ocorrerá.
- O parâmetro é válido somente para importação de dados usando a opção **COPY FROM**.

Passo 7 Veja os dados na tabela **test01** no banco de dados B.

```
b=> select * from test01;
 c_customer_sk | c_customer_id | c_first_name | c_last_name
---------------+---------------+--------------+-------------
          3769 | hello         | Grace        |
           456 | good          |              |
(2 rows)
```

Passo 8 Depois que as operações anteriores são executadas, os dados são importados do banco de dados A (UTF8) para o banco de dados B (GBK).

----Fim

6.7 Posso importar e exportar dados de e para o OBS entre regiões?

Não, o GaussDB(DWS) não oferece suporte à importação ou exportação de dados do OBS entre regiões. O cluster do GaussDB(DWS) e o OBS devem estar na mesma região.

Por exemplo, a exportação direta de dados de um cluster do GaussDB(DWS) em Beijing 4 para o OBS em Beijing 1 não é suportada. No entanto, você pode primeiro exportar os dados para o OBS de Beijing 4 e, em seguida, usar o CDM para exportar os dados para o OBS de Beijing 1.

6.8 Como importar dados do GaussDB(DWS)/Oracle/MySQL/SQL Server para o GaussDB(DWS) (migração de banco de dados inteiro)?

Dados heterogêneos podem ser importados para GaussDB(DWS) usando CDM. Você pode migrar um banco de dados Oracle, MySQL, SQL Server ou GaussDB(DWS) inteiro para um

banco de dados GaussDB(DWS). Para obter detalhes, consulte **Criação de um trabalho de migração de banco de dados inteiro**.

Você também pode armazenar dados no OBS e depois despejar os dados no GaussDB(DWS). Para obter detalhes, consulte **Sobre a importação de dados paralelos do OBS**.

6.9 É possível importar dados pela rede pública/externa usando o GDS?

Não. O servidor GDS e o GaussDB (DWS) só podem se comunicar uns com os outros na intranet. Cada DN no cluster do GaussDB (DWS) é usado para conectar-se ao servidor GDS em paralelo para importar uma grande quantidade de dados. O servidor GDS e o cluster devem estar na mesma rede. Se o GDS for implantado em um servidor off-line, o firewall precisa ser habilitado e o cluster precisa de um EIP. No entanto, um cluster pode ser vinculado apenas a um EIP e a importação de dados com vários DNs não pode ser implementada.

6.10 Quais são os fatores que afetam o desempenho de importação do GaussDB(DWS)?

O desempenho de importação do GaussDB(DWS) é afetado pelos seguintes fatores:

1. Especificações do cluster: I/O de disco, taxa de transferência da rede, memória e especificações de CPU
2. Planejamento de serviço: tipo de campos de tabela, compactação e armazenamento de linha ou armazenamento de coluna
3. Armazenamento de dados: cluster local, OBS
4. Modo de importação de dados

7 Conta, senha e permissão

7.1 Como o GaussDB(DWS) implementa o isolamento da carga de trabalho?

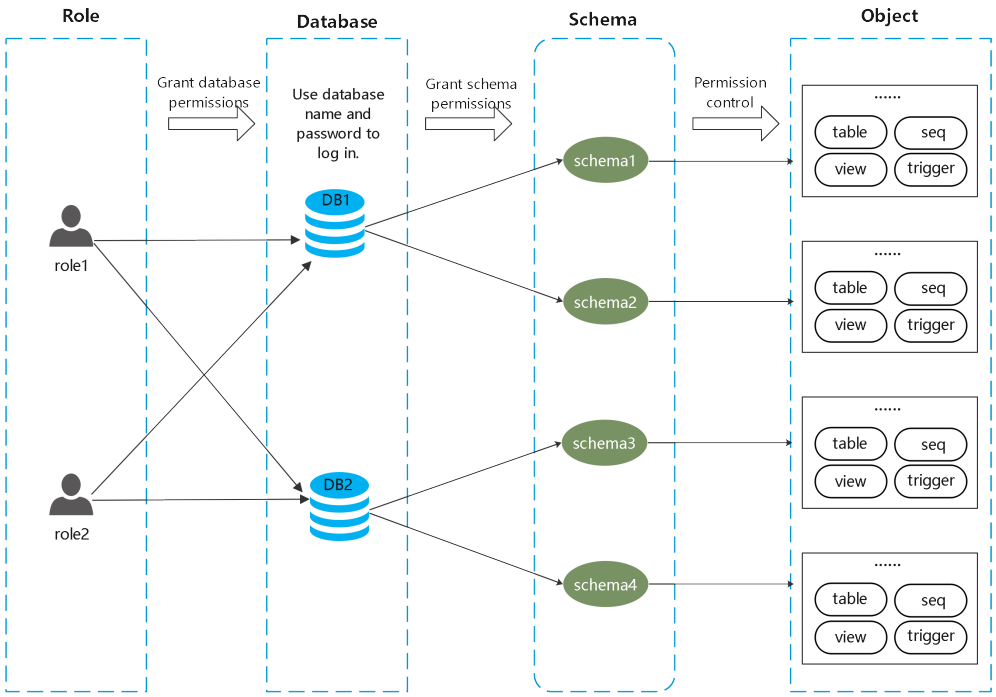
Isolamento da carga de trabalho

No GaussDB(DWS), você pode isolar cargas de trabalho por meio de configurações de banco de dados e esquemas. As suas diferenças são as seguintes:

- Os bancos de dados não podem se comunicar uns com os outros e compartilham muito poucos recursos. Suas conexões e permissões podem ser isoladas.
- Esquemas compartilham mais recursos do que bancos de dados. As permissões de usuário em esquemas e objetos subordinados podem ser configuradas de forma flexível usando a sintaxe **GRANT** e **REVOKE**.

É aconselhável usar esquemas para isolar serviços para conveniência e compartilhamento de recursos. Recomenda-se que os administradores de sistema criem esquemas e bancos de dados e atribuam as permissões necessárias aos usuários.

Figura 7-1 Usado para controle de permissão.



Banco de dados

Um banco de dados é uma coleção física de objetos de banco de dados. Recursos de diferentes bancos de dados são completamente isolados (exceto alguns objetos compartilhados). Os bancos de dados são usados para isolar cargas de trabalho. Objetos em bancos de dados diferentes não podem acessar uns aos outros. Por exemplo, objetos no Banco de dados B não podem ser acessados no Banco de dados A. Portanto, ao fazer logon em um cluster, você deve se conectar ao banco de dados especificado.

Esquema

Em um banco de dados, os objetos de banco de dados são logicamente divididos e isolados com base em esquemas.

Com o gerenciamento de permissões, você pode acessar e operar objetos em esquemas diferentes na mesma sessão. Esquemas contêm objetos que os aplicativos podem acessar, como tabelas, índices, dados em vários tipos, funções e operadores.

Objetos de banco de dados com o mesmo nome não podem existir no mesmo esquema, mas os nomes de objetos em esquemas diferentes podem ser os mesmos.

```
gaussdb=> CREATE SCHEMA myschema;
CREATE SCHEMA
gaussdb=> CREATE SCHEMA myschema_1;
CREATE SCHEMA

gaussdb=> CREATE TABLE myschema.t1(a int, b int) DISTRIBUTE BY HASH(b);
CREATE TABLE
gaussdb=> CREATE TABLE myschema.t1(a int, b int) DISTRIBUTE BY HASH(b);
ERROR:  relation "t1" already exists
gaussdb=> CREATE TABLE myschema_1.t1(a int, b int) DISTRIBUTE BY HASH(b);
CREATE TABLE
```


Os esquemas dividem logicamente as cargas de trabalho. Essas cargas de trabalho são interdependentes com os esquemas. Portanto, se um esquema contiver objetos, excluí-lo causará erros com informações de dependência exibidas.

```
gaussdb=> DROP SCHEMA myschema_1;
ERROR: cannot drop schema myschema_1 because other objects depend on it
Detail: table myschema_1.t1 depends on schema myschema_1
Hint: Use DROP ... CASCADE to drop the dependent objects too.
```

Quando um esquema é excluído, a opção **CASCADE** é usada para excluir os objetos que dependem do esquema.

```
gaussdb=> DROP SCHEMA myschema_1 CASCADE;
NOTICE:  drop cascades to table myschema_1.t1
gaussdb=> DROP SCHEMA
```

Usuário/função

Os usuários e as funções são usados para implementar o controle de permissão no servidor de banco de dados (cluster). Eles são os proprietários e executores de cargas de trabalho de cluster e gerenciam todas as permissões de objeto em clusters. Uma função não está confinada em um banco de dados específico. No entanto, ao efetuar logon no cluster, ele deve especificar explicitamente um nome de usuário para garantir a transparência da operação. As permissões de um usuário para um banco de dados podem ser especificadas por meio do gerenciamento de permissões.

Um usuário é o sujeito de permissões. O gerenciamento de permissões é, na verdade, o processo de decidir se um usuário tem permissão para executar operações em objetos de banco de dados.

Gerenciamento de permissões

O gerenciamento de permissões no GaussDB(DWS) se divide em três categorias:

- Permissão do sistema

  As permissões do sistema também são chamadas de atributos de usuário, incluindo **SYSADMIN**, **CREATEDB**, **CREATEROLE**, **AUDITADMIN** e **LOGIN**.

  Elas podem ser especificadas apenas pela sintaxe **CREATE ROLE** ou **ALTER ROLE**. A permissão **SYSADMIN** pode ser concedida e revogada usando **GRANT ALL PRIVILEGE** e **REVOKE ALL PRIVILEGE**, respectivamente. As permissões do sistema não podem ser herdadas por um usuário de uma função e não podem ser concedidas usando o **PUBLIC**.

- Permissões

  Conceda permissões de uma função ou usuário a uma ou mais funções ou usuários. Nesse caso, cada função ou usuário pode ser considerado como um conjunto de uma ou mais permissões de banco de dados.

  Se **WITH ADMIN OPTION** for especificada, o membro poderá, por sua vez, conceder permissões na função a outras pessoas e revogar permissões na função também. Se uma função ou usuário concedido com determinadas permissões for alterado ou revogado, as permissões herdadas da função ou usuário também serão alteradas.

  Um administrador de banco de dados pode conceder permissões e revogá-las de qualquer função ou usuário. As funções com permissão **CREATEROLE** podem conceder ou revogar a participação em qualquer função que não seja de administrador.

- Permissão de objeto

Permissões em um objeto de banco de dados (tabela, visão, coluna, banco de dados, função, esquema ou tablespace) podem ser concedidas a uma atribuição ou usuário. O comando **GRANT** pode ser usado para conceder permissões a um usuário ou função. Essas permissões concedidas são adicionadas às já existentes.

Exemplo de isolamento de esquema

Exemplo 1:

Por padrão, o proprietário de um esquema tem todas as permissões em objetos no esquema, incluindo a permissão de exclusão. O proprietário de um banco de dados tem todas as permissões em objetos no banco de dados, incluindo a permissão de exclusão. Portanto, é aconselhável controlar estritamente a criação de bancos de dados e esquemas. Crie bancos de dados e esquemas como administrador e atribua permissões relacionadas aos usuários.

Passo 1 Atribua a permissão para criar esquemas no banco de dados **testdb** ao usuário **user_1** como usuário **dbadmin**.

```
testdb=> GRANT CREATE ON DATABASE testdb to user_1;
GRANT
```

Passo 2 Alterne para o usuário **user_1**.

```
testdb=> SET SESSION AUTHORIZATION user_1 PASSWORD '********';
SET
```

Crie um esquema chamado **myschema_2** no banco de dados **testdb** como **user_1**.

```
testdb=> CREATE SCHEMA myschema_2;
CREATE SCHEMA
```

Passo 3 Alterne para o administrador **dbadmin**.

```
testdb=> RESET SESSION AUTHORIZATION;
RESET
```

Crie **table t1** no esquema **myschema_2** como administrador **dbadmin**.

```
testdb=> CREATE TABLE myschema_2.t1(a int, b int) DISTRIBUTE BY HASH(b);
CREATE TABLE
```

Passo 4 Alterne para o usuário **user_1**.

```
testdb=> SET SESSION AUTHORIZATION user_1 PASSWORD '********';
SET
```

Exclua a tabela **t1** criada pelo administrador **dbadmin** no esquema **myschema_2** como usuário **user_1**.

```
testdb=> drop table myschema_2.t1;
DROP TABLE
```

----Fim

Exemplo 2:

Devido ao isolamento lógico dos esquemas, os objetos do banco de dados precisam ser verificados tanto no nível do esquema quanto no nível do objeto.

Passo 1 Conceda a permissão na tabela **myschema.t1** para **user_1**.

```
gaussdb=> GRANT SELECT ON TABLE myschema.t1 TO user_1;
GRANT
```

Passo 2 Alterne para o usuário **user_1**.

```
SET SESSION AUTHORIZATION user_1 PASSWORD '********';
SET
```


Consulte a tabela **myschema.t1**.

```
gaussdb=> SELECT * FROM myschema.t1;
ERROR:  permission denied for schema myschema
LINE 1: SELECT * FROM myschema.t1;
```

Passo 3 Alterne para o administrador **dbadmin**.

```
gaussdb=> RESET SESSION AUTHORIZATION;
RESET
```

Conceda a permissão na tabela **myschema.t1** ao usuário **user_1**.

```
gaussdb=> GRANT USAGE ON SCHEMA myschema TO user_1;
GRANT
```

Passo 4 Alterne para o usuário **user_1**.

```
gaussdb=> SET SESSION AUTHORIZATION user_1 PASSWORD '********';
SET
```

Consulte a tabela **myschema.t1**.

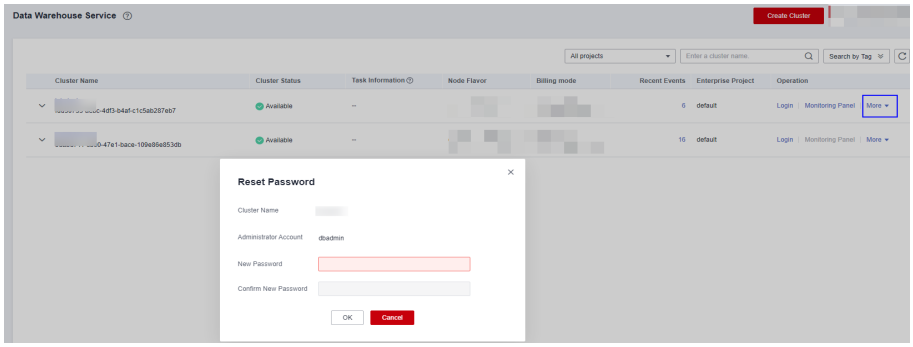
```
gaussdb=> SELECT * FROM myschema.t1;
 a | b
---+---
(0 rows)
```

----Fim

7.2 Como alterar a senha de uma conta de banco de dados quando a senha expira?

- Para alterar a senha do administrador do banco de dados **dbadmin**, faça logon no console e escolha **More** > **Reset Password** do cluster.

Figura 7-2 Redefinir a senha do usuário dbadmin



Por segurança, os dois parâmetros a seguir gerenciam senhas de conta. Efetue logon no console, clique no nome do cluster e alterne para a página de modificação de parâmetros para modificá-los.

- **failed_login_attempts**: número máximo de tentativas consecutivas de senha incorreta antes que a conta seja bloqueada. Execute a seguinte instrução como usuário **dbadmin** para desbloquear a conta:
  ```
  ALTER USER user_name ACCOUNT UNLOCK;
  ```
- **password_effect_time**: período de validade da senha da conta, em dias. O valor padrão é **90**.

- Você também pode se conectar ao banco de dados e executar o comando **ALTER USER** para alterar o período de validade da senha de uma conta de banco de dados (usuário comum e administrador dbadmin).

```
ALTER USER username PASSWORD EXPIRATION 90;
```

7.3 Como conceder permissões de tabela a um usuário?

Esta seção descreve como conceder aos usuários as permissões SELECT, INSERT, UPDATE ou permissões completas de tabelas aos usuários.

Sintaxe

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | TRUNCATE | REFERENCES | TRIGGER |
ANALYZE | ANALYSE } [, ...]
      | ALL [ PRIVILEGES ] }
    ON { [ TABLE ] table_name [, ...]
       | ALL TABLES IN SCHEMA schema_name [, ...] }
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

Cenário

Suponha que existam usuários **u1**, **u2**, **u3**, **u4** e **u5** e cinco esquemas nomeados após esses usuários. Seus requisitos de permissão são os seguintes:

- O usuário **u2** é um usuário somente leitura e requer a permissão SELECT para a **u1.t1**.
- O usuário **u3** requer a permissão SELECT para a tabela **u1.t1**.
- O usuário **u3** requer a permissão UPDATE para a tabela **u1.t1**.
- O usuário **u5** requer todas as permissões da tabela **u1.t1**.

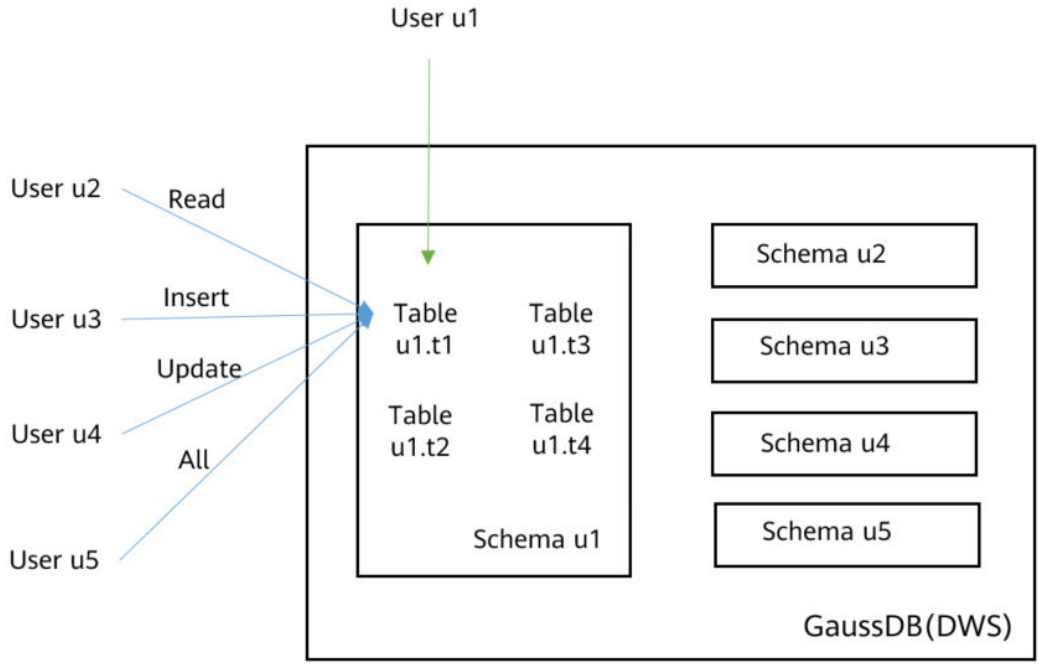


Tabela 7-1 Permissões da tabela u1.t1

Usuário	Tipo	Instrução GRANT	Consultar	Inserir	Atualizar	Excluir
u1	Proprietário	-	√	√	√	√
u2	Usuário de somente leitura	`GRANT SELECT ON u1.t1 TO u2;`	√	x	x	x
u3	Usuário de INSERT	`GRANT INSERT ON u1.t1 TO u3;`	x	√	x	x
u4	Usuário de UPDATE	`GRANT SELECT,UPDATE ON u1.t1 TO u4;` **AVISO** A permissão UPDATE deve ser concedida juntamente com a permissão SELECT, ou pode ocorrer vazamento de informações.	√	x	√	x
u5	Usuários com todas as permissões	`GRANT ALL PRIVILEGES ON u1.t1 TO u5;`	√	√	√	√

Procedimento

Execute as seguintes etapas para conceder e verificar permissões:

Passo 1 Conecte-se ao seu banco de dados como **dbadmin**. Execute as instruções a seguir para criar os usuários de **u1** a **u5**. Cinco esquemas serão criados e nomeados após os usuários por padrão.

```
CREATE USER u1 PASSWORD '{password}';
CREATE USER u2 PASSWORD '{password}';
CREATE USER u3 PASSWORD '{password}';
CREATE USER u4 PASSWORD '{password}';
CREATE USER u5 PASSWORD '{password}';
```

Passo 2 Criar tabela **u1.t1** no esquema **u1**.

```
CREATE TABLE u1.t1 (c1 int, c2 int);
```

Passo 3 Insira dois registros na tabela.

```
INSERT INTO u1.t1 VALUES (1,2);
INSERT INTO u1.t1 VALUES (1,2);
```

Passo 4 Conceda permissões de esquema aos usuários.

```
GRANT USAGE ON SCHEMA u1 TO u2,u3,u4,u5;
```

Passo 5 Conceda ao usuário **u2** a permissão para consultar a tabela **u1.t1**.

```
GRANT SELECT ON u1.t1 TO u2;
```

Passo 6 Inicie uma nova sessão e conecte-se ao banco de dados como o usuário **u2**. Verifique se o usuário **u2** pode consultar a tabela **u1.t1**, mas não pode gravar ou modificar a tabela.

```
SELECT * FROM u1.t1;
INSERT INTO u1.t1 VALUES (1,20);
UPDATE u1.t1 SET c2 = 3 WHERE c1 =1;
```

```
gaussdb=> SELECT * FROM u1.t1;
 c1 | c2
----+----
  1 |  2
  1 |  2
(2 rows)

gaussdb=> INSERT INTO u1.t1 VALUES (1,20);
ERROR:  permission denied for relation t1
gaussdb=> UPDATE u1.t1 SET c2 = 3 WHERE c1 =1;
ERROR:  permission denied for relation t1
```

Passo 7 Na sessão iniciada pelo usuário **dbadmin**, conceda permissões aos usuários **u3**, **u4** e **u5**.

```
GRANT INSERT ON u1.t1 TO u3; -- Allow u3 to insert data.
GRANT SELECT,UPDATE ON u1.t1 TO u4; -- Allow u4 to modify the table.
GRANT ALL PRIVILEGES ON u1.t1 TO u5; -- Allow u5 to query, insert, modify, and
delete table data.
```

Passo 8 Inicie uma nova sessão e conecte-se ao banco de dados como o usuário **u3**. Verifique se o usuário **u3** pode consultar a tabela **u1.t1**, mas não pode consultar ou modificar a tabela.

```
SELECT * FROM u1.t1;
INSERT INTO u1.t1 VALUES (1,20);
UPDATE u1.t1 SET c2 = 3 WHERE c1 =1;
```

```
gaussdb=> SELECT * FROM u1.t1;
ERROR:  permission denied for relation t1
gaussdb=> INSERT INTO u1.t1 VALUES (1,20);
INSERT 0 1
gaussdb=> UPDATE u1.t1 SET c2 = 3 WHERE c1 =1;
ERROR:  permission denied for relation t1
```


Passo 9 Inicie uma nova sessão e conecte-se ao banco de dados como o usuário **u4**. Verifique se o usuário **u4** pode modificar e consultar a tabela **u1.t1**, mas não pode inserir dados na tabela.

```
SELECT * FROM u1.t1;
INSERT INTO u1.t1 VALUES (1,20);
UPDATE u1.t1 SET c2 = 3 WHERE c1 =1;
```

```
gaussdb=> SELECT * FROM u1.t1;
 c1 | c2
----+----
  1 |  2
  1 |  2
  1 | 20
(3 rows)

gaussdb=> INSERT INTO u1.t1 VALUES (1,20);
ERROR:  permission denied for relation t1
gaussdb=> UPDATE u1.t1 SET c2 = 3 WHERE c1 =1;
UPDATE 3
```

Passo 10 Inicie uma nova sessão e conecte-se ao banco de dados como o usuário **u5**. Verifique se o usuário **u5** pode consultar, inserir, modificar e excluir dados na tabela **u1.t1**.

```
SELECT * FROM u1.t1;
INSERT INTO u1.t1 VALUES (1,20);
UPDATE u1.t1 SET c2 = 3 WHERE c1 =1;
DELETE FROM u1.t1;
```

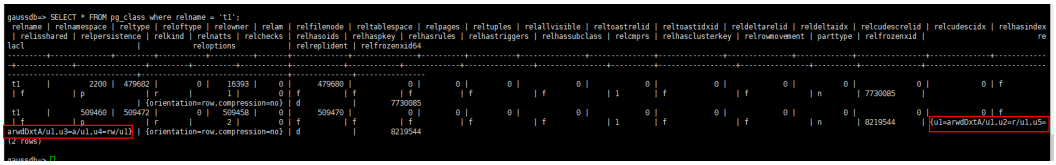
```
gaussdb=> SELECT * FROM u1.t1;
 c1 | c2
----+----
  1 |  3
  1 |  3
  1 |  3
(3 rows)

gaussdb=> INSERT INTO u1.t1 VALUES (1,20);
INSERT 0 1
gaussdb=> UPDATE u1.t1 SET c2 = 3 WHERE c1 =1;
UPDATE 4
gaussdb=> DELETE FROM u1.t1;
DELETE 4
```

Passo 11 Na sessão iniciada pelo usuário **dbadmin**, execute a função has_table_privilege para consultar as permissões do usuário.

```
SELECT * FROM pg_class WHERE relname = 't1';
```

Verifique a coluna **relacl** na saída do comando. *rolename*=*xxxx*/*yyyy* indica que *rolename* tem a permissão *xxxx* na tabela e a permissão é obtida de *yyyy*.

A figura a seguir mostra a saída do comando.



- **u1=arwdDxtA/u1** indica que **u1** é o proprietário e tem permissões completas.
- **u2=r/u1** indica que **u2** tem a permissão de leitura.

- **u3=a/u1** indica que **u3** tem a permissão de inserção.
- **u4=rw/u1** indica que **u4** tem as permissões de leitura e atualização.
- **u5=arwdDxtA/u1** indica que **u5** tem permissões completas.

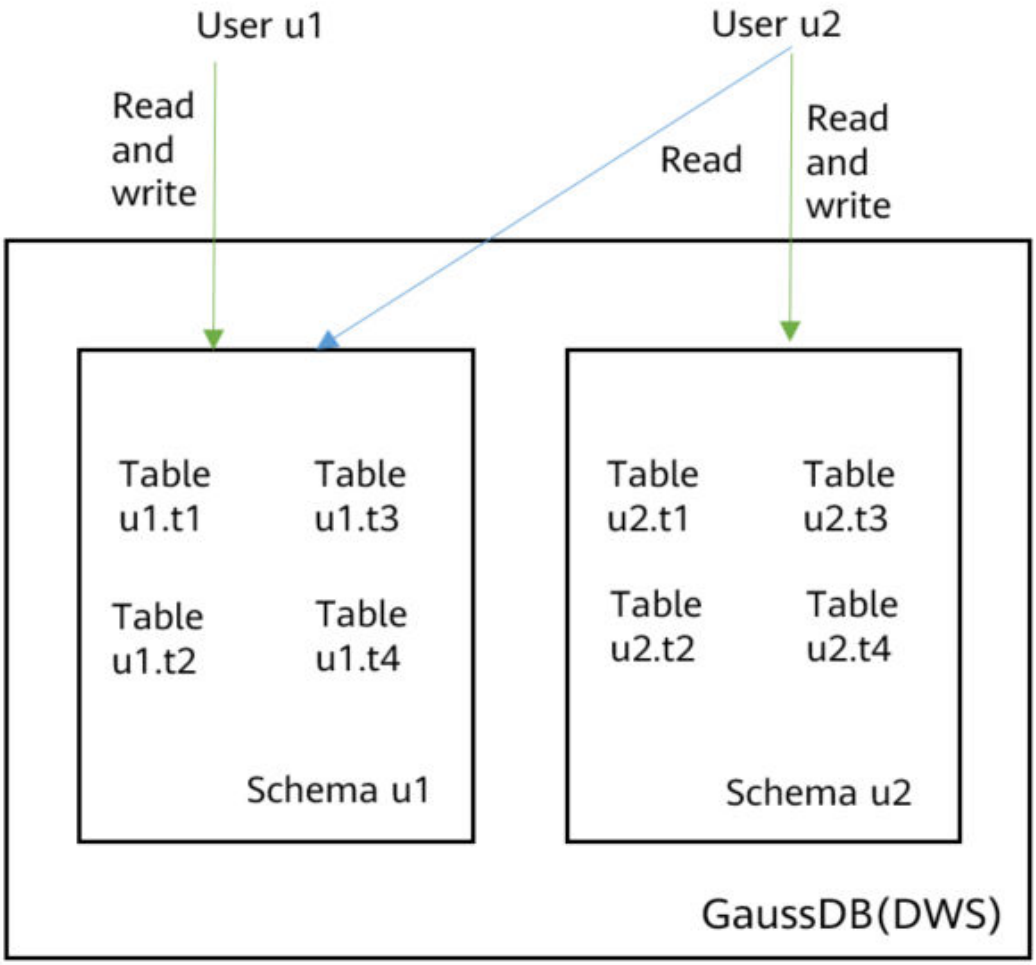
----Fim

7.4 Como conceder permissões de esquema a um usuário?

Esta seção descreve como conceder a permissão de consulta para um esquema como um exemplo. Para obter mais informações, consulte **Como conceder permissões de tabela a um usuário?** :

- Permissão para uma tabela em um esquema
- Permissão para todas as tabelas em um esquema
- Permissão para que as tabelas sejam criadas no esquema

Suponha que existem usuários **u1** e **u2**, e dois esquemas nomeados após eles. O usuário **u2** precisa acessar tabelas no esquema **u1**.



Passo 1 Conecte-se ao seu banco de dados como **dbadmin**. Execute as instruções a seguir para criar os usuários **u1** e **u2**. Dois esquemas serão criados e nomeados após os usuários por padrão.


```
CREATE USER u1 PASSWORD '{password}';
CREATE USER u2 PASSWORD '{password}';
```

Passo 2 Crie as tabelas **u1.t1** e **u1.t2** no esquema **u1**.

```
CREATE TABLE u1.t1 (c1 int, c2 int);
CREATE TABLE u1.t2 (c1 int, c2 int);
```

Passo 3 Conceda a permissão de acesso do esquema **u1** ao usuário **u2**.

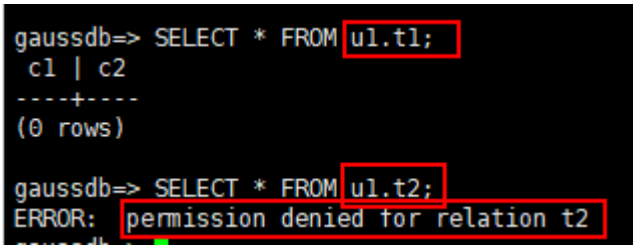
```
GRANT USAGE ON SCHEMA u1 TO u2;
```

Passo 4 Conceda ao usuário **u2** a permissão para consultar a tabela **u1.t1** no esquema **u1**.

```
GRANT SELECT ON u1.t1 TO u2;
```

Passo 5 Inicie uma nova sessão e conecte-se ao banco de dados como usuário **u2**. Verifique se o usuário **u2** pode consultar a tabela **u1.t1**, mas não a tabela **u1.t2**.

```
SELECT * FROM u1.t1;
SELECT * FROM u1.t2;
```

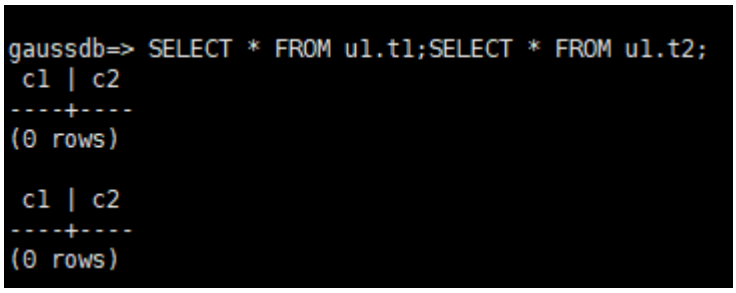


Passo 6 Na sessão iniciada pelo usuário **dbadmin**, conceda ao usuário **u2** a permissão para consultar todas as tabelas no esquema **u1**.

```
GRANT SELECT ON ALL TABLES IN SCHEMA u1 TO u2;
```

Passo 7 Na sessão iniciada pelo usuário **u2**, verifique se **u2** pode consultar todas as tabelas.

```
SELECT * FROM u1.t1;
SELECT * FROM u1.t2;
```

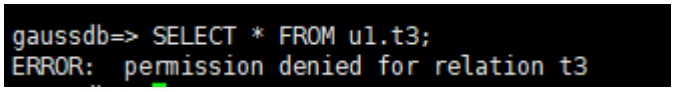


Passo 8 Na sessão iniciada pelo usuário **dbadmin**, crie a tabela **u1.t3**.

```
CREATE TABLE u1.t3 (c1 int, c2 int);
```

Passo 9 Na sessão iniciada pelo usuário **u2**, verifique se o usuário **u2** não tem a permissão de consulta para **u1.t3**. Indica que o usuário **u2** tem permissão para acessar todas as tabelas existentes no esquema **u1**, mas não as tabelas a serem criadas no futuro.

```
SELECT * FROM u1.t3;
```



Passo 10 Na sessão iniciada pelo usuário **dbadmin**, conceda ao usuário **u2** a permissão para consultar as tabelas a serem criadas no esquema **u1**. Crie tabela **u1.t4**.

NOTA

ALTER DEFAULT PRIVILEGES é usado para conceder permissões em objetos a serem criados.

```
ALTER DEFAULT PRIVILEGES FOR ROLE u1 IN SCHEMA u1 GRANT SELECT ON TABLES TO u2;
CREATE TABLE u1.t4 (c1 int, c2 int);
```

Passo 11 Na sessão iniciada pelo usuário **u2**, verifique se o usuário **u2** pode acessar a tabela **u1.t4**, mas não tem permissão para acessar **u1.t3**. Para permitir que o usuário acesse a tabela **u1.t3**, você pode conceder permissões executando **Passo 4**.

```
SELECT * FROM u1.t4;
```

```
gaussdb=> SELECT * FROM u1.t4;
 c1 | c2
----+----
(0 rows)
```

----Fim

7.5 Como criar um usuário somente leitura do banco de dados?

Cenário

No desenvolvimento de serviços, os administradores de banco de dados usam esquemas para classificar dados. Por exemplo, no setor financeiro, os dados de passivo pertencem ao esquema **s1**, e os dados de ativos pertencem ao esquema **s2**.

Agora você precisa criar um usuário somente leitura **user1** no banco de dados. O usuário pode acessar todas as tabelas (incluindo novas tabelas a serem criadas no futuro) no esquema **s1** para leitura diária, mas não pode inserir, modificar ou excluir dados.

Princípios

O DWS fornece gerenciamento de usuários baseado em funções. Você precisa criar uma função somente leitura **role1** e conceder a função ao **user1**. Para obter detalhes, consulte **Controle de acesso baseado em função**.

Procedimento

Passo 1 Conecte-se ao banco de dados do DWS como usuário **dbadmin**.

Passo 2 Execute a seguinte instrução SQL para criar a função **role1**:

```
CREATE ROLE role1 PASSWORD disable;
```

Passo 3 Execute a seguinte instrução SQL para conceder permissões à **role1**:

```
The GRANT usage ON SCHEMA s1 TO role1; -- grants the access permission to schema
s1.
GRANT select ON ALL TABLES IN SCHEMA s1 TO role1; -- grants the query permission
on all tables in schema s1.
ALTER DEFAULT PRIVILEGES FOR USER tom IN SCHEMA s1 GRANT select ON TABLES TO
role1; -- grants schema s1 the permission to create tables. tom is the owner of
schema s1.
```

Passo 4 Execute a seguinte instrução SQL para conceder a função **role1** ao usuário atual **user1**:


```
GRANT role1 TO user1;
```

Passo 5 Leia todos os dados da tabela no esquema **s1** como usuário somente leitura **user1**.

----Fim

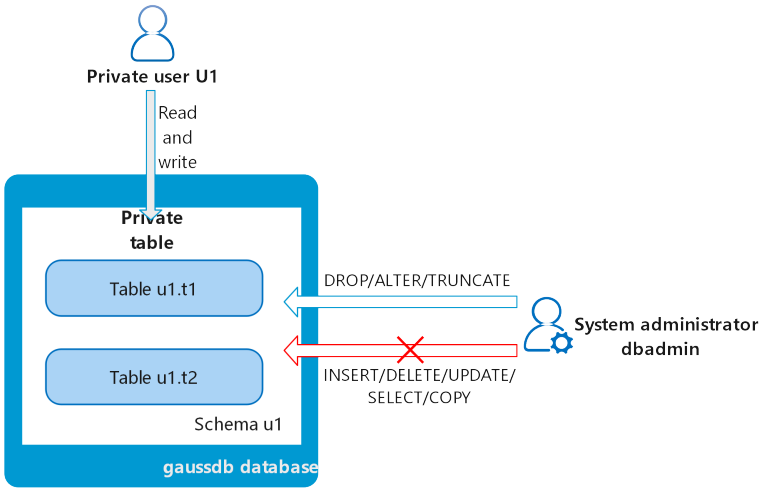
7.6 Como criar usuários e tabelas de banco de dados privados?

Cenário

O administrador do sistema **dbadmin** tem permissão para acessar tabelas criadas por usuários comuns por padrão. Quando a **Separação de permissões** está ativada, o administrador **dbadmin** não tem permissão para acessar tabelas de usuários comuns ou executar operações de controle (DROP, ALTER e TRUNCATE).

Se um usuário privado e uma tabela privada (tabela criada pelo usuário privado) precisarem ser criados, e a tabela privada puder ser acessada apenas pelo usuário privado e o administrador do sistema **dbadmin** e outros usuários comuns não tiverem permissão para acessar a tabela (INSERT, DELETE, UPDATE, SELECT e COPY). No entanto, o administrador do sistema **dbadmin** às vezes precisa executar as operações DROP, ALTER ou TRUNCATE sem autorização do usuário privado. Neste caso, pode criar um usuário (usuário privado) com o atributo INDEPENDENT.

Figura 7-3 Usuários privados



Princípios

Esta função é implementada através da criação de um usuário com o atributo INDEPENDENT.

INDEPENDENT | NOINDEPENDENT define funções privadas e independentes. Para uma função com o atributo **INDEPENDENT**, os direitos dos administradores para controlar e acessar essa função são separados. As regras específicas são as seguintes:

- Os administradores não têm direitos para adicionar, excluir, consultar, modificar, copiar ou autorizar os objetos de tabela correspondentes sem a autorização da função INDEPENDENT.

- Os administradores não têm direitos para modificar a relação de herança da função INDEPENDENT sem a autorização desta função.
- Os administradores não têm direitos para modificar o proprietário dos objetos de tabela para a função INDEPENDENT.
- Os administradores não têm direitos para alterar a senha do banco de dados da função INDEPENDENT. A função INDEPENDENT deve gerenciar sua própria senha, que não pode ser redefinida se perdida.
- O atributo **SYSADMIN** de um usuário não pode ser alterado para o atributo **INDEPENDENT**.

Procedimento

Passo 1 Conecte-se ao banco de dados do DWS como usuário **dbadmin**.

Passo 2 Execute a seguinte instrução SQL para criar o usuário privado **u1**:

```
CREATE USER u1 WITH INDEPENDENT IDENTIFIED BY 'password';
```

Passo 3 Alterne para o usuário **u1**, crie a tabela **test** e insira dados na tabela.

```
CREATE TABLE test (id INT, name VARCHAR(20));
INSERT INTO test VALUES (1, 'joe');
INSERT INTO test VALUES (2, 'jim');
```

Passo 4 Alterne para o usuário **dbadmin** e execute a seguinte instrução SQL para verificar se o usuário **dbadmin** pode acessar a tabela privada **test** criada pelo usuário privado **u1**:

```
SELECT * FROM u1.test;
```

O resultado da consulta indica que o usuário **dbadmin** não tem a permissão de acesso. Isso significa que o usuário privado e a tabela privada foram criados com sucesso.

```
gaussdb=> SELECT * FROM u1.test;
ERROR:  SELECT permission denied to user "dbadmin" for relation "u1.test"
```

Passo 5 Execute a instrução **DROP** como usuário **dbadmin** para excluir a tabela **test**.

```
DROP TABLE u1.test;
```

```
gaussdb=> drop table u1.test;
DROP TABLE
```

----Fim

7.7 Como revogar a permissão CONNECT ON DATABASE de um usuário?

Cenário

Em um serviço, a permissão do usuário **u1** para se conectar a um banco de dados precisa ser revogada. Depois que o comando **REVOKE CONNECT ON DATABASE *gaussdb* FROM u1;** é executado com sucesso, o usuário **u1** ainda pode se conectar ao banco de dados. Isso significa que a revogação não terá efeito.

Análise de causa

Se você executar o comando **REVOKE CONNECT ON DATABASE gaussdb from u1** para revogar as permissões do usuário **u1**, a revogação não terá efeito porque a permissão **CONNECT** do banco de dados é concedida ao **PUBLIC**. Portanto, você precisa especificar **PUBLIC**.

- GaussDB(DWS) fornece um grupo implicitamente definido **PUBLIC** que contém todas as funções. Por padrão, todos os novos usuários e funções têm as permissões de **PUBLIC**. Para revogar permissões de **PUBLIC** de um usuário ou função, ou conceder novamente essas permissões a eles, adicione a palavra-chave **PUBLIC** na instrução **REVOKE** ou **GRANT**.
- GaussDB(DWS) concede as permissões para objetos de certos tipos ao **PUBLIC**. Por padrão, as permissões em tabelas, colunas, sequências, fontes de dados externas, servidores externos, esquemas e tablespaces não são concedidas ao **PUBLIC**, mas as seguintes permissões são concedidas ao **PUBLIC**;
  - Permissão **CONNECT** de um banco de dados
  - permissão **CREATE TEMP TABLE** de um banco de dados
  - Permissão **EXECUTE** de uma função
  - Permissão **USAGE** para idiomas e tipos de dados (incluindo domínios)
- Um proprietário de objeto pode revogar as permissões padrão concedidas a **PUBLIC** e conceder permissões a outros usuários conforme necessário.

Exemplo de operações

Execute o seguinte comando para revogar a permissão do usuário **u1** para acessar o banco de dados **gaussdb**:

Passo 1 Conecte-se ao banco de dados GaussDB(DWS) **gaussdb**.

```
gsql -d gaussdb -p 8000 -h 192.168.x.xx -U dbadmin -W password -r
gaussdb=>
```

Passo 2 Crie usuário **u1**.

```
gaussdb=> CREATE USER u1 IDENTIFIED BY 'xxxxxxxx';
```

Passo 3 Verifique se o usuário **u1** pode acessar o GaussDB.

```
gsql -d gaussdb -p 8000 -h 192.168.x.xx -U u1 -W password -r
gaussdb=>
```

Passo 4 Conecte-se ao banco de dados **gaussdb** como administrador **dbadmin** e execute o comando REVOKE para revogar a permissão **connect on database** do usuário **public**.

```
gsql -d gaussdb -h 192.168.x.xx -U dbadmin -p 8000 -r
gaussdb=> REVOKE CONNECT ON DATABASE gaussdb FROM public;
REVOKE
```

Passo 5 Verifique o resultado. Use **u1** para se conectar ao banco de dados. Se as seguintes informações forem exibidas, a permissão **connect on database** do usuário **u1** foi revogada com êxito:

```
gsql -d gaussdb -p 8000 -h 192.168.x.xx -U u1 -W password -r
gsql: FATAL:  permission denied for database "gaussdb"
DETAIL:  User does not have CONNECT privilege.
```

----Fim

7.8 Como visualizar as permissões da tabela de um usuário?

Cenário 1: execute o comando **information_schema.table_privileges** para **exibir as permissões de tabela de um usuário**. Exemplo:

```
SELECT * FROM information_schema.table_privileges WHERE GRANTEE='user_name';
```

```
gaussdb=> SELECT * FROM information_schema.table_privileges WHERE GRANTEE='u2';
 grantor | grantee | table_catalog | table_schema | table_name | privilege_type | is_grantable | with_hierarchy
---------+---------+---------------+--------------+------------+----------------+--------------+----------------
 u2      | u2      | gaussdb       | u2           | t2         | INSERT         | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | SELECT         | YES          | YES
 u2      | u2      | gaussdb       | u2           | t2         | UPDATE         | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | DELETE         | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | TRUNCATE       | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | REFERENCES     | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | TRIGGER        | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | ANALYZE        | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | VACUUM         | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | ALTER          | YES          | NO
 u2      | u2      | gaussdb       | u2           | t2         | DROP           | YES          | NO
 u1      | u2      | gaussdb       | u1           | t1         | SELECT         | NO           | YES
(12 rows)
```

Tabela 7-2 Colunas table_privileges

Coluna	Tipos de dados	Descrição
grantor	sql_identifier	Concedente da permissão
grantee	sql_identifier	Beneficiário de permissão
table_catalog	sql_identifier	Banco de dados onde a tabela é
table_schema	sql_identifier	Esquema em que a tabela está
table_name	sql_identifier	Nome da tabela
privilege_type	character_data	Tipo da permissão concedida. O valor pode ser **SELECT**, **INSERT**, **UPDATE**, **DELETE**, **TRUNCATE**, **REFERENCES**, **ANALYZE**, **VACUUM**, **ALTER**, **DROP** ou **TRIGGER**.
is_grantable	yes_or_no	Indica se a permissão pode ser concedida a outros usuários. **YES** indica que a permissão pode ser concedida a outros usuários e **NO** indica que a permissão não pode ser concedida a outros usuários.
with_hierarchy	yes_or_no	Indica se operações específicas podem ser herdadas no nível da tabela. Se a operação específica for **SELECT**, **YES** será exibido. Caso contrário, **NO** será exibido.

Na figura anterior, o usuário **u2** tem todas as permissões da tabela **t2** no esquema **u2** e a permissão **SELECT** da tabela **t1** no esquema **u1**.

information_schema.table_privileges pode consultar apenas as permissões concedidas diretamente ao usuário, a função **has_table_privilege()** pode consultar tanto permissões

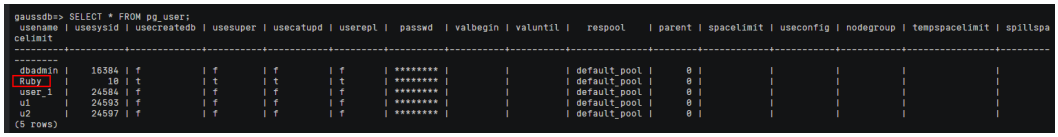
concedidas diretamente quanto permissões indiretas (obtidas pela função GRANT para o usuário). Por exemplo:

```
CREATE TABLE t1 (c1 int);
CREATE USER u1 password '********';
CREATE USER u2 password '********';
GRANT dbadmin to u2; //Indirectly grant permissions through roles.
GRANT SELECT on t1 to u1; // Directly grant the permission.

SET ROLE u1 password '********';
SELECT * FROM public.t1; // Directly grant the permission to access the table.
 c1
----
(0 rows)

SET ROLE u2 password '********';
SELECT * FROM public.t1; // Indirectly grant the permission to access the table.
 c1
----
(0 rows)

RESET role; //Switch back to dbadmin.
SELECT * FROM information_schema.table_privileges WHERE table_name = 't1'; // Can
only view direct grants.
 grantor |  grantee   | table_catalog | table_schema | table_name |
privilege_type | is_grantable | with_hierarchy
---------+------------+---------------+--------------+------------
+----------------+--------------+----------------
 dbadmin | u1         | gaussdb       | public       | t1         |
SELECT         | NO           | YES
(1 rows)

SELECT has_table_privilege('u2', 'public.t1', 'select'); // Can view both direct
and indirect grants.
 has_table_privilege
---------------------
 t
(1 row)
```

7.9 Quem é o usuário Ruby?

Quando você executa a instrução **SELECT * FROM pg_user** para exibir usuários no sistema atual, você pode ver o usuário **Ruby** muitas vezes, que tem muitas permissões.

O usuário **Ruby** é uma conta oficial de O&M. Depois que um banco de dados GaussDB(DWS) é criado, o usuário **Ruby** é gerado por padrão. Não há risco de segurança em relação ao usuário **Ruby**.



8 Desempenho do banco de dados

8.1 Por que a execução de SQL é lenta após o longo uso do GaussDB(DWS)?

Depois que um banco de dados é usado por um período de tempo, os dados da tabela aumentam à medida que os serviços crescem ou os dados da tabela são frequentemente adicionados, excluídos ou modificados. Como resultado, tabelas de inchaço e estatísticas imprecisas são incorridos, deteriorando o desempenho do banco de dados.

Recomenda-se que você execute **VACUUM FULL** e **ANALYZE** periodicamente em tabelas que são frequentemente adicionadas, excluídas ou modificadas. Realize as operações a seguir:

Passo 1 Por padrão, 100 dos 30.000 registros de estatísticas são coletados. Quando uma grande quantidade de dados está envolvida, a execução de SQL é instável, o que pode ser causado por um plano de execução alterado. Nesse caso, a taxa de amostragem precisa ser ajustada para fins estatísticos. Você pode executar **set default_statistics_target** para aumentar a taxa de amostragem, o que ajuda o otimizador a gerar o plano ideal.

```
gaussdb=> set default_statistics_target=-2;
SET
```

Passo 2 Execute **ANALYZE** novamente. Para obter detalhes, consulte **ANALYZE | ANALYSE**.

```
gaussdb=> ANALYZE customer_t1;
ANALYZE
```

----Fim

NOTA

Para testar se os fragmentos de disco afetam o desempenho do banco de dados, use a seguinte função:

```
SELECT * FROM pgxc_get_stat_dirty_tables(30,100000);
```


8.2 Por que o GaussDB(DWS) tem desempenho pior do que um banco de dados de servidor único em cenários extremes?

Devido à limitação da arquitetura MPP do GaussDB(DWS), alguns métodos e funções do PostgreSQL não podem ser enviados para DNs para execução. Como resultado, os gargalos de desempenho ocorrem em CNs.

Explicação:

- Uma operação só pode ser executada concorrentemente quando for logicamente uma operação concorrente. Por exemplo, SUM executada em todos os DNs simultaneamente deve centralizar o resumo final em um CN. Neste caso, a maior parte do trabalho de resumo foi concluída em DNs, de modo que o trabalho no CN é relativamente leve.
- Em alguns cenários, a operação deve ser executada centralmente em um nó. Por exemplo, atribuir um nome globalmente exclusivo a um ID de transação é implementado usando o sistema GTM. Portanto, o GTM também é um componente globalmente exclusivo (ativo/em espera). Todas as tarefas globalmente únicas são implementadas através do GTM em GaussDB(DWS), mas o código do software é otimizado para reduzir este tipo de tarefas. Portanto, o GTM não tem muitos gargalos. Em alguns cenários, GTM-Free e GTM-Lite podem ser implementados.
- Para garantir um excelente desempenho, os serviços precisam ser ligeiramente modificados para adaptação durante a migração do modo de desenvolvimento de aplicações do tradicional banco de dados de nó único para o banco de dados paralelo, especialmente para o tradicional aninhamento de procedimentos armazenados de Oracle.

Soluções:

- Se esse problema ocorrer, consulte "Otimização de desempenho de consulta" no ***Guia de desenvolvedor do Data Warehouse Service (DWS)***.
- Como alternativa, entre em contato com o suporte técnico para modificar e otimizar os serviços.

8.3 Como exibir registros de execução de SQL em um determinado período quando as solicitações de leitura e gravação são bloqueadas?

Você pode usar o recurso de top SQL para exibir instruções SQL executadas em um período especificado. As instruções SQL do CN atual ou de todos os CNs podem ser exibidas.

Top SQL permite visualizar instruções SQL históricas e em tempo real.

- Para obter detalhes sobre a consulta de instrução SQL em tempo real, consulte a seção **TopSQL em tempo real**.
- Para obter detalhes sobre como consultar instruções SQL históricas, consulte a seção **TopSQL histórica** .

8.4 O que devo fazer se meu cluster não estiver disponível devido a espaço insuficiente?

Você pode usar um snapshot para restaurar o cluster para um novo que tenha espaço de armazenamento maior e, em seguida, excluir o cluster anterior para evitar o desperdício de recursos. Você pode aprender o armazenamento de cluster verificando **Available Storage**. Para restaurar o cluster para um novo, consulte **Restauração de um snapshot em um novo cluster** Para excluir um cluster, consulte **Exclusão de um cluster**.

NOTA

Este método só é suportado pelo armazém de dados padrão.

8.5 O que é derramamento de operador no GaussDB(DWS)?

Durante a execução da consulta, se a memória do cluster for insuficiente, o banco de dados optará por armazenar os resultados temporários no disco. Quando o armazenamento em disco usado pelos resultados temporários exceder um determinado valor, os usuários receberão um alarme: "data spilling exceeds the threshold". O que significa derramamento?

Derramamento de operador para discos

Qualquer computação consome espaço de memória. Se muita memória for consumida, a memória para executar outros trabalhos será insuficiente, causando a execução instável do trabalho. Portanto, você precisa limitar o uso de memória de instruções de consulta para garantir a estabilidade da execução do trabalho.

Se uma tarefa em execução exigir 500 MB de memória, mas apenas 300 MB de memória forem alocados a ela, os dados que não forem usados temporariamente precisarão ser gravados em discos e apenas os dados que estiverem sendo usados serão retidos na memória. Isso é chamado de derramamento de dados. Também é chamado de **derramamento de operador**. O grande tamanho dos dados derramados para o disco pode causar 100% de uso do disco. Se isso acontecer, o banco de dados se transformará em somente leitura e o desempenho da consulta será muito afetado. Portanto, o GaussDB(DWS) impõe um limite ao derramamento do operador. Se o derramamento do operador exceder o limite, um erro é relatado e a consulta é encerrada.

Quais operadores podem derramar?

Os operadores que podem derramar para o disco incluem **Hash(VecHashJoin)**, **Agg(VecAgg)**, **Sort(VecSort)**, **Material(VecMaterial)**, **SetOp(VecSetOp)** e **WindowAgg(VecWindowAgg)**. Eles podem ser vetorizados ou não vetorizados.

Quais parâmetros podem ser usados para controlar o derramamento do operador?

- **work_mem**: define limite de derramamento. O uso do disco que exceda esse parâmetro causará o derramamento do operador. Este parâmetro tem efeito somente quando a memória não é autoadaptativa. (**enable_dynamic_workload=off**). Ele garante a taxa de

transferência simultânea e o desempenho de uma única tarefa de consulta. Portanto, você precisa otimizar o parâmetro com base na saída de **Explain Performance**.

- **temp_file_limit**: limita o tamanho dos arquivos derramados em discos. É aconselhável definir esse parâmetro com base nos requisitos do site para evitar que os arquivos derramados usem o espaço em disco. Arquivos derramados que excedam o valor deste parâmetro causarão um erro.

Como saber se uma instrução é derramada para discos?

- Verifique os arquivos de derramamento. Os arquivos de derramamento são armazenados no diretório **base/pgsql_tmp** do diretório da instância. Os arquivos de derramamento são nomeados *pgsql_tmp$queryid_$pid*. Você pode determinar qual instrução SQL é derramada para o disco com base na **queryid**.
- Verifique a exibição de espera (**pgxc_thread_wait_status**). Se **write file** for exibido no modo de exibição de espera, há resultados temporários derramados em discos.
- Verifique o plano de execução (**EXPLAIN PERFORMANCE**). Palavras-chave como **spill**, **written disk** e **temp file num** indicam que há um derramamento de operador.
- Verifique se a coluna **spill_info** em **TopSQL em tempo real** ou **TopSQL histórica** contém informações de derramamento. Se essa coluna não estiver vazia, os dados foram derramados para discos em DNs. (Pré-requisito: a função topsql foi habilitada.)

Como evitar o derramamento?

Quando os operadores são derramados em discos, os dados de cálculo do operador são gravados em discos. Em comparação com o acesso à memória, as operações de disco são lentas, causando deterioração do desempenho e deterioração do tempo de resposta da consulta. Portanto, tente evitar o derramamento do operador durante a execução da consulta. É recomendável usar os seguintes métodos:

- Reduzir o conjunto de resultados intermediários: se o conjunto de resultados intermediários for muito grande, você poderá adicionar critérios de filtro para reduzir o tamanho do conjunto de resultados intermediários.
- Evitar distorção de dados: se houver uma distorção grave de dados, o derramamento ocorrerá em um DN que tenha uma grande quantidade de dados.
- Realizar a **ANALYZE** em tempo hábil: quando as estatísticas são imprecisas, o número de linhas pode ser estimado como menor. Como resultado, o plano não é ideal e os dados são derramados para os discos.
- Realizar otimização de ponto único: realize otimização em instruções SQL únicas.
- Se a memória não for autoadaptativa e o conjunto de resultados intermediários não puder ser reduzido, aumente o valor de **work_mem** como apropriado.
- Se a memória for autoadaptativa, aumente a memória disponível do banco de dados o máximo possível para reduzir a probabilidade de derramamento de dados.

8.6 Gerenciamento de recursos da CPU do GaussDB(DWS)

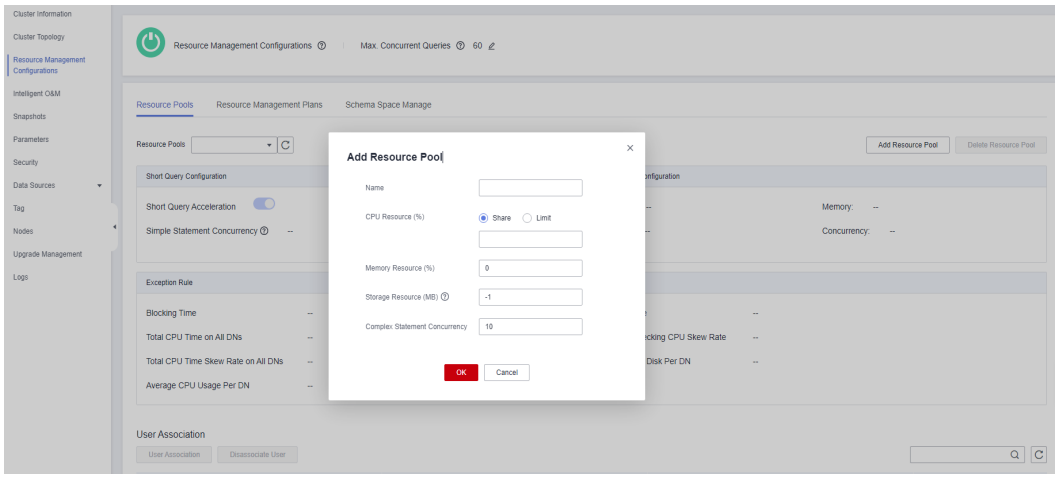
Visão geral do gerenciamento de recursos da CPU

Em diferentes cenários de serviço, os recursos do sistema (CPU, memória, I/O e recursos de armazenamento) do banco de dados são alocados adequadamente às consultas para garantir o desempenho da consulta e a estabilidade do serviço.

GaussDB(DWS) fornece a função de gerenciamento de recursos. Você pode colocar recursos em diferentes pools de recursos, que são isolados uns dos outros. Em seguida, você pode associar usuários de banco de dados a esses pools de recursos. Quando um usuário inicia uma consulta SQL, a consulta será transferida para o pool de recursos associado ao usuário. Você pode especificar o número de consultas que podem ser executadas simultaneamente em um pool de recursos, o limite superior de memória usado para uma única consulta e os recursos de memória e CPU que podem ser usados por um pool de recursos. Desta forma, você pode limitar e isolar os recursos ocupados por diferentes cargas de trabalho.

O GaussDB(DWS) usa cgroups para gerenciar e controlar os recursos da CPU, envolvendo os subsistemas da CPU, cpuacct e cpuset. O compartilhamento de CPU é implementado com base no subsistema da CPU cpu.shares. As vantagens do compartilhamento de CPU são as seguintes: O controle da CPU não é acionado quando a CPU do sistema operacional não está totalmente ocupada. O limite de CPU é implementado com base no cpuset, que é um subsistema de CPU usado para monitorar o uso de recursos da CPU.

Ao adicionar um pool de recursos no console de gerenciamento do GaussDB(DWS), você deve escolher entre **Share** e **Limit**.



Compartilhamento de CPU

CPU Share: percentual de tempo de CPU que pode ser usado por usuários associados ao pool de recursos atual para executar trabalhos.

O compartilhamento tem dois significados:

- **Share**: a CPU é compartilhada por todos os Cgroups, e outros Cgroups podem usar recursos de CPU ociosos.
- **Limit**: quando a CPU está totalmente carregada durante as horas de pico, os Cgroups antecipam os recursos da CPU com base em seus limites.

O compartilhamento de CPU é implementado com base em cpu.shares e entra em vigor apenas quando a CPU está totalmente carregada. Quando a CPU está ociosa, não há garantia de que um Cgroup irá antecipar os recursos da CPU apropriados à sua cota. Ainda pode haver contenção de recursos quando a CPU está ociosa. Tarefas em um Cgroup podem usar recursos da CPU sem restrições. Embora o uso médio da CPU não seja alto, a contenção de recursos da CPU ainda pode ocorrer em um momento específico.

Por exemplo, 10 trabalhos estão sendo executados em 10 CPUs e um trabalho está sendo executado em cada CPU. Nesse caso, qualquer solicitação de trabalho para recursos da CPU

será respondida instantaneamente e não há disputa. Se 20 trabalhos estiverem sendo executados em 10 CPUs, o uso da CPU ainda pode não ser alto porque os trabalhos nem sempre ocupam a CPU e podem esperar por recursos de I/O e de rede. Os recursos da CPU parecem ociosos. No entanto, se 2 ou mais trabalhos solicitarem uma CPU ao mesmo tempo, ocorrerá a contenção de recursos da CPU, afetando o desempenho do trabalho.

Limite de CPU

CPU Limit: especifica a porcentagem do número máximo de núcleos de CPU que podem ser usados por um usuário do banco de dados no pool de recursos.

O limite tem dois significados:

- Dedicated: a CPU é dedicada a um Cgroup. Outros Cgroups não podem usar recursos de CPU ociosos.
- Quota: somente os recursos da CPU na cota alocada podem ser usados. Os recursos de CPU ociosos de outros Cgroups não podem ser preempcionados.

O limite de CPU é implementado com base em cpuset.cpu. Você pode definir uma cota adequada para implementar o isolamento absoluto dos recursos da CPU entre Cgroups. Desta forma, as tarefas de diferentes Cgroups não afetarão umas às outras. No entanto, o isolamento absoluto da CPU fará com que os recursos da CPU ociosa em um Cgroup sejam desperdiçados. Portanto, o limite não pode ser muito grande. Um limite maior pode não trazer um melhor desempenho.

Por exemplo, em um caso, 10 trabalhos estão sendo executados em 10 CPUs e o uso médio da CPU é de cerca de 5%. Em outro caso, 10 trabalhos estão sendo executados em 5 CPUs e o uso médio da CPU é de cerca de 10%. De acordo com a análise anterior, embora o uso da CPU seja baixo quando 10 trabalhos são executados em cinco CPUs. No entanto, a contenção de recursos da CPU ainda existe. Portanto, o desempenho da execução de 10 trabalhos em 10 CPUs é melhor do que o da execução de 10 trabalhos em 5 CPUs. No entanto, não é quanto mais CPUs, melhor. Se dez trabalho forem executados em 20 CPUs, em qualquer ponto de tempo, pelo menos 10 CPUs ficarão ociosas. Portanto, teoricamente, executar 10 trabalhos em 20 CPUs não tem melhor desempenho do que executar 10 CPUs. Para um grupo C com uma simultaneidade de N, se o número de CPUs alocadas for menor que N, o desempenho do trabalho será melhor com mais CPUs. No entanto, se o número de CPUs alocadas for maior que N, o desempenho do trabalho não será melhorado com mais CPUs.

Cenários de aplicações do gerenciamento de recursos da CPU

O limite de CPU e o compartilhamento de CPU têm suas próprias vantagens e desvantagens. O compartilhamento de CPU pode utilizar totalmente os recursos da CPU. No entanto, os recursos de diferentes Cgroups não são completamente isolados, o que pode afetar o desempenho da consulta. O limite da CPU pode implementar o isolamento absoluto dos recursos da CPU. No entanto, os recursos de CPU ociosos serão desperdiçados. Em comparação com o limite de CPU, o compartilhamento de CPU tem maior uso de CPU e taxa de transferência geral de trabalhos. Comparado com o compartilhamento de CPU, o limite de CPU tem isolamento completo da CPU, o que pode atender melhor aos requisitos dos usuários sensíveis ao desempenho.

Se a disputa da CPU ocorrer quando vários tipos de trabalhos estiverem em execução no sistema de banco de dados, você poderá selecionar diferentes modos de controle de recursos da CPU com base em diferentes cenários.

- Cenário 1: utilize totalmente os recursos da CPU. Concentre-se na taxa de transferência geral da CPU em vez do desempenho de um único tipo de trabalho.

Sugestão: não é aconselhável isolar CPUs entre usuários. Não importa qual tipo de controle da CPU seja implementado, o uso geral da CPU é afetado.

- Cenário 2: um certo grau de contenção de recursos da CPU e perda de desempenho são permitidos. Quando a CPU está ociosa, os recursos da CPU são totalmente utilizados. Quando a CPU está totalmente carregada, cada tipo de serviço precisa usar a CPU proporcionalmente.

  Sugestão: você pode usar o compartilhamento de CPU para melhorar o uso geral da CPU enquanto implementa o isolamento e o controle da CPU quando as CPUs estão totalmente carregadas.

- Cenário 3: alguns trabalhos são sensíveis ao desempenho e o desperdício de recursos da CPU é permitido.

  Sugestão: você pode usar o limite de CPU para implementar o isolamento absoluto de CPU entre diferentes tipos de trabalhos.

8.7 Por que as tarefas executadas por um usuário comum são mais lentas do que as executadas pelo usuário dbadmin?

A velocidade de execução de um usuário comum é mais lenta que a do usuário dbadmin nos seguintes cenários:

Cenário 1: os usuários comuns estão sujeitos à gestão de recursos.

Enfileiramento de usuários comuns: **waiting in queue/waiting in global queue/waiting in ccn queue**

1. Os usuários comuns estarão **aguardando na fila/aguardando na fila global** quando o número de instruções ativas exceder o valor de **max_active_statements**. Enquanto os administradores não precisam enfileirar.

   Você pode aumentar o valor desse parâmetro ou limpar algumas declarações para evitar o enfileiramento.

   Altere o valor de **max_active_statements** no console de gerenciamento.

   a. Faça logon no console de gerenciamento do GaussDB(DWS).

   b. Na árvore de navegação à esquerda, escolha **Clusters** > **Dedicated Clusters**.

   c. Na lista de clusters, localize o cluster de destino e clique no nome do cluster. A página **Basic Information** é exibida.

   d. Vá para a página **Parameter Modifications** do cluster, procure o parâmetro **max_active_statements**, altere seu valor e clique em **Save**.

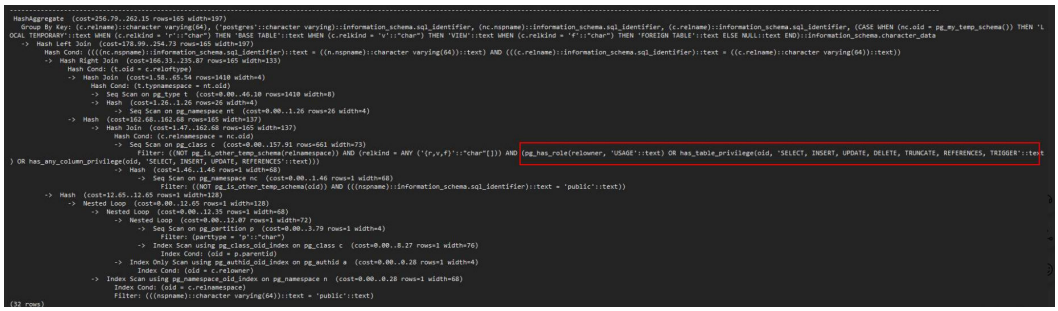
2. Demora muito tempo para que os usuários comuns aguardem na fila ccn. Quando o gerenciamento dinâmico de recursos está ativado(**enable_dynamic_workload** está definido como **on**), se a simultaneidade for alta e a memória disponível for pequena, os usuários comuns podem entrar nesse estado ao executar instruções. Os administradores não são controlados. Você pode parar algumas instruções ou aumentar o valor do parâmetro de memória. Se o uso de memória de cada DN não for alto, você poderá desativar o parâmetro de gerenciamento dinâmico de recursos **enable_dynamic_workload** definindo-o como **off**.

Cenário 2: a condição OR no plano de execução verifica as instruções executadas por usuários comuns, uma a uma. Isso consome muito tempo.

As condições **OR** nos planos de execução contêm verificações relacionadas à permissão. Esse cenário geralmente ocorre quando o modo de exibição do sistema é usado. Por exemplo, na seguinte instrução SQL:

```
SELECT distinct(dtp.table_name),
 ta.table_catalog,
 ta.table_schema,
 ta.table_name,
 ta.table_type
from information_schema.tables ta left outer join DBA_TAB_PARTITIONS dtp
on (dtp.schema = ta.table_schema and dtp.table_name = ta.table_name)
where ta.table_schema = 'public';
```

Parte do plano de execução é a seguinte:



Na exibição do sistema, a condição **OR** é usada para verificação de permissão.

```
pg_has_role(c.relowner, 'USAGE'::text) OR has_table_privilege(c.oid, 'SELECT,
INSERT, UPDATE, DELETE, TRUNCATE, REFERENCES, TRIGGER'::text) OR
has_any_column_privilege(c.oid, 'SELECT, INSERT, UPDATE, REFERENCES'::text)
```

true é sempre retornado para **pg_has_role** do uso de **dbadmin**. Portanto, as condições após **OR** não precisam ser verificadas.

Enquanto as condições **OR** de um usuário comum precisam ser verificadas uma por uma. Se houver um grande número de tabelas no banco de dados, o tempo de execução do usuário comum é maior que o do usuário **dbadmin**.

Nesse cenário, se o número de conjuntos de resultados de saída for pequeno, você poderá definir **set enable_hashjoin** e **enable_seqscan** como **off**, para usar o plano index+nestloop.

Cenário 3: os pools de recursos alocados para usuários comuns e administradores são diferentes.

Execute o comando a seguir para verificar se os pools de recursos correspondentes a um usuário comum são os mesmos do usuário administrador. Se eles forem diferentes, verifique se os recursos do locatário alocados aos dois usuários são diferentes.

```
SELECt * FROM pg_user;
```


8.8 Quais são os fatores relacionados ao desempenho da consulta de tabela única no GaussDB(DWS)?

O GaussDB(DWS) usa a arquitetura do nada compartilhado, e os dados são armazenados de maneira distribuída. Portanto, a chave de distribuição, o volume de dados e o número de partições afetam o desempenho geral da consulta de uma única tabela.

1. Design da chave de distribuição

   Por padrão, o GaussDB(DWS) usa a primeira coluna da chave primária como chave de distribuição. Quando você define uma chave primária e uma chave de distribuição para uma tabela, a chave de distribuição deve ser um subconjunto da chave primária. As chaves de distribuição determinam a distribuição de dados entre partições. Se as chaves de distribuição forem bem distribuídas entre partições, o desempenho da consulta pode ser melhorado.

   Se a chave de distribuição estiver selecionada incorretamente, a distorção de dados poderá ocorrer após a importação dos dados. O uso de alguns discos pode ser muito maior do que o de outros discos, e o cluster pode se tornar somente leitura em alguns casos extremos. A seleção adequada das chaves de distribuição é fundamental para o desempenho da consulta de tabela. Além disso, as chaves de distribuição adequadas permitem que os índices de dados sejam criados e mantidos mais rapidamente.

2. Volume de dados armazenados em uma única tabela

   Quanto maior a quantidade de dados armazenados em uma única tabela, pior o desempenho da consulta. Se uma tabela contém uma grande quantidade de dados, você precisa armazenar os dados em partições. Para converter uma tabela comum em uma tabela particionada, você precisa criar uma tabela particionada e importar dados para ela a partir da tabela comum. Ao projetar tabelas, planeje se usará tabelas particionadas com base nos requisitos de serviço.

   Para particionar uma tabela, siga os seguintes princípios:

   – Use campos com intervalos óbvios para particionamento, por exemplo, data ou região.
   – O nome da partição deve refletir as características de dados da partição. Por exemplo, seu formato pode ser características de Keyword+Range.
   – Defina o limite superior de uma partição para **MAXVALUE** para evitar excesso de dados.

3. Número de partições

   Tabelas e índices podem ser divididos em unidades menores e mais fáceis de gerenciar. Isso reduz significativamente o espaço de pesquisa e melhora o desempenho de acesso.

   O número de partições afeta o desempenho da consulta. Se o número de partições for demasiado pequeno, o desempenho da consulta poderá deteriorar-se.

   GaussDB(DWS) suporta particionamento de intervalos e particionamento de listas. No particionamento por intervalo, os registros são divididos e inseridos em várias partições de uma tabela. Cada partição armazena dados de um intervalo específico (intervalos em diferentes partições não se sobrepõem). O particionamento de lista é suportado apenas por clusters de 8.1.3 e versões posteriores.

Ao projetar um armazém de dados, você precisa considerar esses fatores e realizar experimentos para determinar o esquema de projeto ideal.

9 Backup e restauração do snapshot

9.1 Por que criar um snapshot automatizado é tão lento?

Isso acontece quando os dados a serem copiados são grandes. Os snapshots automatizados são backups incrementais e, quanto menor a frequência definida (por exemplo, uma semana), mais tempo levará. Aumente a frequência de backup para acelerar o processo.

A tabela a seguir lista as taxas de backup e restauração de snapshots. (As taxas são obtidas do ambiente de teste de laboratório com SSDs locais como mídia de backup. As taxas são somente para referência. A taxa real depende dos recursos de disco, rede e largura de banda.)

- Taxa de backup: 200 MB/s/DN
- Taxa de restauração: 125 MB/s/DN

9.2 Um snapshot de DWS tem a mesma função que um snapshot de EVS?

Não.

Os snapshots de GaussDB(DWS) são usados para restaurar todas as configurações e dados de serviço de um cluster. Os snapshots do EVS são usados para restaurar os dados de serviço de um disco de dados ou disco do sistema dentro de um período de tempo específico.

Snapshot do DWS

Um snapshot de GaussDB(DWS) é um backup completo ou incremental de um cluster do GaussDB(DWS) em um ponto específico no tempo. Ele registra os dados do banco de dados atual e as informações do cluster, incluindo o número de nós, as especificações do nó e o nome do administrador. Snapshots podem ser criados manualmente ou automaticamente.

Quando um snapshot é usado para restauração, o GaussDB(DWS) cria um novo cluster com base nas informações do cluster registradas no snapshot e restaura os dados do snapshot.

Para obter mais informações sobre snapshots, consulte **Gerenciamento de snapshots**.

Snapshot do EVS

Um snapshot do EVS é uma cópia ou imagem completa dos dados do disco obtidos em um ponto de tempo específico. O snapshot é uma abordagem importante de recuperação de desastres, e você pode restaurar completamente os dados de um snapshot até o momento em que o snapshot foi criado.

Você pode criar snapshots para salvar rapidamente os dados do disco em pontos de tempo especificados. Além disso, você também pode usar snapshots para criar novos discos para que os discos criados contenham os dados do snapshot no início.

Você pode criar snapshots para salvar rapidamente os dados do disco em pontos de tempo especificados para implementar a recuperação de desastres de dados.

- Se ocorrer perda de dados, você pode usar um snapshot para restaurar completamente os dados para o ponto de tempo em que o snapshot foi criado.
- Você pode usar snapshots para criar novos discos para que os discos criados contenham os dados do snapshot.

Para obter mais informações sobre snapshots, consulte **Snapshots do EVS**.

10 Cobrança

10.1 Como renovar o serviço?

Sobre a renovação

Atualmente, o GaussDB(DWS) suporta o modo de cobrança de pagamento por uso e o modo de cobrança anual/mensal.

- No modo anual/mensal, você paga apenas uma vez ao adquirir o serviço e nenhuma taxa extra será cobrada durante o uso do serviço. Depois que uma assinatura anual/mensal expirar, os recursos entrarão em um período de retenção. Se você quiser continuar usando os recursos, renove a assinatura.
- Modo de pagamento por uso: o sistema liquida as taxas por hora. Você pode usar os recursos de OBS, desde que o saldo da sua conta seja suficiente. Se o saldo da sua conta for insuficiente, ele estará em atraso. Carregue sua conta em tempo hábil.

Modos de renovação

Pay-per-use

Para recarregar sua conta, execute as seguintes etapas:

Passo 1 Faça logon no console de gerenciamento.

Passo 2 Escolha **Billing Center > Renewal** no canto superior direito da página.

Passo 3 No painel de navegação à esquerda, clique em **Overview**. Em seguida, clique em **Top Up** para que seu saldo seja suficiente para continuar os serviços.

----Fim

Yearly/Monthly

Execute as seguintes operações para renovar sua conta:

Passo 1 Faça logon no console de gerenciamento.

Passo 2 Clique em **Service List** à esquerda e vá para a página **Data Warehouse Service**.

Passo 3 Na página **Dedicated Clusters**, localize o cluster de destino e clique em **Renew**.

Passo 4 Conclua a renovação conforme solicitado.

----Fim

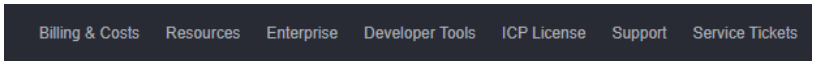
10.2 O reembolso é suportado?

Sim. O GaussDB (DWS) suporta reembolso total de 5 dias. Nos primeiros 5 dias após a compra, você pode solicitar um reembolso total. O pagamento integral será reembolsado, excluindo os cupons de dinheiro usados e cupons de desconto. Cada conta pode solicitar um reembolso total de 5 dias por até 10 vezes dentro de um ano civil (a partir de 1º de janeiro).

Cancelar a assinatura de um pacote de descontos

Passo 1 Faça logon no console de gerenciamento.

Passo 2 No canto superior direito, clique em **Billing & Costs**.



Passo 3 No painel de navegação à esquerda, escolha **Orders > Unsubscriptions**.

Passo 4 Localize o recurso na lista, clique em **Unsubscribe from Resources** na linha onde o recurso está localizado e cancele a assinatura do recurso conforme solicitado.

Para obter detalhes, consulte **Regras de cancelamento de assinatura**.

----Fim

10.3 Como experimentar o GaussDB(DWS) gratuitamente?

Somente novos usuários podem participar da campanha de teste gratuito. Se sua conta nunca criou um cluster de armazém de dados, mas passou na autenticação de nome real, você está qualificado para uma avaliação gratuita de um mês do GaussDB(DWS).

Para se inscrever para a avaliação gratuita, faça logon no console de gerenciamento do GaussDB(DWS) e clique em **Apply for free trial**. Os pacotes de avaliação gratuita não podem ser usados entre regiões. Selecione a região com base em suas necessidades.

Depois de assinar um pacote de avaliação gratuita, você pode fazer logon no console de gerenciamento do GaussDB(DWS) para criar um cluster com a região, o flavor de nó e a quantidade de nó correspondentes dentro do período de avaliação gratuita. O sistema não cobrará pelo cluster criado. Se você optar por usar outros tipos de nó, você será cobrado à taxa padrão de pagamento por uso. Para obter detalhes, consulte **Detalhes de preços do GaussDB(DWS)**.

Quando a avaliação gratuita de um mês terminar, você poderá excluir o cluster para evitar taxas. Você também pode manter o cluster, mas será cobrado pela taxa de pagamento por uso normal.

10.4 Por que as taxas foram deduzidas após uma avaliação gratuita do GaussDB (DWS) expirar?

Conforme indicado na página de atividade do GaussDB(DWS), o cluster criado não será liberado automaticamente após a expiração da avaliação gratuita. Se você não excluir o cluster manualmente, o sistema deduzirá a taxa do cluster.

Para evitar taxas adicionais, faça login no console de gerenciamento do GaussDB(DWS) e vá para a página **Clusters** para excluir o cluster se não quiser usá-lo após o término do período de avaliação. Se o cluster tiver um EIP, selecione **Release the EIP bound to the cluster.** na caixa de diálogo **Delete Cluster**. Se você não liberar o EIP, ele será cobrado conforme a regra de definição de preço do EIP da VPC.

10.5 Por que não poder ver um cluster após a assinatura de uma avaliação gratuita do GaussDB (DWS)?

O sistema não cria automaticamente um cluster após assinar uma avaliação gratuita. Faça login no console de gerenciamento do GaussDB(DWS) para criar um manualmente.

10.6 Como parar o faturamento do GaussDB(DWS)?

A seguir, descrevemos como interromper a cobrança em pagamento por uso e anual/mensal.

- **Pagamento por uso**

  Se você não usar mais um cluster de GaussDB(DWS) que está no modo de pagamento por uso, exclua-o e seus recursos para interromper a cobrança. Para fazer isso, faça logon no console de gerenciamento do GaussDB(DWS) e vá para a página **Clusters** > **Dedicated Clusters**. A tabela a seguir lista os itens faturáveis do GaussDB(DWS).

  Tabela 10-1 Itens faturáveis do GaussDB(DWS)

Item de cobrança	Descrição
Nó de armazém de dados	A cobrança será interrompida após o cluster ser excluído. Para obter detalhes sobre como excluir um cluster, consulte **Exclusão de um cluster**.
Espaço de armazenamento de snapshot	Quando um cluster é excluído, seus snapshots automatizados são excluídos, mas seus snapshots manuais são mantidos. Depois de excluir os snapshots manuais, a cobrança é interrompida. Para excluir os snapshots manuais, faça logon no console de gerenciamento do GaussDB(DWS) e vá para a página **Clusters** > **Dedicated Clusters**. Se os snapshots manuais ainda forem retidos após a exclusão do cluster e o tamanho total do snapshot exceder o espaço livre, a parte excedente será cobrada com base na taxa de cobrança do OBS.

Item de cobrança	Descrição
(Opcional) EIP e largura de banda	Se um cluster tiver um EIP, selecione **Release the EIP bound to the cluster.** ao excluir o cluster. A cobrança será interrompida após a liberação do EIP. Se você não liberar o EIP, ele será cobrado de acordo com as regras de definição de preço do EIP da VPC.
(Opcional) DEK	Se você ativou a função **Encrypt DataStore** para um cluster e comprou uma chave no DEW, a chave não será excluída após a exclusão do cluster. Cancele a assinatura e exclua manualmente a chave para interromper a cobrança. Para fazer isso, faça logon no console de gerenciamento do DEW e escolha **Data Encryption Workshop > Key Pair Service**.

- Pacote anual/mensal

  Os clusters do GaussDB(DWS) cobrados no modo anual/mensal suportam o cancelamento incondicional da assinatura no prazo de cinco dias a partir da sua compra. Para obter detalhes, consulte **Cancelamento de assinatura anual/mensal**. Se você não cancelar a assinatura do pacote, o sistema não reembolsará as taxas.

10.7 A cobrança de pagamento por uso pára quando o meu cluster é interrompido?

Não. O sistema liquida as taxas por hora para o modo de pagamento por uso. Você pode usar os recursos de OBS, desde que o saldo da sua conta seja suficiente. Para reduzir custos:

- Exclua os clusters comprados se você não precisar deles e crie novos quando necessário.
- Mude para o modo anual/mensal que lhe permite utilizar o serviço dentro do período de tempo específico sem taxas adicionais.

10.8 Por que o botão de compra não está disponível quando criar um cluster?

As possíveis causas para o botão de compra indisponível durante a criação do cluster são as seguintes:

- O flavor a ser comprado pode ser vendido, ou a região não tem esse flavor.

  Solução: antes de comprar, verifique se o flavor está disponível na região ou compre um pacote depois que o cluster for criado. Após o pacote, o pacote é automaticamente associado ao cluster.

- A conta pode estar em atraso ou restrita, portanto, novos recursos não podem ser criados.

  Solução: se o seu saldo for insuficiente, recarregue a conta para amortizar o valor em atraso.

10.9 Como descongelar um cluster?

Análise de causa

Se o saldo da sua conta for insuficiente e a dedução da taxa falhar, o período de retenção será iniciado. Durante o período de retenção, os recursos do serviço serão congelados e não poderão ser usados, mas os recursos e os dados serão reservados.

Procedimento de manuseio

Para descongelar um cluster, você precisa recarregar sua conta para garantir que o saldo da conta não seja **0**. Para obter detalhes, consulte **Como renovar o serviço?** Depois que um cluster é descongelado, o status do cluster é alterado para **Available**.

10.10 É possível congelar ou desligar um cluster do GaussDB(DWS) para interromper a cobrança?

Não. Se você não precisar de um cluster, exclua-o e cujos recursos.

