

云数据库 RDS for PostgreSQL

最佳实践

文档版本 01
发布日期 2025-04-09



版权所有 © 华为云计算技术有限公司 2025。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目录

1 最佳实践汇总	1
2 RDS for PostgreSQL 搭建跨区域容灾关系	2
2.1 方案概述	2
2.2 资源规划	3
2.3 操作流程	5
2.4 生产中心 RDS for PostgreSQL 实例准备	6
2.5 灾备中心 RDS for PostgreSQL 实例准备	9
2.6 配置跨区域网络互通	12
2.7 搭建容灾关系	15
2.8 灾备升主	18
2.9 解除灾备	18
2.10 常见问题	19
3 RDS for PostgreSQL 发布与订阅	20
4 RDS for PostgreSQL 自定义数据类型转换	23
5 使用客户端驱动程序实现故障转移和读写分离	25
6 PoWA 插件使用最佳实践	29
6.1 插件介绍	29
6.2 支持的性能指标	30
6.2.1 数据库级性能指标	30
6.2.2 实例级性能指标	33
6.3 部署 PoWA	36
6.3.1 云上 PostgreSQL 实例部署 PoWA	36
6.3.2 自建 PostgreSQL 实例部署 PoWA	38
6.4 在 PoWA 上查看指标详情	41
7 pg_dump 使用最佳实践	44
8 PgBouncer 使用最佳实践	48
9 RDS for PostgreSQL 安全最佳实践	51

1 最佳实践汇总

本文汇总了云数据库 RDS for PostgreSQL常见应用场景的操作实践，并为每个实践提供详细的方案描述和操作指导，帮助您轻松使用RDS for PostgreSQL。

表 1-1 RDS for PostgreSQL 最佳实践

相关文档	说明
RDS for PostgreSQL搭建跨区域容灾关系	介绍如何搭建RDS for PostgreSQL实例跨区域容灾。
RDS for PostgreSQL发布与订阅	介绍RDS for PostgreSQL提供的发布和订阅功能。
RDS for PostgreSQL自定义数据类型转换	介绍RDS for PostgreSQL自定义数据类型转换的语法使用。
使用客户端驱动程序实现故障转移和读写分离	介绍如何使用PostgreSQL客户端驱动程序实现故障转移和读写分离。
PoWA插件使用最佳实践	介绍如何使用PoWA插件，用于对RDS for PostgreSQL数据库进行性能监控。
pg_dump使用最佳实践	介绍pg_dump备份工具的使用方法。
PgBouncer使用最佳实践	介绍PgBouncer连接池工具的安裝配置和使用方法。
RDS for PostgreSQL安全最佳实践	提供RDS for PostgreSQL安全配置的规范性指导。

2 RDS for PostgreSQL 搭建跨区域容灾关系

2.1 方案概述

场景描述

当生产机发生损坏或因其他不可抗力造成业务系统宕机的情况下，异地跨区域容灾实例可以保证生产系统的数据不丢失，保持生产系统的业务不间断地运行，从而提高系统的可用性。

本实践主要包含以下内容：

- 介绍如何创建RDS for PostgreSQL实例。
- 介绍如何搭建RDS for PostgreSQL实例跨区域容灾。

前提条件

- 拥有华为云实名认证账号。
- 账户余额大于等于0美元。

约束条件

- 主实例和灾备实例状态正常，主实例和灾备实例在不同云或不同区域上，且主实例为主备实例，灾备实例为单机实例。
- 主实例配置容灾能力成功后才能配置灾备实例容灾能力，否则容灾关系会建立失败。
- 灾备实例的规格要大于等于主实例的规格。
- RDS for PostgreSQL 12及以上支持建立跨云或跨区域容灾关系。
- 灾备实例的底层架构和数据库大版本要与主实例一致。
- 如果灾备实例与主实例的小版本不一致，则在灾备搭建完成后，会自动重置灾备实例小版本，并与主实例的小版本保持一致。
- 不支持跨大版本建立跨云或跨区域容灾关系。
- 主实例和灾备实例的容灾关系已建立完成，才能进行灾备实例升主和查询容灾复制状态。

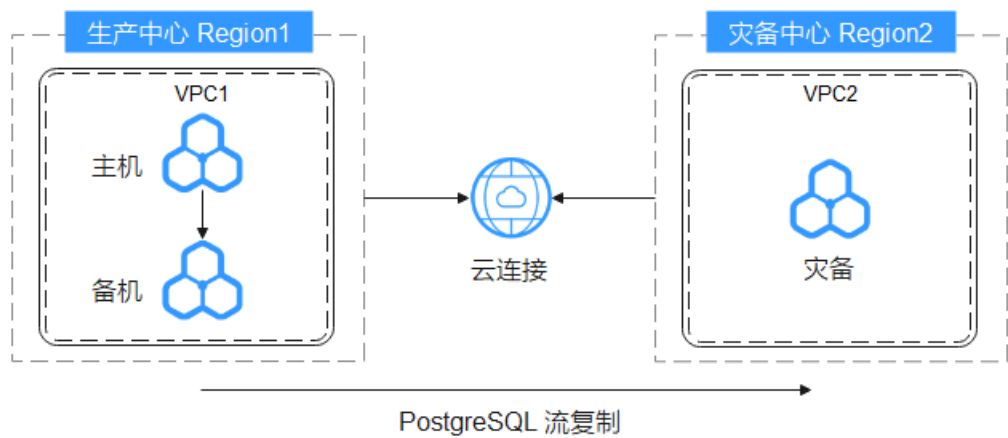
- 实施前确认需要搭建的主实例和灾备实例所在区域，处于云连接/虚拟专有网络服务已上线区域内。
- 灾备实例不支持PITR恢复和CBR快照备份功能，如需使用此功能，请在主实例上完成。

实现原理

RDS for PostgreSQL跨区域容灾实现原理说明：

在两个数据中心独立部署RDS for PostgreSQL实例，通过RDS接口将生产中心RDS for PostgreSQL库中的数据同步到灾备中心RDS for PostgreSQL库中，实现RDS for PostgreSQL主实例和跨区域灾备实例之间的实时同步。使用该功能前，必须需要确认可以使用云连接服务完成跨区域网络连通。

图 2-1 原理图



服务列表

- 云连接 CC
- 虚拟私有云 VPC
- 云数据库 RDS

使用说明

- 本实践的资源规划仅作为演示，实际业务场景资源以用户实际需求为准。
- 本实践端到端的数据为测试数据，仅供参考。

2.2 资源规划

表 2-1 资源规划

类别	子类	规划	备注
生产中心 VPC	VPC名称	vpc-pg-01	自定义，易理解可识别。

类别	子类	规划	备注
	所属Region	中国-香港	选择和自己业务区最近的Region，减少网络时延。
	可用区	可用区一	-
	子网网段	192.168.10.0/24	子网选择时建议预留足够的网络资源。
	子网名称	subnet-2aa1	自定义，易理解可识别。
灾备中心 VPC	VPC名称	vpc-pg-02	自定义，易理解可识别。
	所属Region	亚太-新加坡	选择和自己业务区最近的Region，减少网络时延。
	可用区	可用区一	-
	子网网段	192.168.20.0/24	子网选择时建议预留足够的网络资源。
	子网名称	subnet-a388	自定义，易理解可识别。
生产中心 RDS for PostgreSQL实例	RDS实例名称/ID	rds-pg-01 04**in03	自定义，易理解可识别。
	所属Region	中国-香港	选择和自己业务区最近的Region，减少网络时延。
	数据库版本	PostgreSQL 12	-
	内网IP	192.168.10.117	-
	实例类型	主备	生产中心实例为主备。
	存储类型	SSD云盘	-
	可用区	可用区一、可用区三	-
	性能规格	独享型 2 vCPUs 4GB	-
	磁盘容量	40 GB	-
灾备中心 RDS for PostgreSQL实例	RDS实例名称/ID	rds-pg-02 5f**in03	自定义，易理解可识别。
	所属Region	亚太-新加坡	选择和自己业务区最近的Region，减少网络时延。
	数据库版本	PostgreSQL 12	-
	内网IP	192.168.20.69	-
	实例类型	单机	灾备中心实例为单机。
	存储类型	SSD云盘	-

类别	子类	规划	备注
	可用区	可用区一	-
	性能规格	独享型 8 vCPUs 16GB	灾备中心实例的CPU和内存规格要大于或等于主实例的规格。
	磁盘容量	100 GB	灾备中心实例的磁盘容量要大于或等于主实例的磁盘容量。

2.3 操作流程

创建RDS for PostgreSQL业务实例以及灾备实例，并且将业务实例数据迁移到灾备实例的整个流程的主要任务流如下图所示。

图 2-2 流程图



2.4 生产中心 RDS for PostgreSQL 实例准备

本章节介绍在生产中心创建RDS for PostgreSQL实例所属VPC和安全组，然后创建RDS for PostgreSQL业务实例。

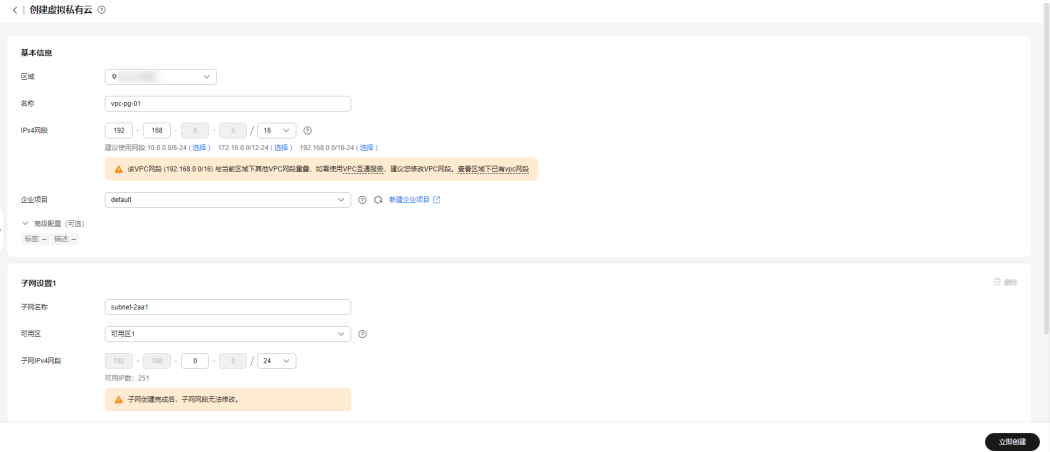
- **步骤1：创建VPC和安全组**
- **步骤2：创建RDS for PostgreSQL实例**

步骤 1：创建 VPC 和安全组

步骤1 进入[创建虚拟私有云](#)页面。

步骤2 在“创建虚拟私有云”页面，根据页面完成基本信息、子网配置和地址配置。选择区域“中国-香港”。

图 2-3 创建 VPC

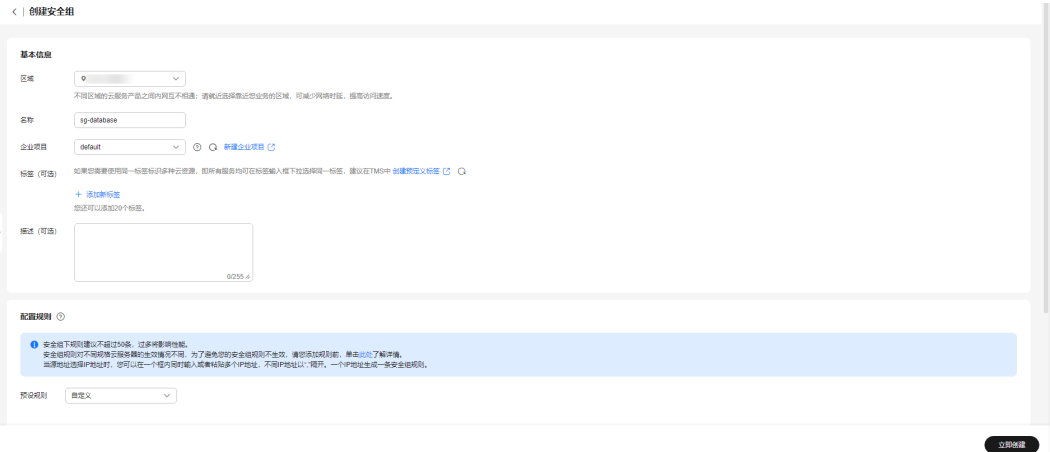


步骤3 单击“立即创建”，完成生产VPC创建。

步骤4 在网络控制台左侧导航树，选择“访问控制 > 安全组”。

步骤5 单击“创建安全组”。

图 2-4 创建安全组



步骤6 单击“立即创建”，完成生产安全组创建。

----结束

步骤 2：创建 RDS for PostgreSQL 实例

- 步骤1 进入[购买云数据库RDS页面](#)。
- 步骤2 选择区域“中国-香港”。填选实例信息后，单击“立即购买”。

图 2-5 选择引擎版本信息

计费模式

包年/包月

按需计费

区域

华东-上海一

项目

华东-上海一

实例名称

rds-bdf6

数据库引擎

MySQL

PostgreSQL

Microsoft SQL Server

MariaDB

数据库版本

15

14

13

12

11

10

实例类型

主备

单机

存储类型

SSD云盘

极速型SSD

主可用区

可用区1

可用区3

可用区2

可用区4

备可用区

可用区1

可用区3

可用区2

可用区4

时区

(UTC+08:00) 北京, 重庆, 香港, ...

图 2-6 选择规格信息

性能规格

独享型

通用型

vCPUs | 内存

建议连接数

2 vCPUs | 4 GB

400

2 vCPUs | 8 GB

800

2 vCPUs | 16 GB

1,600

4 vCPUs | 8 GB

800

4 vCPUs | 16 GB

1,600

4 vCPUs | 32 GB

3,200

当前选择实例 rds.pg.x1.large.2.ha | 2 vCPUs | 4 GB (独享型), 建议连接数 400

建议连接数的值的并不等同于数据库允许的最大连接数。建议连接数是结合系统资源等因素综合考虑的参考值；数据库允许的最大连接数（由max_connections决定）可以在实例详情中的参数修改页查看或修改。

40 GB

40

830

1620

2410

4000

40

GB

磁盘加密

不加密

加密

图 2-7 选择已规划的网络信息

虚拟私有云

vpc-pg-01

C

subnet-2aa1(192.168.10.0/24)

C

.

.

.

查看已使用IP地址 (可用私有IP数量251个)

目前RDS实例创建完成后不支持切换虚拟私有云与子网，请谨慎选择。不同虚拟私有云里面的弹性云服务器网络默认不通。如需创建新的虚拟私有云，可前往控制台创建。
通过公网访问数据库实例需要购买绑定弹性公网EIP。查看弹性公网IP

安全组

sg-database

C

查看内网安全组

创建安全组

安全组规则详情 ^ 设置规则

图 2-8 设置管理员密码

设置密码

现在设置

创建后设置

管理员账户名

root

管理员密码

请妥善保管密码，系统无法获取您设置的密码内容。

确认密码

参数模板

Default-PostgreSQL-12

C

查看参数模板

企业项目

default

C

查看项目管理

标签

如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议创建定义标签。查看预定义标签
在下方键/值输入框输入内容后单击“添加”，即可将标签加入此处

请输入标签键

请输入标签值

添加

您还可以添加20个标签。

购买数量

-

1

+

您还可以创建49个数据库实例，包括主实例和只读实例。如需申请更多配额请点击申请扩大配额。

- 步骤3 进行规格确认。
- 如果需要重新选择实例规格，单击“上一步”，回到上个页面修改实例信息。
 - 如果规格确认无误，单击“提交”，完成购买实例的申请。
- 结束

2.5 灾备中心 RDS for PostgreSQL 实例准备

本章节介绍在灾备中心创建RDS for PostgreSQL实例所属VPC和安全组，然后创建RDS for PostgreSQL业务实例。

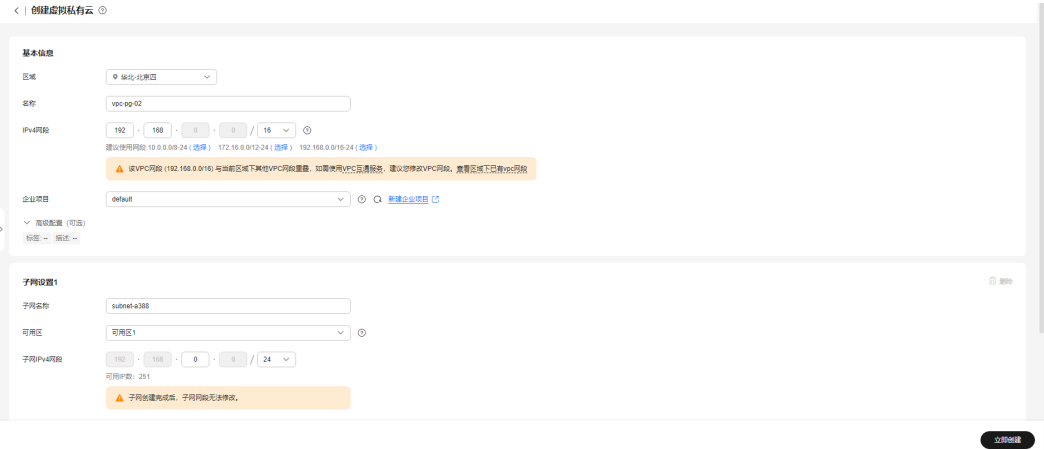
须知

- 灾备中心实例的VPC子网网段需与生产中心VPC子网网段不同，这是实现跨区域网络连接的前提。
 - 生产中心和灾备中心的安全组需要相互放通实例所属VPC子网网段的数据库端口。
- 步骤1：创建VPC和安全组
 - 步骤2：创建RDS for PostgreSQL实例

步骤 1：创建 VPC 和安全组

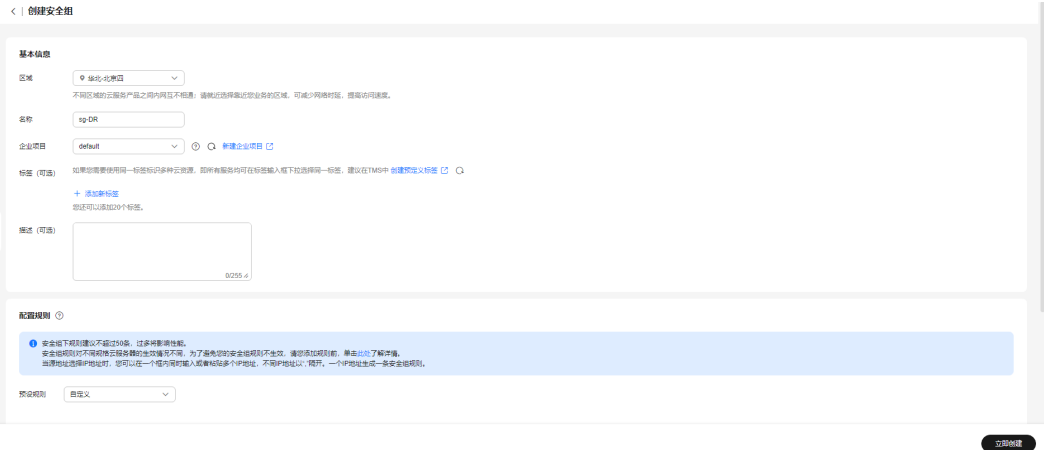
- 步骤1 进入[创建虚拟私有云](#)页面。
- 步骤2 在“创建虚拟私有云”页面，根据页面完成基本信息、子网配置和地址配置。选择区域“亚太-新加坡”。

图 2-9 创建 VPC



- 步骤3 单击“立即创建”，完成生产VPC创建。
- 步骤4 在网络控制台左侧导航树，选择“访问控制 > 安全组”。
- 步骤5 单击“创建安全组”。

图 2-10 创建安全组



步骤6 单击“立即创建”，完成生产安全组创建。

----结束

步骤 2：创建 RDS for PostgreSQL 实例

- 步骤1 进入[购买云数据库RDS页面](#)。
- 步骤2 选择区域“亚太-新加坡”。填选实例信息后，单击“立即购买”。

图 2-11 选择引擎版本信息

计费模式

包年/包月

按需计费

区域

华北-北京四

项目

华北-北京四

实例名称

rds-pg-02

数据库引擎

MySQL

PostgreSQL

Microsoft SQL Server

MariaDB

数据库版本

15

14

13

12

11

10

实例类型

主备

单机

存储类型

SSD云盘

可用区

可用区一

可用区二

可用区三

可用区七

时区

(UTC+08:00) 北京、重庆、香港、...

图 2-12 选择规格信息

性能规格

独享型

通用型

vCPUs | 内存

建议连接数

8 vCPUs | 16 GB

1,600

8 vCPUs | 32 GB

3,200

8 vCPUs | 64 GB

6,400

16 vCPUs | 32 GB

3,200

16 vCPUs | 64 GB

6,400

16 vCPUs | 128 GB

12,800

存储空间

100 GB

40

830

1620

2410

4000

100

GB

磁盘加密

不加密

加密

图 2-13 选择已规划的网络信息

虚拟私有云

vpc-pg-02

C

subnet-a388(192.168.20.0/24)

C

查看已使用IP地址 (可用私有IP数量251个)

目前RDS实例创建完成后不支持切换虚拟私有云与子网，请谨慎选择。不同虚拟私有云里面的弹性云服务器网络默认不通。如需创建新的虚拟私有云，可前往控制台创建。
通过公网访问数据库实例需要购买绑定弹性公网IP。查看弹性公网IP

安全组

default

C

查看内网安全组

创建安全组

安全组规则详情 ^ 设置规则

图 2-14 设置管理员密码

设置密码

现在设置

创建后设置

管理员账户名

root

管理员密码

请妥善保管密码，系统无法获取您设置的密码内容。

确认密码

参数模板

Default-PostgreSQL-12

C

查看参数模板

企业项目

default

C

查看项目管理

标签

如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议创建预定义标签。
在下方键/值输入框输入内容后单击添加，即可将标签加入此处

请输入标签键

请输入标签值

添加

您还可以添加20个标签。

购买数量

-

1

+

您还可以创建49个数据库实例，包括主实例和只读实例。如需申请更多配额请点击申请扩大配额。

- 步骤3 进行规格确认。
- 如果需要重新选择实例规格，单击“上一步”，回到上个页面修改实例信息。
 - 如果规格确认无误，单击“提交”，完成购买实例的申请。
- 结束

2.6 配置跨区域网络互通

搭建异地容灾实例，首先需要完成跨区域网络的连通。当前提供**方式一：通过云连接配置跨区域VPC互通**和**方式二：通过虚拟专用网络配置跨区域VPC互通**两种方式实现跨区域网络连通。

在选择带宽大小时，建议根据事务日志生成速率监控指标进行选择，大于等于该指标最高值的10倍即可，因为网络带宽单位为Mbit/s，而事务日志生成速率监控指标的单位为MB/s。

例如事务日志生成速率监控指标最高值为10MB/s，则网络带宽建议选择100Mbit/s，以便于灾备节点有足够的带宽可以及时同步主节点的数据。

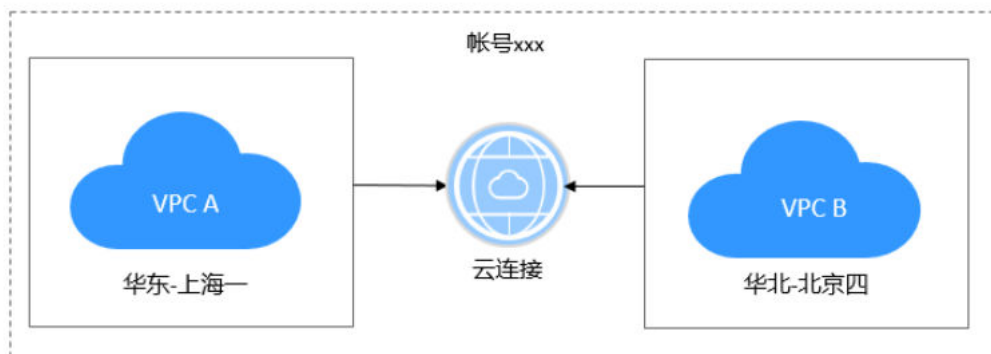
在网络打通后，需要配置主实例和容灾实例互相放通安全组，详见[配置安全组](#)。

方式一：通过云连接配置跨区域 VPC 互通

搭建异地容灾实例，首先需要完成跨区域网络的连通。

可以使用[云连接 CC](#)产品完成跨区域VPC网络连通。

图 2-15 跨区域同账号互通



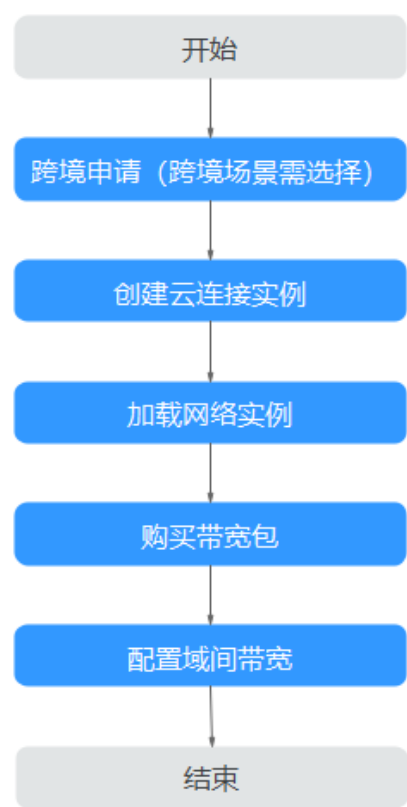
须知

配置前请确认搭建的主实例和灾备实例在云连接[已上线区域清单](#)列表上。

配置云连接两端需放通主实例和灾备实例所属VPC子网的网段，仅放通单个数据库IP会搭建失败。

跨区域VPC网络连通具体操作步骤可以按照[跨区域同账号VPC互通](#)进行配置。

图 2-16 云连接流程图



方式二：通过虚拟专用网络配置跨区域 VPC 互通

可以使用[虚拟专用网络 VPN](#)服务完成跨区域VPC网络连通。

须知

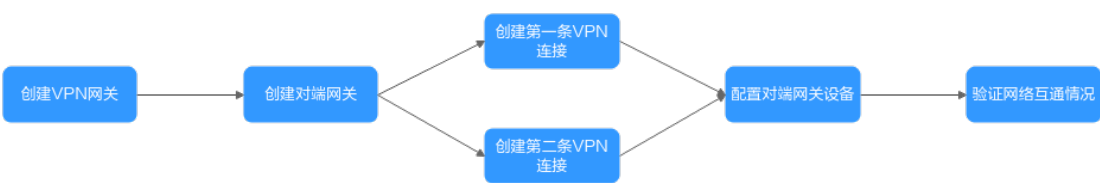
配置前请确认搭建的主实例和灾备实例在虚拟专用网络[功能支持区域](#)列表上。

当前在使用虚拟专用网络连通时，需要在配置完VPN服务后，联系VPN服务客服，进行网络配置调配，才能使用。

配置跨VPC两端需放通主实例和灾备实例所属VPC子网的网段，仅放通单个数据库IP会搭建失败。

虚拟专用网络的具体操作步骤可以参考[虚拟专用网络入门指引](#)进行配置。

图 2-17 操作流程



配置安全组

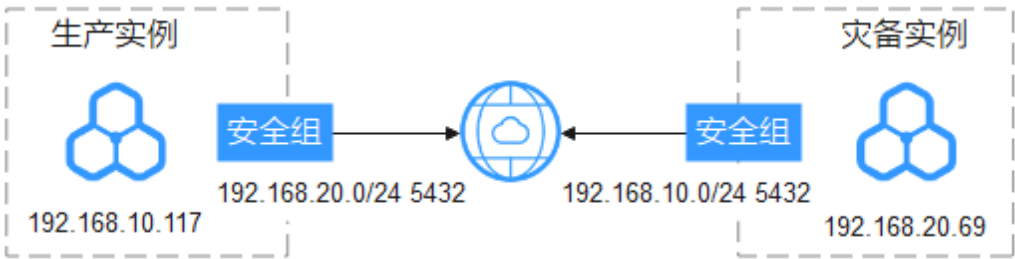
完成跨区域VPC互通后，还需要配置主实例和灾备实例的安全组，使得VPC网段间的端口互通。

假设存在表2-2所示的实例配置，其数据库端口都为默认值5432。则生产实例和灾备实例的的防火墙配置如图2-18所示。

表 2-2 实例网段

类别	VPC网段	IP地址
生产实例	192.168.10.0/24	192.168.10.117
灾备实例	192.168.20.0/24	192.168.20.69

图 2-18 防火墙配置



2.7 搭建容灾关系

操作场景

建立跨区域容灾关系后，当主实例所在区域发生突发性自然灾害等状况，主节点无法连接，可将异地灾备实例升为主实例，在应用端修改数据库连接地址后，即可快速恢复应用的业务访问。

注意事项

- 使用该功能前，必须要确保跨区域数据库实例之间的网络打通，可以使用[云连接 CC](#)或[虚拟专用网络 VPN](#)产品完成跨区域VPC网络连通。
- 使用该功能前，确保主实例和灾备实例状态正常，主实例和灾备实例在不同区域上，且主实例为主备实例，灾备实例为单机实例。
- 灾备实例的CPU和内存规格以及磁盘容量要大于或等于主实例的规格以及磁盘容量。
- 灾备实例的底层架构和数据库大版本要与主实例一致。
- 不支持跨大版本建立跨云或跨Region容灾关系。
- 调用配置主实例容灾接口后直至成功搭建容灾关系，不能进行规格变更、主备切换操作。
- 搭建容灾后，灾备实例支持变更CPU和内存规格，如需使用该功能，请[提交工单](#)申请。

- 修改主实例的端口或内网地址后需要重新搭建灾备关系。
- 灾备实例搭建成功后，不能进行小版本升级。
- RDS for PostgreSQL 12及以上支持建立跨区域容灾关系。
- 主实例参数被修改后，灾备实例无法同步修改该参数，需结合业务自行修改灾备实例参数。
- RDS for PostgreSQL灾备实例不支持PITR恢复和CBR快照备份功能，如需使用此功能，请在主实例上完成。

操作步骤

步骤1 配置生产实例容灾能力，即将灾备实例的配置信息复制到生产实例上。

1. [登录管理控制台](#)。
2. 单击管理控制台左上角的，选择灾备实例所在的区域，例如“亚太-新加坡”。
3. 单击页面左上角的，选择“数据库>云数据库 RDS”，进入RDS信息页面。
4. 在“实例管理”页面，单击灾备实例的实例名称进入“概览”页面。
5. 单击“灾备配置信息”。
6. 在弹框中，单击“一键复制”。

图 2-19 复制灾备实例配置信息



7. 单击管理控制台左上角的，选择生产实例所在的区域，例如“中国-香港”。
8. 在“实例管理”页面，选择生产实例，单击“操作”列的“更多 > 查看容灾详情”，进入“容灾管理”页面。

9. 单击“搭建容灾”，在弹框中，将步骤1.6中复制的灾备配置信息粘贴至输入框中，单击“确定”，开始配置生产实例容灾能力。

图 2-20 粘贴灾备实例信息



10. 在生产实例的“容灾管理”页面查看搭建情况。当“搭建状态”显示为“已搭建”，则表示配置生产实例容灾能力成功。确保此步骤搭建成功后再进行后续操作。

图 2-21 查看搭建容灾成功



步骤2 配置灾备实例容灾能力，即将生产实例的配置信息复制到灾备实例上。

1. 在“实例管理”页面，单击生产实例的实例名称进入“概览”页面。
2. 单击“灾备配置信息”。
3. 在弹框中，单击“一键复制”。
4. 单击管理控制台左上角的📍，选择灾备实例所在的区域，例如“亚太-新加坡”。
5. 在“实例管理”页面，选择灾备实例，单击“操作”列的“更多 > 查看容灾详情”，进入“容灾管理”页面。
6. 单击“搭建容灾”，在弹框中，将步骤2.3中复制的灾备配置信息粘贴至输入框中，单击“确定”，开始配置灾备实例容灾能力。

7. 在“容灾管理”页面查看搭建情况。当“搭建状态”显示为“已搭建”，则表示配置灾备实例容灾能力成功，至此，容灾搭建成功。
8. 在“容灾管理”页面可以查看灾备复制状态、发送延迟大小、端到端延迟大小和回放延迟时间。

----结束

2.8 灾备升主

操作场景

当主实例所在区域发生突发性自然灾害等状况，主节点无法连接，可将异地灾备实例升为主实例，在应用端修改数据库连接地址后，即可快速恢复应用的业务访问。

注意事项

灾备升主后的新主实例和原主实例将会解除灾备关系。

操作步骤

步骤1 [登录管理控制台](#)。

步骤2 单击管理控制台左上角的，选择灾备实例所在的区域，例如“亚太-新加坡”。

步骤3 单击页面左上角的，选择“数据库>云数据库 RDS”，进入RDS信息页面。

步骤4 单击灾备实例名称，进入实例的概览页面。

步骤5 在左侧导航栏，选择“容灾管理”。

步骤6 在容灾关系列表单击“操作”列的“灾备升主”。

步骤7 在弹框中，单击“确认”，开始下发灾备升主任务。

步骤8 可在“任务中心”中查看任务的执行结果，当任务状态为完成时，表示灾备升主任务成功。

步骤9 需用户在应用端修改数据库连接地址，手动将业务切换到新主实例。

----结束

2.9 解除灾备

操作场景

当不需要容灾关系时，可以将搭建好的容灾关系解除。

注意事项

当前只支持解除搭建成功的容灾关系，并且需要先解除灾备实例的容灾关系，再解除主实例的容灾关系，否则可能会引起异常告警。

操作步骤

步骤1 [登录管理控制台](#)。

步骤2 首先解除灾备实例的容灾关系。

1. 进入主实例的概览页面，单击“灾备配置信息”。
2. 在弹框中，单击“一键复制”。
3. 单击灾备实例名称，进入实例的概览页面。
4. 在左侧导航栏，选择“容灾管理”。
5. 在容灾关系列表单击“操作”列的“解除容灾关系”。
6. 将刚复制的灾备配置信息粘贴进弹框中。
7. 可在“容灾管理”中查看任务的执行结果，当列表被删除时，则执行成功。

步骤3 然后解除主实例的容灾关系，步骤参考[步骤2](#)，复制灾备实例的“灾备配置信息”，然后在主实例的“容灾管理”页面解除。

----结束

2.10 常见问题

- 问题1：配置灾备实例容灾能力时，任务执行失败。
排查生产中心和灾备中心的安全组是否互相放通数据库VPC子网网段的端口。如果使用VPN连接，确认是否按照[方式二：通过虚拟专用网络配置跨区域VPC互通](#)的注意事项配置。
- 问题2：使用云连接配置VPC时，系统提示“当前VPC存在路由冲突，此VPC路由已存在”。
参考[加载网络实例时出现系统异常怎么办](#)进行排查处理。

3 RDS for PostgreSQL 发布与订阅

逻辑定义

发布可以被定义在任何物理复制的主服务器上。定义有发布的节点被称为发布者。发布是从一个表或者一组表生成的改变的集合，也可以被描述为更改集合或者复制集合。每个发布都只存在于一个数据库中。

订阅是逻辑复制的下游端。订阅被定义在其中的节点被称为订阅者。一个订阅会定义到另一个数据库的连接以及它想要订阅的发布集合（一个或者多个）。逻辑订阅者的行为与一个普通的PostgreSQL实例（主库）无异，逻辑订阅者也可以创建自己的发布，拥有自己的订阅者。

使用权限

- 创建发布时，发布者必须有流复制权限。即replication权限。
- 创建发布时，使用all tables发布所有表时，需确保发布者是提权起始及之后版本的root用户。
- 创建/删除订阅时，需确保订阅者是提权起始及之后版本的root用户。
- 创建发布/订阅时，需确保发布端和订阅端实例在同一VPC下。

各个版本root用户提权情况参见[root用户权限说明](#)。

发布使用限制

- 发布目前只能包含表（即：索引，序列号，物化视图这些不会被发布），每个表可以添加到多个发布中。
- 一个publication允许有多个订阅者。
- 允许使用all tables发布所有表。
- 在同一个数据库中，可以创建多个publication，但是不能重名。已创建的publication可以通过查询pg_publication获取。
- 发布可以筛选所需的变更类型：包括insert、update、delete 和truncate的任意组合，类似触发器事件，默认所有变更都会被发布。

例如：发布表t1的**update**和**delete**操作。

```
CREATE PUBLICATION update_delete_only FOR TABLE t1
WITH (publish = 'update, delete');
```

- 复制标识：当发布了表的**update**, **delete**时，表必须设置复制标识（Replica Identity），如果设置了nothing，则执行**update**, **delete**时会报错。

表上的复制标识可以通过查阅pg_class.relreplident获取。

这是一个字符类型的“枚举”，标识用于组装“复制标识”的列：d = default，f = 所有的列，i 使用特定的索引，n 没有复制标识。

表上是否具有可用作复制标识的索引约束，可以通过以下查询获取：

```
SELECT quote_ident(nspname) || '.' || quote_ident(relname) AS name, con.ri AS keys,
       CASE relreplident WHEN 'd' THEN 'default' WHEN 'n' THEN 'nothing' WHEN 'f' THEN
'full' WHEN 'i' THEN 'index' END AS replica_identity
FROM pg_class c JOIN pg_namespace n ON c.relnamespace = n.oid, LATERAL (SELECT
array_agg(contype) AS ri FROM pg_constraint WHERE conrelid = c.oid) con
WHERE relkind = 'r' AND nspname NOT IN ('pg_catalog', 'information_schema', 'monitor',
'repack', 'pg_toast')
ORDER BY 2,3;
```

- 复制标识配置

表到复制标识可以通过ALTER TABLE进行修改。

```
ALTER TABLE table_name REPLICA IDENTITY_
{ DEFAULT | USING INDEX index_name | FULL | NOTHING };
-- 具体有四种形式
ALTER TABLE t_normal REPLICA IDENTITY DEFAULT;           -- 使用主键，如果没有主
键则为FULL
ALTER TABLE t_normal REPLICA IDENTITY FULL;               -- 使用整行作为标识
ALTER TABLE t_normal REPLICA IDENTITY USING INDEX t_normal_v_key; -- 使用唯一索引
ALTER TABLE t_normal REPLICA IDENTITY NOTHING;            -- 不设置复制标识
```

- 复制标识在实际使用中的注意事项

- 表上有主键，使用默认的default复制标识。
- 表上没有主键，但是有非空唯一索引，显式配置index复制标识。
- 表上既没有主键，也没有非空唯一索引，显式配置full复制标识（运行效率非常低，仅能作为兜底方案）。
- 其他所有情况，都无法正常完成逻辑复制功能。输出的信息不足，可能会报错。
- 特别需要注意：如果nothing复制标识的表纳入到逻辑复制中，对其进行删改会导致发布端报错。

订阅使用限制

- 为了确保使用Failover Slot，必须在发布端手工创建逻辑复制槽(Failover Slot)，并通过create_slot = false关联已有复制槽，如下：
CREATE SUBSCRIPTION sub1 CONNECTION 'host=192.168.0.1 port=5432 user=user1 dbname=db1' PUBLICATION pub_name with (create_slot = false,slot_name = FailoverSlot_name);
- 逻辑复制不会复制DDL变更，因此发布集中的表必须已经存在于订阅端上。
- 同一个数据库中，可以创建多个subscription，这些subscription可以来自一个或多个发布者。
- 订阅者的同一张表，不能接受来自同一个源的多个发布。
- 在创建subscription或者alter subscription时，可以使用enable来启用该订阅，或者使用disable暂停该订阅。
- 如果要完全删除订阅，使用**DROP SUBSCRIPTION**，注意，删除订阅后，本地的表不会被删除，数据也不会清除，仅仅是不再接收该订阅的上游信息。

须知

如果订阅与复制槽相关联，就不能在事务块内部执行**DROP SUBSCRIPTION**。可以使用**ALTER SUBSCRIPTION**取消关联复制槽。

删除订阅可参考以下步骤：

- a. 在订阅端查询订阅关联的复制槽。
select subname,subconninfo,subslotname from pg_subscription where subname = 'sub2';
 - subname为订阅者名称。
 - subconninfo为连接远程主机信息。
 - subslotname 为远程主机复制槽名称。
- b. 在订阅端执行**ALTER SUBSCRIPTION**取消关联复制槽并删除。
ALTER SUBSCRIPTION subname SET (slot_name = NONE);
DROP SUBSCRIPTION subname;
- c. 在发布端删除关联的复制槽。
select pg_drop_replication_slot(' slot_name');

语法参考

- 发布

CREATE PUBLICATION用于创建发布，**DROP PUBLICATION**用于移除发布，**ALTER PUBLICATION**用于修改发布。

发布创建之后，可以通过**ALTER PUBLICATION**动态地向发布中添加或移除表，这些操作都是事务性的。

- 订阅

CREATE SUBSCRIPTION用于创建订阅，**DROP SUBSCRIPTION**用于移除订阅，**ALTER SUBSCRIPTION**用于修改订阅。

订阅创建之后，可以通过**ALTER SUBSCRIPTION**随时**暂停**与**恢复**订阅。移除并重建订阅会导致**同步信息丢失**，这意味着相关数据需要重新进行同步。

具体使用说明请参考以下官方文档，以PostgreSQL 13版本为例：

- 创建发布：<https://www.postgresql.org/docs/13/sql-createpublication.html>
- 删除发布：<https://www.postgresql.org/docs/13/sql-droppublication.html>
- 修改发布：<https://www.postgresql.org/docs/13/sql-alterpublication.html>

4 RDS for PostgreSQL 自定义数据类型转换

简介

PostgreSQL数据类型有三种转换方式：隐式转换，赋值转换，显式转换。对应的转换类型在系统表“pg_cast”中分别对应：i（Implicit）、a（Assignment）、e（Explicit）。

- 隐式转换（Implicit）：同一类型间，低字节到高字节为隐式转换，比如int到bigint。
- 赋值转换（Assignment）：同一类型间，高字节到低字节为赋值转换，比如smallint到int。
- 显式转换（Explicit）：不同类型间，称为显示转换。

基本使用

1. 在进行数据类型转换前，可以通过如下命令查看RDS for PostgreSQL是否已经支持数据类型转换。

```
select * from pg_catalog.pg_cast ;
oid | castsource | casttarget | castfunc | castcontext | castmethod
-----+-----+-----+-----+-----+-----
11277 | 20 | 21 | 714 | a | f
11278 | 20 | 23 | 480 | a | f
11279 | 20 | 700 | 652 | i | f
11280 | 20 | 701 | 482 | i | f
.....
```

2. 通过如下命令查询int4是否支持转换text。

```
select * from pg_catalog.pg_cast where castsource = 'int4'::regtype and casttarget = 'bool'::regtype;
oid | castsource | casttarget | castfunc | castcontext | castmethod
-----+-----+-----+-----+-----+-----
11311 | 23 | 16 | 2557 | e | f
(1 row)
```

此时查出结果是默认支持的，转换类型是隐式转换。

如果没有内置的转换函数，需要自定义转换函数来支持这种转换，具体参考[自定义类型转换](#)。

自定义类型转换

- 通过双冒号方式进行强制转换

```
select '10'::int,'2023-10-05'::date;
int4 | date
-----+-----
```

```
10 | 2023-10-05  
(1 row)
```

- 通过类型转换函数**CAST**进行转换

```
select CAST('10' as int),CAST('2023-10-05' as date);
```

```
int4 | date
```

```
-----+-----
```

```
10 | 2023-10-05
```

```
(1 row)
```

- 自定义类型转换

具体语法及使用可查看：<https://www.postgresql.org/docs/14/sql-createcast.html>

须知

由于新增自定义类型转换会影响RDS for PostgreSQL已有的执行计划，一般不建议自定义类型转换。

- 时间与字符类型的转换

CREATE CAST(varchar as date) WITH INOUT AS IMPLICIT;

- boolean类型与数值类型转换

create cast(boolean as numeric) with INOUT AS IMPLICIT;

- 数值类型与字符类型转换

create cast(varchar as numeric) with INOUT AS IMPLICIT;

示例：将**text**转换为**date**

```
create or replace function public.text_to_date(text) returns date as
```

```
$$
```

```
select to_date($1,'yyyy-mm-dd');
```

```
$$
```

```
language sql strict;
```

```
create cast (text as date) with function public.text_to_date(text) as implicit;
```

```
select text '2023-09-09' + 1;
```

```
?column?
```

```
-----
```

```
2023-09-10
```

```
(1 row)
```

5

使用客户端驱动程序实现故障转移和读写分离

离

从PostgreSQL 10（libpq.so.5.10）开始，libpq驱动层开始支持故障转移和读写分离，JDBC驱动层则支持读写分离、故障转移和负载均衡。

PostgreSQL客户端连接程序向下兼容，对于RDS for PostgreSQL 9.5及9.6版本，使用新版本的libpq驱动程序也可以实现故障转移。

 说明

- 本章节中故障转移指的是读业务的故障转移。
- libpq是PostgreSQL的C应用程序接口，包含一组库函数，允许客户端程序将查询请求发送给PostgreSQL后端服务器并接收这些查询的结果。
 - JDBC是Java语言中用来规范客户端程序如何访问数据库的应用程序接口，在PostgreSQL中JDBC支持故障转移和负载均衡。

表 5-1 libpq 和 JDBC 驱动支持的功能

驱动	读写分离	负载均衡	故障转移
libpq驱动	√	×	√
JDBC驱动	√	√	√

libpq 实现故障转移和读写分离

通过libpq函数连接多个数据库，当出现故障时会自动切换到可用的数据库。

```
postgres://[user[:password]@][netloc][:port][,...][/dbname][?  
param1=value1&...]
```

示例：连接1个RDS for PostgreSQL主实例数据库和对应的2个只读实例数据库，只要确保至少有一个数据库可用，读请求就不会失败。

```
postgres://  
<instance_ip>:<instance_port>,<instance_ip>:<instance_port>,<instance_ip>:<instance_port>|<database_name>?target_session_attrs=any
```

表 5-2 参数说明

参数	说明	取值样例
<instance_ip>	数据库的主机 IP。	如果通过内网连接，“instance_ip”是主机 IP，即“概览”页面该实例的“内网地址”。 如果通过连接了公网的设备访问，“instance_ip”为该实例已绑定的“弹性公网 IP”。
<instance_port>	数据库端口。	默认 5432，当前端口，参考“概览”页面该实例的“数据库端口”。
<database_name>	数据库名，即需要连接的数据库名。	默认的管理数据库是 postgres，可根据业务实际情况填写数据库名。
target_session_attrs	允许连接到指定状态的数据库。	- any：默认值，表示允许连接到任意数据库，会连接到第一个允许连接的数据库，如果连接的数据库出现故障导致连接断开，会尝试连接其他数据库，从而实现故障转移。 - read-write：只会连接到支持读写的数据库，即从第一个数据库开始尝试连接，如果连接后发现不支持读写，则会断开连接，然后尝试连接第二个数据库，以此类推，直至连接到支持读写的数据库。 - read-only：只会连接只读数据库，即从第一个数据库开始尝试连接，如果连接后发现支持读写，则会断开连接，然后尝试连接第二个数据库，以此类推，直至连接到只读的数据库。RDS for PostgreSQL 13（libpq.so.5.13）及以下版本，不支持该取值。

更多 libpq 的使用方法和参数说明请参见 [Connection Strings](#)。

您还可以在应用程序中结合 pg_is_in_recovery() 函数，判断连接的数据库是主实例数据库（结果为“f”表示主数据库）还是只读实例数据库，进而实现读写分离。

使用 Python 代码的示例如下（psycopg2 使用的为 libpq）：

```
// 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放，使用时解密)，确保安全。  
// 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
```

```
import psycopg2  
import os  
  
username = os.getenv("EXAMPLE_USERNAME_ENV")  
password = os.getenv("EXAMPLE_PASSWORD_ENV")  
conn = psycopg2.connect(database=<database_name>, host=<instance_ip>, user=username,  
password=password, port=<instance_port>, target_session_attrs="read-write")  
cur = conn.cursor()  
cur.execute("select pg_is_in_recovery()")  
row = cur.fetchone()  
print("recovery =", row[0])
```

JDBC 实现故障转移和读写分离

您可以在连接URL中定义多个数据库（主机和端口），并用逗号分隔，驱动程序将尝试按顺序连接到它们中的每一个，直到连接成功。如果没有成功，会返回连接异常报错。

```
jdbc:postgresql://node1,node2,node3/${database}?  
targetServerType=preferSecondary&loadBalanceHosts=true
```

示例：

```
jdbc:postgresql://  
<instance_ip>:<instance_port>,<instance_ip>:<instance_port>,<instance_ip>:<instance_port>/<database_name>?  
targetServerType=preferSecondary&loadBalanceHosts=true
```

JDBC连接实例java实现代码请参考：[通过JDBC连接RDS for PostgreSQL实例](#)。

表 5-3 参数说明

参数	说明	取值样例
targetServerType	允许连接到指定状态的数据库。	- any：任何数据库。 - primary：主数据库（可写可读）。JDBC 42.2.0以下版本请使用参数值“master”。 - secondary：从数据库（可读）。JDBC 42.2.0以下版本请使用参数值“slave”。 - preferSecondary：优先从数据库，如果没有从数据库才连接到主数据库。JDBC 42.2.0以下版本请使用参数值“preferSlave”。
loadBalanceHosts	尝试连接数据库的顺序。	- False：默认值，按URL中的定义顺序连接数据库。 - True：随机连接数据库。

说明

区别数据库主从的方式是通过查询数据库是否允许写入，允许写入数据的判断为主数据库，不允许写入数据的判断为从数据库。参考[libpq实现故障转移和读写分离](#)中通过pg_is_in_recovery()函数来判断，结果为“f”表示为主数据库。

为实现读写分离，需要在配置JDBC时设置2个数据源，首先设置targetServerType=primary，用于写操作。另一个可以根据以下情况进行设置：

- 有一个只读实例，为实现高可用设置targetServerType=preferSecondary，用于读操作。假设主实例IP为10.1.1.1，只读实例IP为10.1.1.2。

```
jdbc:postgresql://10.1.1.2:5432,10.1.1.1:5432/${database}?  
targetServerType=preferSecondary
```

- 有两个以上只读实例，可设置targetServerType=any，用于读操作。假设只读实例IP分别为10.1.1.2、10.1.1.3。

```
jdbc:postgresql://10.1.1.2:5432,10.1.1.3:5432/${database}?  
targetServerType=any&loadBalanceHosts=true
```

6 PoWA 插件使用最佳实践

6.1 插件介绍

PoWA是一套开源的用于对RDS for PostgreSQL数据库进行性能监控的系统。它由PoWA-archivist、PoWA-collector、PoWA-web三个部件组成工作系统，并通过安装于RDS for PostgreSQL数据库中的其他插件获取性能数据。重要组成如下所示：

- PoWA-archivist：作为PostgreSQL的一个插件，用于收集其他插件获取到的性能数据。
- PoWA-collector：是通过部署于远程服务器上的专用于存储库收集目标PostgreSQL数据库性能指标的守护进程。
- PoWA-web：通过Web图形用户界面展示PoWA-collector收集到的性能指标。
- 其他插件：性能指标数据的实际来源，安装于目标PostgreSQL数据库实例上。
- PoWA：整个系统的总称。

安全风险提醒

PoWA在进行部署配置中，会存在以下几个安全风险点：

- （远程模式时）在powa-repository中配置采集性能指标实例信息时，需要输入目标实例的IP、root用户账号、连接密码，并且可以通过powa_servers表查询到相关信息，其中连接密码以明文形式呈现，存在安全风险。
- 在PoWA-collector配置文件中，powa-repository的连接信息中无连接密码配置，表示powa-repository对于PoWA-collector的连接配置项必须为trust，存在安全风险。
- 在PoWA-web配置文件中，可选配置username、password对应powa-repository（远程模式）或者数据库实例（本地模式）的root用户及连接密码，且以明文形式存储，存在安全风险。

在使用PoWA插件时，请悉知以上安全风险。可以参考[PoWA官方手册](#)进行安全加固。

其他扩展插件

除了pg_stat_statements、btree_gist、powa为必须的插件，PoWA还支持以下几个插件作为性能指标采集的扩展：

- pg_qualstats
- pg_stat_kcache
- pg_wait_sampling
- pg_track_settings
- hypopg

每个插件都可以扩展不同的对应的性能指标。当前RDS for PostgreSQL支持的插件，请参见[支持的插件列表](#)。

6.2 支持的性能指标

6.2.1 数据库级性能指标

General Overview

图 6-1 General Overview

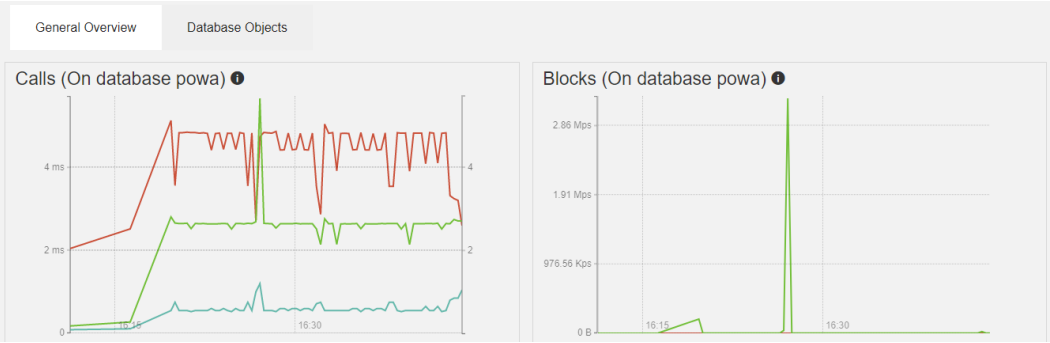


表 6-1 Calls 字段解释

字段	说明
Queries per sec	每秒执行查询的次数。
Runtime per sec	每秒内执行查询的总耗时。
Avg runtime	查询的平均耗时。

表 6-2 Blocks 字段解释

字段	说明
Total shared buffers hit	命中共享缓冲区的数据量。
Total shared buffers miss	未命中共享缓冲区的数据量。

Database Objects

图 6-2 Database Objects

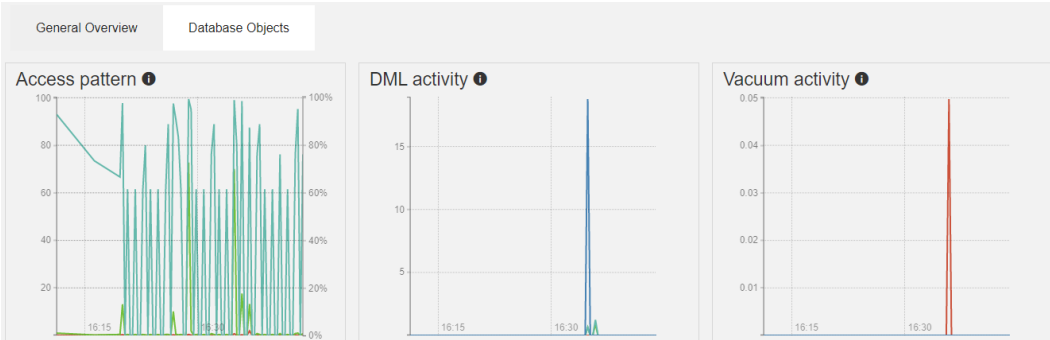


表 6-3 Access pattern 字段解释

字段	说明
Index scans ratio	索引扫描/序列扫描的比率。
Index scans	每秒索引扫描次数。
Sequential scans	每秒顺序扫描次数。

表 6-4 DML activity 字段解释

字段	说明
Tuples inserted	每秒插入的行数。
Tuples updated	每秒更新的行数。
Tuples HOT updated	每秒更新(HOT)的行数。
Tuples deleted	每秒删除的行数。

表 6-5 Vacuum activity 字段解释

字段	说明
# Vacuum	每秒手动清理的次数。
# Autovacuum	每秒自动清理的次数。
# Analyze	每秒手动分析的次数。
# Autoanalyze	每秒自动分析的次数。

Details for all databases

图 6-3 Details for all databases

Details for all queries											
Q Export CSV											
Query	Execution		I/O Time			Blocks			Temp blocks		
	#	Time	Avg time	Read	Write	Read	Hit	Dirtied	Written	Read	Written
COPY (SELECT 1, * FROM public.powa_user_functions_src(0)) TO stdout	75	3 s 783 ms	50 ms 442 μs	0	0	0 B	768.00 K	0 B	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.powa_all_relations_src(0)) TO stdout	75	82 ms 323 μs	1 ms 98 μs	0	0	0 B	720.00 K	0 B	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.powa_statements_src(0)) TO stdout	75	66 ms 99 μs	881 μs	0	0	0 B	5.38 M	0 B	0 B	0 B	0 B
select control_extension(\$1, \$2)	1	52 ms 437 μs	52 ms 437 μs	0	0	0 B	44.88 M	568.00 K	96.00 K	0 B	0 B
COPY (SELECT 1, * FROM public.powa_stat_bgwriter_src(0)) TO stdout	75	11 ms 886 μs	158 μs	0	0	0 B	688.00 K	0 B	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.powa_databases_src(0)) TO stdout	75	11 ms 186 μs	149 μs	0	0	0 B	1.01 M	0 B	0 B	0 B	0 B
/* sql from das */select r.* from (SELECT n.nspname AS schema_name, c...	1	5 ms 355 μs	5 ms 355 μs	0	0	0 B	12.34 M	72.00 K	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.pg_track_settings_settings_src(0)) TO st...	2	2 ms 367 μs	1 ms 184 μs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
/* sql from das */select count(\$1) as table_count from information_sch...	1	916 μs	916 μs	0	0	0 B	7.29 M	8.00 K	0 B	0 B	0 B
SELECT pg_catalog.set_config(name, \$1, \$2) FROM pg_catalog.pg_settings...	1	783 μs	783 μs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
SELECT setting FROM pg_settings WHERE name = \$1 --WHERE name = 'server...	1	696 μs	696 μs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
SAVEPOINT src	384	322 μs	1 μs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
SELECT \$1	35	148 μs	4 μs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
/* sql from das */SELECT \$3 FROM pg_class c LEFT JOIN pg_namespace n O...	1	148 μs	148 μs	0	0	0 B	136.00 K	0 B	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.pg_track_settings_rds_src(0)) TO stdout	2	144 μs	72 μs	0	0	0 B	0 B	0 B	0 B	0 B	0 B

表 6-6 Details for all databases 字段解释

字段	注释
Query	执行的SQL。
(Execution) #	执行该SQL次数。
(Execution) Time	执行该SQL总时间。
(Execution) Avg time	执行该SQL平均时间。
(I/O Time) Read	读I/O等待时间。
(I/O Time) Write	写I/O等待时间。
(Blocks) Read	磁盘读页面数。
(Blocks) Hit	共享缓冲区命中页面数。
(Blocks) Dirtied	脏页面数。
(Blocks) Written	磁盘写页面数。
(Temp blocks) Read	磁盘读临时页面数。
(Temp blocks) Write	磁盘写临时页面数。

6.2.2 实例级性能指标

General Overview

图 6-4 General Overview 性能指标

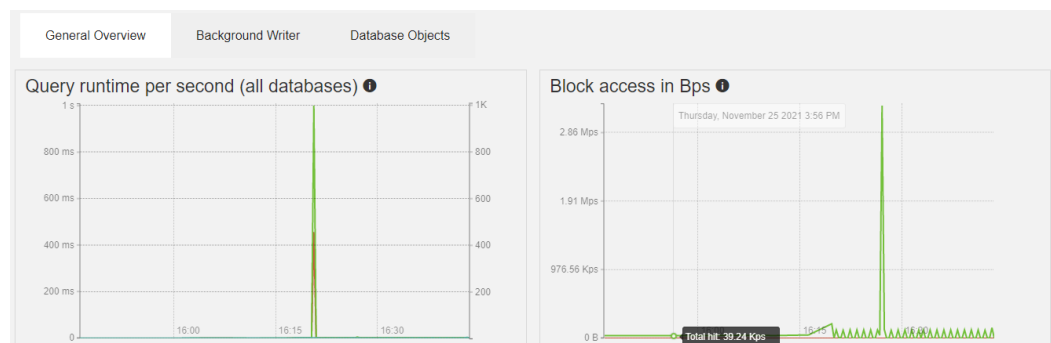


表 6-7 Query runtime per second (all databases) 字段解释

字段	说明
Queries per sec	每秒执行查询的次数。
Runtime per sec	每秒执行的查询的总持续时间。
Avg runtime	平均查询时长。

表 6-8 Block access in Bps 字段解释

字段	说明
Total hit	在共享缓冲区中找到的数据量。
Total read	在操作系统缓存中找到或从磁盘读取的数据量。

Background Writer

图 6-5 Background Writer 性能指标

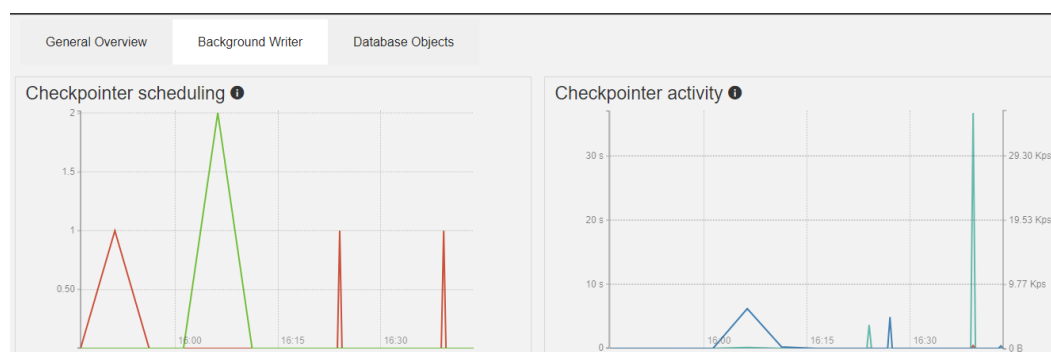


表 6-9 Checkpointer scheduling 字段解释

字段	说明
of requested checkpoints	已执行的请求检查点数。
of scheduled checkpoints	已执行的预定检查点数。

表 6-10 Checkpointer activity 字段解释

字段	说明
Buffers alloc	分配的缓冲区数。
Sync time	文件同步到磁盘的检查点处理部分所花费的总时间（单位：毫秒）。
Write time	在将文件写入磁盘的检查点处理部分中花费的总时间（单位：毫秒）。

表 6-11 Background writer 字段解释

字段	说明
Maxwritten clean	后台编写器因写入过多缓冲区而停止清理扫描的次数。
Buffers clean	后台写入器写入的缓冲区数。

表 6-12 Backends 字段解释

字段	说明
Buffers backend fsync	后端必须执行自己的 fsync 调用的次数。
Buffers backend	后端直接写入的缓冲区数。

Database Objects

图 6-6 Database Objects 性能指标

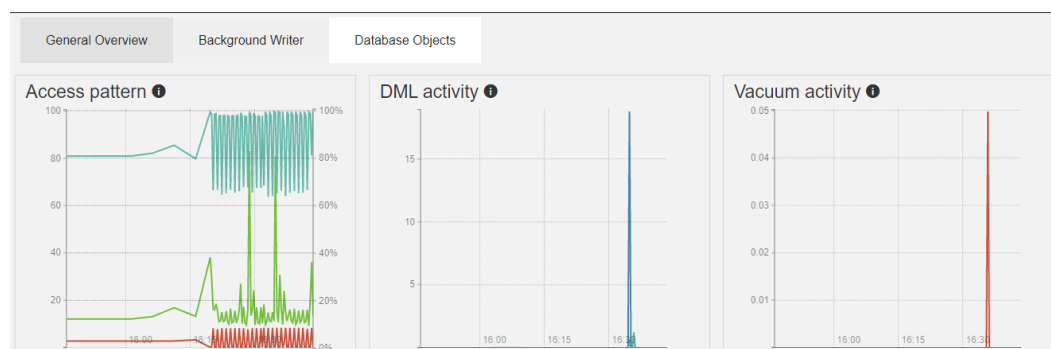


表 6-13 Access pattern 字段解释

字段	说明
Index scans ratio	索引扫描/序列扫描的比率。
Index scans	每秒索引扫描次数。
Sequential scans	每秒顺序扫描次数。

表 6-14 DML activity 字段解释

字段	说明
Tuples inserted	每秒插入的行数。
Tuples updated	每秒更新的行数。
Tuples HOT updated	每秒更新（HOT）。
Tuples deleted	每秒删除的行数。

表 6-15 Vacuum activity 字段解释

字段	说明
# Vacuum	每秒手动清理的次数。
# Autovacuum	每秒自动清理的次数。
# Analyze	每秒手动分析的次数。
# Autoanalyze	每秒自动分析的次数。

Details for all databases

图 6-7 Details for all databases 性能指标

Details for all databases

🔍

Export CSV

Database	#Calls	Runtime	Avg runtime	Blocks read	Blocks hit	Blocks dirtied	Blocks written	Temp Blocks written	I/O time
postgres	4,340	817 ms 136 μs	190 μs	0 B	133.86 M	56.00 K	0 B	0 B	0
powa	983	4 s 128 ms	4 ms 200 μs	8.00 K	75.25 M	664.00 K	96.00 K	0 B	20 μs
test	238	20 s 18 ms	84 ms 110 μs	8.00 K	864.00 K	0 B	0 B	0 B	10 μs

< (1) >

表 6-16

字段	注释
Database	数据库名称。

字段	注释
#Calls	执行SQL总数。
Runtime	执行SQL总耗时。
Avg runtime	执行SQL平均耗时。
Blocks read	磁盘读取的页面数。
Blocks hit	共享缓冲区命中的页面数。
Blocks dirtied	脏页数。
Blocks written	磁盘写页面数。
Temp Blocks written	磁盘写临时页面数。
I/O time	I/O等待时间。

6.3 部署 PoWA

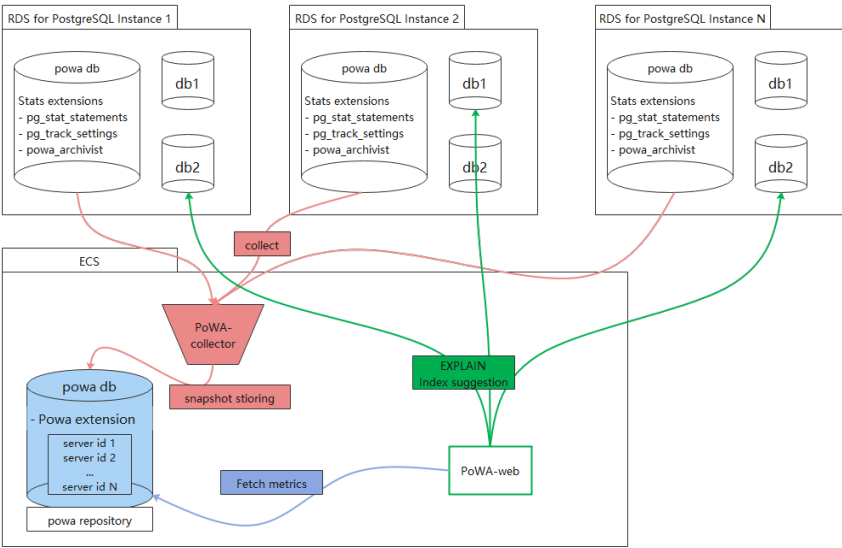
6.3.1 云上 PostgreSQL 实例部署 PoWA

华为云上进行远程模式部署PoWA，需要一台ECS，并在ECS上安装PoWA-archivist、PoWA-collector、PoWA-web。本章节主要介绍PoWA-archivist、PoWA-collector、PoWA-web的安装过程。

架构图

远程模式部署PoWA的架构如下图所示：

图 6-8 远程模式架构图



准备工作

- 已创建RDS for PostgreSQL 12.6实例。
- 已创建ECS并绑定弹性公网IP，本次演示所创建的ECS系统镜像为：CentOS 8.2 64bit。

Python3 的安装部署

安装PoWA-collecotor、PoWA-web依赖Python3环境，且使用pip3安装可以减少很多依赖环境安装的工作量。当前ECS已默认安装了 Python 3.6.8版本，由于版本偏低，安装最新版本的PoWA失败，建议安装最新的Python版本，详情请参见[安装Python 3.9.9](#)。

安装 PoWA-archivist

1. 通过wget命令获取[PoWA-archivist源码](#):

```
wget https://github.com/powa-team/powa-archivist/archive/refs/tags/REL_4_1_2.tar.gz
```
2. 将下载好的REL_4_1_2.tar.gz进行解压。
3. 进入解压后的目录，执行命令完成安装。

```
make && make install
```

安装 PoWA-collector、PoWA-web

1. 切换到RDS for PostgreSQL数据库用户下，本次演示使用的是postgres。

```
su - postgres
```
2. psycopg2 是 PoWA-collector、powa-web安装必不可少的依赖环境。

```
pip install psycopg2  
pip install powa-collector  
pip install powa-web
```

安装完成后，查看路径树如下所示，表示PoWA-collector、PoWA-web均已安装完成。

```
/home/postgres/.local/bin  
├── powa-collector.py  
├── powa-web  
└── __pycache__
```

创建 powa 插件

步骤1 使用root用户登录RDS for PostgreSQL实例的powa数据库。（若不存在，请先自行创建powa数据库）

步骤2 在powa数据库中创建powa插件。

```
select control_extension('create', 'pg_stat_statements');  
select control_extension('create', 'btree_gist');  
select control_extension('create', 'powa');
```

----结束

常见问题

Q：在执行**pip install psycopg2**时可能会遇到报错python setup.py build_ext --pg-config /path/to/pg_config build。

A：配置RDS for PostgreSQL的bin、lib路径到环境变量中，重新执行 pip install psycopg2 即可完成安装。

安装 Python 3.9.9

1. 环境准备。

请按照以下顺序执行，否则安装Python3.9.9可能会有部分失败（SSL组件依赖失败），导致后续无法安装PoWA-collector、PoWA-web。

```
yum install readline* -y
yum install zlib* -y
yum install gcc-c++ -y
yum install sqlite* -y
yum install openssl-* -y
yum install libffi* -y
```

2. 安装python 3.9.9。

a. 使用root用户执行下列命令。

```
mkdir env
cd env
wget https://www.python.org/ftp/python/3.9.9/Python-3.9.9.tgz
tar -xzf Python-3.9.9.tgz
cd Python-3.9.9
./configure --prefix=/usr/local/python3.9.9
make && make install
```

b. 创建软链接。

```
ln -s /usr/local/python3.9.9/bin/python3.9 /usr/bin/python
ln -s /usr/local/python3.9.9/bin/pip3.9 /usr/bin/pip
```

3. 验证安装是否成功。

a. 验证安装，重点验证SSL功能。

```
[root@ecs-ad4d Python-3.9.9]# python
Python 3.9.9 (main, Nov 25 2021, 12:36:32)
[GCC 8.4.1 20200928 (Red Hat 8.4.1-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
import ssl
import urllib.request
context = ssl._create_unverified_context()
urllib.request.urlopen('https://www.example.com/',context=context).read()
```

b. 如果有返回，说明安装成功。执行以下命令退出。

```
quit()
```

6.3.2 自建 PostgreSQL 实例部署 PoWA

本章节主要介绍ECS自建PostgreSQL数据库部署PoWA的过程。

准备工作

已有自建PostgreSQL实例：

- 版本：PostgreSQL 12.6
- 管理员账号：postgres
- PostgreSQL专用存储数据库：powa-repository
- data路径：/home/postgres/data

部署 PoWA

步骤1 修改配置文件 /home/postgres/data/postgresql.conf，添加 pg_stat_statements 至参数 shared_preload_libraries，如下图所示：

```
shared_preload_libraries = 'pg_stat_statements' # (change requires restart)
```

步骤2 执行命令重启数据库。

```
pg_ctl restart -D /home/postgres/data/
```

步骤3 使用postgres登录数据库，并创建database: powa，以及安装相关插件。**须知**

创建的database必须命名为powa，否则PoWA运行过程中会报错，部分功能失效。

```
[postgres@ecs-ad4d ~]$ psql -U postgres -d postgres
psql (12.6)
Type "help" for help.
postgres=# create database powa;
CREATE DATABASE
postgres=# \c powa
You are now connected to database "powa" as user "postgres".
powa=# create extension pg_stat_statements ;
CREATE EXTENSION
powa=# create extension btree_gist ;
CREATE EXTENSION
powa=# create extension powa;
CREATE EXTENSION
```

步骤4 配置需要采集性能指标的实例信息。

1. 执行SQL，添加目标实例信息。

```
powa=# select powa_register_server(
    hostname => '192.168.0.1',
    alias => 'myInstance',
    port => 5432,
    username => 'user1',
    password => '*****',
    frequency => 300);
 powa_register_server
-----
 t
(1 row)
```

2. 通过查看“powa_servers”表来获取当前采集指标的实例信息。

```
powa=# select * from powa_servers;
 id | hostname | alias | port | username | password | dbname | frequency | powa_coalesce | retention |
allow_ui_connection | version
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
 0 |          | <local> | 0 |          |          |          | -1 | 100 | 00:00:00 | t |          |
 1 | 192.168.0.1 | myInstance | 5432 | user1 | ***** | powa | 300 | 100 | 1 day |
 t
(2 rows)
```

须知

在录入目标实例信息、查询目标实例都会涉及到目标实例的IP、root用户账户、明文密码等重要隐私信息，可能会面临信息安全风险。

请谨慎评估该插件带来的安全风险后决定是否使用该插件。

----结束

PoWA-collector 配置

启动PoWA-collector:

```
cd /home/postgres/.local/bin
./powa-collector.py &
```

PoWA-collector启动时，将按以下顺序搜索配置文件作为其配置：

1. /etc/powa-collector.conf
2. ~/.config/powa-collector.conf
3. ~/.powa-collector.conf
4. ./powa-collector.conf

配置文件中需要包含以下选项：

- repository.dsn : URL，用于通知 powa-collector 如何连接专用存储数据库（powa-repository）。
- debug : Boolean类型，用于指定是否在调试模式下启动 powa-collector。

本次演示中将配置写入文件./powa-collector.conf

```
{
  "repository": {
    "dsn": "postgresql://postgres@localhost:5432/powa"
  },
  "debug": true
}
```

PoWA-collector 的配置中并没有密码的配置，所以powa-repository数据库的pg_hba.conf中需要配置对应的连接策略为trust免密连接。

PoWA-web 配置

启动PoWA-web：

```
cd /home/postgres/.local/bin
./powa-web &
```

PoWA-web启动时，将按以下顺序搜索配置文件作为其配置：

1. /etc/powa-web.conf
2. ~/.config/powa-web.conf
3. ~/.powa-web.conf
4. ./powa-web.conf

本次实例中需要将配置内容写入文件./powa-web.conf中。

```
# cd /home/postgres/.local/bin
# vim ./powa-web.conf
# 写入配置内容，并保存
servers={
  'main': {
    'host': 'localhost',
    'port': '5432',
    'database': 'powa',
    'username': 'postgres',
    'query': {'client_encoding': 'utf8'}
  }
}
cookie_secret="SECRET_STRING"
```

本章节中powa-repository数据库pg_hab.conf中配置为trust，免密连接，因此未配置password。

6.4 在 PoWA 上查看指标详情

PoWA部署完成并成功启动PoWA-collector、PoWA-web后，可以通过浏览器登录PoWA，查看监控实例的指标详情。

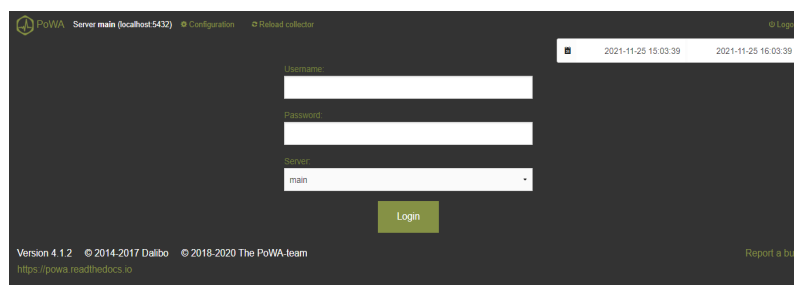
访问 PoWA

步骤1 使用浏览器访问PoWA。

说明

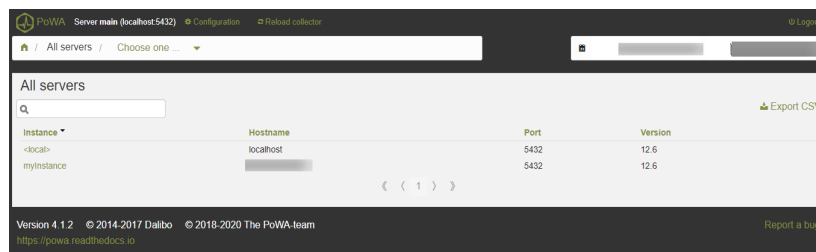
- powa-web.conf中未配置port选项，使用默认值8888
- 浏览器访问链接：<http://演示使用的ECS IP地址:8888/>

图 6-9 访问 PoWA



步骤2 输入用户名和密码，单击“Login”。

图 6-10 PoWA 首页



该PoWA中采集了两个PostgreSQL实例的信息：

- <local>: ECS自建PostgreSQL，作为powa-repository角色。
- myinstance: 云数据库 RDS for PostgreSQL实例，作为性能数据采集目标。（myinstance为在powa-repository中注册实例别名(alias字段)）。

步骤3 单击对应的实例，即可查看实例具体的性能指标。

----结束

查看指标详情

PoWA可以采集、显示的性能指标非常丰富，下面的步骤以查看慢SQL指标为例。

步骤1 [登录管理控制台](#)。

步骤2 单击管理控制台左上角的 ，选择区域。

步骤3 单击页面左上角的，选择“数据库>云数据库 RDS”，进入RDS信息页面。

步骤4 在“实例管理”页面，选择目标实例，单击操作列的“登录”，进入数据管理服务实例登录界面。

您也可以在“实例管理”页面，单击目标实例名称，在页面右上角，单击“登录”，进入数据管理服务实例登录界面。

步骤5 正确输入数据库用户名和密码，单击“登录”，即可进入您的数据库并进行管理。

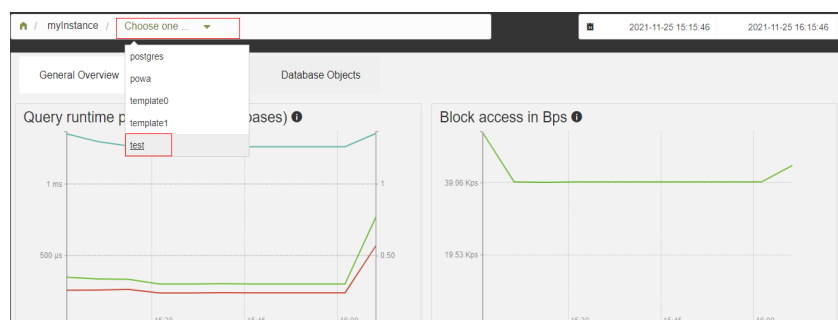
步骤6 在首页数据库列表栏单击“新建数据库”。

步骤7 在弹窗中填写数据库名称“test”、字符集等信息。

步骤8 单击“SQL查询”，在test数据库上执行慢SQL。

步骤9 等待大概5分钟后，在PoWA首页选中目标实例，选择test数据库，如下图所示：

图 6-11 PoWA 首页



在 Details for all queries中可以看到SELECT pg_sleep(\$1)语句，对应执行时间20s。

Details for all queries

Export CSV

SELECT pg_sleep(5);

SELECT pg_sleep(11);
set time zone 'PAC';
SELECT t.oid, typarray FROM pg_type t JOIN pg_namespace ns ON typnames=...
SELECT 1;

#	Execution			I/O Time			Blocks			Temp blocks		
	Time	Avg time		Read	Write	Read	Hit	Dirtyed	Written	Read	Written	
1	20 s 17 ms	20 s 17 ms		0	0	0 B	0 B	0 B	0 B	0 B	0 B	0 B
1	98 μs	98 μs		0	0	0 B	0 B	0 B	0 B	0 B	0 B	0 B
6	77 μs	13 μs		0	0	0 B	96.00 K	0 B	0 B	0 B	0 B	0 B
14	65 μs	5 μs		0	0	0 B	0 B	0 B	0 B	0 B	0 B	0 B

----结束

安装扩展插件并采集性能指标

下面步骤以pg_track_settings插件为例。

步骤1 登录管理控制台。

步骤2 单击管理控制台左上角的，选择区域。

步骤3 单击页面左上角的，选择“数据库>云数据库 RDS”，进入RDS信息页面。

步骤4 在“实例管理”页面，选择目标实例，单击操作列的“登录”，进入数据管理服务实例登录界面。

您也可以在“实例管理”页面，单击目标实例名称，在页面右上角，单击“登录”，进入数据管理服务实例登录界面。

步骤5 正确输入数据库用户名和密码，单击“登录”，即可进入您的数据库并进行管理。

步骤6 选择powa数据库，执行SQL命令创建pg_track_settings插件。

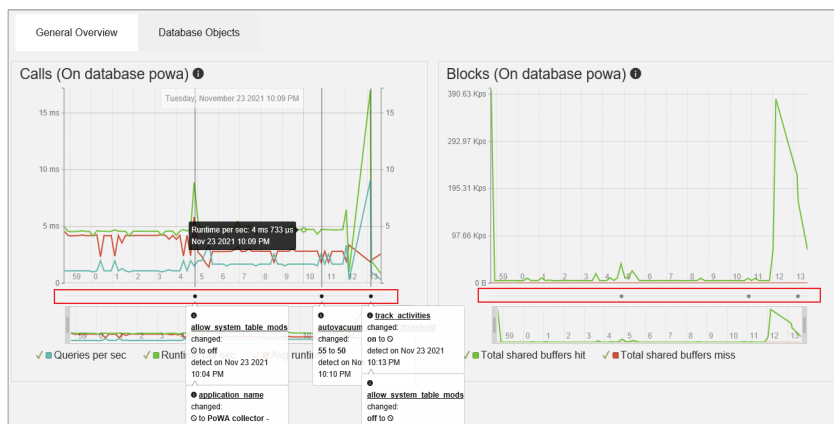
```
select control_extension('create', 'pg_track_settings');
```

步骤7 ECS上自建PostgreSQL(powa-repository)，安装pg_track_settings插件，并激活pg_track_settings插件采集性能指标。

```
# pg_track_settings
cd /home/postgres/env
wget https://github.com/rjuju/pg_track_settings/archive/refs/tags/2.0.1.tar.gz
mv 2.0.1.tar.gz pg_track_settings.2.0.1.tar.gz
tar -xzf pg_track_settings.2.0.1.tar.gz
cd pg_track_settings-2.0.1
make && make install
# powa-repository
psql -d powa
powa=# create extension pg_track_settings ;
CREATE EXTENSION
# 激活目标实例 pg_track_settings 采集功能
dbpowa=# select powa_activate_extension(1, 'pg_track_settings');
powa_activate_extension
-----
t
(1 row)
```

步骤8 pg_track_settings插件扩展完成，进行验证。

在目标实例上修改参数“autovacuum_analyze_threshold”，原始默认值为50，修改后为55，等待大概5分钟，在PoWA页面上就可以看到对应参数修改的记录了，如下图所示：



上图中3个说明框中记录内容如下：

- 记录了pg_track_settings插件激活的时间点及当时数据库参数值。
- 记录了“autovacuum_analyze_threshold”参数被修改的时间以及原始值和修改后的值。
- 记录了pg_track_settings插件被取消的时间点及当时数据库参数值。

----结束

7 pg_dump 使用最佳实践

简介

pg_dump是PostgreSQL原生的备份工具。pg_dump生成的备份文件可以是一个SQL脚本文件，也可以是一个归档文件。详细信息请查看[pg_dump官方说明](#)。

- SQL脚本文件：纯文本格式的文件，其中包含将数据库重建到备份时状态的SQL命令。
- 归档格式的备份文件：必须与pg_restore一起来重建数据库，这种格式允许pg_restore选择恢复哪些数据。

注意事项

pg_dump适合单个库、schema级、表级导出，只会导出表及数据、函数等，数据库和用户需要提前在要恢复的库创建。

- --format=custom：备份为二进制格式，二进制格式的备份只能使用pg_restore来还原，并且可以指定还原的表。
- --format=plain：备份为文本，文本格式的备份还原，直接使用用户连接到对应的数据库执行备份文本即可。

限制条件

在使用pg_dump和pg_restore进行备份和恢复时，确保源数据库和目标库的版本一致，以避免出现兼容性问题，如果版本不一致可能会导致数据丢失或无法正确还原数据。

准备测试数据

```
# 创建数据库
create database dump_database;

# 登录dump_database数据库
\c dump_database

# 创建表1并插入数据
create table dump_table(id int primary key, content char(50));
insert into dump_table values(1,'aa');
insert into dump_table values(2,'bb');
```

```
# 创建表2并插入数据
create table dump_table2(id int primary key, content char(50));
insert into dump_table2 values(1,'aaaa');
insert into dump_table2 values(2,'bbbb');
```

使用 pg_dump 将数据库导出至 SQL 文件

语法

```
pg_dump --username=<DB_USER> --host=<DB_IPADDRESS> --port=<DB_PORT> --format=plain --
file=<BACKUP_FILE><DB_NAME>
```

- DB_USER为数据库用户。
- DB_IPADDRESS为数据库地址。
- DB_PORT为数据库端口。
- BACKUP_FILE为要导出的文件名称。
- DB_NAME为要导出的数据库名称。
- --format为导出的文件格式，plain为输出纯文本SQL脚本文件（默认）。其他选项详见[pg_dump官方说明](#)。

示例

- 导出数据库至SQL文件（INSERT语句）。

```
$ pg_dump --username=root --host=192.168.61.143 --port=5432 --format=plain --inserts --
file=backup.sql dump_database
Password for user root:
```
- 导出数据库中所有表结构至SQL文件。

```
$ pg_dump --username=root --host=192.168.61.143 --port=5432 --format=plain --schema-only --
file=backup.sql dump_database
Password for user root:
```
- 导出数据库中所有表数据至SQL文件。

```
$ pg_dump --username=root --host=192.168.61.143 --port=5432 --format=plain --data-only --
file=backup.sql dump_database
Password for user root:
```

命令执行完会生成“backup.sql”文件，如下：

```
[rds@localhost ~]$ ll backup.sql
-rw-r----- 1 rds rds 5657 May 24 09:21 backup.sql
```

使用 pg_dump 将数据库中的表导出至 SQL 文件

语法

```
pg_dump --username=<DB_USER> --host=<DB_IPADDRESS> --port=<DB_PORT> --format=plain --
file=<BACKUP_FILE> <DB_NAME> --table=<TABLE_NAME>
```

- DB_USER为数据库用户。
- DB_IPADDRESS为数据库地址。
- DB_PORT为数据库端口。
- BACKUP_FILE为要导出的文件名称。
- DB_NAME为要迁移的数据库名称。
- TABLE_NAME为要迁移的数据库中指定表名称。
- --format为导出的文件格式，plain为输出纯文本SQL脚本文件（默认）。其他选项详见[pg_dump官方说明](#)。

示例

- 导出数据库中指定的单表至SQL文件。

```
$ pg_dump --username=root --host=192.168.61.143 --port=5432 --format=plain --file=backup.sql  
dump_database --table=dump_table  
Password for user root:
```
- 导出数据库中指定的多表至SQL文件。

```
$ pg_dump --username=root --host=192.168.61.143 --port=5432 --format=plain --file=backup.sql  
dump_database --table=dump_table --table=dump_table2  
Password for user root:
```
- 导出数据库中以ts_开头的表至SQL文件。

```
$ pg_dump --username=root --host=192.168.61.143 --port=5432 --format=plain --file=backup.sql  
dump_database --table=ts_*
```
- 导出数据库中除ts_开头之外的所有表至SQL文件。

```
$ pg_dump --username=root --host=192.168.61.143 --port=5432 --format=plain --file=backup.sql  
dump_database -T=ts_*
```

命令执行完会生成“backup.sql”文件，如下：

```
[rds@localhost ~]$ ll backup.sql  
-rw-r----- 1 rds rds 5657 May 24 09:21 backup.sql
```

使用 pg_dump 导出特定 schema 下的数据

语法

```
pg_dump --username=<DB_USER> --host=<DB_IPADDRESS> --port=<DB_PORT> --format=plain --  
schema=<SCHEMA> <DB_NAME> --table=<BACKUP_FILE>
```

- DB_USER为数据库用户。
- DB_IPADDRESS为数据库地址。
- DB_PORT为数据库端口。
- BACKUP_FILE为要导出的文件名称。
- DB_NAME为要导出的数据库名称。
- SCHEMA为要导出的schema名称。
- --format为导出的文件格式，plain为输出纯文本SQL脚本文件（默认）。其他选项详见[pg_dump官方说明](#)。

示例

- 导出指定库中public schema的所有数据。

```
pg_dump --username=root --host=192.168.61.143 --port=5432 --format=plain --schema=public  
dump_database > backup.sql
```
- 导出指定库中除public schema以外的所有数据，结果以自定义压缩格式导出。

```
pg_dump --username=root --host=192.168.61.143 --port=5432 --format=custom -b -v -N public  
dump_database > all_sch_except_pub.backup
```

还原数据

使用纯文本SQL脚本文件导出的数据，还原时直接使用psql命令即可，比如：

```
# 还原特定数据库  
psql --username=root --host=192.168.61.143 --port=5432 backup_database < backup.sql  
  
# 还原特定表  
psql --username=root --host=192.168.61.143 --port=5432 backup_database --  
table=dump_table < backup.sql
```

```
# 还原特定schema
psql --username=root --host=192.168.61.143 --port=5432 backup_database --schema=public <
backup.sql
```

说明

在恢复之前在目标库中创建数据库backup_database。

使用其他格式导出的数据进行还原时，需要使用pg_restore。[pg_restore](#)用于恢复由pg_dump转储的任何非纯文本格式中的PostgreSQL数据库。

```
pg_restore --username=root --host=192.168.61.143 --port=5432 --dbname=backup_database --
format=custom all_sch_except_pub.backup --verbose
```

常见问题

1. 如果在使用pg_dump导出时，报错用户权限不足。

解决方法：

检查是否使用的root用户导出，如果不是root用户则会报权限不足；如果使用root用户导出，仍然提示没有权限，请检查数据库版本，root用户执行pg_dump命令需要内核版本为支持root提权的版本，支持root提权版本情况见[root用户权限说明](#)。

2. 将备份的文件导入到RDS for PostgreSQL目标数据库时发现control_extension等几个函数报错。

解决方法：

由于目标库中自带这些函数，因此该报错可以忽略。

8 PgBouncer 使用最佳实践

PgBouncer 介绍

PgBouncer是为PostgreSQL提供的轻量级连接池工具，作用如下：

- 能够缓存和PostgreSQL的连接，当有连接请求进来的时候，直接分配空闲进程，而不需要PostgreSQL fork出新进程来建立连接，以节省创建新进程，创建连接的资源消耗。
- 能够有效提高连接的利用率，避免过多的无效连接，导致数据库消耗资源过大，CPU占用过高。
- 对客户端连接进行限制，预防过多或恶意的连接请求。

轻量级体现在：

- 使用libevent进行socket通信，通信效率高。
- C语言编写，效率高，每个连接仅消耗2KB内存。

PgBouncer目前支持三种连接池模型：

- session会话级连接。只有与当客户端的会话结束时，PgBouncer才会收回已分配的连接。
- transaction事务级连接。当事务完成后，PgBouncer会回收已分配的连接。也就是说客户端只是在事务中才能独占此连接，非事务请求没有独享的连接。
- statement语句级连接。任何数据库请求完成后，PgBouncer都会回收连接。此种模式下，客户端不能使用事务，否则会造成数据的不一致。

PgBouncer默认选项是session，建议修改为transaction。

安装配置

在云上部署PgBouncer连接池，需要先[购买弹性云服务器](#)。建议选择与后端RDS实例相同VPC、相同子网创建，降低网络通信时延。购买成功后登录ECS进行环境搭建。

1. 由于PgBouncer是基于libevent开发，需要安装libevent-devel openssl-devel两个依赖包。命令如下：

```
yum install -y libevent-devel
yum install -y openssl-devel
```
2. 依赖包安装后，在PgBouncer官网下载源码，以普通用户身份进行编译、安装。

```
su - pgbouncer
tar -zxvf pgbouncer-1.19.0.tar.gz
```

```
cd pgbouncer-1.19.0
./configure --prefix=/usr/local
make
make install
```

3. 创建如下目录，用于保存PgBouncer的生成文件（日志、进程标识等）。

```
mkdir -p /etc/pgbouncer/
mkdir -p /var/log/pgbouncer/
mkdir -p /var/run/pgbouncer/
```

4. 启动PgBouncer前，需要构建配置文件“pgbouncer.ini”。

```
[databases]
* = host=127.0.0.1 port=5432
[pgbouncer]
logfile = /var/log/pgbouncer/pgbouncer.log
pidfile = /var/run/pgbouncer/pgbouncer.pid
listen_addr = *
listen_port = 6432
auth_type = md5
auth_file = /etc/pgbouncer/userlist.txt
admin_users = postgres
stats_users = stats, postgres
pool_mode = transaction
server_reset_query = DISCARD ALL
max_client_conn = 100
default_pool_size = 20
;; resolve: unsupported startup parameter: extra_float_digits
;; ignore_startup_parameters = extra_float_digits
```

配置文件中各参数含义可参考[PgBouncer官方文档](#)。

启动 pgbouncer

PgBouncer不能以root身份启动，需要基于普通用户身份进行启动。

```
pgbouncer -d /etc/pgbouncer/pgbouncer.ini
```

启动后，可通过**netstat -tunlp | grep pgbouncer**查看连接池的监听端口，而后进行连接：

```
psql -U root -d postgres -h 127.0.0.1 -p 6432
Password for user root:
psql (12.13)
Type "help" for help.
postgres=> \l

               List of databases
  Name  | Owner  | Encoding | Collate  | Ctype   | Access privileges
-----+-----+-----+-----+-----+-----
postgres | pgbouncer | UTF8    | en_US.UTF-8 | en_US.UTF-8 | 
template0 | pgbouncer | UTF8    | en_US.UTF-8 | en_US.UTF-8 | =c/pgbouncer
          |          |          |          |          | pgbouncer=CTc/pgbouncer
template1 | pgbouncer | UTF8    | en_US.UTF-8 | en_US.UTF-8 | =c/pgbouncer
```

停止 PgBouncer

可以直接通过Kill命令停止。

```
kill `cat /var/run/pgbouncer/pgbouncer.pid`
cat /var/run/pgbouncer/pgbouncer.pid | xargs kill -9
```

PgBouncer 管理

PgBouncer对外提供了一个虚拟数据库pgbouncer，之所以称为虚拟数据库，是因为它可以提供像PostgreSQL那样的数据库操作界面，但是这个数据库却并不是真实存在的，而是PgBouncer虚拟出来的一个命令行界面。登录虚拟数据库：

```
psql -p 6432 -d pgbouncer
```

如果修改了一些配置参数，可以不用重启PgBouncer而是**reload**使其生效。

```
pgbouncer=# reload;  
RELOAD
```

登录后可以通过**show help**查看命令帮助，通过**show clients**查看客户端连接信息，通过**show pools**查看连接池信息。

实现读写分离示例

PgBouncer并不支持自动解析读写请求，并进行读写分离路由，需要业务侧对读写操作进行区分。

1. 首先，修改配置文件“pgbouncer.ini”配置的数据库信息，添加主库、只读库连接配置。本例中参数设置如下：

```
[databases]  
;; * = host=127.0.0.1 port=5432  
#配置只读库的连接信息  
mydb_read: host=10.7.131.69 port=5432 dbname=postgres user=root password=***  
# 配置主库的连接信息  
mydb_write: host=10.8.115.171 port=5432 dbname=postgres user=root password=***  
[pgbouncer]  
logfile = /var/log/pgbouncer/pgbouncer.log  
pidfile = /var/run/pgbouncer/pgbouncer.pid  
listen_addr = *  
listen_port = 6432  
auth_type = md5  
auth_file = /etc/pgbouncer/userlist.txt  
admin_users = postgres  
stats_users = stats, postgres  
pool_mode = transaction  
server_reset_query = DISCARD ALL  
max_client_conn = 100  
default_pool_size = 20  
;; resolve: unsupported startup parameter: extra_float_digits  
;; ignore_startup_parameters = extra_float_digits
```

2. 验证是否能连接主库和只读数据库。通过psql成功连接只读、主库，可支持业务读写分离。

```
psql -U root -d mydb_write -h 127.0.0.1 -p 6432  
Password for user root:  
psql (14.6)  
mydb_write=> SELECT pg_is_in_recovery();  
pg_is_in_recovery  
-----  
f  
(1 row)  
psql -U root -d mydb_read -h 127.0.0.1 -p 6432  
Password for user root:  
psql (14.6)  
mydb_read=> SELECT pg_is_in_recovery();  
pg_is_in_recovery  
-----  
t  
(1 row)
```

9 RDS for PostgreSQL 安全最佳实践

PostgreSQL数据库在可靠性、稳定性、数据一致性等获得了业内极高的声誉，已成为许多企业的首选开源关系数据库，业界简称PG。RDS for PostgreSQL是一种基于云计算平台的即开即用、稳定可靠、弹性伸缩、便捷管理的在线云数据库服务。

为加强RDS for PostgreSQL数据库安全性，本文将从以下几个维度给出建议，您可以根据业务需要在本指导的基础上进行安全配置。

- [配置数据库的最大连接数](#)
- [配置客户端认证超时时间](#)
- [配置SSL连接和加密算法](#)
- [配置密码加密功能](#)
- [配置合理的pg_hba规则](#)
- [配置服务器拒绝带反斜杠转义的引号](#)
- [定期检查并删除业务不再使用的角色](#)
- [建议回收public模式的所有权限](#)
- [设置合理的用户角色密码有效期](#)
- [配置日志级别记录发生错误的SQL语句](#)
- [确保数据库账号的最低权限](#)
- [开启备份功能](#)
- [开启数据库审计功能](#)
- [避免绑定EIP直接通过公网访问RDS for PostgreSQL](#)
- [数据库版本更新到最新版本](#)
- [配置账号认证失败延迟时间](#)

配置数据库的最大连接数

max_connections 决定了数据库的最大并发连接数。增加这个参数值可能引起RDS for PostgreSQL请求更多的System V共享内存或者信号量，会导致超出操作系统默认配置允许的值。请根据业务的复杂度，合理配置max_connections，具体可参考[实例使用规范](#)。

配置客户端认证超时时间

`authentication_timeout`控制完成客户端认证的时间上限，单位是秒。该参数可以防止客户端长时间占用连接通道，默认是60s。如果在指定的时间内没有完成认证，连接将被强制关闭。该超时时间的配置是为了增强PostgreSQL的安全性。

配置 SSL 连接和加密算法

尽可能利用SSL进行TCP/IP连接，使用SSL加密通信可确保客户端和服务端之间的所有通信都经过加密，防止数据被泄露和篡改，确保数据的完整性。在设置SSL加密时，服务端需要配置安全的TLS协议和加密算法，推荐使用TLSv1.2协议，加密算法推荐使用EECDH+ECDSA+AESGCM:EECDH+aRSA+AESGCM:EDH+aRSA+AESGCM:EDH+aDSS+AESGCM:!aNULL:!eNULL:!LOW:!3DES:!MD5:!EXP:!SRP:!RC4，具体请参考[SSL连接](#)。

可通过修改参数“`ssl_min_protocol_version`”配置TLS协议，修改参数“`ssl_ciphers`”配置加密算法。

配置密码加密功能

密码必须要加密。使用**CREATE USER**或者**ALTER ROLE**修改密码时候，默认使用加密的方式，推荐使用scram-sha-256，可通过修改参数“`password_encryption`”进行配置。

MD5选项仅供与低版本兼容场景使用，新建数据库实例默认使用scram-sha-256。

须知

参数“`password_encryption`”修改后需要重置密码后才能生效。

配置合理的 pg_hba 规则

为了确保PostgreSQL数据库的安全性，配置合理的pg_hba.conf规则至关重要。建议仅允许必要的用户和主机访问数据库，限制特定IP地址或子网的访问，定期审查和更新pg_hba.conf文件以确保规则符合当前的业务需要。具体配置方法请参考[修改pg_hba配置](#)。

配置服务器拒绝带反斜杠转义的引号

参数“`backslash_quote`”控制字符串里面引号是否可以被\''代替。推荐使用SQL标准方法，表示一个引号是写两遍 (")，如果客户端代码不能正确的转义，可能发生SQL注入攻击。建议配置参数“`backslash_quote`”值为“`safe_encoding`”，拒绝带反斜杠转义的引号的查询，可以避免SQL注入的风险。

定期检查并删除业务不再使用的角色

对于每个查询出来的角色，检查是否必须存在，任何未知的角色都需要被审视，确保每个角色都是正常使用的，否则删除这些角色。可通过如下命令进行查询：

```
SELECT rolname FROM pg_roles;
```

建议回收 public 模式的所有权限

public模式是默认的模式，所有用户都可以访问其中的对象，包括表、函数、视图等。如果public模式中的对象被授予了权限，那么所有用户都可以访问这些对象，这可能会导致安全漏洞。root用户可通过如下命令回收权限：

```
revoke all on schema public from public;
```

设置合理的用户角色密码有效期

当创建角色时候，使用**VALID UNTIL**关键字设置过多长时间后角色的密码不再有效。如果这个关键字被忽略，密码会长期有效。建议定期更改密码，例如每三个月更改一次密码。可通过如下命令进行设置：

```
CREATE ROLE name WITH PASSWORD 'password' VALID UNTIL 'timestamp';
```

检查是否设置密码有效期：

```
SELECT rolname,rolvaliduntil FROM pg\_roles WHERE rolsuper = false AND  
rolvaliduntil IS NULL;
```

配置日志级别记录发生错误的 SQL 语句

参数“log_min_error_statement”控制哪些引起错误的SQL语句记录到服务器日志中。对于大于等于当前配置等级的SQL语句消息，会记录到日志里。有效的值包括 debug5, debug4, debug3, debug2, debug1, info, notice, warning, error, log, fatal, panic。“log_min_error_statement”至少配置为“error”，具体可参考[日志配置管理](#)。

确保数据库账号的最低权限

RDS for PostgreSQL支持“基于角色”的方法授予账号对数据和命令的访问权限。建议管理员结合业务需要，遵从最低授权原则，创建合适的[数据库账号](#)，对账号进行授权。如果发现存在不符合该角色的账号权限，请结合业务需要，对账号权限进行更新或者[删除](#)。由于PostgreSQL存在一些[内置账号](#)，用于给数据库实例提供完善的后台运维管理服务，禁止用户使用和删除。

开启备份功能

创建云数据库RDS实例时，系统默认开启自动备份策略，默认自动备份保留7天，可根据业务需要调整备份保留时长。RDS for PostgreSQL实例支持[自动备份](#)和[手动备份](#)，您可以定期对数据库进行备份，当数据库故障或数据损坏时，可以[通过备份文件恢复数据库](#)，从而保证数据可靠性，详情请参见[数据备份](#)。

开启数据库审计功能

通过将PostgreSQL审计扩展（pgAudit）与RDS for PostgreSQL数据库实例一起使用，可以捕获审计员通常需要或满足法规要求的详细记录。例如，您可以设置pgAudit扩展来跟踪对特定数据库和表所做的更改、记录进行更改的用户以及许多其他详细信息。pgAudit默认不开启，根据业务需要开启插件。具体配置可参考[使用pgAudit插件](#)。

避免绑定 EIP 直接通过公网访问 RDS for PostgreSQL

避免RDS for PostgreSQL部署在公网或者DMZ里，应该将RDS for PostgreSQL部署在华为云内部网络，使用路由器或者防火墙技术把RDS for PostgreSQL保护起来，避免

直接绑定EIP方式从公网访问RDS for PostgreSQL。通过这种方式防止未授权的访问及DDos攻击等。建议解绑弹性公网IP，如果您的业务必须绑定EIP，请务必通过设置[安全组规则](#)限制访问数据库的源IP。

数据库版本更新到最新版本

对于PostgreSQL社区停止维护的版本，云上RDS for PostgreSQL服务也会相应发布[产品生命周期](#)。使用较老的版本可能存在安全风险，运行最新版本的软件可以避免受到某些攻击。如果业务需要，可通过[升级内核小版本](#)或者[使用转储与还原升级大版本](#)。

配置账号认证失败延迟时间

PostgreSQL数据库默认内置了auth_delay插件，auth_delay会使服务器在返回认证失败之前短暂停止，使得暴力破解数据库密码更难。可通过[修改RDS for PostgreSQL实例参数](#)设置auth_delay.milliseconds参数（该参数为返回认证失败之前等待的毫秒数）延迟账号登录认证失败的等待时间，缺省值是3000。