



# 数据库配置

## Enterprise applications

NetApp  
February 11, 2026

This PDF was generated from <https://docs.netapp.com/zh-cn/ontap-apps-dbs/postgres/postgres-architecture.html> on February 11, 2026. Always check [docs.netapp.com](https://docs.netapp.com) for the latest.

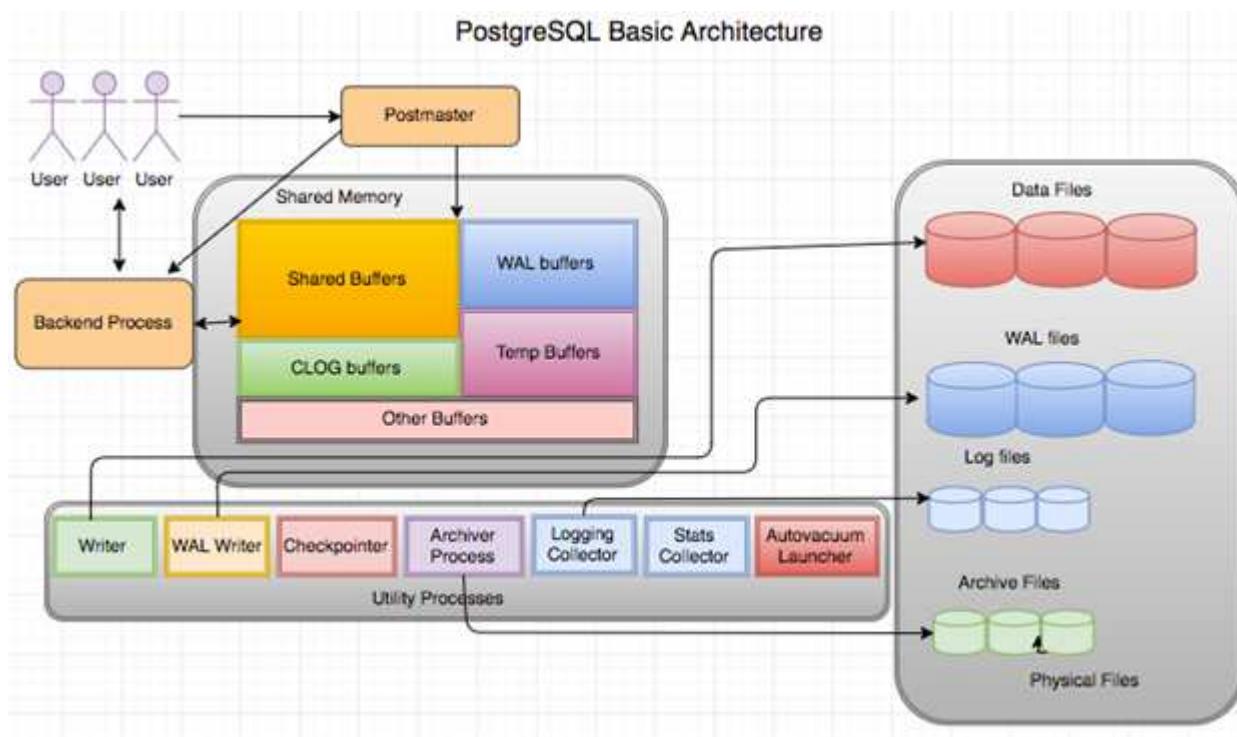
# 目录

数据库配置 . . . . .	1
架构 . . . . .	1
初始化参数 . . . . .	2
设置 . . . . .	2
吐司 . . . . .	3
真空 . . . . .	3
表空间 . . . . .	3

# 数据库配置

## 架构

PostgreSQL是基于客户端和服务器架构的RDBMS。PostgreSQL实例称为数据库集群、它是一组数据库、而不是一组服务器。



PostgreSQL数据库中有三个主要元素：PostMAIN、前端(客户端)和后端客户端向邮件服务器发送请求、并提供IP协议和要连接到的数据库等信息。邮件服务器对连接进行身份验证，并将其传递到后端进程以进行进一步的通信。后端进程执行查询并将结果直接发送到前端(客户端)。

PostgreSQL实例基于多进程模型、而不是多线程模型。它会为不同的作业生成多个进程、每个进程都有自己的功能。主要进程包括客户端进程、WAL写入程序进程、后台写入程序进程和检查指针进程：

- 当客户端(前台)进程向PostgreSQL实例发送读取或写入请求时、它不会直接向磁盘读取或写入数据。它首先将数据缓冲在共享缓冲区和预写日志记录(Write-Ahead Logging、WAL)缓冲区中。
- WAL写入程序进程操纵共享缓冲区和WAL缓冲区的内容以写入WAL日志。WAL日志通常是PostgreSQL的事务日志、并按顺序写入。因此、为了缩短数据库的响应时间、PostgreSQL首先写入事务日志并确认客户端。
- 为了使数据库处于一致状态、后台写入程序进程会定期检查共享缓冲区中是否存在脏页。然后、它会将数据转储到存储在NetApp卷或LUN上的数据文件中。
- 检查指针进程也会定期运行(比后台进程频率低)、并防止对缓冲区进行任何修改。它会向WAL写入程序进程发出信号、将检查点记录写入并刷新到NetApp磁盘上存储的WAL日志的末尾。它还会向后台写入程序进程发出信号、指示将所有脏页写入磁盘并将其刷新。

## 初始化参数

您可以使用创建新的数据库集群 initdb 计划。一个 initdb 脚本将创建用于定义集群的数据文件、系统表和模板数据库(template0和template1)。

模板数据库表示一个常用数据库。它包含系统表、标准视图、函数和数据类型的定义。 pgdata 用作的参数 initdb 指定数据库集群位置的脚本。

PostgreSQL中的所有数据库对象都由各自的OID在内部管理。表和索引也由各个OID管理。数据库对象及其各自的OID之间的关系存储在相应的系统目录表中、具体取决于对象类型。例如、数据库和堆表的OID存储在中 pg\_database 和`pg\_class。您可以通过在PostgreSQL客户端上发出查询来确定这些OID。

每个数据库都有自己的表和索引文件、这些表和索引文件的大小限制为1 GB。每个表都有两个关联文件、后缀分别为 \_ fsm 和 \_ vm。它们称为可用空间映射和可见性映射。这些文件用于存储有关可用空间容量的信息、并可查看表文件中的每个页面。索引只具有单个可用空间映射、而不具有可见性映射。

。 pg\_xlog/pg\_wal 目录包含预写日志。预写日志用于提高数据库可靠性和性能。每当您更新表中的某一行时、PostgreSQL都会首先将更改写入预写日志、然后将修改写入实际数据页面到磁盘。。 pg\_xlog 目录通常包含多个文件、但initdb仅创建第一个文件。根据需要添加其他文件。每个xlog文件的长度为16 MB。

## 设置

可以通过多种PostgreSQL调整配置来提高性能。

最常用的参数如下：

- max\_connections = <num>：一次可建立的最大数据库连接数。使用此参数可限制磁盘交换并导致性能下降。根据您的应用程序要求、您还可以针对连接池设置调整此参数。
- shared\_buffers = <num>：提高数据库服务器性能的最简单方法。对于大多数现代硬件、默认值为 low。在部署期间、它会设置为系统上可用RAM的大约25%。此参数设置因其在特定数据库实例中的工作方式而异；您可能需要按试用和错误增加和减小值。但是、将其设置为高可能会降低性能。
- effective\_cache\_size = <num>：该值告诉PostgreSQL的优化器PostgreSQL有多少内存可用于缓存数据，并帮助确定是否使用索引。值越大、使用索引的可能性就越大。此参数应设置为分配给的内存量 shared\_buffers 加上可用的操作系统缓存容量。此值通常超过系统总内存的50%。
- work\_mem = <num>：该参数控制在排序操作和哈希表中使用的内存量。如果您在应用程序中进行大量排序，则可能需要增加内存量，但要小心。它不是系统范围的参数、而是每个操作的参数。如果复杂查询包含多个排序操作、则它会使用多个work\_mEM单元内存、多个后端可以同时执行此操作。如果值过大、此查询通常会导致数据库服务器进行交换。此选项以前在早期版本的PostgreSQL中称为Sort\_mm。
- fsync = <boolean> (on or off)：此参数用于确定在提交事务之前是否应使用fsync()将所有WAL页面同步到磁盘。关闭它有时可以提高写入性能、而打开它可以增强保护、防止在系统崩溃时发生损坏。
- checkpoint\_timeout：检查点进程将已提交的数据转至磁盘。这涉及到磁盘上的大量读/写操作。该值以秒为单位进行设置、较低的值可缩短崩溃恢复时间、增加该值可通过减少检查点调用来减少系统资源的负载。根据应用程序的严重程度、使用情况和数据库可用性、设置checkpoint\_timeout的值。
- commit\_delay = <num> 和 commit\_siblings = <num>：这些选项结合使用，可以同时写出多个提交的事务，从而帮助提高性能。如果在提交事务时有多个commit兄弟姐妹对象处于活动状态、则服务器将等待commit delay微秒尝试一次提交多个事务。
- max\_worker\_processes / max\_parallel\_workers：为流程配置最佳数量的员

工。`max_parallel_workers`对应于可用的CPU数量。根据应用程序设计、查询可能需要较少的工作人员来执行并行操作。最好保持两个参数的值相同、但在测试后调整该值。

- `random_page_cost = <num>`: 该值控制PostgreSQL查看非顺序磁盘读取的方式。值越高、意味着PostgreSQL更有可能使用顺序扫描而不是索引扫描、这表示您的服务器具有快速磁盘在评估基于计划的优化、清理、索引以更改查询或架构等其他选项后修改此设置。
- `effective_io_concurrency = <num>`: 此参数用于设置PostgreSQL尝试同时执行的并发磁盘I/O操作的数量。提高此值会增加任何单个PostgreSQL会话尝试并行启动的I/O操作的数量。允许的范围为1到1,000或零、用于禁止发出异步I/O请求。目前、此设置仅影响位图堆扫描。固态驱动器(SSD)和其他基于内存的存储(NVMe)通常可以处理多个并发请求、因此、数百个请求的最大价值可能是。

有关PostgreSQL配置参数的完整列表、请参见PostgreSQL文档。

## 吐司

TOAST表示超大属性存储技术。PostgreSQL使用固定的页面大小(通常为8 KB)、并且不允许元组跨越多个页面。因此、无法直接存储大字段值。当您尝试存储超过此大小的行时、TOAST会将大型列的数据拆分成较小的"部分"、并将其存储在TOAST表中。

只有在将结果集发送到客户端时、才会提取已分配属性的大值(如果已选中)。表本身要小得多、并且可以在共享缓冲区缓存中容纳更多行、而不是在没有任何线外存储(TOAST)的情况下。

## 真空

在正常的PostgreSQL操作中，被更新删除或废弃的元组不会从其表中物理删除；它们会一直存在，直到运行真空为止。因此、您必须定期运行真空、尤其是在频繁更新的表上。然后、必须回收占用的空间、供新行重复使用、以避免磁盘空间中断。但是、它不会将空间归还给操作系统。

页面中的可用空间不会碎片化。真空会重新写入整个数据块、从而高效地填充其余行、并在一页中保留一个连续的可用空间块。

相比之下、真空全满会通过写入不含死空间的全新表文件来主动压缩表。此操作可最大程度地减小表大小、但可能需要很长时间。此外、还需要额外的磁盘空间来创建新的表副本、直到操作完成为止。常规真空的目标是避免真空完全发挥作用。此过程不仅可以将表保持在其最小大小、还可以保持磁盘空间的稳定使用状态。

## 表空间

初始化数据库集群时、系统会自动创建两个表空间。

。 `pg_global` 表空间用于共享系统目录。。 `pg_default` 表空间是`template1`和`template0`数据库的默认表空间。如果用于初始化集群的分区或卷用尽空间且无法扩展、则可以在其他分区上创建并使用表空间、直到可以重新配置系统为止。

使用率较高的索引可以放置在快速、高可用性的磁盘上、就像固态设备一样。此外、存储很少使用或不影响性能的归档数据的表可以存储在成本较低、速度较慢的磁盘系统上、例如SAS或SATA驱动器。

表空间是数据库集群的一部分、不能视为数据文件的自主集合。它们依赖于主数据目录中包含的元数据、因此无法连接到其他数据库集群或单独备份。同样、如果丢失表空间(由于文件删除、磁盘故障等原因)、数据库集群可能会变得不可读或无法启动。将表空间放置在临时文件系统(如RAM磁盘)上会危及整个集群的可靠性。

创建表空间后、如果请求用户有足够的权限、则可以使用任何数据库中的表空间。PostgreSQL使用符号链接

来简化表空间的实现。PostgreSQL在中添加一行 pg\_tablespace 表(一个全组范围的表)、并为该行分配一个新的对象标识符(OID)。最后、服务器使用OID在集群和给定目录之间创建符号链接。目录 \$PGDATA/pg\_tblspc 包含指向集群中定义的每个非内置表空间的符号链接。

## 版权信息

版权所有 © 2026 NetApp, Inc.。保留所有权利。中国印刷。未经版权所有者事先书面许可，本文档中受版权保护的任何部分不得以任何形式或通过任何手段（图片、电子或机械方式，包括影印、录音、录像或存储在电子检索系统中）进行复制。

从受版权保护的 NetApp 资料派生的软件受以下许可和免责声明的约束：

本软件由 NetApp 按“原样”提供，不含任何明示或暗示担保，包括但不限于适销性以及针对特定用途的适用性的隐含担保，特此声明不承担任何责任。在任何情况下，对于因使用本软件而以任何方式造成的任何直接性、间接性、偶然性、特殊性、惩罚性或后果性损失（包括但不限于购买替代商品或服务；使用、数据或利润方面的损失；或者业务中断），无论原因如何以及基于何种责任理论，无论出于合同、严格责任或侵权行为（包括疏忽或其他行为），NetApp 均不承担责任，即使已被告知存在上述损失的可能性。

NetApp 保留在不另行通知的情况下随时对本文档所述的任何产品进行更改的权利。除非 NetApp 以书面形式明确同意，否则 NetApp 不承担因使用本文档所述产品而产生的任何责任或义务。使用或购买本产品不表示获得 NetApp 的任何专利权、商标权或任何其他知识产权许可。

本手册中描述的产品可能受一项或多项美国专利、外国专利或正在申请的专利的保护。

有限权利说明：政府使用、复制或公开本文档受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中“技术数据权利 — 非商用”条款第 (b)(3) 条规定的限制条件的约束。

本文档中所含数据与商业产品和/或商业服务（定义见 FAR 2.101）相关，属于 NetApp, Inc. 的专有信息。根据本协议提供的所有 NetApp 技术数据和计算机软件具有商业性质，并完全由私人出资开发。美国政府对这些数据的使用权具有非排他性、全球性、受限且不可撤销的许可，该许可既不可转让，也不可再许可，但仅限在与交付数据所依据的美国政府合同有关且受合同支持的情况下使用。除本文档规定的情形外，未经 NetApp, Inc. 事先书面批准，不得使用、披露、复制、修改、操作或显示这些数据。美国政府对国防部的授权仅限于 DFARS 的第 252.227-7015(b)（2014 年 2 月）条款中明确的权利。

## 商标信息

NetApp、NetApp 标识和 <http://www.netapp.com/TM> 上所列的商标是 NetApp, Inc. 的商标。其他公司和产品名称可能是其各自所有者的商标。