資料庫組態
Enterprise applications

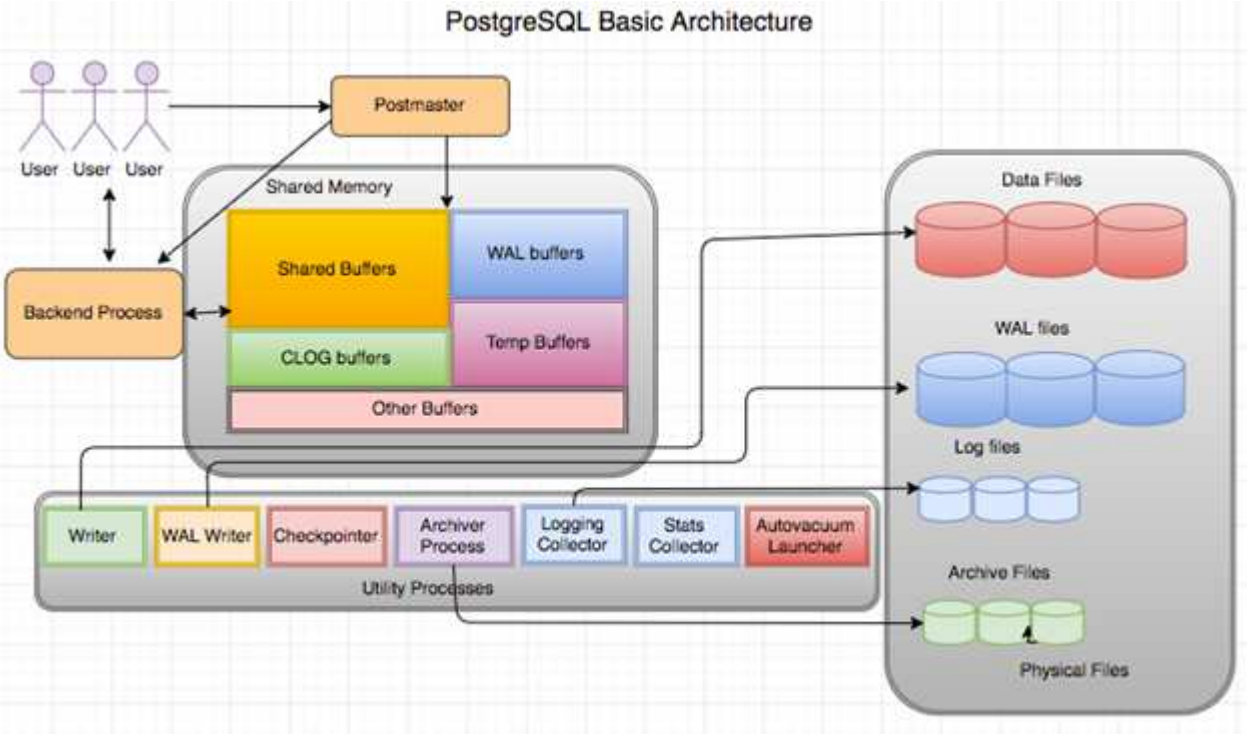
NetApp
May 09, 2024

目錄

資料庫組態

PostgreSQL 架構

PostgreSQL 是以用戶端和伺服器架構為基礎的 RDBMS 。PostgreSQL 執行個體稱為資料庫叢集、是資料庫的集合、而非伺服器集合。



PostgreSQL 資料庫中有三個主要元素：郵政局長、前端（用戶端）和後端用戶端會傳送要求給郵政局長、其中包含 IP 傳輸協定等資訊、以及要連線的資料庫。郵政局長會驗證連線、並將其傳送至後端程序以進行進一步的通訊。後端程序會執行查詢、並將結果直接傳送至前端（用戶端）。

PostgreSQL 執行個體是以多重處理模式為基礎、而非多重執行緒模式。它會為不同的工作產生多個處理程序、而且每個處理程序都有自己的功能。主要程序包括用戶端程序、 Wal 寫入程序、背景寫入程序及檢查指標程序：

- 當用戶端（前景）程序傳送讀取或寫入要求至 PostgreSQL 執行個體時、它不會直接讀取或寫入資料至磁碟。它首先緩衝共享緩衝區和預先寫入記錄（ Wal ）緩衝區中的資料。
- Wal 寫入器程序會操控共用緩衝區和 Wal 緩衝區的內容、以寫入 Wal 記錄檔。Wal 記錄檔通常是 PostgreSQL 的交易記錄檔、並依序寫入。因此、為了改善資料庫的回應時間、 PostgreSQL 會先寫入交易記錄檔、並確認用戶端。
- 若要將資料庫置於一致的狀態、背景寫入器程序會定期檢查共用緩衝區是否有髒頁。然後將資料排清至儲存在 NetApp 磁碟區或 LUN 上的資料檔案。
- checkpointer 程序也會定期執行（比背景程序更少）、並防止對緩衝區進行任何修改。它會向 Wal 寫入器程序發出訊號、將檢查點記錄寫入並清除至儲存在 NetApp 磁碟上的 Wal 記錄檔結尾。它也會向背景寫入器程序發出訊號、要求將所有髒頁寫入磁碟並清除。

PostgreSQL 初始化參數

您可以使用建立新的資料庫叢集 `initdb` 方案。一 `initdb` 指令碼會建立定義叢集的資料檔案、系統表格和範本資料庫（ template0 和 template1 ）。

範本資料庫代表常用資料庫。其中包含系統表格、標準檢視、功能和資料類型的定義。 `pgdata` 做為的引數 `initdb` 指定資料庫叢集位置的指令碼。

PostgreSQL 中的所有資料庫物件都是由各自的 OID 在內部管理。表格和索引也由個別的 OID 管理。資料庫物件與其各自的 OID 之間的關係會儲存在適當的系統目錄表格中、視物件類型而定。例如、資料庫和堆積表格的 OID 會儲存在中 `pg_database` 和 "pg_class" 。您可以在 PostgreSQL 用戶端上發佈查詢來判斷 OID 。

每個資料庫都有自己的個別資料表和索引檔案、限制為 1GB 。每個表格都有兩個相關的檔案、分別以後綴表示 `_fsm` 和 `_vm`。它們稱為可用空間地圖和可見度地圖。這些檔案會儲存可用空間容量的相關資訊、並在表格檔案中的每個頁面上都有可見度。索引只有個別的可用空間地圖、而且沒有可見度地圖。

。 `pg_xlog/pg_wal` 目錄包含預先寫入記錄。預先寫入記錄可用來改善資料庫的可靠性和效能。每當您更新表格中的列時、 PostgreSQL 會先將變更寫入預先寫入記錄、然後將修改寫入實際的資料頁面到磁碟。。 `pg_xlog` 目錄通常包含數個檔案、但 initdb 只會建立第一個檔案。視需要新增額外檔案。每個 xlog 檔案長度為 16MB 。

ONTAP 的 PostgreSQL 資料庫組態

有幾種 PostgreSQL 調校組態可以改善效能。

最常用的參數如下：

- `max_connections = <num>`：一次擁有的最大資料庫連線數。使用此參數可限制磁碟交換並終止效能。根據應用程式需求、您也可以針對連線集區設定調整此參數。
- `shared_buffers = <num>`：提高資料庫伺服器效能的最簡單方法。對於大多數現代硬體而言、預設值為低。在部署期間、系統上的可用 RAM 約為 25% 。此參數設定會因其與特定資料庫執行個體的運作方式而異；您可能必須根據試用和錯誤來增加和減少值。不過、將其設為高可能會降低效能。
- `effective_cache_size = <num>`：此值告訴 PostgreSQL 的最佳化程式 PostgreSQL 有多少記憶體可用於快取資料、並有助於判斷是否使用索引。較大的值會增加使用索引的可能性。此參數應設為分配給的記憶體容量 `shared_buffers` 加上可用的作業系統快取容量。此值通常超過系統總記憶體的 50% 。
- `work_mem = <num>`：此參數控制用於排序作業和雜湊表的記憶體容量。如果您在應用程式中進行大量排序、可能需要增加記憶體容量、但請謹慎。它不是系統範圍的參數、而是每次操作的參數。如果複雜查詢中有多個排序作業、則會使用多個 work_mem 記憶體單元、而多個後端也可能同時執行此作業。如果值太大、此查詢通常會引導您的資料庫伺服器進行切換。此選項先前在舊版 PostgreSQL 中稱為 sort_mem 。
- `fsync = <boolean> (on or off)`：此參數確定在提交事務之前是否應使用 fsync （）將所有 Wal 頁面同步到磁盤。關閉它有時會改善寫入效能、並將其開啟、以提高系統當機時避免毀損的風險。
- `checkpoint_timeout`：檢查點處理程序會將已提交的資料清除至磁碟。這涉及磁碟上的大量讀寫作業。此值以秒為單位設定、較低的值可減少損毀恢復時間、而增加的值則可減少檢查點呼叫、進而降低系統資源的負載。根據應用程式的關鍵程度、使用量、資料庫可用度、設定 checkpoint 逾時的值。
- `commit_delay = <num>` 和 `commit_siblings = <num>`：這些選項可同時用於撰寫多筆同時提交的交易、以協助改善效能。如果交易提交時有多個 command_sibliblings 物件處於作用中狀態、伺服器會等待 command_delay 微秒、嘗試一次提交多個交易。

- `max_worker_processes / max_parallel_workers`：配置流程的最佳工作人員數量。max_paralle_founds 對應可用的 CPU 數量。視應用程式設計而定、查詢可能需要較少的工作人員來執行平行作業。最好保持兩個參數的值相同、但在測試後調整值。
- `random_page_cost = <num>`：此值控制 PostgreSQL 檢視非連續磁碟讀取的方式。較高的值表示 PostgreSQL 較可能使用連續掃描、而非索引掃描、表示伺服器有快速磁碟。請在評估其他選項（例如計畫型最佳化、吸塵、索引以變更查詢或架構）之後、修改此設定。
- `effective_io_concurrency = <num>`：此參數設置 PostgreSQL 嘗試同時執行的並行磁盤 I/O 操作數。提高此值會增加任何個別 PostgreSQL 工作階段嘗試平行啟動的 I/O 作業數。允許範圍為 1 到 1 、 000 、或為零、以停用非同步 I/O 要求的發出。目前、此設定只會影響點陣圖堆疊掃描。固態硬碟（ SSD ）和其他記憶體型儲存設備（ NVMe ）通常可以處理許多並行要求、因此最佳價值可能在數百種環境中。

如需 PostgreSQL 組態參數的完整清單、請參閱 PostgreSQL 文件。

吐司

Toast 代表「超大型屬性儲存技術」。PostgreSQL 使用固定的頁面大小（通常為 8KB ）、不允許 Tuple 跨越多個頁面。因此、無法直接儲存大欄位值。當您嘗試儲存超過此大小的資料列時、 Toast 會將大型資料欄的資料分成較小的「片段」、並將其儲存在 Toast 資料表中。

只有在將結果集傳送至用戶端時、才會拔出（如果完全選取）已烘烤的屬性大值。表格本身比沒有任何離線儲存設備（ Toast ）時更小、可容納更多資料列到共用緩衝區快取中。

真空

在正常的 PostgreSQL 作業中、因為更新而刪除或過時的 Tuple 不會從其表格中實際移除、直到執行真空為止。因此、您必須定期執行吸氣、尤其是在經常更新的表格上。接著必須回收它所佔用的空間、以便新列重複使用、以避免磁碟空間中斷。不過、它不會將空間傳回作業系統。

頁面內的可用空間不會分散。真空會重新寫入整個區塊、有效地將剩餘的資料列封裝起來、並在頁面中留下一個連續的可用空間區塊。

相反地、使用「完全真空」技術、可以撰寫完全新版的表格檔案、而不會有任何空間。此動作可將表格大小減至最小、但可能需要很長時間。在作業完成之前、它也需要額外的磁碟空間來容納表格的新複本。例行真空的目標是避免完全真空。此程序不僅能將資料表保持在最小大小、還能維持磁碟空間的穩定狀態使用量。

PostgreSQL 表格空間

初始化資料庫叢集時會自動建立兩個資料表空間。

。 `pg_global` 表空間用於共享系統目錄。。 `pg_default` 表空間是 template1 和 template0 資料庫的預設資料表空間。如果初始化叢集的磁碟分割或磁碟區空間不足且無法擴充、則可在不同的磁碟分割上建立資料表空間、並在重新設定系統之前使用。

大量使用的索引可以放在快速、高可用度的磁碟上、例如固態裝置。此外、儲存歸檔資料的表格、無論是極少使用或非效能關鍵、都可以儲存在成本較低、速度較慢的磁碟系統上、例如 SAS 或 SATA 磁碟機。

資料表空間是資料庫叢集的一部分、無法視為資料檔案的獨立集合。它們取決於主資料目錄中包含的中繼資料、因此無法附加至不同的資料庫叢集或個別備份。同樣地、如果您遺失資料表空間（透過檔案刪除、磁碟故障等方式）、資料庫叢集可能會變得無法讀取或無法啟動。將資料表空間放在像 RAM 磁碟一樣的暫存檔案系統上、會危及整個叢集的可靠性。

建立表格後、如果要求的使用者擁有足夠的權限、則可以從任何資料庫使用表格區。PostgreSQL 使用符號連結來簡化資料表空間的實作。PostgreSQL 會在中新增一列 `pg_tablespace` 表（叢集範圍表格）、並將新的物件識別碼（ OID ）指派給該列。最後、伺服器會使用 OID 在叢集與指定目錄之間建立符號連結。目錄 `$PGDATA/pg_tblspc` 包含指向叢集中定義的每個非內建表格空間的符號連結。