TDSQL-A for PostgreSQL

Glossary

Product Documentation

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Glossary

Last updated : 2021-07-02 16:46:16

Instance

It is a complete database that contains GTM, CN, and DN with independent and isolated resources.

Node

It is a DN, CN, or GTM.

Node set

The DN and CN forms a node set (aka set) in the primary-standby architecture.

Node name

For CN and DN, a set of primary/standby nodes share the same name, such as `cn0001` . For GTM, each GTM has a unique name, which is not shared by primary/standby GTMs.

Coordinator

A coordinator (CN) provides APIs and is responsible for data distribution, query planning, pairing of multiple node locations, etc. Each node provides the same database views. In terms of functionality, it stores only the global metadata of the system rather than actual business data.

Datanode

A datanode (DN) stores metadata related to itself and shards of the business data. In terms of functionality, it executes the execution requests distributed by the CN.

Forward node

A forward node (FN) is a data forwarding node used to reduce the network overheads and links for cross-node data interactions. Each instance has an FN on each physical machine to sustain some data interactions between logical nodes on the machine. The network connection has nothing to do with the number of redistribution layers, and it is more efficient to redistribute a large amount of data. In addition, data forwarding from the same node can bypass the network layer, which further improves the data transfer efficiency.

Data forward bus

The data forward bus (DFB) consists of FNs on servers in the cluster. The main purpose of using FNs is to reduce the number of connections created during data exchange between DNs and between CNs and DNs, thus ensuring that connection will not become a bottleneck for large-scale clusters.

Global transaction manager

The global transaction manager (GTM) manages instance transaction information and global instance objects such as sequences.

cgroup

It is the abbreviation of control groups, which is a feature of the Linux kernel and used to limit, control, and separate resources (such as CPU, memory, and disk I/O) of a process group.

dngroup

It is a logical group where DNs of TBase instances are grouped, and each group contains one or multiple nodes and one shardmap.

shardmap

It is a view for managing logical mapping relationships and maps business data to DNs determined according to the intermediate result values calculated by the shardkey value.

xlog

It is a WAL log file on a TDSQL-A for PostgreSQL node.

Row storage

In this storage mode, data entries are stored in files in the same logical order. All columns of data in a row are stored sequentially on the physical disk (multiple fields are usually stored sequentially in one disk file). The benefits of this format is obvious, as it generally requires only one disk I/O when multiple columns of data in a row are accessed at the same time, making it ideal for OLTP loads.

Column storage

In this storage mode, each column of data in a table is stored as an independent disk file; for example, each column of data such as "Name", "Department", "Salary", and "Family Information" is a data file. This format can greatly reduce disk I/O overheads compared to row storage when you need to access only a few columns in the table at a time, especially in aggregation scenarios; therefore, it is mostly used in OLAP systems.

Distributed transaction

The effect of implementing database transaction ACID in a distributed topology is the same as that in a standalone database.

Permission separation

The traditional database system DBA role is divided into three independent roles: security admin, audit admin, and data admin, and the three roles are mutually restricted to eliminate super privileges in the system, solving data security problems from the aspect of system role design.

Security admin

This role defines security policies such as forced access, data masking, and encryption policies independently of the database admin.

Audit admin

This role audits all operations and defines audit policies independently of the database admin. Its operations are forcibly logged and cannot be changed.

Data admin

This role has independent access control and OPS permissions but cannot interfere with the operations of security and audit admins.

Column-Level encryption

It refers to the separate encryption of a specified column.

File encryption

It uses block (standard block within database) as the smallest unit of encrypted computing.

Transparent data masking

It refers to mapping the original real data to another (variable) form by a certain algorithm according to an established rule before the real data is returned to the access terminal. The mapping rule is invisible to the querying user, and the converted form cannot be reversed (that is, conversion to the original data is impossible).

Data masking hides sensitive information while retaining the data integrity and validity as much as possible, so that when all users (including admins) in the original system view the table data again, they will only get the results that have been converted according to the masking rule. The rule will take effect immediately after distribution without downtime or restart required.

Users with normal access to the database can be defined as authorized users. The security admin can add such users to the allowlist of certain objects, that is, allowlist attributes are added to such users. After logging into the system and completing authentication, users in the allowlist will be recognized as authorized users by the system. The way they access tables stays the same, and they can get the original data.

Data skew

It refers to the imbalance of data volumes between DNs.

Hot/Cold data separation

It distinguishes the data access frequency at the business level, dumps data with infrequent access to cheap storage servers, retains data with frequent access on high-performance storage servers, and works with the hot/cold migration tasks of OSS to reduce the storage costs of the entire instance while keeping the whole process imperceptible to the business.

Online scaling

It refers to horizontally adjusting the instance topology during business operations, which is equivalent to scale-out. The data migration process is imperceptible to the business.

SQL:2011

It is an ANSI SQL standard as well as a standard programming language used by relational databases.

SQL:2003

It is an ANSI SQL standard as well as a standard programming language used by relational databases.

PostGIS

It is an open-source program that provides the storage of spatial and geographic data to make TDSQL-A for PostgreSQL a spatial database. In this way, TDSQL-A for PostgreSQL can support efficient spatial data management, quantitative measurement, and geometric topology analysis through SQL statements. PostGIS enables the SQL implementations for the basic element classes (Point, LineString, Polygon, MultiPoint, MultiLineString, PolyhedralSurface, etc.) proposed by the Open Geospatial Consortium. It uses well-known text (an annotation method for representing spatial objects in text) and well-known binary (a storage method for representing spatial objects in binary streams) to store spatial objects in the database.

JSON data type

It is a data format with an advantage where each stored value must conform to JSON rules. Many JSON-related functions and operators can be used to store JSON data. JSON data type is divides into JSON and JSONB, which can share the exact same set of values as input, and their main difference lies in the efficiency. The JSON data type stores an exact copy of the input text, and the processing function must parse the data again in every execution, while JSONB data is stored in a decomposed binary format, which is a little slower to be input as additional conversion needs to be performed.

However, JSONB is much faster to process data as there is no parsing required. JSONB also supports indexing, which is also a compelling advantage.