基于fdw的跨Greenplum集群数据库查询实现

郭罡 Greenplum内核研发

需求来源

- 数据库联邦：也存在于不同别的数据库
- 通用需求: 无缝方便在不同Greenplum集群见查询（包括join等）
- 先看看Postgres的做法再看Greenplum做法

Postgres跨节点数据库查询

- Greenplum是基于Postgres的MPP数据库
- Postgres如何跨节点数据库查询？
  - foreign data wrapper (fdw): postgres_fdw
  - dblink
- fdw更透明使用、标准语法、性能更好（每个PG版本都在提高和有更多功能）
- postgres_fdw是postgres连接postgres的fdw模块

Postgres跨节点数据库查询（续）- postgres_fdw使用

- 典型postgres_fdw使用方法
  - create extension postgres_fdw; -- 装载postgres_fdw模块
  - CREATE SERVER myserver FOREIGN DATA WRAPPER postgres_fdw OPTIONS (host 'foo', dbname 'foodb', port '5432');
  - CREATE USER MAPPING FOR bob SERVER myserver OPTIONS (user 'bob', password 'secret');
  - CREATE FOREIGN TABLE foreign_table (id integer NOT NULL, data text)SERVER myserver;
    或者 IMPORT FOREIGN SCHEMA
  - 如何使用呢？和正常表一样操作（读甚至写）Foreign表。

GREENPLUM DATABASE

Postgres跨节点数据库查询（续）- postgres_fdw功能

Postgres社区持续不断提高fdw的功能和性能

Version	Description
9.3	postgres_fdw module is released.
9.6	Consider performing sorts on the remote server. Consider performing joins on the remote server. When feasible, perform UPDATE or DELETE entirely on the remote server. Allow the fetch size to be set as a server or table option.
10	Push aggregate functions to the remote server, when possible.
11	Allow to push down aggregates to foreign tables that are partitions. Allow to push UPDATEs and DELETEs using joins to foreign servers.
12	Allow ORDER BY sorts and LIMIT clauses to be pushed in more cases.

htt

GREENPLUM DATABASE

Postgres跨节点数据库查询（续）- postgres_fdw功能

Postgres 13和今年下半年要发布的Postgres 14做了更多功能和性能增强

https://www.percona.com/blog/2020/09/30/postgresql_fdw-authentication-changes-in-postgresql-13/

功能包括：passwordless, ssl认证

https://www.percona.com/blog/2021/06/02/postgres_fdw-enhancement-in-postgresql-14/

功能包括：parallel foreign scan, bulk insert, truncate等等

Postgres跨节点数据库查询（续）- postgres_fdw如何读?

- Foreign Scan执行器节点
  - 创建一个Cursor来查询远端的表
  - 调用Fetch语句返回给上层执行器tuple
- Postgres_fdw代码提供各种fdw接口的callback，比如用于Scan操作的：

```
/* Functions for scanning foreign tables */
routine->GetForeignRelSize = postgresGetForeignRelSize;
routine->GetForeignPaths = postgresGetForeignPaths;
routine->GetForeignPlan = postgresGetForeignPlan;
routine->BeginForeignScan = postgresBeginForeignScan;
routine->IterateForeignScan = postgresIterateForeignScan;
routine->ReScanForeignScan = postgresReScanForeignScan;
routine->EndForeignScan = postgresEndForeignScan;
```

跨Greenplum集群cluster查询怎么做？

- 借助postgres_fdw
- QD->QD的postgres_fdw其实理论就可用了，但是都走master通常会慢！
- 最好的办法是根据查询计划，本地segments通过各自的postgres_fdw读取远端的segments的数据
- 不要求N:N （本地和远端segment数量一样），允许异构!

跨Greenplum集群cluster查询设计

- 新的cursor类型语法：Parallel Retrieve Cursor
- 修改postgres_fdw，支持和使用这种cursor
- 本地segments通过各自的cursor连接远端一个segment来做fdw的查询
- 为什么需要parallel retrieve cursor，不用local的cursor？
  - Coordinator/Master上统一认证、统一管理、全局事务。
  - 注意Parallel Retrieve Cursor可以不局限于跨Greenplum查询，有更多的使用想象空间。

Parallel Retrieve Cursor使用demo

- 启动一个到Greenplum集群的连接，然后运行下面SQL语句

```
=# BEGIN;
=# DECLARE c1 PARALLEL RETRIEVE CURSOR FOR SELECT * FROM t1;
=# SELECT * FROM gp_endpoints() WHERE cursorname='c1';
 gp_segment_id |          auth_token          | cursorname | sessionid | hostname | port | userid | state | endpointname
---------------+------------------------------+------------+-----------+----------+------+--------+-------+--------------
             0 | a612d36ac7bfb6b4a1f147cfd283dea2 | c1         |      3586 | host67   | 7002 |     10 | READY | c100000e02
             1 | d39e6b62200a9e19ba5ec21ac1fc8cab | c1         |      3586 | host67   | 7003 |     10 | READY | c100000e02
             2 | a4e6fcd468039e77f91e62578deb40b9 | c1         |      3586 | host67   | 7004 |     10 | READY | c100000e02
(3 rows)
```

解释：endpoint代表这个cursor在某个执行节点的内部单元（根据计划，endpoints可能在某个节点或者多个节点）。

- 启动retrieve类型连接到某个segment读取数据

```
$ PGOPTIONS='-c gp_retrieve_conn=true' psql -p 7002
Password for user pguo: <输入：auth_token>
postgres=# RETRIEVE 2 FROM ENDPOINT c100000e02; <- 新的语法获取数据，类似于FETCH。
 a
---
 2
 3
(2 rows)
```

Parallel Retrieve Cursor使用

- Retrieve connection只允许RETRIEVE语句（也只能在Retrieve连接上使用）
- 问题：
  - 为什么不在RETRIEVE语句中用cursor？

    因为要防止不同session的parallel retrieve cursor名字重名冲突

  - 为什么需要指定endpoint？

    因为一个session用户可以declare多个cursor

- 其他UDFs

gp_check_parallel_retrieve_cursor(cursorname) - 返回是否所有数据获取完毕

gp_wait_parallel_retrieve_cursor(cursorname) - 等候直到所有数据获取完毕

gp_segment_endpoints() - 某个segment上endpoint信息

GREENPLUM DATABASE

Parallel Retrieve Cursor技术实现

- 类似于Greenplum、生成Plan，区别是不需要有一个最顶层的gather motion
- 类似的Greenplum的gang管理（QD->QE连接，cursor语句reader gang）
- Retrieve连接后端和QE的cursor后端进程通过PG的shm_mq来通信数据

  shm_mq也用于PG的Parallel Scanning处理

  具体来说，通过基于shm_mq创建的Dest Receiver，使用Endpoint数据结构

- 可靠的代码逻辑处理：错误处理代码要很小心

如何做跨GP集群查询？

- 本地GP集群QD节点：Foreign Scan执行器节点预先在远程GP集群创建相应的Parallel Retrieve Cursor并记录相应信息(调用gp_endpoints())到plan中（比如登录token等）

  调用BeginForeignScan()接口

- 本地GP集群的QE节点，通过这些信息建立retrieve连接

  同样调用BeginForeignScan()接口

- 本地QE节点从远端读取数据（通过RETRIEVE语句）

  调用IterateForeignScan()接口

项目情况

- 计划：Greenplum 6 and above
- Parallel Retrieve Cursor部分基本完成，主要是稳定性等余下工作
- Fdw方面工作(local cluster)相对简单，已近完成，主要是联调和测试
- 现在基于master分支代码，会Backport到Greenplum 6
- 更多将来相关工作（Foreign表Distributed by)

GREENPLUM DATABASE®

微信技术讨论群
微信搜索添加“gp_assistant”
加入技术讨论

微信公众号
搜索添加“Greenplum中文社区”
技术干货、行业热点、活动预告

GREENPLUM DATABASE

Greenplum中文社区

https://cn.greenplum.org

文档 · 博文 · 问答 · 项目 · 下载