

# 基于 PostgreSQL 的海量准实时数据服务平台访问方案<sup>①</sup>



李 钢, 茅海泉

(南瑞集团有限公司(国网电力科学研究院有限公司), 南京 211000)  
(江苏瑞中数据股份有限公司, 南京 210012)

**摘 要:** 大数据服务平台在电网系统中具有重要的应用价值. 海量准实时数据平台作为电网重要数据中心, 存储了大量有价值的信息, 其存储的数据多为非结构化数据, 无法直接通过 SQL 语句获取. 为了最大限度挖掘数据价值, 需要在平台中集成 SQL 访问实时数据的功能. 为此, 本文基于 PostgreSQL Foreign Data Wrapper (Postgres\_fdw) 开发了 SQL 引擎. 通过分析 PostgreSQL 的功能特点以及已有 UAPI 接口的不足, 针对原有大数据平台框架, 采用分层策略, 基于 Postgres\_fdw 开发了 SQL 引擎, 并提出了针对海量平台中实时数据库的外部表空间估算方案和 SQL 解析方案, 以此实现数据平台运用 SQL 的方式获取底层海量平台数据的功能. 现场功能和性能测试, 验证了该引擎的有效性.

**关键词:** 大数据平台; PostgreSQL; 海量准实时数据; SQL 引擎

引用格式: 李钢, 茅海泉. 基于 PostgreSQL 的海量准实时数据服务平台访问方案. 计算机系统应用, 2019, 28(2): 274-279. <http://www.c-s-a.org.cn/1003-3254/6775.html>

## PostgreSQL-Based Big Data Access Scheme of Massive Quasi-Real-Time Data Service Platform

LI Gang, MAO Hai-Quan

(NARI Group Corporation/State Grid Electric Power Research Institute, Nanjing 211000, China)  
(China Realtime Database Co. Ltd., Nanjing 210012, China)

**Abstract:** Big data services platform has important applications in the power distribution system. Massive quasi-real-time data service platform is a key data center in power grid, and a lot of valuable data is stored in the platform. The data in platform is unstructured, and cannot be selected by SQL statement. To maximize mining the data value, platform needs to integrate function that the accesses to real-time data, for this develop SQL engine based on PostgreSQL foreign data wrapper (Postgres\_fdw). The SQL engine proposes a foreign table space estimation scheme and SQL resolution scheme for real-time databases in massive platforms. This access scheme provided good support for big data platform to get data from the underlying massive data service platform with an SQL way. Field function and performance tests validated the effectiveness of the engine.

**Key words:** big data platform; PostgreSQL; quasi-real-time data; SQL engine

### 引言

在电网系统中产生的准实时数据<sup>[1]</sup>, 可以被应用在调度、在线监测、计量等多种场景中. 海量准实时数

据服务平台(以下简称海量平台)是南方电网公司面向数据资源统一管理及针对实时数据管理的有力支撑平台, 可以对生产运行过程中各业务应用形成的实时历

① 收稿时间: 2018-08-01; 修改时间: 2018-09-05; 采用时间: 2018-09-20; csa 在线出版时间: 2019-01-28

史数据进行存储、集中、整合、共享和分析,同时提供了标准统一的 NOSQL 访问方式。

随着电网规模的不断扩大、信息系统的不断升级扩展,准实时数据呈现海量的特点,具有明显的大数据<sup>[2]</sup>特征。Spark<sup>[3]</sup>、HBase<sup>[4]</sup>等大数据服务平台是目前流行的大数据解决方案,得到广泛的应用。大数据服务平台可以运用到电网系统中,更好的解决海量准实时数据的存储、计算、分析等问题,具有很高的实用价值。大数据平台可以通过 JDBC 与外部数据源进行连接,获取电网档案、模型等关系数据。而海量准实时数据服务平台提供统一访问接口 UAPI 不支持 SQL 查询。如何将海量平台中非结构化的量测数据与模型、档案等关系数据通过统一 SQL 访问引擎结合起来,成为运用大数据服务平台的关键问题。

本文采用分层的策略,在上层大数据平台与底层海量平台间设计实现专门的 SQL 引擎。大数据平台通过该引擎,运用 SQL 的方式访问海量平台中的数据。SQL 引擎基于 PostgreSQL 设计实现,能解析大数据层的 SQL 语句并通过 UAPI 获取海量平台数据。现场测试结果验证了该接口的有效性。

## 1 关键技术

海量平台中,非结构化数据主要存储在实时数据库中。目前国产的实时数据库仅支持简单的 SQL 查询,无法支持复杂 SQL 的查询,如与结构化数据结合的嵌套查询或者 join 查询等。因此,需要开发专用的 SQL 引擎来实现结构化数据与非结构化数据的联合查询,为大数据应用提供数据。基于 PostgreSQL 的 SQL 引擎是一种很有效的解决方案。

### 1.1 PostgreSQL

PostgreSQL 是由加州大学伯克利分校<sup>[5,6]</sup>开发的开源数据库系统,可以运行在 Windows、Linux 等多种平台上。PostgreSQL 是世界顶级关系数据库之一,被广泛运用在医疗、天文、商业等领域<sup>[7]</sup>。PostgreSQL 的主要技术特点包括:

1) 标准通用。PostgreSQL 基于广泛应用的关系数据模型,遵循 SQL 国际标准。PostgreSQL 为应用开发提供了符合标准的 ODBC3.0、JDBC3.0、OLEDB 2.7 和嵌入式 SQL 接口。PostgreSQL 经过大量实际应用的磨合与验证,已成为标准、通用、安全、稳定、实用、高效的数据存储与管理平台。

2) 海量存储。PostgreSQL 结合结构化查询语言的操作能力和过程化语言的数据处理能力,可以有效地支持大规模数据存储与存取,如 TB 级的数据库的表空间、GB 级的 BLOB 二进制大对象和 CLOB 文本大对象等,并通过各种约束保证数据的完整性和安全性。

3) Postgres\_fdw。Postgres\_fdw 为 PostgreSQL 数据库的外部数据封装器,可以用来访问存储在数据库外的数据。Postgres\_fdw 支持支持标准的 SQL select 语句,不支持 insert, update 等语句,通过 Postgres\_fdw,可采用统一的 SQL 访问不同数据源的数据。

4) 访问高效。PostgreSQL 采用 c 语言编写,运用多进程模型,支持高并发访问。将结果存储在内存中,有效避开无用的磁盘读写,具有高效访问的特点。

基于以上技术优势,本文采用 PostgreSQL 来设计统一 SQL 引擎。

### 1.2 UAPI

统一访问接口 (UAPI) 是海量平台对外提供的一套实时数据访问接口,UAPI 不支持 SQL 访问。UAPI 屏蔽了底层数据库的具体实现细节及差异,对外提供统一的访问接口为上层应用服务,对用户来说是只有一个逻辑实时数据库,从而实现了透明性以及分布式访问。

实时数据库统一访问接口的实现以标准 C/C++ 语言为编程语言并充分考虑接口的跨平台性,接口至少支持 Windows、Linux 等平台。

## 2 总体架构设计

### 2.1 引擎功能分析

在电网系统中运用大数据存储、分析工具,需要实现将底层海量平台的数据导入到大数据平台。大数据平台通过 JDBC 接口连接外部数据源,而海量平台已实现 UAPI 供外部调用。本文设计实现了基于 PostgreSQL 的 SQL 引擎,实现在大数据端通过 SQL 的方式访问、抽取海量平台实时数据。本文访问引擎实现的主要功能包括:

不包括 where 条件的查询。用于查询所有测点的信息。

带 where 条件的测点查询。查询符合特定条件的测点信息。此处的条件包括测点标识和测点名。

原始值、断面、实时值、差值<sup>[8]</sup>查询。查询测点不同类型的值信息。

模糊查询。便于查询符合特定条件的相关测点集

信息。

带子查询的 where 条件查询. 可以在查询语句中嵌套子查询, 方便更深层次的应用需求。

指定字段的查询. 用于查询测点的部分信息。

### 2.2 系统整体架构

大数据平台与海量平台间通过本文的访问插件进行连接. 具体架构如图 1 所示。

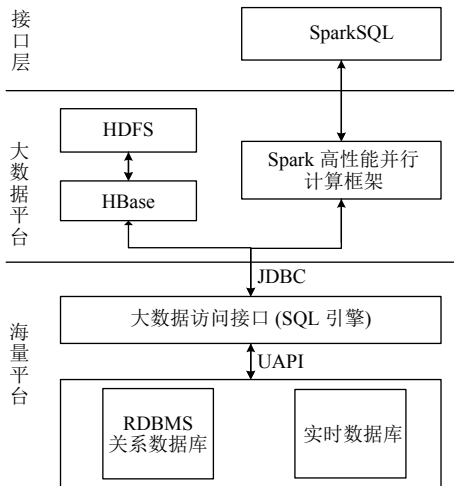


图 1 整体架构图

如图 1, 整体架构分为三层, 分别是:

接口层: 接口层是大数据平台的管理接口, 主要部署 SparkSQL<sup>[9]</sup>. SparkSQL 支持在 Java, Scala, Python 和 R 等高级语言中使用标准的 SQL 语句在 Spark 中查询结构化数据. 大数据分析应用调用 SparkSQL 接口实现对海量准实时数据服务平台的数据查询和分析计算。

大数据平台层: 大数据平台是电网系统中大数据工具的部署层, 主要涵盖 Spark、HDFS<sup>[10]</sup>和 HBase. Spark 负责准实时数据的计算分析, HDFS 和 HBase 解决实时历史数据的存储问题。

海量平台层: 海量平台层是目前电网系统中部署的实时数据管理平台, 负责对关系数据库、实时数据库等数据源的统一管理. 在底层的数据源上是本文重点设计实现的 SQL 解析引擎. 大数据访问接口通过 JDBC 与大数据平台连接, 与底层的海量平台通过 UAPI 连接. 整体上, 海量平台作为大数据平台的外部数据源, 通过本文的访问插件为上层提供数据支撑。

### 2.3 海量平台大数据接口模块架构

本文基于 PostgreSQL 的 FDW(Foreign Data

Wrapper) 框架, 设计海量平台大数据访问接口. 本接口对外支持标准 JDBC 访问方式. 海量准实时数据服务平台大数据接口模块架构图如图 2.



图 2 海量平台大数据接口模块架构图

海量准实时数据服务平台大数据接口模块由三部分组成:

- 1) 标准 JDBC 接口: 基于 PostgreSQL 对外提供的 JDBC 接口. 用于与上层的 Spark、HBase 等连接。
- 2) PostgreSQL 插件: 把 PostgreSQL 数据库作为海量平台大数据接口的一个功能插件. 此处主要利用 PostgreSQL 的外部表访问功能以及 PostgreSQL 对 SQL 语句的初步处理。
- 3) 海量平台 FDW 模块: 基于 PostgreSQL 的 FDW 模块, 实现对海量平台外部数据源的访问。

整个大数据接口模块中 FDW 模块是本文的核心. FDW 模块位于接口的底层, 负责连接 PostgreSQL 与海量平台. FDW 模块将系统上层大数据部分传入的 SQL 语句具体解析后调用相关的 UAPI 函数以实现准实时数据的传输。

## 3 海量平台 SQL 引擎设计

### 3.1 总体设计思路与主要流程

海量平台 SQL 引擎位于海量平台大数据接口的底层, 基于 PostgreSQL 的 FDW 模块设计实现. 整体的设计思路是将海量准实时数据服务平台中的数据抽象成 PostgreSQL 的外部表以支持访问。

电网应用中的准实时数据主要包括测点信息和测点值信息, 海量平台 FDW 模块相应的映射为 PostgreSQL 的外部点表 (upoint) 和外部值表 (uvalue). 外部表的字段与实时数据中包含的信息相对应, 例如外部值表中包含了测点标识、时间戳、质量位、值等字段。

基于上述思路, 设计海量平台 FDW 模块的流程图, 如图 3 所示。

海量平台 FDW 模块的主要实现流程包括:

连接海量准实时数据服务平台. 与海量平台建立连接是进行数据传输的第一步。



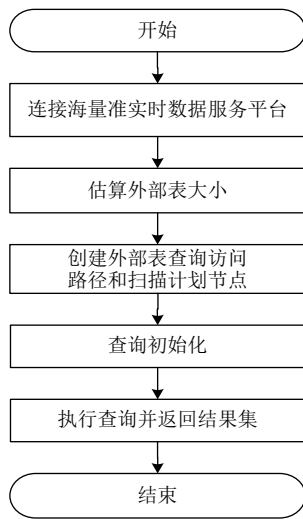


图3 海量平台 FDW 模块实现流程图

估算外部表大小. 估算映射的外部表的大小, 为后面查询做准备.

创建外部表查询访问路径和扫描计划节点. 根据外部调用的 IP 地址是否是本地, 设置不同的权重参数; 创建访问路径.

查询初始化. 主要是解析 SQL 语句. PostgreSQL 对 SQL 语句进行了简单的分词处理, 在 FDW 模块中还需进一步分析 SQL 的语义信息, 特别是 SQL 语句中 where 条件的处理.

执行查询并返回结果集. 根据解析的 SQL 语句, 调用对应的 UAPI 函数, 将结果集组织成 PostgreSQL 中的记录格式返回.

海量平台 SQL 引擎的各功能点协调有序处理, 能保证上层的 SQL 查询正确解析; 在映射成外部表的基础上, 调用相关 UAPI 函数, 顺利完成访问任务.

### 3.2 外部表大小估算方案

在查询外部表获得具体的字段信息前, 需要对外部表进行大小估算, 以生成更优的访问路径. 本文中外部表包括外部点表和外部值表, 分别设计了不同的外部表大小估算方案.

外部点表大小估算: 首先查询出总共的点数, 然后乘以每个点结构所占的大小即可得整体点表的大小.

外部值表大小估算方案较复杂, 具体流程如图4所示.

外部值表大小估算: 如图4, 在海量平台中, 每个测点包含的值的个数不等, 不能简单的根据单个测点值的个数乘以测点总数来估算. 而海量平台中测点数在

千万级别, 采用逐个测点查询累加的方式会严重损害效率. 本文采用随机抽样的方式, 抽取总点数的二分之一, 计算此部分的测点值总个数, 最后乘以比例得到整体的估计值.

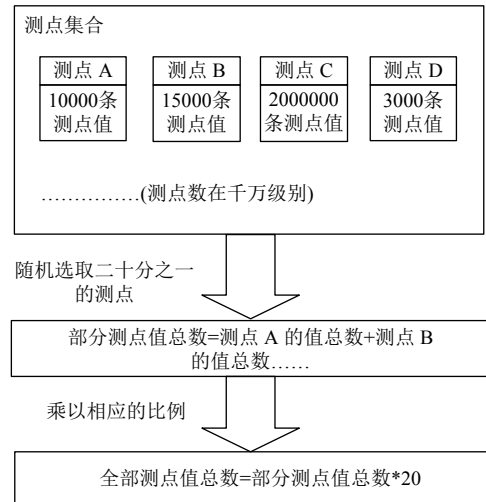


图4 外部值表大小估算方案示意图

### 3.3 SQL 语句解析方案

SQL 语句解析是查询过程中的重要一环. 海量平台 FDW 模块在 PostgreSQL 词法分析的基础上, 结合本系统的查询要求, 进行 SQL 语句的补充改造与语义解析, 方便查询时对应到具体的 UAPI 函数. SQL 语句解析的整体框架如图5所示.

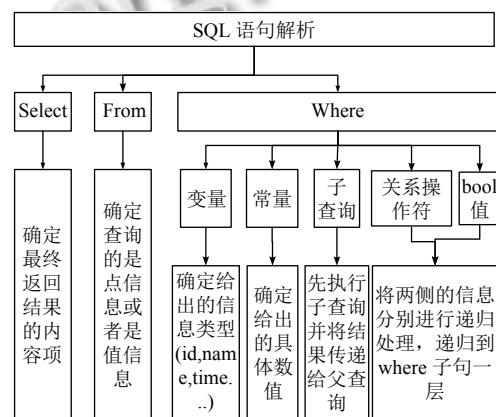


图5 SQL 语句解析框架图

SQL 语句解析主要包括以下解析过程:

根据 select 后的字段, 确定最终返回结果的内容项. 如“select name, tagtype, compress”表示需要返回测点的名称、测点类型以及是否压缩等信息.

根据 from 后的字段, 确定查询的是点信息或者是值信息. 如“from upoint”表示查询的是点信息, 而“from uvalue”表示返回的是值信息, 具体是何种值还需进一步判断.

解析 Where 后的信息, 获得 UAPI 函数调用时的参数. 按照变量、常量、子查询、关系操作符、bool 值的分类方法, 对每个分词进行处理.

变量确定给出的信息类型, 常量确定信息的具体值. 对于 id 以范围给出的 (between A and B 等形式), 拆封成单个具体数值. 涉及时间的, 转换成海量平台所用时区的时间. 如“where id between 100 and 135” id 是变量, 100 和 135 是常量, id 区间根据 UAPI 的调用规则转换成 100, 101, 102, ..., 135 的形式.

对于包含子查询的, 先执行子查询并将结果传递给父查询. 涉及关系操作符和 bool 值的, 需要将两侧的信息分别递归处理. 如“where id=100 or id=109”or 是 bool 值, 需要将两侧信息 id=100 和 id=109 分别递归到 where 子句层继续处理.

增加 mode 和 step 字段. 根据 mode 字段的值区分查询测点值的具体形式 (实时值、插值、断面和原始值). 对于查询差值的, 根据 step 字段确定每次的查询步长. 如“mode=3 and step=1500”表示查询的是差值, 且查询步长是 1500ms.

综合来看, SQL 语句“select \* from uvalue where id=4000000 or id=4000001 and time > '2016-02-25 16:47:50.10' and time < '2016-02-25 16:50:52.10' and mode=3 and step=1000;”表示查询 id 为 4000000 和 4000001 的测点在‘2016-02-25 16:47:50.10’和‘2016-02-25 16:50:52.10’时间段内的差值信息, 查询步长为 1000 ms.

创建外部表, 执行 create foreign table 语句, 为每个需要访问的远程表创建外部表. 在本系统中就是创建外部点表和外部值表.

正确执行上述步骤后, 大数据访问插件即安装完成. 在大数据端可以通过 SQL 语句访问底层的实时数据了.

#### 4 功能验证

为了验证 SQL 引擎功能的正确性, 基于海量平台进行了相关的测试.

测试中, 选用的海量平台中关系库表有 3000 万条数据<sup>[10]</sup>.

#### 4.1 根据测点直接从海量平台获取数据

海量平台存储了大量有价值的历史数据, 采用大数据手段对数据进行分析处理能够最大限度利用数据的价值. 如果采用 API 接口的方式从海量平台获取数据将会增加程序的复杂度和开发难度, 而本文的 SQL 引擎可以让大数据平台通过 SQL 的方式访问海量平台数据, 降低了开发难度.

为了验证功能, 进行了如下的实验. 实验中海量平台中实时库存储了 9000 万测点, 每个测点均有两年左右历史数据, 总数据量超过 5 TB.

图 6 为查询两个测点 1 个小时内数据, 测点数据频率为 15 分钟, 结果如图所示. 从图中可以看出 SQL 语句能够完整的获取所需的数据, 并且耗时 0.08 秒, 在功能和性能方面均有很强实用价值.

```
postgres=# select * from hsvalue where pointname in \
('GD.FS.SCADA.P00001', 'GD.FS.SCADA.P00002')\
and pointtime < '2018-05-01 01:00:01.000' and \
pointtime > '2018-04-30 23:59:59.000' and mode=0;
```

pointname	pointtime	value
GD.FS.SCADA.P00001	2018-05-01 00:00:00.000	5.123
GD.FS.SCADA.P00001	2018-05-01 00:15:00.000	5.223
GD.FS.SCADA.P00001	2018-05-01 00:30:00.000	5.156
GD.FS.SCADA.P00001	2018-05-01 00:45:00.000	5.356
GD.FS.SCADA.P00001	2018-05-01 01:00:00.000	5.056
GD.FS.SCADA.P00001	2018-05-01 00:00:00.000	6.556
GD.FS.SCADA.P00001	2018-05-01 00:15:00.000	6.234
GD.FS.SCADA.P00001	2018-05-01 00:30:00.000	6.543
GD.FS.SCADA.P00001	2018-05-01 00:45:00.000	6.098
GD.FS.SCADA.P00001	2018-05-01 01:00:00.000	6.778

(0.08s)

图 6 直接获取海量平台 1 个小时测点数据

图 7 为查询两个测点 1 个月内数据, 测点与图 6 相同, 结果如图所示. 当数据量很大时, 可通过异步轮循的方式获取实时数据.

```
postgres=# select * from hsvalue where pointname in \
('GD.FS.SCADA.P00001', 'GD.FS.SCADA.P00002')\
and pointtime < '2018-06-01 00:00:00.000' and \
pointtime > '2018-04-30 23:59:59.000' and mode=0;
```

pointname	pointtime	value
GD.FS.SCADA.P00001	2018-05-01 00:00:00.000	5.123
GD.FS.SCADA.P00001	2018-05-01 00:15:00.000	5.223
GD.FS.SCADA.P00001	2018-05-01 00:30:00.000	5.156
GD.FS.SCADA.P00001	2018-05-01 00:45:00.000	5.356
GD.FS.SCADA.P00001	2018-05-01 01:00:00.000	5.056
GD.FS.SCADA.P00001	2018-05-01 01:15:00.000	4.998
GD.FS.SCADA.P00001	2018-05-01 01:30:00.000	5.234
GD.FS.SCADA.P00001	2018-05-01 01:45:00.000	5.443
GD.FS.SCADA.P00001	2018-05-01 02:00:00.000	5.189
GD.FS.SCADA.P00001	2018-05-01 02:15:00.000	5.568

(press space to continue)

图 7 直接获取海量平台一个月测点数据

#### 4.2 根据测点直接从海量平台获取数据

通常, 为了挖掘数据价值, 需要将海量平台实时数据与关系数据库中电网档案模型数据相结合. 测试中, SQL 引擎需要从海量平台关系数据库获取电网档案数据, 然后再从海量平台中实时数据库获取量测数据.

如图 8 为从关系库获取测点信息后, 再从海量平

台实时库获取数据。图中 SQL 语句执行流程为, SQL 引擎将查询语句提交到另一个关系库获取测点信息后,将测点信息传入 SQL 引擎,通过 SQL 引擎获取海量平台中实时数据。图 8 中 SQL 语句获取的数据与图 6 一致。

```
postgres=# select * from hvalue where pointname in\
(select name from HLPT_HL_POINT_INFO where name like 'GD.FS.SCADA.P0000*') \
and pointtime < '2018-05-01 01:00:01.000' and \
pointtime > '2018-04-30 23:59:59.000' and mode=0;
```

pointname	pointtime	value
GD.FS.SCADA.P00001	2018-05-01 00:00:00.000	5.123
GD.FS.SCADA.P00001	2018-05-01 00:15:00.000	5.223
GD.FS.SCADA.P00001	2018-05-01 00:30:00.000	5.156
GD.FS.SCADA.P00001	2018-05-01 00:45:00.000	5.356
GD.FS.SCADA.P00001	2018-05-01 01:00:00.000	5.056
GD.FS.SCADA.P00002	2018-05-01 00:00:00.000	6.556
GD.FS.SCADA.P00002	2018-05-01 00:15:00.000	6.234
GD.FS.SCADA.P00002	2018-05-01 00:30:00.000	6.543
GD.FS.SCADA.P00002	2018-05-01 00:45:00.000	6.098
GD.FS.SCADA.P00002	2018-05-01 01:00:00.000	6.778

(0.23s)

图 8 从关系库中获取测点信息后再从海量平台获取 1 个月小测点数据

图 9 为获取 1 个月数据,数据与图 7 一致,也可通过轮循的方式获取 SQL 结果集数据。

```
postgres=# select * from hvalue where pointname in\
(select name from HLPT_HL_POINT_INFO where name like 'GD.FS.SCADA.P0000*') \
and pointtime < '2018-06-01 00:00:00.000' and \
pointtime > '2018-04-30 23:59:59.000' and mode=0;
```

pointname	pointtime	value
GD.FS.SCADA.P00001	2018-05-01 00:00:00.000	5.123
GD.FS.SCADA.P00001	2018-05-01 00:15:00.000	5.223
GD.FS.SCADA.P00001	2018-05-01 00:30:00.000	5.156
GD.FS.SCADA.P00001	2018-05-01 00:45:00.000	5.356
GD.FS.SCADA.P00001	2018-05-01 01:00:00.000	5.056
GD.FS.SCADA.P00001	2018-05-01 01:15:00.000	4.998
GD.FS.SCADA.P00001	2018-05-01 01:30:00.000	5.234
GD.FS.SCADA.P00001	2018-05-01 01:45:00.000	5.443
GD.FS.SCADA.P00001	2018-05-01 02:00:00.000	5.189
GD.FS.SCADA.P00001	2018-05-01 02:15:00.000	5.568

(press space to continue)

图 9 从关系库中获取测点信息后再从海量平台获取 1 个月测点数据

通过以上测试可以说明本文的 SQL 引擎在功能和性能上都满足文中提出的通过 SQL 方式获取海量平台数据并提供给大数据平台使用的要求。

## 5 结语

针对电网运用中,大数据服务平台访问海量平台中数据的问题,本文设计了基于 PostgreSQL 的访问方案。设计了大数据接口模块的整体架构,详细给出了海量平台 FDW 模块的设计思路与主要流程。采用随机抽样的方式估算外部值表的大小,结合本系统的查询需

求,对 SQL 语句进行补充改造以及语义分析。利用本文的 SQL 引擎,上层 Spark 等大数据工具可以 SQL 的方式成功访问底层海量平台的数据。目前,SQL 引擎已集成到南方电网海量准实时数据服务平台中,大数据应用可通过 SQL 引擎访问海量平台底层实时数据,极大的提高了数据资源的利用效率。

## 参考文献

- Drake JD. PostgreSQL 的发展动态. 软件和集成电路, 2018, (7): 58-59.
- 宋亚奇,周国亮,朱永利. 智能电网大数据处理技术现状与挑战. 电网技术, 2013, 37(4): 927-935.
- Zaharia M, Chowdhury M, Franklin MJ, *et al.* Spark: Cluster computing with working sets. Book of Extremes, 2010, 15(1): 1765-1773.
- 陈宇,阳国贵,吴妍,等. GKD-PL/SQL 引擎中表达式处理模块的设计与实现. 计算机工程, 2005, 31(14): 105-106, 110. [doi: 10.3969/j.issn.1000-3428.2005.14.038]
- Jiang ZM, Hassan AE. Examining the evolution of code comments in PostgreSQL. Proceedings of 2006 International Workshop on Mining Software Repositories. Shanghai, China. 2006. 179-180.
- 刘宏杰. PostgreSQL 数据库及应用. 石油工业计算机应用, 2002, (3): 17-21.
- 孙乔,焦艳斌. 实时数据库系统研发及其在智能电网中的应用. 电力信息化, 2012, 10(11): 105-109. [doi: 10.3969/j.issn.1672-4844.2012.11.023]
- Armbrust M, Xin R S, Lian C, *et al.* Spark SQL: Relational data processing in spark. Proceedings of 2015 ACM SIGMOD International Conference on Management of Data. Melbourne, Victoria, Australia. 2015. 1383-1394. [doi: 10.1145/2723372.2742797]
- Karun KA, Chitharanjan K. A review on hadoop — HDFS infrastructure extensions. Proceedings of 2013 IEEE Conference on Information & Communication Technologies. Thuckalay, Tamil Nadu, India. 2013. 132-137.
- 李绍飞,汤朝波,茅海泉. 基于 SOA 的海量准实时数据服务平台接入/访问研究. 计算机测量与控制, 2018, 26(4): 250-253.