Pivotal

Petabyte Scale Data Warehousing Greenplum

Postgres Conf 2018

PostgreSQL versus Greenplum Database

Marshall Presser
Craig Sylvester
Andreas Scherbaum
17 April 2018

PostgreSQL versus Greenplum Database

This chapter gives you an overview of a number of main differences between PostgreSQL and Greenplum Databases

OLTP versus OLAP

- PostgreSQL is for OLTP
  - Online Transaction Processing
- Greenplum Database is for OLAP
  - Online Analytical Processing

OLTP versus OLAP

- PostgreSQL:
  - Have your hot data in memory
  - Use any means to rapidly identify that one dataset you want
- Greenplum Database:
  - Scan all your data, and again
  - Calculate results over all of your data
  - No way this fits into memory - more hardware required

Scalability

- PostgreSQL:
  - Scales well on a single machine
  - Using multiple machines involves primary + secondary setups, failover, replication ect
  - Again: good for OLTP on a single node
- Greenplum Database:
  - Purpose built to scale across many machines
  - Comes with replication built-in
  - Slower on OLTP workloads

Query Planner

- PostgreSQL:
  - Grown over time
  - Complex code, hard to debug
  - Not many people understand it
- Greenplum Database:
  - Comes with GPORCA
  - Pluggable query planner, written from scratch, in C++
  - “Powers” both SQL and NoSQL workloads:
    - SQL: Greenplum Database
    - NoSQL: SQL-on-Hadoop for Apache HAWQ
  - Comes with it’s own test suite
  - Debugging tool allows to extract query information from problematic queries

Partitioning

- PostgreSQL:
  - Included only in last version
  - Partition pruning only during planning time:
    - Partitions are always scanned when they can’t be included by static values
- Greenplum Database:
  - Has partitioning for a decade now
  - Allows partitions and subpartitions (two levels)
  - Partitioning is on top of sharding
  - GPDB planner can do partition pruning during runtime

Resource Queues

- PostgreSQL:
  - No resource management
- Greenplum Database:
  - Resources like CPU, Queries and Memory can be managed on a per-user bases
  - Queries can be limited, or aborted, or queued

Pipelining

- PostgreSQL:
  - Has Background Workers now
  - Still no complete parallel execution
- Greenplum Database:
  - Pipelines flow data from one processing part to the next
  - Makes efficient use of CPU resources, as more cores can be used in parallel

Parallel loading and unloading

- PostgreSQL:
  - gpLoader speeds up data loading
  - But is still one process
- Greenplum Database:
  - Can load data using external tables, in parallel on all segments
  - Scales out

Access to external resources

- PostgreSQL:
  - Uses SQL/MED
  - Requires additional plugins to read/write different formats, or connect resources
- Greenplum Database:
  - Uses External Tables to access resources
  - Scales out on all segments
  - Can load data in parallel
  - Can execute processes in parallel
    - gp_toolkit is using scripts/tools on segments to provide cluster information
  - Data can be imported/exported in many formats, including transformations

Hadoop

- PostgreSQL:
  - ???
- Greenplum Database:
  - External Tables and the PXF Connector allow access to Hadoop systems

Table compression

- PostgreSQL:
  - ???
- Greenplum Database:
  - Tables (rows or columns) can be compressed using different algorithms
    - zlib
    - quicklz
    - RLE

Column orientation

- PostgreSQL:
  - ???
- Greenplum Database:
  - Tables can be row-oriented or column-oriented
    - Row-oriented: traditional HEAP table
    - Column-oriented: every column is stored in one file
      - Access to a few columns do not read the entire row
      - Every column can be compressed, even using different algorithms

Text Search

- PostgreSQL:
  - Full Text Search included in the product
  - Kind of complicated for different languages
- Greenplum Database:
  - Integrates Apache Solr
    - Runs in parallel on the segments, no single instance

Expansion

- PostgreSQL:
  - Expansion beyond one system - includes replication, and connection pooler and such
  - Does not improve the overall performance
- Greenplum Database:
  - Just add more segments, and re-distribute the database

MADlib - Machine Learning at scale

- PostgreSQL:
  - Works with PostgreSQL
- Greenplum Database:
  - Works with Greenplum Database
  - Originally designed for Greenplum Database
  - Scaling out will improve the MADlib performance

Foreign Keys

- PostgreSQL:
  - Work as expected
- Greenplum Database:
  - FK are not enforced
  - Because of the distributed nature, FK enforcement requires a scan across the entire DB, for every single row

Transforming How The World Builds Software