
Pivotal
A NEW PLATFORM FOR A NEW ERA

Pivotal Query Optimizer (PQO)

Pivotal® **Greenplum Database**

Agenda

- Introduction
- Why PQO?
- Features / Enhancements
- Fall Back to Planner
- References, Review
- To the Labs!

Pivotal

Why PQO?

- We need to efficiently query very large data sets
- Our preferred BI tools generate very complex SQL
- TPC-H, query 21: PQO evaluates *1.2 billion* plans in 250 ms
- Modular: integrated with GPDB and HDB / HAWQ

Do What Works

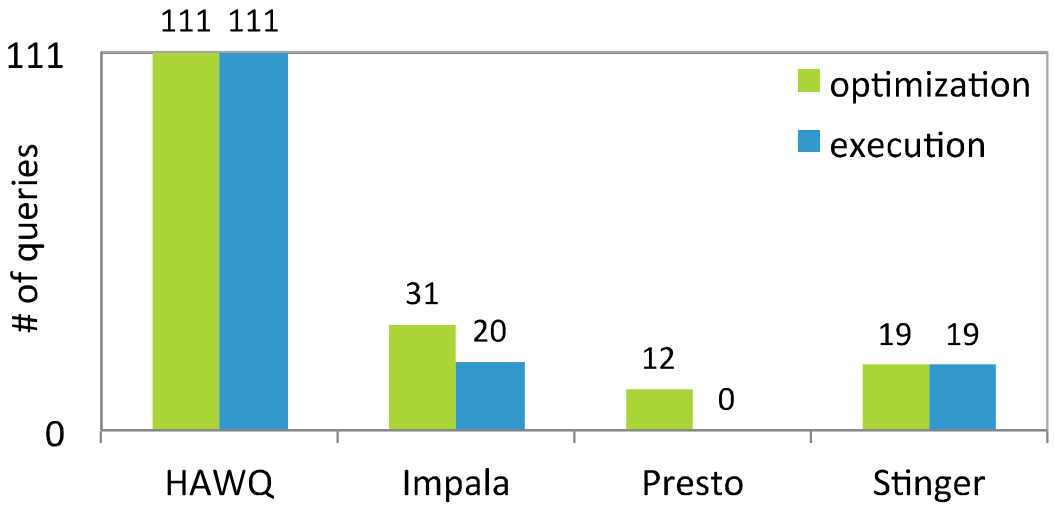


Figure 15: TPC-DS query support

Ref.: *Orca: A Modular Query Optimizer Architecture for Big Data*
http://pivotal.io/big-data/white-paper/orca-a-modular-query-optimizer-architecture-for-big-data)

PQO's New Features

- PQO includes enhancements for these **specific** types of queries and operations
  - Queries against partitioned tables
  - Queries that contain sub-queries
  - Queries that contain a common table expression (CTE)
- PQO contains these optimization enhancements
  - Improved join ordering
  - Join-Aggregate reordering
  - Sort order optimization
  - Data skew estimates included in query optimization

Pivotal

A Bit More on Features

- Queries on partitioned table
  - Dynamic Partition Elimination
  - No more OOM during query planning
- Correlated Subqueries (CSQ)
- Join on columns having differing types no longer uses Nested Loop Join
- Queries with Common Table Expressions (CTE)
- Distribution and partition keys are now updatable

Pivotal

Using Pivotal Query Optimizer GUCs

- *optimizer_control*
  - System level control to enable / disable setting OPTIMIZER
- *optimizer*
  - System level:

    ```
    $ gpconfig -c optimizer -v on --masteronly
    ```

  - Specific database level:

    ```
    ALTER DATABASE <db_name> SET OPTIMIZER = ON ;
    ```

  - Session and Query level:

    ```
    SET OPTIMIZER = ON ;
    ```

- **All** legacy optimizer (Planner) GUCs are ignored by PQO

Pivotal

ANALYZEDB or ANALYZE ROOTPARTITION

- Before using PQO at the system, database, session or query level ensure root partition statistics are available
- PQO **must** have root partition statistics
- The legacy query optimizer does not use these statistics
- SET optimizer_analyze_root_partition = ON ;
- The new ANALYZEDB is a parallel, incremental analyze that *automatically analyzes the root partition*
- ANALYZEDB is **highly** recommended

Pivotal

PQO Performance Considerations

- Short running queries
  - PQO performs extensive optimizations to determine the optimal query execution plan and for short running queries this additional optimization overhead may be encountered
- ANALYZE
  - PQO requires root partition statistics
  - The legacy optimizer does not use these statistics
- DML regression is due to the increased number of responsibilities for DML (e.g. the support of updates on partition and distribution keys)

Pivotal

How to Use PQO for Existing Installations

- For existing production installations, **do not** set at the system or database level
  - There may be slight changes in the existing workload or expected behavior
- Set PQO at the **session** or **query** level for to improve query performance for under performing queries
  - Ensure the tables do not contain multi-column partition keys
  - Ensure the query does not run against master only tables such as the system table pg_attribute
- Also consider PQO performance enhancements for
  - Queries against large partitioned tables
  - Queries that contain sub-queries
  - Queries that contain a common table expression (CTE)
  - DML operations

Pivotal

How to Use PQO for New Installations

- For new installations, not in production, set PQO at the system or database level, based on the following:
  - If the data model does not contain multi-column partition keys
  - If a partitioned table contains more than 20,000 partitions, redesign the schema before production
- If the PQO can not generate a query execution plan it will automatically fallback to the legacy optimizer

Pivotal

Partitioning Pain Points in the Planner

- Query plans can get very large
  - *Append* node listing all partitions to be scanned
  - OOM when number of partitions is large
  - Updates with joins on partitioned tables can generate prohibitively large query plans
- Dynamic partition elimination
  - Partition-selecting side of join gets executed **TWICE**
  - Plan size still dependent upon the number of partitions

Partitioned Tables in PQO

- Addresses plan size
  - Plan size **independent** of number of partitions
  - Partitions are not enumerated in plans
  - No OOM because of large number of partitions
- Optimal partition elimination
  - If there is a cost-effective way to eliminate partitions, PQO finds it!
  - ***Partition-selecting side of join is not executed twice***

Subqueries: Definition

- A query that is nested inside an outer query block
- Correlated Subquery (CSQ) is a subquery that *uses values from the outer query*

```
SELECT * FROM part p1
WHERE price >
(SELECT avg(price) FROM part p2  WHERE  p2.brand = p1.brand)
```

Pivotal

Subqueries: Impact

- Heavily used in many workloads
  - BI/Reporting tools generate substantial number of subqueries
  - TPC-H workload: 40% of the 22 queries
  - TPC-DS workload: 20% of the 111 queries
- Inefficient plans lead to long running queries
- Optimizations
  - De-correlation
  - Conversion of subqueries to joins

Pivotal

Common Table Expressions (CTEs)

- Define in a WITH Clause
  - 46 queries in TPC-DS
- Planner always expand CTEs by default
  - Possibility of deadlock during execution
- PQO guarantees deadlock free execution plans!

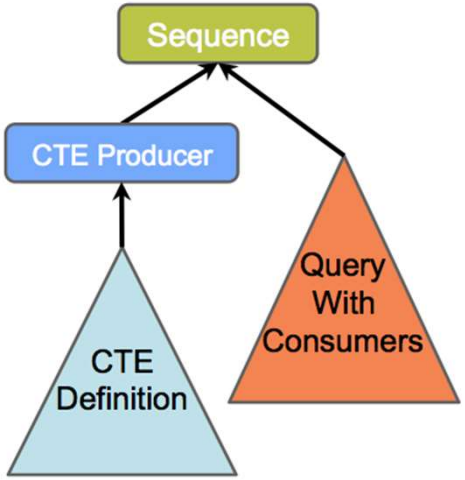
Pivotal

CTE Query Plan in PQO

```
WITH v AS (SELECT a, sum(b) as s FROM T where c < 10 GROUP BY a)
SELECT *
FROM v AS v1, v AS v2
WHERE v1.a <> v2.a
AND v1.s < v2.s;
```

CTE definition (producer)

CTE references (consumers)




```
                                    QUERY PLAN
-------------------------------------------------------------------------------
 Sequence  (cost=0.00..5.32 rows=1 width=32)
   ->  Shared Scan (share slice:id 0:0)  (cost=0.00..3.04 rows=1 width=1)
         ->  Materialize  (cost=0.00..3.04 rows=1 width=1)
               ->  Gather Motion 2:1  (slice1; segments: 2)
                     ->  Table Scan on t  (cost=0.00..1.03 rows=1 width=16)
                           Filter: c < 10
   ->  Hash Join  (cost=0.00..1.22 rows=1 width=32)
         Hash Cond: "outer".b = "inner".a
         ->  Shared Scan (share slice:id 0:0)  (cost=... rows=1 width=16)
         ->  Hash  (cost=0.02..0.02 rows=1 width=16)
               ->  Shared Scan (share slice:id 0:0)  (cost=0.00..0.02 rows=1 ...)
```

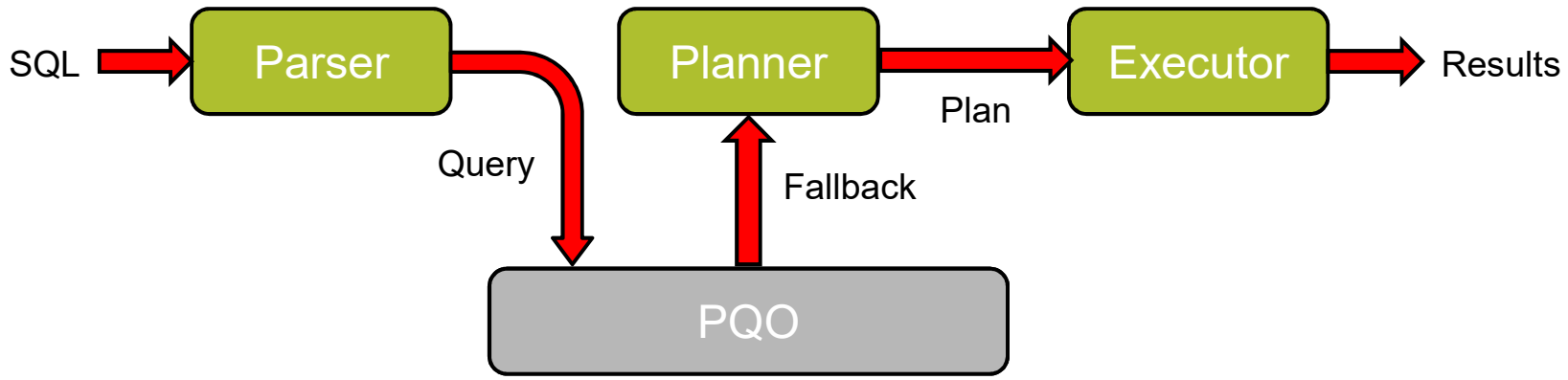
Pivotal

Features Not Supported by PQO

- PERCENTILE window function
- External parameters
- SortMergeJoin (SMJ)
- CUBE operator
- Multiple grouping sets
- Utility and DDL commands (e.g. Alter Table)
- Catalog queries

Pivotal

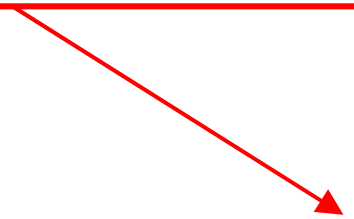
When PQO falls back



PQO will automatically fall back to the legacy optimizer for unsupported features

How to determine if it is PQO or Planner?

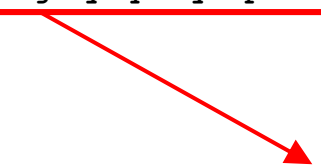
```
bootcamp=# explain select * from products;
                                    QUERY PLAN
-------------------------------------------------------------------------------
 Gather Motion 2:1  (slice1; segments: 2)  (cost=0.00..431.00 rows=1 width=11)
   ->  Table Scan on products  (cost=0.00..431.00 rows=1 width=11)
 Settings:  optimizer=on
 Optimizer status: PQO version 1.595
(4 rows)
```



PQO: Pivotal Query Optimizer
+
Optimizer Version #

How to determine if it is PQO or Planner?

```
bootcamp=# explain select * from products;
                                    QUERY PLAN
-------------------------------------------------------------------------------
 Gather Motion 2:1  (slice1; segments: 2)  (cost=0.00..1.01 rows=1 width=11)
   ->  Seq Scan on products  (cost=0.00..1.01 rows=1 width=11)
 Optimizer status: legacy query optimizer
(3 rows)
```



Planner

For unsupported PQO features

Additional log message entries (in pg_log)

```
2015-08-14 17:00:07:401231 PDT,THD000,NOTICE,"Feature not supported by the Pivotal Query
Optimizer: Rollup and cube"
```

References

- PQO Features
  - http://gpdb.docs.pivotal.io/4360/admin_guide/query/topics/query-piv-opt-features.html
- PQO Limitations
  - http://gpdb.docs.pivotal.io/4360/admin_guide/query/topics/query-piv-opt-limitations.html
- Enabling PQO
- http://gpdb.docs.pivotal.io/4360/admin_guide/query/topics/query-piv-opt-enable.html
- Blog
  - https://blog.pivotal.io/big-data-pivotal/products/greenplum-database-adds-the-pivotal-query-optimizer
- SIGMOD paper
  - http://pivotal.io/big-data/white-paper/orca-a-modular-query-optimizer-architecture-for-big-data
- GitHub Repository
  - https://github.com/greenplum-db/gporca

Pivotal

Wrap Up

- Why PQO?
- How to enable / disable PQO
- PQO requires statistics on root partition
- Do the Lab
- Thank you!

Pivotal


Pivotal
A NEW PLATFORM FOR A NEW ERA