Postgres Overview

New tools in Postgres:

- pg_class and pg_stats tables: They store the statistics that the DBMS uses for estimating the cost of query plans.

- EXPLAIN command: To see the execution plan of a statement

- ENABLE command: Can be used to enable or disable specific algorithms (e.g., hash join, nested loop join, merge join etc.).

Introduction to PostgreSQL

PostgreSQL is the most advanced open source relational database

Documentation is excellent and is at www.postgresql.org/docs/8.4/...

PostgreSQL Book: http://momjian.us/main/writings/pgsql/aw_pgsql_book/

Examine System Catalog

Following three tables contain statistics information:

- pg_class
  - Total number of entries in each table and index
  - Number of disk blocks occupied by each table or index
- pg_statistics
  - It is used by the planner to estimate the selectivity (i.e., expected number of matching rows) of WHERE clause
- pg_stats
  - Shows the information of pg_statistics restricted to specific user. It only shows information about tables that the user can access.

Further documentation:
http://www.postgresql.org/docs/8.4/static/planner-stats.html

Examples of System Catalog Queries

```
SELECT relname, relkind, reltuples, relpages
FROM pg_class
WHERE relname LIKE 'tenk1%';

       relname        | relkind | reltuples | relpages
----------------------+---------+-----------+----------
 tenk1                | r       |     10000 |      358
 tenk1_hundred        | i       |     10000 |       30
 tenk1_thous_tenthous | i       |     10000 |       30
 tenk1_unique1        | i       |     10000 |       30
 tenk1_unique2        | i       |     10000 |       30
(5 rows)
```

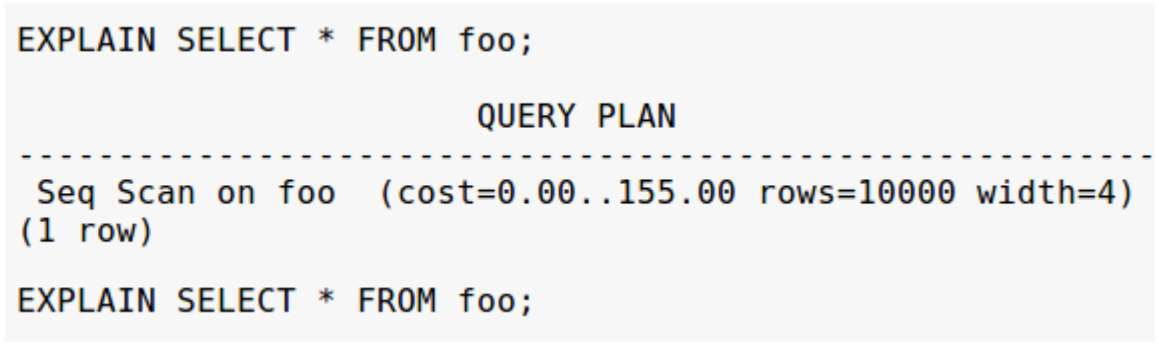
To get up-to-date statistics information use: VACUUM and ANALYSE

```
SELECT attname, n_distinct, most_common_vals
FROM pg_stats
WHERE tablename = 'road';

 attname | n_distinct |
---------+------------+-----------------------------------------------------
 name    |  -0.467008 | {"I- 580                        Ramp","I- 880
 thepath |         20 | {"[(-122.089,37.71),(-122.0886,37.711)]"}
(2 rows)
```

EXPLAIN Command

- It shows the execution plan of a statement.
- It can be used for any kind of statement, such as SELECT, INSERT, DELETE, VALUES, EXECUTE, or DECLARE
- Syntax: EXPLAIN [ANALYZE] [VERBOSE] statement
- Example: A simple query on a table with single integer column and 10,000 rows



```
EXPLAIN SELECT * FROM foo;

                       QUERY PLAN
---------------------------------------------------------
 Seq Scan on foo  (cost=0.00..155.00 rows=10000 width=4)
(1 row)

EXPLAIN SELECT * FROM foo;
```

More Complex EXPLAIN Command

Planning for an aggregate query
The table 'foo' has an index 'fi' on attribute 'i‘

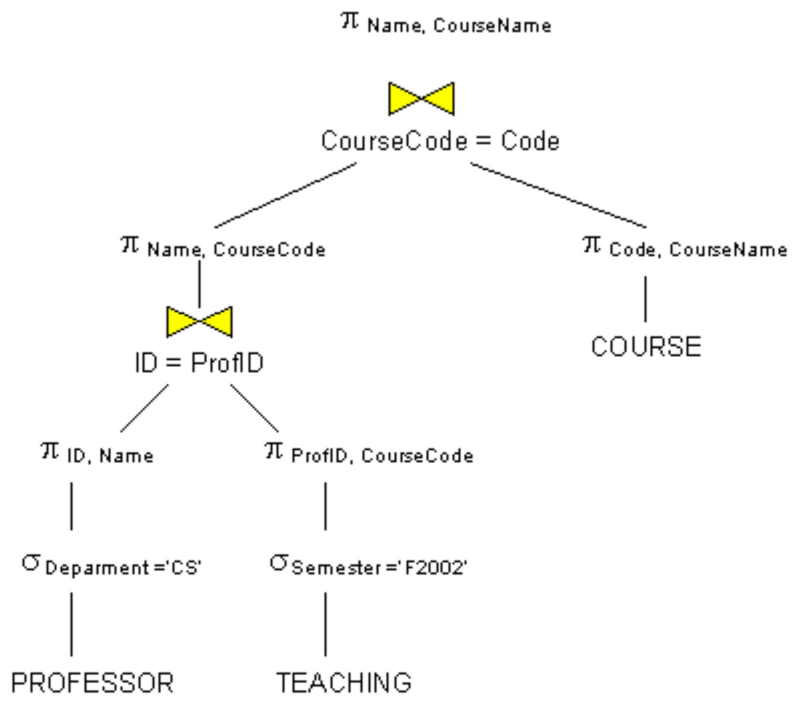
```
EXPLAIN SELECT sum(i) FROM foo WHERE i < 10;

                             QUERY PLAN
---------------------------------------------------------------------
 Aggregate  (cost=23.93..23.93 rows=1 width=4)
   ->  Index Scan using fi on foo  (cost=0.00..23.92 rows=6 width=4)
         Index Cond: (i < 10)
(3 rows)
```

For more details on how to interpret the output of EXPLAIN command: http://www.postgresql.org/docs/8.4/static/using-explain.html

Relational Algebra Notation

Convert the output from EXPLAIN command into equivalent relational-algebra tree notation.



Creating Indexes

Creating Simple B+ Tree Index: Index on column `year` of table `films`:

```
CREATE UNIQUE INDEX year_idx ON films (year)
```

Creating clustered index:

- For accessing single rows randomly within a table, the actual order of data within the table does not matter.
- However, clustering is beneficial if we need to access:
  - Some data more frequently as compared to others, or
  - Access a range of indexed value from a table. E.g. Accessing movies released between 2011-2013

```
CLUSTER films USING year_idx;
```

Planner Method Configuration

- Postgres allows a crude method of influencing the query plans chosen by the query optimizer.
- We can force the optimizer to select some other plan by disabling certain query plans.
- Some of the plans cannot be entirely turned off, but can be discouraged if other options are available.
  - enable_seqscan
  - enable_sort
- Following plans can be enabled or fully disabled
  - SET enable_bitmapscan=off;
  - SET enable_bitmapscan=on;
  - SET enable_hashjoin=off;
  - SET enable_mergejoin=off;