Statistics in PostgreSQL

Heikki Linnakangas

Why statistics?

- For query planning!

Statistics collected

EnterpriseDB®
The Enterprise Postgres Company

For each table:

- # of tuples
- # of pages

For each column:

- Most Common Values
- Histogram
- # of distinct values
- Fraction of NULLs
- Avg row width
- Correlation between physical and logical order

Most Common Values

EnterpriseDB®
The Enterprise Postgres Company

```
SELECT most_common_vals, most_common_freqs
FROM pg_stats
WHERE tablename='city'
AND attname='countrycode';

most_common_vals  | {IND,CHN,USA,BRA,JPN}
most_common_freqs |
{0.0953333,0.084,0.072,0.0706667,0.0613333
}
```

MCVs in action

EnterpriseDB®
The Enterprise Postgres Company

```
EXPLAIN ANALYZE
SELECT * FROM city WHERE countrycode='IND';

                          QUERY PLAN
---------------------------------------------------------------
 Seq Scan on city  (cost=0.00..82.99 rows=389 width=31)
(actual time=1.060..4.420 rows=341 loops=1)
   Filter: (countrycode = 'IND'::bpchar)
 Total runtime: 4.533 ms


SELECT reltuples * 0.0953333 FROM pg_class WHERE
relname='city';
  ?column?
-------------
 388.8645307
```

Histogram

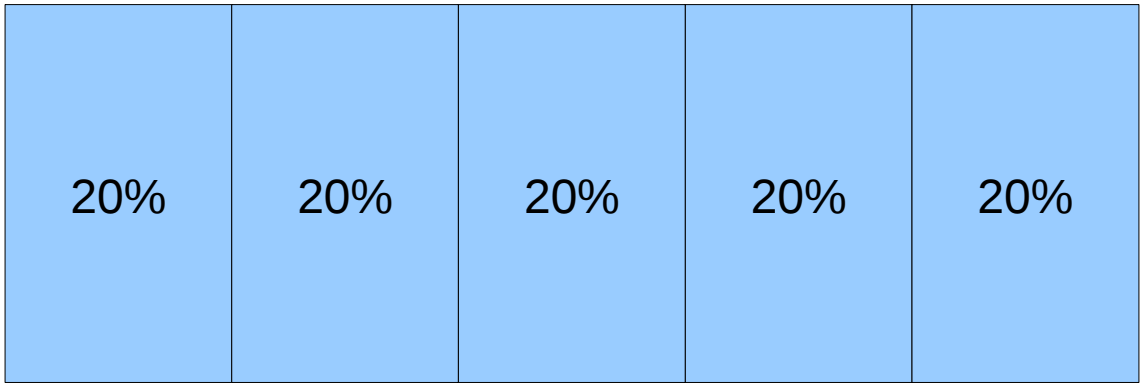
EnterpriseDB®
The Enterprise Postgres Company

```
SELECT histogram_bounds FROM pg_stats
WHERE tablename='city'
AND attname='population';
              histogram_bounds
-----------------------------------------
 {42,107800,137893,200453,380755,10500000}
(1 row)
```

Histogram visualized

EnterpriseDB®
The Enterprise Postgres Company

```
{42,107800,137893,200453,380755,10500000}
```



Doesn't include Most Common Values!

Histogram in action

EnterpriseDB®
The Enterprise Postgres Company

```
SELECT COUNT(*),MIN(population),
MAX(population) FROM city;
 count | min |    max
-------+-----+----------
  4079 |  42 | 10500000

SELECT COUNT(*) FROM city
WHERE population BETWEEN 107800 AND 137893;
 count
-------
   783
```

of distinct values

EnterpriseDB®
The Enterprise Postgres Company

```
SELECT n_distinct FROM pg_stats WHERE tablename='city' AND
attname='countrycode';
 n_distinct
------------
        232


SELECT COUNT(DISTINCT countrycode) FROM city;
 count
-------
   232
```

- Absolute value, not a fraction
- Difficult to calculate from sample, estimate can be inaccurate
- Can be set manually:
- ALTER TABLE ... ALTER COLUMN ... SET (n_distinct = ...)

Statistics collected

EnterpriseDB®
The Enterprise Postgres Company

For each table:

- # of tuples
- # of pages

For each column:

- Most Common Values
- Histogram
- # of distinct values

- Fraction of NULLs
- Avg row width
- Correlation between physical and logical order

Clustering

EnterpriseDB®
The Enterprise Postgres Company

- Clustering a table means sorting it physically.
  - CLUSTER city USING i_pop;
- Speeds up range queries.

```
SELECT * FROM city ORDER BY population LIMIT 10;
                    QUERY PLAN
-----------------------------------------------------
 Limit  (cost=0.00..0.37 rows=10 width=31)
   ->  Index Scan using i_pop on city
(cost=0.00..152.44 rows=4079 width=31)
```

Correlation measures clustering

EnterpriseDB®
The Enterprise Postgres Company

```
SELECT attname, correlation FROM pg_stats
WHERE tablename='city' AND
attname IN('population', 'countrycode');

   attname    | correlation
--------------+-------------
 countrycode | 0.000834526
 population  |           1
```

Exotic datatypes

EnterpriseDB®
The Enterprise Postgres Company

Full text search (tsvector)

- MCV only

PostGIS datatypes

- no statistics

Functions

- `CREATE FUNCTION myfunc() RETURNS SETOF int4 AS $$ VALUES (1), (2); $$ ROWS 2 LANGUAGE sql;`

ANALYZE

EnterpriseDB®
The Enterprise Postgres Company

ANALYZE [VERBOSE] [table [(column [, ...])]]
(or ANALYSE)

- Updates statistics for a table.
  - or just some columns
  - or all tables in the database
- Random sample, doesn't scan the whole table
  - # of distinct rows can be very inaccurate

Statistics target

- Specifies the # of Most Common Values and Histogram bins
- All the examples in this presentation use target of 5
- Default is 100 for versions >= 8.4
  - 10 for older versions
- Can be changed
  - System-wide in postgresql.conf
    - `default_statistics_target=1000`
  - Per session
    - `SET default_statistics_target = 1000`
  - Per column
    - `ALTER TABLE ... ALTER COLUMN ... SET STATISTICS 1000`
- Higher values can increase planning time

Statistics and inheritance

- For inherited (partitioned) tables, statistics are collected for:
  - the parent,
  - each child,
  - the partitioned table as whole (since 9.0)

- Note: If the parent table is not modified, autovacuum will only update the per-child statistics.

Statistics drift

- MCV and histogram store **fractions**, not absolute counts!
- Statistics stay accurate even if the table grows, assuming the new data has same distribution.
  - Timestamp column is a common pitfall

Auto-analyze

- Autovacuum runs ANALYZE automatically as a table grows.
- Controlled by
  - autovacuum_analyze_threshold
  - autovacuum_analyze_scale_factor
  - Or per-table

Cross-column Correlation

EnterpriseDB®
The Enterprise Postgres Company

```
explain SELECT * FROM city
WHERE district = 'Baden-Württemberg';
                        QUERY PLAN
-----------------------------------------------------
 Seq Scan on city  (cost=0.00..82.99 rows=10 width=31)
   Filter: (district = 'Baden-Württemberg'::text)

explain SELECT * FROM city
WHERE district = 'Baden-Württemberg' AND countrycode='DEU';
                                QUERY PLAN

-----------------------------------------------------------------
---
 Seq Scan on city  (cost=0.00..93.19 rows=1 width=31)
   Filter: ((district = 'Baden-Württemberg'::text) AND (countrycode
= 'DEU'))
```

Thank you

EnterpriseDB®
The Enterprise Postgres Company

- If your query is slow, always check the plan with

  EXPLAIN ANALYZE

Thank you!

Remember to leave feedback:
http://2010.pgday.eu/feedback