



# pg\_index\_stats

manage extended statistics automatically

Lepikhov A., Rybakina A.

Postgres Professional

2024

# Self Introduction

- Core Developer in Postgres Professional since 2017.
- Contributing to the PostgreSQL project since 2017:  
*Self-Join Elimination, GROUP-BY optimisation, OR <-> ANY transformation.*
- Ph.D. in Computer Sciences (Distributed Databases), MSU, 2008.
- Designed extensions: AQO, sr\_plan, pg\_index\_stats ...



# Quick start

```
CREATE EXTENSION 'pg_index_stats';
CREATE TABLE test (x int, y int, z text);
CREATE INDEX ON test (x,y);
CREATE INDEX ON test (z,y);
\dX
```

List of extended statistics						
Schema	Name	Definition	Ndistinct	Dependencies	MCV	
public	test_x_y_stat	x, y <b>FROM</b> test	defined	defined	defined	
public	test_z_y_stat	y, z <b>FROM</b> test	defined	defined	defined	



## Quick start - II

```
DROP INDEX test_x_y_idx;
```

List of extended statistics					
Schema	Name	Definition	Ndistinct	Dependencies	MCV
public	test_z_y_stat	y, z <b>FROM</b> test	defined	defined	defined

# GUCs & Funcs

- *mode* - disabled | all | univariate | multivariate
- *columns\_limit*
- *pg\_index\_stats\_build(relname, mode)*
- *pg\_index\_stats\_rebuild()*
- *pg\_index\_stats\_remove()*

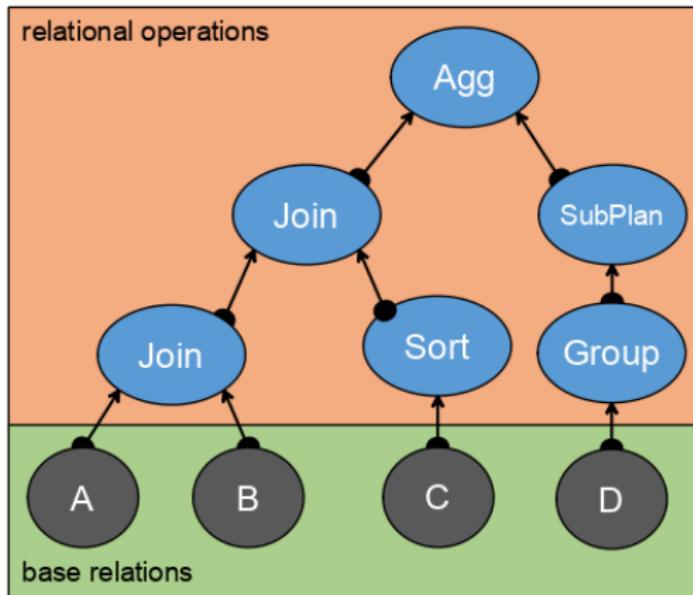
Two-step process:

- Object Access Hook - gather candidate OIDs
- Utility Hook - create extended statistics after a successful utility statement

Set dependency of auto-generated statistics on the extension and index relation.

Add specific description for auto-generated statistics

## What is the reason?



## Cardinality estimation:

- Adaptive Query Optimizer  
(*aqo*)
  - Query replanning  
(*replan*)
  - Plan freezing  
(*sr\_plan*)
  - Selectivity by index  
(*JoinSel*)

## Cardinality estimation:

- ## ➤ statistics ???

# Query auto-generation frameworks



# Redundant expressions

Query	Planned rows	Actual rows
<pre>SELECT * FROM power_plants WHERE     country = 'RUS';</pre>	544	544
<pre>SELECT * FROM power_plants WHERE     country = 'RUS' AND     country_long = 'Russia';</pre>	8	544



\*Global Power Plant Database  
Copyright 2018-2021 World Resources Institute and Data Contributors



# The question

*Why can database systems, which manage all the data, not analyse it and find at least in-table dependencies and interconnections?*



# Why indexes?

Table "Parcels":

Indexes:

- "parcel\_pkey" PRIMARY KEY, btree (id)
- "parcel\_parcel\_id" btree (parcel\_id)
- "parcel\_id\_prik" btree (parcel\_id, prik)
- "parcel\_id\_sn\_pol" btree (parcel\_id, sn\_pol)
- "parcel\_par\_begin" btree (parcel\_id, prik\_begin\_period)
- "parcel\_par\_patient" btree (parcel\_id, patient\_id)
- "parcel\_par\_recid" btree (parcel\_id, recid)
- "parcel\_per\_prik" btree (period, prik)

# Extended v/s Plain Statistics

SELECT \* FROM power\_plants WHERE ...

Query	Plain stat	Extended stat	Actual rows
country = 'RUS';	544	545	544
<b>AND</b> primary_fuel IN ('Solar')	166	57	57
primary_fuel IN ('Solar', 'Biomass')	189	60	60
primary_fuel IN ('Solar', 'Biomass', 'Coal')	225	156	156
<b>AND</b> source = 'Wiki—Solar'	24	46	40
<b>AND</b> longitude > 40.	1	1	33
<b>AND</b> longitude < 70.			



\*Global Power Plant Database

Copyright 2018-2021 World Resources Institute and Data Contributors



# Does multicolumn histogram make sense?



# Multicolumn histogram advantage

SELECT \* FROM power\_plants WHERE ...

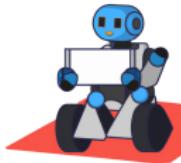
Query	Plain stat	Extended stat	Histogram	Actual rows
country = 'RUS';	544	545	544	544
<b>AND primary_fuel IN ('Solar')</b>	166	57	57	57
<b>primary_fuel IN ('Solar', 'Biomass')</b>	189	60	60	60
<b>primary_fuel IN ('Solar', 'Biomass', 'Coal')</b>	225	156	156	156
<b>AND source = 'Wiki-Solar'</b>	24	46	42	40
<b>AND longitude &gt; 40.</b>	1	1	35	33
<b>AND longitude &lt; 70.</b>				



\*Global Power Plant Database

Copyright 2018-2021 World Resources Institute and Data Contributors





# Questions ?



The pg\_index\_stats extension  
github link



Postgres Professional LLC  
The Russian Postgres Company