# cbo_stat_dump : To reproduce complex query planner issues

Gaurav Kukreja, Software Engineer

yugabyteDB

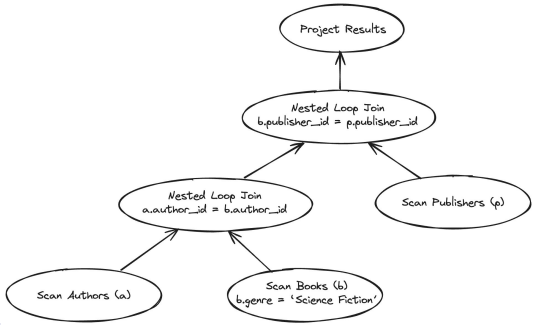
“You can’t improve, what you can’t measure”

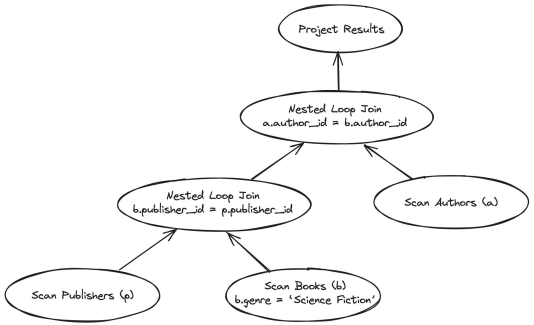
- `cbo_stat_dump`
  - https://github.com/yugabyte/cbo_stat_dump
- Testing the Accuracy of Query Optimizers (TAQO)
  - https://github.com/yugabyte/taqo

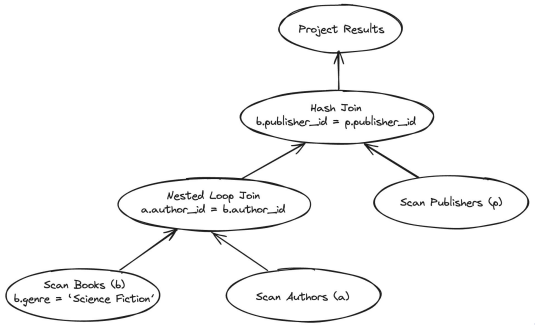
# What is CBO aka. query planner?

- SQL is a declarative language.
  - Query defines how the result should look, not how it is computed.
- CBO must choose an optimal execution plan for the query.
  - Which index to use? Which join methods? Which join order?

```
SELECT b.book_title, a.author_name, p.publisher_name
FROM authors a
INNER JOIN books b ON a.author_id = b.author_id
INNER JOIN publishers p ON b.publisher_id = p.publisher_id
WHERE b.genre = 'Science Fiction';
```



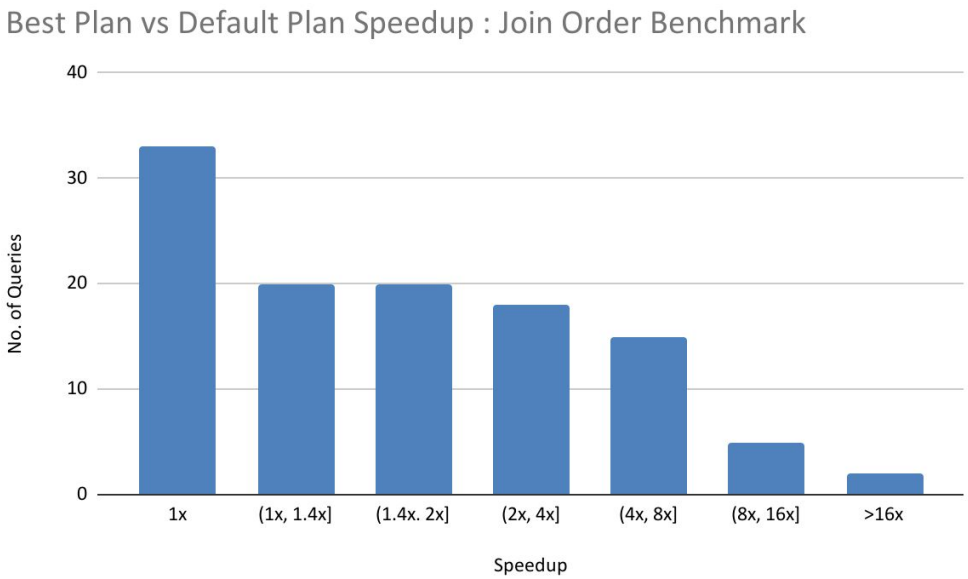



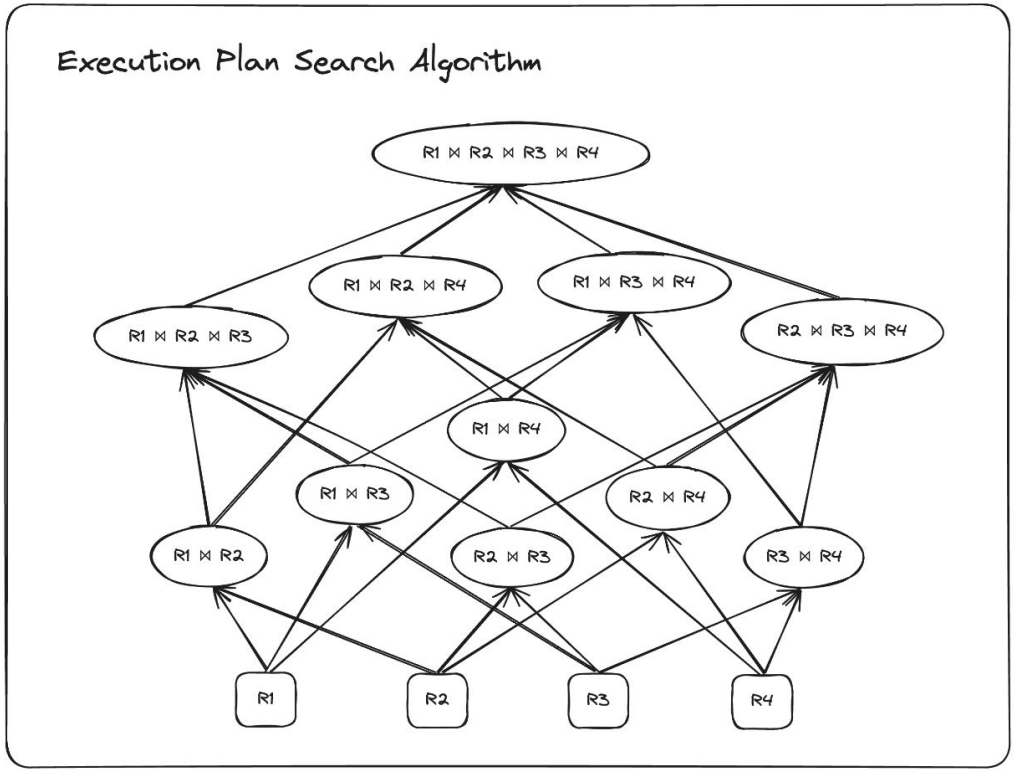
# How does PG Query Planner perform?

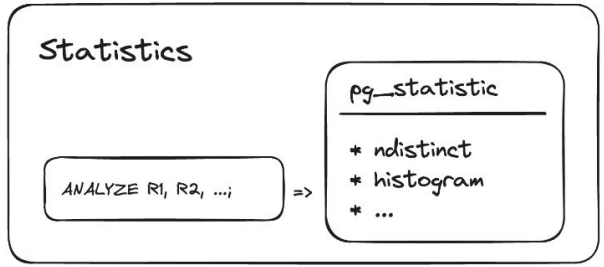
- TAQO Framework to test efficacy of query planners
- Tested with PostgreSQL 15
- Join Order Benchmark

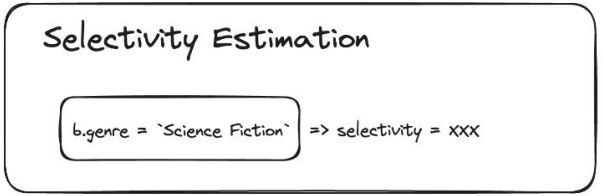
- Number of queries = 113
- Best plans picked for 33 queries or 29.2% of queries
- Geometric mean of speedup with best plans = 1.95x

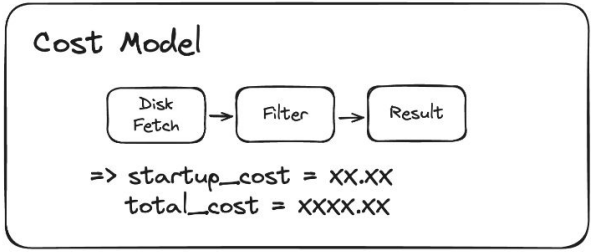


# How does CBO work?

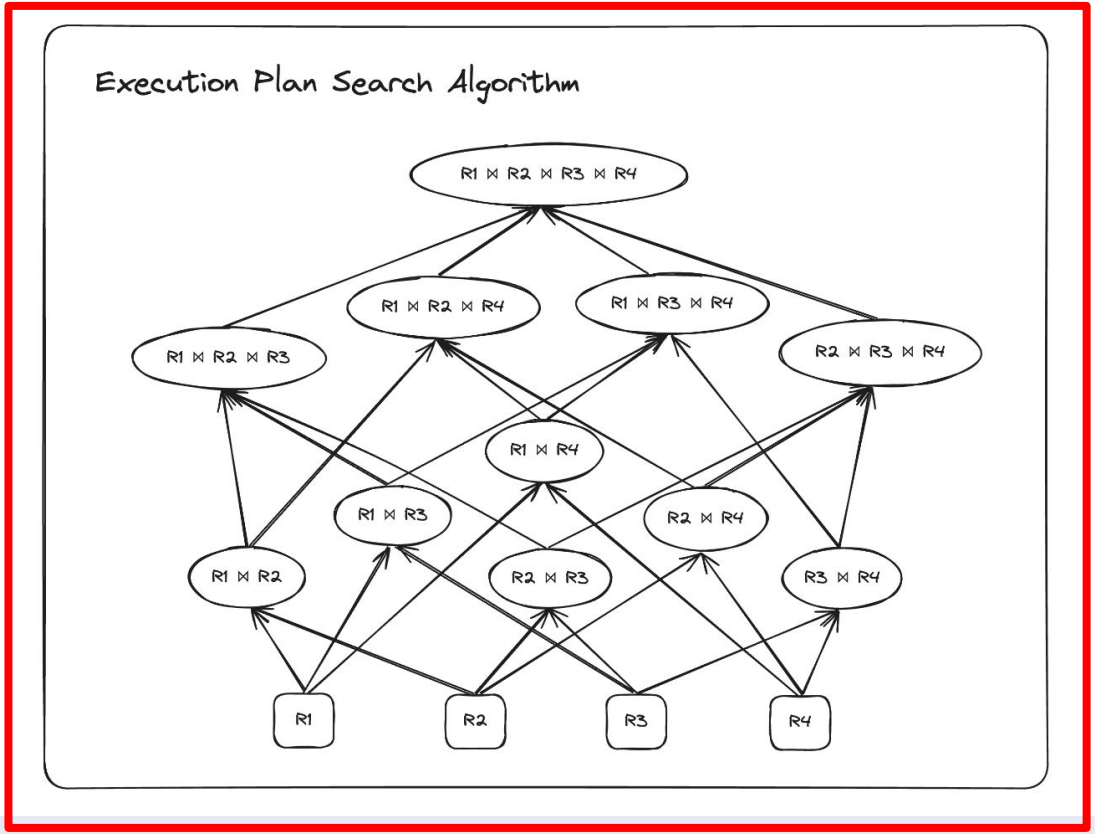


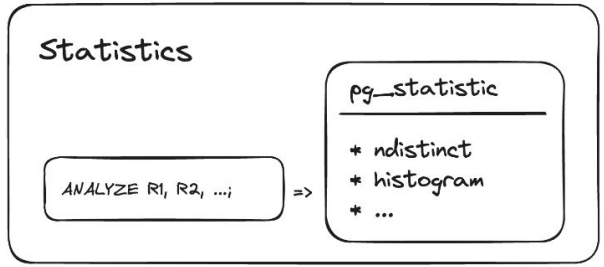



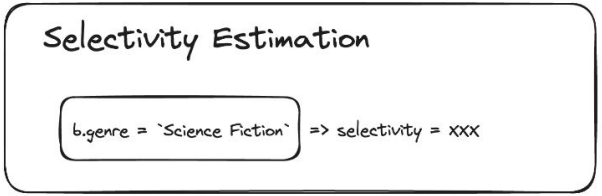



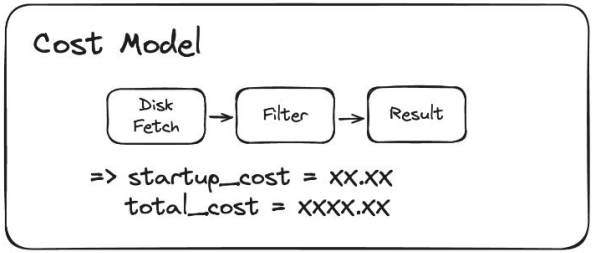



# How does CBO work?

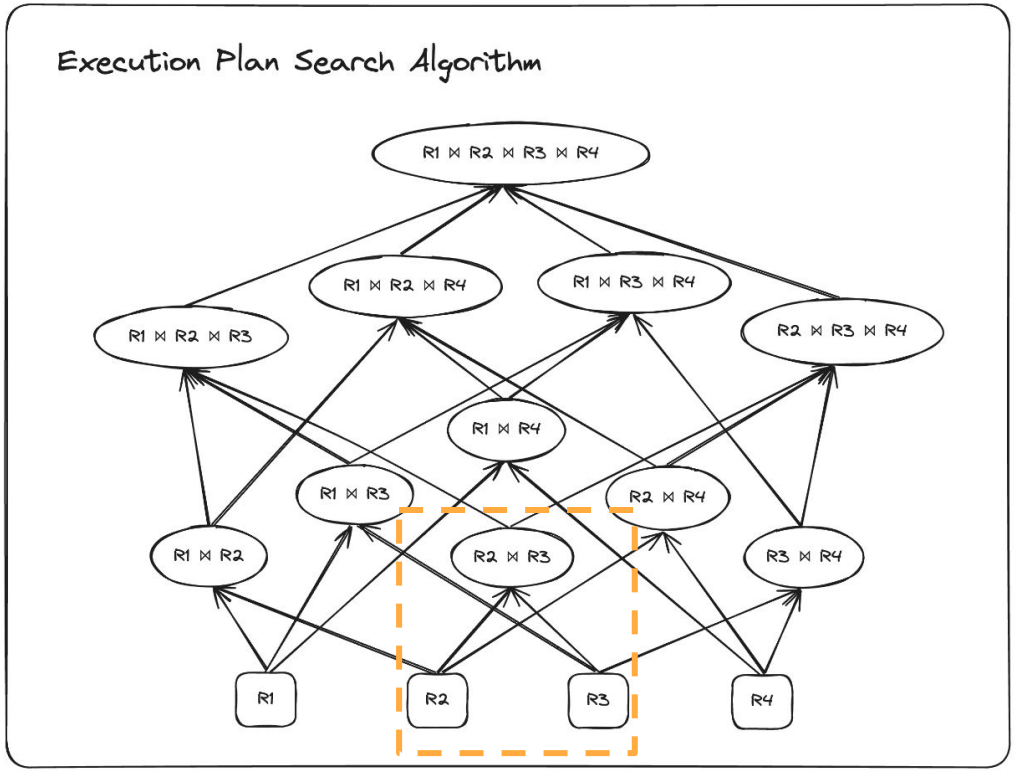


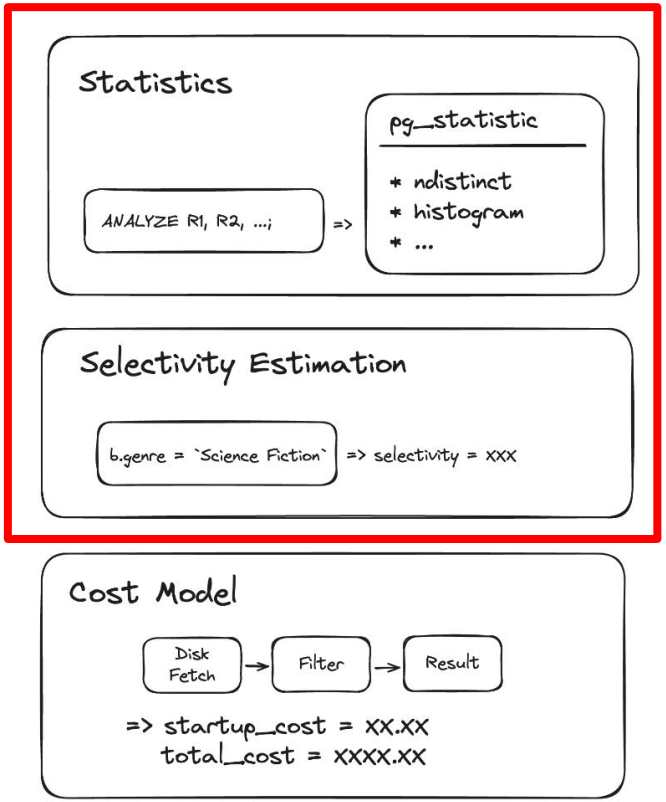



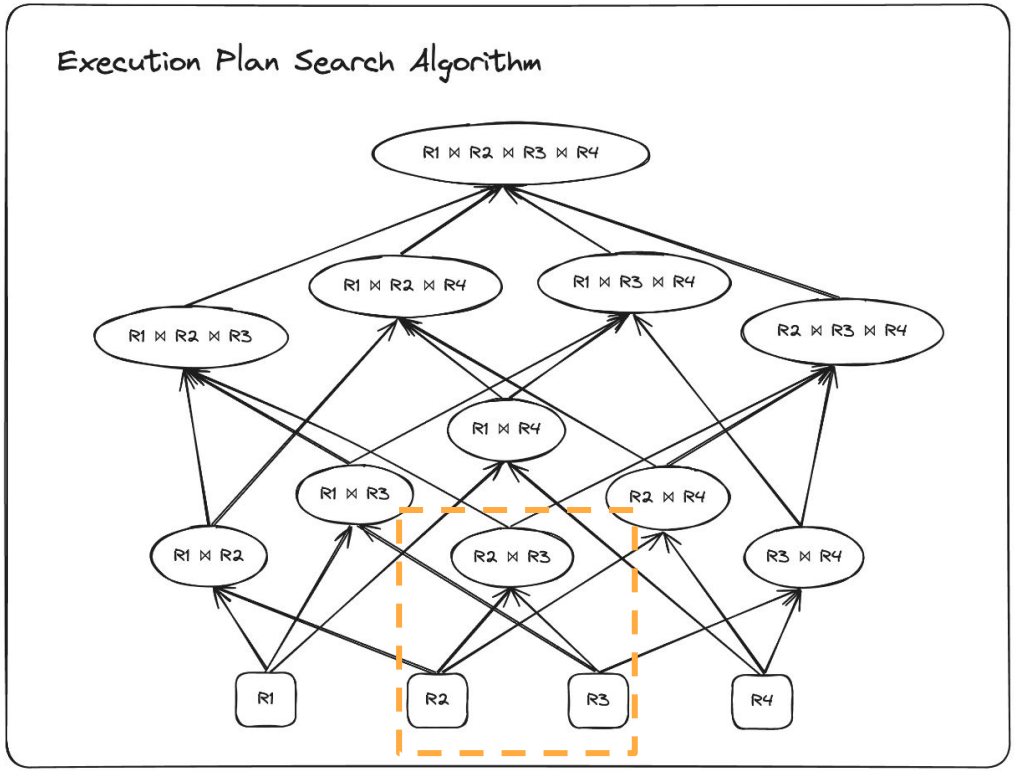



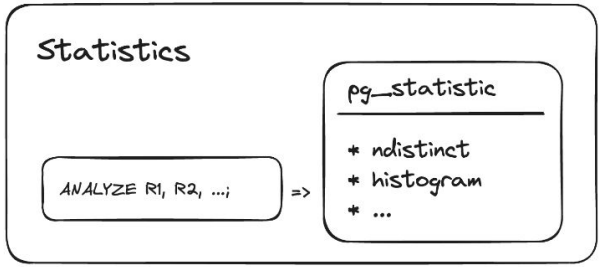
# How does CBO work?

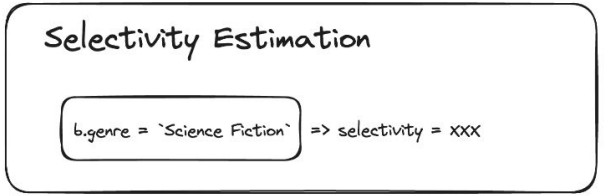


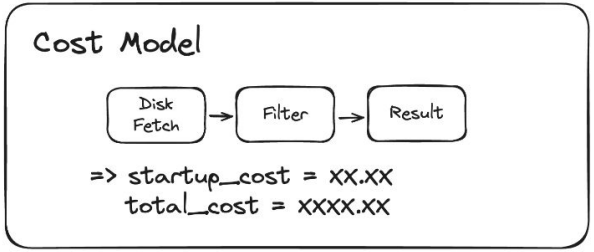



# How does CBO work?



# cbo_stat_dump

- To pick a plan, Query Planner relies on
  - `reltuples` and `relpages` from `pg_class`
  - `pg_statistic`
  - `pg_statistic_ext`
  - `pg_statistic_ext_data`
  - GUC parameters
- To reproduce plans, we also need
  - DDL to create the tables, views, indexes, extended statistics and UDFs.
- https://github.com/yugabyte/cbo_stat_dump

# Caveat #1

- To solve problems caused by stale statistics, PG added optimization in PG15
  - Check pages used by table in storage.
  - Extrapolate `reltuples` by the factor of actual pages / `relpages`.
- Since table is empty on test cluster, planner disregards `reltuples` and assumes table has 0 rows.
- Patch needed on test cluster to disable this optimization.
- If statistics are stale, plan may not reproduce accurately.

# Caveat #2

- PostgreSQL does not support inserting to `pg_statistic_ext_data` due to “pseudo types” used.
- If extended statistics are used in production, patch is needed on the test cluster to allow importing statistics.

# cbo_stat_dump --help

```
cbo_stat_dump [--help]
              [-h HOST] [-p PORT]
              [-u USERNAME] [-W PASSWORD]
              [-d DATABASE]
              [-s SCHEMAS]
              [-q QUERY_FILE]
              [-o OUT_DIR]
              [--yb_mode]
```

# `cbo_stat_dump` Output

| File name | Description |
|---|---|
| version.txt | PostgreSQL or YugabyteDB version |
| overridden_gucs.sql | Relevant GUCs that have been overridden from default |
| gflags.json | Relevant YugabyteDB gFlag that have been overridden from default |
| ddl.sql | DDL for the object used in the query. |
| query.sql | Query optionally provided using -q option. |
| query_plan.txt | Query plan for the query on the production instance. |
| import_statistics.sql | SQL statements to import statistics |
| import_statistics_ext.sql | SQL statements to import extended statistics |
| Statistics.json | Data from pg_statistic and pg_class in JSON format |
| statistics_ext.json | Data from pg_statistic_ext and pg_statistic_ext_data |

# Caveat #3

- `cbo_stat_dump` uses `pg_dump` to extract DDL.
- When exporting DDL for all objects in a schema, it works as expected.
- When exporting DDL for specific table using `pg_dump -t table_name`, `CREATE STATISTICS` are not exported.
  - Likely a bug, we plan to upstream fix to Postgres.
- Patch available, but this is run on customer deployment.
  - Manually extract `CREATE STATISTICS` statements.

# How to reproduce plan?

- Checkout code for the customer version.
- Apply patches.
- Execute `ddl.sql`
- Execute `overridden_gucs.sql`
- Execute `import_statistics.sql`
- If needed, execute `import_statistics_ext.sql`
- `SET enable_cbo_statistics_simulation = ON;`
- Execute query with `ANALYZE`

# What can you do with this?

- Collect all information needed to debug issues
- Provide effective workarounds such as,
  - Planner hints to force a better plan
  - Recommending changes to indexes
- Root cause analysis using debugger
- Verifying that fix solves the issue
- Add test to regression suite to avoid recurrence of problem

# What else?

- Regression Testing for past customer issues.
- However,
  - Tests can be sensitive and hard to maintain.
  - Changes to CBO may change plans and it can be hard to judge if changes are indeed good.
  - Non-blocking tests? 🤢 Usually bad idea, but may be useful in some cases.
  - Use judiciously.

# What else?

- In tandem with TAQO, quick regression tests.
- TAQO takes a long time to run.
- Export PG regression tests from TAQO
  - Run DDL, import statistics, run queries with `EXPLAIN (COSTS OFF)`
- These tests run fast and serve as sanity check for developers.
- TAQO reruns can validate if plan change was good or bad.

# What else?

- Predict impact of upgrades on query performance.
- Import statistics from old version on new version in test cluster.
- Identify plan changes for critical queries.
  - Validate these are not worse plans, by forcing these plans on older version using hints.

# What else?

- Backup and Restore statistics.
- Save time in recomputing statistics.

- Maybe, restore old statistics?
  - Sometimes updated statistics makes query planner choose worse plans. 😓

# Similar feature coming in PG18

**Transfer statistics during pg_upgrade.**

```
author    Jeff Davis <jdavis@postgresql.org>
          Thu, 20 Feb 2025 09:29:06 +0000 (01:29 -0800)
committer Jeff Davis <jdavis@postgresql.org>
          Thu, 20 Feb 2025 09:29:06 +0000 (01:29 -0800)
commit    1fd1bd871012732e3c6c482667d2f2c56f1a9395
tree      232adb265bbbfdb2c2aa89fe9d2ec8f6c14839c2        tree
parent    7da344b9f84f0c63590a34136f3fa5d0ab128657        commit | diff
```

```
Transfer statistics during pg_upgrade.

Add support to pg_dump for dumping stats, and use that during
pg_upgrade so that statistics are transferred during upgrade. In most
cases this removes the need for a costly re-analyze after upgrade.

Some statistics are not transferred, such as extended statistics or
statistics with a custom stakind.

Now pg_dump accepts the options --schema-only, --no-schema,
--data-only, --no-data, --statistics-only, and --no-statistics; which
allow all combinations of schema, data, and/or stats. The options are
named this way to preserve compatibility with the previous
--schema-only and --data-only options.

Statistics are in SECTION_DATA, unless the object itself is in
SECTION_POST_DATA.

The stats are represented as calls to pg_restore_relation_stats() and
pg_restore_attribute_stats().

Author: Corey Huinker, Jeff Davis
Reviewed-by: Jian He
```

- Intended use case is to restore statistics.
- `cbo_stat_dump` works on existing deployments without modifications
- This change doesn't export extended statistics.

Thank you!