



EDB

EDB Query Advisor

Release 1.0.0

EnterpriseDB Corporation

Jun 07, 2023

CONTENTS

1	EDB Query Advisor release notes	3
1.1	Release notes for Query Advisor version 1.0.0	3
2	Installing Query Advisor	5
2.1	Prerequisites	5
2.2	Install the package	5
3	Parameters	7
4	Using EDB Query Advisor	9
4.1	Functions	9
4.1.1	query_advisor_index_recommendations(min_filter, min_selectivity)	9
5	running as postgres user	11
6	running as postgres user	13
6.1	Views	13
6.1.1	query_advisor_qualstat	13
6.1.2	query_advisor_qualstats_pretty	14
7	running as ro user	15

EDB Query Advisor is a Postgres extension that provides index recommendations by keeping statistics on predicates found in WHERE statements, JOIN clauses, and workload queries. It analyzes the cost-benefit to actual workload queries by replanning them with the hypothetical index. Hypothetical indexes are created based on the predicates collected from the workload queries.

This extension is useful if you want to analyze the most useful indexes that can benefit the workload queries without creating all possible indexes.

The predicate collection works by looking for known patterns in queries. These patterns includes:

- Binary OpExpr — At least one side is a column from a table. Whenever possible, the predicate is swapped so that `CONST OP VAR` expressions are turned into `VAR COMMUTED_OP CONST`. AND and OR expression members are counted as separate entries, for example, `WHERE column1 = 2`, `WHERE column1 = column2`, `WHERE 3 = column3`.
- ScalarArrayOpExpr — The left side is a VAR, and the right side is an array constant. Those are counted one time per element in the array. For example, `WHERE column1 IN (2, 3)` is counted as two occurrences for the `(column1, '=')` operator pair.
- BooleanTest — The expression is a simple Boolean column reference, for example, `WHERE column1 IS TRUE`. Clauses like `WHERE columns1`, `WHERE NOT column1` aren't currently processed.

The extension saves the first query text, as is, for each distinct queryid executed in the workload table.

The gathered data isn't saved when the Postgres server is restarted.

See [Using EDB Query Advisor](#) for a description of the `query_advisor_index_recommendations` function you use to generate the index recommendation.

Limitations

- Only single and two columns indexes are considered.
- Statistics aren't collected for utility commands.
- The number of predicates and workload entries are static. A change requires a restart.

EDB QUERY ADVISOR RELEASE NOTES

The EDB Query Advisor documentation describes the latest version of EDB Query Advisor, including minor releases and patches. These release notes cover what was new in each release. For new functionality introduced in a minor or patch release, there are also indicators in the content about the release that introduced the feature.

Version	Release Date
<i>Release notes for Query Advisor version 1.0.0</i>	2023 May 10

1.1 Release notes for Query Advisor version 1.0.0

This release of Query Advisor includes:

Type	Description
Feature	This is the initial release.

INSTALLING QUERY ADVISOR

EDB Query Advisor is supported on the same platforms as the Postgres distribution you're using. Support for EDB Query Advisor starts with Postgres 12. For details, see:

- [EDB Postgres Advanced Server Product Compatibility](#)
- [PostgreSQL Product Compatibility](#)
- [EDB Postgres Extended Server Product Compatibility](#)

2.1 Prerequisites

Before you begin the installation process:

- Install Postgres. See: - [Installing EDB Postgres Advanced Server](#)
- [Installing PostgreSQL](#)
- [Installing EDB Postgres Extended Server](#)
- Set up the repository.

Setting up the repository is a one-time task. If you've already set up your repository, you don't need to perform this step.

To set up the repository, go to [EDB repositories](#) and follow the instructions provided there.

2.2 Install the package

The syntax for the package install command is:

```
sudo <package-manager> -y install edb-<postgres><postgres_version>-query-advisor
```

Where:

- `<package-manager>` is the package manager used with your operating system:

Package manager	Operating system
dnf	RHEL 8 and derivatives
yum	RHEL 7 and derivatives, CentOS 7
zypper	SLES

continues on next page

Table 1 – continued from previous page

Package manager	Operating system
apt-get	Debian and derivatives

- `<postgres>` is the distribution of Postgres you're using:

Postgres distribution	Value
PostgreSQL	pg
EDB Postgres Advanced Server	as
EDB Postgres Extended Server	postgresextended

- `<postgres_version>` is the version of Postgres you're using.

For example, to install EDB Query Advisor for EDB Postgres Advanced Server 15 on a RHEL 8 platform:

```
sudo dnf -y install edb-as15-query-advisor
```

- – title: Configuring EDB Query Advisor navTitle: Configuring —

To run EDB Query Advisor, you need to add it to `shared_preload_libraries` and create the extension in the database.

1. In the `postgresql.conf` file, add `query_advisor` to the `shared_preload_libraries` parameter:

```
shared_preload_libraries = query_advisor
```

Note: If `shared_preload_libraries` has other extensions, then you can add `query_advisor` to the list. The order doesn't matter. !!!

1. Restart Postgres.
2. Create the EDB Query Advisor extension in your database:

```
CREATE EXTENSION query_advisor;
```

PARAMETERS

The following custom GUCs control the EDB Query Advisor extension behavior.

If you modify these parameters, then reload Postgres to apply the changes:

- `query_advisor.enabled` — Specifies whether to enable `query_advisor`. It takes a Boolean value. The default is `true`.
- `query_advisor.sample_rate` — Sets the fraction of queries to sample. It takes a double value. For example, 0.1 indicates to sample only 1 out of 10 queries. The default is -1, which specifies automatic and results in a value of $1 / \text{max_connections}$.

If you modify these parameters, then restart Postgres to apply the changes:

- `query_advisor.max_qual_entries` — Sets the maximum number of predicates tracked. The default is 100000.
- `query_advisor.max_workload_entries` — Sets the maximum number of workload queries tracked. The default is 100000.
- `query_advisor.max_workload_query_size` — Sets the maximum size of the workload query. The default is 1024 bytes.

USING EDB QUERY ADVISOR

EDB Query Advisor defines functions and views.

4.1 Functions

4.1.1 `query_advisor_index_recommendations(min_filter, min_selectivity)`

This function performs a global index suggestion.

By default, only predicates filtering at least 1000 rows and 30% of the rows in average are considered. You can use the `min_filter` and `min_selectivity` parameters to override the default.

The function generates the one- and two-column index candidates based on the predicates it collects. It replans all related workload queries in the presence of the hypothetical index with respect to each candidate. It recommends the list of indexes that bring the most value to the workload. It also shows the estimated index size and percentage cost reduction in the workload queries. You can decide, based on the size and benefit ratio, which indexes are the most useful for you.

For example:

RUNNING AS POSTGRES USER

```
select * from query_advisor_index_recommendations(0,0);
__OUTPUT__
|          index          | estimated_size_in_
| bytes | estimated_pct_cost_reduction |
+-----+-----+
| CREATE INDEX ON nation USING btree (n_name); | 8192
| 14.619857 |
| CREATE INDEX ON supplier USING btree (s_suppkey,s_nationkey); | 933888
| 0.63003576 |
| CREATE INDEX ON partsupp USING btree (ps_suppkey); | 50159616
| 13.254544 |
(3 rows)
```

This function returns the counts for every qualifier identified by the expression hash. This hash identifies each expression.

Qualifier	Description
userid	ID of the user who executed the query.
dbid	ID of the database in which the query was executed.
lrelid, lattnum	ID of the relation and attribute number of the VAR on the left side, if any.
opno	ID of the operator used in the expression
rrelid, rattnum	ID of the relation and attribute number of the VAR on the right side, if any.
qualid	Normalized identifier of the parent AND expression, if any. This identifier is computed excluding the constants. This qualifier is useful for identifying predicates that are used together.
uniquequalid	Unique identifier of the parent AND expression, if any. This identifier is computed including the constants.
qualnodeid	Normalized identifier of this simple predicate. This identifier is computed excluding the constants.
uniquequalnodeid	Unique identifier of this simple predicate. This identifier is computed including the constants.
occurrences	Number of times this predicate was invoked, that is, number of related query execution.
execution_count	Number of times this predicate was executed, that is, the number of rows it processed.

continues on next page

Table 1 – continued from previous page

Qualifier	Description
nbfiltered	Number of tuples this predicate discarded.
queryid	Query identifier.
eval_typ	The evaluation type: <i>f</i> for a predicate evaluated after a scan or <i>i</i> for an index predicate.

For example:

RUNNING AS POSTGRES USER

```
select * from query_advisor_qualstats();
__OUTPUT__
| userid | dbid | lrelid | lattnum | opno | rrelid | rattnum | qualid | uniquequalid |
| qualnodeid | uniquequalnodeid | occurrence | execution_count | nbfiltered | min_err_
| estimate_ratio | max_err_estimate_ratio | mean_err_estimate_ratio | stddev_err_
| estimate_ratio | min_err_estimate_num | max_err_estimate_num | mean_err_estimate_num |
| stddev_err_estimate_num | queryid | eval_type |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 10 | 5 | 16501 | 1 | 96 | | | | |
| 262905824 | 262905824 | 1 | 3000000 | 2999997 | 3 |
| | 3 | | 3 | | 0 |
| | 2 | | 2 | | 2 | | 0 |
| | -3469822585968758916 | f | |
```

Returns all the stored query texts.

Returns the percentage usage of the workload and qualifier hash table.

Resets the internal counters and forgets about every encountered qualifier and workload entry.

6.1 Views

The extension defines the following views on top of the `pg_qualstats` function.

6.1.1 query_advisor_qualstat

Filters calls to `query_advisor_qualstats()` by the current database.

6.1.2 query_advisor_qualstats_pretty

Performs the appropriate joins to display a readable, aggregated form for every attribute from the query_advisor_qualstats view.

For example:

RUNNING AS RO USER

```
select * from query_advisor_qualstats_pretty;
```

__OUTPUT__

left_schema	left_table	left_column	operator	right_schema	right_
↪table	right_column	occurrences	execution_count	nbfiltered	
-----+-----+-----+-----+-----+-----					
↪	+	-----+-----+-----+-----+-----+-----			
public	pgbench_accounts	aid	pg_catalog.=		↪
↪		5	50000000	4999995	
public	pgbench_tellers	tid	pg_catalog.=		↪
↪		10	100000000	9999990	
public	pgbench_branches	bid	pg_catalog.=		↪
↪		10	20000000	1999990	
public	t1	id	pg_catalog.=	public	↪
↪	id_t1	1	10000	9999	t2