Quick Links

- Community
- Contributors
- Mailing Lists
- IRC
- Local User Groups
- Events
- International Sites

CREATE OR REPLACE MATERIALIZED VIEW

Lists:pgsql-hackers

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: pgsql-hackers(at)postgresql(dot)org
Subject: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-02 01:22:00
Message-ID:3c86a16f-4272-4df3-9959-70a9a7d88a71@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
I like to add CREATE OR REPLACE MATERIALIZED VIEW with the attached
patches.

Patch 0001 adds CREATE OR REPLACE MATERIALIZED VIEW similar to CREATE OR
REPLACE VIEW.  It also includes regression tests and changes to docs.

Patch 0002 deprecates CREATE MATERIALIZED VIEW IF NOT EXISTS because it
no longer seems necessary with patch 0001.  Tom Lane commented[1] about
the general dislike of IF NOT EXISTS, to which I agree, but maybe this
was meant only in response to adding new commands.  Anyway, my idea is
to deprecate that usage in PG18 and eventually remove it in PG19, if
there's consensus for it.  We can drop that clause without violating any
standard because matviews are a Postgres extension.  I'm not married to
the idea, just want to put it on the table for discussion.

Motivation
----------

At $JOB we use materialized views for caching a couple of expensive
views.  But every now and then those views have to be changed, e.g., new
logic, new columns, etc.  The matviews have to be dropped and re-created
to include new columns.  (Just changing the underlying view logic
without adding new columns is trivial because the matviews are just thin
wrappers that just have to be refreshed.)

We also have several views that depend on those matviews.  The views
must also be dropped in order to re-create the matviews.  We've already
automated this with two procedures that stash and re-create dependent
view definitions.

Native support for replacing matviews would simplify our setup and it
would make CREATE MATERIALIZED VIEW more complete when compared to
CREATE VIEW.

I searched the lists for previous discussions on this topic but couldn't
find any.  So, I don't know if this was ever tried, but rejected for
some reason.  I've found slides[2] from 2013 (when matviews landed in
9.3) which have OR REPLACE on the roadmap:

> Materialised Views roadmap
>
> * CREATE **OR REPLACE** MATERIALIZED VIEW
>   * Just an oversight that it wasn't added
>  [...]

Replacing Matviews
------------------

With patch 0001, a matview can be replaced without having to drop it and
its dependent objects.  In our use case it is no longer necessary to
define the actual query in a separate view.  Replacing a matview works
analogous to CREATE OR REPLACE VIEW:

* the new query may change SELECT list expressions of existing columns
* new columns can be added to the end of the SELECT list
* existing columns cannot be renamed
* the data type of existing columns cannot be changed

In addition to that, CREATE OR REPLACE MATERIALIZED VIEW also replaces
access method, tablespace, and storage parameters if specified.  The
clause WITH [NO] DATA works as expected: it either populates the matview
or leaves it in an unscannable state.

It is an error to specify both OR REPLACE and IF NOT EXISTS.

Example
-------

postgres=# CREATE MATERIALIZED VIEW test AS SELECT 1 AS a;
SELECT 1
postgres=# SELECT * FROM test;
 a
---
 1
(1 row)

postgres=# CREATE OR REPLACE MATERIALIZED VIEW test AS SELECT 2 AS a, 3 AS b;
CREATE MATERIALIZED VIEW
postgres=# SELECT * FROM test;
 a | b
---+---
 2 | 3
(1 row)

Implementation Details
----------------------

Patch 0001 extends create_ctas_internal in order to adapt an existing
matview to the new tuple descriptor, access method, tablespace, and
storage parameters.  This logic is mostly based on DefineViewRelation.
This also reuses checkViewColumns, but adds argument is_matview in order
to tell if we want error messages for a matview (true) or view (false).
I'm not sure if that flag is the correct way to do that, or if I should
just create a separate function just for matviews with the same logic.
Do we even need to distinguish between view and matview in those error
```

messages?

The patch also adds tab completion in psql for CREATE OR REPLACE
MATERIALIZED VIEW.

[1] **https://www.postgresql.org/message-id/226806.1693430777%40sss.pgh.pa.us**
[2] **https://wiki.postgresql.org/images/a/ad/Materialised_views_now_and_the_future-pgconfeu_2013.pdf#page=23**

--
Erik

Attachment	Content-Type	Size
v1-0001-Add-CREATE-OR-REPLACE-MATERIALIZED-VIEW.patch	text/x-diff	30.8 KB
v1-0002-Deprecate-CREATE-MATERIALIZED-VIEW-IF-NOT-EXISTS.patch	text/x-diff	8.5 KB

From: Aleksander Alekseev <aleksander(at)timescale(dot)com>
To: pgsql-hackers(at)postgresql(dot)org
Cc: Erik Wienhold <ewie(at)ewie(dot)name>
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-02 10:46:21
Message-ID: CAJ7c6TOB66xgbVXTcuje31WS+t0r81TTkMQOv941YRNiBQbORA@mail.gmail.com
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

Hi,

> Patch 0002 deprecates CREATE MATERIALIZED VIEW IF NOT EXISTS because it
> no longer seems necessary with patch 0001. Tom Lane commented[1] about
> the general dislike of IF NOT EXISTS, to which I agree, but maybe this
> was meant only in response to adding new commands. Anyway, my idea is
> to deprecate that usage in PG18 and eventually remove it in PG19, if
> there's consensus for it. We can drop that clause without violating any
> standard because matviews are a Postgres extension. I'm not married to
> the idea, just want to put it on the table for discussion.

I can imagine how this may impact many applications and upset many
software developers worldwide. Was there even a precedent (in the
recent decade or so) when PostgreSQL broke the SQL syntax?

To clarify, I'm not opposed to this idea. If we are fine with breaking
backward compatibility on the SQL level, this would allow dropping the
support of inherited tables some day, a feature that in my humble
opinion shouldn't exist (I realize this is another and very debatable
question though). I just don't think this is something we ever do in
this project. But I admit that this information may be incorrect
and/or outdated.

--
Best regards,
Aleksander Alekseev

From: Daniel Gustafsson <daniel(at)yesql(dot)se>
To: Erik Wienhold <ewie(at)ewie(dot)name>
Cc: PostgreSQL Hackers <pgsql-hackers(at)postgresql(dot)org>
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-02 12:27:27
Message-ID: FD0EB80A-D77C-4F28-8AC3-165014AE17A9@yesql.se
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

> On 2 Jul 2024, at 03:22, Erik Wienhold <ewie(at)ewie(dot)name> wrote:

> Patch 0002 deprecates CREATE MATERIALIZED VIEW IF NOT EXISTS because it
> no longer seems necessary with patch 0001. Tom Lane commented[1] about
> the general dislike of IF NOT EXISTS, to which I agree, but maybe this
> was meant only in response to adding new commands. Anyway, my idea is
> to deprecate that usage in PG18 and eventually remove it in PG19, if
> there's consensus for it.

Considering the runway we typically give for deprecations, that seems like a
fairly short timeframe for a SQL level command which isn't unlikely to exist
in application code.

--
Daniel Gustafsson

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: Daniel Gustafsson <daniel(at)yesql(dot)se>
Cc: PostgreSQL Hackers <pgsql-hackers(at)postgresql(dot)org>, Aleksander Alekseev <aleksander(at)timescale(dot)com>
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-02 13:58:06
Message-ID: 3172b11f-1c06-4168-af81-740b72a77979@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

I wrote:
> Patch 0002 deprecates CREATE MATERIALIZED VIEW IF NOT EXISTS because it
> no longer seems necessary with patch 0001. Tom Lane commented[1] about
> the general dislike of IF NOT EXISTS, to which I agree, but maybe this
> was meant only in response to adding new commands.

One could also argue that since matviews are a hybrid of tables and
views, that CREATE MATERIALIZED VIEW should accept both OR REPLACE (as
in CREATE VIEW) and IF NOT EXISTS (as in CREATE TABLE). But not in the
same invocation of course.

On 2024-07-02 12:46 +0200, Aleksander Alekseev wrote:
> > Anyway, my idea is to deprecate that usage in PG18 and eventually
> > remove it in PG19, if there's consensus for it. We can drop that
> > clause without violating any standard because matviews are a
> > Postgres extension. I'm not married to the idea, just want to put
> > it on the table for discussion.
>
> I can imagine how this may impact many applications and upset many
> software developers worldwide. Was there even a precedent (in the
> recent decade or so) when PostgreSQL broke the SQL syntax?

A quick spelunking through the changelog with

    git log --grep deprecat -i --since '10 years ago'

turned up two commits:

```
578b229718 "Remove WITH OIDS support, change oid catalog column visibility."
e8d016d819 "Remove deprecated COMMENT ON RULE syntax"

Both were committed more than 10 years after deprecating the respective
feature.  My proposed one-year window seems a bit harsh in comparison.

On 2024-07-02 14:27 +0200, Daniel Gustafsson wrote:
> Considering the runway we typically give for deprecations, that seems like a
> fairly short timeframe for a SQL level command which isn't unlikely to exist
> in application code.

Is there some general agreed upon timeframe, or is decided on a
case-by-case basis?  I can imagine waiting at least until the last
release without the deprecation reaches EOL.  That would be 5 years with
the current versioning policy.

--
Erik
```

From: Daniel Gustafsson <daniel(at)yesql(dot)se>
To: Erik Wienhold <ewie(at)ewie(dot)name>
Cc: PostgreSQL Hackers <pgsql-hackers(at)postgresql(dot)org>, Aleksander Alekseev <aleksander(at)timescale(dot)com>
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-02 14:01:45
Message-ID: 366B2727-F134-4FB6-93B9-BCB7DFF6F208@yesql.se
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
> On 2 Jul 2024, at 15:58, Erik Wienhold <ewie(at)ewie(dot)name> wrote:
> On 2024-07-02 14:27 +0200, Daniel Gustafsson wrote:
>> Considering the runway we typically give for deprecations, that seems like a
>> fairly short timeframe for a SQL level command which isn't unlikely to exist
>> in application code.
>
> Is there some general agreed upon timeframe, or is decided on a
> case-by-case basis?  I can imagine waiting at least until the last
> release without the deprecation reaches EOL.  That would be 5 years with
> the current versioning policy.

AFAIK it's all decided on a case-by-case basis depending on impact.  There are
for example the removals you listed, and there are functions in libpq which
were deprecated in the postgres 6.x days which are still around to avoid
breaking ABI.

--
Daniel Gustafsson
```

From: Said Assemlal <sassemlal(at)neurorx(dot)com>
To: Erik Wienhold <ewie(at)ewie(dot)name>, pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-04 20:18:11
Message-ID: d9db9bdb-593d-4cef-9274-0db640fcff42@neurorx.com
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
Hi,

+1 for this feature.

> Replacing Matviews
> ------------------
>
> With patch 0001, a matview can be replaced without having to drop it and
> its dependent objects.  In our use case it is no longer necessary to
> define the actual query in a separate view.  Replacing a matview works
> analogous to CREATE OR REPLACE VIEW:
>
> * the new query may change SELECT list expressions of existing columns
> * new columns can be added to the end of the SELECT list
> * existing columns cannot be renamed
> * the data type of existing columns cannot be changed
>
> In addition to that, CREATE OR REPLACE MATERIALIZED VIEW also replaces
> access method, tablespace, and storage parameters if specified.  The
> clause WITH [NO] DATA works as expected: it either populates the matview
> or leaves it in an unscannable state.
>
> It is an error to specify both OR REPLACE and IF NOT EXISTS.

I noticed replacing the materialized view is blocking all reads. Is that
expected ? Even if there is a unique index ?

Best,
Sa_id_
```

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: Said Assemlal <sassemlal(at)neurorx(dot)com>
Cc: pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-05 23:42:48
Message-ID: e4382244-6b55-450b-a4f0-32959056ade4@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
On 2024-07-04 22:18 +0200, Said Assemlal wrote:
> +1 for this feature.

Thanks!

> I noticed replacing the materialized view is blocking all reads. Is that
> expected ? Even if there is a unique index ?

That is expected because AccessExclusiveLock is acquired on the existing
matview.  This is also the case for CREATE OR REPLACE VIEW.

My initial idea, while writing the patch, was that one could replace the
matview without populating it and then run the concurrent refresh, like
this:

    CREATE OR REPLACE MATERIALIZED VIEW foo AS ... WITH NO DATA;
    REFRESH MATERIALIZED VIEW CONCURRENTLY foo;

But that won't work because concurrent refresh requires an already
populated matview.

Right now the patch either populates the replaced matview or leaves it
```

```
in an unscannable state.  Technically, it's also possible to skip the
refresh and leave the old data in place, perhaps by specifying
WITH *OLD* DATA.  New columns would just be null.  Of course you can't
tell if you got stale data without knowing how the matview was replaced.
Thoughts?

--
Erik
```

From: Said Assemlal <sassemlal(at)neurorx(dot)com>
To: Erik Wienhold <ewie(at)ewie(dot)name>
Cc: pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-12 14:49:14
Message-ID: e55c4930-f788-4bb0-a684-743621d6cfc5@neurorx.com
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
> That is expected because AccessExclusiveLock is acquired on the existing
> matview.  This is also the case for CREATE OR REPLACE VIEW.

Right, had this case many times.

>
> My initial idea, while writing the patch, was that one could replace the
> matview without populating it and then run the concurrent refresh, like
> this:
>
>      CREATE OR REPLACE MATERIALIZED VIEW foo AS ... WITH NO DATA;
>      REFRESH MATERIALIZED VIEW CONCURRENTLY foo;
>
> But that won't work because concurrent refresh requires an already
> populated matview.
>
> Right now the patch either populates the replaced matview or leaves it
> in an unscannable state.  Technically, it's also possible to skip the
> refresh and leave the old data in place, perhaps by specifying
> WITH *OLD* DATA.  New columns would just be null.  Of course you can't
> tell if you got stale data without knowing how the matview was replaced.
> Thoughts?

I believe the expectation is to get materialized views updated whenever
it gets replaced so likely to confuse users ?
```

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: Said Assemlal <sassemlal(at)neurorx(dot)com>
Cc: pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-07-27 00:45:15
Message-ID: 7afe68b0-f983-4a9f-a1b4-32188cebbf38@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
On 2024-07-12 16:49 +0200, Said Assemlal wrote:
> > My initial idea, while writing the patch, was that one could replace the
> > matview without populating it and then run the concurrent refresh, like
> > this:
> >
> >      CREATE OR REPLACE MATERIALIZED VIEW foo AS ... WITH NO DATA;
> >      REFRESH MATERIALIZED VIEW CONCURRENTLY foo;
> >
> > But that won't work because concurrent refresh requires an already
> > populated matview.
> >
> > Right now the patch either populates the replaced matview or leaves it
> > in an unscannable state.  Technically, it's also possible to skip the
> > refresh and leave the old data in place, perhaps by specifying
> > WITH *OLD* DATA.  New columns would just be null.  Of course you can't
> > tell if you got stale data without knowing how the matview was replaced.
> > Thoughts?
>
> I believe the expectation is to get materialized views updated whenever it
> gets replaced so likely to confuse users ?

I agree, that could be confusing -- unless it's well documented.  The
attached 0003 implements WITH OLD DATA and states in the docs that this
is intended to be used before a concurrent refresh.

Patch 0001 now covers all matview cases in psql's tab completion.  I
missed some of them with v1.

--
Erik
```

Attachment	Content-Type	Size
v2-0001-Add-CREATE-OR-REPLACE-MATERIALIZED-VIEW.patch	text/x-diff	31.8 KB
v2-0002-Deprecate-CREATE-MATERIALIZED-VIEW-IF-NOT-EXISTS.patch	text/x-diff	8.5 KB
v2-0003-Replace-matview-WITH-OLD-DATA.patch	text/x-diff	7.7 KB

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: Said Assemlal <sassemlal(at)neurorx(dot)com>
Cc: pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-09-05 20:33:51
Message-ID: b74736cf-a4b4-4c32-8df1-d08abe9a14ed@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
On 2024-07-27 02:45 +0200, Erik Wienhold wrote:
> On 2024-07-12 16:49 +0200, Said Assemlal wrote:
> > > My initial idea, while writing the patch, was that one could replace the
> > > matview without populating it and then run the concurrent refresh, like
> > > this:
> > >
> > >      CREATE OR REPLACE MATERIALIZED VIEW foo AS ... WITH NO DATA;
> > >      REFRESH MATERIALIZED VIEW CONCURRENTLY foo;
> > >
> > > But that won't work because concurrent refresh requires an already
> > > populated matview.
> > >
```

```
> > > Right now the patch either populates the replaced matview or leaves it
> > > in an unscannable state.  Technically, it's also possible to skip the
> > > refresh and leave the old data in place, perhaps by specifying
> > > WITH *OLD* DATA.  New columns would just be null.  Of course you can't
> > > tell if you got stale data without knowing how the matview was replaced.
> > > Thoughts?
> > >
> > I believe the expectation is to get materialized views updated whenever it
> > gets replaced so likely to confuse users ?
>
> I agree, that could be confusing -- unless it's well documented.  The
> attached 0003 implements WITH OLD DATA and states in the docs that this
> is intended to be used before a concurrent refresh.
>
> Patch 0001 now covers all matview cases in psql's tab completion.  I
> missed some of them with v1.

Here's a rebased version due to conflicts with f683d3a4ca and
1e35951e71.  No other changes since v2.

--
Erik
```

Attachment	Content-Type	Size
v3-0001-Add-CREATE-OR-REPLACE-MATERIALIZED-VIEW.patch	text/x-diff	31.9 KB
v3-0002-Deprecate-CREATE-MATERIALIZED-VIEW-IF-NOT-EXISTS.patch	text/x-diff	8.5 KB
v3-0003-Replace-matview-WITH-OLD-DATA.patch	text/x-diff	7.7 KB

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: Said Assemlal <sassemlal(at)neurorx(dot)com>
Cc: pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-10-31 00:22:17
Message-ID: d4bf3ccf-5c2b-448d-ad16-59dd6fbe2a2e@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
On 2024-09-05 22:33 +0200, Erik Wienhold wrote:
> On 2024-07-27 02:45 +0200, Erik Wienhold wrote:
> > On 2024-07-12 16:49 +0200, Said Assemlal wrote:
> > > > My initial idea, while writing the patch, was that one could replace the
> > > > matview without populating it and then run the concurrent refresh, like
> > > > this:
> > > >
> > > >      CREATE OR REPLACE MATERIALIZED VIEW foo AS ... WITH NO DATA;
> > > >      REFRESH MATERIALIZED VIEW CONCURRENTLY foo;
> > > >
> > > > But that won't work because concurrent refresh requires an already
> > > > populated matview.
> > > >
> > > > Right now the patch either populates the replaced matview or leaves it
> > > > in an unscannable state.  Technically, it's also possible to skip the
> > > > refresh and leave the old data in place, perhaps by specifying
> > > > WITH *OLD* DATA.  New columns would just be null.  Of course you can't
> > > > tell if you got stale data without knowing how the matview was replaced.
> > > > Thoughts?
> > > >
> > > I believe the expectation is to get materialized views updated whenever it
> > > gets replaced so likely to confuse users ?
> > >
> > I agree, that could be confusing -- unless it's well documented.  The
> > attached 0003 implements WITH OLD DATA and states in the docs that this
> > is intended to be used before a concurrent refresh.
> >
> > Patch 0001 now covers all matview cases in psql's tab completion.  I
> > missed some of them with v1.
>
> Here's a rebased version due to conflicts with f683d3a4ca and
> 1e35951e71.  No other changes since v2.

rebased

--
Erik
```

Attachment	Content-Type	Size
v4-0001-Add-CREATE-OR-REPLACE-MATERIALIZED-VIEW.patch	text/x-diff	31.8 KB
v4-0002-Deprecate-CREATE-MATERIALIZED-VIEW-IF-NOT-EXISTS.patch	text/x-diff	11.0 KB
v4-0003-Replace-matview-WITH-OLD-DATA.patch	text/x-diff	7.7 KB

From: Michael Paquier <michael(at)paquier(dot)xyz>
To: Aleksander Alekseev <aleksander(at)timescale(dot)com>
Cc: pgsql-hackers(at)postgresql(dot)org, Erik Wienhold <ewie(at)ewie(dot)name>
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2024-10-31 03:48:21
Message-ID: ZyL-BexBTGJ6Nkat@paquier.xyz
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

```
On Tue, Jul 02, 2024 at 01:46:21PM +0300, Aleksander Alekseev wrote:
> I can imagine how this may impact many applications and upset many
> software developers worldwide. Was there even a precedent (in the
> recent decade or so) when PostgreSQL broke the SQL syntax?

We're usually very careful about that, and maintaining a
backward-compatible grammar has a minimal cost in the parser.  The
closest thing I can think of that has a rather complicated grammar is
COPY, which has *two* legacy grammars still supported, one for ~7.3
and one for ~9.0.

> To clarify, I'm not opposed to this idea. If we are fine with breaking
> backward compatibility on the SQL level, this would allow dropping the
> support of inherited tables some day, a feature that in my humble
> opinion shouldn't exist (I realize this is another and very debatable
> question though). I just don't think this is something we ever do in
> this project. But I admit that this information may be incorrect
```

> and/or outdated.

I am not sure that there is much to gain with this proposal knowing
that the commands with matviews have been around for quite a long time
now. Particularly, after looking at 0001, you'd see that it shortcuts
a couple of areas of the CTAS code because that's what we are relying
on when building the initial data of matviews. Hence,
implementation-wise in the backend, matviews are much closer to
physical relations than views. This is trying to make matviews behave
more consistently with views.

This topic has been mentioned once on pgsql-general back in 2019, for
reference:
https://www.postgresql.org/message-id/CAAWhf%3DPHch2H3ekYnbafuwqWqwyRok8WVPaDxKosZE4GQ2pq5w%40mail.gmail.com
--
Michael

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: Said Assemlal <sassemlal(at)neurorx(dot)com>
Cc: pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2025-01-12 21:33:05
Message-ID: 64495e31-0a91-4894-87f8-31451ee69f44@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

Here's a rebased v5 due to conflicts with de1e298857. No other changes
since v4.

--
Erik Wienhold

Attachment	Content-Type	Size
v5-0001-Add-CREATE-OR-REPLACE-MATERIALIZED-VIEW.patch	text/plain	33.0 KB
v5-0002-Deprecate-CREATE-MATERIALIZED-VIEW-IF-NOT-EXISTS.patch	text/plain	11.0 KB
v5-0003-Replace-matview-WITH-OLD-DATA.patch	text/plain	7.7 KB

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: Said Assemlal <sassemlal(at)neurorx(dot)com>
Cc: pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2025-03-10 02:46:05
Message-ID: bf64dad9-eaaa-4911-b447-13acee89ea80@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

The attached v6 fixes the build. Somehow I missed testing with
--with-cassert the whole time and it turned that out I forgot to pass
queryString to ExecRefreshMatView.

--
Erik Wienhold

Attachment	Content-Type	Size
v6-0001-Add-CREATE-OR-REPLACE-MATERIALIZED-VIEW.patch	text/plain	33.1 KB
v6-0002-Deprecate-CREATE-MATERIALIZED-VIEW-IF-NOT-EXISTS.patch	text/plain	11.0 KB
v6-0003-Replace-matview-WITH-OLD-DATA.patch	text/plain	7.7 KB

From: Tom Lane <tgl(at)sss(dot)pgh(dot)pa(dot)us>
To: Erik Wienhold <ewie(at)ewie(dot)name>
Cc: Said Assemlal <sassemlal(at)neurorx(dot)com>, pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2025-04-05 20:37:08
Message-ID: 2724629.1743885428@sss.pgh.pa.us
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

Erik Wienhold <ewie(at)ewie(dot)name> writes:
> [v6 patches]

A couple of drive-by comments:

* I think the proposal to deprecate IF NOT EXISTS is a nonstarter.
Yeah, I don't like it much, but the standard of proof to remove
features is amazingly high and I don't think it's been reached here.
We're unlikely to remove IF NOT EXISTS for tables, and to the extent
that matviews are like tables it's reasonable for them to have it too.

* On the other hand, the semantics you've implemented for CREATE OR
REPLACE are not right. The contract for any form of C.O.R. is that
it will either fail, or produce exactly the same object definition
that you would have gotten from plain CREATE with no conflicting
object. The v6 code is visibly not doing that for properties such
as tablespace --- if the command doesn't mention that, you don't
get the default tablespace, you get whatever the old object had.

* BTW, I'm inclined to think that WITH OLD DATA ought to fail
if the command isn't replacing an existing matview. It seems
inconsistent to silently reinterpret it as WITH DATA, just as
silently reinterpreting "no tablespace mentioned" as "use the
old tablespace" is inconsistent. I'm not dead set on that
but it feels wrong.

regards, tom lane

From: Erik Wienhold <ewie(at)ewie(dot)name>
To: Tom Lane <tgl(at)sss(dot)pgh(dot)pa(dot)us>
Cc: Said Assemlal <sassemlal(at)neurorx(dot)com>, pgsql-hackers(at)postgresql(dot)org
Subject: Re: CREATE OR REPLACE MATERIALIZED VIEW
Date: 2025-08-05 01:17:06
Message-ID: 5cd7ec92-ee61-4080-8fb6-0aed6a51eeaf@ewie.name
Views: Whole Thread | Raw Message | Download mbox | Resend email
Lists: pgsql-hackers

Sorry for the late reply but I haven't had the time to dig into this.
Here's v7 fixing the points below.

On 2025-04-05 22:37 +0200, Tom Lane wrote:

> * I think the proposal to deprecate IF NOT EXISTS is a nonstarter.
> Yeah, I don't like it much, but the standard of proof to remove
> features is amazingly high and I don't think it's been reached here.
> We're unlikely to remove IF NOT EXISTS for tables, and to the extent
> that matviews are like tables it's reasonable for them to have it too.

Yeah, I got that gist from the replies upthread and dropped that patch.

> * On the other hand, the semantics you've implemented for CREATE OR
> REPLACE are not right. The contract for any form of C.O.R. is that
> it will either fail, or produce exactly the same object definition
> that you would have gotten from plain CREATE with no conflicting
> object. The v6 code is visibly not doing that for properties such
> as tablespace --- if the command doesn't mention that, you don't
> get the default tablespace, you get whatever the old object had.

Thanks a lot. I added a test case for that and v7-0001 now restores the
default options if none are specified. Handling the default tablespace
is a bit cumbersome IMO because its name must be passed to
AlterTableInternal. With v7-0002 I moved that to ATPrepSetTableSpace as
an alternative using the empty string as stand-in for the default
tablespace. What do you think?

> * BTW, I'm inclined to think that WITH OLD DATA ought to fail
> if the command isn't replacing an existing matview. It seems
> inconsistent to silently reinterpret it as WITH DATA, just as
> silently reinterpreting "no tablespace mentioned" as "use the
> old tablespace" is inconsistent. I'm not dead set on that
> but it feels wrong.

Yes that also felt iffy to me. It just didn't occur to me to simply
raise an error in ExecCreateTableAs. Done so in v7-0003.

--
Erik Wienhold

Attachment	Content-Type	Size
v7-0001-Add-OR-REPLACE-option-to-CREATE-MATERIALIZED-VIEW.patch	text/plain	34.9 KB
v7-0002-Handle-default-tablespace-in-AlterTableInternal.patch	text/plain	3.1 KB
v7-0003-Add-WITH-OLD-DATA-to-CREATE-OR-REPLACE-MATERIALIZ.patch	text/plain	8.0 KB