DROP VIEW to Remove Views

by Sophia

WHAT'S COVERED

This tutorial explores using the DROP VIEW command to remove views in three parts:

1. Introduction
2. Dropping a View
3. View Dependencies

1. Introduction

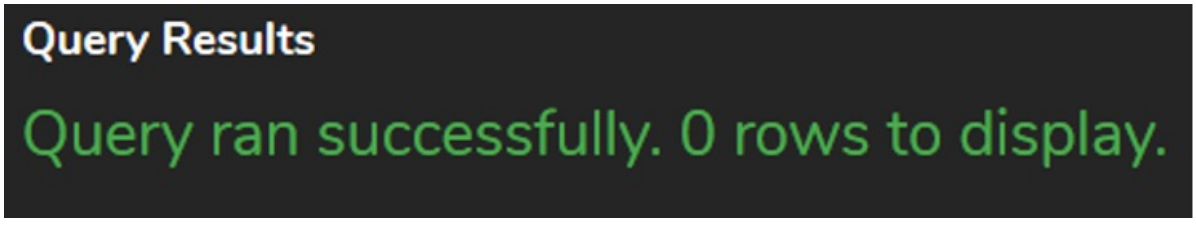
Similar to other objects that we have created, we can use the DROP VIEW statement to remove a view from the database. The syntax of the DROP VIEW command looks like this:

DROP VIEW <viewname>;

2. Dropping a View

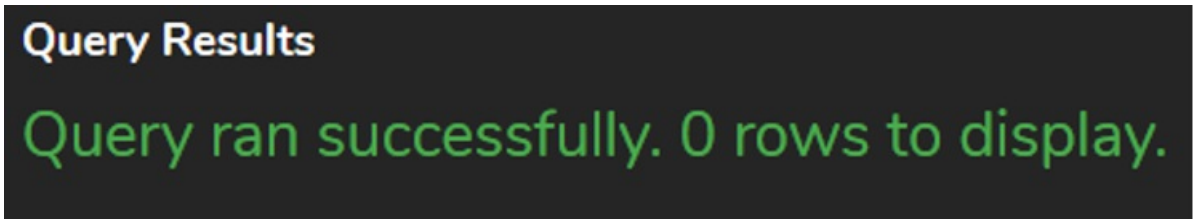
For simplicity, we would just add the view's name after the DROP VIEW keywords. For example, if we created a view for the album and artist names:

```
CREATE VIEW album_artist_names
AS
SELECT album.title, artist.name
FROM album
INNER JOIN artist ON album.artist_id = artist.artist_id;
```



And we wanted to drop the view, it would look like this:

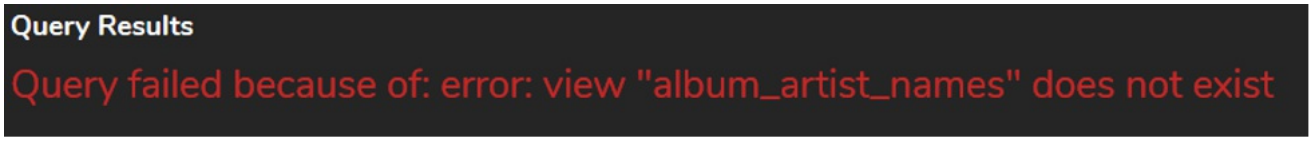
```
DROP VIEW album_artist_names;
```



We were successful in dropping the view. However, it is possible that a view does not exist, if it was labeled incorrectly or was already dropped. There are some additional options that can be used in conjunction with the statement as parameters. One of those includes the IF EXISTS statement, to avoid having an error message presented if the view does not exist.

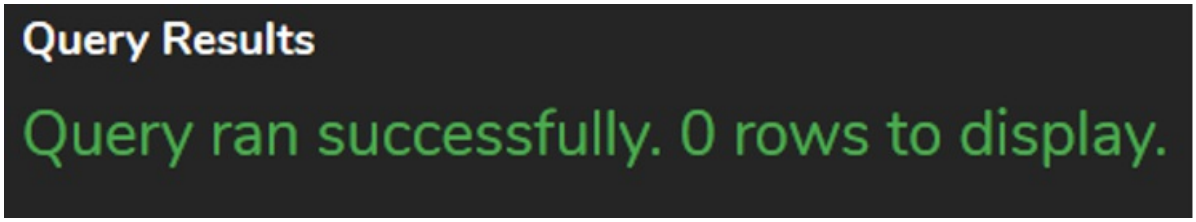
For example, if we tried to drop the view a second time:

```
DROP VIEW album_artist_names;
```



However, by using the IF EXISTS parameter, the database will only drop the view if it exists. If not, no error message will be generated:

```
DROP VIEW IF EXISTS album_artist_names;
```



3. View Dependencies

There are also instances when you may have view dependencies, meaning other objects use a view that you have created. The RESTRICT parameter can be added, although it is used by default to block the view from being dropped. Say we created the album_artist_names view, and then created another view called temp_album_artist_names that queried from the album_artist_names view:

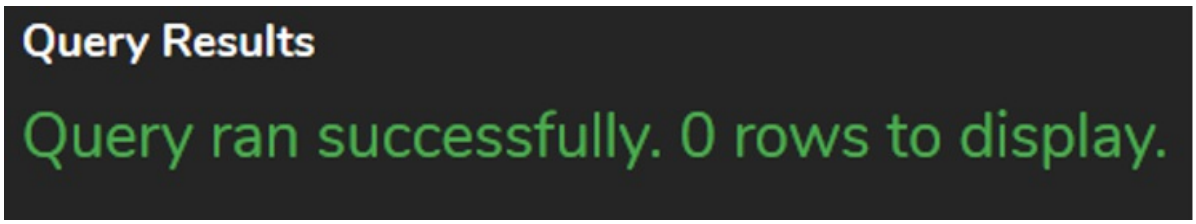
```
CREATE VIEW album_artist_names
AS
SELECT album.title, artist.name
FROM album
INNER JOIN artist ON album.artist_id = artist.artist_id;

CREATE VIEW temp_album_artist_names
AS
SELECT *
```

FROM album_artist_names;
If we tried to drop the album_artist_names view, we would get an error, as the temp_album_artist_names table uses that view:

```
DROP VIEW IF EXISTS album_artist_names
RESTRICT;
```

We can also use the CASCADE option to have the statement automatically drop all of the objects that depend on the view, and all objects that also then depend on those other objects. The statement with the cascade looks like this:

```
DROP VIEW IF EXISTS album_artist_names
CASCADE;
```

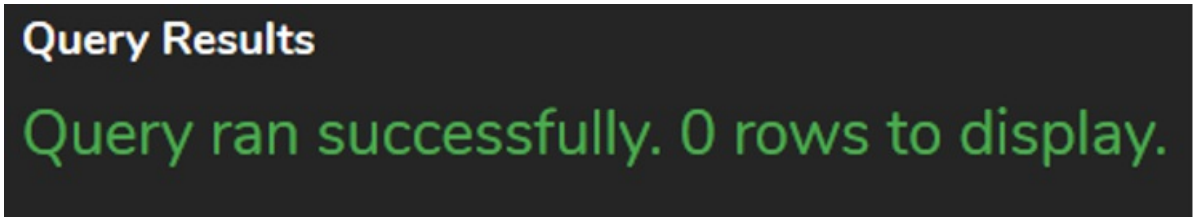


You do have to be careful with such a statement, as you could have many unintended objects getting dropped. Instead, you can drop them in order:

```
DROP VIEW IF EXISTS temp_album_artist_names;

DROP VIEW IF EXISTS album_artist_names;
```

You can also drop multiple views at the same time if you have dependencies:

```
DROP VIEW IF EXISTS album_artist_names, temp_album_artist_names;
```



Notice that even though there is a dependency between the two views, dropping them both on the same line will have the database handle it for you, simplifying that process.

Video Transcription

[MUSIC PLAYING] Once we create a view, we might want to be able to drop it. In this case here, we have a view called album_artist_names that's created to be able to drop it. It's very similar to create table statement with a drop table statement. It'll look like drop view, and then the view name. Go ahead and run that. And once it's done, it'll go ahead and drop the view.

There are some parameters that are also include as part of the drop view statement with the if exists. So if exist, what this will do is verify that the view actually exists to be able to drop it. Otherwise, it'll just run the statement and still provide no error messages. You can also utilize a deep restrict that will have a check in place to be able to ensure that the view can't be dropped if other objects depend on the view.

We can also utilize cascade. This option will allow us to be able to draw this view as well as any other object that depends on this particular view and on the objects that depend on it as well and all at once.

[MUSIC PLAYING]

TRY IT

Your turn! Open the SQL tool by clicking on the LAUNCH DATABASE button below. Then enter in one of the examples above and see how it works. Next, try your own choices for which columns you want the query to provide.

SUMMARY

We can use the DROP VIEW command to remove views that have been created.

Source: Authored by Vincent Tran