Can not change database for pg_cron jobs

Last updated by | Daniel Valero | Sep 28, 2021 at 6:52 AM PDT

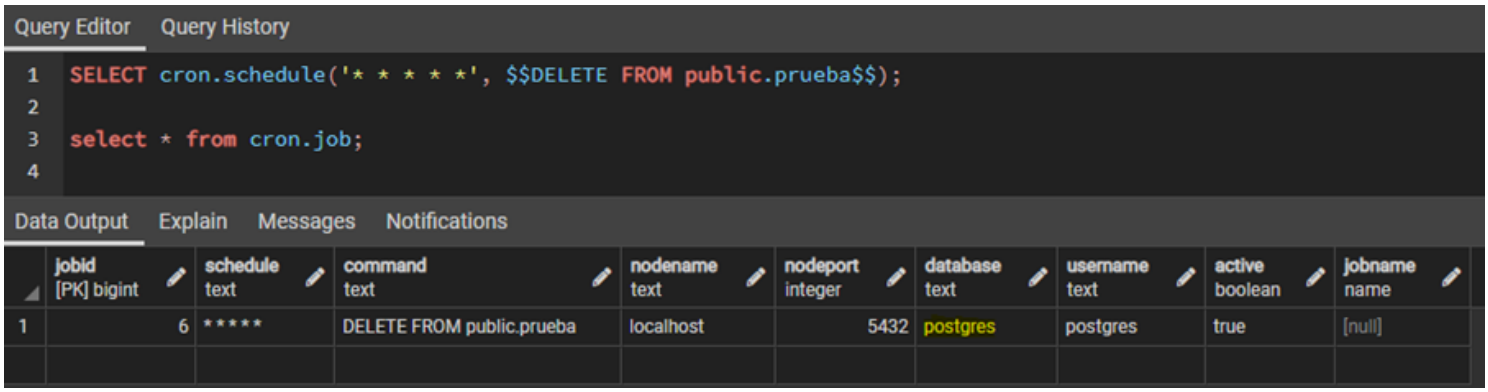
Issue descripiton

The pg_cron metadata is stored in *postgres* (default value for parameter cron.database_name). According to pg_cron doc (https://github.com/citusdata/pg_cron), for security reasons, jobs are executed in the database in which the cron.schedule function is called with the same permissions as the current user

If a job needs to run on a different database, a superuser can update the database column on the cron.job table, using:

```
update cron.job set database='<db_name>' where jobid = <job_id>
```

In PostgreSQL running onPrem on in VM, a job is created on database *postgres* (default value for cron.database_name) as shown in the image bellow:



Query Editor | Query History

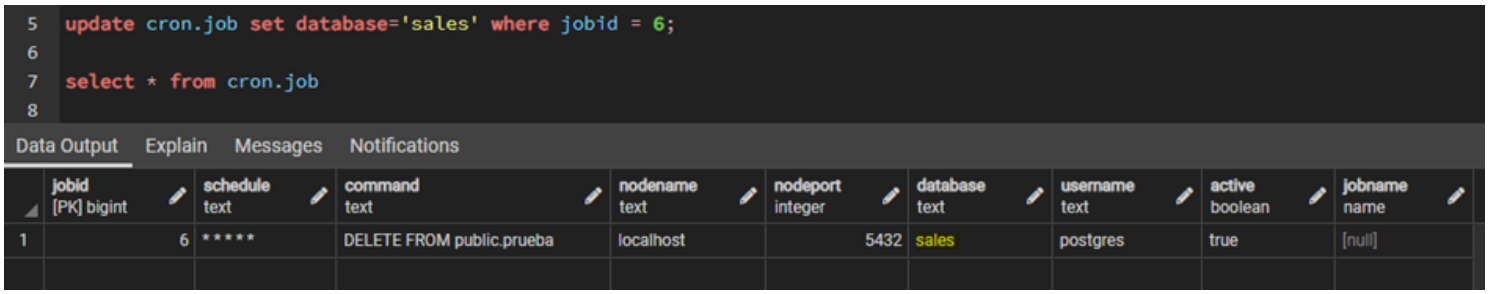
```
1  SELECT cron.schedule('* * * * *', $$DELETE FROM public.prueba$$);
2
3  select * from cron.job;
4
```

Data Output | Explain | Messages | Notifications

	jobid [PK] bigint	schedule text	command text	nodename text	nodeport integer	database text	username text	active boolean	jobname name
1	6	*****	DELETE FROM public.prueba	localhost	5432	postgres	postgres	true	[null]

then you can change de database using something like:

```
update cron.job set database='sales' where jobid = 6;
```



```
5  update cron.job set database='sales' where jobid = 6;
6
7  select * from cron.job
8
```

Data Output | Explain | Messages | Notifications

	jobid [PK] bigint	schedule text	command text	nodename text	nodeport integer	database text	username text	active boolean	jobname name
1	6	*****	DELETE FROM public.prueba	localhost	5432	sales	postgres	true	[null]

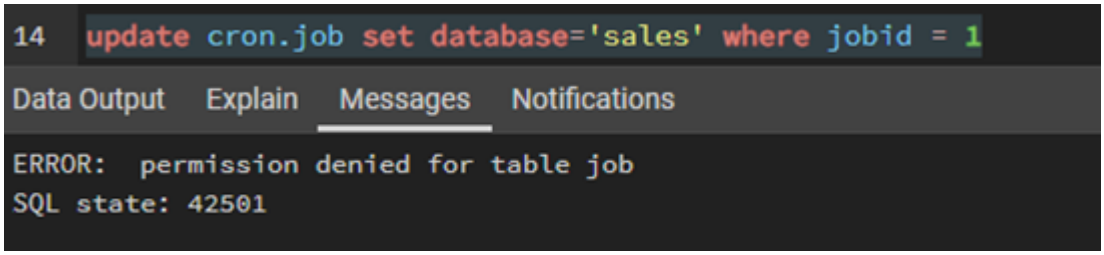
In the image below, you can see the failed execution on the *postgres* database as the table *public.prueba* does not exists there, but it works in *sales* database where the table exists

```
9  select * from cron.job_run_details where runid > 28
10 order by runid;
11
```

Data Output | Explain | Messages | Notifications

	jobid bigint	runid [PK] bigint	job_pid integer	database text	username text	command text	status text	return_message text
1	6	29	30495	postgres	postgres	DELETE FROM public.prueba	failed	ERROR: relation "public.prueba" does not exist
2	6	30	30558	sales	postgres	DELETE FROM public.prueba	succeeded	DELETE 0
3	6	31	30622	sales	postgres	DELETE FROM public.prueba	succeeded	DELETE 0

In Azure Database for PostgreSQL Flexible Server, if you try to update the database you will get an error as it requires superuser privileges, as shown in the image below:



NOTE: consider that PostgreSQL on Amazon RDS allows to update the cron.job table as described at https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/PostgreSQL_pg_cron.html, and a customer could compare both services.

Workarounds

So far there are two workarounds, but it is necessary to work with cx to see if they are viable solutions for their specific needs:

Workaround 1

Update the server parameter cron.database_name, however, if cx needs to schedule jobs to run command in different databases this is not viable

Workaround 2

Use FDW on the database set in parameter cron.database_name (where the jobs run) and make the command run by job to use the foreign table.

For example: a job is needed to delete old data from *public.sales* on db *sales*. A Foreign Table can be created on the db *postgres*

In the example, the Flexible Server name is *pgfs12* and the admin user is *admin2*

- On the database *postgres*, create the Foreign Table:

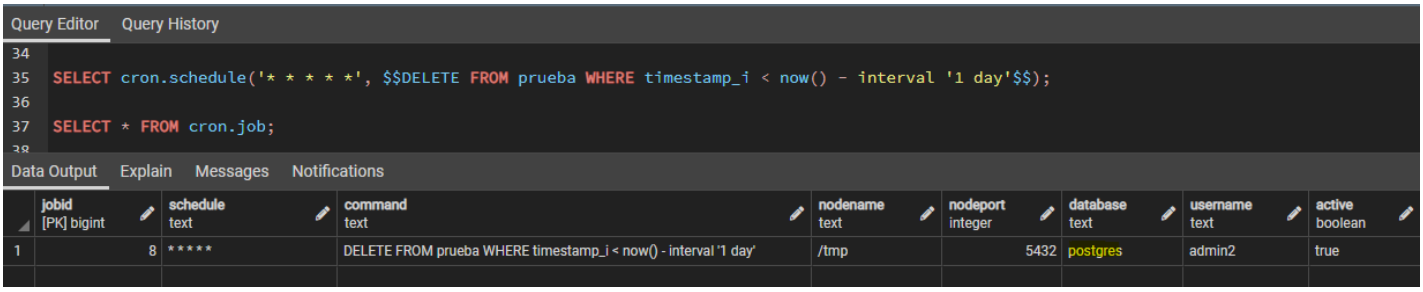
```
CREATE EXTENSION postgres_fdw;

CREATE SERVER pgfs12_sales FOREIGN DATA WRAPPER postgres_fdw OPTIONS (host
'pgfs12.postgres.database.azure.com',dbname 'sales', port '5432');

CREATE USER MAPPING for admin2 SERVER pgfs12_sales OPTIONS (user 'admin2', password 'SuperStrongPassword1

CREATE FOREIGN TABLE prueba
(
   id SERIAL,
   myname character varying(20) ,
   lastname character varying(20) ,
   isvalid boolean,
   timestamp_i timestamp without time zone
) SERVER pgfs12_sales;
```

NOTE: Even if the server is created using Private Access (VNET integration), the FDW can use the FQDN of the server and it will work.

- Now you can create a job that runs on db *postgresql*

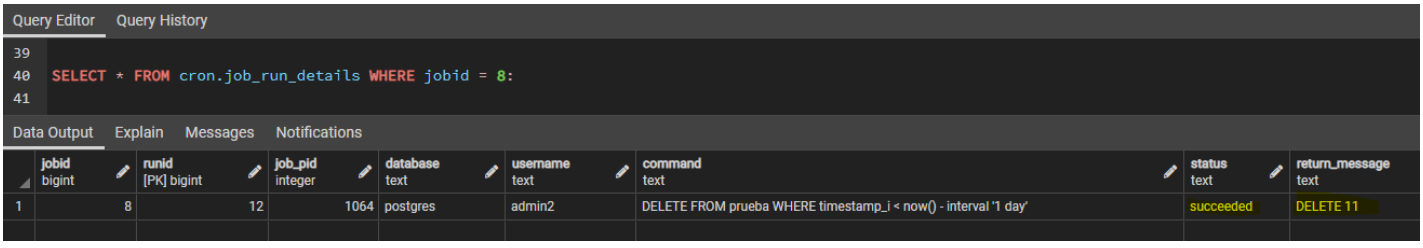
```
SELECT cron.schedule('* * * * *', $$DELETE FROM a WHERE timestamp_i < now() - interval '1 day'$$);

SELECT * FROM cron.job;
```

In the image bellow, you can see the job is created in the *postgres* database but it will delete data from the Foreign Table



- Check the job history and see that the rows were deleted

```
SELECT * FROM cron.job_run_details WHERE jobid = 8
```



the rows where deleted on table *public.prueba* on database *sales*!!!!!!!

Final solution

A function named cron.alter_job will be added to pg_cron to allow job modifications without updating the cron.job table.

September 2021 update: The function cron.alter_job was released in pg_Cron, but there is no ETA for its immplementation in Azure Database for PostgreSQL Flexible Server

You can get information about the cron.alter_job function at https://github.com/citusdata/pg_cron/pull/120