How to change DB Table Primary key using Connect SQL Rule

Step 1.

Before you drop and recreate a primary key, you should first identify the exact name of the primary key constraint, because PostgreSQL automatically generates constraint names if you didn't specify one during creation.

If you skip this step and just guess the name, your ALTER TABLE ... DROP CONSTRAINT ... will fail.

Here's how to find it:

Open Query Runner and run this query (Configure -> system -> database ->QueryRunner)

```
SELECT conname AS constraint_name
FROM pg_constraint AS con
INNER  JOIN pg_class AS t ON con.conrelid = t.oid
INNER  JOIN pg_namespace AS n ON n.oid = t.relnamespace
WHERE c.contype = 'p'
AND t.relname = 'your_table_name'
AND n.nspname = 'your_schema_name';
```

Both query's are ok ...

```
SELECT con.*
FROM pg_catalog.pg_constraint con
INNER JOIN pg_catalog.pg_class rel ON rel.oid = con.conrelid
INNER JOIN pg_catalog.pg_namespace nsp ON nsp.oid = connamspace
WHERE con.contype = 'p' AND rel.relname = '<table_Name>';
```

erandamweerasinghe@gmail.com

Explanation

- contype = 'p' → only primary key constraints
- relname → your table name
- nspname → your schema name (e.g., public)

You're working with PostgreSQL system catalog tables (pg_catalog schema).

- **pg_constraint** → stores info about constraints (PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK, etc.).
- **pg_class** → stores info about database objects like tables, indexes, sequences, etc.

Pg_class : PostgreSQL: Documentation: 17: 51.11. pg_class

The join logic

```
INNER JOIN pg_catalog.pg_class rel
  ON rel.oid = con.conrelid
```

Here's what's happening:

- con.conrelid in pg_constraint → the OID (object ID) of the table that the constraint belongs to.
- rel.oid in pg_class → the OID that identifies each table (or other database object) in PostgreSQL.

Why join them?
Because pg_constraint only stores the table ID (numeric OID), not the table name.
To get the actual table name, you must join to pg_class (which has the relname column for table name).

1 The catalog tables involved

- pg_namespace → stores all schema names in the database (like public, information_schema, etc.).

- pg_constraint → has a column connamespace which stores the OID of the schema where the constraint is defined.

Pg_constraint : PostgreSQL: Documentation: 17: 51.13. pg_constraint

2️⃣ The join

```
INNER JOIN pg_catalog.pg_namespace nsp
  ON nsp.oid = con.connamespace
```

- con.connamespace → the schema ID where the constraint exists.
- nsp.oid → the actual ID of the schema in pg_namespace.

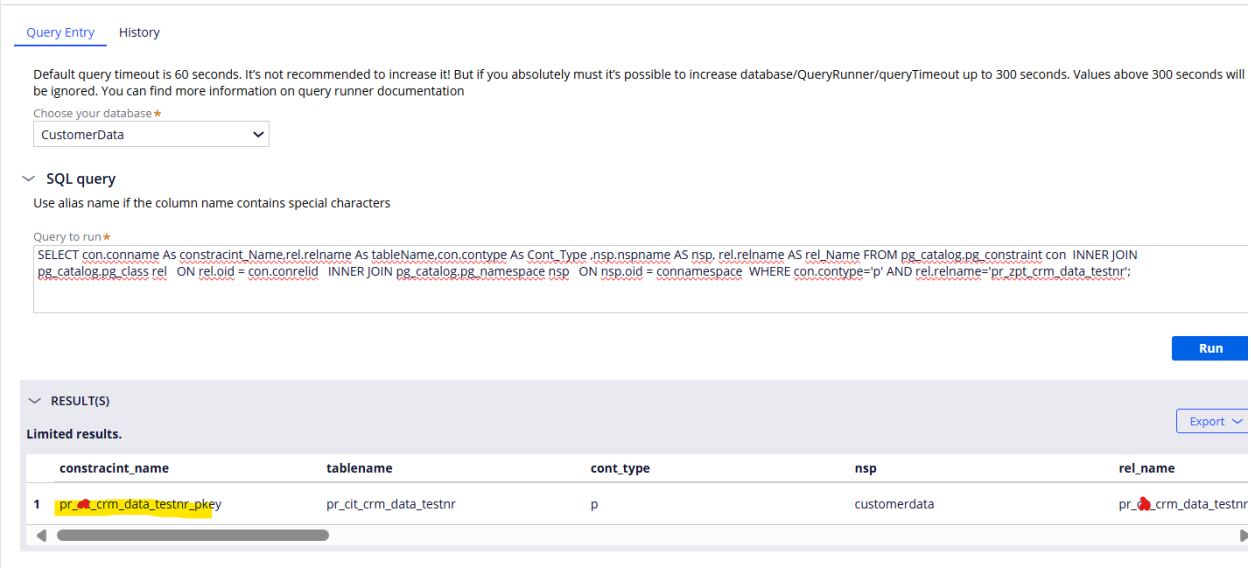
By joining, we can get nsp.nspname (schema name) instead of just seeing a numeric OID.

3️⃣ Why needed

pg_constraint doesn't store the schema name directly — it only keeps the ID (connamespace).
Joining to pg_namespace lets you display the schema name in a readable form.

Pg_namespace : PostgreSQL: Documentation: 17: 51.32. pg_namespace

- conname → this is the exact primary key

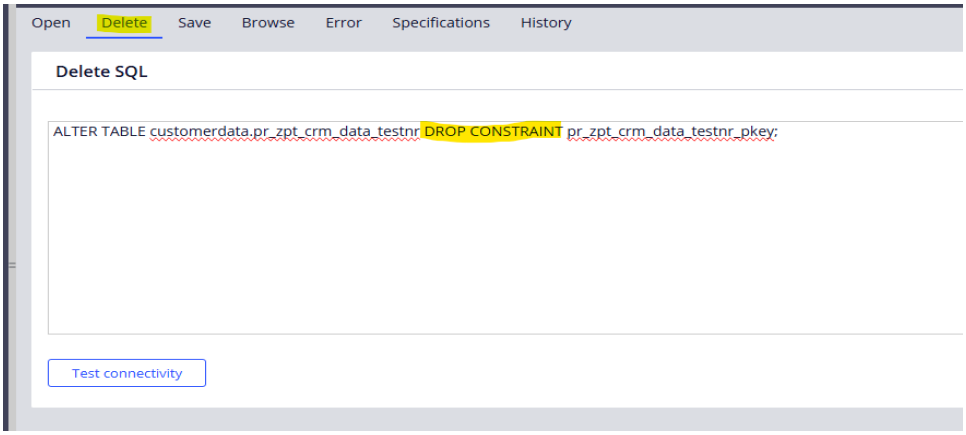


Keep exact constrain key.

Step 2

- Create new ConnectSQL rule and open delete tab.
- Put this Command on the delete Tab (remove existing primary key constraint)

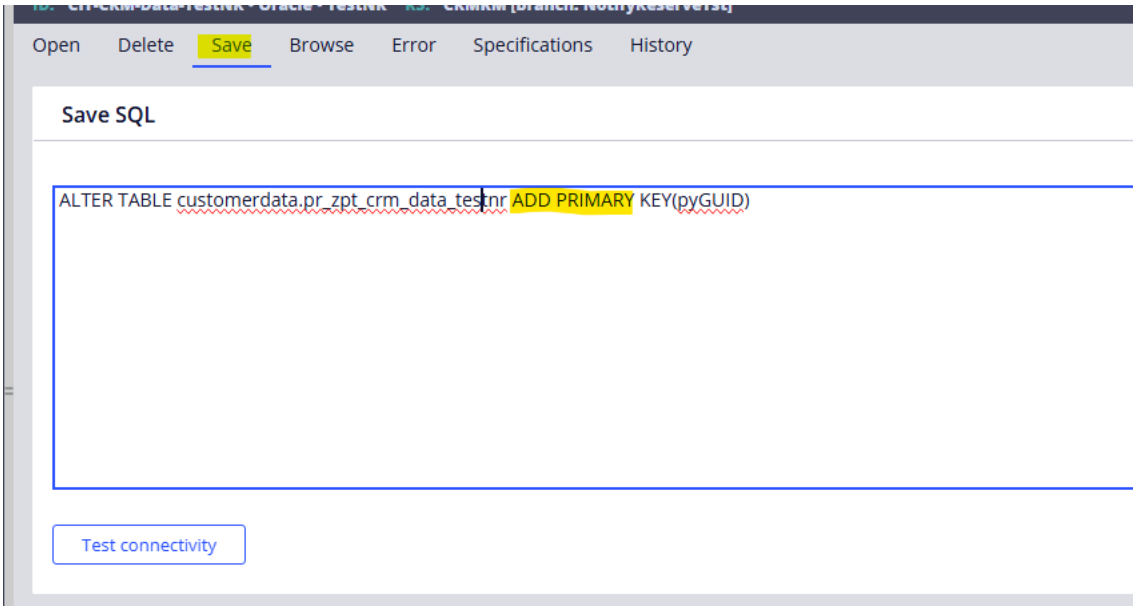
  ALTER TABLE <databaseName>.<tableName> DROP CONSTRAINT <PrimaryKeyConstaint>



erandamweerasinghe@gmail.com

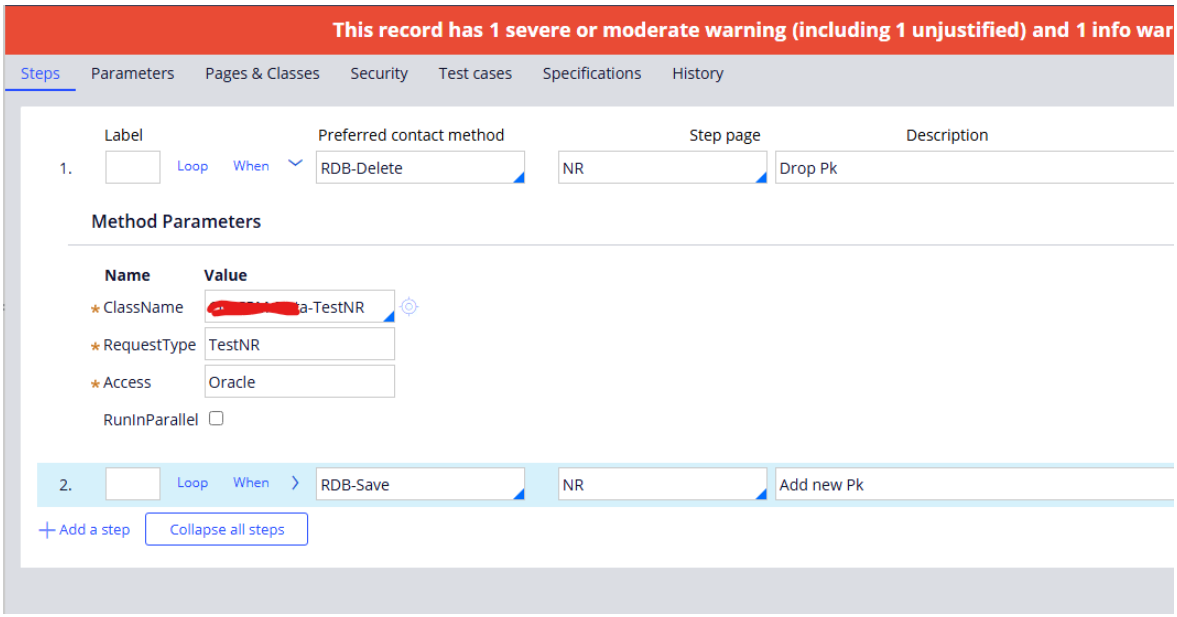
- Put this command for save Tab (add new primary key to table)

  ALTER TABLE <databaseName>.<tableName> ADD PRIMARY (<Your primary key Name>)



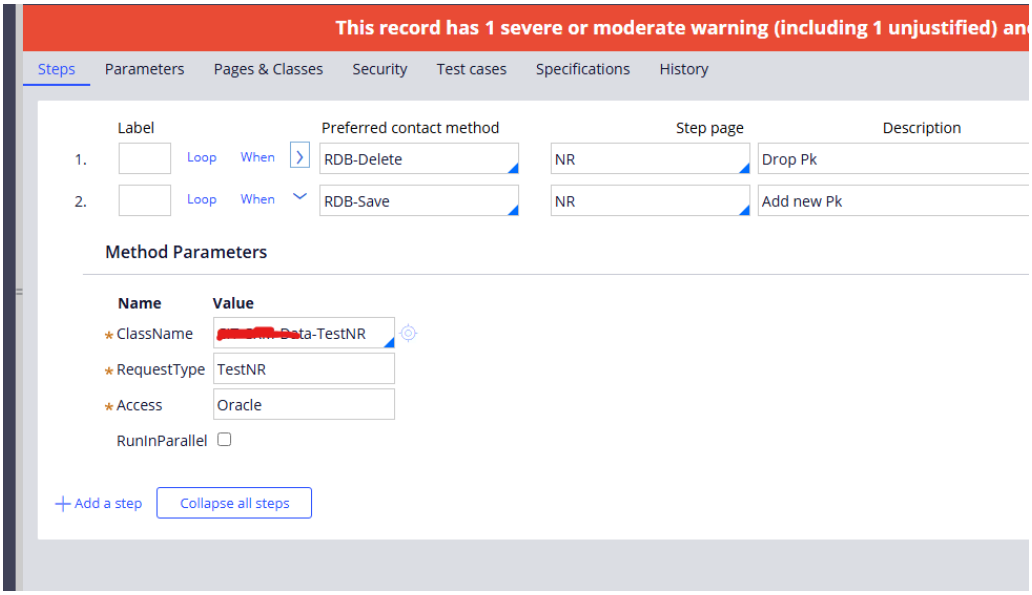
Step 3

- Create activity and add **RDB-Delete** method for Drop pk constraint
- Add RDB-Save method for add new pk constraint
  - RequestType -->created ConnectSQL rule name



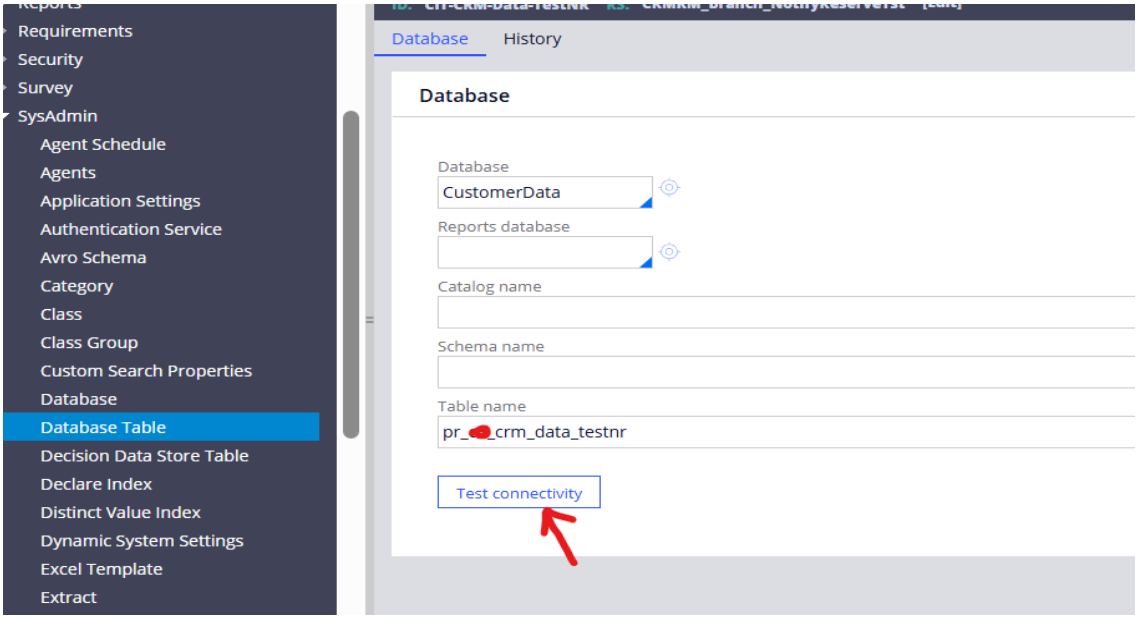
- Comment RDB-Save method and Run Activity, then our pk constraint will drop. You can verify by running bellow query on the query runner. you will see Empty result(s).

- Comment RDB-Delete method and uncomment RDB-Save method , run activity. Then our new pk will added to table.



Open Database Table and click test connectivity button

erandamweerasinghe@gmail.com



You will see your table pk has changed

data_testnr in database CustomerData , but it contains a px, py, or pz property: pxCommitDateTime
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxCreateDateTime
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxCreateOperator
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxCreateOpName
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxCreateSystemID
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxSaveDateTime
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxUpdateDateTime
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxUpdateOperator
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxUpdateOpName
data_testnr in database CustomerData , but it contains a px, py, or pz property: pxUpdateSystemID
the primary key columns for external data table customerdata.pr_cit_crm_data_testnr in database CustomerData ; the class keys are [TestNR_ID] , but the table's primary keys are [pega.customerdata.pyguid]

..... Happy Learning .....

erandamweerasinghe@gmail.com