

Quick Links



[Online Demo](#)



[Download](#)



[FAQ](#)



[Latest Docs](#)



[Ask a
Question](#)



[Report a Bug](#)



[Screenshots](#)

This page in other versions: [Latest \(4.19\)](#) | [4.18](#) | [4.17](#) | [3.6](#) | [2.1](#) | [1.6](#) | [Development](#)

[pgAdmin 4 4.19 documentation](#) » [Creating or Modifying a Table](#) » [previous](#) | [next](#) | [index](#)

Warning: This documentation is for a pre-release version of pgAdmin 4

Contents

- [Rule Dialog](#)
 - [Example](#)
- [Getting Started](#)
- [Connecting To A Server](#)
- [Managing Cluster Objects](#)
- [Managing Database Objects](#)
- [Creating or Modifying a Table](#)
 - [Check Dialog](#)
 - [Column Dialog](#)
 - [Compound Trigger Dialog](#)
 - [Exclusion Constraint Dialog](#)
 - [Foreign key Dialog](#)
 - [Index Dialog](#)
 - [Primary key Dialog](#)
 - [Rule Dialog](#)
 - [Table Dialog](#)
 - [Trigger Dialog](#)
 - [Unique Constraint Dialog](#)
- [Management Basics](#)
- [Backup and Restore](#)
- [Developer Tools](#)
- [pgAgent](#)
- [pgAdmin Project Contributions](#)
- [Release Notes](#)
- [Licence](#)

Rule Dialog ¶

Use the *Rule* dialog to define or modify a rule for a specified table or view. A PostgreSQL rule allows you to define an additional action that will be performed when a SELECT, INSERT, UPDATE, or DELETE is performed against a table.

The *Rule* dialog organizes the development of a rule through the *General*, *Definition*, *Condition*, *Commands* tabs. The *SQL* tab displays the SQL code generated by dialog selections.

The screenshot shows the 'Create - rule' dialog box with the 'General' tab selected. The 'Name' field contains 'replace_expression_answer' and the 'Comment' field contains 'Rule to replace expression answer'. The dialog has tabs for 'General', 'Definition', 'Condition', 'Commands', and 'SQL'. At the bottom, there are buttons for 'Cancel', 'Reset', and 'Save', along with information and help icons.

Use the fields in the *General* tab to identify the rule:

- Use the *Name* field to add a descriptive name for the rule. The name will be displayed in the *pgAdmin* tree control. Multiple rules on the same table are applied in alphabetical name order.
- Store notes about the rule in the *Comment* field.

Click the *Definition* tab to continue.

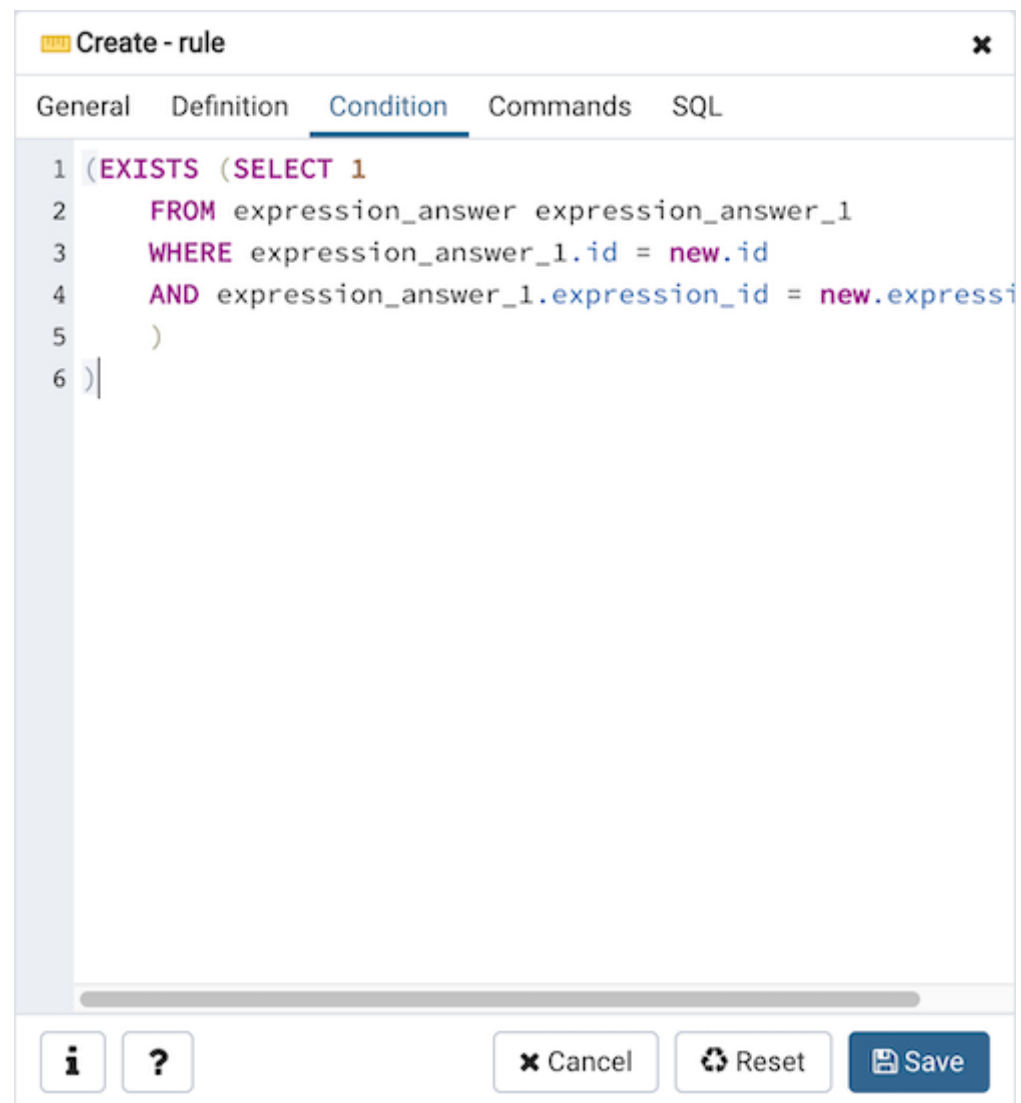
The screenshot shows the 'Create - rule' dialog box with the 'Definition' tab selected. The 'Event' dropdown menu is set to 'Insert' and the 'Do instead?' checkbox is checked. The dialog has tabs for 'General', 'Definition', 'Condition', 'Commands', and 'SQL'. At the bottom, there are buttons for 'Cancel', 'Reset', and 'Save', along with information and help icons.

Use the fields in the *Definition* tab to write parameters:

- Click inside the *Event* field to select the type of event that will invoke the rule; event may be *Select*, *Insert*, *Update*, or *Delete*.

- Move the *Do Instead* switch to *Yes* indicate that the commands should be executed instead of the original command; if *Do Instead* specifies *No*, the rule will be invoked in addition to the original command.

Click the *Condition* tab to continue.



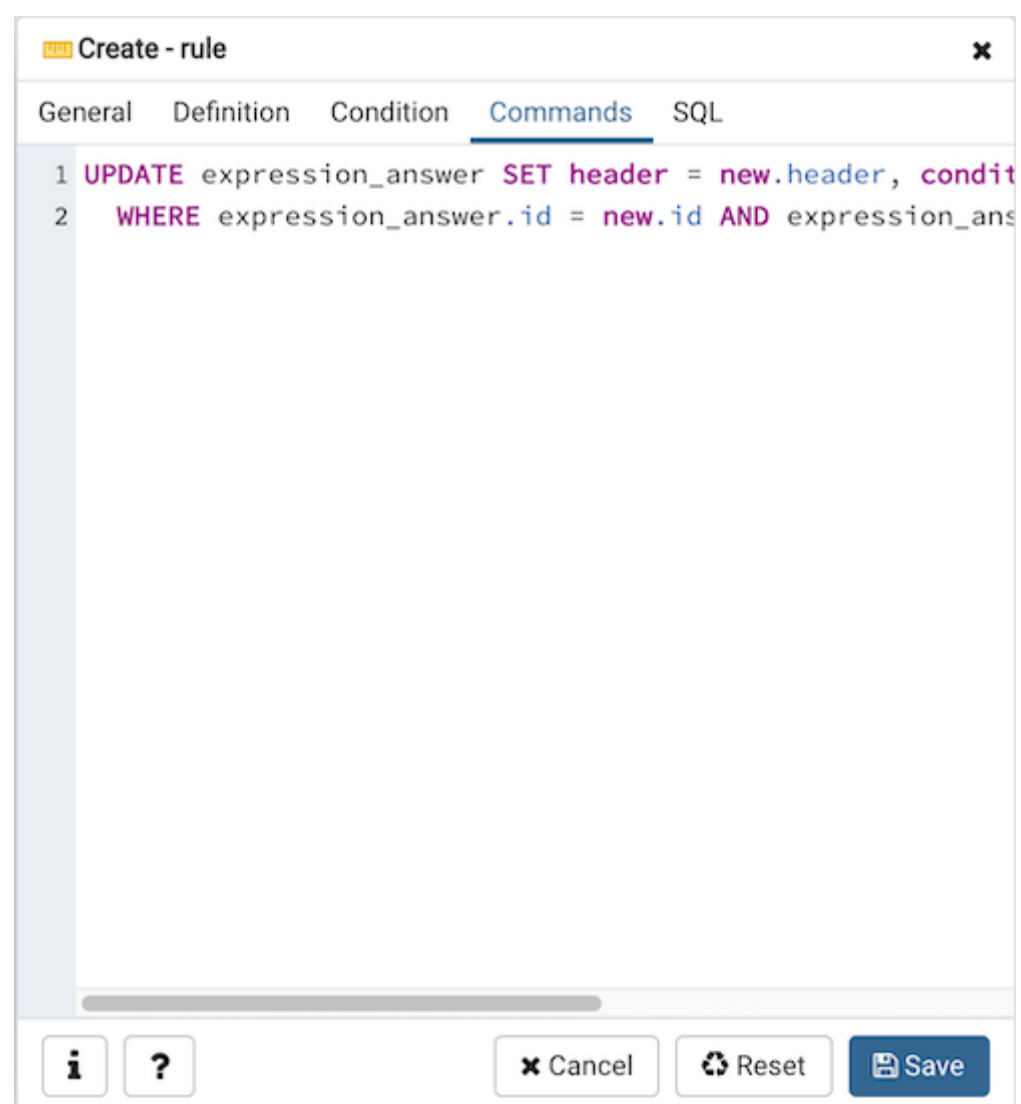
The screenshot shows a 'Create - rule' dialog with a close button (X) in the top right. It has five tabs: 'General', 'Definition', 'Condition' (which is selected and underlined), 'Commands', and 'SQL'. The 'Condition' tab contains a SQL query editor with the following text:

```
1 (EXISTS (SELECT 1
2   FROM expression_answer expression_answer_1
3   WHERE expression_answer_1.id = new.id
4   AND expression_answer_1.expression_id = new.expression_id
5 )
6 )|
```

At the bottom of the dialog, there are three buttons: an information icon (i), a question mark icon (?), and a 'Cancel' button. To the right of these are 'Reset' and 'Save' buttons.

Specify a SQL conditional expression that returns a boolean value in the editor.

Click the *Commands* tab to continue.



The screenshot shows the same 'Create - rule' dialog, but now the 'Commands' tab is selected and underlined. The 'SQL' tab is also visible. The 'Commands' tab contains a SQL query editor with the following text:

```
1 UPDATE expression_answer SET header = new.header, condition_id = new.condition_id
2 WHERE expression_answer.id = new.id AND expression_answer.condition_id = new.condition_id
```

At the bottom of the dialog, there are three buttons: an information icon (i), a question mark icon (?), and a 'Cancel' button. To the right of these are 'Reset' and 'Save' buttons.

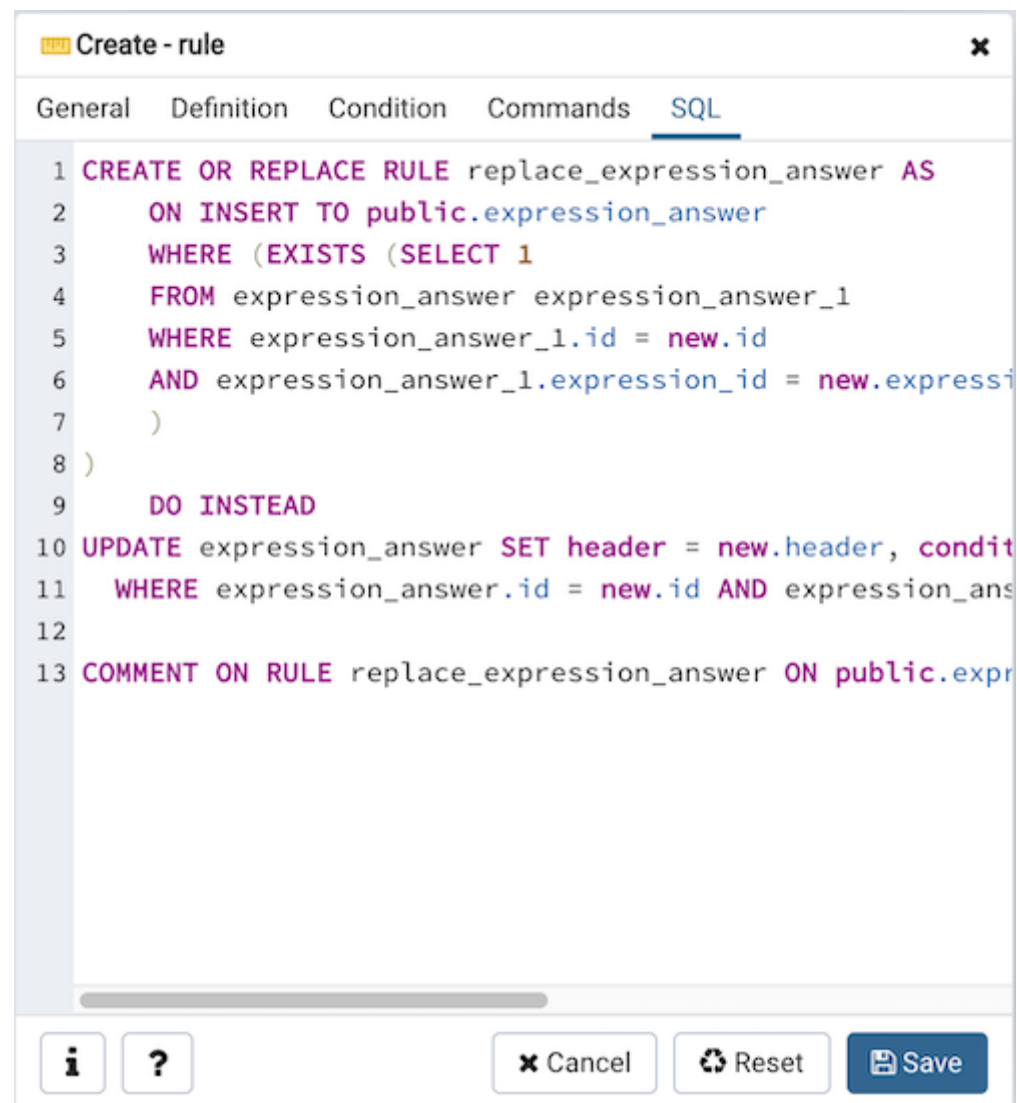
Provide a command in the editor that defines the action performed by the rule.

Click the SQL tab to continue.

Your entries in the *Rule* dialog generate a SQL command (see an example below). Use the SQL tab for review; revisit or switch tabs to make any changes to the SQL command.

Example ¶

The following is an example of the sql command generated by user selections in the *Rule* dialog:



```
1 CREATE OR REPLACE RULE replace_expression_answer AS
2   ON INSERT TO public.expression_answer
3   WHERE (EXISTS (SELECT 1
4     FROM expression_answer expression_answer_1
5     WHERE expression_answer_1.id = new.id
6     AND expression_answer_1.expression_id = new.expression_id
7   )
8 )
9   DO INSTEAD
10  UPDATE expression_answer SET header = new.header, condition = new.condition
11  WHERE expression_answer.id = new.id AND expression_answer.expression_id = new.expression_id
12
13 COMMENT ON RULE replace_expression_answer ON public.expression_answer;
```

The example sends a notification when an UPDATE executes against a table.

- Click the *Info* button (i) to access online help.
- Click the *Save* button to save work.
- Click the *Cancel* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.