
Minerva Documentation

Release 4.6.62

Hendrikx ITC

September 23, 2015

1	Introduction	3
2	User Guide	5
2.1	How To	5
3	Schema Reference	11
3.1	attribute	11
3.2	attribute_base	11
3.3	attribute_directory	12
3.4	attribute_history	30
3.5	attribute_staging	30
3.6	dimension	31
3.7	directory	34
3.8	entity_tag	44
3.9	materialization	46
3.10	notification	60
3.11	olap	65
3.12	public	65
3.13	relation	161
3.14	relation_def	164
3.15	system	165
3.16	trend	168
3.17	trigger	188
3.18	trigger_rule	202
3.19	virtual_entity	202
4	Indices and tables	203

Contents:

Introduction

Minerva

2.1 How To

Examples are executed on a psql prompt.

2.1.1 Create Entities

Before any data is stored, entities have to be created to which the data belongs. Basic information about entities is stored in the tables *entitytype* and *entity*. These tables are automatically populated when the appropriate functions are used to create the entities.

Create an entity using *create_entity*:

```
minerva=# select directory.create_entity('Node=root');
               create_entity
-----
(1,"2015-09-07 12:21:01.594337+00",root,2,Node=root,)
```

The *directory.create_entity* function returns a value of type *entity* and this is exactly the record you will find in the table *entity*:

```
minerva=# select * from directory.entity where id = 1;
 id | first_appearance | name | entitytype_id | dn | parent_id
-----+-----+-----+-----+-----+-----
  1 | 2015-09-07 12:21:01.594337+00 | root |          2 | Node=root |
```

The function *create_entity* will also define new entitytypes if required. So the previous example will have resulted in a new record in the *entitytype* table:

```
minerva=# select * from directory.entitytype where id = 2;
 id | name | description
-----+-----+-----
  2 | Node |
```

Any required intermediate entities are also automatically created:

```
minerva=# select directory.create_entity('Node=root,Slot=c1,Port=12');
               create_entity
-----
(2,"2015-09-07 13:10:11.809796+00",12,3,"Node=root,Slot=c1,Port=12",3)

minerva=# select * from directory.entity;
```

id	first_appearance	name	entitytype_id	dn	parent_id
1	2015-09-07 12:21:01.594337+00	root	2	Node=root	
3	2015-09-07 13:10:11.809796+00	c1	4	Node=root,Slot=c1	1
2	2015-09-07 13:10:11.809796+00	12	3	Node=root,Slot=c1,Port=12	3

The intermediate 'Node=root,Slot=c1' entity and its type are automatically created.

2.1.2 Define Data Sources

All data about entities is linked to a data source. Data sources are used to organize different sets of data for potentially overlapping sets of entities. This solves the problem of having conflicting facts about entities when they have the same name, but come from different sources and have different values and meanings.

To create a data source, use the function `create_datasource`:

```
minerva=# select directory.create_datasource('network-conf');
               create_datasource
-----
(2,network-conf,default,UTC)
```

The function returns a value of type `datasource`, and is the record inserted into the `datasource` table:

```
minerva=# select * from directory.datasource where id = 2;
 id |      name      | description | timezone
-----+-----+-----+-----
  2 | network-conf | default    | UTC
```

2.1.3 Store Attributes

To store attributes of entities, you have to create one or more attribute stores. One attribute store can hold data for exactly one entity type of one data source. What data an attribute store can hold is reflected in the name: <data source name>_<entity type name>

Create the attribute store

Create an attribute store to hold data for the entity type 'Port' of data source 'network-conf':

```
minerva=# select attribute_directory.create_attributestore('network-conf', 'Port', ARRAY[('speed', 'i
               create_attributestore
-----
(1,1,1)
```

Like the functions mentioned in the previous sections, this function also returns a value of its corresponding type `attributestore`, which is the record inserted into the table `attributestore`:

```
minerva=# select * from attribute_directory.attributestore where id = 2;
 id | datasource_id | entitytype_id
-----+-----+-----
  1 |             1 |             1
```

Now this record doesn't read as easily as the records seen in the previous sections about entities and data sources because there is no textual component in the `attributestore` record. An easy way to make this more readable is by using the to-text-cast to obtain the 'name' of the attribute store:

```
minerva=# select attributestore::text, * from attribute_directory.attributestore where id = 1;
    attributestore    | id | datasource_id | entitytype_id |
-----+-----+-----+-----+
network-conf_Port | 1  |              1 |              1 |
```

Here you can see the textual representation of the attribute store that is used for naming the corresponding tables, functions and views of the attribute store.

Store attribute data

Now the attribute store is ready to hold data, add an initial value. First insert the data into the staging table:

```
minerva=# insert into attribute_staging."network-conf_Port"(entity_id, timestamp, speed) values (2, r
INSERT 0 1
```

And then transfer the staged data to the history table:

```
minerva=# select attribute_directory.transfer_staged(attributestore) from attribute_directory.attribu
transfer_staged
-----
(1,1,1)

minerva=# select * from attribute_history."network-conf_Port";
 entity_id |          timestamp          | speed |          first_appearance          |          modifi
-----+-----+-----+-----+-----+
          2 | 2015-09-07 14:11:47.768745+00 | 1000  | 2015-09-07 14:14:51.160655+00 | 2015-09-07 14:2
```

It can be difficult to script the insertion of attribute data when the entity Id is not yet known. For this reason, there is a convenience function to lookup the entity by its Distinguished Name, named *dn_to_entity(character varying) -> directory.entity*. This function returns an existing entity or creates a new one and returns that. Now to combine that with adding a new attribute record that updates the ‘current’ state:

```
minerva=# insert into attribute_staging."network-conf_Port"(entity_id, timestamp, speed) values ((di
INSERT 0 1

minerva=# select attribute_directory.transfer_staged(attributestore) from attribute_directory.attribu
transfer_staged
-----
(2,2,3)

minerva=# select * from attribute_history."network-conf_Port";
 entity_id |          timestamp          | speed |          first_appearance          |          modifi
-----+-----+-----+-----+-----+
          2 | 2015-09-07 14:11:47.768745+00 | 1000  | 2015-09-07 14:14:51.160655+00 | 2015-09-07 14:1
          2 | 2015-09-07 14:27:13.738692+00 | 5000  | 2015-09-07 14:27:18.066607+00 | 2015-09-07 14:2
```

2.1.4 Store Trends

Create the trend store

Create a trend store to hold data for the entity type ‘Port’ of data source ‘network-measurements’ with a granularity of 15 minutes:

```
minerva=# select trend.create_trendstore('network-measurements', 'Port', '900', ARRAY[(('bytes_transf
create_trendstore
```

```
-----
(1,3,3,900,21600,table,4,"1 mon")
-----
```

The return value is of type *trend.trendstore* and holds the record inserted into the *trend.trendstore* table:

```
minerva=# select * from trend.trendstore where id = 1;
 id | entitytype_id | datasource_id | granularity | partition_size | type  | version | retention_per
-----+-----+-----+-----+-----+-----+-----+-----
  1 |             3 |             3 |         900 |         21600 | table |        4 | 1 mon
```

By default, a trend store with a granularity of 900 seconds is partitioned into tables with a partition size of 6 hours (21600 seconds) and has a data retention period of 1 month.

2.1.5 Materialize Trend Data

To be documented

2.1.6 Define Triggers

Triggers are defined in a number of steps:

1. Create a function that returns records with all KPI's, measurement values, etc that are needed to calculate the notifications for a specific timestamp.
2. Create the new rule with it's name and thresholds.
3. Define the actual rule in the form of a where-clause.
4. Set the threshold values for the defined thresholds in step 1.
5. Define the weighing function.
6. Define the notification details text.

Here, we will work out an example trigger named 'high_traffic'.

Create KPI function

This function should return all data that is required for the actual rule and weight calculation. Before we can create this function, we need to define a return record type:

```
CREATE TYPE trigger_rule.high_traffic_kpi (
    entity_id integer,
    timestamp timestamp with time zone,
    traffic bigint
);
```

Now we can create the actual function:

```
CREATE FUNCTION trigger_rule.high_traffic_kpi(timestamp with time zone)
    RETURNS SETOF trigger_rule.high_traffic_kpi
AS $$
BEGIN
    RETURNS QUERY EXECUTE $query$
    SELECT entity_id, timestamp, traffic
    FROM trend."network-measurements_Port_qtr"
    WHERE timestamp = $1
    $query$ USING $1;
```

```
END;
$$ LANGUAGE plpgsql STABLE;
```

Create New Rule

The following statement defines the rule in the system.

```
SELECT trigger.create_rule(
    'high_traffic',
    ARRAY[
        ('max_traffic', 'bigint')
    ]::trigger.threshold_def[]
);
```

Define Rule Condition

Now set the actual condition of the rule that defines when a record results in a notification:

```
SELECT trigger.set_condition(
    'high_traffic',
    $predicate$
    traffic > max_traffic
    $predicate$
);
```

Set Threshold Values

Set the threshold values for the condition. The arguments are in the same order as defined in the create_rule statement:

```
SELECT trigger_rule."high_traffic_set_thresholds"(50000);
```

Define Weighing Function

The logic for determining the weight of a notification can be as complex as you want, or as simple as a constant value. Here we choose a simple case selection:

```
SELECT trigger.set_weight(
    'high_traffic',
    $query$
    SELECT CASE
    WHEN $1.traffic > 1000000 THEN
        100
    WHEN $1.traffic > 500000 THEN
        50
    ELSE
        10
    END
    $query$
);
```

Define Notification Details

```
SELECT trigger.define_notification(  
    'high_traffic',  
    $details$  
    SELECT format('traffic(%s) > max_traffic(%s)', $1.traffic, $1.max_traffic);  
    $details$  
);
```

Schema Reference

3.1 attribute

Contains views pointing to current attribute records

3.1.1 Types

3.1.2 Tables

3.1.3 Views

3.1.4 Functions

Name	Return Type	Description

3.2 attribute_base

Contains the parent/base tables for attribute store data tables

3.2.1 Types

3.2.2 Tables

3.2.3 Views

3.2.4 Functions

Name	Return Type	Description

3.3 attribute_directory

Contains a directory with attribute store meta data

3.3.1 Types

attribute_descr

Name	Type	Description
name	char[]	
datatype	character varying	
description	text	

attribute_info

Name	Type	Description
name	char[]	
data_type	character varying	

source_modified

Name	Type	Description
source_name	text	
modified	timestamp with time zone	

3.3.2 Tables

attribute

Name	Type	Description
attributestore_id	integer	
description	text	
name	char[]	
datatype	character varying	
id	integer	

attribute_tag_link

Name	Type	Description
attribute_id	integer	
tag_id	integer	

attributestore

Name	Type	Description
datasource_id	integer	
entitytype_id	integer	
id	integer	

attributestore_compacted

Name	Type	Description
attributestore_id	integer	
compacted	timestamp with time zone	

attributestore_curr_materialized

Name	Type	Description
attributestore_id	integer	
materialized	timestamp with time zone	

attributestore_materialization

Name	Type	Description
attributestore_id	integer	
last_modified	timestamp with time zone	

attributestore_modified

Name	Type	Description
attributestore_id	integer	
modified	timestamp with time zone	

sampled_view_materialization

Name	Type	Description
attributestore_id	integer	
view_class	oid	
source_modified_proc	oid	
id	integer	
stability_delay	interval	

sampled_view_materialization_state

Name	Type	Description
materialization_id	integer	
fingerprint	text	

3.3.3 Views

dependencies

Name	Type	Description
src	char[]	
column_name	char[]	
dst	char[]	

sampld_view_materialization_runnable

Name	Type	Description
id	integer	
attributestore_id	integer	
view_class	oid	
source_modified_proc	oid	
stability_delay	interval	

3.3.4 Functions

Name	Retu
<i>add_attribute(attribute_directory.attributestore, name char[], datatype character varying, description text)</i>	attri
<i>add_attribute_column(attribute_directory.attributestore, char[], text)</i>	attri
<i>add_attributes(attribute_directory.attributestore, attributes attribute_descr[])</i>	attri
<i>add_first_appearance_to_attribute_table(attribute_directory.attributestore)</i>	attri
<i>at_function_name(attribute_directory.attributestore)</i>	char
<i>at_ptr_function_name(attribute_directory.attributestore)</i>	char
<i>attributestore_fingerprint(attributestore_name text)</i>	text
<i>attributestore_modified(attributestore_name text)</i>	time
<i>changes_view_name(attribute_directory.attributestore)</i>	char
<i>changes_view_query(attribute_directory.attributestore)</i>	text
<i>check_attribute_types(attribute[])</i>	attri
<i>check_attributes_exist(attribute[])</i>	attri
<i>clean(store attribute_directory.attributestore, ts timestamp with time zone)</i>	bigi
<i>cleanup_attribute_after_delete()</i>	trigg
<i>cleanup_attributestore_on_delete()</i>	trigg
<i>cleanup_on_datasource_delete()</i>	trigg
<i>compact(attribute_directory.attributestore)</i>	attri
<i>compactd_tmp_table_name(attribute_directory.attributestore)</i>	char
<i>compactd_view_name(attribute_directory.attributestore)</i>	char
<i>compactd_view_query(attribute_directory.attributestore)</i>	text
<i>create_at_func(attribute_directory.attributestore)</i>	attri
<i>create_at_func_ptr(attribute_directory.attributestore)</i>	attri
<i>create_at_func_ptr_sql(attribute_directory.attributestore)</i>	text
<i>create_attributestore(datasource_name text, entitytype_name text)</i>	attri
<i>create_attributestore(datasource_name text, entitytype_name text, attributes attribute_descr[])</i>	attri
<i>create_attributestore_from_view(oid, datasource_name text, entitytype_name text)</i>	attri
<i>create_base_table(attribute_directory.attributestore)</i>	attri
<i>create_changes_view(attribute_directory.attributestore)</i>	attri

Name	Retu
<i>create_changes_view_sql(attribute_directory.attributestore)</i>	text[
<i>create_compacted_tmp_table(attribute_directory.attributestore)</i>	attri
<i>create_compacted_tmp_table_sql(attribute_directory.attributestore)</i>	text[
<i>create_compacted_view(attribute_directory.attributestore)</i>	attri
<i>create_compacted_view_sql(attribute_directory.attributestore)</i>	text[
<i>create_curr_ptr_table(attribute_directory.attributestore)</i>	attri
<i>create_curr_ptr_table_sql(attribute_directory.attributestore)</i>	text[
<i>create_curr_ptr_view(attribute_directory.attributestore)</i>	attri
<i>create_curr_ptr_view_sql(attribute_directory.attributestore)</i>	text[
<i>create_curr_view(attribute_directory.attributestore)</i>	attri
<i>create_curr_view_sql(attribute_directory.attributestore)</i>	text[
<i>create_dependees(attribute_directory.attributestore)</i>	attri
<i>create_entity_at_func(attribute_directory.attributestore)</i>	attri
<i>create_entity_at_func_ptr(attribute_directory.attributestore)</i>	attri
<i>create_entity_at_func_ptr_sql(attribute_directory.attributestore)</i>	text[
<i>create_entity_at_func_sql(attribute_directory.attributestore)</i>	text[
<i>create_hash_function(attribute_directory.attributestore)</i>	attri
<i>create_hash_function_sql(attribute_directory.attributestore)</i>	text[
<i>create_hash_triggers(attribute_directory.attributestore)</i>	attri
<i>create_hash_triggers_sql(attribute_directory.attributestore)</i>	text[
<i>create_history_table(attribute_directory.attributestore)</i>	attri
<i>create_modified_trigger_function(attribute_directory.attributestore)</i>	attri
<i>create_modified_trigger_function_sql(attribute_directory.attributestore)</i>	text[
<i>create_modified_triggers(attribute_directory.attributestore)</i>	attri
<i>create_modified_triggers_sql(attribute_directory.attributestore)</i>	text[
<i>create_run_length_view(attribute_directory.attributestore)</i>	attri
<i>create_run_length_view_sql(attribute_directory.attributestore)</i>	text[
<i>create_sampled_view_materialization(view_class oid, source_modified_proc oid, datasource_name text, entitytype_name text)</i>	attri
<i>create_staging_modified_view(attribute_directory.attributestore)</i>	attri
<i>create_staging_modified_view_sql(attribute_directory.attributestore)</i>	text[
<i>create_staging_new_view(attribute_directory.attributestore)</i>	attri
<i>create_staging_new_view_sql(attribute_directory.attributestore)</i>	text[
<i>create_staging_table(attribute_directory.attributestore)</i>	attri
<i>create_staging_table_sql(attribute_directory.attributestore)</i>	text[
<i>curr_ptr_table_name(attribute_directory.attributestore)</i>	char
<i>curr_ptr_view_name(attribute_directory.attributestore)</i>	char
<i>curr_view_name(attribute_directory.attributestore)</i>	char
<i>curr_view_query(attribute_directory.attributestore)</i>	text
<i>datatype_order(datatype character varying)</i>	integ
<i>define(attribute_directory.attribute)</i>	attri
<i>define_attributestore(datasource_id integer, entitytype_id integer)</i>	attri
<i>define_attributestore(datasource_name text, entitytype_name text)</i>	attri
<i>define_sampled_view_materialization(attributestore_id integer, view_class oid, source_modified_proc oid)</i>	attri
<i>dependees(attribute_directory.attributestore)</i>	obj_
<i>dependers(name char[], level integer)</i>	TAB
<i>dependers(name char[])</i>	TAB
<i>direct_dependers(name text)</i>	char
<i>drop_compacted_view(attribute_directory.attributestore)</i>	attri
<i>drop_compacted_view_sql(attribute_directory.attributestore)</i>	text

Name	Retu
<i>drop_curr_ptr_view(attribute_directory.attributestore)</i>	attri
<i>drop_curr_ptr_view_sql(attribute_directory.attributestore)</i>	char
<i>drop_curr_view(attribute_directory.attributestore)</i>	attri
<i>drop_curr_view_sql(attribute_directory.attributestore)</i>	char
<i>drop_dependees(attribute_directory.attributestore)</i>	attri
<i>drop_entity_at_func(attribute_directory.attributestore)</i>	attri
<i>drop_entity_at_func_sql(attribute_directory.attributestore)</i>	text
<i>drop_hash_function(attribute_directory.attributestore)</i>	attri
<i>drop_staging_modified_view(attribute_directory.attributestore)</i>	attri
<i>drop_staging_modified_view_sql(attribute_directory.attributestore)</i>	char
<i>drop_staging_new_view(attribute_directory.attributestore)</i>	attri
<i>fingerprint(attribute_directory.sampled_view_materialization)</i>	text
<i>get_attribute(attribute_directory.attributestore, char[])</i>	attri
<i>get_attributes(oid)</i>	attri
<i>get_attributes(attribute_directory.attributestore)</i>	attri
<i>get_attributestore(datasource_id integer, entitytype_id integer)</i>	attri
<i>get_attributestore(attribute_directory.sampled_view_materialization)</i>	attri
<i>greatest_datatype(datatype_a character varying, datatype_b character varying)</i>	char
<i>init(attribute_directory.attributestore)</i>	attri
<i>init(attribute_directory.attribute)</i>	attri
<i>insert_state(attribute_directory.sampled_view_materialization, text)</i>	text
<i>mark_compacted(attributestore_id integer, compacted timestamp with time zone)</i>	attri
<i>mark_compacted(attributestore_id integer)</i>	attri
<i>mark_curr_materialized(attributestore_id integer, materialized timestamp with time zone)</i>	attri
<i>mark_curr_materialized(attributestore_id integer)</i>	attri
<i>mark_modified(attributestore_id integer, modified timestamp with time zone)</i>	attri
<i>mark_modified(attributestore_id integer)</i>	attri
<i>materialize(store attribute_directory.attributestore)</i>	attri
<i>materialize(attribute_directory.sampled_view_materialization)</i>	attri
<i>materialize_curr_ptr(attribute_directory.attributestore)</i>	integ
<i>max_modified(attribute_directory.sampled_view_materialization)</i>	time
<i>modify_column_type(attribute_directory.attributestore, column_name char[], datatype character varying)</i>	attri
<i>modify_column_type_sql(attribute_directory.attributestore, column_name char[], datatype character varying)</i>	text[
<i>modify_datatype(attribute_directory.attribute)</i>	attri
<i>remove_attribute(attribute_directory.attribute)</i>	dep_
<i>remove_attribute(attribute_directory.attributestore, char[])</i>	dep_
<i>remove_attribute_column(attribute_directory.attributestore, char[])</i>	attri
<i>render_hash_query(attribute_directory.attributestore)</i>	text
<i>requires_compacting(attributestore_id integer)</i>	bool
<i>requires_compacting(attribute_directory.attributestore)</i>	bool
<i>run_length_view_name(attribute_directory.attributestore)</i>	char
<i>run_length_view_query(attribute_directory.attributestore)</i>	text
<i>sampled_view_materialization_runnable(timestamp with time zone)</i>	attri
<i>set_hash()</i>	trigg
<i>stage_sample(attribute_directory.sampled_view_materialization)</i>	attri
<i>staging_modified_view_name(attribute_directory.attributestore)</i>	char
<i>staging_new_view_name(attribute_directory.attributestore)</i>	char
<i>store_compacted(attributestore_id integer, compacted timestamp with time zone)</i>	attri
<i>store_curr_materialized(attributestore_id integer, materialized timestamp with time zone)</i>	attri

Name	Retu
<i>store_modified(attributestore_id integer, modified timestamp with time zone)</i>	attri
<i>store_state(attribute_directory.sampled_view_materialization, text)</i>	text
<i>store_state(attribute_directory.sampled_view_materialization)</i>	attri
<i>to_attribute(attribute_directory.attribute)</i>	attri
<i>to_attributestore(datasource_id integer, entitytype_id integer)</i>	attri
<i>to_char(attribute_directory.attributestore)</i>	text
<i>to_char(attribute_directory.sampled_view_materialization)</i>	text
<i>to_table_name(attribute_directory.attributestore)</i>	char
<i>transfer_staged(attribute_directory.attributestore)</i>	attri
<i>update_compacted(attributestore_id integer, compacted timestamp with time zone)</i>	attri
<i>update_curr_materialized(attributestore_id integer, materialized timestamp with time zone)</i>	attri
<i>update_datatype_on_change()</i>	trigg
<i>update_modified(attributestore_id integer, modified timestamp with time zone)</i>	attri
<i>update_state(attribute_directory.sampled_view_materialization, text)</i>	text
<i>upgrade_attribute_table(attribute_directory.attributestore)</i>	attri
<i>view_to_attribute_staging_sql(oid, attribute_directory.attributestore)</i>	text

add_attribute(attribute_directory.attributestore, name char[], datatype character varying, description text) -> attribute_directory.attribute

returns: *attribute_directory.attribute*

add_attribute_column(attribute_directory.attributestore, char[], text) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

add_attributes(attribute_directory.attributestore, attributes attribute_descr[]) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

add_first_appearance_to_attribute_table(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

at_function_name(attribute_directory.attributestore) -> char[]

returns: *char[]*

at_ptr_function_name(attribute_directory.attributestore) -> char[]

returns: *char[]*

attributestore_fingerprint(attributestore_name text) -> text

returns: text

Return easily readable fingerprint text containing the name of the attributestore and the modified timestamp

attributestore_modified(attributestore_name text) -> timestamp with time zone

returns: timestamp with time zone

Return modified timestamp of attributestore as recorded in attribute_directory.attributestore_modified table

changes_view_name(attribute_directory.attributestore) -> char[]

returns: char[]

changes_view_query(attribute_directory.attributestore) -> text

returns: text

check_attribute_types(attribute[]) -> SETOF attribute_directory.attribute

returns: *attribute_directory.attribute*

check_attributes_exist(attribute[]) -> SETOF attribute_directory.attribute

returns: *attribute_directory.attribute*

clean(store attribute_directory.attributestore, ts timestamp with time zone) -> bigint

returns: bigint

cleanup_attribute_after_delete() -> trigger

returns: trigger

cleanup_attributestore_on_delete() -> trigger

returns: trigger

cleanup_on_datasource_delete() -> trigger

returns: trigger

compact(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

Remove all subsequent records with duplicate attribute values and update the modified of the first

compacted_tmp_table_name(attribute_directory.attributestore) -> char[]

returns: char[]

compacted_view_name(attribute_directory.attributestore) -> char[]

returns: char[]

compacted_view_query(attribute_directory.attributestore) -> text

returns: text

create_at_func(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_at_func_ptr(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_at_func_ptr_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_attributestore(datasource_name text, entitytype_name text) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_attributestore(datasource_name text, entitytype_name text, attributes attribute_descr[]) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_attributestore_from_view(oid, datasource_name text, entitytype_name text) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_base_table(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_changes_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_changes_view_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_compacted_tmp_table(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_compacted_tmp_table_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_compacted_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_compacted_view_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_curr_ptr_table(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_curr_ptr_table_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_curr_ptr_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_curr_ptr_view_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_curr_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_curr_view_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_dependees(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_entity_at_func(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_entity_at_func_ptr(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_entity_at_func_ptr_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_entity_at_func_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_hash_function(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_hash_function_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_hash_triggers(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_hash_triggers_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_history_table(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_modified_trigger_function(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_modified_trigger_function_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_modified_triggers(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_modified_triggers_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_run_length_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

Create a view on an attributestore's history table that lists the runs of duplicate attribute data records by their entity Id and start-end. This can be used as a source for compacting actions.

create_run_length_view_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_sampled_view_materialization(view_class oid, source_modified_proc oid, datasource_name text, entitytype_name text) -> attribute_directory.sampled_view_materialization

returns: *attribute_directory.sampled_view_materialization*

create_staging_modified_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_staging_modified_view_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_staging_new_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_staging_new_view_sql(attribute_directory.attributestore) -> text[]

returns: text[]

create_staging_table(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

create_staging_table_sql(attribute_directory.attributestore) -> text[]

returns: text[]

curr_ptr_table_name(attribute_directory.attributestore) -> char[]

returns: char[]

curr_ptr_view_name(attribute_directory.attributestore) -> char[]

returns: char[]

curr_view_name(attribute_directory.attributestore) -> char[]

returns: char[]

curr_view_query(attribute_directory.attributestore) -> text

returns: text

datatype_order(datatype character varying) -> integer

returns: integer

define(attribute_directory.attribute) -> attribute_directory.attribute

returns: *attribute_directory.attribute*

define_attributestore(datasource_id integer, entitytype_id integer) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

define_attributestore(datasource_name text, entitytype_name text) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

define_sampled_view_materialization(attributestore_id integer, view_class oid, source_modified_proc oid) -> attribute_directory.sampled_view_materialization

returns: *attribute_directory.sampled_view_materialization*

dependees(attribute_directory.attributestore) -> obj_ref[]

returns: obj_ref[]

Return array with all managed dependees of attributestore base table

This array is primarily used to alter the base table using dep_recurse.alter so that the alter function can skip the database objects that are already dynamically created and recreated

dependers(name char[], level integer) -> TABLE(char[], integer)

returns: TABLE(char[], integer)

dependers(name char[]) -> TABLE(char[], integer)

returns: TABLE(char[], integer)

direct_dependers(name text) -> SETOF char[]

returns: char[]

drop_compacted_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

drop_compacted_view_sql(attribute_directory.attributestore) -> text

returns: text

drop_curr_ptr_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

drop_curr_ptr_view_sql(attribute_directory.attributestore) -> character varying

returns: character varying

drop_curr_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

drop_curr_view_sql(attribute_directory.attributestore) -> character varying

returns: character varying

drop_dependees(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

drop_entity_at_func(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

drop_entity_at_func_sql(attribute_directory.attributestore) -> text

returns: text

drop_hash_function(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

drop_staging_modified_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

drop_staging_modified_view_sql(attribute_directory.attributestore) -> character varying

returns: character varying

drop_staging_new_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

fingerprint(attribute_directory.sampled_view_materialization) -> text

returns: text

get_attribute(attribute_directory.attributestore, char[]) -> attribute_directory.attribute

returns: *attribute_directory.attribute*

get_attributes(oid) -> SETOF attribute_directory.attribute_descr

returns: *attribute_directory.attribute_descr*

get_attributes(attribute_directory.attributestore) -> SETOF attribute_directory.attribute

returns: *attribute_directory.attribute*

get_attributestore(datasource_id integer, entitytype_id integer) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

get_attributestore(attribute_directory.sampled_view_materialization) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

greatest_datatype(datatype_a character varying, datatype_b character varying) -> character varying

returns: character varying

init(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

init(attribute_directory.attribute) -> attribute_directory.attribute

returns: *attribute_directory.attribute*

insert_state(attribute_directory.sampled_view_materialization, text) -> text

returns: text

mark_compacted(attributestore_id integer, compacted timestamp with time zone) -> attribute_directory.attributestore_compacted

returns: *attribute_directory.attributestore_compacted*

mark_compacted(attributestore_id integer) -> attribute_directory.attributestore_compacted

returns: *attribute_directory.attributestore_compacted*

mark_curr_materialized(attributestore_id integer, materialized timestamp with time zone) -> attribute_directory.attributestore_curr_materialized

returns: *attribute_directory.attributestore_curr_materialized*

mark_curr_materialized(attributestore_id integer) -> attribute_directory.attributestore_curr_materialized

returns: *attribute_directory.attributestore_curr_materialized*

mark_modified(attributestore_id integer, modified timestamp with time zone) -> attribute_directory.attributestore_modified

returns: *attribute_directory.attributestore_modified*

mark_modified(attributestore_id integer) -> attribute_directory.attributestore_modified

returns: *attribute_directory.attributestore_modified*

materialize(store attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

materialize(attribute_directory.sampled_view_materialization) -> attribute_directory.sampled_view_materialization

returns: *attribute_directory.sampled_view_materialization*

materialize_curr_ptr(attribute_directory.attributestore) -> integer

returns: integer

max_modified(attribute_directory.sampled_view_materialization) -> timestamp with time zone

returns: timestamp with time zone

modify_column_type(attribute_directory.attributestore, column_name char[], datatype character varying) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

modify_column_type_sql(attribute_directory.attributestore, column_name char[], datatype character varying) -> text[]

returns: text[]

modify_datatype(attribute_directory.attribute) -> attribute_directory.attribute

returns: *attribute_directory.attribute*

remove_attribute(attribute_directory.attribute) -> dep_recurse.obj_ref

returns: dep_recurse.obj_ref

remove_attribute(attribute_directory.attributestore, char[]) -> dep_recurse.obj_ref

returns: dep_recurse.obj_ref

remove_attribute_column(attribute_directory.attributestore, char[]) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

Remove named column from the attributestore and update all attributestore system functions dependent on the columns (e.g.: hash function). Possible other/user defined functions dependent on the columns in the attributestore are outside of the scope of this function.

render_hash_query(attribute_directory.attributestore) -> text

returns: text

requires_compacting(attributestore_id integer) -> boolean

returns: boolean

requires_compacting(attribute_directory.attributestore) -> boolean

returns: boolean

run_length_view_name(attribute_directory.attributestore) -> char[]

returns: char[]

run_length_view_query(attribute_directory.attributestore) -> text

returns: text

sampled_view_materialization_runnable(timestamp with time zone) -> SETOF attribute_directory.sampled_view_materialization

returns: *attribute_directory.sampled_view_materialization*

set_hash() -> trigger

returns: trigger

stage_sample(attribute_directory.sampled_view_materialization) -> attribute_directory.sampled_view_materialization

returns: *attribute_directory.sampled_view_materialization*

staging_modified_view_name(attribute_directory.attributestore) -> char[]

returns: char[]

staging_new_view_name(attribute_directory.attributestore) -> char[]

returns: char[]

store_compacted(attributestore_id integer, compacted timestamp with time zone) -> attribute_directory.attributestore_compacted

returns: *attribute_directory.attributestore_compacted*

store_curr_materialized(attributestore_id integer, materialized timestamp with time zone) -> attribute_directory.attributestore_curr_materialized

returns: *attribute_directory.attributestore_curr_materialized*

store_modified(attributestore_id integer, modified timestamp with time zone) -> attribute_directory.attributestore_modified

returns: *attribute_directory.attributestore_modified*

store_state(attribute_directory.sampled_view_materialization, text) -> text

returns: text

store_state(attribute_directory.sampled_view_materialization) -> attribute_directory.sampled_view_materialization

returns: *attribute_directory.sampled_view_materialization*

to_attribute(attribute_directory.attribute) -> attribute_directory.attribute

returns: *attribute_directory.attribute*

to_attributestore(datasource_id integer, entitytype_id integer) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

to_char(attribute_directory.attributestore) -> text

returns: text

to_char(attribute_directory.sampled_view_materialization) -> text

returns: text

to_table_name(attribute_directory.attributestore) -> char[]

returns: char[]

transfer_staged(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

update_compacted(attributestore_id integer, compacted timestamp with time zone) -> attribute_directory.attributestore_compacted

returns: *attribute_directory.attributestore_compacted*

update_curr_materialized(attributestore_id integer, materialized timestamp with time zone) -> attribute_directory.attributestore_curr_materialized

returns: *attribute_directory.attributestore_curr_materialized*

update_datatype_on_change() -> trigger

returns: trigger

update_modified(attributestore_id integer, modified timestamp with time zone) -> attribute_directory.attributestore_modified

returns: *attribute_directory.attributestore_modified*

update_state(attribute_directory.sampled_view_materialization, text) -> text

returns: text

upgrade_attribute_table(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

view_to_attribute_staging_sql(oid, attribute_directory.attributestore) -> text

returns: text

3.4 attribute_history

Contains tables with the actual data of attribute stores

3.4.1 Types

3.4.2 Tables

3.4.3 Views

3.4.4 Functions

Name	Return Type	Description

3.5 attribute_staging

Contains tables for staging new data to be added to attribute stores

3.5.1 Types

3.5.2 Tables

3.5.3 Views

3.5.4 Functions

Name	Return Type	Description

3.6 dimension

3.6.1 Types

3.6.2 Tables

5m

Name	Type	Description
timestamp	timestamp with time zone	
start	timestamp with time zone	
end	timestamp with time zone	

day

Name	Type	Description
timestamp	timestamp with time zone	
start	timestamp with time zone	
end	timestamp with time zone	

day_15m

Name	Type	Description
timestamp	timestamp with time zone	
timestamp_15m	timestamp with time zone	

four_consec_qtr

Name	Type	Description
timestamp	timestamp with time zone	
start	timestamp with time zone	
end	timestamp with time zone	

hour

Name	Type	Description
timestamp	timestamp with time zone	
start	timestamp with time zone	
end	timestamp with time zone	

month

Name	Type	Description
timestamp	timestamp with time zone	
start	timestamp with time zone	
end	timestamp with time zone	

month_15m

Name	Type	Description
timestamp	timestamp with time zone	
timestamp_15m	timestamp with time zone	

quarter

Name	Type	Description
timestamp	timestamp with time zone	
start	timestamp with time zone	
end	timestamp with time zone	

week

Name	Type	Description
timestamp	timestamp with time zone	
start	timestamp with time zone	
end	timestamp with time zone	
year	smallint	
week_iso_8601	smallint	

week_15m

Name	Type	Description
timestamp	timestamp with time zone	
timestamp_15m	timestamp with time zone	

3.6.3 Views

3.6.4 Functions

Name	Return Type	Description
<i>update_5m()</i>	void	
<i>update_day()</i>	void	
<i>update_day_15m()</i>	void	
<i>update_four_consec_qtr()</i>	void	
<i>update_hour()</i>	void	
<i>update_month()</i>	void	
<i>update_month_15m()</i>	void	
<i>update_quarter()</i>	void	
<i>update_week()</i>	void	
<i>update_week_15m()</i>	void	

update_5m() -> void

returns: void

update_day() -> void

returns: void

update_day_15m() -> void

returns: void

update_four_consec_qtr() -> void

returns: void

update_hour() -> void

returns: void

update_month() -> void

returns: void

update_month_15m() -> void

returns: void

update_quarter() -> void

returns: void

update_week() -> void

returns: void

update_week_15m() -> void

returns: void

3.7 directory

Stores contextual information for the data. This includes the entities, entitytypes, datasources, etc. It is the entrypoint when looking for data.

3.7.1 Types

dn_part

Name	Type	Description
type_name	character varying	
name	character varying	

existence

Name	Type	Description
timestamp	timestamp with time zone	
exists	boolean	
entity_id	integer	

query_part

Name	Type	Description
c	text[]	
s	text	

query_row

Name	Type	Description
id	integer	
dn	text	
entitytype_id	integer	

3.7.2 Tables

alias

Name	Type	Description
entity_id	integer	
name	character varying	
type_id	integer	

aliastype

Name	Type	Description
name	character varying	
id	integer	

datasource

Describes datasources. A datasource is used to indicate where data came from. Datasources are also used to prevent collisions between sets of data from different sources, where names can be the same, but the meaning of the data differs.

Name	Type	Description
name	character varying	
description	character varying	
timezone	character varying	
id	integer	

entity

Describes entities. An entity is the base object for which the database can hold further information such as attributes, trends and notifications. All data must have a reference to an entity.

Name	Type	Description
first_appearance	timestamp with time zone	
name	character varying	
entitytype_id	integer	
dn	character varying	
parent_id	integer	
id	integer	

entity_link_denorm

The denormalized relation between entity and tag.

Stores one record for each entity, with all tag names in lower case in the tags column. A gin index on tags makes it ideal to lookup entities that have certain tags.

Name	Type	Description
entity_id	integer	
tags	text[]	
name	text	

entitytaglink

The n:n relation between *entity* and *tag*

Name	Type	Description
tag_id	integer	
entity_id	integer	

entitytype

Stores the entity types that exist in the entity table. Entity types are also used to give context to data that is stored for entities.

Name	Type	Description
name	character varying	
description	character varying	
id	integer	

existence_staging

Name	Type	Description
dn	character varying	

tag

Stores all tags. A tag is a simple label that can be attached to a number of object types in the database, such as entities and trends.

Name	Type	Description
name	character varying	
taggroup_id	integer	
description	character varying	
id	integer	

taggroup

Name	Type	Description
name	character varying	
complementary	boolean	
id	integer	

3.7.3 Views

existence_staging_entitytype_ids

Name	Type	Description
entitytype_id	integer	

3.7.4 Functions

Name	Return Type
<i>addentity(timestamp with time zone, character varying, integer, character varying, integer)</i>	integer
<i>array_to_dn_part(vchar[])</i>	directory.dn_part
<i>cluster_entity_tag_denorm(char[])</i>	char[]
<i>compile_minerva_query(query text)</i>	text
<i>compile_minerva_query(query query_part[])</i>	text
<i>create alias for new entity (func)()</i>	trigger
<i>create entitytaglink for new entity (func)()</i>	trigger
<i>create tag for new entitytypes (func)()</i>	trigger
<i>create_datasource(character varying)</i>	directory.datasource
<i>create_entity(character varying)</i>	directory.entity
<i>create_entity_tag_denorm(char[])</i>	char[]
<i>create_entity_tag_denorm_indexes(char[])</i>	char[]
<i>create_entity_tag_denorm_sql(char[])</i>	text[]
<i>create_entitytype(character varying)</i>	directory.entitytype
<i>dn_part_to_string(directory.dn_part)</i>	character varying
<i>dn_to_entity(character varying)</i>	directory.entity
<i>dns_to_entity_ids(vchar[])</i>	integer
<i>entities_by_type(character varying)</i>	directory.entity
<i>entities_by_type(integer)</i>	directory.entity
<i>entity_id(directory.entity)</i>	integer
<i>entitytype_id(directory.entitytype)</i>	integer
<i>existence_staging_state(timestamp with time zone, entitytype_id integer)</i>	directory.existence
<i>existing_staging(timestamp with time zone, entitytype_id integer)</i>	directory.existence
<i>explode_dn(character varying)</i>	dn_part[]
<i>get_alias(entity_id integer, aliastype_name character varying)</i>	character varying
<i>get_datasource(character varying)</i>	directory.datasource
<i>get_entity(character varying)</i>	directory.entity
<i>get_entitytype(character varying)</i>	directory.entitytype
<i>get_entitytype_id(character varying)</i>	integer
<i>getentitybydn(character varying)</i>	TABLE(integer, integer, character varying)
<i>getentitybyid(integer)</i>	TABLE(character varying, integer)
<i>glue_dn(dn_part[])</i>	character varying
<i>last_dn_part(dn_part[])</i>	directory.dn_part
<i>make_c_join(index integer, entity_id_table text, entity_id_column text, tag_index integer, tag text)</i>	text
<i>make_s_join(index integer, entity_id_table text, entity_id_column text, alias text)</i>	text
<i>name_to_datasource(character varying)</i>	directory.datasource
<i>name_to_entitytype(character varying)</i>	directory.entitytype
<i>new_existence_state(timestamp with time zone, entitytype_id integer)</i>	directory.existence
<i>non_existing_staging(timestamp with time zone, entitytype_id integer)</i>	directory.existence
<i>parent_dn(character varying)</i>	character varying
<i>parent_dn_parts(dn_part[])</i>	dn_part[]
<i>populate_entity_tag_denorm(char[])</i>	char[]
<i>rebuild_entity_tag_denorm(char[])</i>	char[]
<i>replace_entity_tag_denorm(char[])</i>	char[]
<i>run_minerva_query(query query_part[])</i>	TABLE(integer, character varying)
<i>split_raw_part(character varying)</i>	directory.dn_part
<i>sumproduct(query query_part[], value_trend text, weight_trend text)</i>	TABLE(timestamp with time zone)
<i>tag_entity(entity_id integer, tag character varying)</i>	integer
<i>tag_entity(dn character varying, tag character varying)</i>	character varying
<i>transfer_existence(timestamp with time zone)</i>	timestamp with time zone

Table 3.2 – continued from previous page

Name	Return Type
<i>update_denormalized_entity_tags(entity_id integer)</i>	text[]
<i>update_entity_link_denorm_for_delete()</i>	trigger
<i>update_entity_link_denorm_for_insert()</i>	trigger
<i>wavg(query query_part[], value_trend_id integer, weight_trend_id integer)</i>	TABLE(timestamp with time zone)

addentity(timestamp with time zone, character varying, integer, character varying, integer) -> integer

returns: integer

array_to_dn_part(varchar[]) -> directory.dn_part

returns: *directory.dn_part*

cluster_entity_tag_denorm(char[]) -> char[]

returns: char[]

compile_minerva_query(query text) -> text

returns: text

compile_minerva_query(query query_part[]) -> text

returns: text

create alias for new entity (func)() -> trigger

returns: trigger

create entitytaglink for new entity (func)() -> trigger

returns: trigger

create tag for new entitytypes (func)() -> trigger

returns: trigger

create_datasource(character varying) -> directory.datasource

returns: *directory.datasource*

Create a new datasource with specified name and return the new record

create_entity(character varying) -> directory.entity

returns: *directory.entity*

create_entity_tag_denorm(char[]) -> char[]

returns: char[]

create_entity_tag_denorm_indexes(char[]) -> char[]

returns: char[]

create_entity_tag_denorm_sql(char[]) -> text[]

returns: text[]

create_entitytype(character varying) -> directory.entitytype

returns: *directory.entitytype*

Create new entitytype with specified name and return it.

dn_part_to_string(directory.dn_part) -> character varying

returns: character varying

dn_to_entity(character varying) -> directory.entity

returns: *directory.entity*

Return existing or new entity with specified DN.

When an existing entity is found with the specified DN, then this is returned. Otherwise, a new entity is created, including any parents.

dns_to_entity_ids(varchar[]) -> SETOF integer

returns: integer

entities_by_type(character varying) -> SETOF directory.entity

returns: *directory.entity*

entities_by_type(integer) -> SETOF directory.entity

returns: *directory.entity*

entity_id(directory.entity) -> integer

returns: integer

entitytype_id(directory.entitytype) -> integer

returns: integer

existence_staging_state(timestamp with time zone, entitytype_id integer) -> SETOF directory.existence

returns: *directory.existence*

existing_staging(timestamp with time zone, entitytype_id integer) -> SETOF directory.existence

returns: *directory.existence*

explode_dn(character varying) -> dn_part[]

returns: dn_part[]

get_alias(entity_id integer, aliasname character varying) -> character varying

returns: character varying

get_datasource(character varying) -> directory.datasource

returns: *directory.datasource*

get_entity(character varying) -> directory.entity

returns: *directory.entity*

get_entitytype(character varying) -> directory.entitytype

returns: *directory.entitytype*

get_entitytype_id(character varying) -> integer

returns: integer

getentitybydn(character varying) -> TABLE(integer, integer, character varying, integer)

returns: TABLE(integer, integer, character varying, integer)

getentitybyid(integer) -> TABLE(character varying, integer, character varying, integer)

returns: TABLE(character varying, integer, character varying, integer)

glue_dn(dn_part[]) -> character varying

returns: character varying

last_dn_part(dn_part[]) -> directory.dn_part

returns: *directory.dn_part*

make_c_join(index integer, entity_id_table text, entity_id_column text, tag_index integer, tag text) -> text

returns: text

make_s_join(index integer, entity_id_table text, entity_id_column text, alias text) -> text

returns: text

name_to_datasource(character varying) -> directory.datasource

returns: *directory.datasource*

name_to_entitytype(character varying) -> directory.entitytype

returns: *directory.entitytype*

Return new or existing entitytype with specified name.

new_existence_state(timestamp with time zone, entitytype_id integer) -> SETOF directory.existence

returns: *directory.existence*

non_existing_staging(timestamp with time zone, entitytype_id integer) -> SETOF directory.existence

returns: *directory.existence*

parent_dn(character varying) -> character varying

returns: character varying

Return DN string of the parent.

parent_dn_parts(dn_part[]) -> dn_part[]

returns: dn_part[]

Return all but the last DN part or NULL if the array is empty.

populate_entity_tag_denorm(char[]) -> char[]

returns: char[]

rebuild_entity_tag_denorm(char[]) -> char[]

returns: char[]

Build a new denormalized entity tags table, populate it and replace the old one by using drop/rename to avoid blocking other users

replace_entity_tag_denorm(char[]) -> char[]

returns: char[]

run_minerva_query(query query_part[]) -> TABLE(integer, character varying, integer)

returns: TABLE(integer, character varying, integer)

split_raw_part(character varying) -> directory.dn_part

returns: *directory.dn_part*

sumproduct(query query_part[], value_trend text, weight_trend text) -> TABLE(timestamp with time zone, double precision)

returns: TABLE(timestamp with time zone, double precision)

tag_entity(entity_id integer, tag character varying) -> integer

returns: integer

tag_entity(dn character varying, tag character varying) -> character varying

returns: character varying

transfer_existence(timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

update_denormalized_entity_tags(entity_id integer) -> text[]

returns: text[]

update_entity_link_denorm_for_delete() -> trigger

returns: trigger

update_entity_link_denorm_for_insert() -> trigger

returns: trigger

wavg(query query_part[], value_trend_id integer, weight_trend_id integer) -> TABLE(timestamp with time zone, double precision)

returns: TABLE(timestamp with time zone, double precision)

3.8 entity_tag

3.8.1 Types

process_staged_links_result

Name	Type	Description
tags_added	bigint	
links_added	bigint	
links_removed	bigint	

update_result

Name	Type	Description
staged	bigint	
tags_added	bigint	
links_added	bigint	
links_removed	bigint	

3.8.2 Tables

entitytaglink_staging

Name	Type	Description
entity_id	integer	
tag_name	text	
taggroup_id	integer	

type

Name	Type	Description
name	char[]	
taggroup_id	integer	
id	integer	

3.8.3 Views

`_new_links_in_staging`

Name	Type	Description
<code>entity_id</code>	integer	
<code>tag_id</code>	integer	

`_new_tags_in_staging`

Name	Type	Description
<code>name</code>	text	
<code>taggroup_id</code>	integer	

`_obsolete_links`

Name	Type	Description
<code>entity_id</code>	integer	
<code>tag_id</code>	integer	

3.8.4 Functions

Name	Return Type	Description
<code>add_new_links(add_limit integer)</code>	bigint	
<code>add_new_tags()</code>	bigint	
<code>define(type_name char[], tag_group text, sql text)</code>	entity_tag.type	
<code>process_staged_links(process_limit integer)</code>	entity_tag.process_staged_links_result	
<code>remove_obsolete_links()</code>	bigint	
<code>transfer_to_staging(name char[])</code>	bigint	
<code>update(type_name char[], update_limit integer)</code>	entity_tag.update_result	

`add_new_links(add_limit integer) -> bigint`

returns: bigint

`add_new_tags() -> bigint`

returns: bigint

`define(type_name char[], tag_group text, sql text) -> entity_tag.type`

returns: *entity_tag.type*

`process_staged_links(process_limit integer) -> entity_tag.process_staged_links_result`

returns: *entity_tag.process_staged_links_result*

remove_obsolete_links() -> bigint

returns: bigint

transfer_to_staging(name char[]) -> bigint

returns: bigint

update(type_name char[], update_limit integer) -> entity_tag.update_result

returns: *entity_tag.update_result*

3.9 materialization

3.9.1 Types

materialization_result

Name	Type	Description
processed_max_modified	timestamp with time zone	
row_count	integer	

source_fragment

Name	Type	Description
trendstore_id	integer	
timestamp	timestamp with time zone	

source_fragment_state

Name	Type	Description
fragment	materialization.source_fragment	
modified	timestamp with time zone	

3.9.2 Tables

group_priority

Name	Type	Description
tag_id	integer	
resources	integer	

state

The Id of the materialization type

Name	Type	Description
type_id	integer	The Id of the materialization type
timestamp	timestamp with time zone	The timestamp of the materialized (materialization result) data
max_modified	timestamp with time zone	The greatest modified timestamp of all materialization sources
source_states	source_fragment_state	Array of trendstore_id/timestamp/modified combinations for all source data fragments
processed_states	source_fragment_state	Array containing a snapshot of the source_states at the time of the most recent materialization
job_id	integer	Id of the most recent job for this materialization

state

The timestamp of the materialized (materialization result) data

Name	Type	Description
type_id	integer	The Id of the materialization type
timestamp	timestamp with time zone	The timestamp of the materialized (materialization result) data
max_modified	timestamp with time zone	The greatest modified timestamp of all materialization sources
source_states	source_fragment_state	Array of trendstore_id/timestamp/modified combinations for all source data fragments
processed_states	source_fragment_state	Array containing a snapshot of the source_states at the time of the most recent materialization
job_id	integer	Id of the most recent job for this materialization

state

The greatest modified timestamp of all materialization sources

Name	Type	Description
type_id	integer	The Id of the materialization type
timestamp	timestamp with time zone	The timestamp of the materialized (materialization result) data
max_modified	timestamp with time zone	The greatest modified timestamp of all materialization sources
source_states	source_fragment_state	Array of trendstore_id/timestamp/modified combinations for all source data fragments
processed_states	source_fragment_state	Array containing a snapshot of the source_states at the time of the most recent materialization
job_id	integer	Id of the most recent job for this materialization

state

Array of trendstore_id/timestamp/modified combinations for all source data fragments

Name	Type	Description
type_id	integer	The Id of the materialization type
timestamp	timestamp with time zone	The timestamp of the materialized (materialization result) data
max_modified	timestamp with time zone	The greatest modified timestamp of all materialization sources
source_states	source_fragment_state	Array of trendstore_id/timestamp/modified combinations for all source data fragments
processed_states	source_fragment_state	Array containing a snapshot of the source_states at the time of the most recent materialization
job_id	integer	Id of the most recent job for this materialization

state

Array containing a snapshot of the source_states at the time of the most recent materialization

Name	Type	Description
type_id	integer	The Id of the materialization type
timestamp	timestamp with time zone	The timestamp of the materialized (materialization result) data
max_modified	timestamp with time zone	The greatest modified timestamp of all materialization sources
source_states	source_fragment_state	Array of trendstore_id/timestamp/modified combinations for all source data fragments
processed_states	source_fragment_state	Array containing a snapshot of the source_states at the time of the most recent materialization
job_id	integer	Id of the most recent job for this materialization

state

Id of the most recent job for this materialization

Name	Type	Description
type_id	integer	The Id of the materialization type
timestamp	timestamp with time zone	The timestamp of the materialized (materialization result) data
max_modified	timestamp with time zone	The greatest modified timestamp of all materialization sources
source_states	source_fragment_state	Array of trendstore_id/timestamp/modified combinations for all source data fragments
processed_states	source_fragment_state	Array containing a snapshot of the source_states at the time of the most recent materialization
job_id	integer	Id of the most recent job for this materialization

type

The Id of the source trendstore, which should be the Id of a view based trendstore

Name	Type	Description
src_trendstore_id	integer	The Id of the source trendstore, which should be the Id of a view based trendstore
dst_trendstore_id	integer	The Id of the destination trendstore, which should be the Id of a table based trendstore
processing_delay	interval	The time after the destination timestamp before this materialization can be executed
stability_delay	interval	The time to wait after the most recent modified timestamp before the source data is considered 'stable'
reprocessing_period	interval	The maximum time after the destination timestamp that the materialization is allowed to be executed
id	integer	
enabled	boolean	Indicates if jobs should be created for this materialization (manual execution is always possible)
cost	integer	

type

The Id of the destination trendstore, which should be the Id of a table based trendstore

Name	Type	Description
src_trendstore_id	integer	The Id of the source trendstore, which should be the Id of a view based trendstore
dst_trendstore_id	integer	The Id of the destination trendstore, which should be the Id of a table based trendstore
processing_delay	interval	The time after the destination timestamp before this materialization can be executed
stability_delay	interval	The time to wait after the most recent modified timestamp before the source data is considered 'stable'
reprocessing_period	interval	The maximum time after the destination timestamp that the materialization is allowed to be executed
id	integer	
enabled	boolean	Indicates if jobs should be created for this materialization (manual execution is always possible)
cost	integer	

type

The time after the destination timestamp before this materialization can be executed

Name	Type	Description
src_trendstore_id	integer	The Id of the source trendstore, which should be the Id of a view based trendstore
dst_trendstore_id	integer	The Id of the destination trendstore, which should be the Id of a table based trendstore
processing_delay	interval	The time after the destination timestamp before this materialization can be executed
stability_delay	interval	The time to wait after the most recent modified timestamp before the source data is considered 'stable'
reprocessing_period	interval	The maximum time after the destination timestamp that the materialization is allowed to be executed
id	integer	
enabled	boolean	Indicates if jobs should be created for this materialization (manual execution is always possible)
cost	integer	

type

The time to wait after the most recent modified timestamp before the source data is considered 'stable'

Name	Type	Description
src_trendstore_id	integer	The Id of the source trendstore, which should be the Id of a view based trendstore
dst_trendstore_id	integer	The Id of the destination trendstore, which should be the Id of a table based trendstore
processing_delay	interval	The time after the destination timestamp before this materialization can be executed
stability_delay	interval	The time to wait after the most recent modified timestamp before the source data is considered 'stable'
reprocessing_period	interval	The maximum time after the destination timestamp that the materialization is allowed to be executed
id	integer	
enabled	boolean	Indicates if jobs should be created for this materialization (manual execution is always possible)
cost	integer	

type

The maximum time after the destination timestamp that the materialization is allowed to be executed

Name	Type	Description
src_trendstore_id	integer	The Id of the source trendstore, which should be the Id of a view based trendstore
dst_trendstore_id	integer	The Id of the destination trendstore, which should be the Id of a table based trendstore
processing_delay	interval	The time after the destination timestamp before this materialization can be executed
stability_delay	interval	The time to wait after the most recent modified timestamp before the source data is considered 'stable'
reprocessing_period	interval	The maximum time after the destination timestamp that the materialization is allowed to be executed
id	integer	
enabled	boolean	Indicates if jobs should be created for this materialization (manual execution is always possible)
cost	integer	

type

Indicates if jobs should be created for this materialization (manual execution is always possible)

Name	Type	Description
src_trendstore_id	integer	The Id of the source trendstore, which should be the Id of a view based trendstore
dst_trendstore_id	integer	The Id of the destination trendstore, which should be the Id of a table based trendstore
processing_delay	interval	The time after the destination timestamp before this materialization can be executed
stability_delay	interval	The time to wait after the most recent modified timestamp before the source data is considered 'stable'
reprocessing_period	interval	The maximum time after the destination timestamp that the materialization is allowed to be executed
id	integer	
enabled	boolean	Indicates if jobs should be created for this materialization (manual execution is always possible)
cost	integer	

type_tag_link

Name	Type	Description
type_id	integer	
tag_id	integer	

3.9.3 Views

materializable_source_state

Name	Type	Description
type_id	integer	
timestamp	timestamp with time zone	
trendstore_id	integer	
src_timestamp	timestamp with time zone	
modified	timestamp with time zone	

materializables

Name	Type	Description
type_id	integer	
timestamp	timestamp with time zone	
max_modified	timestamp with time zone	
source_states	source_fragment_state[]	

modified_materializables

Name	Type	Description
type_id	integer	
timestamp	timestamp with time zone	
max_modified	timestamp with time zone	
source_states	source_fragment_state[]	

new_materializables

Name	Type	Description
type_id	integer	
timestamp	timestamp with time zone	
max_modified	timestamp with time zone	
source_states	source_fragment_state[]	

next_up_materializations

Name	Type	Description
type_id	integer	
timestamp	timestamp with time zone	
name	character varying	
cost	integer	
cumsum	bigint	
group_resources	integer	
job_active	boolean	

obsolete_state

Name	Type	Description
type_id	integer	
timestamp	timestamp with time zone	

required_resources_by_group

Name	Type	Description
tag_id	integer	
required	bigint	

runnable_materializations

Name	Type	Description
type	materialization.type	
state	materialization.state	

tagged_runnable_materializations

Name	Type	Description
type_id	integer	
timestamp	timestamp with time zone	
tag	character varying	

trend_ext

Convenience view for easy lookup of trends

Name	Type	Description
id	integer	
name	character varying	
datasource_name	character varying	
entitytype_name	character varying	
granularity	character varying	
materialized	boolean	

3.9.4 Functions

Name	Return
<i>add_missing_trends(src trend.trendstore, dst trend.trendstore)</i>	bigint
<i>add_missing_trends(materialization.type)</i>	materialization.type
<i>add_new_state()</i>	integer
<i>create_job(type_id integer, timestamp timestamp with time zone)</i>	integer
<i>create_jobs(tag character varying, job_limit integer)</i>	integer
<i>create_jobs(tag character varying)</i>	integer
<i>create_jobs()</i>	integer

Table 3.3 – co

Name	Return
<i>create_jobs_limited(tag character varying, job_limit integer)</i>	integer
<i>default_processing_delay(granularity character varying)</i>	interval
<i>default_stability_delay(granularity character varying)</i>	interval
<i>define(src_trendstore_id integer, dst_trendstore_id integer)</i>	materialization
<i>define(src trend.trendstore, dst trend.trendstore)</i>	materialization
<i>define(text, text)</i>	materialization
<i>define(trend.trendstore)</i>	materialization
<i>define(trend.view)</i>	materialization
<i>delete_obsolete_state()</i>	integer
<i>dependencies(trend.trendstore, level integer)</i>	TABLE
<i>dependencies(trend.trendstore)</i>	TABLE
<i>dependencies(name text)</i>	TABLE
<i>direct_dependencies(trend.trendstore)</i>	trend.t
<i>direct_table_dependencies(trend.trendstore)</i>	trend.t
<i>direct_view_dependencies(trend.trendstore)</i>	trend.t
<i>disable(materialization.type)</i>	materialization
<i>enable(materialization.type)</i>	materialization
<i>fragments(source_fragment_state[])</i>	source
<i>materialization(src text, dst text, timestamp timestamp with time zone)</i>	materialization
<i>materialize(src trend.trendstore, dst trend.trendstore, timestamp timestamp with time zone)</i>	materialization
<i>materialize(src_trendstore_id integer, dst_trendstore_id integer, timestamp timestamp with time zone)</i>	materialization
<i>materialize(materialization text, timestamp timestamp with time zone)</i>	materialization
<i>materialize(materialization.type, timestamp timestamp with time zone)</i>	materialization
<i>materialize(id integer, timestamp timestamp with time zone)</i>	materialization
<i>missing_columns(src trend.trendstore, dst trend.trendstore)</i>	TABLE
<i>missing_columns(materialization.type)</i>	TABLE
<i>modify_mismatching_trends(src trend.trendstore, dst trend.trendstore)</i>	void
<i>modify_mismatching_trends(materialization.type)</i>	void
<i>no_slave_lag()</i>	boolean
<i>open_job_slots(slot_count integer)</i>	integer
<i>render_job_json(type_id integer, timestamp with time zone)</i>	character
<i>requires_update(materialization.state)</i>	boolean
<i>reset(type_id integer)</i>	materialization
<i>reset(type_id integer, timestamp with time zone)</i>	materialization
<i>reset(materialization.type, timestamp with time zone)</i>	materialization
<i>reset_hard(materialization.type)</i>	void
<i>runnable(materialization.type, materialization.state)</i>	boolean
<i>runnable(type materialization.type, timestamp timestamp with time zone, max_modified timestamp with time zone)</i>	boolean
<i>runnable_materializations(tag character varying)</i>	TABLE
<i>source_data_ready(type materialization.type, timestamp timestamp with time zone, max_modified timestamp with time zone)</i>	boolean
<i>tag(tag_name character varying, type_id integer)</i>	materialization
<i>tag(tag_name character varying, materialization.type)</i>	materialization
<i>to_char(materialization.type)</i>	text
<i>untag(materialization.type)</i>	materialization
<i>update_modified_state()</i>	integer
<i>update_state()</i>	text

add_missing_trends(src trend.trendstore, dst trend.trendstore) -> bigint

returns: bigint

Add trends and actual table columns to destination that exist in the source trendstore but not yet in the destination.

add_missing_trends(materialization.type) -> materialization.type

returns: materialization.type

add_new_state() -> integer

returns: integer

create_job(type_id integer, timestamp timestamp with time zone) -> integer

returns: integer

create_jobs(tag character varying, job_limit integer) -> integer

returns: integer

create_jobs(tag character varying) -> integer

returns: integer

create_jobs() -> integer

returns: integer

create_jobs_limited(tag character varying, job_limit integer) -> integer

returns: integer

Deprecated function that just calls the overloaded create_jobs function.

default_processing_delay(granularity character varying) -> interval

returns: interval

default_stability_delay(granularity character varying) -> interval

returns: interval

define(src_trendstore_id integer, dst_trendstore_id integer) -> materialization.type

returns: materialization.type

define(src trend.trendstore, dst trend.trendstore) -> materialization.type

returns: materialization.type

define(text, text) -> materialization.type

returns: materialization.type

define(trend.trendstore) -> materialization.type

returns: materialization.type

Defines a new materialization with the convention that the datasource of the source trendstore should start with a 'v' for views and that the destination trendstore has the same properties except for a datasource with a name without the leading 'v'. A new trendstore and datasource are created if they do not exist.

define(trend.view) -> materialization.type

returns: materialization.type

Defines a new materialization with the convention that the datasource of the source trendstore should start with a 'v' for views and that the destination trendstore has the same properties except for a datasource with a name without the leading 'v'. A new trendstore and datasource are created if they do not exist.

delete_obsolete_state() -> integer

returns: integer

dependencies(trend.trendstore, level integer) -> TABLE(trend.trendstore, integer)

returns: TABLE(trend.trendstore, integer)

dependencies(trend.trendstore) -> TABLE(trend.trendstore, integer)

returns: TABLE(trend.trendstore, integer)

dependencies(name text) -> TABLE(trend.trendstore, integer)

returns: TABLE(trend.trendstore, integer)

direct_dependencies(trend.trendstore) -> SETOF trend.trendstore

returns: *trend.trendstore*

direct_table_dependencies(trend.trendstore) -> SETOF trend.trendstore

returns: *trend.trendstore*

direct_view_dependencies(trend.trendstore) -> SETOF trend.trendstore

returns: *trend.trendstore*

disable(materialization.type) -> materialization.type

returns: materialization.type

enable(materialization.type) -> materialization.type

returns: materialization.type

fragments(source_fragment_state[]) -> source_fragment[]

returns: source_fragment[]

materialization(src text, dst text, timestamp timestamp with time zone) -> materialization.materialization_result

returns: *materialization.materialization_result*

materialize(src trend.trendstore, dst trend.trendstore, timestamp timestamp with time zone) -> materialization.materialization_result

returns: *materialization.materialization_result*

materialize(src_trendstore_id integer, dst_trendstore_id integer, timestamp timestamp with time zone) -> materialization.materialization_result

returns: *materialization.materialization_result*

materialize(materialization text, timestamp timestamp with time zone) -> materialization.materialization_result

returns: *materialization.materialization_result*

materialize(materialization.type, timestamp timestamp with time zone) -> materialization.materialization_result

returns: *materialization.materialization_result*

materialize(id integer, timestamp timestamp with time zone) -> materialization.materialization_result

returns: *materialization.materialization_result*

missing_columns(src trend.trendstore, dst trend.trendstore) -> TABLE(character varying, character varying)

returns: TABLE(character varying, character varying)

The set of table columns (name, datatype) that exist in the source trendstore but not yet in the destination.

missing_columns(materialization.type) -> TABLE(character varying, character varying)

returns: TABLE(character varying, character varying)

modify_mismatching_trends(src trend.trendstore, dst trend.trendstore) -> void

returns: void

modify_mismatching_trends(materialization.type) -> void

returns: void

no_slave_lag() -> boolean

returns: boolean

open_job_slots(slot_count integer) -> integer

returns: integer

render_job_json(type_id integer, timestamp with time zone) -> character varying

returns: character varying

requires_update(materialization.state) -> boolean

returns: boolean

reset(type_id integer) -> SETOF materialization.state

returns: materialization.state

reset(type_id integer, timestamp with time zone) -> materialization.state

returns: materialization.state

reset(materialization.type, timestamp with time zone) -> materialization.state

returns: materialization.state

reset_hard(materialization.type) -> void

returns: void

Remove data (partitions) resulting from this materialization and the corresponding state records, so materialization for all timestamps can be done again

runnable(materialization.type, materialization.state) -> boolean

returns: boolean

runnable(type materialization.type, timestamp timestamp with time zone, max_modified timestamp with time zone) -> boolean

returns: boolean

runnable_materializations(tag character varying) -> TABLE(integer, timestamp with time zone)

returns: TABLE(integer, timestamp with time zone)

Return table with all combinations (type_id, timestamp) that are ready to run. This includes the check between the master and slave states.

source_data_ready(type materialization.type, timestamp timestamp with time zone, max_modified timestamp with time zone) -> boolean

returns: boolean

tag(tag_name character varying, type_id integer) -> materialization.type_tag_link

returns: *materialization.type_tag_link*

Add tag with name tag_name to materialization type with id type_id. The tag must already exist.

tag(tag_name character varying, materialization.type) -> materialization.type

returns: materialization.type

Add tag with name tag_name to materialization type. The tag must already exist.

to_char(materialization.type) -> text

returns: text

untag(materialization.type) -> materialization.type

returns: materialization.type

Remove all tags from the materialization

update_modified_state() -> integer

returns: integer

update_state() -> text

returns: text

3.10 notification

Stores information of events that can occur at irregular intervals, but still have a fixed, known format.

3.10.1 Types

attr_def

Name	Type	Description
name	char[]	
data_type	char[]	

3.10.2 Tables

attribute

Describes attributes of notificationstores. An attribute of a notificationstore is an attribute that each notification stored in that notificationstore has. An attribute corresponds directly to a column in the main notificationstore table

Name	Type	Description
notificationstore_id	integer	
name	char[]	
data_type	char[]	
description	character varying	
id	integer	

notificationsetstore

Describes notificationsetstores. A notificationsetstore can hold information over sets of notifications that are related to each other.

Name	Type	Description
name	char[]	
notificationstore_id	integer	
id	integer	

notificationstore

Describes notificationstores. Each notificationstore maps to a set of tables and functions that can store and manage notifications of a certain type. These corresponding tables and functions are created automatically for each notificationstore. Because each notificationstore maps one-on-one to a datasource, the name of the notificationstore is the same as that of the datasource. Use the `create_notificationstore` function to create new notificationstores.

Name	Type	Description
datasource_id	integer	
version	integer	
id	integer	

setattribute

Describes attributes of notificationsetstores. A setattribute of a notificationsetstore is an attribute that each notification set has. A setattribute corresponds directly to a column in the main notificationsetstore table.

Name	Type	Description
notificationsetstore_id	integer	
name	char[]	
data_type	char[]	
description	character varying	
id	integer	

3.10.3 Views

3.10.4 Functions

Name	Return Type	Des
<i>action(anyelement, text)</i>	anyelement	
<i>add_attribute_column_sql(char[], notification.attribute)</i>	text	
<i>add_staging_attribute_column_sql(notification.attribute)</i>	text	
<i>cleanup_on_datasource_delete()</i>	trigger	
<i>column_exists(schema_name char[], table_name char[], column_name char[])</i>	boolean	
<i>column_exists(table_name char[], column_name char[])</i>	boolean	
<i>create_attribute(notification.notificationstore, char[], char[])</i>	notification.attribute	
<i>create_attribute_column(notification.attribute)</i>	notification.attribute	
<i>create_attribute_column_on_insert()</i>	trigger	
<i>create_notificationsetstore(name char[], notificationstore_id integer)</i>	notification.notificationsetstore	
<i>create_notificationsetstore(name char[], notification.notificationstore)</i>	notification.notificationsetstore	
<i>create_notificationstore(datasource_id integer)</i>	notification.notificationstore	
<i>create_notificationstore(datasource_name text)</i>	notification.notificationstore	
<i>create_notificationstore(datasource_id integer, attr_def[])</i>	notification.notificationstore	
<i>create_notificationstore(datasource_name text, attr_def[])</i>	notification.notificationstore	
<i>create_staging_table(notification.notificationstore)</i>	notification.notificationstore	
<i>create_table(notification.notificationstore)</i>	notification.notificationstore	
<i>create_table_on_insert()</i>	trigger	
<i>define_notificationsetstore(name char[], notificationstore_id integer)</i>	notification.notificationsetstore	
<i>drop_notificationsetstore_table_on_delete()</i>	trigger	
<i>drop_table_on_delete()</i>	trigger	
<i>get_attr_defs(notification.notificationstore)</i>	notification.attr_def	
<i>get_column_type_name(namespace_name char[], table_name char[], column_name char[])</i>	char[]	
<i>get_column_type_name(notification.notificationstore, char[])</i>	char[]	
<i>get_notificationstore(datasource_name char[])</i>	notification.notificationstore	
<i>init_notificationsetstore(notification.notificationsetstore)</i>	notification.notificationsetstore	
<i>notificationstore(notification.notificationsetstore)</i>	notification.notificationstore	
<i>staging_table_name(notification.notificationstore)</i>	char[]	
<i>table_exists(schema_name char[], table_name char[])</i>	boolean	
<i>table_exists(char[])</i>	boolean	
<i>table_name(notification.notificationstore)</i>	char[]	
<i>to_char(notification.notificationstore)</i>	text	

action(anyelement, text) -> anyelement

returns: anyelement

add_attribute_column_sql(char[], notification.attribute) -> text

returns: text

add_staging_attribute_column_sql(notification.attribute) -> text

returns: text

cleanup_on_datasource_delete() -> trigger

returns: trigger

column_exists(schema_name char[], table_name char[], column_name char[]) -> boolean

returns: boolean

column_exists(table_name char[], column_name char[]) -> boolean

returns: boolean

create_attribute(notification.notificationstore, char[], char[]) -> SETOF notification.attribute

returns: *notification.attribute*

create_attribute_column(notification.attribute) -> notification.attribute

returns: *notification.attribute*

create_attribute_column_on_insert() -> trigger

returns: trigger

create_notificationsetstore(name char[], notificationstore_id integer) -> notification.notificationsetstore

returns: *notification.notificationsetstore*

create_notificationsetstore(name char[], notification.notificationstore) -> notification.notificationsetstore

returns: *notification.notificationsetstore*

create_notificationstore(datasource_id integer) -> notification.notificationstore

returns: *notification.notificationstore*

create_notificationstore(datasource_name text) -> notification.notificationstore

returns: *notification.notificationstore*

create_notificationstore(datasource_id integer, attr_def[]) -> notification.notificationstore

returns: *notification.notificationstore*

create_notificationstore(datasource_name text, attr_def[]) -> notification.notificationstore

returns: *notification.notificationstore*

create_staging_table(notification.notificationstore) -> notification.notificationstore

returns: *notification.notificationstore*

create_table(notification.notificationstore) -> notification.notificationstore

returns: *notification.notificationstore*

create_table_on_insert() -> trigger

returns: trigger

define_notificationsetstore(name char[], notificationstore_id integer) -> notification.notificationsetstore

returns: *notification.notificationsetstore*

drop_notificationsetstore_table_on_delete() -> trigger

returns: trigger

drop_table_on_delete() -> trigger

returns: trigger

get_attr_defs(notification.notificationstore) -> SETOF notification.attr_def

returns: *notification.attr_def*

get_column_type_name(namespace_name char[], table_name char[], column_name char[]) -> char[]

returns: char[]

get_column_type_name(notification.notificationstore, char[]) -> char[]

returns: char[]

get_notificationstore(datasource_name char[]) -> notification.notificationstore

returns: *notification.notificationstore*

init_notificationsetstore(notification.notificationsetstore) -> notification.notificationsetstore

returns: *notification.notificationsetstore*

notificationstore(notification.notificationsetstore) -> notification.notificationstore

returns: *notification.notificationstore*

staging_table_name(notification.notificationstore) -> char[]

returns: char[]

table_exists(schema_name char[], table_name char[]) -> boolean

returns: boolean

table_exists(char[]) -> boolean

returns: boolean

table_name(notification.notificationstore) -> char[]

returns: char[]

to_char(notification.notificationstore) -> text

returns: text

3.11 olap

3.11.1 Types

3.11.2 Tables

3.11.3 Views

3.11.4 Functions

Name	Return Type	Description

3.12 public

standard public schema

3.12.1 Types

`_time_trial_type`

Name	Type	Description
<code>a_time</code>	numeric	

`column_info`

Name	Type	Description
<code>name</code>	char[]	
<code>data_type</code>	text	

3.12.2 Tables

3.12.3 Views

pg_all_foreign_keys

Name	Type	Description
fk_schema_name	char[]	
fk_table_name	char[]	
fk_constraint_name	char[]	
fk_table_oid	oid	
fk_columns	name[]	
pk_schema_name	char[]	
pk_table_name	char[]	
pk_constraint_name	char[]	
pk_table_oid	oid	
pk_index_name	char[]	
pk_columns	name[]	
match_type	text	
on_delete	text	
on_update	text	
is_deferrable	boolean	
is_deferred	boolean	

tap_funky

Name	Type	Description
oid	oid	
schema	char[]	
name	char[]	
owner	char[]	
args	text	
returns	text	
langoid	oid	
is_strict	boolean	
is_agg	boolean	
is_definer	boolean	
returns_set	boolean	
volatility	character	
is_visible	boolean	

3.12.4 Functions

Name	Return Type	Description
<i>_add(text, integer, text)</i>	integer	
<i>_add(text, integer)</i>	integer	
<i>_agg(char[], char[], name[])</i>	boolean	
<i>_agg(char[], char[])</i>	boolean	
<i>_agg(char[], name[])</i>	boolean	

Table 3.5 – continued from previous page

Name	Return Type	Description
<code>_agg(char[])</code>	boolean	
<code>_alike(boolean, anyelement, text, text)</code>	text	
<code>_are(text, name[], name[], text)</code>	text	
<code>_areni(text, text[], text[], text)</code>	text	
<code>_assets_are(text, text[], text[], text)</code>	text	
<code>_cast_exists(char[], char[], char[], char[])</code>	boolean	
<code>_cast_exists(char[], char[], char[])</code>	boolean	
<code>_cast_exists(char[], char[])</code>	boolean	
<code>_cdi(char[], char[], char[], anyelement, text)</code>	text	
<code>_cdi(char[], char[], anyelement, text)</code>	text	
<code>_cdi(char[], char[], anyelement)</code>	text	
<code>_cexists(char[], char[], char[])</code>	boolean	
<code>_cexists(char[], char[])</code>	boolean	
<code>_ckeys(char[], char[], character)</code>	name[]	
<code>_ckeys(char[], character)</code>	name[]	
<code>_cleanup()</code>	boolean	
<code>_cmp_types(oid, char[])</code>	boolean	
<code>_col_is_null(char[], char[], char[], text, boolean)</code>	text	
<code>_col_is_null(char[], char[], text, boolean)</code>	text	
<code>_constraint(char[], char[], character, name[], text, text)</code>	text	
<code>_constraint(char[], character, name[], text, text)</code>	text	
<code>_contract_on(text)</code>	“char”	
<code>_currtest()</code>	integer	
<code>_db_privs()</code>	name[]	
<code>_def_is(text, text, anyelement, text)</code>	text	
<code>_definer(char[], char[], name[])</code>	boolean	
<code>_definer(char[], char[])</code>	boolean	
<code>_definer(char[], name[])</code>	boolean	
<code>_definer(char[])</code>	boolean	
<code>_dexists(char[], char[])</code>	boolean	
<code>_dexists(char[])</code>	boolean	
<code>_do_ne(text, text, text, text)</code>	text	
<code>_docomp(text, text, text, text)</code>	text	
<code>_expand_context(character)</code>	text	
<code>_expand_on(character)</code>	text	
<code>_expand_vol(character)</code>	text	
<code>_extras(character, char[], name[])</code>	name[]	
<code>_extras(character, name[])</code>	name[]	
<code>_finish(integer, integer, integer)</code>	text	
<code>_fkexists(char[], char[], name[])</code>	boolean	
<code>_fkexists(char[], name[])</code>	boolean	
<code>_fprivs_are(text, char[], name[], text)</code>	text	
<code>_func_compare(char[], char[], name[], anyelement, anyelement, text)</code>	text	
<code>_func_compare(char[], char[], name[], boolean, text)</code>	text	
<code>_func_compare(char[], char[], anyelement, anyelement, text)</code>	text	
<code>_func_compare(char[], char[], boolean, text)</code>	text	
<code>_get(text)</code>	integer	
<code>_get_ac_privs(char[], text)</code>	text[]	
<code>_get_col_ns_type(char[], char[], char[])</code>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<code>_get_col_privs(char[], text, char[])</code>	text[]	
<code>_get_col_type(char[], char[], char[])</code>	text	
<code>_get_col_type(char[], char[])</code>	text	
<code>_get_context(char[], char[])</code>	“char”	
<code>_get_db_owner(char[])</code>	char[]	
<code>_get_db_privs(char[], text)</code>	text[]	
<code>_get_dtype(char[], text, boolean)</code>	text	
<code>_get_dtype(char[])</code>	text	
<code>_get_fdw_privs(char[], text)</code>	text[]	
<code>_get_func_owner(char[], char[], name[])</code>	char[]	
<code>_get_func_owner(char[], name[])</code>	char[]	
<code>_get_func_privs(text, text)</code>	text[]	
<code>_get_index_owner(char[], char[], char[])</code>	char[]	
<code>_get_index_owner(char[], char[])</code>	char[]	
<code>_get_lang_privs(char[], text)</code>	text[]	
<code>_get_language_owner(char[])</code>	char[]	
<code>_get_latest(text)</code>	int4[]	
<code>_get_latest(text, integer)</code>	integer	
<code>_get_note(text)</code>	text	
<code>_get_note(integer)</code>	text	
<code>_get_opclass_owner(char[], char[])</code>	char[]	
<code>_get_opclass_owner(char[])</code>	char[]	
<code>_get_rel_owner(character, char[])</code>	char[]	
<code>_get_rel_owner(char[], char[])</code>	char[]	
<code>_get_rel_owner(char[])</code>	char[]	
<code>_get_rel_owner(character, char[], char[])</code>	char[]	
<code>_get_schema_owner(char[])</code>	char[]	
<code>_get_schema_privs(char[], text)</code>	text[]	
<code>_get_sequence_privs(char[], text)</code>	text[]	
<code>_get_server_privs(char[], text)</code>	text[]	
<code>_get_table_privs(char[], text)</code>	text[]	
<code>_get_tablespace_owner(char[])</code>	char[]	
<code>_get_tablespaceprivs(char[], text)</code>	text[]	
<code>_get_type_owner(char[], char[])</code>	char[]	
<code>_get_type_owner(char[])</code>	char[]	
<code>_got_func(char[], char[], name[])</code>	boolean	
<code>_got_func(char[], char[])</code>	boolean	
<code>_got_func(char[], name[])</code>	boolean	
<code>_got_func(char[])</code>	boolean	
<code>_grolist(char[])</code>	oid[]	
<code>_has_def(char[], char[], char[])</code>	boolean	
<code>_has_def(char[], char[])</code>	boolean	
<code>_has_group(char[])</code>	boolean	
<code>_has_role(char[])</code>	boolean	
<code>_has_type(char[], char[], bpchar[])</code>	boolean	
<code>_has_type(char[], bpchar[])</code>	boolean	
<code>_has_user(char[])</code>	boolean	
<code>_hasc(char[], char[], character)</code>	boolean	
<code>_hasc(char[], character)</code>	boolean	

Table 3.5 – continued from previous page

Name	Return Type	Description
<code>_have_index(char[], char[], char[])</code>	boolean	
<code>_have_index(char[], char[])</code>	boolean	
<code>_ident_array_to_string(name[], text)</code>	text	
<code>_ikeys(char[], char[], char[])</code>	text[]	
<code>_ikeys(char[], char[])</code>	text[]	
<code>_is_instead(char[], char[], char[])</code>	boolean	
<code>_is_instead(char[], char[])</code>	boolean	
<code>_is_schema(char[])</code>	boolean	
<code>_is_super(char[])</code>	boolean	
<code>_is_trusted(char[])</code>	boolean	
<code>_is_verbose()</code>	boolean	
<code>_keys(char[], char[], character)</code>	name[]	
<code>_keys(char[], character)</code>	name[]	
<code>_lang(char[], char[], name[])</code>	char[]	
<code>_lang(char[], char[])</code>	char[]	
<code>_lang(char[], name[])</code>	char[]	
<code>_lang(char[])</code>	char[]	
<code>_missing(character, char[], name[])</code>	name[]	
<code>_missing(character, name[])</code>	name[]	
<code>_nosuch(char[], char[], name[])</code>	text	
<code>_op_exists(char[], char[], char[], char[], char[])</code>	boolean	
<code>_op_exists(char[], char[], char[], char[])</code>	boolean	
<code>_op_exists(char[], char[], char[])</code>	boolean	
<code>_opc_exists(char[], char[])</code>	boolean	
<code>_opc_exists(char[])</code>	boolean	
<code>_pg_sv_column_array(oid, int2[])</code>	name[]	
<code>_pg_sv_table_accessible(oid, oid)</code>	boolean	
<code>_pg_sv_type_array(oid[])</code>	name[]	
<code>_query(text)</code>	text	
<code>_quote_ident_like(text, text)</code>	text	
<code>_refine_vol(text)</code>	text	
<code>_relcomp(text, text, text, text)</code>	text	
<code>_relcomp(text, anyarray, text, text)</code>	text	
<code>_relcomp(text, text, text, text, text)</code>	text	
<code>_relexists(char[], char[])</code>	boolean	
<code>_relexists(char[])</code>	boolean	
<code>_relne(text, text, text, text)</code>	text	
<code>_relne(text, anyarray, text, text)</code>	text	
<code>_returns(char[], char[], name[])</code>	text	
<code>_returns(char[], char[])</code>	text	
<code>_returns(char[], name[])</code>	text	
<code>_returns(char[])</code>	text	
<code>_rexists(character, char[], char[])</code>	boolean	
<code>_rexists(character, char[])</code>	boolean	
<code>_rule_on(char[], char[], char[])</code>	“char”	
<code>_rule_on(char[], char[])</code>	“char”	
<code>_runem(text[], boolean)</code>	text	
<code>_runner(text[], text[], text[], text[], text[])</code>	text	
<code>_set(text, integer, text)</code>	integer	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>_set(text, integer)</i>	integer	
<i>_set(integer, integer)</i>	integer	
<i>_strict(char[], char[], name[])</i>	boolean	
<i>_strict(char[], char[])</i>	boolean	
<i>_strict(char[], name[])</i>	boolean	
<i>_strict(char[])</i>	boolean	
<i>_table_privs()</i>	name[]	
<i>_temptable(text, text)</i>	text	
<i>_temptable(anyarray, text)</i>	text	
<i>_temptypes(text)</i>	text	
<i>_time_trials(text, integer, numeric)</i>	<i>_time_trial_type</i>	
<i>_tlike(boolean, text, text, text)</i>	text	
<i>_todo()</i>	text	
<i>_trig(char[], char[], char[])</i>	boolean	
<i>_trig(char[], char[])</i>	boolean	
<i>_types_are(char[], name[], text, bpchar[])</i>	text	
<i>_types_are(name[], text, bpchar[])</i>	text	
<i>_unlike(boolean, anyelement, text, text)</i>	text	
<i>_vol(char[], char[], name[])</i>	text	
<i>_vol(char[], char[])</i>	text	
<i>_vol(char[], name[])</i>	text	
<i>_vol(char[])</i>	text	
<i>action(anyelement, sql text)</i>	anyelement	
<i>action(anyelement, sql text[])</i>	anyelement	
<i>add_array(anyarray, anyarray)</i>	anyarray	
<i>add_result(boolean, boolean, text, text, text)</i>	integer	
<i>alike(anyelement, text, text)</i>	text	
<i>alike(anyelement, text)</i>	text	
<i>any_column_privs_are(char[], char[], char[], name[], text)</i>	text	
<i>any_column_privs_are(char[], char[], char[], name[])</i>	text	
<i>any_column_privs_are(char[], char[], name[], text)</i>	text	
<i>any_column_privs_are(char[], char[], name[])</i>	text	
<i>array_sum(int4[])</i>	bigint	
<i>array_sum(int8[])</i>	numeric	
<i>array_sum(numeric[])</i>	numeric	
<i>array_sum(float8[])</i>	double precision	
<i>array_to_char(anyarray, format text)</i>	text[]	
<i>bag_eq(text, text, text)</i>	text	
<i>bag_eq(text, text)</i>	text	
<i>bag_eq(text, anyarray, text)</i>	text	
<i>bag_eq(text, anyarray)</i>	text	
<i>bag_has(text, text, text)</i>	text	
<i>bag_has(text, text)</i>	text	
<i>bag_hasnt(text, text, text)</i>	text	
<i>bag_hasnt(text, text)</i>	text	
<i>bag_ne(text, text, text)</i>	text	
<i>bag_ne(text, text)</i>	text	
<i>bag_ne(text, anyarray, text)</i>	text	
<i>bag_ne(text, anyarray)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>can(char[], name[], text)</i>	text	
<i>can(char[], name[])</i>	text	
<i>can(name[], text)</i>	text	
<i>can(name[])</i>	text	
<i>cast_context_is(char[], char[], text, text)</i>	text	
<i>cast_context_is(char[], char[], text)</i>	text	
<i>casts_are(text[], text)</i>	text	
<i>casts_are(text[])</i>	text	
<i>check_test(text, boolean, text, text, text, boolean)</i>	text	
<i>check_test(text, boolean, text, text, text)</i>	text	
<i>check_test(text, boolean, text, text)</i>	text	
<i>check_test(text, boolean, text)</i>	text	
<i>check_test(text, boolean)</i>	text	
<i>cmp_ok(anyelement, text, anyelement, text)</i>	text	
<i>cmp_ok(anyelement, text, anyelement)</i>	text	
<i>col_default_is(char[], char[], char[], anyelement, text)</i>	text	
<i>col_default_is(char[], char[], char[], text, text)</i>	text	
<i>col_default_is(char[], char[], anyelement, text)</i>	text	
<i>col_default_is(char[], char[], text, text)</i>	text	
<i>col_default_is(char[], char[], anyelement)</i>	text	
<i>col_default_is(char[], char[], text)</i>	text	
<i>col_has_check(char[], char[], name[], text)</i>	text	
<i>col_has_check(char[], name[], text)</i>	text	
<i>col_has_check(char[], name[])</i>	text	
<i>col_has_check(char[], char[], char[], text)</i>	text	
<i>col_has_check(char[], char[], text)</i>	text	
<i>col_has_check(char[], char[])</i>	text	
<i>col_has_default(char[], char[], char[], text)</i>	text	
<i>col_has_default(char[], char[], text)</i>	text	
<i>col_has_default(char[], char[])</i>	text	
<i>col_hasnt_default(char[], char[], char[], text)</i>	text	
<i>col_hasnt_default(char[], char[], text)</i>	text	
<i>col_hasnt_default(char[], char[])</i>	text	
<i>col_is_fk(char[], char[], name[], text)</i>	text	
<i>col_is_fk(char[], name[], text)</i>	text	
<i>col_is_fk(char[], name[])</i>	text	
<i>col_is_fk(char[], char[], char[], text)</i>	text	
<i>col_is_fk(char[], char[], text)</i>	text	
<i>col_is_fk(char[], char[])</i>	text	
<i>col_is_null(char[], char[], char[], text)</i>	text	
<i>col_is_null(char[], char[], char[])</i>	text	
<i>col_is_null(char[], char[])</i>	text	
<i>col_is_pk(char[], char[], name[], text)</i>	text	
<i>col_is_pk(char[], name[], text)</i>	text	
<i>col_is_pk(char[], name[])</i>	text	
<i>col_is_pk(char[], char[], char[], text)</i>	text	
<i>col_is_pk(char[], char[], text)</i>	text	
<i>col_is_pk(char[], char[])</i>	text	
<i>col_is_unique(char[], char[], name[], text)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>col_is_unique(char[], char[], name[])</i>	text	
<i>col_is_unique(char[], char[], char[])</i>	text	
<i>col_is_unique(char[], name[], text)</i>	text	
<i>col_is_unique(char[], name[])</i>	text	
<i>col_is_unique(char[], char[], char[], text)</i>	text	
<i>col_is_unique(char[], char[], text)</i>	text	
<i>col_is_unique(char[], char[])</i>	text	
<i>col_isnt_fk(char[], char[], name[], text)</i>	text	
<i>col_isnt_fk(char[], name[], text)</i>	text	
<i>col_isnt_fk(char[], name[])</i>	text	
<i>col_isnt_fk(char[], char[], char[], text)</i>	text	
<i>col_isnt_fk(char[], char[], text)</i>	text	
<i>col_isnt_fk(char[], char[])</i>	text	
<i>col_isnt_pk(char[], char[], name[], text)</i>	text	
<i>col_isnt_pk(char[], name[], text)</i>	text	
<i>col_isnt_pk(char[], name[])</i>	text	
<i>col_isnt_pk(char[], char[], char[], text)</i>	text	
<i>col_isnt_pk(char[], char[], text)</i>	text	
<i>col_isnt_pk(char[], char[])</i>	text	
<i>col_not_null(char[], char[], char[], text)</i>	text	
<i>col_not_null(char[], char[], text)</i>	text	
<i>col_not_null(char[], char[])</i>	text	
<i>col_type_is(char[], char[], char[], char[], text, text)</i>	text	
<i>col_type_is(char[], char[], char[], char[], text)</i>	text	
<i>col_type_is(char[], char[], char[], text, text)</i>	text	
<i>col_type_is(char[], char[], char[], text)</i>	text	
<i>col_type_is(char[], char[], text, text)</i>	text	
<i>col_type_is(char[], char[], text)</i>	text	
<i>collect_tap(text[])</i>	RECORD()	
<i>collect_tap(varchar[])</i>	text	
<i>column_names(namespace char[], table char[])</i>	char[]	
<i>column_privs_are(char[], char[], char[], char[], name[], text)</i>	text	
<i>column_privs_are(char[], char[], char[], char[], name[])</i>	text	
<i>column_privs_are(char[], char[], char[], name[], text)</i>	text	
<i>column_privs_are(char[], char[], char[], name[])</i>	text	
<i>columns_are(char[], char[], name[], text)</i>	text	
<i>columns_are(char[], char[], name[])</i>	text	
<i>columns_are(char[], name[], text)</i>	text	
<i>columns_are(char[], name[])</i>	text	
<i>composite_owner_is(char[], char[], char[], text)</i>	text	
<i>composite_owner_is(char[], char[], char[])</i>	text	
<i>composite_owner_is(char[], char[], text)</i>	text	
<i>composite_owner_is(char[], char[])</i>	text	
<i>database_privs_are(char[], char[], name[], text)</i>	text	
<i>database_privs_are(char[], char[], name[])</i>	text	
<i>db_owner_is(char[], char[], text)</i>	text	
<i>db_owner_is(char[], char[])</i>	text	
<i>diag(msg text)</i>	text	
<i>diag(msg anyelement)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>diag(text[])</i>	RECORD()	
<i>diag(anyarray)</i>	RECORD()	
<i>diag_test_name(text)</i>	text	
<i>display_oper(char[], oid)</i>	text	
<i>divide_array(anyarray, anyelement)</i>	anyarray	
<i>divide_array(anyarray, anyarray)</i>	anyarray	
<i>do_tap(char[], text)</i>	text	
<i>do_tap(char[])</i>	text	
<i>do_tap(text)</i>	text	
<i>do_tap()</i>	text	
<i>doesn't_imatch(anyelement, text, text)</i>	text	
<i>doesn't_imatch(anyelement, text)</i>	text	
<i>doesn't_match(anyelement, text, text)</i>	text	
<i>doesn't_match(anyelement, text)</i>	text	
<i>domain_type_is(char[], text, char[], text, text)</i>	text	
<i>domain_type_is(char[], text, char[], text)</i>	text	
<i>domain_type_is(char[], text, text, text)</i>	text	
<i>domain_type_is(char[], text, text)</i>	text	
<i>domain_type_is(text, text, text)</i>	text	
<i>domain_type_is(text, text)</i>	text	
<i>domain_type_isnt(char[], text, char[], text, text)</i>	text	
<i>domain_type_isnt(char[], text, char[], text)</i>	text	
<i>domain_type_isnt(char[], text, text, text)</i>	text	
<i>domain_type_isnt(char[], text, text)</i>	text	
<i>domain_type_isnt(text, text, text)</i>	text	
<i>domain_type_isnt(text, text)</i>	text	
<i>domains_are(char[], name[], text)</i>	text	
<i>domains_are(char[], name[])</i>	text	
<i>domains_are(name[], text)</i>	text	
<i>domains_are(name[])</i>	text	
<i>drop_changes_view(attribute_directory.attribute_store)</i>	attribute_directory.attribute_store	
<i>enum_has_labels(char[], char[], name[])</i>	text	
<i>enum_has_labels(char[], char[], name[], text)</i>	text	
<i>enum_has_labels(char[], name[], text)</i>	text	
<i>enum_has_labels(char[], name[])</i>	text	
<i>enums_are(char[], name[], text)</i>	text	
<i>enums_are(char[], name[])</i>	text	
<i>enums_are(name[], text)</i>	text	
<i>enums_are(name[])</i>	text	
<i>fail(text)</i>	text	
<i>fail()</i>	text	
<i>fdw_privs_are(char[], char[], name[], text)</i>	text	
<i>fdw_privs_are(char[], char[], name[])</i>	text	
<i>findfuncs(char[], text, text)</i>	text[]	
<i>findfuncs(char[], text)</i>	text[]	
<i>findfuncs(text, text)</i>	text[]	
<i>findfuncs(text)</i>	text[]	
<i>finish()</i>	text	
<i>first(anyelement)</i>	anyelement	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>fk_ok(char[], char[], name[], char[], char[], name[], text)</i>	text	
<i>fk_ok(char[], name[], char[], name[], text)</i>	text	
<i>fk_ok(char[], char[], name[], char[], char[], name[])</i>	text	
<i>fk_ok(char[], name[], char[], name[])</i>	text	
<i>fk_ok(char[], char[], char[], char[], char[], char[], text)</i>	text	
<i>fk_ok(char[], char[], char[], char[], char[], text)</i>	text	
<i>fk_ok(char[], char[], char[], char[], text)</i>	text	
<i>fk_ok(char[], char[], char[], char[])</i>	text	
<i>foreign_table_owner_is(char[], char[])</i>	text	
<i>foreign_table_owner_is(char[], char[], char[], text)</i>	text	
<i>foreign_table_owner_is(char[], char[], char[])</i>	text	
<i>foreign_table_owner_is(char[], char[], text)</i>	text	
<i>foreign_tables_are(char[], name[], text)</i>	text	
<i>foreign_tables_are(name[], text)</i>	text	
<i>foreign_tables_are(char[], name[])</i>	text	
<i>foreign_tables_are(name[])</i>	text	
<i>fst(anyelement, anyelement)</i>	anyelement	
<i>function_lang_is(char[], char[], name[], char[], text)</i>	text	
<i>function_lang_is(char[], char[], name[], char[])</i>	text	
<i>function_lang_is(char[], char[], char[], text)</i>	text	
<i>function_lang_is(char[], char[], char[])</i>	text	
<i>function_lang_is(char[], name[], char[], text)</i>	text	
<i>function_lang_is(char[], name[], char[])</i>	text	
<i>function_lang_is(char[], char[], text)</i>	text	
<i>function_lang_is(char[], char[])</i>	text	
<i>function_owner_is(char[], char[], name[], char[], text)</i>	text	
<i>function_owner_is(char[], char[], name[], char[])</i>	text	
<i>function_owner_is(char[], name[], char[], text)</i>	text	
<i>function_owner_is(char[], name[], char[])</i>	text	
<i>function_privs_are(char[], char[], name[], char[], name[], text)</i>	text	
<i>function_privs_are(char[], char[], name[], char[], name[])</i>	text	
<i>function_privs_are(char[], name[], char[], name[], text)</i>	text	
<i>function_privs_are(char[], name[], char[], name[])</i>	text	
<i>function_returns(char[], char[], name[], text, text)</i>	text	
<i>function_returns(char[], char[], name[], text)</i>	text	
<i>function_returns(char[], char[], text, text)</i>	text	
<i>function_returns(char[], char[], text)</i>	text	
<i>function_returns(char[], name[], text, text)</i>	text	
<i>function_returns(char[], name[], text)</i>	text	
<i>function_returns(char[], text, text)</i>	text	
<i>function_returns(char[], text)</i>	text	
<i>functions_are(char[], name[], text)</i>	text	
<i>functions_are(char[], name[])</i>	text	
<i>functions_are(name[], text)</i>	text	
<i>functions_are(name[])</i>	text	
<i>groups_are(name[], text)</i>	text	
<i>groups_are(name[])</i>	text	
<i>has_cast(char[], char[], char[], char[], text)</i>	text	
<i>has_cast(char[], char[], char[], char[])</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>has_cast(char[], char[], char[], text)</i>	text	
<i>has_cast(char[], char[], char[])</i>	text	
<i>has_cast(char[], char[], text)</i>	text	
<i>has_cast(char[], char[])</i>	text	
<i>has_check(char[], char[], text)</i>	text	
<i>has_check(char[], text)</i>	text	
<i>has_check(char[])</i>	text	
<i>has_column(char[], char[], char[], text)</i>	text	
<i>has_column(char[], char[], text)</i>	text	
<i>has_column(char[], char[])</i>	text	
<i>has_composite(char[], char[], text)</i>	text	
<i>has_composite(char[], text)</i>	text	
<i>has_composite(char[])</i>	text	
<i>has_domain(char[], char[], text)</i>	text	
<i>has_domain(char[], char[])</i>	text	
<i>has_domain(char[], text)</i>	text	
<i>has_domain(char[])</i>	text	
<i>has_enum(char[], char[], text)</i>	text	
<i>has_enum(char[], char[])</i>	text	
<i>has_enum(char[], text)</i>	text	
<i>has_enum(char[])</i>	text	
<i>has_fk(char[], char[], text)</i>	text	
<i>has_fk(char[], text)</i>	text	
<i>has_fk(char[])</i>	text	
<i>has_foreign_table(char[], char[], text)</i>	text	
<i>has_foreign_table(char[], char[])</i>	text	
<i>has_foreign_table(char[], text)</i>	text	
<i>has_foreign_table(char[])</i>	text	
<i>has_function(char[], char[], name[], text)</i>	text	
<i>has_function(char[], char[], name[])</i>	text	
<i>has_function(char[], char[], text)</i>	text	
<i>has_function(char[], char[])</i>	text	
<i>has_function(char[], name[], text)</i>	text	
<i>has_function(char[], name[])</i>	text	
<i>has_function(char[], text)</i>	text	
<i>has_function(char[])</i>	text	
<i>has_group(char[], text)</i>	text	
<i>has_group(char[])</i>	text	
<i>has_index(char[], char[], char[], name[], text)</i>	text	
<i>has_index(char[], char[], char[], name[])</i>	text	
<i>has_index(char[], char[], char[], char[], text)</i>	text	
<i>has_index(char[], char[], char[], char[])</i>	text	
<i>has_index(char[], char[], name[], text)</i>	text	
<i>has_index(char[], char[], name[])</i>	text	
<i>has_index(char[], char[], char[], text)</i>	text	
<i>has_index(char[], char[], char[])</i>	text	
<i>has_index(char[], char[], text)</i>	text	
<i>has_index(char[], char[])</i>	text	
<i>has_language(char[], text)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>has_language(char[])</i>	text	
<i>has_lefttop(char[], char[], char[], char[], text)</i>	text	
<i>has_lefttop(char[], char[], char[], char[])</i>	text	
<i>has_lefttop(char[], char[], char[], text)</i>	text	
<i>has_lefttop(char[], char[], char[])</i>	text	
<i>has_lefttop(char[], char[], text)</i>	text	
<i>has_lefttop(char[], char[])</i>	text	
<i>has_materialized_view(char[], char[], text)</i>	text	
<i>has_materialized_view(char[], text)</i>	text	
<i>has_materialized_view(char[])</i>	text	
<i>has_opclass(char[], char[], text)</i>	text	
<i>has_opclass(char[], char[])</i>	text	
<i>has_opclass(char[], text)</i>	text	
<i>has_opclass(char[])</i>	text	
<i>has_operator(char[], char[], char[], char[], char[], text)</i>	text	
<i>has_operator(char[], char[], char[], char[], char[])</i>	text	
<i>has_operator(char[], char[], char[], char[], text)</i>	text	
<i>has_operator(char[], char[], char[], char[])</i>	text	
<i>has_operator(char[], char[], char[], text)</i>	text	
<i>has_operator(char[], char[], char[])</i>	text	
<i>has_pk(char[], char[], text)</i>	text	
<i>has_pk(char[], text)</i>	text	
<i>has_pk(char[])</i>	text	
<i>has_relation(char[], char[], text)</i>	text	
<i>has_relation(char[], text)</i>	text	
<i>has_relation(char[])</i>	text	
<i>has_righttop(char[], char[], char[], char[], text)</i>	text	
<i>has_righttop(char[], char[], char[], char[])</i>	text	
<i>has_righttop(char[], char[], char[], text)</i>	text	
<i>has_righttop(char[], char[], char[])</i>	text	
<i>has_righttop(char[], char[], text)</i>	text	
<i>has_righttop(char[], char[])</i>	text	
<i>has_role(char[], text)</i>	text	
<i>has_role(char[])</i>	text	
<i>has_rule(char[], char[], char[], text)</i>	text	
<i>has_rule(char[], char[], char[])</i>	text	
<i>has_rule(char[], char[], text)</i>	text	
<i>has_rule(char[], char[])</i>	text	
<i>has_schema(char[], text)</i>	text	
<i>has_schema(char[])</i>	text	
<i>has_sequence(char[], char[], text)</i>	text	
<i>has_sequence(char[], char[])</i>	text	
<i>has_sequence(char[], text)</i>	text	
<i>has_sequence(char[])</i>	text	
<i>has_table(char[], char[], text)</i>	text	
<i>has_table(char[], char[])</i>	text	
<i>has_table(char[], text)</i>	text	
<i>has_table(char[])</i>	text	
<i>has_tablespace(char[], text, text)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>has_tablespace(char[], text)</i>	text	
<i>has_tablespace(char[])</i>	text	
<i>has_trigger(char[], char[], char[], text)</i>	text	
<i>has_trigger(char[], char[], char[])</i>	text	
<i>has_trigger(char[], char[], text)</i>	text	
<i>has_trigger(char[], char[])</i>	text	
<i>has_type(char[], char[], text)</i>	text	
<i>has_type(char[], char[])</i>	text	
<i>has_type(char[], text)</i>	text	
<i>has_type(char[])</i>	text	
<i>has_unique(text, text, text)</i>	text	
<i>has_unique(text, text)</i>	text	
<i>has_unique(text)</i>	text	
<i>has_user(char[], text)</i>	text	
<i>has_user(char[])</i>	text	
<i>has_view(char[], char[], text)</i>	text	
<i>has_view(char[], text)</i>	text	
<i>has_view(char[])</i>	text	
<i>hasnt_cast(char[], char[], char[], char[], text)</i>	text	
<i>hasnt_cast(char[], char[], char[], char[])</i>	text	
<i>hasnt_cast(char[], char[], char[], text)</i>	text	
<i>hasnt_cast(char[], char[], char[])</i>	text	
<i>hasnt_cast(char[], char[], text)</i>	text	
<i>hasnt_cast(char[], char[])</i>	text	
<i>hasnt_column(char[], char[], char[], text)</i>	text	
<i>hasnt_column(char[], char[], text)</i>	text	
<i>hasnt_column(char[], char[])</i>	text	
<i>hasnt_composite(char[], char[], text)</i>	text	
<i>hasnt_composite(char[], text)</i>	text	
<i>hasnt_composite(char[])</i>	text	
<i>hasnt_domain(char[], char[], text)</i>	text	
<i>hasnt_domain(char[], char[])</i>	text	
<i>hasnt_domain(char[], text)</i>	text	
<i>hasnt_domain(char[])</i>	text	
<i>hasnt_enum(char[], char[], text)</i>	text	
<i>hasnt_enum(char[], char[])</i>	text	
<i>hasnt_enum(char[], text)</i>	text	
<i>hasnt_enum(char[])</i>	text	
<i>hasnt_fk(char[], char[], text)</i>	text	
<i>hasnt_fk(char[], text)</i>	text	
<i>hasnt_fk(char[])</i>	text	
<i>hasnt_foreign_table(char[], char[], text)</i>	text	
<i>hasnt_foreign_table(char[], char[])</i>	text	
<i>hasnt_foreign_table(char[], text)</i>	text	
<i>hasnt_foreign_table(char[])</i>	text	
<i>hasnt_function(char[], char[], name[], text)</i>	text	
<i>hasnt_function(char[], char[], name[])</i>	text	
<i>hasnt_function(char[], char[], text)</i>	text	
<i>hasnt_function(char[], char[])</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>hasnt_function(char[], name[], text)</i>	text	
<i>hasnt_function(char[], name[])</i>	text	
<i>hasnt_function(char[], text)</i>	text	
<i>hasnt_function(char[])</i>	text	
<i>hasnt_group(char[], text)</i>	text	
<i>hasnt_group(char[])</i>	text	
<i>hasnt_index(char[], char[], char[], text)</i>	text	
<i>hasnt_index(char[], char[], char[])</i>	text	
<i>hasnt_index(char[], char[], text)</i>	text	
<i>hasnt_index(char[], char[])</i>	text	
<i>hasnt_language(char[])</i>	text	
<i>hasnt_language(char[], text)</i>	text	
<i>hasnt_materialized_view(char[], char[], text)</i>	text	
<i>hasnt_materialized_view(char[], text)</i>	text	
<i>hasnt_materialized_view(char[])</i>	text	
<i>hasnt_opclass(char[], char[], text)</i>	text	
<i>hasnt_opclass(char[], char[])</i>	text	
<i>hasnt_opclass(char[], text)</i>	text	
<i>hasnt_opclass(char[])</i>	text	
<i>hasnt_pk(char[])</i>	text	
<i>hasnt_pk(char[], char[], text)</i>	text	
<i>hasnt_pk(char[], text)</i>	text	
<i>hasnt_relation(char[], char[], text)</i>	text	
<i>hasnt_relation(char[], text)</i>	text	
<i>hasnt_relation(char[])</i>	text	
<i>hasnt_role(char[], text)</i>	text	
<i>hasnt_role(char[])</i>	text	
<i>hasnt_rule(char[], char[], char[], text)</i>	text	
<i>hasnt_rule(char[], char[], char[])</i>	text	
<i>hasnt_rule(char[], char[], text)</i>	text	
<i>hasnt_rule(char[], char[])</i>	text	
<i>hasnt_schema(char[], text)</i>	text	
<i>hasnt_schema(char[])</i>	text	
<i>hasnt_sequence(char[], char[], text)</i>	text	
<i>hasnt_sequence(char[], text)</i>	text	
<i>hasnt_sequence(char[])</i>	text	
<i>hasnt_table(char[], char[], text)</i>	text	
<i>hasnt_table(char[], char[])</i>	text	
<i>hasnt_table(char[], text)</i>	text	
<i>hasnt_table(char[])</i>	text	
<i>hasnt_tablespace(char[])</i>	text	
<i>hasnt_tablespace(char[], text)</i>	text	
<i>hasnt_trigger(char[], char[], char[], text)</i>	text	
<i>hasnt_trigger(char[], char[], char[])</i>	text	
<i>hasnt_trigger(char[], char[], text)</i>	text	
<i>hasnt_trigger(char[], char[])</i>	text	
<i>hasnt_type(char[], char[], text)</i>	text	
<i>hasnt_type(char[], char[])</i>	text	
<i>hasnt_type(char[], text)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>hasnt_type(char[])</i>	text	
<i>hasnt_user(char[], text)</i>	text	
<i>hasnt_user(char[])</i>	text	
<i>hasnt_view(char[], char[], text)</i>	text	
<i>hasnt_view(char[], text)</i>	text	
<i>hasnt_view(char[])</i>	text	
<i>ialike(anyelement, text, text)</i>	text	
<i>ialike(anyelement, text)</i>	text	
<i>imatches(anyelement, text, text)</i>	text	
<i>imatches(anyelement, text)</i>	text	
<i>in_todo()</i>	boolean	
<i>index_is_primary(char[], char[], char[], text)</i>	text	
<i>index_is_primary(char[], char[], char[])</i>	text	
<i>index_is_primary(char[], char[])</i>	text	
<i>index_is_primary(char[])</i>	text	
<i>index_is_type(char[], char[], char[], char[], text)</i>	text	
<i>index_is_type(char[], char[], char[], char[])</i>	text	
<i>index_is_type(char[], char[], char[])</i>	text	
<i>index_is_type(char[], char[])</i>	text	
<i>index_is_unique(char[], char[], char[], text)</i>	text	
<i>index_is_unique(char[], char[], char[])</i>	text	
<i>index_is_unique(char[], char[])</i>	text	
<i>index_is_unique(char[])</i>	text	
<i>index_owner_is(char[], char[], char[], char[], text)</i>	text	
<i>index_owner_is(char[], char[], char[], char[])</i>	text	
<i>index_owner_is(char[], char[], char[], text)</i>	text	
<i>index_owner_is(char[], char[], char[])</i>	text	
<i>indexes_are(char[], char[], name[], text)</i>	text	
<i>indexes_are(char[], char[], name[])</i>	text	
<i>indexes_are(char[], name[], text)</i>	text	
<i>indexes_are(char[], name[])</i>	text	
<i>integer_to_array(value integer)</i>	int4[]	
<i>is(anyelement, anyelement, text)</i>	text	
<i>is(anyelement, anyelement)</i>	text	
<i>is_aggregate(char[], char[], name[], text)</i>	text	
<i>is_aggregate(char[], char[], name[])</i>	text	
<i>is_aggregate(char[], char[], text)</i>	text	
<i>is_aggregate(char[], char[])</i>	text	
<i>is_aggregate(char[], name[], text)</i>	text	
<i>is_aggregate(char[], name[])</i>	text	
<i>is_aggregate(char[], text)</i>	text	
<i>is_aggregate(char[])</i>	text	
<i>is_clustered(char[], char[], char[], text)</i>	text	
<i>is_clustered(char[], char[], char[])</i>	text	
<i>is_clustered(char[], char[])</i>	text	
<i>is_clustered(char[])</i>	text	
<i>is_definer(char[], char[], name[], text)</i>	text	
<i>is_definer(char[], char[], name[])</i>	text	
<i>is_definer(char[], char[], text)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>is_definer(char[], char[])</i>	text	
<i>is_definer(char[], name[], text)</i>	text	
<i>is_definer(char[], name[])</i>	text	
<i>is_definer(char[], text)</i>	text	
<i>is_definer(char[])</i>	text	
<i>is_empty(text, text)</i>	text	
<i>is_empty(text)</i>	text	
<i>is_member_of(char[], char[])</i>	text	
<i>is_member_of(char[], name[], text)</i>	text	
<i>is_member_of(char[], char[], text)</i>	text	
<i>is_member_of(char[], name[])</i>	text	
<i>is_strict(char[], char[], name[], text)</i>	text	
<i>is_strict(char[], char[], name[])</i>	text	
<i>is_strict(char[], char[], text)</i>	text	
<i>is_strict(char[], char[])</i>	text	
<i>is_strict(char[], name[], text)</i>	text	
<i>is_strict(char[], name[])</i>	text	
<i>is_strict(char[], text)</i>	text	
<i>is_strict(char[])</i>	text	
<i>is_superuser(char[], text)</i>	text	
<i>is_superuser(char[])</i>	text	
<i>isa_ok(anyelement, regtype, text)</i>	text	
<i>isa_ok(anyelement, regtype)</i>	text	
<i>isnt(anyelement, anyelement, text)</i>	text	
<i>isnt(anyelement, anyelement)</i>	text	
<i>isnt_empty(text, text)</i>	text	
<i>isnt_empty(text)</i>	text	
<i>isnt_strict(char[], char[], name[], text)</i>	text	
<i>isnt_strict(char[], char[], name[])</i>	text	
<i>isnt_strict(char[], char[], text)</i>	text	
<i>isnt_strict(char[], char[])</i>	text	
<i>isnt_strict(char[], name[], text)</i>	text	
<i>isnt_strict(char[], name[])</i>	text	
<i>isnt_strict(char[], text)</i>	text	
<i>isnt_strict(char[])</i>	text	
<i>isnt_superuser(char[], text)</i>	text	
<i>isnt_superuser(char[])</i>	text	
<i>language_is_trusted(char[], text)</i>	text	
<i>language_is_trusted(char[])</i>	text	
<i>language_owner_is(char[], char[], text)</i>	text	
<i>language_owner_is(char[], char[])</i>	text	
<i>language_privs_are(char[], char[], name[], text)</i>	text	
<i>language_privs_are(char[], char[], name[])</i>	text	
<i>languages_are(name[], text)</i>	text	
<i>languages_are(name[])</i>	text	
<i>last(anyelement)</i>	anyelement	
<i>lives_ok(text, text)</i>	text	
<i>lives_ok(text)</i>	text	
<i>matches(anyelement, text, text)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>matches(anyelement, text)</i>	text	
<i>materialized_view_owner_is(char[], char[], char[], text)</i>	text	
<i>materialized_view_owner_is(char[], char[], char[])</i>	text	
<i>materialized_view_owner_is(char[], char[], text)</i>	text	
<i>materialized_view_owner_is(char[], char[])</i>	text	
<i>materialized_views_are(char[], name[], text)</i>	text	
<i>materialized_views_are(name[], text)</i>	text	
<i>materialized_views_are(char[], name[])</i>	text	
<i>materialized_views_are(name[])</i>	text	
<i>multiply_array(anyarray, anyelement)</i>	anyarray	
<i>multiply_array(anyarray, anyarray)</i>	anyarray	
<i>no_plan()</i>	boolean	
<i>num_failed()</i>	integer	
<i>ok(boolean, text)</i>	text	
<i>ok(boolean)</i>	text	
<i>opclass_owner_is(char[], char[], char[], text)</i>	text	
<i>opclass_owner_is(char[], char[], char[])</i>	text	
<i>opclass_owner_is(char[], char[], text)</i>	text	
<i>opclass_owner_is(char[], char[])</i>	text	
<i>opclasses_are(char[], name[], text)</i>	text	
<i>opclasses_are(char[], name[])</i>	text	
<i>opclasses_are(name[], text)</i>	text	
<i>opclasses_are(name[])</i>	text	
<i>operators_are(char[], text[], text)</i>	text	
<i>operators_are(char[], text[])</i>	text	
<i>operators_are(text[], text)</i>	text	
<i>operators_are(text[])</i>	text	
<i>os_name()</i>	text	
<i>pass(text)</i>	text	
<i>pass()</i>	text	
<i>performs_ok(text, numeric, text)</i>	text	
<i>performs_ok(text, numeric)</i>	text	
<i>performs_within(text, numeric, numeric, integer, text)</i>	text	
<i>performs_within(text, numeric, numeric, integer)</i>	text	
<i>performs_within(text, numeric, numeric, text)</i>	text	
<i>performs_within(text, numeric, numeric)</i>	text	
<i>pg_version()</i>	text	
<i>pg_version_num()</i>	integer	
<i>pgtap_version()</i>	numeric	
<i>plan(integer)</i>	text	
<i>proreftype(oid)</i>	oid	
<i>relation_owner_is(char[], char[], char[], text)</i>	text	
<i>relation_owner_is(char[], char[], char[])</i>	text	
<i>relation_owner_is(char[], char[], text)</i>	text	
<i>relation_owner_is(char[], char[])</i>	text	
<i>results_eq(refcursor, refcursor)</i>	text	
<i>results_eq(refcursor, refcursor)</i>	text	
<i>results_eq(text, text, text)</i>	text	
<i>results_eq(text, text)</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>results_eq(text, anyarray, text)</i>	text	
<i>results_eq(text, anyarray)</i>	text	
<i>results_eq(text, refcursor, text)</i>	text	
<i>results_eq(text, refcursor)</i>	text	
<i>results_eq(refcursor, text, text)</i>	text	
<i>results_eq(refcursor, text)</i>	text	
<i>results_eq(refcursor, anyarray, text)</i>	text	
<i>results_eq(refcursor, anyarray)</i>	text	
<i>results_ne(refcursor, refcursor, text)</i>	text	
<i>results_ne(refcursor, refcursor)</i>	text	
<i>results_ne(text, text, text)</i>	text	
<i>results_ne(text, text)</i>	text	
<i>results_ne(text, anyarray, text)</i>	text	
<i>results_ne(text, anyarray)</i>	text	
<i>results_ne(text, refcursor, text)</i>	text	
<i>results_ne(text, refcursor)</i>	text	
<i>results_ne(refcursor, text, text)</i>	text	
<i>results_ne(refcursor, text)</i>	text	
<i>results_ne(refcursor, anyarray, text)</i>	text	
<i>results_ne(refcursor, anyarray)</i>	text	
<i>roles_are(name[], text)</i>	text	
<i>roles_are(name[])</i>	text	
<i>row_eq(text, anyelement, text)</i>	text	
<i>row_eq(text, anyelement)</i>	text	
<i>rule_is_instead(char[], char[], char[], text)</i>	text	
<i>rule_is_instead(char[], char[], char[])</i>	text	
<i>rule_is_instead(char[], char[], text)</i>	text	
<i>rule_is_instead(char[], char[])</i>	text	
<i>rule_is_on(char[], char[], char[], text, text)</i>	text	
<i>rule_is_on(char[], char[], char[], text)</i>	text	
<i>rule_is_on(char[], char[], text, text)</i>	text	
<i>rule_is_on(char[], char[], text)</i>	text	
<i>rules_are(char[], char[], name[], text)</i>	text	
<i>rules_are(char[], char[], name[])</i>	text	
<i>rules_are(char[], name[], text)</i>	text	
<i>rules_are(char[], name[])</i>	text	
<i>runtests(char[], text)</i>	text	
<i>runtests(char[])</i>	text	
<i>runtests(text)</i>	text	
<i>runtests()</i>	text	
<i>safe_division(numerator anyelement, denominator anyelement)</i>	anyelement	
<i>schema_owner_is(char[], char[], text)</i>	text	
<i>schema_owner_is(char[], char[])</i>	text	
<i>schema_privs_are(char[], char[], name[], text)</i>	text	
<i>schema_privs_are(char[], char[], name[])</i>	text	
<i>schemas_are(name[], text)</i>	text	
<i>schemas_are(name[])</i>	text	
<i>sequence_owner_is(char[], char[], char[], text)</i>	text	
<i>sequence_owner_is(char[], char[], char[])</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>sequence_owner_is(char[], char[], text)</i>	text	
<i>sequence_owner_is(char[], char[])</i>	text	
<i>sequence_privs_are(char[], char[], char[], name[], text)</i>	text	
<i>sequence_privs_are(char[], char[], char[], name[])</i>	text	
<i>sequence_privs_are(char[], char[], name[], text)</i>	text	
<i>sequence_privs_are(char[], char[], name[])</i>	text	
<i>sequences_are(char[], name[], text)</i>	text	
<i>sequences_are(name[], text)</i>	text	
<i>sequences_are(char[], name[])</i>	text	
<i>sequences_are(name[])</i>	text	
<i>server_privs_are(char[], char[], name[], text)</i>	text	
<i>server_privs_are(char[], char[], name[])</i>	text	
<i>set_eq(text, text, text)</i>	text	
<i>set_eq(text, text)</i>	text	
<i>set_eq(text, anyarray, text)</i>	text	
<i>set_eq(text, anyarray)</i>	text	
<i>set_has(text, text, text)</i>	text	
<i>set_has(text, text)</i>	text	
<i>set_hasnt(text, text, text)</i>	text	
<i>set_hasnt(text, text)</i>	text	
<i>set_ne(text, text, text)</i>	text	
<i>set_ne(text, text)</i>	text	
<i>set_ne(text, anyarray, text)</i>	text	
<i>set_ne(text, anyarray)</i>	text	
<i>skip(why text, how_many integer)</i>	text	
<i>skip(text)</i>	text	
<i>skip(integer, text)</i>	text	
<i>skip(integer)</i>	text	
<i>smallint_to_array(value smallint)</i>	int2[]	
<i>smallint_to_timestamp_with_time_zone(smallint)</i>	timestamp with time zone	
<i>smallint_to_timestamp_without_time_zone(smallint)</i>	timestamp without time zone	
<i>snd(anyelement, anyelement)</i>	anyelement	
<i>sum_array(anyarray)</i>	anyarray	
<i>table_owner_is(char[], char[], char[], text)</i>	text	
<i>table_owner_is(char[], char[], char[])</i>	text	
<i>table_owner_is(char[], char[], text)</i>	text	
<i>table_owner_is(char[], char[])</i>	text	
<i>table_privs_are(char[], char[], char[], name[], text)</i>	text	
<i>table_privs_are(char[], char[], char[], name[])</i>	text	
<i>table_privs_are(char[], char[], name[], text)</i>	text	
<i>table_privs_are(char[], char[], name[])</i>	text	
<i>tables_are(char[], name[], text)</i>	text	
<i>tables_are(name[], text)</i>	text	
<i>tables_are(char[], name[])</i>	text	
<i>tables_are(name[])</i>	text	
<i>tablespace_owner_is(char[], char[], text)</i>	text	
<i>tablespace_owner_is(char[], char[])</i>	text	
<i>tablespace_privs_are(char[], char[], name[], text)</i>	text	
<i>tablespace_privs_are(char[], char[], name[])</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>tablespaces_are(name[])</i>	text	
<i>tablespaces_are(name[], text)</i>	text	
<i>throws_ilike(text, text, text)</i>	text	
<i>throws_ilike(text, text)</i>	text	
<i>throws_imatching(text, text, text)</i>	text	
<i>throws_imatching(text, text)</i>	text	
<i>throws_like(text, text, text)</i>	text	
<i>throws_like(text, text)</i>	text	
<i>throws_matching(text, text, text)</i>	text	
<i>throws_matching(text, text)</i>	text	
<i>throws_ok(text, character, text, text)</i>	text	
<i>throws_ok(text, text, text)</i>	text	
<i>throws_ok(text, text)</i>	text	
<i>throws_ok(text)</i>	text	
<i>throws_ok(text, integer, text, text)</i>	text	
<i>throws_ok(text, integer, text)</i>	text	
<i>throws_ok(text, integer)</i>	text	
<i>to_pdf(text)</i>	int4[]	
<i>todo(why text, how_many integer)</i>	boolean	
<i>todo(how_many integer, why text)</i>	boolean	
<i>todo(why text)</i>	boolean	
<i>todo(how_many integer)</i>	boolean	
<i>todo_end()</i>	boolean	
<i>todo_start(text)</i>	boolean	
<i>todo_start()</i>	boolean	
<i>trigger_is(char[], char[], char[], char[], char[], text)</i>	text	
<i>trigger_is(char[], char[], char[], char[], char[])</i>	text	
<i>trigger_is(char[], char[], char[], text)</i>	text	
<i>trigger_is(char[], char[], char[])</i>	text	
<i>triggers_are(char[], char[], name[], text)</i>	text	
<i>triggers_are(char[], char[], name[])</i>	text	
<i>triggers_are(char[], name[], text)</i>	text	
<i>triggers_are(char[], name[])</i>	text	
<i>type_columns(oid)</i>	column_info	
<i>type_columns(namespace char[], type char[])</i>	column_info	
<i>type_owner_is(char[], char[], char[], text)</i>	text	
<i>type_owner_is(char[], char[], char[])</i>	text	
<i>type_owner_is(char[], char[], text)</i>	text	
<i>type_owner_is(char[], char[])</i>	text	
<i>types_are(char[], name[], text)</i>	text	
<i>types_are(char[], name[])</i>	text	
<i>types_are(name[], text)</i>	text	
<i>types_are(name[])</i>	text	
<i>unlike(anyelement, text, text)</i>	text	
<i>unlike(anyelement, text)</i>	text	
<i>unialike(anyelement, text, text)</i>	text	
<i>unialike(anyelement, text)</i>	text	
<i>users_are(name[], text)</i>	text	
<i>users_are(name[])</i>	text	

Table 3.5 – continued from previous page

Name	Return Type	Description
<i>view_exists(char[], char[])</i>	boolean	
<i>view_owner_is(char[], char[], char[], text)</i>	text	
<i>view_owner_is(char[], char[], char[])</i>	text	
<i>view_owner_is(char[], char[], text)</i>	text	
<i>view_owner_is(char[], char[])</i>	text	
<i>views_are(char[], name[], text)</i>	text	
<i>views_are(name[], text)</i>	text	
<i>views_are(char[], name[])</i>	text	
<i>views_are(name[])</i>	text	
<i>volatility_is(char[], char[], name[], text, text)</i>	text	
<i>volatility_is(char[], char[], name[], text)</i>	text	
<i>volatility_is(char[], char[], text, text)</i>	text	
<i>volatility_is(char[], char[], text)</i>	text	
<i>volatility_is(char[], name[], text, text)</i>	text	
<i>volatility_is(char[], name[], text)</i>	text	
<i>volatility_is(char[], text, text)</i>	text	
<i>volatility_is(char[], text)</i>	text	
<i>wal_location_to_int(text)</i>	bigint	Convert a textual WAL lo

`_add(text, integer, text) -> integer`

returns: integer

`_add(text, integer) -> integer`

returns: integer

`_agg(char[], char[], name[]) -> boolean`

returns: boolean

`_agg(char[], char[]) -> boolean`

returns: boolean

`_agg(char[], name[]) -> boolean`

returns: boolean

`_agg(char[]) -> boolean`

returns: boolean

`_alike(boolean, anyelement, text, text) -> text`

returns: text

`_are(text, name[], name[], text) -> text`

returns: text

`_areni(text, text[], text[], text) -> text`

returns: text

`_assets_are(text, text[], text[], text) -> text`

returns: text

`_cast_exists(char[], char[], char[], char[]) -> boolean`

returns: boolean

`_cast_exists(char[], char[], char[]) -> boolean`

returns: boolean

`_cast_exists(char[], char[]) -> boolean`

returns: boolean

`_cdi(char[], char[], char[], anyelement, text) -> text`

returns: text

`_cdi(char[], char[], anyelement, text) -> text`

returns: text

`_cdi(char[], char[], anyelement) -> text`

returns: text

`_cexists(char[], char[], char[]) -> boolean`

returns: boolean

`_cexists(char[], char[]) -> boolean`

returns: boolean

`_ckeys(char[], char[], character) -> name[]`

returns: name[]

_ckeys(char[], character) -> name[]

returns: name[]

_cleanup() -> boolean

returns: boolean

_cmp_types(oid, char[]) -> boolean

returns: boolean

_col_is_null(char[], char[], char[], text, boolean) -> text

returns: text

_col_is_null(char[], char[], text, boolean) -> text

returns: text

_constraint(char[], char[], character, name[], text, text) -> text

returns: text

_constraint(char[], character, name[], text, text) -> text

returns: text

_contract_on(text) -> "char"

returns: "char"

_currtest() -> integer

returns: integer

_db_privs() -> name[]

returns: name[]

_def_is(text, text, anyelement, text) -> text

returns: text

_definer(char[], char[], name[]) -> boolean

returns: boolean

`_definer(char[], char[]) -> boolean`

returns: boolean

`_definer(char[], name[]) -> boolean`

returns: boolean

`_definer(char[]) -> boolean`

returns: boolean

`_dexists(char[], char[]) -> boolean`

returns: boolean

`_dexists(char[]) -> boolean`

returns: boolean

`_do_ne(text, text, text, text) -> text`

returns: text

`_docomp(text, text, text, text) -> text`

returns: text

`_expand_context(character) -> text`

returns: text

`_expand_on(character) -> text`

returns: text

`_expand_vol(character) -> text`

returns: text

`_extras(character, char[], name[]) -> name[]`

returns: name[]

`_extras(character, name[]) -> name[]`

returns: name[]

`_finish(integer, integer, integer) -> SETOF text`

returns: text

`_fkexists(char[], char[], name[]) -> boolean`

returns: boolean

`_fkexists(char[], name[]) -> boolean`

returns: boolean

`_fprivs_are(text, char[], name[], text) -> text`

returns: text

`_func_compare(char[], char[], name[], anyelement, anyelement, text) -> text`

returns: text

`_func_compare(char[], char[], name[], boolean, text) -> text`

returns: text

`_func_compare(char[], char[], anyelement, anyelement, text) -> text`

returns: text

`_func_compare(char[], char[], boolean, text) -> text`

returns: text

`_get(text) -> integer`

returns: integer

`_get_ac_privs(char[], text) -> text[]`

returns: text[]

`_get_col_ns_type(char[], char[], char[]) -> text`

returns: text

`_get_col_privs(char[], text, char[]) -> text[]`

returns: text[]

`_get_col_type(char[], char[], char[]) -> text`

returns: text

`_get_col_type(char[], char[]) -> text`

returns: text

`_get_context(char[], char[]) -> "char"`

returns: "char"

`_get_db_owner(char[]) -> char[]`

returns: char[]

`_get_db_privs(char[], text) -> text[]`

returns: text[]

`_get_dtype(char[], text, boolean) -> text`

returns: text

`_get_dtype(char[]) -> text`

returns: text

`_get_fdw_privs(char[], text) -> text[]`

returns: text[]

`_get_func_owner(char[], char[], name[]) -> char[]`

returns: char[]

`_get_func_owner(char[], name[]) -> char[]`

returns: char[]

`_get_func_privs(text, text) -> text[]`

returns: text[]

`_get_index_owner(char[], char[], char[]) -> char[]`

returns: char[]

`_get_index_owner(char[], char[]) -> char[]`

returns: char[]

`_get_lang_privs(char[], text) -> text[]`

returns: text[]

`_get_language_owner(char[]) -> char[]`

returns: char[]

`_get_latest(text) -> int4[]`

returns: int4[]

`_get_latest(text, integer) -> integer`

returns: integer

`_get_note(text) -> text`

returns: text

`_get_note(integer) -> text`

returns: text

`_get_opclass_owner(char[], char[]) -> char[]`

returns: char[]

`_get_opclass_owner(char[]) -> char[]`

returns: char[]

`_get_rel_owner(character, char[]) -> char[]`

returns: char[]

`_get_rel_owner(char[], char[]) -> char[]`

returns: char[]

`_get_rel_owner(char[]) -> char[]`

returns: char[]

`_get_rel_owner(character, char[], char[]) -> char[]`

returns: char[]

`_get_schema_owner(char[]) -> char[]`

returns: char[]

`_get_schema_privs(char[], text) -> text[]`

returns: text[]

`_get_sequence_privs(char[], text) -> text[]`

returns: text[]

`_get_server_privs(char[], text) -> text[]`

returns: text[]

`_get_table_privs(char[], text) -> text[]`

returns: text[]

`_get_tablespace_owner(char[]) -> char[]`

returns: char[]

`_get_tablespace_privs(char[], text) -> text[]`

returns: text[]

`_get_type_owner(char[], char[]) -> char[]`

returns: char[]

`_get_type_owner(char[]) -> char[]`

returns: char[]

`_got_func(char[], char[], name[]) -> boolean`

returns: boolean

`_got_func(char[], char[]) -> boolean`

returns: boolean

`_got_func(char[], name[]) -> boolean`

returns: boolean

`_got_func(char[]) -> boolean`

returns: boolean

`_grolist(char[]) -> oid[]`

returns: oid[]

`_has_def(char[], char[], char[]) -> boolean`

returns: boolean

`_has_def(char[], char[]) -> boolean`

returns: boolean

`_has_group(char[]) -> boolean`

returns: boolean

`_has_role(char[]) -> boolean`

returns: boolean

`_has_type(char[], char[], bpchar[]) -> boolean`

returns: boolean

`_has_type(char[], bpchar[]) -> boolean`

returns: boolean

`_has_user(char[]) -> boolean`

returns: boolean

`_hasc(char[], char[], character) -> boolean`

returns: boolean

`_hasc(char[], character) -> boolean`

returns: boolean

`_have_index(char[], char[], char[]) -> boolean`

returns: boolean

`_have_index(char[], char[]) -> boolean`

returns: boolean

`_ident_array_to_string(name[], text) -> text`

returns: text

`_ikeys(char[], char[], char[]) -> text[]`

returns: text[]

`_ikeys(char[], char[]) -> text[]`

returns: text[]

`_is_instead(char[], char[], char[]) -> boolean`

returns: boolean

`_is_instead(char[], char[]) -> boolean`

returns: boolean

`_is_schema(char[]) -> boolean`

returns: boolean

`_is_super(char[]) -> boolean`

returns: boolean

`_is_trusted(char[]) -> boolean`

returns: boolean

`_is_verbose() -> boolean`

returns: boolean

`_keys(char[], char[], character) -> SETOF name[]`

returns: name[]

`_keys(char[], character) -> SETOF name[]`

returns: name[]

`_lang(char[], char[], name[]) -> char[]`

returns: char[]

`_lang(char[], char[]) -> char[]`

returns: char[]

`_lang(char[], name[]) -> char[]`

returns: char[]

`_lang(char[]) -> char[]`

returns: char[]

`_missing(character, char[], name[]) -> name[]`

returns: name[]

`_missing(character, name[]) -> name[]`

returns: name[]

`_nosuch(char[], char[], name[]) -> text`

returns: text

`_op_exists(char[], char[], char[], char[], char[]) -> boolean`

returns: boolean

`_op_exists(char[], char[], char[], char[]) -> boolean`

returns: boolean

`_op_exists(char[], char[], char[]) -> boolean`

returns: boolean

`_opc_exists(char[], char[]) -> boolean`

returns: boolean

`_opc_exists(char[]) -> boolean`

returns: boolean

`_pg_sv_column_array(oid, int2[]) -> name[]`

returns: name[]

`_pg_sv_table_accessible(oid, oid) -> boolean`

returns: boolean

`_pg_sv_type_array(oid[]) -> name[]`

returns: name[]

`_query(text) -> text`

returns: text

`_quote_ident_like(text, text) -> text`

returns: text

`_refine_vol(text) -> text`

returns: text

`_relcomp(text, text, text, text) -> text`

returns: text

`_relcomp(text, anyarray, text, text) -> text`

returns: text

`_relcomp(text, text, text, text, text) -> text`

returns: text

`_relexists(char[], char[]) -> boolean`

returns: boolean

`_relexists(char[]) -> boolean`

returns: boolean

`_relne(text, text, text, text) -> text`

returns: text

`_relne(text, anyarray, text, text) -> text`

returns: text

`_returns(char[], char[], name[]) -> text`

returns: text

`_returns(char[], char[]) -> text`

returns: text

`_returns(char[], name[]) -> text`

returns: text

`_returns(char[]) -> text`

returns: text

`_rexists(character, char[], char[]) -> boolean`

returns: boolean

`_rexists(character, char[]) -> boolean`

returns: boolean

`_rule_on(char[], char[], char[]) -> "char"`

returns: "char"

`_rule_on(char[], char[]) -> "char"`

returns: "char"

`_runem(text[], boolean) -> SETOF text`

returns: text

`_runner(text[], text[], text[], text[], text[]) -> SETOF text`

returns: text

_set(text, integer, text) -> integer

returns: integer

_set(text, integer) -> integer

returns: integer

_set(integer, integer) -> integer

returns: integer

_strict(char[], char[], name[]) -> boolean

returns: boolean

_strict(char[], char[]) -> boolean

returns: boolean

_strict(char[], name[]) -> boolean

returns: boolean

_strict(char[]) -> boolean

returns: boolean

_table_privs() -> name[]

returns: name[]

_temptable(text, text) -> text

returns: text

_temptable(anyarray, text) -> text

returns: text

_temptypes(text) -> text

returns: text

_time_trials(text, integer, numeric) -> SETOF _time_trial_type

returns: *public._time_trial_type*

`_tlike(boolean, text, text, text) -> text`

returns: text

`_todo() -> text`

returns: text

`_trig(char[], char[], char[]) -> boolean`

returns: boolean

`_trig(char[], char[]) -> boolean`

returns: boolean

`_types_are(char[], name[], text, bpchar[]) -> text`

returns: text

`_types_are(name[], text, bpchar[]) -> text`

returns: text

`_unlike(boolean, anyelement, text, text) -> text`

returns: text

`_vol(char[], char[], name[]) -> text`

returns: text

`_vol(char[], char[]) -> text`

returns: text

`_vol(char[], name[]) -> text`

returns: text

`_vol(char[]) -> text`

returns: text

`action(anyelement, sql text) -> anyelement`

returns: anyelement

action(anyelement, sql text[]) -> anyelement

returns: anyelement

add_array(anyarray, anyarray) -> anyarray

returns: anyarray

add_result(boolean, boolean, text, text, text) -> integer

returns: integer

alike(anyelement, text, text) -> text

returns: text

alike(anyelement, text) -> text

returns: text

any_column_privs_are(char[], char[], char[], name[], text) -> text

returns: text

any_column_privs_are(char[], char[], char[], name[]) -> text

returns: text

any_column_privs_are(char[], char[], name[], text) -> text

returns: text

any_column_privs_are(char[], char[], name[]) -> text

returns: text

array_sum(int4[]) -> bigint

returns: bigint

array_sum(int8[]) -> numeric

returns: numeric

array_sum(numeric[]) -> numeric

returns: numeric

array_sum(float8[]) -> double precision

returns: double precision

array_to_char(anyarray, format text) -> text[]

returns: text[]

bag_eq(text, text, text) -> text

returns: text

bag_eq(text, text) -> text

returns: text

bag_eq(text, anyarray, text) -> text

returns: text

bag_eq(text, anyarray) -> text

returns: text

bag_has(text, text, text) -> text

returns: text

bag_has(text, text) -> text

returns: text

bag_hasnt(text, text, text) -> text

returns: text

bag_hasnt(text, text) -> text

returns: text

bag_ne(text, text, text) -> text

returns: text

bag_ne(text, text) -> text

returns: text

bag_ne(text, anyarray, text) -> text

returns: text

bag_ne(text, anyarray) -> text

returns: text

can(char[], name[], text) -> text

returns: text

can(char[], name[]) -> text

returns: text

can(name[], text) -> text

returns: text

can(name[]) -> text

returns: text

cast_context_is(char[], char[], text, text) -> text

returns: text

cast_context_is(char[], char[], text) -> text

returns: text

casts_are(text[], text) -> text

returns: text

casts_are(text[]) -> text

returns: text

check_test(text, boolean, text, text, text, boolean) -> SETOF text

returns: text

check_test(text, boolean, text, text, text) -> SETOF text

returns: text

check_test(text, boolean, text, text) -> SETOF text

returns: text

check_test(text, boolean, text) -> SETOF text

returns: text

check_test(text, boolean) -> SETOF text

returns: text

cmp_ok(anyelement, text, anyelement, text) -> text

returns: text

cmp_ok(anyelement, text, anyelement) -> text

returns: text

col_default_is(char[], char[], char[], anyelement, text) -> text

returns: text

col_default_is(char[], char[], char[], text, text) -> text

returns: text

col_default_is(char[], char[], anyelement, text) -> text

returns: text

col_default_is(char[], char[], text, text) -> text

returns: text

col_default_is(char[], char[], anyelement) -> text

returns: text

col_default_is(char[], char[], text) -> text

returns: text

col_has_check(char[], char[], name[], text) -> text

returns: text

col_has_check(char[], name[], text) -> text

returns: text

col_has_check(char[], name[]) -> text

returns: text

col_has_check(char[], char[], char[], text) -> text

returns: text

col_has_check(char[], char[], text) -> text

returns: text

col_has_check(char[], char[]) -> text

returns: text

col_has_default(char[], char[], char[], text) -> text

returns: text

col_has_default(char[], char[], text) -> text

returns: text

col_has_default(char[], char[]) -> text

returns: text

col_hasnt_default(char[], char[], char[], text) -> text

returns: text

col_hasnt_default(char[], char[], text) -> text

returns: text

col_hasnt_default(char[], char[]) -> text

returns: text

col_is_fk(char[], char[], name[], text) -> text

returns: text

col_is_fk(char[], name[], text) -> text

returns: text

col_is_fk(char[], name[]) -> text

returns: text

col_is_fk(char[], char[], char[], text) -> text

returns: text

col_is_fk(char[], char[], text) -> text

returns: text

col_is_fk(char[], char[]) -> text

returns: text

col_is_null(char[], char[], char[], text) -> text

returns: text

col_is_null(char[], char[], char[]) -> text

returns: text

col_is_null(char[], char[]) -> text

returns: text

col_is_pk(char[], char[], name[], text) -> text

returns: text

col_is_pk(char[], name[], text) -> text

returns: text

col_is_pk(char[], name[]) -> text

returns: text

col_is_pk(char[], char[], char[], text) -> text

returns: text

col_is_pk(char[], char[], text) -> text

returns: text

col_is_pk(char[], char[]) -> text

returns: text

col_is_unique(char[], char[], name[], text) -> text

returns: text

col_is_unique(char[], char[], name[]) -> text

returns: text

col_is_unique(char[], char[], char[]) -> text

returns: text

col_is_unique(char[], name[], text) -> text

returns: text

col_is_unique(char[], name[]) -> text

returns: text

col_is_unique(char[], char[], char[], text) -> text

returns: text

col_is_unique(char[], char[], text) -> text

returns: text

col_is_unique(char[], char[]) -> text

returns: text

col_isnt_fk(char[], char[], name[], text) -> text

returns: text

col_isnt_fk(char[], name[], text) -> text

returns: text

col_isnt_fk(char[], name[]) -> text

returns: text

col_isnt_fk(char[], char[], char[], text) -> text

returns: text

col_isnt_fk(char[], char[], text) -> text

returns: text

col_isnt_fk(char[], char[]) -> text

returns: text

col_isnt_pk(char[], char[], name[], text) -> text

returns: text

col_isnt_pk(char[], name[], text) -> text

returns: text

col_isnt_pk(char[], name[]) -> text

returns: text

col_isnt_pk(char[], char[], char[], text) -> text

returns: text

col_isnt_pk(char[], char[], text) -> text

returns: text

col_isnt_pk(char[], char[]) -> text

returns: text

col_not_null(char[], char[], char[], text) -> text

returns: text

col_not_null(char[], char[], text) -> text

returns: text

col_not_null(char[], char[]) -> text

returns: text

col_type_is(char[], char[], char[], char[], text, text) -> text

returns: text

col_type_is(char[], char[], char[], char[], text) -> text

returns: text

col_type_is(char[], char[], char[], text, text) -> text

returns: text

col_type_is(char[], char[], char[], text) -> text

returns: text

col_type_is(char[], char[], text, text) -> text

returns: text

col_type_is(char[], char[], text) -> text

returns: text

collect_tap(text[]) -> RECORD()

returns: RECORD()

collect_tap(varchar[]) -> text

returns: text

column_names(namespace char[], table char[]) -> SETOF char[]

returns: char[]

column_privs_are(char[], char[], char[], char[], name[], text) -> text

returns: text

column_privs_are(char[], char[], char[], char[], name[]) -> text

returns: text

column_privs_are(char[], char[], char[], name[], text) -> text

returns: text

column_privs_are(char[], char[], char[], name[]) -> text

returns: text

columns_are(char[], char[], name[], text) -> text

returns: text

columns_are(char[], char[], name[]) -> text

returns: text

columns_are(char[], name[], text) -> text

returns: text

columns_are(char[], name[]) -> text

returns: text

composite_owner_is(char[], char[], char[], text) -> text

returns: text

composite_owner_is(char[], char[], char[]) -> text

returns: text

composite_owner_is(char[], char[], text) -> text

returns: text

composite_owner_is(char[], char[]) -> text

returns: text

database_privs_are(char[], char[], name[], text) -> text

returns: text

database_privs_are(char[], char[], name[]) -> text

returns: text

db_owner_is(char[], char[], text) -> text

returns: text

db_owner_is(char[], char[]) -> text

returns: text

diag(msg text) -> text

returns: text

diag(msg anyelement) -> text

returns: text

diag(text[]) -> RECORD()

returns: RECORD()

diag(anyarray) -> RECORD()

returns: RECORD()

diag_test_name(text) -> text

returns: text

display_oper(char[], oid) -> text

returns: text

divide_array(anyarray, anyelement) -> anyarray

returns: anyarray

divide_array(anyarray, anyarray) -> anyarray

returns: anyarray

do_tap(char[], text) -> SETOF text

returns: text

do_tap(char[]) -> SETOF text

returns: text

do_tap(text) -> SETOF text

returns: text

do_tap() -> SETOF text

returns: text

doesn't_imatch(anyelement, text, text) -> text

returns: text

doesn't_imatch(anyelement, text) -> text

returns: text

doesn't_match(anyelement, text, text) -> text

returns: text

doesn't_match(anyelement, text) -> text

returns: text

domain_type_is(char[], text, char[], text, text) -> text

returns: text

domain_type_is(char[], text, char[], text) -> text

returns: text

domain_type_is(char[], text, text, text) -> text

returns: text

domain_type_is(char[], text, text) -> text

returns: text

domain_type_is(text, text, text) -> text

returns: text

domain_type_is(text, text) -> text

returns: text

domain_type_isnt(char[], text, char[], text, text) -> text

returns: text

domain_type_isnt(char[], text, char[], text) -> text

returns: text

domain_type_isnt(char[], text, text, text) -> text

returns: text

domain_type_isnt(char[], text, text) -> text

returns: text

domain_type_isnt(text, text, text) -> text

returns: text

domain_type_isnt(text, text) -> text

returns: text

domains_are(char[], name[], text) -> text

returns: text

domains_are(char[], name[]) -> text

returns: text

domains_are(name[], text) -> text

returns: text

domains_are(name[]) -> text

returns: text

drop_changes_view(attribute_directory.attributestore) -> attribute_directory.attributestore

returns: *attribute_directory.attributestore*

enum_has_labels(char[], char[], name[]) -> text

returns: text

enum_has_labels(char[], char[], name[], text) -> text

returns: text

enum_has_labels(char[], name[], text) -> text

returns: text

enum_has_labels(char[], name[]) -> text

returns: text

enums_are(char[], name[], text) -> text

returns: text

enums_are(char[], name[]) -> text

returns: text

enums_are(name[], text) -> text

returns: text

enums_are(name[]) -> text

returns: text

fail(text) -> text

returns: text

fail() -> text

returns: text

fdw_privs_are(char[], char[], name[], text) -> text

returns: text

fdw_privs_are(char[], char[], name[]) -> text

returns: text

findfuncs(char[], text, text) -> text[]

returns: text[]

findfuncs(char[], text) -> text[]

returns: text[]

findfuncs(text, text) -> text[]

returns: text[]

findfuncs(text) -> text[]

returns: text[]

finish() -> SETOF text

returns: text

first(anyelement) -> anyelement

returns: anyelement

fk_ok(char[], char[], name[], char[], char[], name[], text) -> text

returns: text

fk_ok(char[], name[], char[], name[], text) -> text

returns: text

fk_ok(char[], char[], name[], char[], char[], name[]) -> text

returns: text

fk_ok(char[], name[], char[], name[]) -> text

returns: text

fk_ok(char[], char[], char[], char[], char[], char[], text) -> text

returns: text

fk_ok(char[], char[], char[], char[], char[], text) -> text

returns: text

fk_ok(char[], char[], char[], char[], text) -> text

returns: text

fk_ok(char[], char[], char[], char[]) -> text

returns: text

foreign_table_owner_is(char[], char[]) -> text

returns: text

foreign_table_owner_is(char[], char[], char[], text) -> text

returns: text

foreign_table_owner_is(char[], char[], char[]) -> text

returns: text

foreign_table_owner_is(char[], char[], text) -> text

returns: text

foreign_tables_are(char[], name[], text) -> text

returns: text

foreign_tables_are(name[], text) -> text

returns: text

foreign_tables_are(char[], name[]) -> text

returns: text

foreign_tables_are(name[]) -> text

returns: text

fst(anyelement, anyelement) -> anyelement

returns: anyelement

function_lang_is(char[], char[], name[], char[], text) -> text

returns: text

function_lang_is(char[], char[], name[], char[]) -> text

returns: text

function_lang_is(char[], char[], char[], text) -> text

returns: text

function_lang_is(char[], char[], char[]) -> text

returns: text

function_lang_is(char[], name[], char[], text) -> text

returns: text

function_lang_is(char[], name[], char[]) -> text

returns: text

function_lang_is(char[], char[], text) -> text

returns: text

function_lang_is(char[], char[]) -> text

returns: text

function_owner_is(char[], char[], name[], char[], text) -> text

returns: text

function_owner_is(char[], char[], name[], char[]) -> text

returns: text

function_owner_is(char[], name[], char[], text) -> text

returns: text

function_owner_is(char[], name[], char[]) -> text

returns: text

function_privs_are(char[], char[], name[], char[], name[], text) -> text

returns: text

function_privs_are(char[], char[], name[], char[], name[]) -> text

returns: text

function_privs_are(char[], name[], char[], name[], text) -> text

returns: text

function_privs_are(char[], name[], char[], name[]) -> text

returns: text

function_returns(char[], char[], name[], text, text) -> text

returns: text

function_returns(char[], char[], name[], text) -> text

returns: text

function_returns(char[], char[], text, text) -> text

returns: text

function_returns(char[], char[], text) -> text

returns: text

function_returns(char[], name[], text, text) -> text

returns: text

function_returns(char[], name[], text) -> text

returns: text

function_returns(char[], text, text) -> text

returns: text

function_returns(char[], text) -> text

returns: text

functions_are(char[], name[], text) -> text

returns: text

functions_are(char[], name[]) -> text

returns: text

functions_are(name[], text) -> text

returns: text

functions_are(name[]) -> text

returns: text

groups_are(name[], text) -> text

returns: text

groups_are(name[]) -> text

returns: text

has_cast(char[], char[], char[], char[], text) -> text

returns: text

has_cast(char[], char[], char[], char[]) -> text

returns: text

has_cast(char[], char[], char[], text) -> text

returns: text

has_cast(char[], char[], char[]) -> text

returns: text

has_cast(char[], char[], text) -> text

returns: text

has_cast(char[], char[]) -> text

returns: text

has_check(char[], char[], text) -> text

returns: text

has_check(char[], text) -> text

returns: text

has_check(char[]) -> text

returns: text

has_column(char[], char[], char[], text) -> text

returns: text

has_column(char[], char[], text) -> text

returns: text

has_column(char[], char[]) -> text

returns: text

has_composite(char[], char[], text) -> text

returns: text

has_composite(char[], text) -> text

returns: text

has_composite(char[]) -> text

returns: text

has_domain(char[], char[], text) -> text

returns: text

has_domain(char[], char[]) -> text

returns: text

has_domain(char[], text) -> text

returns: text

has_domain(char[]) -> text

returns: text

has_enum(char[], char[], text) -> text

returns: text

has_enum(char[], char[]) -> text

returns: text

has_enum(char[], text) -> text

returns: text

has_enum(char[]) -> text

returns: text

has_fk(char[], char[], text) -> text

returns: text

has_fk(char[], text) -> text

returns: text

has_fk(char[]) -> text

returns: text

has_foreign_table(char[], char[], text) -> text

returns: text

has_foreign_table(char[], char[]) -> text

returns: text

has_foreign_table(char[], text) -> text

returns: text

has_foreign_table(char[]) -> text

returns: text

has_function(char[], char[], name[], text) -> text

returns: text

has_function(char[], char[], name[]) -> text

returns: text

has_function(char[], char[], text) -> text

returns: text

has_function(char[], char[]) -> text

returns: text

has_function(char[], name[], text) -> text

returns: text

has_function(char[], name[]) -> text

returns: text

has_function(char[], text) -> text

returns: text

has_function(char[]) -> text

returns: text

has_group(char[], text) -> text

returns: text

has_group(char[]) -> text

returns: text

has_index(char[], char[], char[], name[], text) -> text

returns: text

has_index(char[], char[], char[], name[]) -> text

returns: text

has_index(char[], char[], char[], char[], text) -> text

returns: text

has_index(char[], char[], char[], char[]) -> text

returns: text

has_index(char[], char[], name[], text) -> text

returns: text

has_index(char[], char[], name[]) -> text

returns: text

has_index(char[], char[], char[], text) -> text

returns: text

has_index(char[], char[], char[]) -> text

returns: text

has_index(char[], char[], text) -> text

returns: text

has_index(char[], char[]) -> text

returns: text

has_language(char[], text) -> text

returns: text

has_language(char[]) -> text

returns: text

has_leftop(char[], char[], char[], char[], text) -> text

returns: text

has_leftop(char[], char[], char[], char[]) -> text

returns: text

has_leftop(char[], char[], char[], text) -> text

returns: text

has_leftop(char[], char[], char[]) -> text

returns: text

has_leftop(char[], char[], text) -> text

returns: text

has_leftop(char[], char[]) -> text

returns: text

has_materialized_view(char[], char[], text) -> text

returns: text

has_materialized_view(char[], text) -> text

returns: text

has_materialized_view(char[]) -> text

returns: text

has_opclass(char[], char[], text) -> text

returns: text

has_opclass(char[], char[]) -> text

returns: text

has_opclass(char[], text) -> text

returns: text

has_opclass(char[]) -> text

returns: text

has_operator(char[], char[], char[], char[], char[], text) -> text

returns: text

has_operator(char[], char[], char[], char[], char[]) -> text

returns: text

has_operator(char[], char[], char[], char[], text) -> text

returns: text

has_operator(char[], char[], char[], char[]) -> text

returns: text

has_operator(char[], char[], char[], text) -> text

returns: text

has_operator(char[], char[], char[]) -> text

returns: text

has_pk(char[], char[], text) -> text

returns: text

has_pk(char[], text) -> text

returns: text

has_pk(char[]) -> text

returns: text

has_relation(char[], char[], text) -> text

returns: text

has_relation(char[], text) -> text

returns: text

has_relation(char[]) -> text

returns: text

has_rightop(char[], char[], char[], char[], text) -> text

returns: text

has_rightop(char[], char[], char[], char[]) -> text

returns: text

has_rightop(char[], char[], char[], text) -> text

returns: text

has_rightop(char[], char[], char[]) -> text

returns: text

has_rightop(char[], char[], text) -> text

returns: text

has_rightop(char[], char[]) -> text

returns: text

has_role(char[], text) -> text

returns: text

has_role(char[]) -> text

returns: text

has_rule(char[], char[], char[], text) -> text

returns: text

has_rule(char[], char[], char[]) -> text

returns: text

has_rule(char[], char[], text) -> text

returns: text

has_rule(char[], char[]) -> text

returns: text

has_schema(char[], text) -> text

returns: text

has_schema(char[]) -> text

returns: text

has_sequence(char[], char[], text) -> text

returns: text

has_sequence(char[], char[]) -> text

returns: text

has_sequence(char[], text) -> text

returns: text

has_sequence(char[]) -> text

returns: text

has_table(char[], char[], text) -> text

returns: text

has_table(char[], char[]) -> text

returns: text

has_table(char[], text) -> text

returns: text

has_table(char[]) -> text

returns: text

has_tablespace(char[], text, text) -> text

returns: text

has_tablespace(char[], text) -> text

returns: text

has_tablespace(char[]) -> text

returns: text

has_trigger(char[], char[], char[], text) -> text

returns: text

has_trigger(char[], char[], char[]) -> text

returns: text

has_trigger(char[], char[], text) -> text

returns: text

has_trigger(char[], char[]) -> text

returns: text

has_type(char[], char[], text) -> text

returns: text

has_type(char[], char[]) -> text

returns: text

has_type(char[], text) -> text

returns: text

has_type(char[]) -> text

returns: text

has_unique(text, text, text) -> text

returns: text

has_unique(text, text) -> text

returns: text

has_unique(text) -> text

returns: text

has_user(char[], text) -> text

returns: text

has_user(char[]) -> text

returns: text

has_view(char[], char[], text) -> text

returns: text

has_view(char[], text) -> text

returns: text

has_view(char[]) -> text

returns: text

hasnt_cast(char[], char[], char[], char[], text) -> text

returns: text

hasnt_cast(char[], char[], char[], char[]) -> text

returns: text

hasnt_cast(char[], char[], char[], text) -> text

returns: text

hasnt_cast(char[], char[], char[]) -> text

returns: text

hasnt_cast(char[], char[], text) -> text

returns: text

hasnt_cast(char[], char[]) -> text

returns: text

hasnt_column(char[], char[], char[], text) -> text

returns: text

hasnt_column(char[], char[], text) -> text

returns: text

hasnt_column(char[], char[]) -> text

returns: text

hasnt_composite(char[], char[], text) -> text

returns: text

hasnt_composite(char[], text) -> text

returns: text

hasnt_composite(char[]) -> text

returns: text

hasnt_domain(char[], char[], text) -> text

returns: text

hasnt_domain(char[], char[]) -> text

returns: text

hasnt_domain(char[], text) -> text

returns: text

hasnt_domain(char[]) -> text

returns: text

hasnt_enum(char[], char[], text) -> text

returns: text

hasnt_enum(char[], char[]) -> text

returns: text

hasnt_enum(char[], text) -> text

returns: text

hasnt_enum(char[]) -> text

returns: text

hasnt_fk(char[], char[], text) -> text

returns: text

hasnt_fk(char[], text) -> text

returns: text

hasnt_fk(char[]) -> text

returns: text

hasnt_foreign_table(char[], char[], text) -> text

returns: text

hasnt_foreign_table(char[], char[]) -> text

returns: text

hasnt_foreign_table(char[], text) -> text

returns: text

hasnt_foreign_table(char[]) -> text

returns: text

hasnt_function(char[], char[], name[], text) -> text

returns: text

hasnt_function(char[], char[], name[]) -> text

returns: text

hasnt_function(char[], char[], text) -> text

returns: text

hasnt_function(char[], char[]) -> text

returns: text

hasnt_function(char[], name[], text) -> text

returns: text

hasnt_function(char[], name[]) -> text

returns: text

hasnt_function(char[], text) -> text

returns: text

hasnt_function(char[]) -> text

returns: text

hasnt_group(char[], text) -> text

returns: text

hasnt_group(char[]) -> text

returns: text

hasnt_index(char[], char[], char[], text) -> text

returns: text

hasnt_index(char[], char[], char[]) -> text

returns: text

hasnt_index(char[], char[], text) -> text

returns: text

hasnt_index(char[], char[]) -> text

returns: text

hasnt_language(char[]) -> text

returns: text

hasnt_language(char[], text) -> text

returns: text

hasnt_materialized_view(char[], char[], text) -> text

returns: text

hasnt_materialized_view(char[], text) -> text

returns: text

hasnt_materialized_view(char[]) -> text

returns: text

hasnt_opclass(char[], char[], text) -> text

returns: text

hasnt_opclass(char[], char[]) -> text

returns: text

hasnt_opclass(char[], text) -> text

returns: text

hasnt_opclass(char[]) -> text

returns: text

hasnt_pk(char[]) -> text

returns: text

hasnt_pk(char[], char[], text) -> text

returns: text

hasnt_pk(char[], text) -> text

returns: text

hasnt_relation(char[], char[], text) -> text

returns: text

hasnt_relation(char[], text) -> text

returns: text

hasnt_relation(char[]) -> text

returns: text

hasnt_role(char[], text) -> text

returns: text

hasnt_role(char[]) -> text

returns: text

hasnt_rule(char[], char[], char[], text) -> text

returns: text

hasnt_rule(char[], char[], char[]) -> text

returns: text

hasnt_rule(char[], char[], text) -> text

returns: text

hasnt_rule(char[], char[]) -> text

returns: text

hasnt_schema(char[], text) -> text

returns: text

hasnt_schema(char[]) -> text

returns: text

hasnt_sequence(char[], char[], text) -> text

returns: text

hasnt_sequence(char[], text) -> text

returns: text

hasnt_sequence(char[]) -> text

returns: text

hasnt_table(char[], char[], text) -> text

returns: text

hasnt_table(char[], char[]) -> text

returns: text

hasnt_table(char[], text) -> text

returns: text

hasnt_table(char[]) -> text

returns: text

hasnt_tablespace(char[]) -> text

returns: text

hasnt_tablespace(char[], text) -> text

returns: text

hasnt_trigger(char[], char[], char[], text) -> text

returns: text

hasnt_trigger(char[], char[], char[]) -> text

returns: text

hasnt_trigger(char[], char[], text) -> text

returns: text

hasnt_trigger(char[], char[]) -> text

returns: text

hasnt_type(char[], char[], text) -> text

returns: text

hasnt_type(char[], char[]) -> text

returns: text

hasnt_type(char[], text) -> text

returns: text

hasnt_type(char[]) -> text

returns: text

hasnt_user(char[], text) -> text

returns: text

hasnt_user(char[]) -> text

returns: text

hasnt_view(char[], char[], text) -> text

returns: text

hasnt_view(char[], text) -> text

returns: text

hasnt_view(char[]) -> text

returns: text

ialike(anyelement, text, text) -> text

returns: text

ialike(anyelement, text) -> text

returns: text

imatches(anyelement, text, text) -> text

returns: text

imatches(anyelement, text) -> text

returns: text

in_todo() -> boolean

returns: boolean

index_is_primary(char[], char[], char[], text) -> text

returns: text

index_is_primary(char[], char[], char[]) -> text

returns: text

index_is_primary(char[], char[]) -> text

returns: text

index_is_primary(char[]) -> text

returns: text

index_is_type(char[], char[], char[], char[], text) -> text

returns: text

index_is_type(char[], char[], char[], char[]) -> text

returns: text

index_is_type(char[], char[], char[]) -> text

returns: text

index_is_type(char[], char[]) -> text

returns: text

index_is_unique(char[], char[], char[], text) -> text

returns: text

index_is_unique(char[], char[], char[]) -> text

returns: text

index_is_unique(char[], char[]) -> text

returns: text

index_is_unique(char[]) -> text

returns: text

index_owner_is(char[], char[], char[], char[], text) -> text

returns: text

index_owner_is(char[], char[], char[], char[]) -> text

returns: text

index_owner_is(char[], char[], char[], text) -> text

returns: text

index_owner_is(char[], char[], char[]) -> text

returns: text

indexes_are(char[], char[], name[], text) -> text

returns: text

indexes_are(char[], char[], name[]) -> text

returns: text

indexes_are(char[], name[], text) -> text

returns: text

indexes_are(char[], name[]) -> text

returns: text

integer_to_array(value integer) -> int4[]

returns: int4[]

is(anyelement, anyelement, text) -> text

returns: text

is(anyelement, anyelement) -> text

returns: text

is_aggregate(char[], char[], name[], text) -> text

returns: text

is_aggregate(char[], char[], name[]) -> text

returns: text

is_aggregate(char[], char[], text) -> text

returns: text

is_aggregate(char[], char[]) -> text

returns: text

is_aggregate(char[], name[], text) -> text

returns: text

is_aggregate(char[], name[]) -> text

returns: text

is_aggregate(char[], text) -> text

returns: text

is_aggregate(char[]) -> text

returns: text

is_clustered(char[], char[], char[], text) -> text

returns: text

is_clustered(char[], char[], char[]) -> text

returns: text

is_clustered(char[], char[]) -> text

returns: text

is_clustered(char[]) -> text

returns: text

is_definer(char[], char[], name[], text) -> text

returns: text

is_definer(char[], char[], name[]) -> text

returns: text

is_definer(char[], char[], text) -> text

returns: text

is_definer(char[], char[]) -> text

returns: text

is_definer(char[], name[], text) -> text

returns: text

is_definer(char[], name[]) -> text

returns: text

is_definer(char[], text) -> text

returns: text

is_definer(char[]) -> text

returns: text

is_empty(text, text) -> text

returns: text

is_empty(text) -> text

returns: text

is_member_of(char[], char[]) -> text

returns: text

is_member_of(char[], name[], text) -> text

returns: text

is_member_of(char[], char[], text) -> text

returns: text

is_member_of(char[], name[]) -> text

returns: text

is_strict(char[], char[], name[], text) -> text

returns: text

is_strict(char[], char[], name[]) -> text

returns: text

is_strict(char[], char[], text) -> text

returns: text

is_strict(char[], char[]) -> text

returns: text

is_strict(char[], name[], text) -> text

returns: text

is_strict(char[], name[]) -> text

returns: text

is_strict(char[], text) -> text

returns: text

is_strict(char[]) -> text

returns: text

is_superuser(char[], text) -> text

returns: text

is_superuser(char[]) -> text

returns: text

isa_ok(anyelement, regtype, text) -> text

returns: text

isa_ok(anyelement, regtype) -> text

returns: text

isnt(anyelement, anyelement, text) -> text

returns: text

isnt(anyelement, anyelement) -> text

returns: text

isnt_empty(text, text) -> text

returns: text

isnt_empty(text) -> text

returns: text

isnt_strict(char[], char[], name[], text) -> text

returns: text

isnt_strict(char[], char[], name[]) -> text

returns: text

isnt_strict(char[], char[], text) -> text

returns: text

isnt_strict(char[], char[]) -> text

returns: text

isnt_strict(char[], name[], text) -> text

returns: text

isnt_strict(char[], name[]) -> text

returns: text

isnt_strict(char[], text) -> text

returns: text

isnt_strict(char[]) -> text

returns: text

isnt_superuser(char[], text) -> text

returns: text

isnt_superuser(char[]) -> text

returns: text

language_is_trusted(char[], text) -> text

returns: text

language_is_trusted(char[]) -> text

returns: text

language_owner_is(char[], char[], text) -> text

returns: text

language_owner_is(char[], char[]) -> text

returns: text

language_privs_are(char[], char[], name[], text) -> text

returns: text

language_privs_are(char[], char[], name[]) -> text

returns: text

languages_are(name[], text) -> text

returns: text

languages_are(name[]) -> text

returns: text

last(anyelement) -> anyelement

returns: anyelement

lives_ok(text, text) -> text

returns: text

lives_ok(text) -> text

returns: text

matches(anyelement, text, text) -> text

returns: text

matches(anyelement, text) -> text

returns: text

materialized_view_owner_is(char[], char[], char[], text) -> text

returns: text

materialized_view_owner_is(char[], char[], char[]) -> text

returns: text

materialized_view_owner_is(char[], char[], text) -> text

returns: text

materialized_view_owner_is(char[], char[]) -> text

returns: text

materialized_views_are(char[], name[], text) -> text

returns: text

materialized_views_are(name[], text) -> text

returns: text

materialized_views_are(char[], name[]) -> text

returns: text

materialized_views_are(name[]) -> text

returns: text

multiply_array(anyarray, anyelement) -> anyarray

returns: anyarray

multiply_array(anyarray, anyarray) -> anyarray

returns: anyarray

no_plan() -> SETOF boolean

returns: boolean

num_failed() -> integer

returns: integer

ok(boolean, text) -> text

returns: text

ok(boolean) -> text

returns: text

opclass_owner_is(char[], char[], char[], text) -> text

returns: text

opclass_owner_is(char[], char[], char[]) -> text

returns: text

opclass_owner_is(char[], char[], text) -> text

returns: text

opclass_owner_is(char[], char[]) -> text

returns: text

opclasses_are(char[], name[], text) -> text

returns: text

opclasses_are(char[], name[]) -> text

returns: text

opclasses_are(name[], text) -> text

returns: text

opclasses_are(name[]) -> text

returns: text

operators_are(char[], text[], text) -> text

returns: text

operators_are(char[], text[]) -> text

returns: text

operators_are(text[], text) -> text

returns: text

operators_are(text[]) -> text

returns: text

os_name() -> text

returns: text

pass(text) -> text

returns: text

pass() -> text

returns: text

performs_ok(text, numeric, text) -> text

returns: text

performs_ok(text, numeric) -> text

returns: text

performs_within(text, numeric, numeric, integer, text) -> text

returns: text

performs_within(text, numeric, numeric, integer) -> text

returns: text

performs_within(text, numeric, numeric, text) -> text

returns: text

performs_within(text, numeric, numeric) -> text

returns: text

pg_version() -> text

returns: text

pg_version_num() -> integer

returns: integer

pgtap_version() -> numeric

returns: numeric

plan(integer) -> text

returns: text

proreftype(oid) -> oid

returns: oid

relation_owner_is(char[], char[], char[], text) -> text

returns: text

relation_owner_is(char[], char[], char[]) -> text

returns: text

relation_owner_is(char[], char[], text) -> text

returns: text

relation_owner_is(char[], char[]) -> text

returns: text

results_eq(refcursor, refcursor, text) -> text

returns: text

results_eq(refcursor, refcursor) -> text

returns: text

results_eq(text, text, text) -> text

returns: text

results_eq(text, text) -> text

returns: text

results_eq(text, anyarray, text) -> text

returns: text

results_eq(text, anyarray) -> text

returns: text

results_eq(text, refcursor, text) -> text

returns: text

results_eq(text, refcursor) -> text

returns: text

results_eq(refcursor, text, text) -> text

returns: text

results_eq(refcursor, text) -> text

returns: text

results_eq(refcursor, anyarray, text) -> text

returns: text

results_eq(refcursor, anyarray) -> text

returns: text

results_ne(refcursor, refcursor, text) -> text

returns: text

results_ne(refcursor, refcursor) -> text

returns: text

results_ne(text, text, text) -> text

returns: text

results_ne(text, text) -> text

returns: text

results_ne(text, anyarray, text) -> text

returns: text

results_ne(text, anyarray) -> text

returns: text

results_ne(text, refcursor, text) -> text

returns: text

results_ne(text, refcursor) -> text

returns: text

results_ne(refcursor, text, text) -> text

returns: text

results_ne(refcursor, text) -> text

returns: text

results_ne(refcursor, anyarray, text) -> text

returns: text

results_ne(refcursor, anyarray) -> text

returns: text

roles_are(name[], text) -> text

returns: text

roles_are(name[]) -> text

returns: text

row_eq(text, anyelement, text) -> text

returns: text

row_eq(text, anyelement) -> text

returns: text

rule_is_instead(char[], char[], char[], text) -> text

returns: text

rule_is_instead(char[], char[], char[]) -> text

returns: text

rule_is_instead(char[], char[], text) -> text

returns: text

rule_is_instead(char[], char[]) -> text

returns: text

rule_is_on(char[], char[], char[], text, text) -> text

returns: text

rule_is_on(char[], char[], char[], text) -> text

returns: text

rule_is_on(char[], char[], text, text) -> text

returns: text

rule_is_on(char[], char[], text) -> text

returns: text

rules_are(char[], char[], name[], text) -> text

returns: text

rules_are(char[], char[], name[]) -> text

returns: text

rules_are(char[], name[], text) -> text

returns: text

rules_are(char[], name[]) -> text

returns: text

runtests(char[], text) -> SETOF text

returns: text

runtests(char[]) -> SETOF text

returns: text

runtests(text) -> SETOF text

returns: text

runtests() -> SETOF text

returns: text

safe_division(numerator anyelement, denominator anyelement) -> anyelement

returns: anyelement

schema_owner_is(char[], char[], text) -> text

returns: text

schema_owner_is(char[], char[]) -> text

returns: text

schema_privs_are(char[], char[], name[], text) -> text

returns: text

schema_privs_are(char[], char[], name[]) -> text

returns: text

schemas_are(name[], text) -> text

returns: text

schemas_are(name[]) -> text

returns: text

sequence_owner_is(char[], char[], char[], text) -> text

returns: text

sequence_owner_is(char[], char[], char[]) -> text

returns: text

sequence_owner_is(char[], char[], text) -> text

returns: text

sequence_owner_is(char[], char[]) -> text

returns: text

sequence_privs_are(char[], char[], char[], name[], text) -> text

returns: text

sequence_privs_are(char[], char[], char[], name[]) -> text

returns: text

sequence_privs_are(char[], char[], name[], text) -> text

returns: text

sequence_privs_are(char[], char[], name[]) -> text

returns: text

sequences_are(char[], name[], text) -> text

returns: text

sequences_are(name[], text) -> text

returns: text

sequences_are(char[], name[]) -> text

returns: text

sequences_are(name[]) -> text

returns: text

server_privs_are(char[], char[], name[], text) -> text

returns: text

server_privs_are(char[], char[], name[]) -> text

returns: text

set_eq(text, text, text) -> text

returns: text

set_eq(text, text) -> text

returns: text

set_eq(text, anyarray, text) -> text

returns: text

set_eq(text, anyarray) -> text

returns: text

set_has(text, text, text) -> text

returns: text

set_has(text, text) -> text

returns: text

set_hasnt(text, text, text) -> text

returns: text

set_hasnt(text, text) -> text

returns: text

set_ne(text, text, text) -> text

returns: text

set_ne(text, text) -> text

returns: text

set_ne(text, anyarray, text) -> text

returns: text

set_ne(text, anyarray) -> text

returns: text

skip(why text, how_many integer) -> text

returns: text

skip(text) -> text

returns: text

skip(integer, text) -> text

returns: text

skip(integer) -> text

returns: text

smallint_to_array(value smallint) -> int2[]

returns: int2[]

smallint_to_timestamp_with_time_zone(smallint) -> timestamp with time zone

returns: timestamp with time zone

smallint_to_timestamp_without_time_zone(smallint) -> timestamp without time zone

returns: timestamp without time zone

snd(anelement, anelement) -> anelement

returns: anelement

sum_array(anyarray) -> anyarray

returns: anyarray

table_owner_is(char[], char[], char[], text) -> text

returns: text

table_owner_is(char[], char[], char[]) -> text

returns: text

table_owner_is(char[], char[], text) -> text

returns: text

table_owner_is(char[], char[]) -> text

returns: text

table_privs_are(char[], char[], char[], name[], text) -> text

returns: text

table_privs_are(char[], char[], char[], name[]) -> text

returns: text

table_privs_are(char[], char[], name[], text) -> text

returns: text

table_privs_are(char[], char[], name[]) -> text

returns: text

tables_are(char[], name[], text) -> text

returns: text

tables_are(name[], text) -> text

returns: text

tables_are(char[], name[]) -> text

returns: text

tables_are(name[]) -> text

returns: text

tablespace_owner_is(char[], char[], text) -> text

returns: text

tablespace_owner_is(char[], char[]) -> text

returns: text

tablespace_privs_are(char[], char[], name[], text) -> text

returns: text

tablespace_privs_are(char[], char[], name[]) -> text

returns: text

tablespaces_are(name[]) -> text

returns: text

tablespaces_are(name[], text) -> text

returns: text

throws_ilike(text, text, text) -> text

returns: text

throws_ilike(text, text) -> text

returns: text

throws_imatching(text, text, text) -> text

returns: text

throws_imatching(text, text) -> text

returns: text

throws_like(text, text, text) -> text

returns: text

throws_like(text, text) -> text

returns: text

throws_matching(text, text, text) -> text

returns: text

throws_matching(text, text) -> text

returns: text

throws_ok(text, character, text, text) -> text

returns: text

throws_ok(text, text, text) -> text

returns: text

throws_ok(text, text) -> text

returns: text

throws_ok(text) -> text

returns: text

throws_ok(text, integer, text, text) -> text

returns: text

throws_ok(text, integer, text) -> text

returns: text

throws_ok(text, integer) -> text

returns: text

to_pdf(text) -> int4[]

returns: int4[]

todo(why text, how_many integer) -> SETOF boolean

returns: boolean

todo(how_many integer, why text) -> SETOF boolean

returns: boolean

todo(why text) -> SETOF boolean

returns: boolean

todo(how_many integer) -> SETOF boolean

returns: boolean

todo_end() -> SETOF boolean

returns: boolean

todo_start(text) -> SETOF boolean

returns: boolean

todo_start() -> SETOF boolean

returns: boolean

trigger_is(char[], char[], char[], char[], char[], text) -> text

returns: text

trigger_is(char[], char[], char[], char[], char[]) -> text

returns: text

trigger_is(char[], char[], char[], text) -> text

returns: text

trigger_is(char[], char[], char[]) -> text

returns: text

triggers_are(char[], char[], name[], text) -> text

returns: text

triggers_are(char[], char[], name[]) -> text

returns: text

triggers_are(char[], name[], text) -> text

returns: text

triggers_are(char[], name[]) -> text

returns: text

type_columns(oid) -> SETOF column_info

returns: *public.column_info*

type_columns(namespace char[], type char[]) -> SETOF column_info

returns: *public.column_info*

type_owner_is(char[], char[], char[], text) -> text

returns: text

type_owner_is(char[], char[], char[]) -> text

returns: text

type_owner_is(char[], char[], text) -> text

returns: text

type_owner_is(char[], char[]) -> text

returns: text

types_are(char[], name[], text) -> text

returns: text

types_are(char[], name[]) -> text

returns: text

types_are(name[], text) -> text

returns: text

types_are(name[]) -> text

returns: text

unlike(anyelement, text, text) -> text

returns: text

unlike(anyelement, text) -> text

returns: text

unialike(anyelement, text, text) -> text

returns: text

unialike(anyelement, text) -> text

returns: text

users_are(name[], text) -> text

returns: text

users_are(name[]) -> text

returns: text

view_exists(char[], char[]) -> boolean

returns: boolean

view_owner_is(char[], char[], char[], text) -> text

returns: text

view_owner_is(char[], char[], char[]) -> text

returns: text

view_owner_is(char[], char[], text) -> text

returns: text

view_owner_is(char[], char[]) -> text

returns: text

views_are(char[], name[], text) -> text

returns: text

views_are(name[], text) -> text

returns: text

views_are(char[], name[]) -> text

returns: text

views_are(name[]) -> text

returns: text

volatility_is(char[], char[], name[], text, text) -> text

returns: text

volatility_is(char[], char[], name[], text) -> text

returns: text

volatility_is(char[], char[], text, text) -> text

returns: text

volatility_is(char[], char[], text) -> text

returns: text

volatility_is(char[], name[], text, text) -> text

returns: text

volatility_is(char[], name[], text) -> text

returns: text

volatility_is(char[], text, text) -> text

returns: text

volatility_is(char[], text) -> text

returns: text

wal_location_to_int(text) -> bigint

returns: bigint

Convert a textual WAL location in the form of '1752F/CDC6E050' into a bigint. Use this function to monitor slave delay on the master:

```
SELECT client_addr,          (public.wal_location_to_int(pg_current_xlog_location()))          -          pub-
      lic.wal_location_to_int(replay_location)) / 2^20 AS distance_mb
FROM pg_stat_replication;
```

3.13 relation

Stores directional relations between entities.

3.13.1 Types

3.13.2 Tables

all

Name	Type	Description
source_id	integer	
target_id	integer	
type_id	integer	

all_materialized

Name	Type	Description
source_id	integer	
target_id	integer	
type_id	integer	

group

Name	Type	Description
name	character varying	
id	integer	

self

Name	Type	Description
source_id	integer	
target_id	integer	
type_id	integer	

type

Name	Type	Description
name	character varying	
cardinality	relation.type_cardinality_enum	
group_id	integer	
id	integer	

3.13.3 Views

dependencies

Name	Type	Description
type	relation.type	
depth	integer	

materialization_order

Name	Type	Description
id	integer	
name	character varying	
cardinality	relation.type_cardinality_enum	
group_id	integer	
depth	integer	

3.13.4 Functions

Name	Return Type	Description
<i>create_all_materialized(char[])</i>	char[]	
<i>create_all_materialized_indexes(char[])</i>	char[]	
<i>create_relation_table(name text, type_id integer)</i>	void	
<i>create_relation_table_on_insert()</i>	trigger	
<i>create_self_relation()</i>	trigger	
<i>create_type(character varying)</i>	relation.type	
<i>define(char[], text)</i>	relation.type	
<i>define_reverse(reverse char[], original char[])</i>	relation.type	
<i>define_reverse(reverse char[], original relation.type)</i>	relation.type	
<i>drop_table_on_type_delete()</i>	trigger	
<i>get_type(character varying)</i>	relation.type	
<i>materialize_relation(type relation.type)</i>	integer	
<i>name_to_type(character varying)</i>	relation.type	
<i>populate_all_materialized(char[])</i>	char[]	
<i>replace_all_materialized(char[])</i>	char[]	
<i>set_view_permissions(relation.type)</i>	relation.type	
<i>update(relation.type, text)</i>	relation.type	
<i>update_all_materialized(intermediate_name char[])</i>	char[]	

create_all_materialized(char[]) -> char[]

returns: char[]

create_all_materialized_indexes(char[]) -> char[]

returns: char[]

create_relation_table(name text, type_id integer) -> void

returns: void

create_relation_table_on_insert() -> trigger

returns: trigger

create_self_relation() -> trigger

returns: trigger

create_type(character varying) -> relation.type

returns: *relation.type*

define(char[], text) -> relation.type

returns: *relation.type*

define_reverse(reverse char[], original char[]) -> relation.type

returns: *relation.type*

define_reverse(reverse char[], original relation.type) -> relation.type

returns: *relation.type*

drop_table_on_type_delete() -> trigger

returns: trigger

get_type(character varying) -> relation.type

returns: *relation.type*

materialize_relation(type relation.type) -> integer

returns: integer

name_to_type(character varying) -> relation.type

returns: *relation.type*

populate_all_materialized(char[]) -> char[]

returns: char[]

replace_all_materialized(char[]) -> char[]

returns: char[]

set_view_permissions(relation.type) -> relation.type

returns: *relation.type*

update(relation.type, text) -> relation.type

returns: *relation.type*

update_all_materialized(intermediate_name char[]) -> char[]

returns: char[]

3.14 relation_def

3.14.1 Types

3.14.2 Tables

3.14.3 Views

self

Name	Type	Description
source_id	integer	
target_id	integer	

3.14.4 Functions

Name	Return Type	Description

3.15 system

3.15.1 Types

job_type

Name	Type	Description
id	integer	
type	character varying	
description	character varying	
size	bigint	
config	text	

version_tuple

Name	Type	Description
major	smallint	
minor	smallint	
patch	smallint	

3.15.2 Tables

job

Name	Type	Description
type	character varying	
description	character varying	
size	bigint	
started	timestamp with time zone	
finished	timestamp with time zone	
job_source_id	integer	
created	timestamp with time zone	
state	system.job_state_enum	
id	integer	

job_error_log

Name	Type	Description
job_id	integer	
message	character varying	

job_queue

Name	Type	Description
job_id	integer	

job_source

Name	Type	Description
name	character varying	
job_type	character varying	
config	character varying	
id	integer	

setting

Name	Type	Description
name	text	
value	text	
id	integer	

3.15.3 Views

3.15.4 Functions

Name	Return Type	Description
<i>add_job_source(character varying, character varying, character varying)</i>	integer	
<i>add_setting(name text, value text)</i>	system.setting	
<i>create_job(type character varying, description character varying, size bigint, job_source_id integer)</i>	integer	
<i>fail_job(job_id integer)</i>	void	
<i>fail_job(job_id integer, message character varying)</i>	void	
<i>finish_job(job_id integer)</i>	void	
<i>get_job()</i>	system.job_type	
<i>get_job_source(integer)</i>	TABLE(character varying, character varying, character varying)	
<i>get_setting(name text)</i>	system.setting	
<i>get_setting_value(name text)</i>	text	
<i>get_setting_value(name text, default text)</i>	text	
<i>remove_jobs(before timestamp with time zone)</i>	integer	
<i>set_setting(name text, value text)</i>	system.setting	
<i>set_version(system.version_tuple)</i>	system.version_tuple	
<i>set_version(integer, integer, integer)</i>	system.version_tuple	
<i>update_setting(name text, value text)</i>	system.setting	
<i>version()</i>	system.version_tuple	
<i>version_glt_version(system.version_tuple, system.version_tuple)</i>	boolean	

add_job_source(character varying, character varying, character varying) -> integer

returns: integer

add_setting(name text, value text) -> system.setting

returns: *system.setting*

**create_job(type character varying, description character varying, size bigint, job_source_id integer)
-> integer**

returns: integer

fail_job(job_id integer) -> void

returns: void

fail_job(job_id integer, message character varying) -> void

returns: void

finish_job(job_id integer) -> void

returns: void

get_job() -> system.job_type

returns: *system.job_type*

get_job_source(integer) -> TABLE(character varying, character varying, character varying)

returns: TABLE(character varying, character varying, character varying)

get_setting(name text) -> system.setting

returns: *system.setting*

get_setting_value(name text) -> text

returns: text

get_setting_value(name text, default text) -> text

returns: text

remove_jobs(before timestamp with time zone) -> integer

returns: integer

set_setting(name text, value text) -> system.setting

returns: *system.setting*

set_version(system.version_tuple) -> system.version_tuple

returns: *system.version_tuple*

set_version(integer, integer, integer) -> system.version_tuple

returns: *system.version_tuple*

update_setting(name text, value text) -> system.setting

returns: *system.setting*

version() -> system.version_tuple

returns: *system.version_tuple*

version_gtlt_version(system.version_tuple, system.version_tuple) -> boolean

returns: boolean

3.16 trend

Stores information with fixed interval and format, like periodic measurements.

3.16.1 Types

column_info

Name	Type	Description
name	character varying	
datatype	character varying	

transfer_result

Name	Type	Description
row_count	integer	
max_modified	timestamp with time zone	

trend_descr

Name	Type	Description
name	char[]	
datatype	character varying	
description	text	

trend_with_type

Name	Type	Description
id	integer	
name	character varying	
data_type	character varying	

upgrade_record

Name	Type	Description
timestamp	timestamp with time zone	
number_of_rows	integer	

3.16.2 Tables**attribute_to_trend**

Name	Type	Description
attributestore_id	integer	
granularity	character varying	
id	integer	
enabled	boolean	

attribute_to_trend_state

Name	Type	Description
attribute_to_trend_id	integer	
timestamp	timestamp with time zone	
processed_modified	timestamp with time zone	

modified

Name	Type	Description
timestamp	timestamp with time zone	
table_name	character varying	
start	timestamp with time zone	
end	timestamp with time zone	

partition

Name	Type	Description
table_name	char[]	
trendstore_id	integer	
data_start	timestamp with time zone	
data_end	timestamp with time zone	
version	integer	

to_be_vacuumed

Name	Type	Description
table_name	char[]	

trend

Name	Type	Description
name	character varying	
description	character varying	
id	integer	

trend_tag_link

Name	Type	Description
trend_id	integer	
tag_id	integer	

trendstore

Name	Type	Description
entitytype_id	integer	
datasource_id	integer	
granularity	character varying	
partition_size	integer	
type	trend.storetype	
version	integer	
retention_period	interval	
id	integer	

trendstore_trend_link

Name	Type	Description
trendstore_id	integer	
trend_id	integer	

view

Name	Type	Description
description	character varying	
trendstore_id	integer	
sql	text	
id	integer	

view_trendstore_link

Name	Type	Description
view_id	integer	
trendstore_id	integer	

3.16.3 Views**attribute_to_trend_todo**

Name	Type	Description
done	boolean	
attribute_to_trend_id	integer	
granularity	character varying	
timestamp	timestamp with time zone	
processed_modified	timestamp with time zone	
compacted	timestamp with time zone	

view_dependencies

Name	Type	Description
src	char[]	
column_name	char[]	
dst	char[]	

3.16.4 Functions

Name
<i>add_trend_to_trendstore(trendstore_obj trend.trendstore, trend trend.trend_with_type)</i>
<i>add_trend_to_trendstore(trendstore trend.trendstore, trend_name char[], data_type character varying)</i>
<i>alter_column_types(namespace_name char[], table_name char[], columns column_info[])</i>
<i>alter_view(trend.view, text)</i>
<i>attribute_at_trend_ptr_table_name(trend.attribute_to_trend)</i>
<i>attribute_at_trend_view_name(trend.attribute_to_trend)</i>
<i>attribute_at_trend_view_sql(trend.attribute_to_trend)</i>
<i>attribute_at_updated(attribute_to_trend_id integer, timestamp timestamp with time zone)</i>
<i>attribute_at_updated(trend.trendstore, timestamp timestamp with time zone)</i>
<i>attributes_to_partition(trendstore trend.trendstore, index integer)</i>
<i>attributes_to_trend(trend.trendstore, char[])</i>

Table 3.6 – continued from

Name
<i>attributes_to_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying)</i>
<i>attributes_to_view_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying)</i>
<i>available_timestamps(partition trend.partition)</i>
<i>changes_on_datasource_update()</i>
<i>changes_on_partition_update()</i>
<i>changes_on_trend_update()</i>
<i>cleanup_on_datasource_delete()</i>
<i>cleanup_trendstore_on_delete()</i>
<i>clear_attribute_at_ptr(trend.attribute_to_trend, timestamp with time zone)</i>
<i>clear_attribute_at_ptr_sql(trend.attribute_to_trend, timestamp with time zone)</i>
<i>cluster_table_on_timestamp(name text)</i>
<i>column_exists(table_name character varying, column_name character varying)</i>
<i>create_attribute_at_trend_ptr_table(trend.attribute_to_trend)</i>
<i>create_attribute_at_trend_ptr_table_sql(trend.attribute_to_trend)</i>
<i>create_attribute_at_trend_view(trend.attribute_to_trend)</i>
<i>create_attribute_at_trend_view_sql(trend.attribute_to_trend)</i>
<i>create_attribute_at_trendstore(attribute_directory.attribute_store, granularity text)</i>
<i>create_attribute_to_trend_view(trend.attribute_to_trend)</i>
<i>create_base_table_on_insert()</i>
<i>create_partition(trendstore trend.trendstore, index integer)</i>
<i>create_partition_column(partition_name character varying, trend_id integer, datatype character varying)</i>
<i>create_partition_table(name text)</i>
<i>create_partition_table_on_insert()</i>
<i>create_partition_table_v4(base_name text, name text, data_start timestamp with time zone, data_end timestamp with time zone)</i>
<i>create_staging_table(trendstore trend.trendstore)</i>
<i>create_staging_table_sql(trendstore trend.trendstore)</i>
<i>create_trend(name character varying, description character varying)</i>
<i>create_trend_for_trendstore(trendstore trend.trendstore, trend_name character varying)</i>
<i>create_trends(trend.trendstore, trend_descr[])</i>
<i>create_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying)</i>
<i>create_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying, type trend..)</i>
<i>create_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying, trends trend..)</i>
<i>create_view(trend.view)</i>
<i>create_view(text)</i>
<i>create_view_sql(trend.view)</i>
<i>create_view_trends(view trend.view)</i>
<i>datatype_order(datatype character varying)</i>
<i>define_attribute_to_trend(attribute_directory.attribute_store, granularity text)</i>
<i>define_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying)</i>
<i>define_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying, type trend..)</i>
<i>define_view(trendstore_id integer, sql text)</i>
<i>define_view(trend.trendstore, sql text)</i>
<i>delete_view_trends(view trend.view)</i>
<i>drop_attribute_at_trend_ptr_table(trend.attribute_to_trend)</i>
<i>drop_attribute_at_trend_view(trend.attribute_to_trend)</i>
<i>drop_attribute_at_trend_view_sql(trend.attribute_to_trend)</i>
<i>drop_partition_table_on_delete()</i>
<i>drop_view(view trend.view)</i>
<i>drop_view_on_delete()</i>

Table 3.6 – continued from

Name
<i>generate_table_name(datasource_id integer, entitytype_id integer, granularity character varying, data_start timestamp with time zone)</i>
<i>get_attribute_to_trend(trend.trendstore)</i>
<i>get_column_names(table_name character varying)</i>
<i>get_default_partition_size(granularity character varying)</i>
<i>get_dependent_view_names(table_name char[])</i>
<i>get_dependent_view_names(table_name char[], column_name char[])</i>
<i>get_dependent_view_names(table_name char[], column_names name[])</i>
<i>get_dependent_views(table_name char[])</i>
<i>get_dependent_views(trend.trendstore)</i>
<i>get_dependent_views(trendstore_id integer)</i>
<i>get_dependent_views(table_name char[], column_name char[])</i>
<i>get_dependent_views(table_name char[], column_names name[])</i>
<i>get_index_on(character varying, character varying)</i>
<i>get_max_modified(trend.trendstore, timestamp with time zone)</i>
<i>get_most_recent_timestamp(dest_granularity integer, ts timestamp with time zone)</i>
<i>get_most_recent_timestamp(dest_granularity character varying, ts timestamp with time zone)</i>
<i>get_partition(trendstore trend.trendstore, index integer)</i>
<i>get_timestamp_for(granularity integer, ts timestamp with time zone)</i>
<i>get_timestamp_for(granularity character varying, ts timestamp with time zone)</i>
<i>get_trend(trendstore trend.trendstore, trend_name character varying)</i>
<i>get_trends(trendstore_id integer)</i>
<i>get_trends_for_v3_trendstore(trendstore_obj trend.trendstore)</i>
<i>get_trends_for_v4_trendstore(trendstore_obj trend.trendstore)</i>
<i>get_trendstore(view trend.view)</i>
<i>get_trendstore(id integer)</i>
<i>get_trendstore(trend.attribute_to_trend)</i>
<i>get_trendstore_by_attributes(datasource_name character varying, entitytype_name character varying, granularity character varying)</i>
<i>get_view_column_names(view_name character varying)</i>
<i>granularity_seconds(text)</i>
<i>granularity_to_text(granularity character varying)</i>
<i>greatest_datatype(datatype_a character varying, datatype_b character varying)</i>
<i>index_to_timestamp(partition_size integer, index integer)</i>
<i>infer_trendstore_type(trend.trendstore)</i>
<i>initialize_trendstore(trend.trendstore)</i>
<i>insert_attribute_to_trend_state(trend.attribute_to_trend, timestamp with time zone, timestamp with time zone)</i>
<i>is_integer(character varying)</i>
<i>link_view_dependencies(trend.view)</i>
<i>mark_attribute_to_trend_as_processed(trend.attribute_to_trend, timestamp with time zone, timestamp with time zone)</i>
<i>mark_modified(table_name char[], timestamp timestamp with time zone, modified timestamp with time zone)</i>
<i>mark_modified(table_name char[], timestamp timestamp with time zone)</i>
<i>max_datatype(character varying)</i>
<i>modified(trend.trendstore, timestamp with time zone)</i>
<i>modify_partition_column(partition_name character varying, column_name character varying, datatype character varying)</i>
<i>modify_trendstore_column(trendstore_id integer, column_name character varying, datatype character varying)</i>
<i>modify_trendstore_columns(trendstore_id integer, columns column_info[])</i>
<i>parse_granularity(character varying)</i>
<i>partition_exists(table_name character varying)</i>
<i>partition_name(trendstore trend.trendstore, index integer)</i>
<i>partition_name(trendstore trend.trendstore, timestamp with time zone)</i>

Table 3.6 – continued from

Name
<i>populate_attribute_at_ptr(trend.attribute_to_trend, timestamp with time zone)</i>
<i>populate_attribute_at_ptr_sql(trend.attribute_to_trend, timestamp with time zone)</i>
<i>populate_modified(partition trend.partition)</i>
<i>populate_modified(trend.trendstore)</i>
<i>populate_modified(character varying)</i>
<i>recreate_view(view trend.view)</i>
<i>recreate_view(text)</i>
<i>remove_trend_from_trendstore(trendstore trend.trendstore, trend_name character varying)</i>
<i>remove_trend_from_trendstore(trendstore text, trend_name character varying)</i>
<i>render_view_query(view_id integer)</i>
<i>set_trendstore_defaults()</i>
<i>show_trends(trend.trendstore)</i>
<i>show_trends(trendstore_id integer)</i>
<i>staging_table_name(trend.trendstore)</i>
<i>store_modified(table_name char[], timestamp timestamp with time zone, modified timestamp with time zone)</i>
<i>table_columns(namespace char[], table char[])</i>
<i>timestamp_to_index(partition_size integer, timestamp timestamp with time zone)</i>
<i>to_base_table_name(trendstore trend.trendstore)</i>
<i>to_char(trend.trendstore)</i>
<i>to_char(trend.view)</i>
<i>to_char(trend.attribute_to_trend)</i>
<i>to_table_name_v3(partition trend.partition)</i>
<i>to_table_name_v4(partition trend.partition)</i>
<i>to_trendstore(text)</i>
<i>transfer(source trend.trendstore, target trend.trendstore, timestamp timestamp with time zone, trend_names text[])</i>
<i>transfer_staged(trendstore trend.trendstore)</i>
<i>trendstore_has_trend_with_name(trendstore trend.trendstore, trend_name character varying)</i>
<i>unlink_view_dependencies(trend.view)</i>
<i>update_attribute_at_ptr(trend.attribute_to_trend, timestamp with time zone)</i>
<i>update_attribute_to_trend_state(trend.attribute_to_trend, timestamp with time zone, timestamp with time zone)</i>
<i>update_modified(table_name char[], timestamp timestamp with time zone, modified timestamp with time zone)</i>
<i>update_modified_column()</i>
<i>update_view_sql(trend.view, text)</i>
<i>view_name(trend.view)</i>

add_trend_to_trendstore(trendstore_obj trend.trendstore, trend trend.trend_with_type) -> void

returns: void

add_trend_to_trendstore(trendstore trend.trendstore, trend_name char[], data_type character varying) -> trend.trend

returns: *trend.trend*

alter_column_types(namespace_name char[], table_name char[], columns column_info[]) -> void

returns: void

alter_view(trend.view, text) -> trend.view

returns: *trend.view*

attribute_at_trend_ptr_table_name(trend.attribute_to_trend) -> char[]

returns: char[]

attribute_at_trend_view_name(trend.attribute_to_trend) -> char[]

returns: char[]

attribute_at_trend_view_sql(trend.attribute_to_trend) -> text

returns: text

attribute_at_updated(attribute_to_trend_id integer, timestamp timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

attribute_at_updated(trend.trendstore, timestamp timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

attributes_to_partition(trendstore trend.trendstore, index integer) -> trend.partition

returns: *trend.partition*

attributes_to_trend(trend.trendstore, char[]) -> trend.trend

returns: *trend.trend*

attributes_to_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying) -> trend.trendstore

returns: *trend.trendstore*

attributes_to_view_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying) -> trend.trendstore

returns: *trend.trendstore*

available_timestamps(partition trend.partition) -> SETOF timestamp with time zone

returns: timestamp with time zone

changes_on_datasource_update() -> trigger

returns: trigger

changes_on_partition_update() -> trigger

returns: trigger

changes_on_trend_update() -> trigger

returns: trigger

cleanup_on_datasource_delete() -> trigger

returns: trigger

cleanup_trendstore_on_delete() -> trigger

returns: trigger

clear_attribute_at_ptr(trend.attribute_to_trend, timestamp with time zone) ->
trend.attribute_to_trend

returns: *trend.attribute_to_trend*

clear_attribute_at_ptr_sql(trend.attribute_to_trend, timestamp with time zone) -> text

returns: text

cluster_table_on_timestamp(name text) -> void

returns: void

column_exists(table_name character varying, column_name character varying) -> boolean

returns: boolean

create_attribute_at_trend_ptr_table(trend.attribute_to_trend) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

create_attribute_at_trend_ptr_table_sql(trend.attribute_to_trend) -> text[]

returns: text[]

create_attribute_at_trend_view(trend.attribute_to_trend) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

create_attribute_at_trend_view_sql(trend.attribute_to_trend) -> text

returns: text

create_attribute_at_trendstore(attribute_directory.attributestore, granularity text) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

create_attribute_to_trend_view(trend.attribute_to_trend) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

create_base_table_on_insert() -> trigger

returns: trigger

create_partition(trendstore trend.trendstore, index integer) -> trend.partition

returns: *trend.partition*

create_partition_column(partition_name character varying, trend_id integer, datatype character varying) -> void

returns: void

create_partition_table(name text) -> void

returns: void

create_partition_table_on_insert() -> trigger

returns: trigger

create_partition_table_v4(base_name text, name text, data_start timestamp with time zone, data_end timestamp with time zone) -> void

returns: void

create_staging_table(trendstore trend.trendstore) -> trend.trendstore

returns: *trend.trendstore*

create_staging_table_sql(trendstore trend.trendstore) -> text[]

returns: text[]

create_trend(name character varying, description character varying) -> trend.trend

returns: *trend.trend*

create_trend_for_trendstore(trendstore trend.trendstore, trend_name character varying) -> trend.trend

returns: *trend.trend*

create_trends(trend.trendstore, trend_descr[]) -> SETOF trend.trend

returns: *trend.trend*

create_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying) -> trend.trendstore

returns: *trend.trendstore*

create_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying, type trend.storetype) -> trend.trendstore

returns: *trend.trendstore*

create_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying, trends trend_descr[]) -> trend.trendstore

returns: *trend.trendstore*

create_view(trend.view) -> trend.view

returns: *trend.view*

create_view(text) -> trend.view

returns: *trend.view*

create_view_sql(trend.view) -> text[]

returns: text[]

create_view_trends(view trend.view) -> SETOF trend.trend

returns: *trend.trend*

datatype_order(datatype character varying) -> integer

returns: integer

define_attribute_to_trend(attribute_directory.attributestore, granularity text) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

define_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying) -> trend.trendstore

returns: *trend.trendstore*

Add a new trendstore record, initialize the trendstore base table, and return the new record.

Later on, the definition and initialization will be split into separate steps, but the old mechanism still uses triggers that automatically initialize the trendstore.

define_trendstore(datasource_name character varying, entitytype_name character varying, granularity character varying, type trend.storetype) -> trend.trendstore

returns: *trend.trendstore*

define_view(trendstore_id integer, sql text) -> trend.view

returns: *trend.view*

define_view(trend.trendstore, sql text) -> trend.view

returns: *trend.view*

delete_view_trends(view trend.view) -> void

returns: void

drop_attribute_at_trend_ptr_table(trend.attribute_to_trend) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

drop_attribute_at_trend_view(trend.attribute_to_trend) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

drop_attribute_at_trend_view_sql(trend.attribute_to_trend) -> text

returns: text

drop_partition_table_on_delete() -> trigger

returns: trigger

drop_view(view trend.view) -> trend.view

returns: *trend.view*

drop_view_on_delete() -> trigger

returns: trigger

generate_table_name(datasource_id integer, entitytype_id integer, granularity character varying, data_start timestamp with time zone) -> text

returns: text

get_attribute_to_trend(trend.trendstore) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

get_column_names(table_name character varying) -> varchar[]

returns: varchar[]

get_default_partition_size(granularity character varying) -> integer

returns: integer

get_dependent_view_names(table_name char[]) -> SETOF char[]

returns: char[]

get_dependent_view_names(table_name char[], column_name char[]) -> SETOF char[]

returns: char[]

get_dependent_view_names(table_name char[], column_names name[]) -> SETOF char[]

returns: char[]

get_dependent_views(table_name char[]) -> SETOF trend.view

returns: *trend.view*

get_dependent_views(trend.trendstore) -> SETOF trend.view

returns: *trend.view*

get_dependent_views(trendstore_id integer) -> SETOF trend.view

returns: *trend.view*

get_dependent_views(table_name char[], column_name char[]) -> SETOF trend.view

returns: *trend.view*

get_dependent_views(table_name char[], column_names name[]) -> SETOF trend.view

returns: *trend.view*

get_index_on(character varying, character varying) -> char[]

returns: char[]

get_max_modified(trend.trendstore, timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

get_most_recent_timestamp(dest_granularity integer, ts timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

get_most_recent_timestamp(dest_granularity character varying, ts timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

get_partition(trendstore trend.trendstore, index integer) -> trend.partition

returns: *trend.partition*

get_timestamp_for(granularity integer, ts timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

get_timestamp_for(granularity character varying, ts timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

get_trend(trendstore trend.trendstore, trend_name character varying) -> trend.trend

returns: *trend.trend*

get_trends(trendstore_id integer) -> SETOF trend.trend_with_type

returns: *trend.trend_with_type*

get_trends_for_v3_trendstore(trendstore_obj trend.trendstore) -> SETOF trend.trend_with_type

returns: *trend.trend_with_type*

get_trends_for_v4_trendstore(trendstore_obj trend.trendstore) -> SETOF trend.trend_with_type

returns: *trend.trend_with_type*

get_trendstore(view trend.view) -> trend.trendstore

returns: *trend.trendstore*

get_trendstore(id integer) -> trend.trendstore

returns: *trend.trendstore*

get_trendstore(trend.attribute_to_trend) -> trend.trendstore

returns: *trend.trendstore*

get_trendstore_by_attributes(datasource_name character varying, entitytype_name character varying, granularity character varying) -> trend.trendstore

returns: *trend.trendstore*

get_view_column_names(view_name character varying) -> SETOF char[]

returns: *char[]*

granularity_seconds(text) -> integer

returns: *integer*

granularity_to_text(granularity character varying) -> text

returns: *text*

greatest_datatype(datatype_a character varying, datatype_b character varying) -> character varying

returns: character varying

index_to_timestamp(partition_size integer, index integer) -> timestamp with time zone

returns: timestamp with time zone

infer_trendstore_type(trend.trendstore) -> trend.storetype

returns: trend.storetype

initialize_trendstore(trend.trendstore) -> trend.trendstore

returns: *trend.trendstore*

Create all database objects required for the trendstore to be fully functional and capable of storing data.

insert_attribute_to_trend_state(trend.attribute_to_trend, timestamp with time zone, timestamp with time zone) -> trend.attribute_to_trend_state

returns: *trend.attribute_to_trend_state*

is_integer(character varying) -> boolean

returns: boolean

link_view_dependencies(trend.view) -> trend.view

returns: *trend.view*

mark_attribute_to_trend_as_processed(trend.attribute_to_trend, timestamp with time zone, timestamp with time zone) -> trend.attribute_to_trend_state

returns: *trend.attribute_to_trend_state*

mark_modified(table_name char[], timestamp timestamp with time zone, modified timestamp with time zone) -> trend.modified

returns: *trend.modified*

mark_modified(table_name char[], timestamp timestamp with time zone) -> trend.modified

returns: *trend.modified*

max_datatype(character varying) -> character varying

returns: character varying

modified(trend.trendstore, timestamp with time zone) -> timestamp with time zone

returns: timestamp with time zone

modify_partition_column(partition_name character varying, column_name character varying, datatype character varying) -> void

returns: void

modify_trendstore_column(trendstore_id integer, column_name character varying, datatype character varying) -> void

returns: void

modify_trendstore_columns(trendstore_id integer, columns column_info[]) -> void

returns: void

parse_granularity(character varying) -> interval

returns: interval

partition_exists(table_name character varying) -> boolean

returns: boolean

partition_name(trendstore trend.trendstore, index integer) -> char[]

returns: char[]

partition_name(trendstore trend.trendstore, timestamp with time zone) -> char[]

returns: char[]

populate_attribute_at_ptr(trend.attribute_to_trend, timestamp with time zone) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

populate_attribute_at_ptr_sql(trend.attribute_to_trend, timestamp with time zone) -> text

returns: text

populate_modified(partition trend.partition) -> SETOF trend.modified

returns: *trend.modified*

populate_modified(trend.trendstore) -> SETOF trend.modified

returns: *trend.modified*

populate_modified(character varying) -> SETOF trend.modified

returns: *trend.modified*

recreate_view(view trend.view) -> trend.view

returns: *trend.view*

recreate_view(text) -> trend.view

returns: *trend.view*

remove_trend_from_trendstore(trendstore trend.trendstore, trend_name character varying) -> void

returns: void

remove_trend_from_trendstore(trendstore text, trend_name character varying) -> void

returns: void

render_view_query(view_id integer) -> text

returns: text

set_trendstore_defaults() -> trigger

returns: trigger

show_trends(trend.trendstore) -> SETOF trend.trend_descr

returns: *trend.trend_descr*

show_trends(trendstore_id integer) -> SETOF trend.trend_descr

returns: *trend.trend_descr*

staging_table_name(trend.trendstore) -> char[]

returns: char[]

store_modified(table_name char[], timestamp timestamp with time zone, modified timestamp with time zone) -> trend.modified

returns: *trend.modified*

table_columns(namespace char[], table char[]) -> SETOF trend.column_info

returns: *trend.column_info*

timestamp_to_index(partition_size integer, timestamp timestamp with time zone) -> integer

returns: integer

to_base_table_name(trendstore trend.trendstore) -> text

returns: text

to_char(trend.trendstore) -> text

returns: text

to_char(trend.view) -> text

returns: text

to_char(trend.attribute_to_trend) -> text

returns: text

to_table_name_v3(partition trend.partition) -> text

returns: text

to_table_name_v4(partition trend.partition) -> text

returns: text

to_trendstore(text) -> trend.trendstore

returns: *trend.trendstore*

transfer(source trend.trendstore, target trend.trendstore, timestamp timestamp with time zone, trend_names text[]) -> trend.transfer_result

returns: *trend.transfer_result*

transfer_staged(trendstore trend.trendstore) -> void

returns: void

trendstore_has_trend_with_name(trendstore trend.trendstore, trend_name character varying) -> boolean

returns: boolean

unlink_view_dependencies(trend.view) -> trend.view

returns: *trend.view*

update_attribute_at_ptr(trend.attribute_to_trend, timestamp with time zone) -> trend.attribute_to_trend

returns: *trend.attribute_to_trend*

update_attribute_to_trend_state(trend.attribute_to_trend, timestamp with time zone, timestamp with time zone) -> trend.attribute_to_trend_state

returns: *trend.attribute_to_trend_state*

update_modified(table_name char[], timestamp timestamp with time zone, modified timestamp with time zone) -> trend.modified

returns: *trend.modified*

update_modified_column() -> trigger

returns: trigger

update_view_sql(trend.view, text) -> trend.view

returns: *trend.view*

view_name(trend.view) -> character varying

returns: character varying

3.17 trigger

3.17.1 Types

kpi_def

Name	Type	Description
name	char[]	
data_type	char[]	

notification

Name	Type	Description
entity_id	integer	
timestamp	timestamp with time zone	
weight	integer	
details	text	

threshold_def

Name	Type	Description
name	char[]	
data_type	char[]	

3.17.2 Tables

exception_base

Name	Type	Description
entity_id	integer	
start	timestamp with time zone	
expires	timestamp with time zone	
id	integer	
created	timestamp with time zone	

rule

Name	Type	Description
name	char[]	
notificationstore_id	integer	
granularity	interval	
default_interval	interval	
id	integer	
enabled	boolean	

rule_state

Name	Type	Description
rule_id	integer	
timestamp	timestamp with time zone	
fingerprint	text	

rule_tag_link

Name	Type	Description
rule_id	integer	
tag_id	integer	

3.17.3 Views**todo**

Name	Type	Description
rule	trigger.rule	
timestamp	timestamp with time zone	
modified	boolean	

3.17.4 Functions

Name	Return Type	
<i>add_rule(char[])</i>	trigger.rule	
<i>cleanup_on_rule_delete()</i>	trigger	
<i>cleanup_rule(trigger.rule)</i>	trigger.rule	
<i>contains_null(anyarray)</i>	boolean	
<i>create_details_type(trigger.rule, threshold_def[])</i>	trigger.rule	
<i>create_details_type_sql(trigger.rule, threshold_def[])</i>	text	
<i>create_dummy_default_weight(trigger.rule)</i>	trigger.rule	
<i>create_dummy_notification_message_fn(trigger.rule)</i>	trigger.rule	
<i>create_dummy_thresholds(trigger.rule, threshold_def[])</i>	trigger.rule	
<i>create_exception_threshold_table(trigger.rule, threshold_def[])</i>	trigger.rule	
<i>create_exception_threshold_table_sql(trigger.rule, threshold_def[])</i>	text	
<i>create_exception_weight_table(trigger.rule)</i>	trigger.rule	Cr
<i>create_fingerprint_fn(trigger.rule)</i>	trigger.rule	
<i>create_fingerprint_fn_sql(trigger.rule, fn_sql text)</i>	text	
<i>create_fingerprint_fn_sql(trigger.rule)</i>	text	
<i>create_notification_fn(trigger.rule)</i>	trigger.rule	
<i>create_notification_fn_sql(trigger.rule)</i>	text[]	
<i>create_notification_message_fn(trigger.rule, expression text)</i>	trigger.rule	
<i>create_notifications(rule_name char[], interval)</i>	integer	
<i>create_notifications(rule_name char[])</i>	integer	
<i>create_notifications(trigger.rule, notification.notificationstore, timestamp with time zone)</i>	integer	
<i>create_notifications(trigger.rule, timestamp with time zone)</i>	integer	
<i>create_notifications(trigger.rule, notification.notificationstore, interval)</i>	integer	

Table 3.7 – continued from previous page

Name	Return Type	De
<i>create_notifications(trigger.rule, interval)</i>	integer	
<i>create_notifications(trigger.rule)</i>	integer	
<i>create_notifications(rule_name char[], notificationstore_name char[], timestamp with time zone)</i>	integer	
<i>create_notifications(rule_name char[], timestamp with time zone)</i>	integer	
<i>create_notifications(rule_name char[], notificationstore_name char[], interval)</i>	integer	
<i>create_notifications_classic(trigger.rule, notification.notificationstore, timestamp with time zone)</i>	integer	
<i>create_notifications_new(trigger.rule, notification.notificationstore, timestamp with time zone)</i>	integer	
<i>create_rule(char[], threshold_def[])</i>	trigger.rule	De
<i>create_rule_fn(trigger.rule, rule_view_sql text)</i>	trigger.rule	
<i>create_rule_fn_sql(trigger.rule, rule_view_sql text)</i>	text[]	
<i>create_runnable_fn(trigger.rule)</i>	trigger.rule	
<i>create_runnable_fn_sql(trigger.rule, fn_body text)</i>	text	
<i>create_runnable_fn_sql(trigger.rule)</i>	text	
<i>create_set_thresholds_fn(trigger.rule)</i>	trigger.rule	
<i>create_set_thresholds_fn_sql(trigger.rule)</i>	text	
<i>create_trigger_notificationstore(char[])</i>	notification.notificationstore	
<i>create_with_threshold_fn(trigger.rule)</i>	trigger.rule	
<i>define(char[])</i>	trigger.rule	
<i>define_notification(char[], expression text)</i>	trigger.rule	
<i>define_thresholds(trigger.rule, threshold_def[])</i>	trigger.rule	
<i>details_type_name(trigger.rule)</i>	char[]	
<i>drop_details_type(trigger.rule)</i>	trigger.rule	
<i>drop_details_type_sql(trigger.rule)</i>	text	
<i>drop_exception_threshold_table_sql(trigger.rule)</i>	text	
<i>drop_exception_weight_table_sql(trigger.rule)</i>	text	Re
<i>drop_fingerprint_fn(trigger.rule)</i>	trigger.rule	
<i>drop_fingerprint_fn_sql(trigger.rule)</i>	text	
<i>drop_kpi_fn_sql(trigger.rule)</i>	text	
<i>drop_notification_fn_sql(trigger.rule)</i>	text	
<i>drop_notification_message_fn_sql(trigger.rule)</i>	text	
<i>drop_rule_fn_sql(trigger.rule)</i>	text	
<i>drop_runnable_fn_sql(trigger.rule)</i>	text	
<i>drop_set_thresholds_fn_sql(trigger.rule)</i>	text	
<i>drop_thresholds_view_sql(trigger.rule)</i>	text	
<i>drop_weight_fn_sql(trigger.rule)</i>	text	Re
<i>drop_with_threshold_fn_sql(trigger.rule)</i>	text	
<i>exception_threshold_table_name(trigger.rule)</i>	char[]	
<i>exception_weight_table_name(trigger.rule)</i>	char[]	
<i>exception_weight_table_sql(trigger.rule)</i>	text	Re
<i>fingerprint(trigger.rule, timestamp with time zone)</i>	text	
<i>fingerprint_fn_name(trigger.rule)</i>	char[]	
<i>function_oid(obj_schema char[], obj_name char[], signature text[])</i>	oid	
<i>get_function_def(schema_name char[], fn_name char[])</i>	text	
<i>get_kpi_def(trigger.rule, char[])</i>	trigger.kpi_def	
<i>get_kpi_defs(trigger.rule)</i>	trigger.kpi_def	
<i>get_notification_message_fn_sql(trigger.rule)</i>	text	
<i>get_rule(char[])</i>	trigger.rule	Re
<i>get_threshold_defs(trigger.rule)</i>	trigger.threshold_def	
<i>get_weight_fn_sql(trigger.rule)</i>	text	Re

Table 3.7 – continued from previous page

Name	Return Type	De
<i>has_notification_function(trigger.rule)</i>	boolean	
<i>has_thresholds(trigger.rule)</i>	boolean	Re
<i>insert_state(integer, timestamp with time zone, text)</i>	trigger.rule_state	
<i>kpi_def_arr_from_proc(oid)</i>	kpi_def[]	
<i>kpi_def_arr_from_type(namespace char[], type char[])</i>	kpi_def[]	
<i>kpi_fn_name(trigger.rule)</i>	char[]	
<i>kpi_type_name(trigger.rule)</i>	char[]	
<i>modified_to_fingerprint(timestampt[])</i>	text	
<i>notification_fn_name(trigger.rule)</i>	char[]	
<i>notification_fn_sql(trigger.rule)</i>	text	
<i>notification_message_fn_name(trigger.rule)</i>	char[]	
<i>notification_message_fn_sql(trigger.rule, expression text)</i>	text	
<i>notification_test_threshold_fn_sql(trigger.rule)</i>	text	
<i>notification_threshold_test_fn_name(trigger.rule)</i>	char[]	
<i>rule_fn_name(trigger.rule)</i>	char[]	
<i>rule_fn_sql(trigger.rule, where_clause text)</i>	text	
<i>runnable_fn_name(trigger.rule)</i>	char[]	
<i>set_condition(trigger.rule, sql text)</i>	trigger.rule	
<i>set_condition(char[], sql text)</i>	trigger.rule	
<i>set_fingerprint(trigger.rule, fn_sql text)</i>	trigger.rule	
<i>set_fingerprint(char[], fn_sql text)</i>	char[]	
<i>set_runnable(trigger.rule, fn_sql text)</i>	trigger.rule	
<i>set_state(integer, timestamp with time zone, text)</i>	trigger.rule_state	
<i>set_state(trigger.rule, timestamp with time zone)</i>	trigger.rule	
<i>set_thresholds(trigger.rule, exprs text)</i>	trigger.rule	
<i>set_thresholds(char[], exprs text)</i>	trigger.rule	
<i>set_thresholds_fn_name(trigger.rule)</i>	char[]	
<i>set_weight(trigger.rule, expression text)</i>	trigger.rule	
<i>set_weight(char[], expression text)</i>	trigger.rule	
<i>setup_rule(trigger.rule, threshold_def[])</i>	trigger.rule	
<i>tag(tag_name character varying, rule_id integer)</i>	trigger.rule_tag_link	Ac
<i>tag(tag_name character varying, rule_name char[])</i>	trigger.rule_tag_link	
<i>threshold_view_name(trigger.rule)</i>	char[]	
<i>timestamps(trigger.rule)</i>	timestamp with time zone	
<i>transfer_notifications_from_staging(notification.notificationstore)</i>	integer	
<i>truncate(timestamp with time zone, interval)</i>	timestamp with time zone	
<i>update_state(integer, timestamp with time zone, text)</i>	trigger.rule_state	
<i>weight_fn_name(trigger.rule)</i>	char[]	
<i>weight_fn_sql(trigger.rule, expression text)</i>	text	Re
<i>with_threshold_fn_name(trigger.rule)</i>	char[]	
<i>with_threshold_fn_sql(trigger.rule)</i>	text	
<i>with_threshold_fn_sql_no_thresholds(trigger.rule)</i>	text	
<i>with_threshold_fn_sql_normal(trigger.rule)</i>	text	

add_rule(char[]) -> trigger.rule

returns: *trigger.rule*

cleanup_on_rule_delete() -> trigger

returns: trigger

cleanup_rule(trigger.rule) -> trigger.rule

returns: *trigger.rule*

contains_null(anyarray) -> boolean

returns: boolean

create_details_type(trigger.rule, threshold_def[]) -> trigger.rule

returns: *trigger.rule*

create_details_type_sql(trigger.rule, threshold_def[]) -> text

returns: text

create_dummy_default_weight(trigger.rule) -> trigger.rule

returns: *trigger.rule*

create_dummy_notification_message_fn(trigger.rule) -> trigger.rule

returns: *trigger.rule*

create_dummy_thresholds(trigger.rule, threshold_def[]) -> trigger.rule

returns: *trigger.rule*

create_exception_threshold_table(trigger.rule, threshold_def[]) -> trigger.rule

returns: *trigger.rule*

create_exception_threshold_table_sql(trigger.rule, threshold_def[]) -> text

returns: text

create_exception_weight_table(trigger.rule) -> trigger.rule

returns: *trigger.rule*

Create the exception weight table for specified rule.

create_fingerprint_fn(trigger.rule) -> trigger.rule

returns: *trigger.rule*

create_fingerprint_fn_sql(trigger.rule, fn_sql text) -> text

returns: text

create_fingerprint_fn_sql(trigger.rule) -> text

returns: text

create_notification_fn(trigger.rule) -> trigger.rule

returns: *trigger.rule*

create_notification_fn_sql(trigger.rule) -> text[]

returns: text[]

create_notification_message_fn(trigger.rule, expression text) -> trigger.rule

returns: *trigger.rule*

create_notifications(rule_name char[], interval) -> integer

returns: integer

create_notifications(rule_name char[]) -> integer

returns: integer

create_notifications(trigger.rule, notification.notificationstore, timestamp with time zone) -> integer

returns: integer

create_notifications(trigger.rule, timestamp with time zone) -> integer

returns: integer

create_notifications(trigger.rule, notification.notificationstore, interval) -> integer

returns: integer

create_notifications(trigger.rule, interval) -> integer

returns: integer

create_notifications(trigger.rule) -> integer

returns: integer

create_notifications(rule_name char[], notificationstore_name char[], timestamp with time zone) -> integer

returns: integer

create_notifications(rule_name char[], timestamp with time zone) -> integer

returns: integer

create_notifications(rule_name char[], notificationstore_name char[], interval) -> integer

returns: integer

create_notifications_classic(trigger.rule, notification.notificationstore, timestamp with time zone) -> integer

returns: integer

create_notifications_new(trigger.rule, notification.notificationstore, timestamp with time zone) -> integer

returns: integer

create_rule(char[], threshold_def[]) -> trigger.rule

returns: *trigger.rule*

Define a new rule and create accompanying functions and views.

Important: A KPI function <trigger_name>_kpi(timestamp with time zone) must already exist.

create_rule_fn(trigger.rule, rule_view_sql text) -> trigger.rule

returns: *trigger.rule*

create_rule_fn_sql(trigger.rule, rule_view_sql text) -> text[]

returns: text[]

create_runnable_fn(trigger.rule) -> trigger.rule

returns: *trigger.rule*

create_runnable_fn_sql(trigger.rule, fn_body text) -> text

returns: text

create_runnable_fn_sql(trigger.rule) -> text

returns: text

create_set_thresholds_fn(trigger.rule) -> trigger.rule

returns: *trigger.rule*

create_set_thresholds_fn_sql(trigger.rule) -> text

returns: text

create_trigger_notificationstore(char[]) -> notification.notificationstore

returns: *notification.notificationstore*

create_with_threshold_fn(trigger.rule) -> trigger.rule

returns: *trigger.rule*

define(char[]) -> trigger.rule

returns: *trigger.rule*

define_notification(char[], expression text) -> trigger.rule

returns: *trigger.rule*

define_thresholds(trigger.rule, threshold_def[]) -> trigger.rule

returns: *trigger.rule*

details_type_name(trigger.rule) -> char[]

returns: char[]

drop_details_type(trigger.rule) -> trigger.rule

returns: *trigger.rule*

drop_details_type_sql(trigger.rule) -> text

returns: text

drop_exception_threshold_table_sql(trigger.rule) -> text

returns: text

drop_exception_weight_table_sql(trigger.rule) -> text

returns: text

Return code to drop the exception weight table for specified rule.

drop_fingerprint_fn(trigger.rule) -> trigger.rule

returns: *trigger.rule*

drop_fingerprint_fn_sql(trigger.rule) -> text

returns: text

drop_kpi_fn_sql(trigger.rule) -> text

returns: text

drop_notification_fn_sql(trigger.rule) -> text

returns: text

drop_notification_message_fn_sql(trigger.rule) -> text

returns: text

drop_rule_fn_sql(trigger.rule) -> text

returns: text

drop_runnable_fn_sql(trigger.rule) -> text

returns: text

drop_set_thresholds_fn_sql(trigger.rule) -> text

returns: text

drop_thresholds_view_sql(trigger.rule) -> text

returns: text

drop_weight_fn_sql(trigger.rule) -> text

returns: text

Return code to drop weight function for specified rule.

drop_with_threshold_fn_sql(trigger.rule) -> text

returns: text

exception_threshold_table_name(trigger.rule) -> char[]

returns: char[]

exception_weight_table_name(trigger.rule) -> char[]

returns: char[]

exception_weight_table_sql(trigger.rule) -> text

returns: text

Return code to create the exception weight table for specified rule.

fingerprint(trigger.rule, timestamp with time zone) -> text

returns: text

fingerprint_fn_name(trigger.rule) -> char[]

returns: char[]

function_oid(obj_schema char[], obj_name char[], signature text[]) -> oid

returns: oid

get_function_def(schema_name char[], fn_name char[]) -> text

returns: text

get_kpi_def(trigger.rule, char[]) -> trigger.kpi_def

returns: *trigger.kpi_def*

get_kpi_defs(trigger.rule) -> SETOF trigger.kpi_def

returns: *trigger.kpi_def*

get_notification_message_fn_sql(trigger.rule) -> text

returns: text

get_rule(char[]) -> trigger.rule

returns: *trigger.rule*

Return rule with specified name.

get_threshold_defs(trigger.rule) -> SETOF trigger.threshold_def

returns: *trigger.threshold_def*

get_weight_fn_sql(trigger.rule) -> text

returns: text

Return current implementation of the weight function for specified rule.

has_notification_function(trigger.rule) -> boolean

returns: boolean

has_thresholds(trigger.rule) -> boolean

returns: boolean

Return true if there is a view with thresholds for the specified rule

insert_state(integer, timestamp with time zone, text) -> trigger.rule_state

returns: *trigger.rule_state*

kpi_def_arr_from_proc(oid) -> kpi_def[]

returns: kpi_def[]

kpi_def_arr_from_type(namespace char[], type char[]) -> kpi_def[]

returns: kpi_def[]

kpi_fn_name(trigger.rule) -> char[]

returns: char[]

kpi_type_name(trigger.rule) -> char[]

returns: char[]

modified_to_fingerprint(timestampz[]) -> text

returns: text

notification_fn_name(trigger.rule) -> char[]

returns: char[]

notification_fn_sql(trigger.rule) -> text

returns: text

notification_message_fn_name(trigger.rule) -> char[]

returns: char[]

notification_message_fn_sql(trigger.rule, expression text) -> text

returns: text

notification_test_threshold_fn_sql(trigger.rule) -> text

returns: text

notification_threshold_test_fn_name(trigger.rule) -> char[]

returns: char[]

rule_fn_name(trigger.rule) -> char[]

returns: char[]

rule_fn_sql(trigger.rule, where_clause text) -> text

returns: text

runnable_fn_name(trigger.rule) -> char[]

returns: char[]

set_condition(trigger.rule, sql text) -> trigger.rule

returns: *trigger.rule*

set_condition(char[], sql text) -> trigger.rule

returns: *trigger.rule*

set_fingerprint(trigger.rule, fn_sql text) -> trigger.rule

returns: *trigger.rule*

set_fingerprint(char[], fn_sql text) -> char[]

returns: char[]

set_runnable(trigger.rule, fn_sql text) -> trigger.rule

returns: *trigger.rule*

set_state(integer, timestamp with time zone, text) -> trigger.rule_state

returns: *trigger.rule_state*

set_state(trigger.rule, timestamp with time zone) -> trigger.rule

returns: *trigger.rule*

set_thresholds(trigger.rule, exprs text) -> trigger.rule

returns: *trigger.rule*

set_thresholds(char[], exprs text) -> trigger.rule

returns: *trigger.rule*

set_thresholds_fn_name(trigger.rule) -> char[]

returns: char[]

set_weight(trigger.rule, expression text) -> trigger.rule

returns: *trigger.rule*

set_weight(char[], expression text) -> trigger.rule

returns: *trigger.rule*

setup_rule(trigger.rule, threshold_def[]) -> trigger.rule

returns: *trigger.rule*

tag(tag_name character varying, rule_id integer) -> trigger.rule_tag_link

returns: *trigger.rule_tag_link*

Add tag with name tag_name to rule with id rule_id. The tag must already exist.

tag(tag_name character varying, rule_name char[]) -> trigger.rule_tag_link

returns: *trigger.rule_tag_link*

threshold_view_name(trigger.rule) -> char[]

returns: char[]

timestamps(trigger.rule) -> SETOF timestamp with time zone

returns: timestamp with time zone

transfer_notifications_from_staging(notification.notificationstore) -> integer

returns: integer

truncate(timestamp with time zone, interval) -> timestamp with time zone

returns: timestamp with time zone

update_state(integer, timestamp with time zone, text) -> trigger.rule_state

returns: *trigger.rule_state*

weight_fn_name(trigger.rule) -> char[]

returns: char[]

weight_fn_sql(trigger.rule, expression text) -> text

returns: text

Return code to create weight function based on the provided expression.

with_threshold_fn_name(trigger.rule) -> char[]

returns: char[]

with_threshold_fn_sql(trigger.rule) -> text

returns: text

with_threshold_fn_sql_no_thresholds(trigger.rule) -> text

returns: text

with_threshold_fn_sql_normal(trigger.rule) -> text

returns: text

3.18 trigger_rule

Holds trigger specific auto-generated code.

This schema is mostly automatically and dynamically populated when creating rules.

3.18.1 Types

3.18.2 Tables

3.18.3 Views

3.18.4 Functions

Name	Return Type	Description

3.19 virtual_entity

3.19.1 Types

3.19.2 Tables

3.19.3 Views

3.19.4 Functions

Name	Return Type	Description
<i>update(name char[])</i>	integer	

update(name char[]) -> integer

returns: integer

Indices and tables

- `genindex`
- `modindex`
- `search`