

R documentation

of all in 'pkg/REmbeddedPostgres/man'

April 11, 2001

R topics documented:

EmbeddedPostgres	1
tupleTypes	1
[[<-PostgresTupleRef	2
[.PostgresTupleRef	3
evalPostgresAggregatorInitialization	4
names.PostgresTupleRef	5
parsePostgresFunction	6
postgresError	7
setTupleElements	8

<code>EmbeddedPostgres</code>	<i>Indicates R being run from within Postgres</i>
-------------------------------	---

Description

This is a logical value that indicates that the R session is embedded in a Postgres database server.

<code>tupleTypes</code>	<i>Retrieves the values or the types of a Postgres Tuple's elements.</i>
-------------------------	--

Description

These functions allow the caller to access the values and descriptions of each of the elements in a Postgres tuple stored internally at the C-level.

Usage

```
tupleTypes(ref)
tupleValues(ref)
```

Arguments

ref an object of class `PostgresTupleRef` that is passed to the R function from a Postgres trigger function invocation and is given as an opaque reference. Such an object is stored as a numeric vector of length 3, and its contents are actually memory addresses that should not be used directly.

Details

These functions call C code that resolves the Postgres objects and perform the appropriate operations by querying the Postgres tables for a description of the tuple.

Value

`tupleValues` returns a named list containing the values contained in the tuple, having been converted to regular R objects (where possible). The names are the field/column names of the Postgres tuple.

`tupleTypes` returns a named character vector whose elements are the names of the Postgres data type for the corresponding element in the tuple. The names of the elements are again the field/column names of the Postgres tuple.

Author(s)

Duncan Temple Lang

References

<http://www.postgresql.org>, <http://www.omegahat.org/DBMS/Postgres/R>

See Also

[\[.PostgresTupleRef setTupleElements](#)

Examples

```
tupleValues(tup)
tupleTypes(tup)
```

```
[[<-.PostgresTupleRef
```

Assign a value to an element of a Postgres Tuple

Description

When an R function is called as a Postgres trigger function, it receives an argument which is a reference to the Postgres tuple that caused the trigger event. The R function can modify the contents of the tuple by assigning values to its elements.

Usage

```
[[<-.PostgresTupleRef(x, name, value)
```

Arguments

<code>x</code>	the tuple reference
<code>name</code>	the name of the attribute to set.
<code>value</code>	the value to assign to the tuple. An attempt is made to convert this to a Postgres object.

Details**Value****Author(s)**

Duncan Temple Lang

References

<http://www.omegahat.org/>

See Also**Examples**

[.PostgresTupleRef *Extracts one or more elements of a Postgres Tuple, by index or name.*

Description

This allows the caller to treat a Postgres tuple given to R as a result of a trigger function invocation as a regular R vector and to extract the value of one or more elements from it using names or indices.

Usage

```
[.PostgresTupleRef(tup, ...)
```

Arguments

<code>tup</code>	an object of class <code>PostgresTupleRef</code> which has been passed to R as an argument to a Postgres trigger function implemented in the embedded R procedural programming language in Postgres. Such an object should never be used outside of the function to which it was passed (or other functions called within that function's body).
<code>...</code>	either names or indices of elements within the Postgres tuple whose values are to be returned. Indices are 1-based (i.e. they start at 1 rather than 0)

Details

This calls C code to resolve the Postgres reference and extract the elements. It then queries the Postgres tables to determine the types of the objects and performs the conversions from those Postgres values to R objects.

Value

A named list containing the elements of the Postgres tuple converted to R objects. The names are the field/column names of the Postgres tuple.

Author(s)

Duncan Temple Lang

References

<http://www.postgresql.org>, <http://www.omegahat.org/DBMS/Postgres/R>

See Also

[tupleValues](#) [tupleTypes](#) [setTupleElements](#)

Examples

```
tup[1,2]
tup["joint1_width"]
```

`evalPostgresAggregatorInitialization`

Parses and evaluates a Postgres R-Aggregator initialization string.

Description

An `RAggregator` is a Postgres data-type whose functionality is implemented in R. Such an object is initialized by a user-specified Postgres string which is expected to create an R object (usually a closure) that can be used to incrementally compute some statistic by being passed each record within an SQL query one at a time. This function converts the Postgres initialization string into an R object which can be used as the aggregator.

Usage

```
evalPostgresAggregatorInitialization(txt)
```

Arguments

`txt` the string from Postgres that is to be parse and evaluated

Value

An object which can be used in the `plRaggregate` procedural language within Postgres. This typically is a list returned from a closure definition. By default, the first element of the list should be the function that is called for each record and which updates the internal state of the closure. It is called with a single argument - the value of the record. The second function is typically called with no arguments to retrieve the final result when all the records have been processed by Postgres and passed to the update function. The positions of these functions can be changed depending on how the aggregator is defined within Postgres. It is much simpler to user this ordering.

Author(s)

Duncan Temple Lang

References

<http://www.postgresql.org>, <http://www.omegahat.org/DBMS/Postgres/R>

Examples

```
evalPostgresAggregatorInitialization('jsumGenerator()')
evalPostgresAggregatorInitialization('(function() {
  minVal <- Inf
  update <- function(recordValue) {
    minVal <- min(minVal, recordValue)
  }
  getMinVal <- function() {
    minVal
  }

  return(list(update, getMinVal))
})()')
```

`names.PostgresTupleRef`

Retrieves the field/column names of a Postgres Tuple

Description

This retrieves the names of each of the fields within a Postgres tuple that is involved in a trigger event on a database table due to an INSERTION, DELETION or UPDATE action on that table. In conjunction with `tupleValues`, `tupleTypes` and `setTupleElements`, an R trigger function can access and modify the entries in the tuple.

Usage

```
names.PostgresTupleRef(ref)
```

Arguments

`ref` an object of class `PostgresTupleRef` that contains the address of a C-level object which stores the record for which a trigger event was generated.

Details

The tuple reference is passed to the R function as a result of a trigger event (INSERT, UPDATE or DELETE action on a table) within Postgres and the trigger function being implemented in the Postgres procedural language `plR`. Rather than copy the contents of the tuple being modified and passing them to R, we give a reference to the instance of that tuple so that the user can modify it directly before returning it to Postgres. This allows the R function to alter insertions that are pending.

Value

A character vector containing the names of the fields within the Postgres tuple reference by the argument.

Author(s)

Duncan Temple Lang

References

<http://www.postgresql.org>, <http://www.omegahat.org/DBMS/Postgres/R>

See Also

[tupleValues](#), [tupleTypes](#) [[.PostgresTupleRef](#)]

Examples

```
names(tup)
tup[names(tup)] # tupleValues(tup)
```

`parsePostgresFunction`

Parses an R Constructor from Postgres

Description

This function converts a Postgres string representation of an R function or object constructor and converts it into an evaluable expression which is then used internally (i.e. within R embedded in Postgres) to create the appropriate R object.

Usage

```
parsePostgresFunction(txt)
```

Arguments

`txt` the string passed by Postgres that contains the constructor expression for creating the R object for handling the Postgres function. This is registered when the Postgres function is defined via the `CREATE FUNCTION` statement.

Value

This returns an R expression that can be evaluated.

Author(s)

Duncan Temple Lang

References

<http://www.omegahat.org/>

See Also**Examples**

postgresError	<i>Raise a Postgres error from R</i>
---------------	--------------------------------------

Description

This function allows one to raise an error in Postgres from within an R function. These R functions are called by Postgres having been registered as procedural language functions.

Usage

```
postgresError(msg)
```

Arguments

msg	a string that describes the error. This should not contain any formatting (i.e. C <code>printf()</code> -style) entries.
-----	--

Value

The return value is unimportant as this function does not return. The Postgres engine will take over in error mode.

Author(s)

Duncan Temple Lang

References

<http://www.omegahat.org/>

Examples

`setTupleElements` *Sets one or more elements within a Postgres trigger record/tuple.*

Description

This allows the caller to modify the contents of an internal, C-level Postgres tuple or record which is being operated on in the context of a trigger event (INSERT, UPDATE or DELETE) on an table.

Usage

```
setTupleElements(x, ...)
```

Arguments

`x` an object of class `PostgresTupleReference` passed from Postgres as an element of a trigger event object.

`...` the new values that are to be inserted into the Postgres tuple. These can be named arguments where the names correspond to those of the tuple as returned by `tupleValues`, or they can be unnamed arguments in which case the *i*-th value is inserted into the *i*-th element of the tuple. Using named values is preferred, as usual, as it provides more understandable code that is independent of the ordering of the tuple elements.

Details

This converts each of the R values to the Postgres type of the element in the tuple into which the value is being inserted.

Value

A modified version of the `PostgresTupleReference` with a reference to the newly created internal `HeapTuple` object containing the new values. It is vital that this object not be discarded if subsequent operations involving the Postgres tuple are intended. This new value is usually returned as the value of the R trigger function, allowing Postgres to use the modified version of the tuple in the trigger event.

Note

Originally, this was written as a `[<-PostgresTupleRef`

Author(s)

Duncan Temple Lang

References

<http://www.postgresql.org>, <http://www.omegahat.org/DBMS/Postgres/R>

See Also

`tupleValues`, `tupleTypes`, `names.PostgresTupleRef`

Examples

```
setTupleElements(tup, "joint1_width" = 200)  
setTupleElements(tup, 200)
```

```
setTupleElements(tup, "joint1_width"=200, "joint2_width"= 150)  
setTupleElements(tup, "joint1_width"=200, 150)
```

```
setTupleElements(tup, "joint1_width"=200, 150)
```

```
setTupleElements(tup, 200, 150)
```

```
setTupleElements(tup, 200, 150)
```

Index

*Topic **Database**

- [.PostgresTupleRef, 3
- [[<-.PostgresTupleRef, 2
- EmbeddedPostgres, 1
- evalPostgresAggregatorInitial-
ization,
4
- names.PostgresTupleRef, 5
- parsePostgresFunction, 6
- postgresError, 7
- setTupleElements, 8
- tupleTypes, 1

*Topic **Embedded**

- [[<-.PostgresTupleRef, 2
- EmbeddedPostgres, 1
- parsePostgresFunction, 6
- postgresError, 7
- [.PostgresTupleRef, 2, 3, 6
- [[<-.PostgresTupleRef, 2

- EmbeddedPostgres, 1
- evalPostgresAggregatorInitialization,
4

- names.PostgresTupleRef, 5, 8

- parsePostgresFunction, 6
- postgresError, 7

- setTupleElements, 2, 4, 5, 8

- tupleTypes, 1, 4-6, 8
- tupleValues, 4-6, 8
- tupleValues (*tupleTypes*), 1