NAME

SPI_cursor_open_with_args - set up a cursor using a query and parameters

SYNOPSIS

Portal SPI_cursor_open_with_args(const char *_name_,
const char *_command_,
int _nargs_, Oid *_argtypes_,
Datum *_values_, const char *_nulls_,
bool _read_only_, int _cursorOptions_)

DESCRIPTION

SPI_cursor_open_with_args sets up a cursor (internally, a portal) that will execute the specified query. Most of the parameters have the same meanings as the corresponding parameters to **SPI_prepare_cursor** and **SPI_cursor_open**.

For one-time query execution, this function should be preferred over **SPI_prepare_cursor** followed by **SPI_cursor_open**. If the same command is to be executed with many different parameters, either method might be faster, depending on the cost of re-planning versus the benefit of custom plans.

The passed-in parameter data will be copied into the cursor's portal, so it can be freed while the cursor still exists.

This function is now deprecated in favor of **SPI_cursor_parse_open**, which provides equivalent functionality using a more modern API for handling query parameters.

ARGUMENTS

const char * _name_

name for portal, or NULL to let the system select a name

const char * _command_

command string

int _nargs_

number of input parameters ($1, $2, etc.)

Oid * _argtypes_

an array of length _nargs_, containing the OIDs of the data types of the parameters

Datum * _values_

an array of length _nargs_, containing the actual parameter values const char * *nulls*

an array of length *nargs*, describing which parameters are null

If *nulls* is NULL then **SPI_cursor_open_with_args** assumes that no parameters are null. Otherwise, each entry of the *nulls* array should be ' ' if the corresponding parameter value is non-null, or 'n' if the corresponding parameter value is null. (In the latter case, the actual value in the corresponding *values* entry doesn't matter.) Note that *nulls* is not a text string, just an array: it does not need a '\0' terminator.

bool *read_only*

true for read-only execution

int *cursorOptions*

integer bit mask of cursor options; zero produces default behavior

RETURN VALUE

Pointer to portal containing the cursor. Note there is no error return convention; any error will be reported via **elog**.