Transfer of handle to database connection (Databases and Database Tables)

Transfer of handle to database connection between the running ESL scripts

A transfer of handle to database connection between ESL scripts may be done by RPC procedures. In declaration of RPC procedure, you have to tag the parameter that represents the handle to database connection by enumerated type **DB_HANDLE**. An algorithm is contingent on the existence of database connection. If handle to database connection is an invalid value or it points to missing database connection, the algorithm ends with error.

The enumerated type **DB_HANDLE** is INT.

RPC procedure declaration:

```
 RPC PROCEDURE ProcName [([IN]  DB_HANDLE _db_handle[,_db_handle2, ...] [IN] DB_HANDLE _db_handle3]...)]

;actions

END ProcName
```

_db_handle that can be transferred between ESL scripts is created by these actions: DB_CONNECT, PG_DISCONNECT, SQL_CONNECT, DB_TRANS_OPEN.

Notes:

- The owner of data container can be only one ESL script, which also ensures the cancelling of this handle.
- Handle to database connection **can not** be transferred between ESL scripts of different processes.
- When calling RPC procedure, if you use the value that is not a handle to database connection on the place for a formal parameter DB_HANDLE, ESL script will search handle according to an input value:
    1. If database connection, tagged by input value, **exists**, the script **transfers** it.
    2. If database connection, tagged by input value, **does not** exist, ESL script **displays an error**.

```
INT _db_handle

_db_handle := 5

CALL [objIdent]  INSERT (_db_handle) ON procIdent
```

    3. If the input value, representing handle, is invalid, the calling Esl script ends with RunTime error.
- If the calling of RPC procedure is **asynchronous**, the database connection is terminated in this script. Then ESL script that has been called becomes the owner of this connection:

```
 ******************

 ; ESL script that is called
 RPC PROCEDURE InsertToDB(DB_HANDLE _handle)
 .....
 END InsertToDB

 **************


  ; ESL script that is calling
 INT _db_handle

 CALL[...] InsertToDB(_db_handle) ASYNC ON
 ....
 ; after this calling, handle to database connection will terminate in this script, the called ESL
script becomes the owner

 ****************
```

- If the calling RPC procedure is **synchronous**, there are two options:

1. If the formal parameter, which represents DB_HANDL, is tagged by the key word **IN**, when calling the RPC procedure, the ESL script containing the declaration of called RPC procedure will be **permanently** the owner of handle to database connection.

```
 ******************
 ; ESL script that is called
 RPC PROCEDURE InsertToDB(IN DB_HANDLE _handle)
 .....
 END InsertToDB
 ***************


 ; ESL script that is calling
 INT _db_handle

 CALL[...] InsertToDB(_db_handle) ON ....
 ; after this calling the database connection is terminated in this script, the called ESL script
becomes the owner
 ***************
```

2. If the formal parameter, which represents DB_HANDLE, **is not** tagged by the key word IN, when calling RPC procedure, ESL script containing the declaration of called RPC procedure will be the **temporal** owner of handle to database connection. After finishing the called RPC procedure, the script, from which the RPC procedure has been called, will become the owner.

```
 ******************
 ; ESL script that is called
 RPC PROCEDURE InsertToDB(DB_HANDLE _handle)
 .....
 END InsertToDB
 ***************


 ; ESL script that is calling
 INT _db_handle

 CALL[...] InsertToDB(_db_handle)  SYNC ON ....
 ; after this calling, ESL script, that is calling, is till the owner of database connection
 ***************
```

Related pages:

Script actions