

# Adding a new object linked to a parent object that already exists using CI\_OBJECTS and CI\_PARENTS

- Based on object's ID
- Based on object's GUID
  - Linked to a parent object that has an OBJECT\_GUID
  - Linked to a parent object that has no OBJECT\_GUID

When adding a new object to the CAST Analysis Service you MUST also define a link with a parent object. If you do not do so, an error ("The parent of the object is missing") will be generated. See for more information on error handling.

## Based on object's ID

In this example, we are going to add a new object (a T-SQL View called "TEST") and link it as a child to the existing database "CASTPUBS" that is already synchronized with the CAST Analysis Service. Remember that T-SQL objects are not automatically given OBJECT\_GUIDs, but this method will allow you to create your own OBJECT\_GUID for the new object and will automatically create the OBJECT\_GUID for the existing object (where it does not exist).

The first thing to do is to check that no other objects in the CAST Analysis Service are using what will be the new OBJECT\_GUID for the new object (please see [Creating OBJECT\\_GUIDs](#) for more information about creating OBJECT\_GUIDs for objects that do not have them):

```
select count(1) Cnt
from CTV_GUID_OBJECTS
where OBJECT_GUID = 'SQL_VIEW.WESLEY.CASTPUBS..test.DATABASE.WESLEY.CASTPUBS'
```

The results return 0 for this OBJECT\_GUID:

```
Cnt
[int]
----
0
```

The next step is to insert the data into the CAST entry tables that will create a new T-SQL object "TEST" and add a link to the existing parent database "CASTPUBS" when the tool is run. Add a new **Update CAST Knowledge Base Tool** and enter the following query:

```
insert into <KB_name>.dbo.CI_OBJECTS (OBJECT_GUID, OBJECT_TYPE, OBJECT_NAME, OBJECT_FULLNAME, ERROR_ID)
select 'SQL_VIEW.WESLEY.CASTPUBS..test.DATABASE.WESLEY.CASTPUBS', TYPE_NAME, 'TEST', 'WESLEY.CASTPUBS..TEST', 0
from <KB_name>.dbo.CTV_OBJECT_TYPES
where TYPE_DESCRIPTION = 'SQL View'
go
insert into <KB_name>.dbo.CI_PARENTS (OBJECT_GUID, PARENT_ID, ERROR_ID)
select 'SQL_VIEW.WESLEY.CASTPUBS..test.DATABASE.WESLEY.CASTPUBS', OBJECT_ID PARENT_ID, 0
from <KB_name>.dbo.CTV_GUID_OBJECTS
where OBJECT_NAME = 'CASTPUBS'
go
```

This is in effect two queries:

- The first creates the new object in the CI\_OBJECTS table
- The second inserts the data in the CI\_PARENTS table to create the link to the existing parent object. Please note the fact that this query fetches the parent object's OBJECT\_ID and gives it the PARENT\_ID alias (note that there is no comma between OBJECT\_ID and PARENT\_ID in the SELECT clause).



- The 0 parameter will enter 0 in the ERROR\_ID column of the tables in question.
- Note that you need to specify the Analysis Service database and user as above "<KB\_name>.dbo" otherwise the query will fail when the job is run.
- You can also use variables to replace the Analysis Service name if required. See the on-line Help for the CAST Management Studio for more information.

Then:

- Complete the configuration of the job and run it as outlined in [Using the Update CAST Knowledge Base Tool](#).

- Make sure you then update the CAST System Views.

If you want to check that the object and link have been created successfully, you can use the following query in an SQL IDE against your Analysis Service (adapting it for your environment):

```
select OBJECT_ID, OBJECT_NAME, OBJECT_GUID
from CTV_GUID_OBJECTS
where OBJECT_NAME = 'TEST'
```

The result shows the OBJECT\_GUID for the newly created object "TEST":

| OBJECT_ID | OBJECT_NAME | OBJECT_GUID   |
|-----------|-------------|---|
| [int]     | [char]      | [char]  |
| 1843      | TEST        | SQL_VIEW.WESLEY.CASTPUBS..test.DATABASE.WESLEY.CASTPUBS |

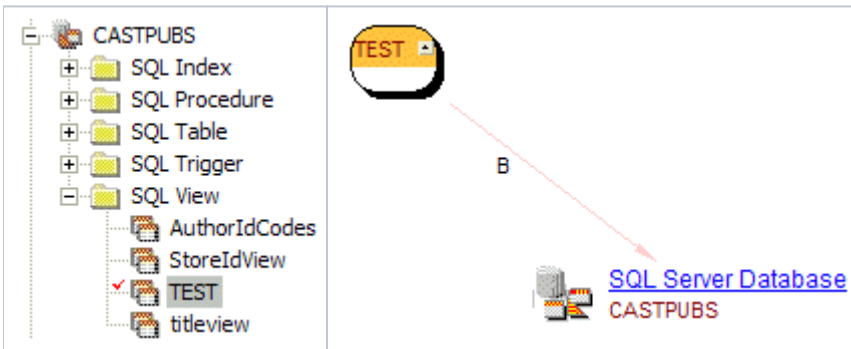
You can also check that the link to the parent object has also been created:

```
select OBJECT_GUID, PARENT_GUID
from CTV_GUID_PARENTS
where OBJECT_NAME = 'TEST'
```

The result shows the OBJECT\_GUID and PARENT GUID for the newly created object "TEST" and the link:

| OBJECT_GUID   | PARENT_GUID              |
|---|--------------------------|
| [char]  | [char]                   |
| SQL_VIEW.WESLEY.CASTPUBS..test.DATABASE.WESLEY.CASTPUBS | DATABASE.WESLEY.CASTPUBS |

Finally, you can check in CAST Enlighten - F5 to refresh the view - that everything is as it should be:



## Based on object's GUID

### Linked to a parent object that has an OBJECT\_GUID

In this example, we are going to add a new object (a Function called "TEST ()" and link it as a child to the existing C++ object (a file called "EXAMPLE.C") in the CAST Analysis Service. The first thing to do is to check that no other objects in the CAST Analysis Service are using what will be the new OBJECT\_GUID of this new object (please see chapter [Creating OBJECT\\_GUIDs](#) for more information about creating OBJECT\_GUIDs for objects that do not have them):

```
select count(1) Cnt
from CTV_GUID_OBJECTS
where OBJECT_GUID = 'C_Fct.TEST.C_Fi.'D:\TESTAPPLICATIONS\SQL SERVER\DEMO_C\EXAMPLE..C'
```

The result returns 0 for this OBJECT\_GUID:

```
Cnt
[int]
----
0
```

The next step is to identify the OBJECT\_GUID of the existing parent object "EXAMPLE.C":

```
select OBJECT_GUID
from CTV_GUID_OBJECTS
where OBJECT_NAME = 'EXAMPLE.C'
```

The result returns the following OBJECT\_GUID for "EXAMPLE.C":

```
OBJECT_GUID
[char]
-----
C_Fi."D:\TESTAPPLICATIONS\SQL SERVER\DEMO_C\EXAMPLE..C"
```

The next step is to insert the data into the CAST entry tables that will create a new object "TEST" and add a link to the existing parent object "EXAMPLE.C" when the tool is run. Add a new **Update CAST Knowledge Base Tool** and enter the following query:

```
insert into <KB_name>.dbo.CI_OBJECTS (OBJECT_GUID, OBJECT_TYPE, OBJECT_NAME, OBJECT_FULLNAME, ERROR_ID)
select 'C_Fct.TEST.C_Fi."D:\TESTAPPLICATIONS\SQL SERVER\DEMO_C\EXAMPLE..C"', TYPE_NAME, 'TEST', 'TEST', 0
from <KB_name>.dbo.CTV_OBJECT_TYPES
where TYPE_DESCRIPTION = 'C/C++ Function'
go
insert into <KB_name>.dbo.CI_PARENTS (OBJECT_GUID, PARENT_GUID, ERROR_ID)
values ('C_Fct.TEST.C_Fi."D:\TESTAPPLICATIONS\SQL SERVER\DEMO_C\EXAMPLE..C"', 'C_Fi."D:\TESTAPPLICATIONS\SQL
SERVER\DEMO_C\EXAMPLE..C"', 0)
go
```

This is in effect two queries:

- The first creates the new object in the CI\_OBJECTS table
- The second inserts the data in the CI\_PARENTS table to create the link to the existing parent object.



- The 0 parameter will enter 0 in the ERROR\_ID column of the tables in question.
- Note that you need to specify the Analysis Service database and user as above "<KB\_name>.dbo" otherwise the query will fail when the job is run.
- You can also use variables to replace the Analysis Service name if required. See the on-line Help for the CAST Management Studio for more information.

Then:

- Complete the configuration of the job and run it as outlined in [Using the Update CAST Knowledge Base Tool](#).
- Make sure you then update the CAST System Views.

If you want to check that the object and link have been created successfully, you can use the following query in an SQL IDE against your Analysis Service (adapting it for your environment):

```
select OBJECT_ID, OBJECT_NAME, OBJECT_GUID
from CTV_GUID_OBJECTS
where OBJECT_NAME = 'TEST'
```

The result shows the OBJECT\_GUID for the newly created object "TEST":

```
OBJECT_ID OBJECT_NAME OBJECT_GUID
[int]      [char]      [char]
-----
5563      TEST        C_Fct.TEST.C_Fi."D:\TESTAPPLICATIONS\SQL SERVER\DEMO_C\EXAMPLE..C"
```

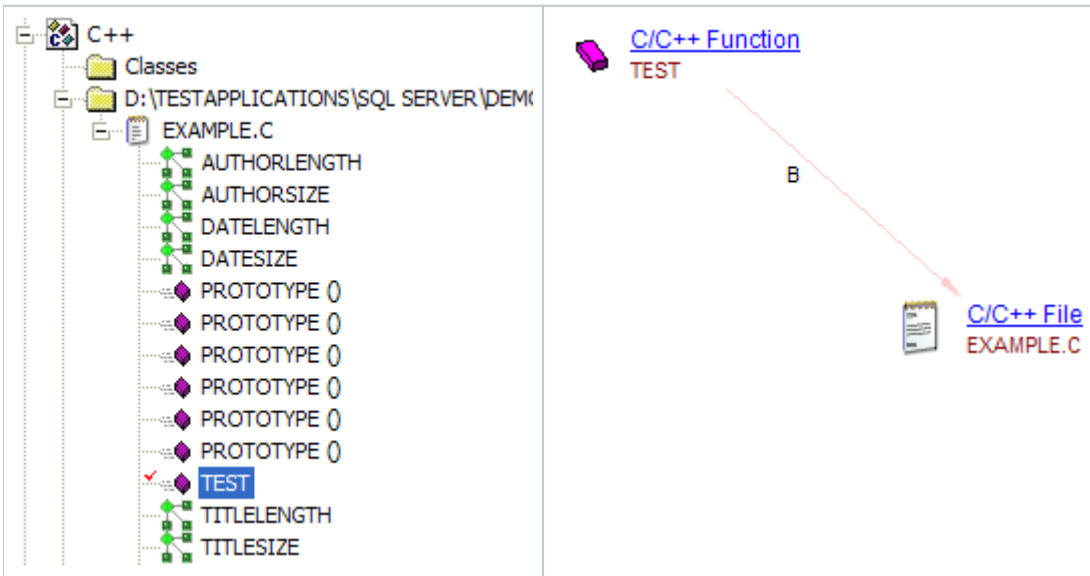
You can also check that the link to the parent object has also been created:

```
select OBJECT_GUID, PARENT_GUID
from CTV_GUID_PARENTS
where OBJECT_NAME = 'TEST'
```

The result shows the OBJECT\_GUID and PARENT GUID for the newly created object "TEST" and the link:

```
OBJECT_GUID
[ char]
C_Fct.TEST.C_Fi."D:\TESTAPPLICATIONS\SQL_SERVER\DEMO_C\EXAMPLE..C"
PARENT_GUID
[ char]
C_Fi."D:\TESTAPPLICATIONS\SQL_SERVER\DEMO_C\EXAMPLE..C"
```

Finally, you can check in CAST Enlighten - F5 to refresh the view - that everything is as it should be:



## Linked to a parent object that has no OBJECT\_GUID

In this example, we are going to add a new T-SQL server object (a stored procedure called "TEST\_PROC") to the CAST Analysis Service. This will be added to the parent database "CASTPubs".

The first thing to do is to check that there are no other objects in the Analysis Service called "TEST\_PROC". You can do so in CAST Enlighten, or you can use the following query in an SQL IDE against your Analysis Service (adapting it for your environment) to check that no other objects in the CAST Analysis Service are using what will be the new OBJECT\_GUID of this new object (please see [Creating OBJECT\\_GUIDs](#) for more information about creating OBJECT\_GUIDs for objects that do not have them):

```
select count(1) Cnt
from CTV_GUID_OBJECTS
where OBJECT_GUID = 'CASTPubs.PROC.TEST_PROC'
```

The result returns nothing:

```
Cnt
[ int]
----
0
```

The next step is to check what object is the parent of a stored procedure - in this case, we know it is the database "CASTPubs". You can check in CAST Enlighten.

Next you need to check whether the parent object already has a GUID:

```
select OBJECT_ID, OBJECT_NAME, OBJECT_GUID
from CTV_GUID_OBJECTS
where OBJECT_NAME = 'CASTPubs'
```

The result shows us that it does not, so one will need to be created:

```
OBJECT_ID OBJECT_NAME OBJECT_GUID
[int]      [char]      [char]
4445      CASTPubs      NULL
```

The next step is to insert the data into the CAST entry tables that will create a new object "TEST\_PROC", add a link to the existing parent object "CASTPubs" and create an OBJECT\_GUID for "CASTPubs" when the tool is run. Add a new **Update CAST Knowledge Base Tool** and enter the following query:

```
insert into <KB_name>.dbo.GUID_OBJECTS (OBJECT_ID, OBJECT_GUID, ERROR_ID)
select OBJECT_ID, OBJECT_NAME, 0
from <KB_name>.dbo.CTV_GUID_OBJECTS
where OBJECT_NAME = 'CASTPubs'
go
insert into <KB_name>.dbo.CI_OBJECTS (OBJECT_GUID, OBJECT_TYPE, OBJECT_NAME, OBJECT_FULLNAME, ERROR_ID)
select 'CASTPubs.PROC.TEST_PROC', TYPE_NAME, 'TEST_PROC', 'TEST_PROC', 0
from <KB_name>.dbo.CTV_OBJECT_TYPES
where TYPE_DESCRIPTION = 'SQL Procedure'
go
insert into <KB_name>.dbo.CI_PARENTS (OBJECT_GUID, PARENT_GUID, ERROR_ID)
values ('CASTPubs.PROC.TEST_PROC', 'CASTPubs', 0)
go
```

This is in effect three queries:

- The first creates the OBJECT\_GUID for the parent object "CASTPubs"
- The second creates the new object in the CI\_OBJECTS table
- The second inserts the data in the CI\_PARENTS table to create the link to the existing parent object "CASTPubs".



- The 0 parameter will enter 0 in the ERROR\_ID column of the tables in question.
- Note that you need to specify the Analysis Service database and user as above "<KB\_name>.dbo" otherwise the query will fail when the job is run.
- You can also use variables to replace the Analysis Service name if required. See the on-line Help for the CAST Management Studio for more information.
- Please see [Creating OBJECT\\_GUIDs](#) for more information about creating OBJECT\_GUIDs for objects that do not have them.

Then:

- Complete the configuration of the job and run it as outlined in [Using the Update CAST Knowledge Base Tool](#).
- Make sure you then update the CAST System Views.

If you want to check that the object and link have been created successfully, you can use the following query in an SQL IDE against your Analysis Service (adapting it for your environment):

```
select OBJECT_ID, OBJECT_NAME, OBJECT_GUID
from CTV_GUID_OBJECTS
where OBJECT_NAME = 'TEST_PROC'
```

The result shows the OBJECT\_GUID for the newly created object "TEST":

```
OBJECT_ID OBJECT_NAME OBJECT_GUID
[int]      [char]      [char]
-----
5566      TEST_PROC      CASTPubs.PROC.TEST_PROC
```

You can also check that the link to the parent object has also been created:

```
select OBJECT_GUID, PARENT_GUID
from CTV_GUID_PARENTS
where OBJECT_NAME = 'TEST_PROC'
```

The result shows the OBJECT\_GUID and PARENT GUID for the newly created object "TEST" and the link:

```
OBJECT_GUID
[ char]
CASTPubs.PROC.TEST_PROC
PARENT_GUID
[ char]
CASTPubs
```

Finally, you can check in CAST Enlighten - F5 to refresh the view - that everything is as it should be:

