

Creating a GoldenGate Exception Handler to Trap and Log Oracle Errors

GoldenGate does not provide a standard exceptions handler. By default, a Replicat process will abend should any operational failure occur, and will rollback the transaction to the last known checkpoint. This may not be ideal in a production environment.

The `HANDLECOLLISIONS` and `NOHANDLECOLLISIONS` parameters can be used to control whether or not Replicat tries to resolve duplicate-record and missing-record errors, but should these errors be ignored?

The way to determine what error has occurred, by which Replicat, caused by what data, create an Exceptions handler.

Steps

The steps below create an Exceptions handler that will trap and log the specified Oracle error(s), but allow the Replicat to continue to process data:

1. The first step is to create an Exceptions table, similar to the example DDL below:

```
create table ggs_admin.exceptions
( rep_name varchar2(8)
, table_name varchar2(61)
, errno number
, dberrmsg varchar2(4000)
, optype varchar2(20)
, errtype varchar2(20)
, logrba number
, logposition number
, committimestamp timestamp
);
```

```
ALTER TABLE ggs_admin.exceptions ADD (
  CONSTRAINT PK_CTS
  PRIMARY KEY
  (logrba, logposition, committimestamp) USING INDEX PCTFREE 0
  TABLESPACE MY_INDEXES);
```

The Exceptions table must be created in the GoldenGate Admin user schema. It can log exception data for all Replicat processes.

2. Edit each Replicat process parameter file and add the exception handler Macro code block.

```
[oracle@linuxserver1 ggs]$ ggsci
```

```
GGSCI (linuxserver1) 1> edit params RTARGET1
```

```
-- This starts the macro
MACRO #exception_handler
BEGIN
, TARGET ggs_admin.exceptions
, COLMAP ( rep_name = "RTARGET1"
, table_name = @GETENV ("GGHEADER", "TABLERNAME")
, errno = @GETENV ("LASTERR", "DBERRNUM")
, dberrmsg = @GETENV ("LASTERR", "DBERRMSG")
, optype = @GETENV ("LASTERR", "OPTYPE")
, errtype = @GETENV ("LASTERR", "ERRTYPE")
, logrba = @GETENV ("GGHEADER", "LOGRBA")
, logposition = @GETENV ("GGHEADER", "LOGPOSITION")
, committimestamp = @GETENV ("GGHEADER", "COMMITTIMESTAMP"))
, INSERTALLRECORDS
, EXCEPTIONSONLY;
END;
-- This ends the macro
```

3. Remaining within the editor (vi), edit the MAP statements to include the #exception_handler(). Also add the REPEROR parameter to reference to the Oracle error(s) you wish to trap.

```
REPEROR (DEFAULT, EXCEPTION)
REPEROR (DEFAULT2, ABEND)
REPEROR (-1, EXCEPTION)
MAP SRC.ORDERS, TARGET TGT.ORDERS;
MAP SRC.ORDERS #exception_handler()
MAP SRC.ORDER_ITEMS, TARGET TGT.ORDER_ITEMS;
MAP SRC.ORDER_ITEMS #exception_handler()
MAP SRC.PRODUCTS, TARGET TGT.PRODUCTS;
MAP SRC.PRODUCTS #exception_handler()
```

- The REPEROR parameter controls how the Replicat process responds to errors when executing the MAP statement.
- The DEFAULT argument sets a global response to all errors except those for which explicit REPEROR statements are specified.

E.g. A MAP statement to trap ORA-01403: "no data found" error.

```
MAP SRC.ORDERS, TARGET TGT.ORDERS, REPEROR (-1403,
EXCEPTION);
```

- The DEFAULT2 argument specifies a "catch all" action for any unanticipated Oracle errors that may occur. In the example in step 3, the Replicat process will Abend.

4. Stop and start the Replicat process.

```
GGSCI (linuxserver1) 3> stop REPLICAT RTARGET1
```

```
Sending STOP request to REPLICAT RTARGET1 ...
```

Request processed.

```
GGSCI (linuxserver1) 4> start replicat RTARGET1
```

```
Sending START request to MANAGER ...  
REPLICAT RTARGET1 starting
```

5. Check Replicat process is running.

```
GGSCI (linuxserver1) 5> info all
```

Program Chkpt	Status	Group	Lag	Time Since
MANAGER	RUNNING			
REPLICAT	RUNNING	RTARGET1	00:00:00	00:00:22

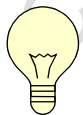
6. Start your application and begin replicating data.

Viewing Exceptions

Below is an example of the data collected following an ORA-00001: "unique constraint violated"

```
SQL> select * from ggs_admin.exceptions where rownum <= 1;
```

```
REP_NAME TABLE_NAME ERRNO DBERRMSG  
-----  
RTARGET1 SRC.ORDERS      1 OCI Error ORA-00001: unique  
constraint (TGT.PK_ORD) violated (status = 1), SQL  
          <INSERT INTO "TGT"."ORDERS"  
("ORDER_ID", "CUST_ID", "PRODUCT_ID" ..  
  
OPTYPE ERRTYPE LOGRBA LOGPOSITION COMMITTIMESTAMP  
-----  
INSERT DB          988  171211460 02-APR-10 12.41.42.999468
```



Tip: The DBERRMSG column will store the error, the error description and the complete SQL up to 4000 bytes.

The exception handler can be modified to also include the before and after images of an UPDATE operation. This information is valuable for conflict resolution.