Splitting Integrated Extract and Replicat

Edward Whalen
2019-04-10

During a recent meeting with Jinyu Wang of Oracle I learned that running and Integrated Extract and Integrated Replicat from a remote system can offload much of the work from your production database systems. See Jinyu's blog at http://jinyuwang.weebly.com/oracle-goldengate. Using this method is a little counter intuitive for those of us who have been using GoldenGate for a while but it makes sense.

Much of the overhead incurred by GoldenGate (CPU, Memory, IO) is in handling the trail file. In fact, I have spoken many times about how much memory can be consumed by GoldenGate in long-running updates. By separating the Integrated Extract and Integrated Replicat into two components; database side and GoldenGate side, much of this overhead is moved off of the database server.

Setting this up is easy to do. Let's go through the steps:

1. Install the Oracle Client on the Intermediate System(s)
2. Setup tnsnames.ora on the Intermediate System
3. Install GoldenGate on the Intermediate System
4. Create the Integrated Extract
5. Create Pump or Integrated Replicat

Note: The Source and Intermediate system must reside within the same subnet (No Firewall In-between).

Note: The Intermediate system and Target must reside within the same subnet (No Firewall In-between).

The Configuration

The configuration for this example uses two servers, each with a 12.1.0.2 container database with 3 pluggable databases. A summary of the source and target is shown here:

	Source	Target
Host:	gg21a	gg21b
Database:	gg21adb	gg21cdb
PDB1:	pdb21a1	pdb21c1
PDB2:	pdb21a2	pdb21c2
PDB3:	pdb21a3	pdb21c3

The configuration of the two systems is comparable. The test schema for each PDB is test. The test user is not a common user, but the same name is used in all PDBs.

Install and Configure the Oracle Client

Since we are not actually creating a database on the Intermediate system all that is required is the Oracle Client. If the source and target systems are in the same datacenter and in the same subnet perhaps for an upgrade or migration I don't even need to create a pump process. I can use the same trail file generated by the extract for the replicat.

The tnsnames.ora file should include the root container and all of the PDBs for both source and target. This is critical for both the extract and the replicat.

Configuring the Environment

The GoldenGate administrator should be setup as a common user with DBA access to all PDBs. I'm not showing it here because I've done this in previous blog posts. Configuring GoldenGate 12c in an Oracle Multitenant Environment part 1 and part 2. However, it is important that the credential store use an alias that includes the connection string to the remote server as shown here.

```
GGSCI (gg21b) 8> info credentialstore

Reading from ./dircrd/:

Default domain: OracleGoldenGate

  Alias: c##ggadm21a
  Userid: c##ggadmin@gg21adb

  Alias: c##ggadm1
  Userid: c##ggadmin@pdb21c1

  Alias: c##ggadm2
  Userid: c##ggadmin@pdb21c2

  Alias: c##ggadm3

  Userid: c##ggadmin@pdb21c3
```

Notice that only ggadm21a points to a root container, whereas 21c points to PDBs.

Configuring the Extract

The Extract configuration is no different than a normal except that the alias is to a user that actually points to the source database system such as ggadmroot is an alias to ggadmin@rootdatabase or in the case of my example ggadmin@gg21adb. The extract parameter file is for a container database of which I am extracting from three PDBs.

A parameter file would look like this:

```
EXTRACT ETEST

SETENV (NLS_LANG = AMERICAN_AMERICA.AL32UTF8)
SETENV (ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/client_1)

USERIDALIAS c##ggadm21a
```

```
-- IGNORETRUNCATES

ReportCount Every 30 Minutes, Rate
Report at 01:00
ReportRollover at 01:15 on SUNDAY

TRANLOGOPTIONS USE_ROOT_CONTAINER_TIMEZONE
TRANLOGOPTIONS EXCLUDETAG 01
TRANLOGOPTIONS EXCLUDETAG 02
TRANLOGOPTIONS EXCLUDETAG 03

LOGALLSUPCOLS

EXTTRAIL /u02/ggtrails/test/lc

-- Tables to pull from
TABLE pdb21a1.test.* ;
TABLE pdb21a2.test.* ;
TABLE pdb21a3.test.* ;
```

It is important that in this case, the alias c##ggadmin21a references ggadmin@gg21adb where gg21adb is the tnsnames entry that points to the root container on the source database server.

Configuring the Replicat

The Replicat configuration is also no different from a standard replicat except that it is pointing to an individual PDB. Remember, the replicat must point to an individual PDB, but you should still use the common user c##ggadmin or whatever userid you choose to use. I prefer to name my aliases something like ggadm21c1, ggadmin21c2, ggadmin21c3 for pdb21c1, pdb21c2, pdb21c3 respectively.

A parameter file would look like this:

```
REPLICAT RTEST1

-- Login to database
USERIDALIAS c##ggadm21c1

REPERROR (DEFAULT, ABEND)
REPERROR (-1, IGNORE)
DISCARDFILE ./dirdsc/rtest.dsc, append
DISCARDROLLOVER AT 05:30 ON Wednesday

DBOPTIONS INTEGRATEDPARAMS(max_sga_size 200, parallelism 1,
COMMIT_SERIALIZATION FULL)
DBOPTIONS SETTAG 01

ReportCount Every 30 Minutes, Rate
Report at 01:00
ReportRollover at 01:15 on SUNDAY

-- MAP STATEMENTS

--SOURCECATALOG pdb21a1
```

```
      MAP pdb21a1.test.*, TARGET test.* ;
```

It is important that in this case, the alias c##ggadmin21c1 references ggadmin@pdb21c1 where pdb21c1 is the tnsnames entry that points to PDB1 on the target database server.

Advantages of the Split Extract and Replicat

There are several advantages to splitting the Intgrated Extract and Replicat into two components on different servers. First, the task of managing the trail file is done external to the database server itself, thus oflooding much of the work. Second, I have blogged in the past about the effect of large updates on the Extract process. Large updates can be large memory consumers since the entire update much be assembled in memoy by the Extract during the entire process until the commit record is received. This can consume a large amount of memory which can be offloaded to another server. Third, all of the I/O generated by writing to and reading from the trail file is offloaded to a dedicated server. Finally, forth, you can run this entire architecture by creating only one trail file rather than the traditional model of extract-pump-replicat. However, this is only true of a completely local configuration. If source and target are separated by any distance or separated by a firewall, use a pump.

GoldenGate Cloud Services

Since GoldenGate Cloud Services (GGCS) is designed to work with Oracle Cloud Database as a Service DBaaS it does exactly what has been described in this blog. As someone who has been using GoldenGate for a while the idea of spitting the Integrated Extract and Integrated Replicat into two servers originally seemed counterintuitive.

Since I am a human being like all of you and used to doing things the way that I always do I was originally resistant to this new way of doing business but the more I started thinking about it the more it seems to make sense. Change can be scary at first but then you get used to it it can turn out to bhe the future.

In future blogs I will cover GoldenGate Cloud Services in more detail and relay my experiences.