Configuring Heterogeneous Replication server for MSSQL

By Garrett Devine
Garrett.devine@dbxperts.co.uk

Contents

Recommendations First

Sybase recommends that ECDA for ODBC, and the target database reside on the same machine.
MSSQL database should be set to capability mode 2005(90) - I am not sure that this is true.
Early testing has shown that we can replicate to a MSSQL server DB set to 2008(100).

System requirements

Repserver, 512MB RAM, 380MB disk
ECDA, 512MB RAM, 300BM disk

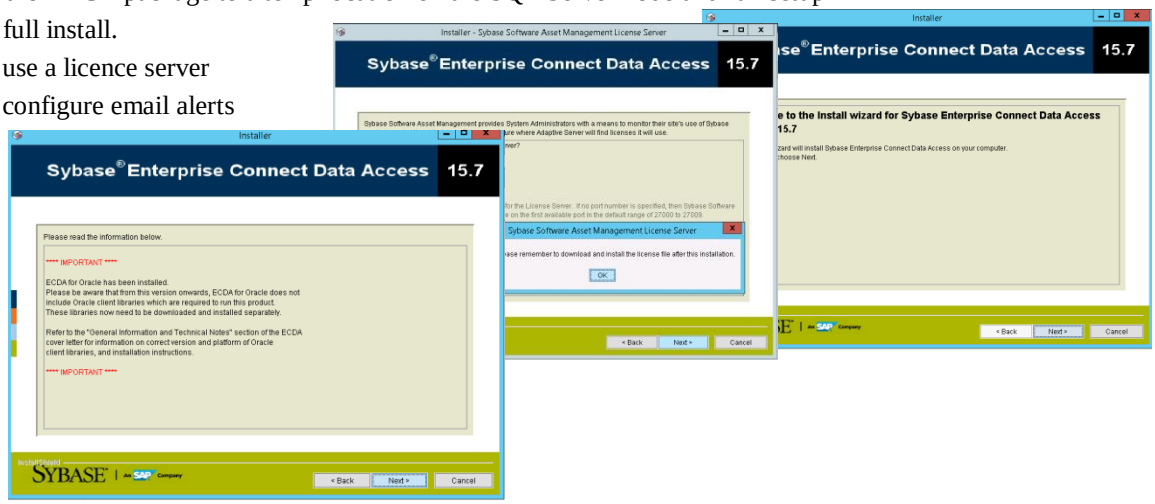
Configuring ECDA Option for ODBC

1. Open ODBC Driver 32-bit manager from %systemroot%\SysWOW64\odbcad32.exe (this is the 32-bit version). You must use the 32-bit version.
2. Add System DNS connection for SQL Server driver called ***ECDA_TRAIN1***
3. Server name is VM_TRAIN\MSSQL_TRAIN1
4. Add sql server login to SQL Server called ecda_user, with sysadmin role (for now)
5. Use "SQL Server authentication", userID=ecda_user, password=ecda_user. (sysadmin role, password does not expire)

*note: in some clients systems, we have found that we had to setup the ODBC connection using a Domain User account and use a User DNS.

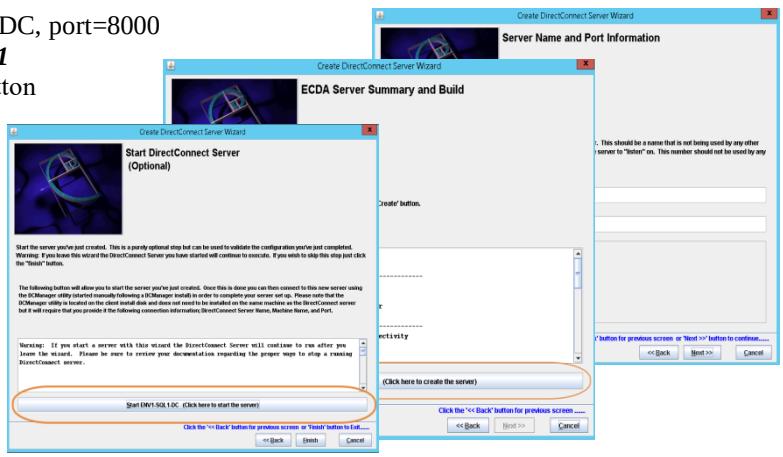
Installing ECDA

1. Extract the EDCA package to a temp location on the SQL Server node and run setup
2. choose full install.
3. Do not use a licence server
4. Do not configure email alerts
5.



Configuring and starting the DirectConnect server

1. Start %SYBASE%\DC-15_0\DCWizard/ DCWizard.bat
2. Select the ECDA Option for ODBC
3. Set server name= MSSQL_TRAIN1_DC, port=8000
4. Enter service name = ***ECDA_TRAIN1***
5. Finally, select the „Create Server" button
6. Start the service



7. Edit C:\sybase\DC-15_0\servers\MSSQL_TRAIN1_DC\cfg\ dcany.cfg to contain the following:-

```
[Service Library]
{Client Interaction}

[ECDA_TRAIN1]
{ACS Required}
ConnectionSpec1=ECDA_TRAIN1
{Client Interaction}
EnableAtStartup=yes
TransactionMode=long
SendWarningMessages=yes
{Target Interaction}
Allocate=connect
SQLTransformation=passthrough
ReturnNativeError=yes
{Catalog Stored Procedures}
CSPColumnODBCVersion=3
```

Note1: the section name and ConnectionSpec1 settings are set to the name of our service. This name **must** match the name given to the ODBC connection.

Adding licence file

SAP provide a pdf of the licence file. Copy and paste the contents of the file into
C:\Sybase\SYSAM-2_0\licenses\SySAMLicenseServer.lic

Stop DC server (close MS-DOS box)

Start DC server by opening new MD-DOS box and running

```
C:\sybase\DC-15_0\bin>DCStart.bat -SMSSQL_TRAIN1_DC
```

Testing

1. Start DirectConnect service, `C:\sybase\DC-15_0\bin\DCStart.bat -SMSSQL_TRAIN1_DC`
2. Connect using „isql“ from a MS-DOS session

```
C:\sybase\DC-15_0>DC_SYBASE.bat
C:\sybase\DC-15_0>isql -Usa -Pecda_user -SECDA_TRAIN1
```

Verify the connection to the replicate Microsoft SQL Server database by obtaining the DBMS name and version number:

```
select
@@sqldbmsname go
SQLDbmsName
--------------------
Microsoft SQL Server
```

Well done. You have configured the DirectConnect server!

Get replicating !

Assuming your Sybase repserver is already installed.

We will add a database, and then define a simple replication definition (repdef), which we will subscribe to in MSSQL Server.

Creating the primary database

ASE

```
disk init name="pubs2_data01",physname="C:\sybase_15\pubs2_data01.dat",size="20M"
go
sp_diskdefault pubs2_data01, defaulton
go
sp_diskdefault master, defaultoff
go

cd $SYBASE/ASE-15_0/scripts ISQL -iinstpbs2
```

MSSQL

```
Create small database using wizard, called pubs2
```

Create a replication maintenance user in Microsoft SQL Server

1. In the Sybase serer (primary)

```
sp_addlogin rep_maint, rep_maint_ps,pubs2
go
sp_role 'grant', replication_role,rep_maint
go
use
pubs2
go
sp_addalias rep_maint,dbo
go
```

2. In MSSQL server (replicate)

```
CREATE LOGIN rep_maint WITH PASSWORD='rep_maint_ps',
DEFAULT_DATABASE=pubs2, CHECK_EXPIRATION=OFF,
CHECK_POLICY=OFF
go
Use pubs2
Go
CREATE USER rep_maint FOR LOGIN rep_maint;
EXEC sp_addrolemember 'db_owner', 'rep_maint';
```

Add databases to Replication System

Primary DB

Use *rs_init* to add primary database. This will configure the rep_agent and rs_* objects to the ASE database and add the connection to the repserver.

1. Start *rs_init* and select to "Add a database" from the replication configuration dialogue
2. Give a server name of SYB_TRAIN1_RS
3. In the Database Information dialogue box fill in all the details and make sure you answer "yes" to "Will the database be replicated". Accept the suggested user** name and password for the maintenance user.
4. Finally, once you are happy that all the information, select "Continue" from the "Add database to replication system" dialogue box.

**Notes: you may need to drop the rep_maint alias, add it as a normal user using sp_adduser, run the above script and finally add the rep_maint user back in as an alias to dbo.

Replicate DB

Add entry to the sql.ini file in the Sybase Replication Server, so the repserver knows how to connect to the replicate database on the SQL Server box. Example below (change IP address).

```
[ECDA_TRAIN1]
master=TCP,10.1.5.255,8000
query=TCP,10.1.5.255,8000
```

With the DC still running, log into the RS server

```
create connection to ECDA_TRAIN1.pubs2
using profile rs_ase_to_msss;standard
set username rep_maint
set password "rep_maint_ps"
go
admin who
go
```

You should see similar entries to this:-

```
    28 DSI EXEC    Awaiting Command     105(1) ECDA_TRAIN1.pubs2
    27 DSI         Awaiting Message     105 ECDA_TRAIN1.pubs2
    26 SQM         Awaiting Message     105:0 ECDA_TRAIN1.pubs2
```

If your connections show as DOWN, see trouble shooting section at the end of this document.

Create the Replication Definition

```
In ASE,

use pubs2
go
-- get the repdef from the table
-- stored proc can be found here
http://www.sypron.nl/misctools.html
1>sp__gen_repdef "create", publishers ,
SYB_TRAIN1, pubs2
2>go

create replication definition publishers_repdef
with primary at SYB_TRAIN1.pubs2
with all tables named 'publishers'
(
  pub_id char(4),
  pub_name varchar(40),
  city varchar(20),
  state char(2)
)
primary key (pub_id)
replicate minimal columns
```

Run the above output into the RS!

Synchronise the table data between ASE & MSSQL

In ASE,

```
-- First, reverse engineer the table, using Ed Barlows excellent stored
proc, sp__revtable or use the Sybase ddlgen tool.
1> sp__revtable publishers
2> go
 -- Table_DDL
 ------------------------------------------------------------------------
 CREATE TABLE  publishers
 (
  pub_id    char(4)                     NOT NULL,
  pub_name  varchar(40)                 NULL,
  city      varchar(20)                 NULL,
  state     char(2)                     NULL
 )
 -- Index_DDL
 ------------------------------------------------------------------------
create unique clustered index pubind on dbo.publishers (pub_id)
```

Run the above output into MSSQL, pubs2 database, to create the empty table.

Use BCP to copy data:-

```
bcp pubs2..publishers out publishers.txt -SSYB_TRAIN1 -Usa -Pecda_user -c

msbcp pubs2..publishers in publishers.txt -Usa -Pecda_user -SVM-TRAIN\MSSQL_TRAIN1 -c
```

*note, I have changed the name of the Microsoft bcp.exe to msbcp.exe, to differentiate between the two versions of this utility. You only need to do this if both databases servers are on the same host (our test environment).

Create Subscription to our new repdef

In RS,

```
create subscription publishers_sub
for publishers_repdef
with replicate at ECDA_TRAIN1.pubs2
without materialization

check subscription publishers_sub
for publishers_repdef
with replicate at ECDA_TRAIN1.pubs2
```

Mark the table for replication

This step is vital, yet very simple. I have lost track of the number of times I have forgotten to do this!
In ASE,

```
sp_setreptable publishers, true
```

Insert sample row

```
insert into publishers values('9999', 'Fancypants publishing', 'London', 'NA')
```

Now in ASE & MSSQL compare the returned rows ‘select * from publishers’ in Sybase and SQL Server

If they are the same, congratulations you have installed replication between Sybase and MSSQL, using a simple repdef-subscription model. If the results are not the same, go back and check each step, one at a time.

Supplementary Note I - How to automatically start DirectConnect server at boot time?

Installing a DirectConnect server as a Windows service
DirectConnect no longer automatically creates the server as a Windows service. However, you can run a DirectConnect server as a Windows service. The following describes how to register, configure, start, stop, and remove DirectConnect as a Windows service.

http://infocenter-archive.sybase.com/help/index.jsp?topic=/com.sybase.dc35394_0110/html/di_inst_win/di_inst_win41.htm

Example:

```
ServiceWrapper.exe --install SYBDC_MSSQL_TRAIN1_DC --username=<username> --password=<password>
C:\sybase\DC-15_0\bin\DCStart.bat -S MSSQL_TRAIN1_DC
```

Make sure account as 'start as a service' permission

Supplementary Note II – adding additional 'Services'

You can have multiple database connections against a single Direct Connect server. These are called (rather confusingly, Services). The following is and example of how to add a new "Service" to an existing DC server.

1. Create new ODBC connection to target server (MSSQL) and specify the name of the target database.
2. Next, to help the DC know how to connect to the new service, add the following to the SQL.INI file. Note that the Port number is the same for the DC and all services that run under it

```
[EDCA_TRAIN2]
query=NLWNSCK, vm-train, 8000
win3_query=NLWNSCK, vm-train, 8000
```

3. Finally, configure the new "Service" by adding the following to the C:\Sybase\DC-15_0\servers\<server_name>\cfg\dcany.cfg file.

```
[ECDA_TRAIN2]
{ACS Required}
ConnectionSpec1=ECDA_TRAIN2
{Client Interaction}
EnableAtStartup=yes
TransactionMode=long
SendWarningMessages=yes
{Target Interaction}
Allocate=connect
SQLTransformation=passthrough
ReturnNativeError=yes
{Catalog Stored Procedures}
CSPColumnODBCVersion=3
```

4. Restart the DirectConnect server

Supplementary Note III – recommended config values from Sybase engineering

Below are the appropriate DC/ECDA config settings for a replication environment. Please compare with your current settings and update accordingly.

Allocate
Controls when an access service allocates conversations with the target database system. Syntax Allocate=[connect | request]
Default connect

Values

connect specifies an access service to allocate the conversation when the client connects, and holds it open for the duration of the client connection.

request specifies an access service to allocate a new conversation each time the client application sends a request, and deallocates the conversation after each request.

Note: There is a large performance penalty when using the request setting.

TransactionMode
Specifies whether the access service or the client application manages commit and rollback statements.

Syntax TransactionMode=[short | long]

Default short
Values:-
long specifies the access service to give commitment control to the client application.
short specifies the access service to issue a commit or a rollback after each request.

Comment: The access service holds open the connection to the data source until the client application issues a commit or rollback, or until the ClientIdleTimeout value is exceeded.

SQLTransformation
Specifies the mode the access service uses for SQL transformation.
Syntax SQLTransformation=[passthrough | sybase | tsql0 | tsql1 | tsql2]

Default passthrough
Values
passthrough specifies an access service to send all SQL statements to the database system as received, without transformation. A client application uses passthrough mode to gain direct access to DBMS capabilities. *sybase* specifies an access service to perform SQL transformation of selected statements. It also allows the use of multi-part table names with the view command in SQL statements.

CSPColumnODBCVersion
Specifies ODBC version that catalog stored procedures results conform to.
This affects interoperability with ASE/CIS.

Syntax CSPColumnODBC Version = [2 | 3]

Default
3 Values
2 specifies ASE/CIS version 12.0.
3 specifies ASE/CIS version 12.5 and later.
This property affects interoperability with ASE/CIS.

ReturnNativeError
Allows a non-localized native error message and a native error severity to be returned to the client.

Syntax ReturnNativeError = [yes | no]

Default
no Values
yes specifies non-localized native error messages are returned to the client.
no specifies non-localized native error messages are not returned to the client.

SendWarningMessages
Specifies whether an access service returns warning messages to the client application.

Syntax SendWarningMessages=[no | yes]

Default
no Values
no specifies the access service not to return warning messages to the client application. *yes* specifies the access service to return warning messages to the client application.

Supplementary Note IV – Tracing
"TraceOpenServer =59" in server.cfg file & restart DC

Then see resulting files:
%SYBASE%\%SYBASE_ECON%\SERVERNAME\log\srv.log
%SYBASE%\%SYBASE_ECON%\ SERVERNAME\log\SERVERNAME.trc

Supplementry Note V – Sample dcany.cfg file

SAP support recently provided me with this sample dcany.cfg filer that they use in-house

```
[Service Library]
{Logging}
LogSvcLibStatistics=0
{Client Interaction}
SvclibDescription=Access Service Library for ODBC.

[dcmssql_DEMO2]
{Logging}
LogConnectionStatistics=yes
{ACS Required}
ConnectionSpec1=dcmssql_DEMO2
{Client Interaction}
EnableAtStartup=yes
quoted_identifier=on
{Target Interaction}
SQLTransformation=sybase
{Tracing}
TraceTarget=yes
TraceEvents=yes
TraceInterface=yes
```

Troubleshooting

The online documentation for this option is not brilliant, so when issue occur, they can prove difficult to fix. SAP tech support are very helpful but you may want to try a few things first.

- Can you connect using ISQL? Use ISQL to test connections from the replication server host to the ECDA service. Make sure you have added in the ECDA details into the SQL.ini file local to the host you are testing from.

- Is the Direct Connect server running? You can start this as a batch file or see notes above to get it added as a service.

- Connections DOWN when running 'admin who' in repserver. If these are down when you run CREATE CONNECTION, then it could be an issue with the script that is silently run when you create a connection using a 'profile'. There is a way you can step through each command the above 'profile' script executes. Re-run the CREAT ECONNECTION command but add the '**display_only'** option on the last line. It returns the list of commands it will execute at the replicate, including creating the rs_info and rs_lstcommit tables. You can take the SQL it spits out and run it manually from a ISQL session into the SQL Server database, then resume the connection.

  ```
  create connection to ECDA_TRAIN1.pubs2
  using profile rs_ase_to_msss;standard
  set username rep_maint
  set password "rep_maint_ps"
  display_only
  go
  ```

- This gives you all the commands you need:-

  ```
  -- I normally run these from eth repserver using an ISQL session like the one below.
  >> isql -S ECDA_TRAIN1 -Urep_maint -Prep_maint_ps
  Copy and paste "your" output into the SQL Server session. Do this first and create the
  connection in the repserver once you have done this bit.

  drop table rs_info
  go
  create table rs_info (rskey varchar (20), rsval varchar (20))
  go
  insert into rs_info values ('charset_name', 'iso_1')
  insert into rs_info values ('sortorder_name', 'bin_iso_1')
  go
  drop table rs_lastcommit
  go
  create table rs_lastcommit (origin int, origin_qid binary(36), secondary_qid
  binary(36), origin_time datetime, dest_commit_time datetime)
  go
  create unique clustered index rs_lastcommit_idx on rs_lastcommit(origin)
  go
  drop procedure rs_update_lastcommit
  ```

```
go
create procedure rs_update_lastcommit @origin int,@origin_qid
binary(36),@secondary_qid binary(36),@origin_time datetime as update rs_lastcommit set
origin_qid = @origin_qid, secondary_qid = @secondary_qid,origin_time =
@origin_time,dest_commit_time = getdate() where origin = @origin if (@@rowcount = 0)
begin insert rs_lastcommit (origin, origin_qid, secondary_qid,origin_time,
dest_commit_time) values (@origin, @origin_qid, @secondary_qid, @origin_time,
getdate()) end
go
drop table rs_ticket_history
go
create table rs_ticket_history (cnt numeric(8,0) identity,h1 varchar(10),h2
varchar(10),h3 varchar(10),h4 varchar(50),pdb varchar(30),prs varchar(30),rrs
varchar(30),rdb varchar(30),pdb_t datetime, exec_t datetime,dist_t datetime,rsi_t
datetime,dsi_t datetime,rdb_t datetime default getdate(),exec_b numeric(22,0),rsi_b
numeric(22,0),dsi_tnx numeric(22,0),dsi_cmd numeric(22,0),ticket varchar(1024))
go
create unique index rs_ticket_idx on rs_ticket_history(cnt)
go
grant all on rs_ticket_history to public
commit
go
drop procedure rs_send_repserver_cmd
go
CREATE PROCEDURE rs_send_repserver_cmd @rs_api VARCHAR(8000) AS declare @cmd
VARCHAR(8000), @sql varchar(50) BEGIN if (patindex('rs_rcl', lower(@rs_api)) > 0)
begin print 'The Replication Server command should not contain the keyword ''rs_rcl'''
return(1) end select @cmd = 'rs_rcl ''' + replace(@rs_api, '''', '''''') + ''' rs_rcl'
if ('rs_rcl' != substring (@cmd, datalength(@cmd) - 5, 6)) begin print 'The
Replication Server command is too long.'  print 'Please split it into two or more
commands' return (1) end set @sql = 'rs_marker' exec @sql @cmd END
commit
go
drop table rs_threads
go
create table rs_threads(id int,seq int CONSTRAINT PK_rs_threads PRIMARY KEY
CLUSTERED(id ASC))
go
sp_indexoption 'rs_threads','disallowpagelocks',TRUE
go
grant select on rs_threads to public
commit
go


--now that all our database objects have been created in your replicate database, you
can now go ahead and create the connection using the commands that were outputted.
Notive this DOES not use the ‘profile’ clause.

create connection to ECDA_TRAIN1.pubs2
set error class to rs_msss_error_class
set function string class to rs_msss_function_class
set username to rep_maint
set password to rep_maint_ps
set batch to 'off'
set dsi_dataserver_make to ase
set dsi_connector_type to ctlib
set dsi_do_decompression to 'on'
go
```