Creating a Scale-Out system With High Availability

Purpose
This document describes how to set up, monitor and maintain a read-only scale-out system which incorporates high availability.

Introduction
This document was written using SQL Anywhere 12.

Gettin Started
When planning a read-only scale out system, you must determine the number of computers you require. You require a base of three computers to act as the primary server, mirror server, and arbiter in the high availability system.

From there, you must plan how many child nodes you will need in the scale-out system. This number should be representative of the read load that your database server will encounter.

Once you have all the computers set up, you must install SQL Anywhere 12 and upgrade all of the computers to the same build and maintenance release.

The remainder of this document describes a scale-out system with three child nodes, an arbiter server, a primary server, and a mirror server. Each server will be referred to as follows:

<table>
<thead>
<tr>
<th>Role</th>
<th>Server Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arbiter</td>
<td>arbiter-pc</td>
</tr>
<tr>
<td>Partner 1</td>
<td>partner1-pc</td>
</tr>
<tr>
<td>Partner 2</td>
<td>partner2-pc</td>
</tr>
<tr>
<td>Child 1</td>
<td>child1-pc</td>
</tr>
<tr>
<td>Child 2</td>
<td>child2-pc</td>
</tr>
<tr>
<td>Child 3</td>
<td>child3-pc</td>
</tr>
</tbody>
</table>

Definitions

**Arbiter Server**: The arbiter’s purpose is to decide which partner database server is to act as the primary, and which is to act as the mirror.

**Partner Servers**: Partner servers can assume the role of the primary or mirror server in a high availability system.

**Primary Server/Root Node**: The purpose of the primary database server is to act as the root node in the scale-out system. The primary server mirrors itself to its partner and all copy nodes. It also manages all connections between child nodes and client connections.

**Mirror Server**: The mirror server acts as a live backup of the primary server. If the connection to the primary server is lost, the arbiter switches the mirror to the new primary. Client connections experience a brief disconnection (losing uncommitted transactions), before being allowed to reconnect to the new primary server.

**Quorum**: Quorum is the state at which one of the partner servers has made a successful connection with the arbiter and can be assigned the role of primary. A mirror does not have to be assigned for quorum to be achieved.

**Live Backup**: A live backup is the process of the primary server updating a mirror server after each committed transaction. This behavior ensures that a minimal amount of time is required to recover if the primary server goes down.

**Copy/Child Node**: A copy node is a read-only mirror of the database. It is used to assist the parent node in load balancing reads on the database.

High Availability
Database Mirroring

Database mirroring is the process of two database servers, a primary and a mirror, co-operating to maintain copies of the database and transaction log file. This configuration ensures minimal downtime if a server was to become unavailable. The primary and mirror servers together are known as partner servers.

A third server, known as an arbiter, determines which of the partner servers is to maintain ownership of the database (known as the primary server). In the case of the primary server becoming unavailable, the arbiter negotiates with the mirror server for the mirror server to become the new primary server. The configuration of three database servers (the primary, mirror, and arbiter servers) is called a mirroring system.

Initial Setup

After installing SQL Anywhere 12 on the computers that will host the primary, mirror and arbiter servers, you are ready to begin configuring a high availability system.

The following steps assume that the database file is located at: c:\database\demo.db on partner1-pc.

Configuring the Database

1. Start the database server on partner1-pc by running the following command:

   dbsrv12 -n demo_partner1 -x tcpip(port=6871;dobroad=no) -su sql "c:\database\demo.db" -xp on

   -n Names the database server demo_partner1.
   -x Specifies that the database server is to communicate using TCPIP (mandatory for high availability), that it is to run on port 6871 and it is not to broadcast itself over the network.
   -su Specifies the password for the database.
   -xp Sets the server to be part of a database mirroring system.

2. Connect to Interactive SQL from the database server on partner1-pc by running the following command:

   dbisql -c "uid=DBA;pwd=sql;server=demo_partner1"

3. Define the primary and mirror servers in the system by executing the following statements:

   CREATE MIRROR SERVER demo_primary AS PRIMARY connection_string='server=demo_primary;host=partner1-pc:6871,partner2-pc:6872';
   CREATE MIRROR SERVER demo_mirror AS MIRROR connection_string='server=demo_mirror;host=partner1-pc:6871,partner2-pc:6872';

4. Define demo_partner1 and demo_partner2 as partners in the database mirroring system:

   CREATE MIRROR SERVER mirror_partner1 AS PARTNER connection_string='server=demo_partner1;host=partner1-pc:6871'
state_file='c:\database\partner1.state';
   CREATE MIRROR SERVER mirror_partner2 AS PARTNER connection_string='server=demo_partner2;host=partner2-pc:6872'
state_file='c:\database\partner2.state';

5. Define the arbiter and the authentication string for the mirroring system and then disconnect from Interactive SQL:

   CREATE MIRROR SERVER demo_arbiter AS ARBITER connection_string = 'server=demo_arbiter;host=arbiter-pc:6870';
   SET MIRROR OPTION authentication_string='sql';

Configuring the System

1. Run the following command on partner1-pc, and copy c:\database\copy\demo.db to c:\database\demo.db on partner2-pc:

   dbbackup -c "uid=DBA;pwd=sql;server=demo_partner1;dbn=demo" c:\database\copy

2. Now that the database has been copied to partner2-pc, the mirror server can be started. On partner2-pc, run the following command:

   dbsrv12 -n demo_partner2 -x tcpip(port=6872;dobroad=no) -su sql "c:\database\demo.db" -xp on

3. Start the arbiter on arbiter-pc with the following command (no database file is needed on the arbiter):

   dbsrv12 -n demo_arbiter -x tcpip(port=6870;dobroad=no) -su sql -xf "c:\database\arbiter.state" -xa "auth=sql;dbn=demo"
   -xf Specifies where to store the state information for the arbiter.
   -xa Specifies the name of the database, as well as the authentication string to connect to the arbiter server.

Quorum

Before a server can assume the role of primary server, it must obtain quorum, which is achieved by making a connection with the arbiter. If the mirror server becomes unavailable while the primary server and arbiter are connected, the primary server continues to provide access to the database. If the primary server loses quorum, access can no longer be permitted to the database. When quorum is lost, the database server stops the mirrored database, and waits to re-gain quorum before making the database available for requests.

When you start a database mirroring system, each of the database servers goes through a startup process to reach quorum and accept client connections. The following steps describe a typical sequence of events for this process:

1. The arbiter server waits for Server 1 and Server 2.
2. Server 1 looks for the arbiter server or Server 2.
3. Server 1 connects to the arbiter server.
4. Server 1 negotiates with the arbiter server to become the primary server.
5. The arbiter server and Server 1 agree that Server 1 can become the primary server.
6. Server 1 starts accepting connections.
7. Server 2 looks for Server 1 and the arbiter.
8. Server 2 connects to the arbiter and to Server 1.
9. Server 2 requests quorum. It does not receive quorum because Server 1 is the primary, so it stands by waiting for transactions from
Connecting to a High Availability System

Connecting to a high availability system is similar to connecting to a stand-alone database server. The primary difference in the connection strings is related to the host parameter. Since either partner can become a primary server, both hosts must be specified, separated by a comma. For example, the connection string for the configuration described above would look like:

```
dbisql -c "uid=DBA,pwd=sql,dbn=demo;host=partner1-pc:6870,partner2-pc:6870"
```

Determining the Primary Database Server

A simple test for determining which database server is currently acting as primary is to run the following statement from Interactive SQL. The statement returns the name of the database server you are currently connected to:

```
SELECT PROPERTY ( 'ServerName' );
```

Another option is to look at the startup messages for the database server message log of a particular server. The messages should look similar to the following:

```
Database "demo" mirroring: determining mirror role ...
Database " demo " mirroring: mirror arbiter connected
Database " demo " mirroring: arbiter status: role=primary, state=synchronizing, sequence=8, yielding=N
Database " demo " mirroring: becoming primary server
```

Scale-Out Architecture

Read-only scale-out is the process of offloading read transactions on the database to child nodes. The root node contains the only writable copy of the database. The root manages connections between the copy node and clients and performs all write transactions.

The following diagram is a representation of what a scale-out system with high availability looks like:

```
Diagram 1:
```

Initial Set-up

After setting up all the child servers, you must install SQL Anywhere 12, and upgrade them to the same EBF as that of the servers in the parent
To configure your scale-out system:

1. Connect to the high availability system from Interactive SQL by running the following command:
   ```bash
dbisql –c “uid=DBA;pwd=sql;dbn=demo;host=partner1-pc:6870,partner2-pc:6870”
   ```
2. Within dbisql, execute the following commands:
   ```bash
   SET MIRROR OPTION auto_add_server = ‘demo_primary’;
   SET MIRROR OPTION child_creation = manual;
   SET MIRROR OPTION authentication_string = ‘sql’;
   SET MIRROR OPTION auto_add_fan_out = ‘10’;
   ```
3. Create a copy of the database by running the following command to copy c:\database\copy\demo.db from partner1-pc to c:\database\demo.db on child1-pc, child2-pc and child3-pc:
   ```bash
dbbbackup –c “uid=DBA;pwd=sql;server=demo_partner1;dbn=demo” c:\database\copy
   ```
4. Start the child database server by running the following command on child1-pc:
   ```bash
dbsrv12 –n demo_child1 –su sql –x TCPIP(port=6873;dobroad=no) c:\database\demo.db –xp on
   ```
5. Connect to the child database from Interactive SQL by running the following command on child1-pc:
   ```bash
dbisql –c “uid=DBA;pwd=sql”
   ```
6. Execute the following statement:
   ```sql
   CREATE MIRROR SERVER demo_child1 AS COPY FROM SERVER PRIMARY;
   ```
7. Repeat steps 4 to 6 on child2-pc and child3-pc, changing the database names respectively.

### Dropping Nodes

You can drop nodes from the system at any time with by executing the following statement:

```sql
DROP MIRROR SERVER server_name;
```

### Determining a Node’s Parent

Details about a particular node can be seen in the `SYSMIRRORSERVER` view by executing the following statement from Interactive SQL:

```sql
SELECT * FROM SYSMIRRORSERVER;
```

This statement gets the object ID, server name, server type, parent, and alternate parent for all the servers. If the parent and alternate parent columns are null, the server is at the root of the system. Otherwise, the parent references the current root, and the alternate parent references the mirror.

### Monitoring System

#### Sybase Central

Sybase Central provides a Health and Statistics tool that lets you monitor your primary, mirror, arbiter, and child servers. Use the following steps to navigate to the Health and Statistics window in Sybase Central:

1. Start Sybase Central by clicking Start > All Programs > SQL Anywhere 12 > Administration Tools > Sybase Central.
2. In Sybase Central, click Connections -> Connect with SQL Anywhere 12.
3. Enter the required connection information for your database. For the system used in this document, you must provide the following information:
4. In the left window (Folders view) click the “demo-dba” database.
5. In the main window, click the Overview tab at the top.
6. Under the Health and Statistics window, click the Database Mirroring and Scale-out field to see details about the system.
SQL Anywhere Monitor

The SQL Anywhere Monitor runs using the Adobe Flash Player, which can be downloaded independently. This means the monitor can be used from a computer that does not have SQL Anywhere installed. The SQL Anywhere Monitor is available for development in all editions of SQL Anywhere.

The following steps configure the Monitor:

1. Open the Monitor by clicking **Start > All Programs > SQL Anywhere 12 > Administration Tools > SQL Anywhere Monitor**.
2. The SQL Anywhere Monitor opens in your default web browser. To log in, use the default user ID (**admin**) and default password (**admin**).
3. In the left pane, under the **Tools** box, click the **Administration** option:

4. This opens a child window where you can add a resource:

5. Another child window appears that guides you through adding the resource. In this window click **SQL Anywhere Server** and then click **Next**.
6. In the next window, type “Demo” in the **Name** text field and then click **Next**.
7. Type, **partner1-pc:6871,partner2-pc:6872** in the **Host** text field and then click the **Create** button at the bottom of the window.
8. A third child window appears prompting you to enter the DBA user-ID and password, which you can enter as **DBA** and **sql**, respectively.

9. The **Administration** child window can now be closed, and the resource should be added in the **Resource List** on the main page.

10. Select the **Demo** resource in the **Resource List** widget. A page appears that contains various widgets to monitor your system.

11. To add the **Scale-Out Topology** widget, you must click **Customize > Add Widget**

12. This opens a child window, where you can select the **SQL Anywhere Scale-out Topology** widget. Click **Next**.

13. Name the widget **scale-out** and then click **Create**.

14. Your widget is added to the main screen, and you can expand it by double-clicking its blue header.
Upgrading The System

Applying EBFs

To apply an EBF in a scale-out system with high availability, there is no need for any downtime. Shut down each node one at a time, and then apply the new EBF. It is recommended that all nodes in the system be of the same build. During the upgrade you can mismatch them to eliminate downtime, but this should be done relatively quickly.

Applying Maintenance Releases

Unlike EBFs, it is mandatory that all nodes in a scale-out system have the same maintenance release at all times. As a result, some system downtime occurs when upgrading the system to a newer maintenance release.

Rebuilding the Database

When rebuilding a database in a scale-out system, all nodes must be started with the rebuilt version of the database at the same time so that the transaction log offsets match. This involves some downtime since one copy of the database must be rebuilt and then deployed to all nodes.