PureConnect®

2023 R3

Generated:

12-July-2024

Content last updated:

11-June-2019

See Change Log for summary of changes.

GENESYS™

CIC Database Configuration and Maintenance for SQL Server

Technical Reference

Abstract

This document describes the procedures for setting up a database server for optimum performance with CIC, with specific guidance for SQL Servers.

For the latest version of this document, see the PureConnect Documentation Library at: http://help.genesys.com/pureconnect.

For copyright and trademark information, see https://help.genesys.com/pureconnect/desktop/copyright_and_trademark_information.htm.

Table of Contents

Table of Contents 2
Introduction to Database Configuration and Maintenace for SQL Server 3
CIC Database Configuration Technical Summary 4
- I/O configuration 4
- Database size 4
- Database options configuration 5
- Database maintenance 5

Hardware 6
- CPU 6
- Memory 6
  - AWE 6
- Drives 6
- I/O and RAID 7
- I/O controller 7
- OLTP vs. OLAP 7
- I/O resting 7
- Networking 7
  - Network protocols 7
  - Routers 8
  - Network cards 8

Software 9
- Service packs 9
- Services 9
- File structure and distribution 9
  - MDF 9
  - LDF 9
  - Multiple file Strategy 9
  - Tempdb 10

SQL Server Database Settings 11
- AutoGrowth 11
- ODBC 11

SQL Server High Availability Configuration 13
- Deployment 13
- Deployment Verification 17
- Setup Assistance with SQL Server 2019 High Availability 31

Database Maintenance Plans 36
Third-party (Application) Access to Database 37
Change Log 38

Introduction to Database Configuration and Maintenace for SQL Server

A database is a collection of data that is organized so that its contents can easily be accessed, managed, and updated. A small database might contain tables of names, addresses and phone numbers. To serve that collection of information, a small database engine would suffice. However, a corporation typically stores a much larger quantity of information. A database can have millions of rows and columns of data, stored in multiple tables linked into a massive storage unit. In that situation, not only must your database engine must be powerful, but a myriad of other issues that can enhance or mire the performance of the database engine should be considered.

The *CIC Database Configuration and Maintenance for SQL Server Technical Reference* describes the procedures for setting up a database server for optimum performance with CIC, with specific guidance for Oracle. Many of the ideas here are focused on a best-case scenario, designed to tune both the hardware and software to operate at peak efficiency.

Much of the information contained within comes from two sources:

1. Website: www.sql-server-performance.com
2. The Microsoft MSDN Library.

For more information, see the Microsoft SQL Server 2008 R2 Best Practices Analyzer tool. It analyzes a database's current configuration, indicate potential problems, and recommend solutions to those problems.

For SQL Server versions supported for CIC 2015 R1 or later, see Testlab http://testlab.genesys.com/.

CIC Database Configuration Technical Summary

I/O configuration

The configuration of the I/O system is extremely important for the performance of the database. CIC can generate thousands of records per second, and all of those records must be written to the database as quickly as possible. A poorly configured I/O system can cause a queuing problem, meaning that the database is not able to keep up with all the records CIC generates.

Some recommended web sites are listed below. (Several sites state that they are for SQL Server 2005, but many of the recommendations are valid for SQL Server 2008 R2.)

- Physical Database Storage Design
- SQL Server I/O Best Practices
- Formatting drives for maximum SQL Server performance

Basic principles when considering the drives to use include:

1. When using traditional hard drives, get the highest RPM available, which is 15,000 RPM.
2. Do not get the largest drives available. It is much better to have many multiple smaller drives than a few large drives. Having a larger number of drives allows the I/O to be spread out over more devices. Drives smaller than 100 GB are ideal, though they may be hard to find. Stay away from drives that are 500 GB and larger.
3. If the storage system is a Storage Area Network (SAN), it is vital to work with the SAN administrators to set up the SAN for database performance. There are many articles available on the web that describe best practices. Here are some very useful websites that discuss using a SAN with SQL Server:
   - SAN Storage Best Practices for SQL Server
   - SAN Multipathing Part 1: What are Paths?
   - SAN Multipathing Part 2: What Multipathing Does
4. Solid State Drives (SDD) can achieve remarkable I/O rates, though they are very expensive.

Configure the server with a minimum of four (4) physically separate data storage arrays, and allocate them as follows:

1. Array1 - Operating system.
   Use RAID 1 or RAID 10.
2. Array2 - SQL (user) database data files.
   Use RAID 10.
3. Array3 - SQL (user) database transaction log files.
   Use RAID 1 or RAID 10.
4. Array4 - SQL tempdb data file and transaction log file.
   Use RAID 0, RAID 1, or RAID 10.

RAID 5 is not recommended, because of the high amount of writes that CIC generates. For databases that have extremely high volume and through put, consider placing indexes and tables onto separate arrays. In addition, potentially large tables such as the tracker tables **IntxSegment** and **Intx_Participant** may benefit from having their own filegroups.

Database size

1. Configure the `tempdb` data file between 500 and 1000mb. If Interaction Tracker and/or Interaction Recorder are used, add additional 500 to 800mb for each of them. The two links below have good recommendations for setting up temp DB. Generally, SQL Server should have one `tempdb` file for each cpu core, e.g. a machine with two dual core CPUs would have four `tempdb` files. It is important that each file is sized the same and the files are preallocated, so that `tempdb` does not have to expand frequently. For machines with many cores, a maximum of eight `tempdb` files may be enough. Monitoring will help to determine if `tempdb` is functioning optimally.
   a. Optimizing tempdb Performance
   b. Properly Sizing the SQL Server TempDB Database
2. Configure the `tempdb` transaction log file between 300 and 500mb. If you have Interaction Tracker and/or Interaction Recorder licenses as well, add additional 250 to 500mb for each of them.
3. Use the database space planning spreadsheet, `IC_DB_SpacePLanning.xls` (available in the CIC `.iso` file and in the PureConnect Documentation Library) to calculate, and then configure, the size of the CIC database. The computation should include the number of years of data planned to be kept in the database. Allow sufficient and constant free space in both the database and transaction log for rebuilding indexes. (For a 2300mb database, 1000 to 1400mb of free space in both the database and transaction log are needed to rebuild indexes that are 50% fragmented.)

4. Allow both the database and transaction logs to grow automatically (as a safeguard), but use a suitably sized growth increment.

Database options configuration

Leave the defaults in place, except for 'optimize for ad hoc workloads'.

```
sp_configure 'show advanced options',1;
reconfigure;
go
sp_configure 'optimize for ad hoc workloads',1;
reconfigure;
go
```

Database maintenance

Perform database integrity checks and/or index reorganization in a scheduled job that is separate from the daily backup maintenance plan. Regular index reorganization is vital for peak system performance. Use `sys.dm_db_index_physical_stats` to identify which indexes need to be rebuilt or reorganized. We do not recommend setting up a regular job to rebuild *all* indexes; only those indexes that have reached a pre-determined threshold of both size and fragmentation should be rebuilt or reorganized. See Microsoft's online documentation about rebuilding indexes.

Hardware

CPU

When selecting a CPU for the server, select one with a large L2 cache. This is especially important for multiple-processor servers. Select at least a 1MB L2 cache for one or two CPUs. Four or more CPUs should have at a least 2MB L2 cache in each CPU. The greater the L2 cache, the greater the server's CPU performance because it reduces the amount of wait time experienced by the CPU when reading and writing data to main memory.

Simple, single table queries and updates, along with query joins on small tables take minimal CPU processing power. On the other hand, large joins, aggregations, and sorting of large result sets use a high level of CPU processing power. Keep this in mind when choosing the hardware configuration for SQL Server.

Memory

In most cases, the more physical RAM SQL Server has the greater SQL Server's performance. If possible, purchase enough RAM to hold the largest database table in memory. If such a purchase is not possible during the initial setup, leave room for adding more RAM at a later date. **We strongly recommend running the 64-bit version of SQL Server with a minimum of 4 GB RAM.** Systems with high performance demands should be prepared to use up to 128 GB RAM or more.

To take advantage of SQL Server's ability to run parallel queries, plan on investing on more RAM. Parallel queries use much more RAM than non-parallel queries.

AWE

If SQL Server is running on a 32-bit Windows box, consider using Address Windowing Extensions (AWE) to increase the amount of memory available to SQL Server. Normally, 32-bit CPUs can only support up to 4GB of RAM because of limited address space. SQL Server supports AWE to bypass this limitation.

> **Note:**
> AWE will be removed in the next version of SQL Server. SQL Server 2008 R2 is the last version that will support AWE. We strongly recommend using the 64-bit version instead of the 32-bit version of SQL Server.

AWE support is not turned on automatically. To enable AWE support change the "awe enabled" advanced option from 0 to 1. See Enabling AWE Memory for SQL Server. To turn on AWE support:

```
SP_CONFIGURE 'show advanced options', 1;
       --you must turn on advanced
RECONFIGURE WITH OVERRIDE;                      --options


 first
GO
SP_CONFIGURE 'awe enabled', 1;
RECONFIGURE WITH OVERRIDE;
GO
```

Once AWE support has been turned on, SQL Server's dynamic memory is turned off. This means that when AWE support is turned on, the entire RAM in the server, with the exception of about 128MB, will be dedicated to use by SQL Server. For most dedicated SQL Servers, 128MB may be enough for the operating system to successfully run. But if you are running additional software on your server, you may have to tell SQL Server to claim all less RAM. To do this, you can use SQL Server's "max server memory" configuration option. For example:

```
SP_CONFIGURE 'max server memory', 4096;
RECONFIGURE WITH OVERRIDE;
GO
```

In the above example, we are telling SQL Server to only use 4GB of RAM, leaving any other RAM available in the server free for other applications.

Drives

Avoid locating read-intensive and write-intensive activity on the same drive or array. For example, do not locate an OLTP and an OLAP database or heavily random and sequential activity on the same physical device. Whenever a drive or array has to change back and forth between activities, efficiency is lost.

NTFS-formatted partitions should not exceed 80% of their capacity. For example, a 20GB drive should never hold more than 16GB. NTFS needs room to work, and when capacity exceed 80%, NTFS become less efficient and I/O suffers. Consider creating a system alert to indicate when an array exceeds 80% of capacity so that immediate action can be taken to correct the problem.

I/O and RAID

Use hardware-based RAID rather than software-based RAID because the latter can't offload the work to a separate processor, making it much slower than a hardware-based RAID solution.

Do not store the operating system, application software, or databases on single disk drives because they do not afford any fault tolerance. Instead, always choose a RAID array made up of three or more physical drives that offers fault tolerance. Common fault tolerant RAID configurations include RAID Level 1 (mirroring or duplexing) and RAID Level 10 (also called 1+0, which includes both striping without parity and mirroring). Non fault tolerant RAID configurations include RAID 0 which is simple disk striping. RAID 0 offers excellent performance, and can be used for the TEMP tablespace. Each of these RAID levels offers different performance levels. Ideally, if the budget allows, chose RAID Level 10, which offers both high-speed and fault tolerance.

I/O controller

Select the best I/O controller possible. Top-notch controllers offload much of the I/O work onto its own local CPU, freeing up CPU time on the server to do other tasks. For the ultimate in I/O controllers, consider a fiber channel connection instead of a SCSI connection. The controller should have the largest amount of cache RAM possible, with a minimum of 128mb of cache RAM. Generally, the greater the RAM cache on the controller, the higher the performance of the overall I/O, because data can be read ahead and stored in the cache, even if the data is not currently requested by Oracle. The data Oracle wants next from the array will likely be in the cache, speeding up data access.

Do not put DAT, DLT, CD-ROM, scanners, or other non-hard disk devices on the same I/O controllers that connect to the hard disk arrays. In addition, do not put hard disks on the same I/O controller if they have different speeds. Putting devices with different speeds on the same I/O controller slows the faster devices. Always put slower devices on their own I/O controller.

For maximum I/O throughput, assign each type of major I/O activity (database, log files, tempdb, etc.) to its own separate RAID controller and dedicated RAID array.

OLTP vs. OLAP

If the budget does not allow for the ideal number of disk controllers and hard disks to maximize the server's I/O performance, remember that optimal OLTP I/O is achieved by increasing disk reads and writes. The best way to do this is to add more hard disks to the array(s) that hold the database files and/or transaction logs. Adding more disks helps OLTP-based applications more than increasing the number or speed of disk controllers would, because OLTP-based applications tend to be limited by the number of transfer operations (read/writes) rather than bandwidth.

However, for OLAP-based applications, adding more and faster disk controllers to the array is generally a better way to boost I/O than increasing the number of disk drives, because OLAP applications tend to be more limited by bandwidth than by read/write operations. Adding faster or more disk controllers increases the bandwidth and helps to remove any bottlenecks.

I/O resting

There are several utilities available to test I/O subsystem performance. We recommend SQLIO and this tutorial.

Networking

If SQL Server is not connected to a switch (as recommended for best performance), try the following suggestions for boosting network performance.

Network protocols

For best performance, SQL Server should be running on a dedicated server. Limit the number of network protocols installed on the server, because unnecessary network protocols increase overhead on the server and send out unnecessary network traffic. For the best overall performance, only install TCP/IP on the server.

Routers

While not always possible (especially for WANs and Internet connections), try to avoid a router between SQL Server clients and SQL Server. In particular, avoid routers between two or more SQL Servers that need to communicate with each other. Routers are often a bottleneck for network traffic and can affect SQL Server client/server performance. If SQL Server must communicate over a router, ensure that the router has been properly tuned for maximum performance.

Network cards

SQL Server should have a minimum of one 100Mbs network card, and perhaps two. Two cards can be used to increase network throughput and to offer redundancy. In addition, the network card(s) should be connected to full-duplex switched ports for best performance.

Be sure that the network card(s) in the server are set to the same duplex level (half or full) and speed as the switched port they are connected to (assuming they are connected to a switch and not a hub). If there is a mismatch, the server may still be able to connect to the network, but network performance can be significantly impaired.

Do not rely on network cards or switches that are supposed to auto-sense duplex or speed settings, because they often do not work correctly. Manually set the duplex and speed for the card from the operating system, and if necessary, manually make the same changes to the switch.

Windows allows network cards to save energy by going to sleep when they are not used. If any network card on a production server has a power management feature, ensure that the power savings feature are off. Otherwise, unexpected results, such as a network card that fails to wake up, or intermittent performance problems, may occur.

Check to see if the network card has a power management feature by viewing the **Properties** sheet for the network card's driver. View the **Power Management** tab on the **Properties** sheet, to verify the settings.

Software

The network libraries chosen during SQL Server installation can affect the speed of communications between the server and its clients. Of the three key network libraries, TCP/IP is the fastest and Multi-Protocol is the slowest. Due to the speed advantage, use TCP/IP on both the servers and clients. Do not install unused network libraries on the server, because they will contribute unnecessary overhead.

Service packs

We recommend staying current with database service packs and maintenance. In most cases, you will want to install the latest SQL Server service packs.

Services

Do not install unnecessary SQL Server services, such as Microsoft Search, OLAP, or English Query, as they only add additional overhead to your server.

If applications do not use the Microsoft Distributed Transaction Coordinator (MS DTC), turn this service off by setting it to manual using the Services icon in the Control Panel. Leaving this service on adds unnecessary overhead to your server.

File structure and distribution

Place the database files (`.mdf`, `.ndf`) and transaction log files (`.ldf`) for all production databases on separate arrays to isolate potentially conflicting reads and writes. This means that the server will have at least two physical RAID arrays, one to store the database files, and a separate one to store the transaction log files. The operating system can be stored on a mirrored set of drives. Use separate storage areas for `tempdb` and the operating system. Since `tempdb` is rebuilt each time SQL Server is started, this storage does not have to be fault tolerant (i.e., you could use RAID 0).

The physical location for the `master`, `msdb`, and `model` databases is not as critical as user databases because they are not used excessively in production environments.

MDF

For database files (`.mdf`), the best performance is gained by storing them on RAID 10 arrays. Each RAID array should have as many physical disks in the array as the controller will support, with the fastest RPM available. This allows reads and writes to be performed simultaneously on each physical drive in the array, significantly boosting disk I/O.

LDF

For database log files (`.ldf`), the best performance is often gained by storing them on a RAID 1 (mirrored or duplexed) array. This assumes that there is only a single log file on the RAID 1 array. If there is only a single log file on the RAID 1 array, the file can be written to sequentially, speeding up log writes. But if there are multiple log files (from multiple databases) sharing the same RAID 1 array, then there is little or no advantage of using a RAID 1 array. This is because although writing to a log is done sequentially, multiple log files on the same array means that the array will no longer be able to write sequentially, but will have to write randomly, negating much of the benefits of a RAID 1 array.

> **Note:**
> You can put each database log on its own separate RAID 1 array. Another option is to put the log on a RAID 10 array. While this is expensive, it will provide optimum performance.

Multiple file Strategy

If your database is very large and very busy, multiple files can be used to increase performance. One example of using multiple files would be a single table with 10 million rows that is heavily queried. If the table is in a single file, such as a single database file, then SQL Server would only use one thread to perform a sequential read of the rows in the table. But if the table were divided into three physical files (all part of the same filegroup), then SQL Server would use three threads (one per physical file) to sequentially read

the table, which potentially could be much faster. In addition, if each file were on its own separate disk or disk array, the performance would even be greater.

Essentially, the more separate physical files that a large table is divided into, the greater the potential performance. Of course there is a point where the additional threads aren't of much use when you max out the server's I/O. But up until you do max out the I/O, additional threads (and files) should increase performance.

Tempdb

If SQL Server's `tempdb` database is heavily used by your application(s), then locate it on an array of its own (such as RAID 0, RAID 1 or RAID 10). If SQL Server has multi-gigabyte databases, create a new file for `tempdb` that is at least 1 GB in size. Having multiple `tembdb` files can also improve performance, taking care to make sure that each file is the same size.

To move the tempdb database after SQL Server is installed, run this script to move it to a more appropriate location:

```
USE master
go
ALTER DATABASE tempdb MODIFY FILE (NAME = tempdev, FILENAME = 'E:\tempdb.mdf')
go
ALTER DATABASE tempdb MODIFY FILE (NAME = templog, FILENAME = 'E:\templog.ldf')
go
```

where `NAME` refers to the logical name of the `tempdb` database and log files, and `FILENAME` refers to the new location of the `tempdb` files. Once this command has run, restart the mssqlserver service so the change can take effect. Increase the size of the `tempdb` database in its new location (to 500mb, or larger) if necessary.

If `tempdb` database is heavily used and `tempdb` grows larger than its default size, permanently increase the default size of the `tempdb` file(s) to a size closer to what is actually used on a day-to-day basis. This is because every time the SQL Server service (mssqlserver) is restarted, the `tempdb` file is recreated to the default size. While the `tempdb` file can grow, it does take some resources to perform this task. With the `tempdb` file sized correctly when SQL Server is restarted, the overhead of increasing the file is eliminated.

Database integrity checks like `DBCC CHECKDB` or `DBCC CHECKCATALOG` will need to use space in `tempdb` during their operation. Therefore it is essential that the `tempdb` be sized correctly to accommodate these commands. Use the `WITH ESTIMATEONLY` clause on the `DBCC` command to determine how much `tempdb` space will be required. Execution of these commands can also realize a significant performance improvement if `tempdb` is located on a disk array that is separate from both the database and transaction log files.

Heavy activity in the `tempdb` database can drag down performance. This is especially true if large temp tables are created, and then used in queries and joins. To help speed these queries, be sure the `AUTOSTATS` database option is turned on (in `tempdb`), and then create one or more indexes on these temp tables that can be used by queries. A proper setup can substantially speed up application performance.

SQL Server Database Settings

AutoGrowth

Every time a database file or transaction log grows automatically, it takes up a little extra CPU and I/O time. Minimize how often automatic growth occurs by ***sizing the database and transaction logs as accurately as possible to their "final" size***.

This recommendation is particularly important for transaction logs, because the more often that SQL Server has to increase the size of a transaction log, the more transaction log virtual files that have to be created and maintained by SQL Server. A transaction virtual file is used by SQL Server to internally divide and manage the physical transaction log file.

In SQL Server, database and log files can be set to grow automatically. The default growth amount is 10%. This automatic growth number may or may not be ideal. If the database is growing automatically often (such as daily or several times a week), change the growth percentage to a larger number, such as 20% or 30%. Each time the database has to be increased, SQL Server will suffer a small performance hit. By increasing the amount the database grows each time, the less often it will have to grow.

If your database is very large, 10GB or larger, you may want to use a fixed growth amount instead of a percentage growth amount. This is because a percentage growth amount can be large on a large database. For example, a 10% growth rate on a 10GB database means that when the database grows, it will increase by 1GB. If the percentage growth is too large, change the settings to use a fixed growth size.

Size the database properly, to ensure that the database is not subject to frequent growth. In addition, set the growth increment to a reasonably sufficient value, to prevent the database from growing in numerous, small increments. In this configuration, the Auto grow feature is used as a safeguard only to prevent the database from stopping if it unexpectedly runs out of space. The table below shows typical growth increments for databases of the indicated size.

Database Size	Growth Inc.	Tran Log Size	Growth Inc.
1gb	100 – 200mb	200 – 300mb	50 – 100mb
2gb	200 – 400mb	400 – 600mb	100 – 200mb
3gb	200 – 400mb	600 – 900mb	150 – 250mb
4gb	250 – 400mb	800 – 1000mb	200 – 300mb
5gb	250 – 500mb	1000 – 1500mb	250 – 350mb
6gb	300 – 500mb	1000 – 1700mb	300 – 400mb
7gb	350 – 600mb	1000 – 1850mb	300 – 400mb
8gb	400 – 650mb	1000 – 2000mb	300 – 450mb
9gb	400 – 700mb	1000 – 2400mb	300 – 500mb
10gb	500 – 1000mb	1000 – 2500mb	300 – 500mb

Additionally, the use of a file defragmentation tool is recommended. Be aware that these tools are very resource intensive, so you should only run them when the server is not servicing production requests.

ODBC

Do not use ODBC connection pooling and temporary stored procedures at the same time, or SQL Server will experience a performance hit. When a DSN is used to make a connection from your application to SQL Server, the MDAC driver, by default, converts any dynamic Transact-SQL from the application to temporary stored procedures in SQL Server. The theory behind this is that if the application resends the same Transact-SQL to SQL Server more than once, then it will save the SQL Server overhead of additional parsing and compilation. The recommended configuration consists of turning the convert T-SQL to temporary stored procedure feature off. This feature is configurable from the ODBC Database Wizard when creating or modifying a DSN.

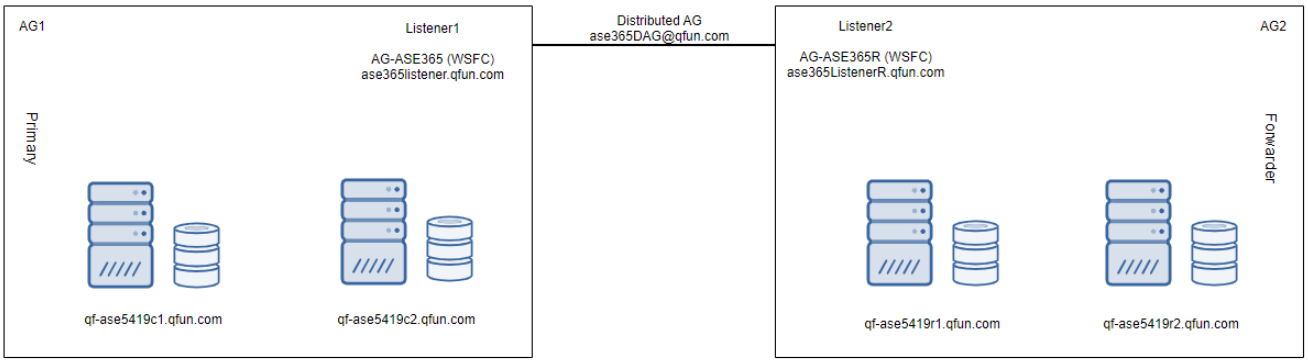
Connection pooling is another option that can be configured using the ODBC Database Wizard when creating or modifying a DSN. It is also on by default, and it pools database connections from the application, which allows connections to be reused, which in turn reduces the overhead of making and breaking database connections. The recommended configuration consists of turning the connection pooling feature on. Note that pooling improves performance more than temporary stored procedures.

The problem is that if both of these options are on, which is often the case in DSNs, SQL Server can take a performance hit. Here's what can happen: When dynamic Transact-SQL is converted into a temporary stored procedure by the MDAC driver, the temporary stored procedure is stored in the `tempdb` database. When connection pooling is not enabled, and the connection between the client application and SQL Server is ended, any temporary stored procedures created during the connection are deleted. But, when connection pooling is enabled, things work differently. When a database connection is ended by the client application, it is not ended at SQL Server. SQL Server still thinks the connection is still open, even though the client application does not. This means the temporary stored procedures created during the connection are not deleted. With a busy client application that often starts and stops database connections, the `tempdb` database can fill up with temporary stored procedures, putting unnecessary overhead on SQL Server.

SQL Server High Availability Configuration

Deployment

SQL 2016 Distributed Availability Group (DAG)



Important Specifications:

Asset	Description	Specification
qf-ase5419c1.qfun.com	AG1 VM	4GB, CPUCore(2), Partition(100GB, qty.2), Win2K19, SQL2019
qf-ase5419c2.qfun.com	AG1 VM2	4GB, CPUCore(2), Partition(100GB, qty.2), Win2K19, SQL2019
qf-ase5419r1.qfun.com	AG2 VM	4GB, CPUCore(2), Partition(100GB, qty.2), Win2K19, SQL2019
qf-ase5419r2.qfun.com	AG2 VM2	4GB, CPUCore(2), Partition(100GB, qty.2), Win2K19, SQL2019
ase365listener.qfun.com	AG1 Listener	alias
ase365listeneR.qfun.com	AG2 Listener	alias
ase365DAG.qfun.com	cname for AG1 Listener	alias (initially points to ase365listener.qfun.com)
AG-ASE365.qfun.com	Windows Failover Cluster1	alias
AG-ASE365R.qfun.com	Windows Failover Cluster2	alias
qf-ase54b.qfun.com	HyperV 2016 Host (Primary Cluster)	HP DL-360 Gen8, CPUCore(24), 16GB, Win2k16
qf-ase54c.qfun.com	HyperV 2016 Host (Secondary Cluster)	HP DL-360 Gen8, CPUCore(24), 128GB, Win2k16

To use on PureConnect IC, Setup Assistant, do the following:

SQL Management Studio, SQL server 2019 and Tools can link to the Listeners directly on qf-ase365Listener.qfun.com (qfun.com subnet) and qf-ase365ListenerR.qfun.com (dev2000.com subnet)

Authentication: Windows Auth: qfun.com\administrator , password: testing and SQL Auth: sa , password: testing (additional authentication: qfun\sqladmin , password: testing)

After authentication, you should be able to add databases on the AG1 . Primary location on qf-ase5419c1.qfun.com that will then be replicated to the other nodes: qf-ase5419c2, qf-ase5419r1, and qf-ase5419r2.

Install IC Server. Using Setup Assistant, create IC Server Database using default SQL driver. Follow steps 10 - 13 to setup link to SQL Server How To - Install CIC Interaction Center Server on your IC Server VM - Testing - Intranet (genesys.com) to the DAG: ase365DAG@qfun.com which is the alias for the Primary Listener: ase365listener.qfun.com

Note: Download and install SQL ODBC Driver 13.1 on Pureconnect Server https://www.microsoft.com/en-in/download/details.aspx?id=36434 and Native Client Driver https://www.connectionstrings.com/sql-server-native-client-11-0-odbc-driver/info-and-download/

Adding the IC Database to the Availability Group

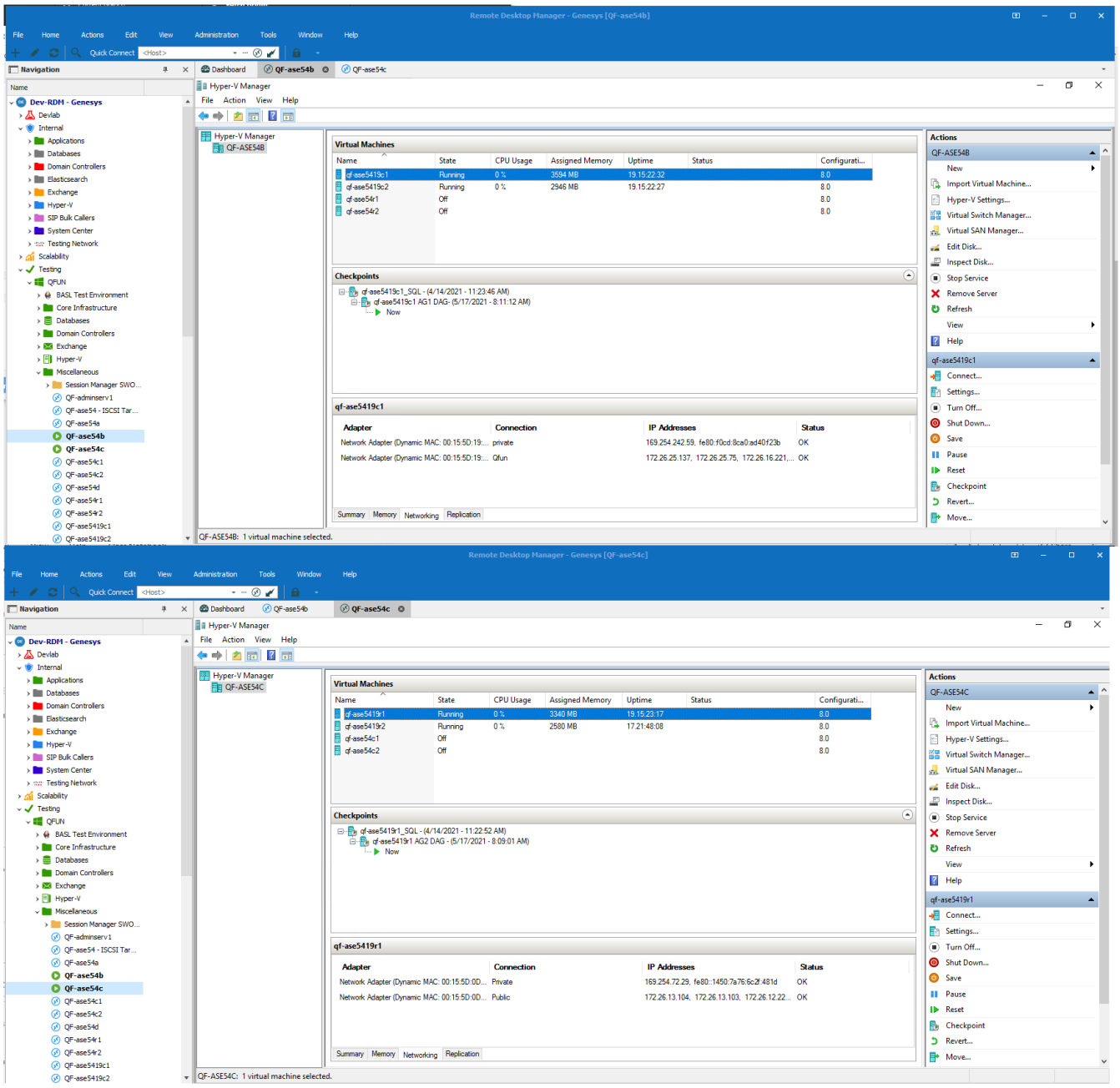
After creating the IC database, it needs to be added to the availability group so it will replicate across the replicas inside the availability group as well as the distributed availability group

In SQL Management Studio, connect to the listener of the availability group that is primary in the DAG (typically going to be ase365listener.qfun.com). You should see the newly created database.

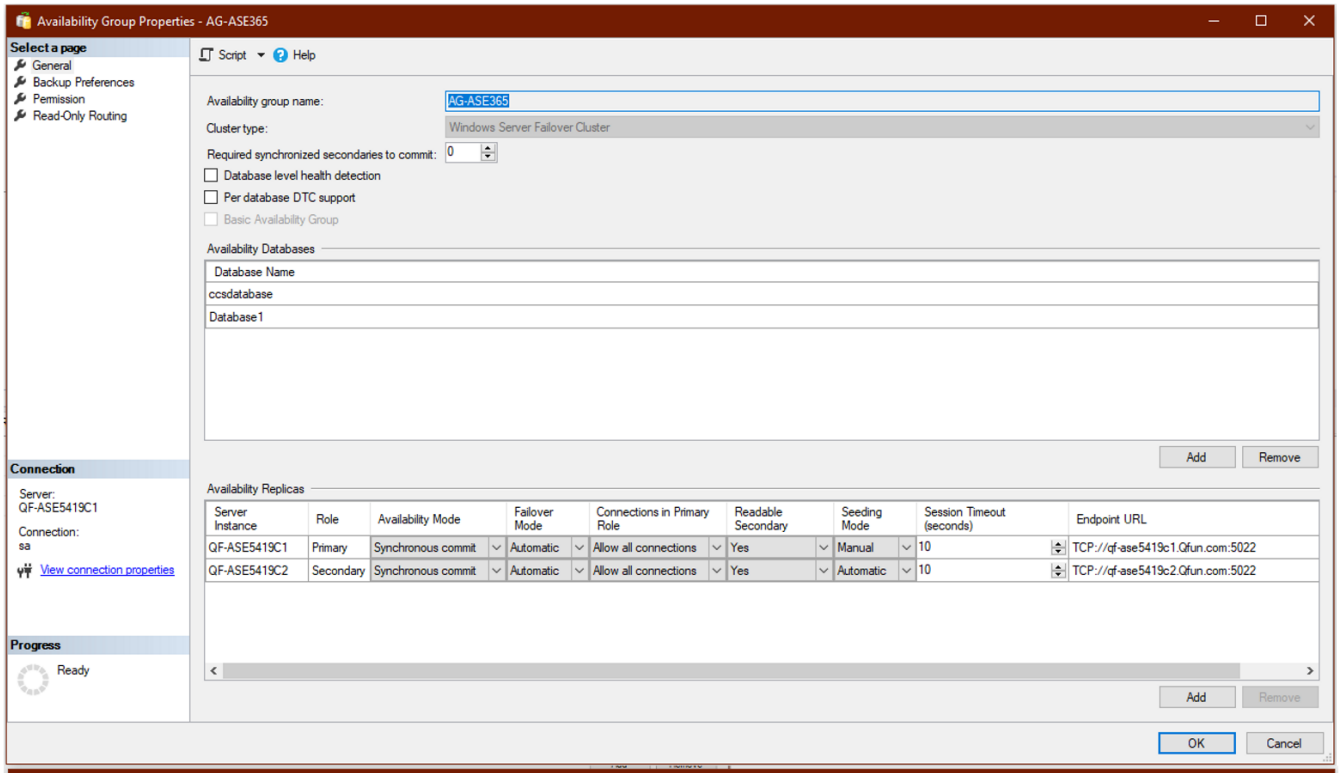
2. Right-click the database and select **Tasks** → **Backup**

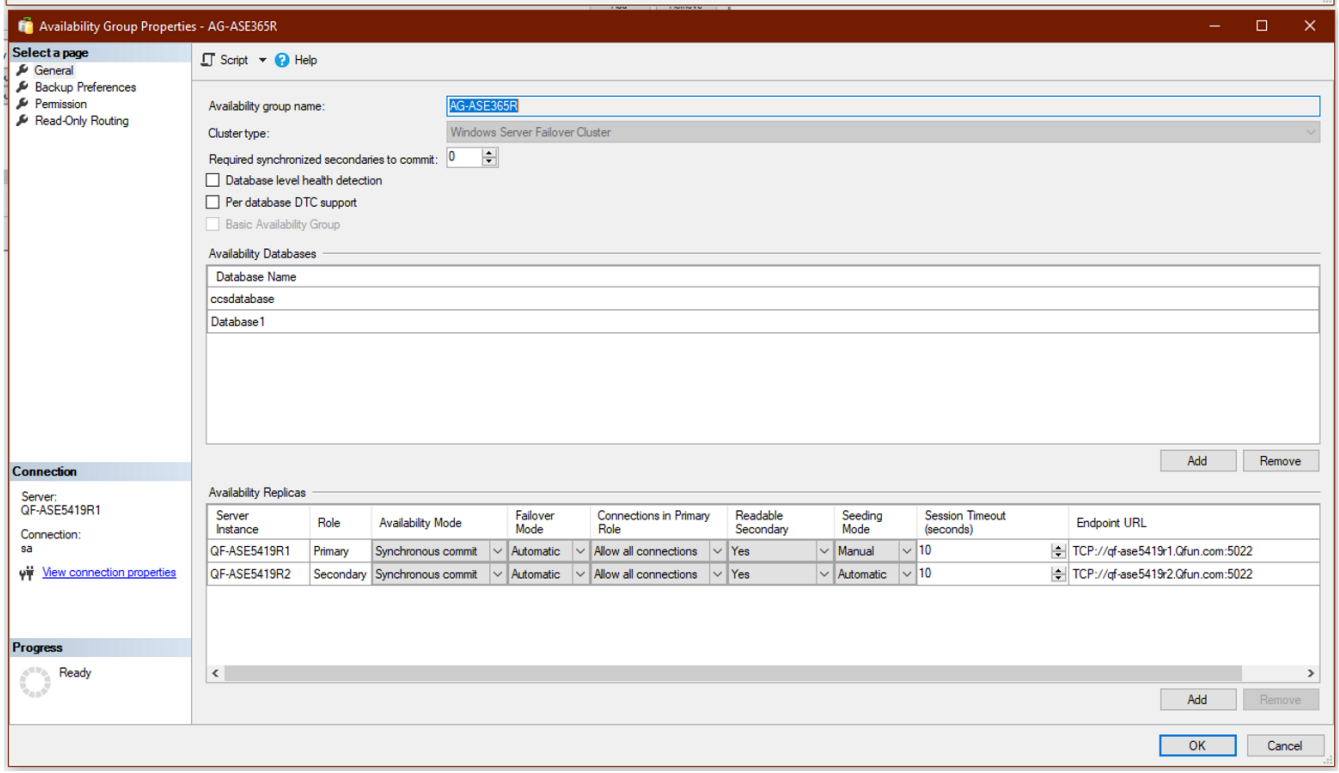
3. In the box that pops up, choose the defaults and select **OK**
4. Expand **Always On High Availability** → **AG-ASE365**, right-click **Availability Databases** and select **add database**
5. Follow the wizard to add the IC database to the availability group
6. Connect to each database instance in the environment and verify the database is synchronized across the replicas and DAG

Remote Desktop Manager - Shortcuts on the Miscellaneous (Section of Remote Desktop Manager - RDM)



Windows Clusters





Important SQL Query used to troubleshoot:

The query was used to get the qfun\sqladmin account access to the endpoint

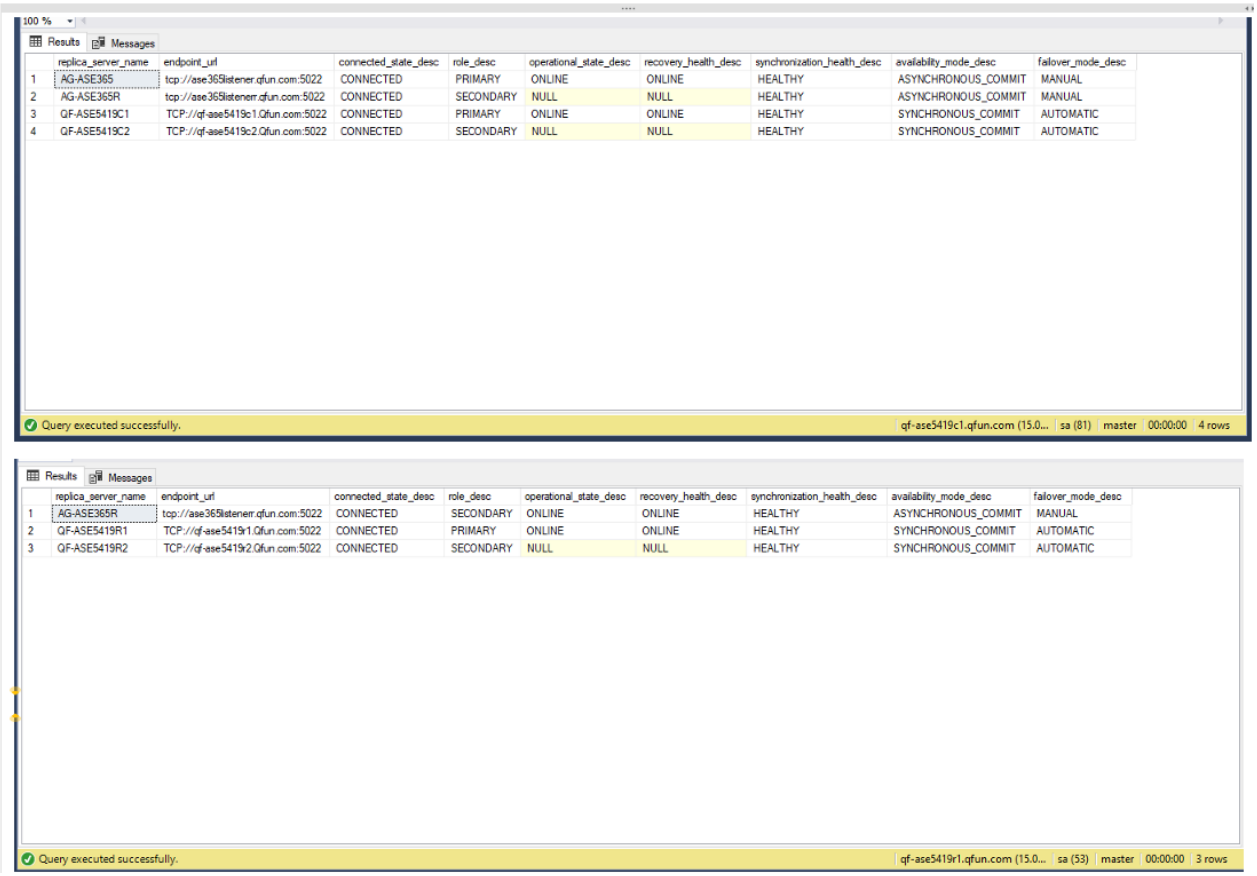
Create login and grant CONNECT permissions to the SQL Server service account

```
USE master
GO
CREATE LOGIN [qfun\sqladmin] FROM WINDOWS;
GO
GRANT CONNECT ON ENDPOINT::Hadr_endpoint
```

```
TO [qfun\sqladmin];
GO
```

The query will allow us to test if the AG1 and AG2 setups are working

```
SELECT r.replica_server_name, r.endpoint_url,
rs.connected_state_desc, rs.role_desc, rs.operational_state_desc,
rs.recovery_health_desc,rs.synchronization_health_desc,
r.availability_mode_desc, r.failover_mode_desc
FROM sys.dm_hadr_availability_replica_states rs
INNER JOIN sys.availability_replicas r
ON rs.replica_id=r.replica_id
ORDER BY r.replica_server_name
```



	replica_server_name	endpoint_url	connected_state_desc	role_desc	operational_state_desc	recovery_health_desc	synchronization_health_desc	availability_mode_desc	failover_mode_desc
1	AG-ASE365	tcp://ase365listener.qfun.com:5022	CONNECTED	PRIMARY	ONLINE	ONLINE	HEALTHY	ASYNCHRONOUS_COMMIT	MANUAL
2	AG-ASE365R	tcp://ase365listenerr.qfun.com:5022	CONNECTED	SECONDARY	NULL	NULL	HEALTHY	ASYNCHRONOUS_COMMIT	MANUAL
3	QF-ASE5419C1	TCP://qf-ase5419c1.Qfun.com:5022	CONNECTED	PRIMARY	ONLINE	ONLINE	HEALTHY	SYNCHRONOUS_COMMIT	AUTOMATIC
4	QF-ASE5419C2	TCP://qf-ase5419c2.Qfun.com:5022	CONNECTED	SECONDARY	NULL	NULL	HEALTHY	SYNCHRONOUS_COMMIT	AUTOMATIC

Query executed successfully. qf-ase5419c1.qfun.com (15.0... sa (81) master 00:00:00 4 rows

	replica_server_name	endpoint_url	connected_state_desc	role_desc	operational_state_desc	recovery_health_desc	synchronization_health_desc	availability_mode_desc	failover_mode_desc
1	AG-ASE365R	tcp://ase365listenerr.qfun.com:5022	CONNECTED	SECONDARY	ONLINE	ONLINE	HEALTHY	ASYNCHRONOUS_COMMIT	MANUAL
2	QF-ASE5419R1	TCP://qf-ase5419r1.Qfun.com:5022	CONNECTED	PRIMARY	ONLINE	ONLINE	HEALTHY	SYNCHRONOUS_COMMIT	AUTOMATIC
3	QF-ASE5419R2	TCP://qf-ase5419r2.Qfun.com:5022	CONNECTED	SECONDARY	NULL	NULL	HEALTHY	SYNCHRONOUS_COMMIT	AUTOMATIC

Query executed successfully. qf-ase5419r1.qfun.com (15.0... sa (53) master 00:00:00 3 rows

The query was used to create the AG and Listener Link

Create Distributed Availability Group

Run this on the primary replica of the primary Availability Group

```
:CONNECT QF-ASE5419C1
DROP AVAILABILITY GROUP [DAG]
ALTER AVAILABILITY GROUP [DAG]
JOIN
AVAILABILITY GROUP ON
'AG-ASE365' WITH
(
LISTENER_URL = 'TCP://ASE365Listener.QFUN.COM:5022',
AVAILABILITY_MODE = ASYNCHRONOUS_COMMIT,
```

```
FAILOVER_MODE = MANUAL,
SEEDING_MODE = AUTOMATIC
),
'AG-ASE365R' WITH
(
LISTENER_URL = 'TCP://ASE365ListenerR.QFUN.COM:5022',
AVAILABILITY_MODE = ASYNCHRONOUS_COMMIT,
FAILOVER_MODE = MANUAL,
SEEDING_MODE = AUTOMATIC
);
GO
```

Additional command needed to grant rights to create DBs

```
ALTER AVAILABILITY GROUP [DAG] GRANT CREATE ANY DATABASE;
GO
```

alter availability group [AG-ASE365R]

grant create any database.

For more Information Availability Groups, see links below

What is a distributed availability groups - SQL Server Always On | Microsoft Docs

https://www.mssqltips.com/sqlservertip/5053/setup-and-implement-sql-server-2016-always-on-distributed-availability-groups/

Download SQL Server PowerShell Module - SQL Server | Microsoft Docs

PowerShell Cmdlets for availability groups - SQL Server Always On | Microsoft Docs

Deployment Verification

For the deployment verification, do the following:

After creating the IC database, it is added to the availability group, so that it replicate across the replicas inside the availability group as well as the distributed availability group.

The SQL server login accounts are to be synchronized between the availability replicas in SQL server Always on the availability groups.

The Synchronize logins between availability replicas in SQL Server is achieved by below two methods.

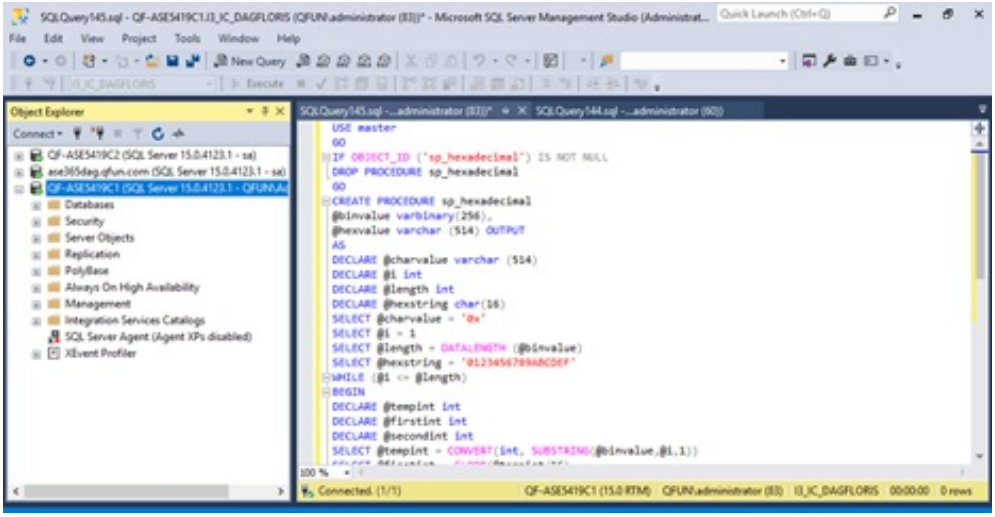
1. Use stored procedure sp_help_revlogin and sp_hexadecimal
2. Create the login on the Secondary replica manually.

The following steps help you to synchronize logins using stored procedure sp_help_revlogin and sp_hexadecimal

The script creates two stored procedures in the master database. The procedures are named **sp_hexadecimal** and **sp_help_revlogin**.

Microsoft provides the URL stored procedure **sp_help_revlogin** to transfer the logins to a different instance.

Open the URL https://docs.microsoft.com/en-US/troubleshoot/sql/security/transfer-logins-passwords-between-instances and copy the scripts to create the stored procedure **sp_help_revlogin** on the primary replica.(QF-ASE5419C1)

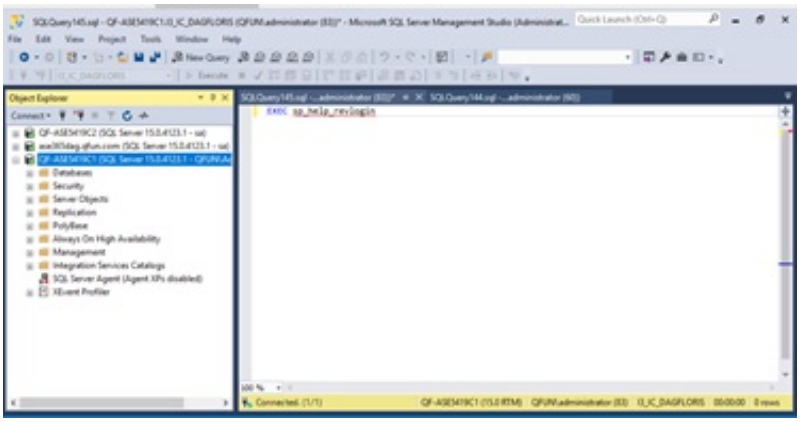


Execute the stored procedure sp_help_revlogin. (QF-ASE5419C1)

Run the following statement in the same or a new query window:

EXEC sp_help_revlogin

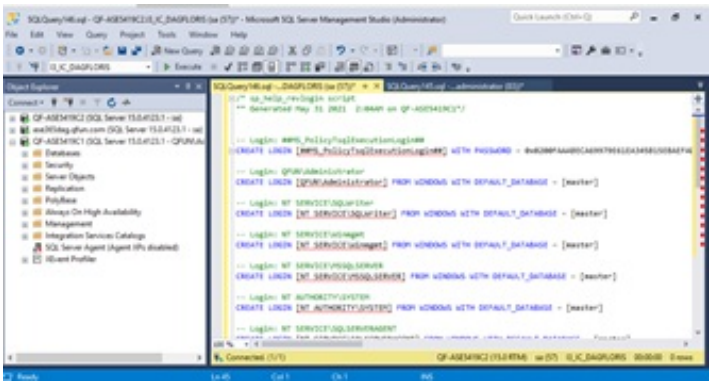
It returns the login script with the original SID and password.



The output of sp_help_revlogin shown below.

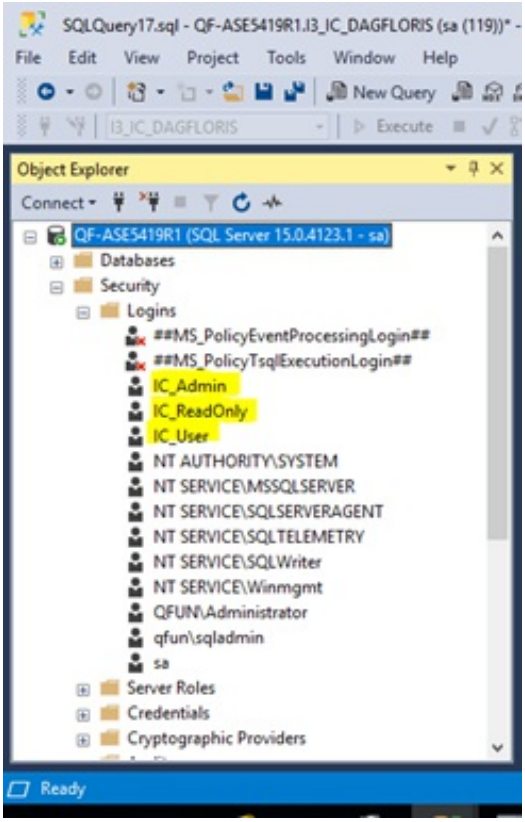
```
/* sp_help_revlogin script
** Generated May 31 2021  2:04AM on QF-ASE5419C1*/


-- Login: ##MS_PolicyTsqlExecutionLogin##
CREATE LOGIN [##MS_PolicyTsqlExecutionLogin##] WITH PASSWORD = 0x0200FAAABECA69979E61EA345815EBAEFADFC3749F48253C5C9120D021454447EA8A7E6A578729C

-- Login: QFUN\Administrator
CREATE LOGIN [QFUN\Administrator] FROM WINDOWS WITH DEFAULT_DATABASE = [master]

-- Login: NT SERVICE\SQLWriter
CREATE LOGIN [NT SERVICE\SQLWriter] FROM WINDOWS WITH DEFAULT_DATABASE = [master]

-- Login: NT SERVICE\Winmgmt
CREATE LOGIN [NT SERVICE\Winmgmt] FROM WINDOWS WITH DEFAULT_DATABASE = [master]

-- Login: NT SERVICE\MSSQLSERVER
CREATE LOGIN [NT SERVICE\MSSQLSERVER] FROM WINDOWS WITH DEFAULT_DATABASE = [master]



-- Login: ##MS_PolicyEventProcessingLogin##
CREATE LOGIN [##MS_PolicyEventProcessingLogin##] WITH PASSWORD = 0x0200E6DC648893938191F08E9DDF9F88EF346797E920B0567A50D9D7FF83237A5

-- Login: qfun\sqladmin
CREATE LOGIN [qfun\sqladmin] FROM WINDOWS WITH DEFAULT_DATABASE = [master]

-- Login: IC_Admin
CREATE LOGIN [IC_Admin] WITH PASSWORD = 0x020097D5D61353830DFBDA206AE46F5C310D18FFA28858865DB1F6501687C1A269CD09DDDFC9AAAFDA8C2848FA

-- Login: IC_User
CREATE LOGIN [IC_User] WITH PASSWORD = 0x020018893D558582D588D3CB2E156DCFD3FA8887C090319E0ED66FE657AB6D8FC6E090AE268D2B05D2616829748

-- Login: IC_ReadOnly
CREATE LOGIN [IC_ReadOnly] WITH PASSWORD = 0x0200C5A0A5885B386F88137AEB58A4090249412632AD0F14885C896CBEA64F512D25A46E3AE1804E24442AD

--Completion time: 2021-05-31T02:04:36.1298976-04:00
```

Connect to the secondary replica(QF-ASE5419C2), execute the output of **sp_help_revlogin**



Connect to the AG2 primary(QF-ASE5419R1) and secondary (QF-ASE5419R2), execute the output of sp_help_revlogin

The SQL server login accounts IC_Admin, IC_User and IC_ReadOnly are synchronized within replica and distributed availability group.



Note: The IC_Admin,IC_ReadOnly,IC_User is created under login tree structure in each replica confirms the login synchronization.

The following sanity test procedure is for always on the availability group in the SQL server

The failover modes in SQL Server Always on the Availability Groups depend upon these **Synchronous** or **Asynchronous** commit. In a high-level, SQL Server supports the following failovers in the AG.

- **Synchronous:** Primarily receives the acknowledgement from the secondary replica, and then it commits transaction on the primary.
- **Asynchronous:** Primarily commits the transactions without waiting for any confirmation from the secondary replica.

Selecting a mode is equivalent to selecting whether you want to favor **data protection or transaction performance**. Both availability modes follow the same work flow, with one small yet critical difference.

With **synchronous commit mode**, the application does not receive confirmation that the transaction committed **until after the log records are hardened on all synchronous secondary replicas**. This is how AGs can guarantee zero data loss.

With **asynchronous commit mode**, the application receives confirmation that the transaction committed after the last log record is flushed (step 1) to the primary replica's log file. This improves performance because the application does not have to wait for the log records to be transmitted but it opens up the AG to the potential of data loss.

The failover modes in SQL Server Always on Availability Groups depend upon these Synchronous or Asynchronous commit. In a high-level, SQL Server supports the following failovers in AG.

- Planned Manual failover
- Automatic failover
- Forced failover (manual)

You cannot have automatic or planned manual failover in an asynchronous commit mode. In a forced failover, you can failover to a secondary replica, especially in a disaster recovery scenario.

In an asynchronous commit, the secondary replica might fall behind with the primary database. Therefore, if failover, it causes data loss depending upon the sync lag. You should do force AG failover only, if you need to restore database availability and you can afford to lose some data.

For a forced failover, your availability database should be in the Asynchronous commit mode. It only supports a manual failover. If your AG database is in synchronous mode, but these databases are not synchronized, it only can have a forced failover.

The following are the Sanity test details:

New user creation and run the SQL query **before and after the failover**.

SQL Query : Select * from Dbo.UserUUid

Make inbound call to IC and run the SQL query **before and after the failover**.

Use the SIP Bulk caller for the inbound call.

SQL Query :

a. Select * from ININ_ANALYTICS.IVR_AllEvents
b. Select * from dbo.calldetail_viw order by InitiatedDateTimeGMT desc
c. Select * from Dbo.user.calldetails_viw
d. Select * from Dbo.IAgentQueueStats

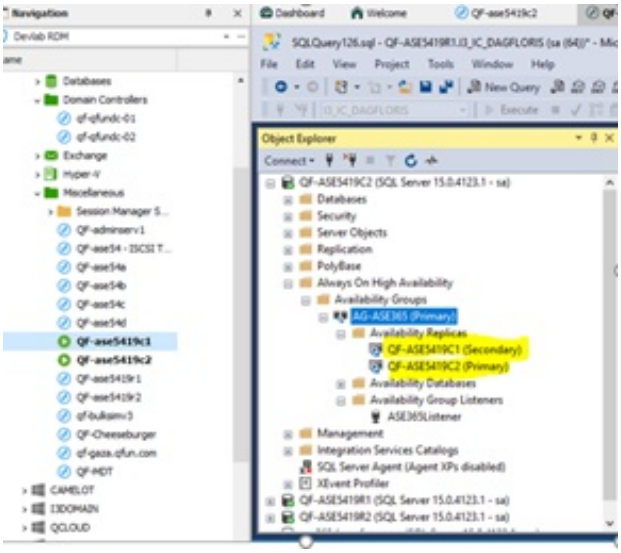
3. Run the SQL query in **AG1 QF-ASE5419C1/QF-ASE5419C2** and **AG2 QF-ASE5419R1/QF-ASE5419R2** to ensure that database is synchronized and there is no data loss after the failover.

Fail over case within the replicas inside the availability group

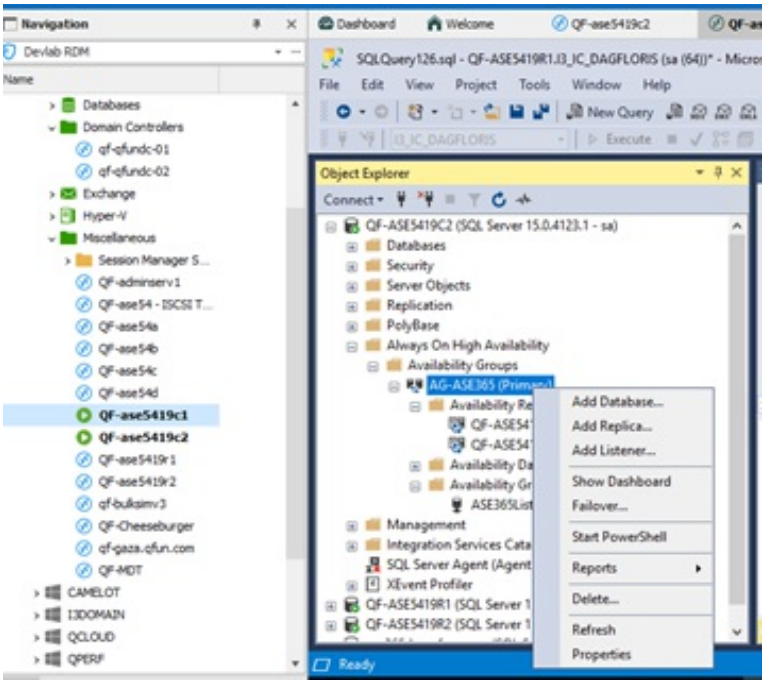
Planned Manual failover mode

The present status distributed availability group and the replicas inside the availability group.

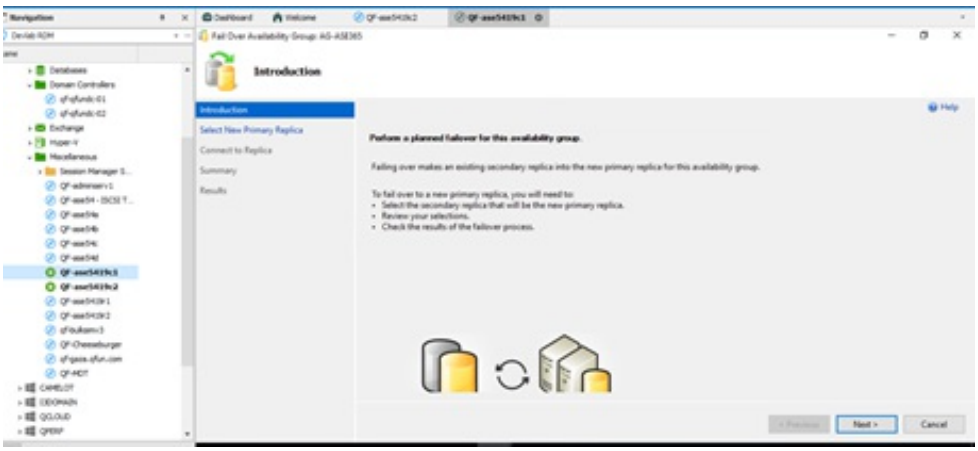
a. Distributed availability group status : AG-ASE365 (Primary)
b. Replicas inside the availability group status : QF-ASE5419C2(Primary) ,QF-ASE5419C1 (Secondary)



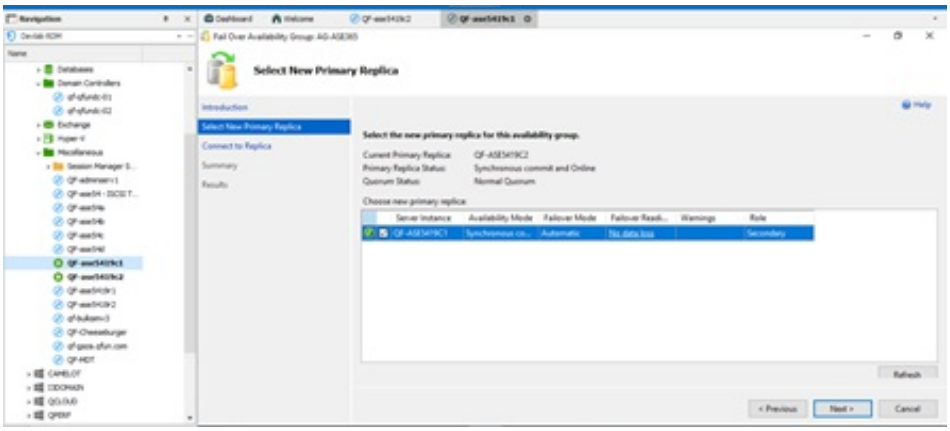
Select the **failover** tab for the failover within replica.



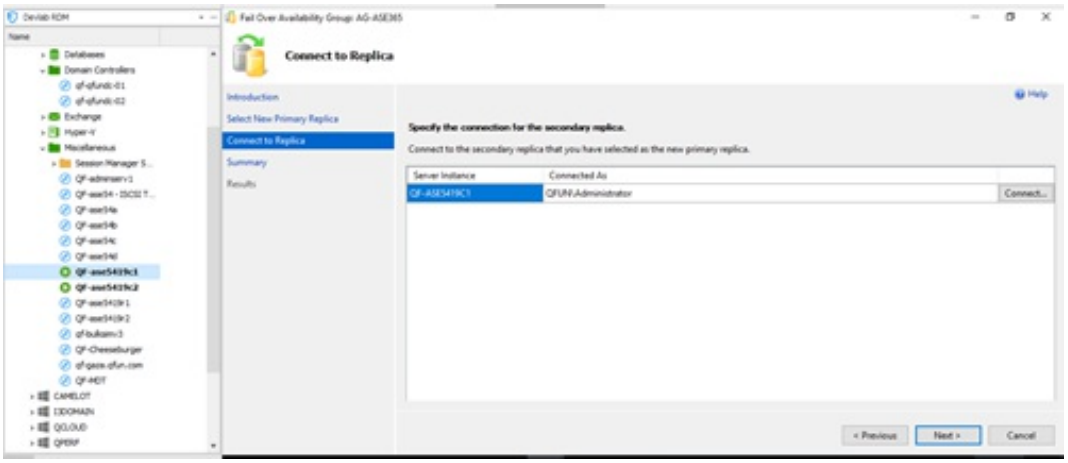
Open the failover wizard.



Select the primary replica of Availability group.

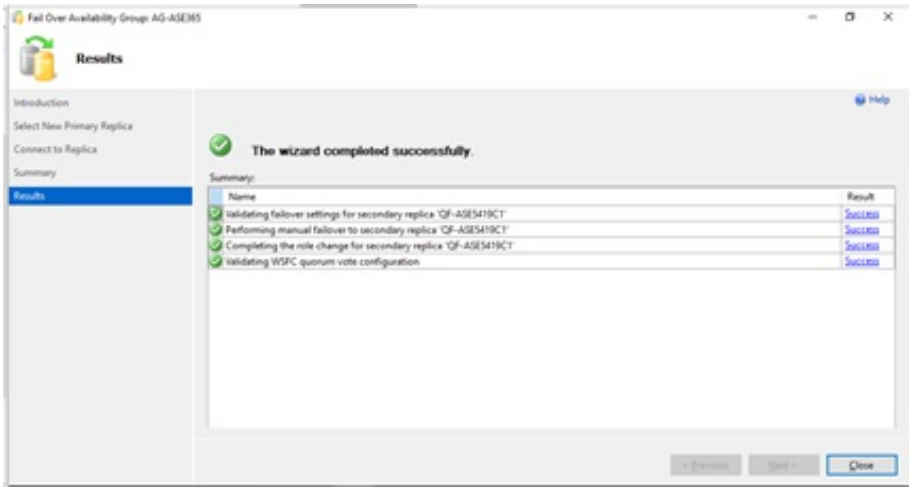


Connect to the primary replica.



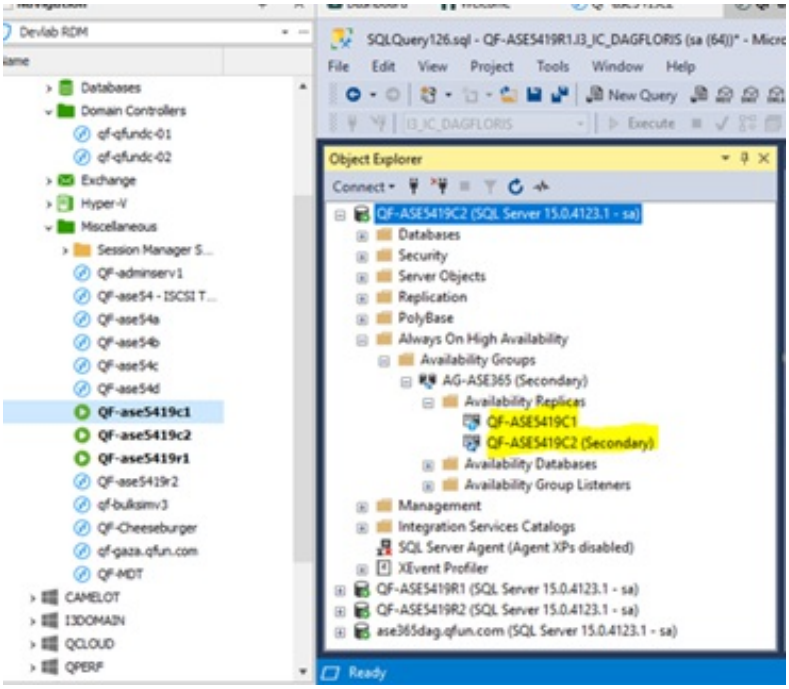
Summary of the failover wizard

Complete the failover wizard.



Status of the replicas after the failover.

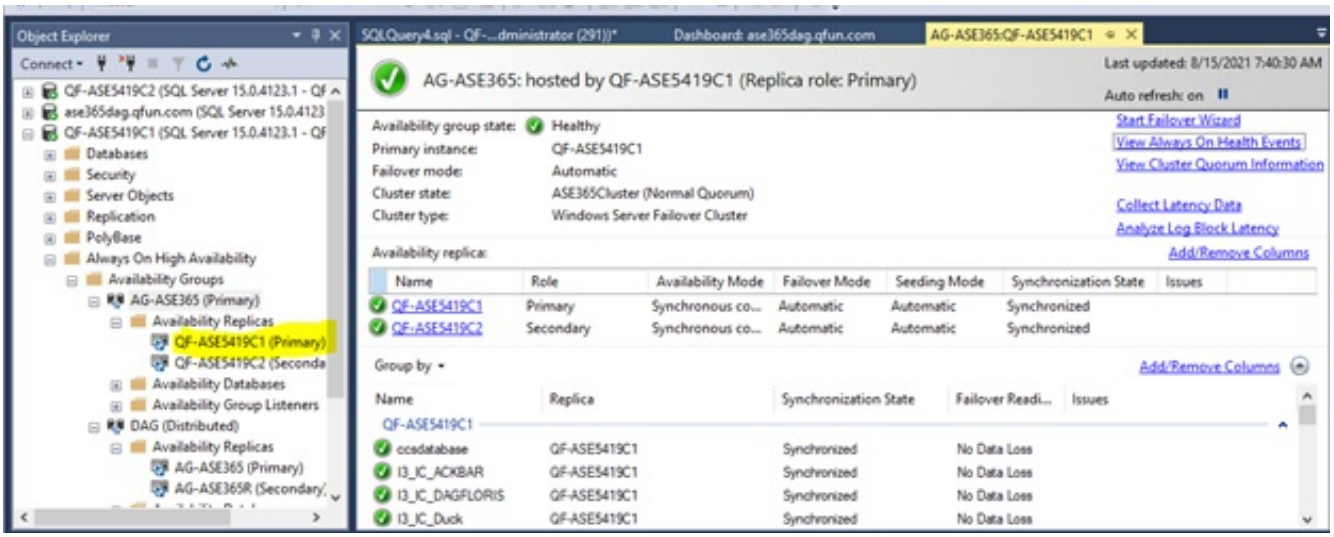
a. Distributed availability group status : AG-ASE365(Primary)
b. Replicas inside the availability group status : QF-ASE5419C2(Secondary) ,QF-ASE5419C1(Primary)



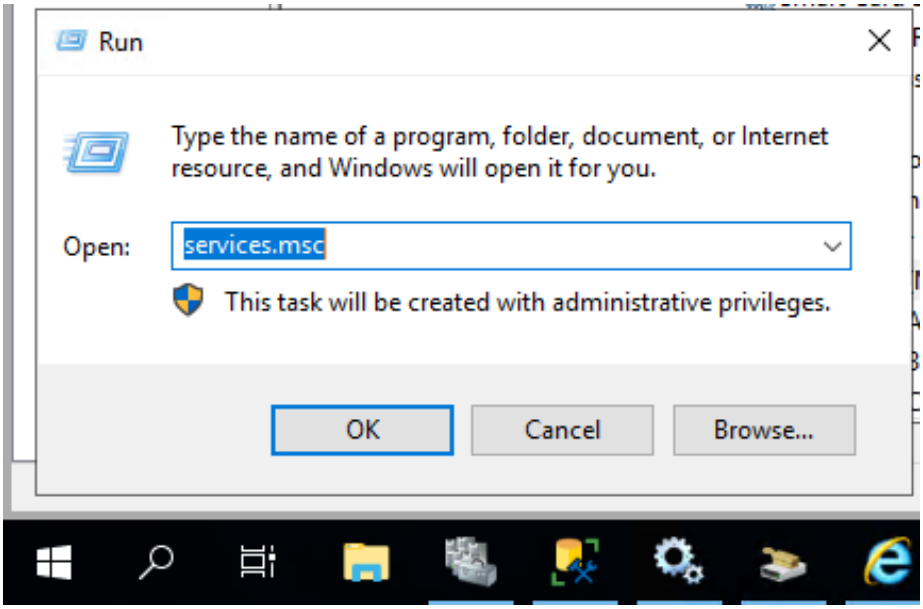
Automatic failover Mode

The present status distributed availability group and the replicas inside the availability group.

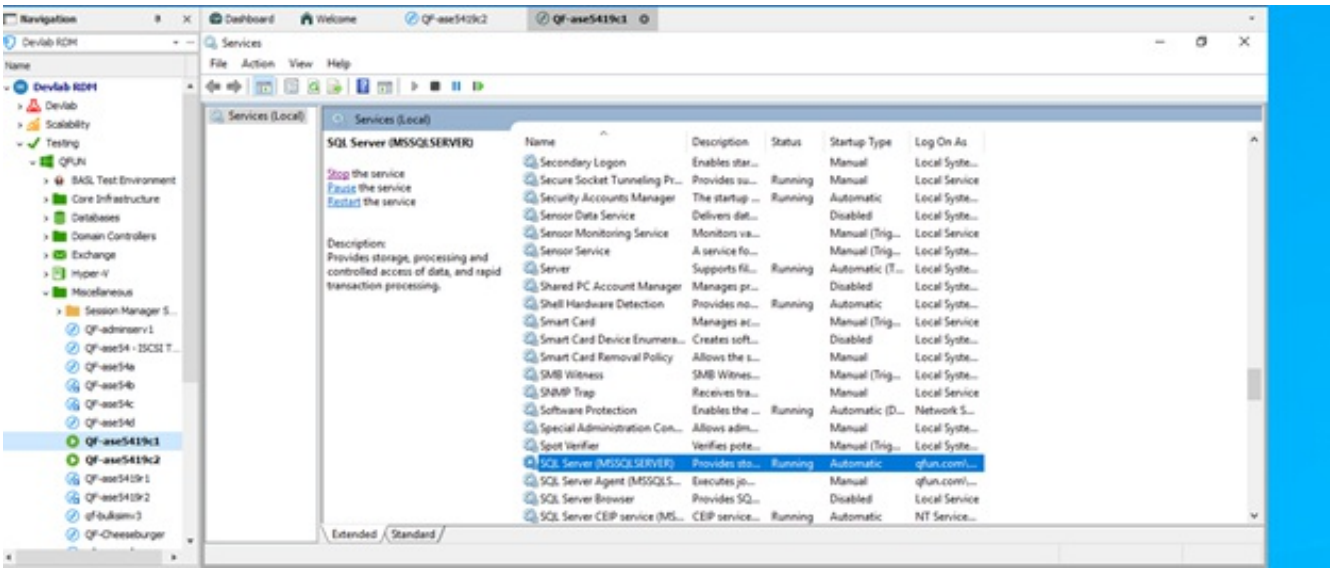
a. Distributed availability group status : AG-ASE365(Primary)
b. Replicas inside the availability group status : QF-ASE5419C1(Primary) ,QF-ASE5419C2(Secondary)



Type **services.msc** in the Run window.



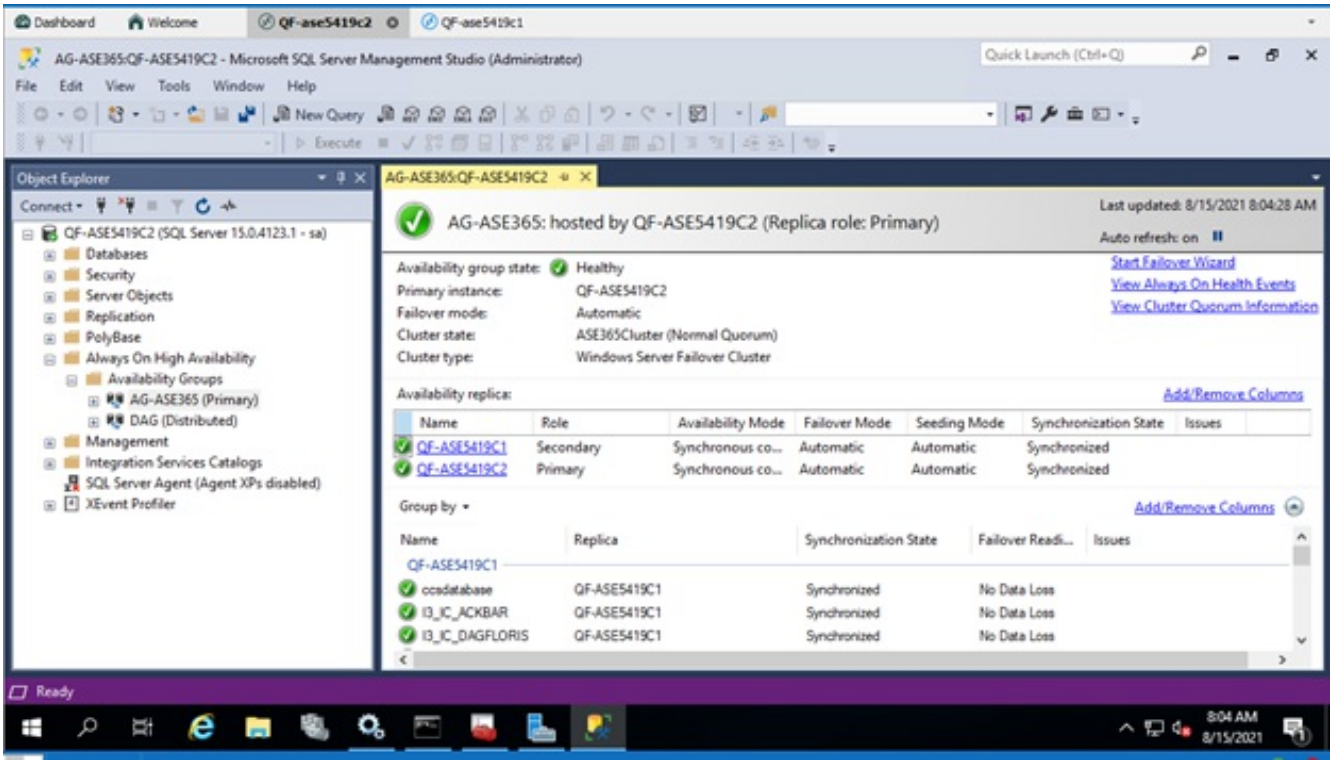
Stop and start the SQL server(MSSQLSERVER)



Status of the replicas after the failover

Distributed availability group status : AG-ASE365(Primary)

Replicas inside the availability group status : QF-ASE5419C1(Secondary) ,QF-ASE5419C2(Primary)



Fail over a distributed availability group

The URL https://www.sqlshack.com/monitor-and-failover-a-distributed-sql-server-always-on-availability-group/ provides more information on the failover a always on the availability group.

The based on the below status update of Availability Group, the failover process of the always-on Available group would be executed.

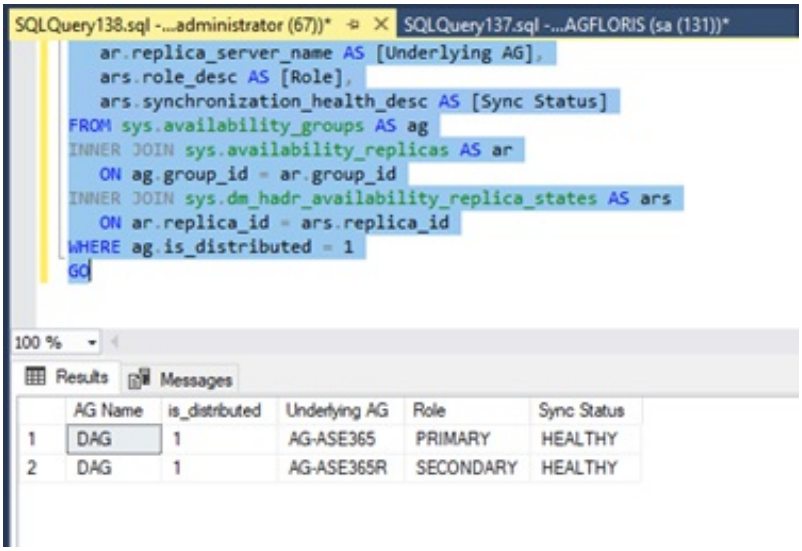
Sync status

Connection status

Synchronization health

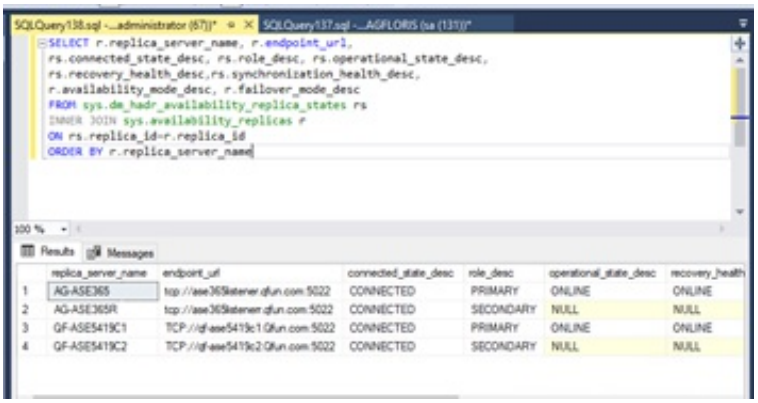
- Run the below query to check the current sync status.

```
SELECT
ag.[name] AS [AG Name],
ag.is_distributed,
ar.replica_server_name AS [Underlying AG],
ars.role_desc AS [Role],
ars.synchronization_health_desc AS [Sync Status]
FROM sys.availability_groups AS ag
INNER JOIN sys.availability_replicas AS ar
ON ag.group_id = ar.group_id
INNER JOIN sys.dm_hadr_availability_replica_states AS ars
ON ar.replica_id = ars.replica_id
WHERE ag.is_distributed = 1
GO
```



- Run the below query to connected state, synchronization health and role.

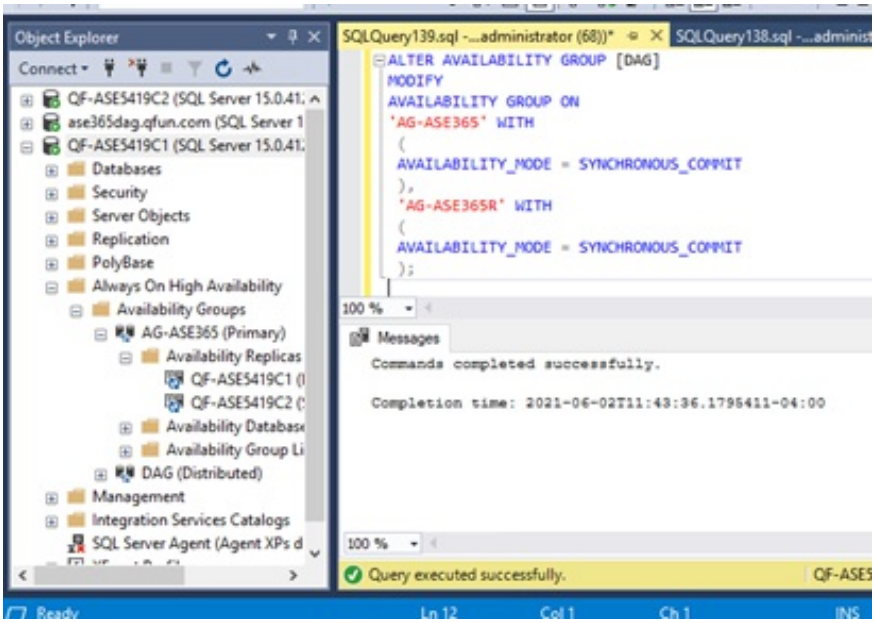
```
SELECT r.replica_server_name, r.endpoint_url,
rs.connected_state_desc, rs.role_desc, rs.operational_state_desc,
rs.recovery_health_desc,rs.synchronization_health_desc,
r.availability_mode_desc, r.failover_mode_desc
FROM sys.dm_hadr_availability_replica_states rs
INNER JOIN sys.availability_replicas r
ON rs.replica_id=r.replica_id
ORDER BY r.replica_server_name
```



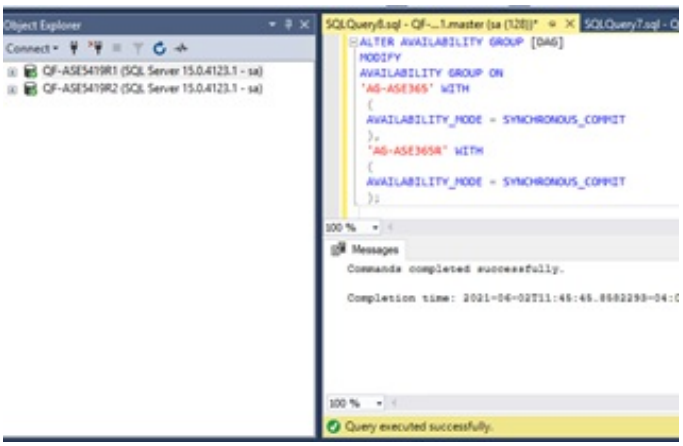
- Run the query in both primary replicas of separate clusters for the manual failover of AG.

```
ALTER AVAILABILITY GROUP [DAG]
MODIFY
AVAILABILITY GROUP ON
'AG-ASE365' WITH
(
AVAILABILITY_MODE = SYNCHRONOUS_COMMIT
),
'AG-ASE365R' WITH
(
AVAILABILITY_MODE = SYNCHRONOUS_COMMIT
);
```

- Run the Query in ASE5419C1



- Run the Query in ASE5419R1

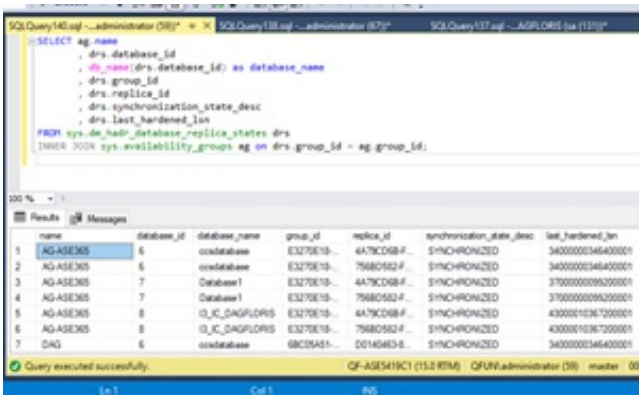


Monitor the distributed availability group. It should be in SYNCHRONIZED mode.

It might take some time for AG to be in the synchronized state depending upon the active transactions and should not proceed until AG's are fully synchronized.

- Run the query in all VMs. (ASE5419C1, ASE5419C2, ASE5419R1 and ASE5419R2)

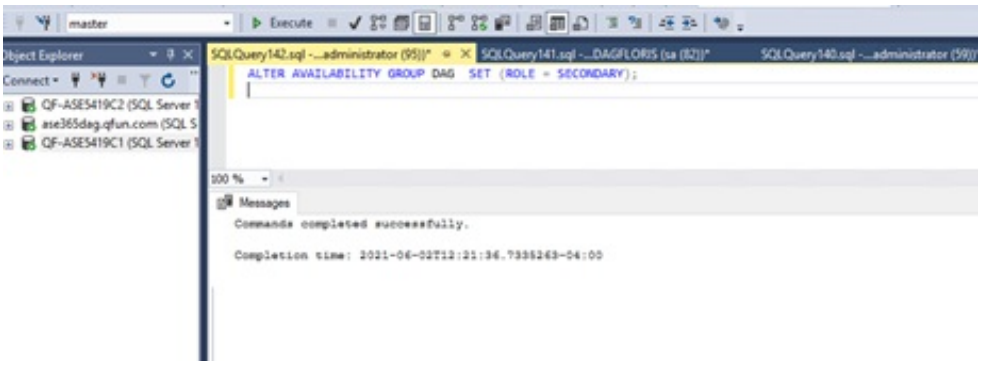
```
SELECT ag.name
, drs.database_id
, db_name(drs.database_id) as database_name
, drs.group_id
, drs.replica_id
, drs.synchronization_state_desc
, drs.last_hardened_lsn
FROM sys.dm_hadr_database_replica_states drs
INNER JOIN sys.availability_groups ag on drs.group_id = ag.group_id;
```



Run the below query to change the role of a primary replica of the primary Availability Group to secondary in a distributed availability group.

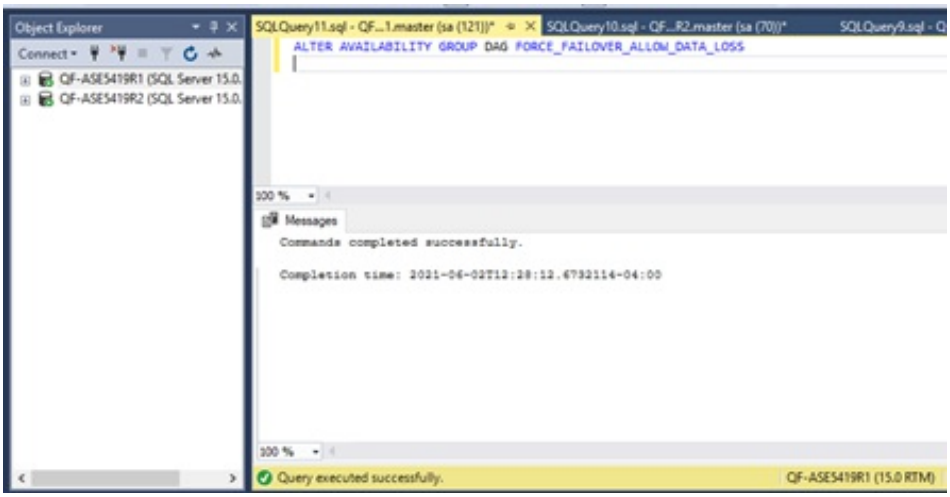
You must run this command on the primary replica of the primary cluster(global primary)

ALTER AVAILABILITY GROUP DAG SET (ROLE = SECONDARY);



Perform a manual failover with FORCE_FAILOVER_ALLOW_DATA_LOSS parameter. Run this command on the current primary replica of the secondary availability group

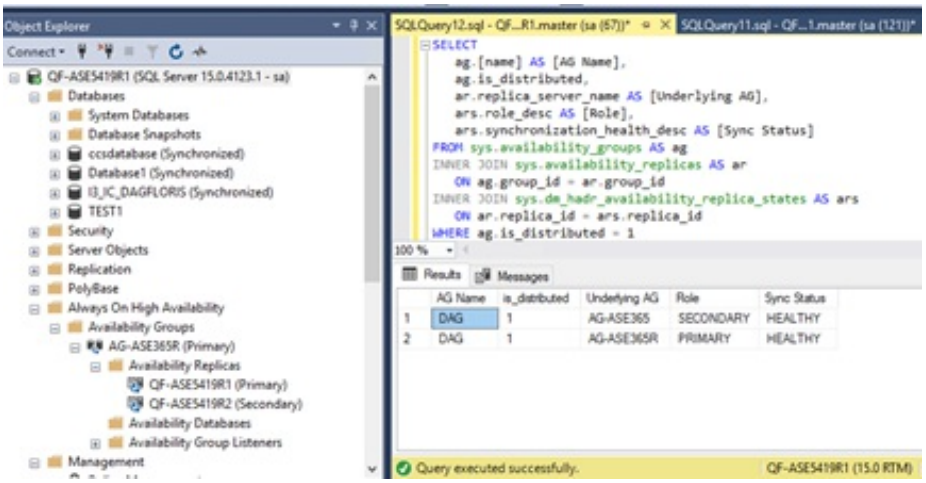
ALTER AVAILABILITY GROUP DAG FORCE_FAILOVER_ALLOW_DATA_LOSS



- To check the status of AG, run SQL query from the ASE5419 R1 primary of AG2

```
SELECT
ag.[name] AS [AG Name],
ag.is_distributed,
ar.replica_server_name AS [Underlying AG],
ars.role_desc AS [Role],
ars.synchronization_health_desc AS [Sync Status]
```

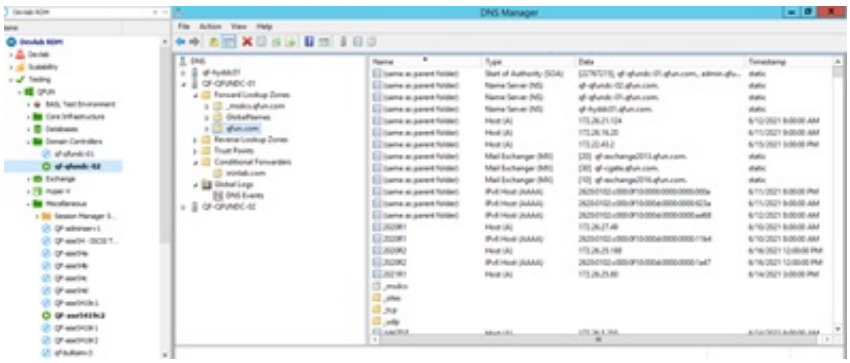
```
FROM sys.availability_groups AS ag
INNER JOIN sys.availability_replicas AS ar
ON ag.group_id = ar.group_id
INNER JOIN sys.dm_hadr_availability_replica_states AS ars
ON ar.replica_id = ars.replica_id
WHERE ag.is_distributed = 1
GO
```



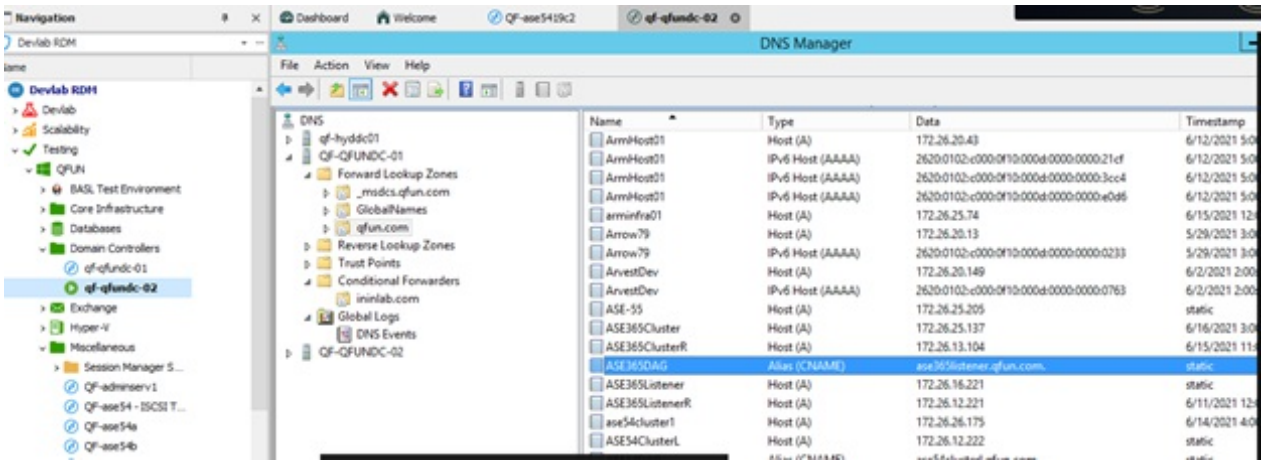
Update the fully qualified domain name(FQDN) CNAME name change

- Navigate to domain controllers in the devlab RDM and open the console qf-qfundc-02.

- Open the DNS Manager application and navigate to QF-QFUNDC-01.



- Navigate to the DNS record where the system (ASE365) CNAME need to be change for the passive Availability group failover.

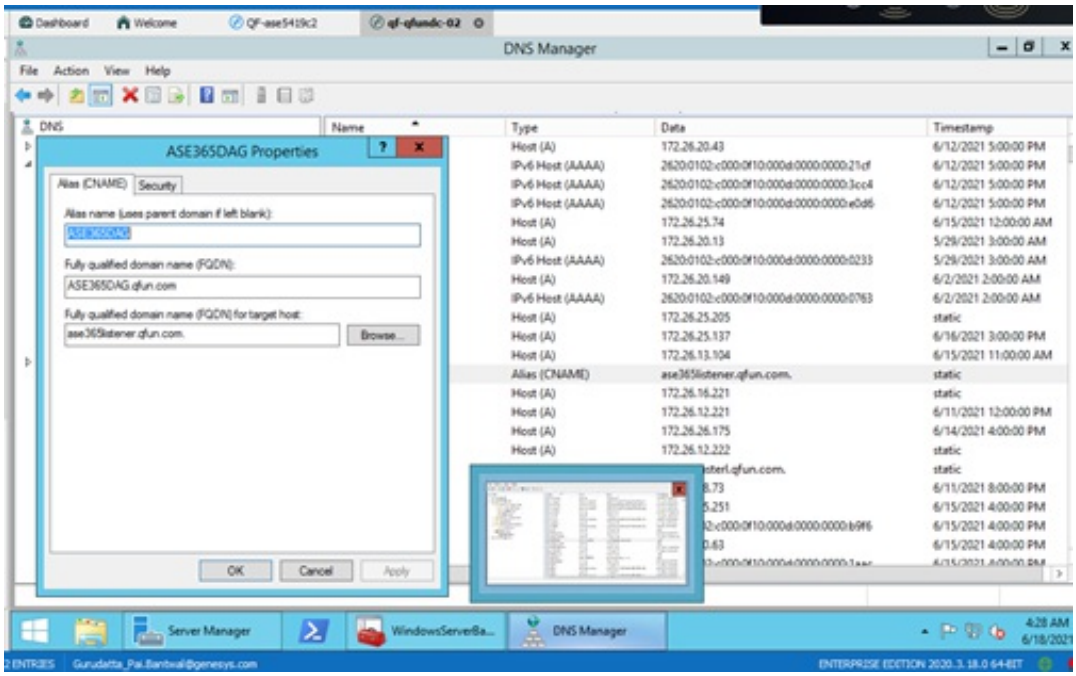


- Edit the fully qualified domain name (FQDN) for target host.

  Present name: **ase365listener.qfun.com**

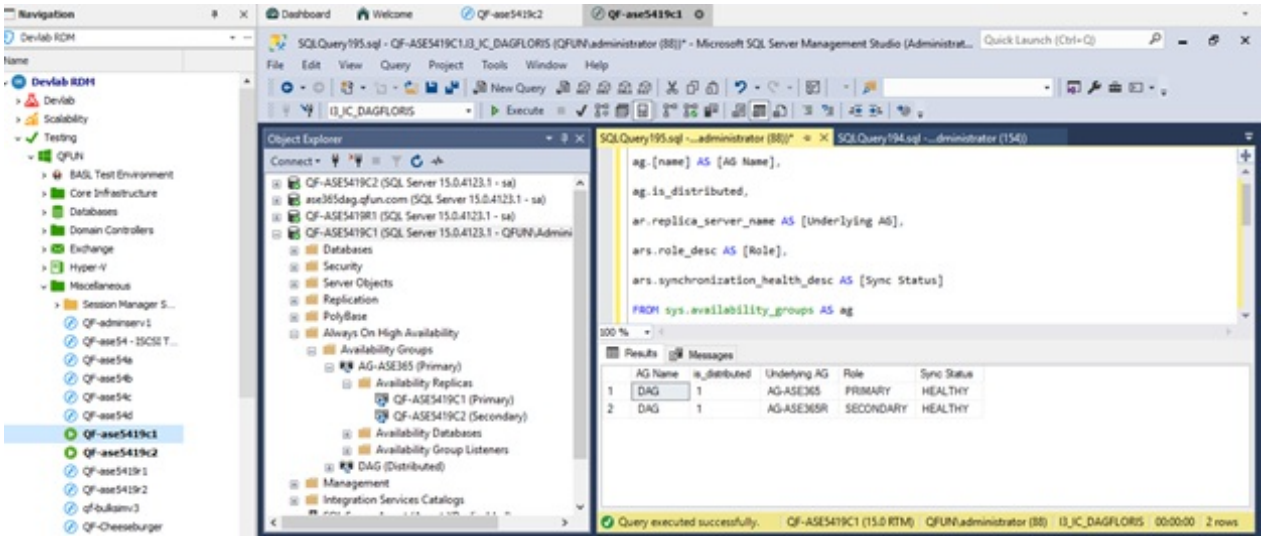
  Now update as: **ase365listenerR.qfun.com**

  Note: It takes nearly 15 minutes to propagate with respect to availability group failover.



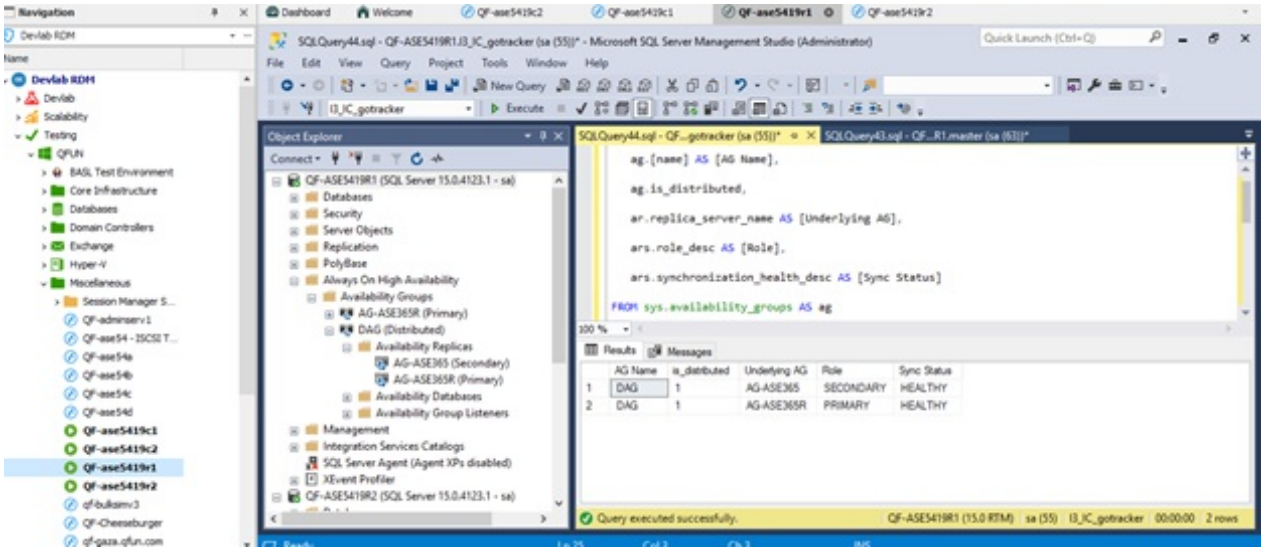
- Before failover

  AG-ASE365 primary (Availability Group)



- After the failover

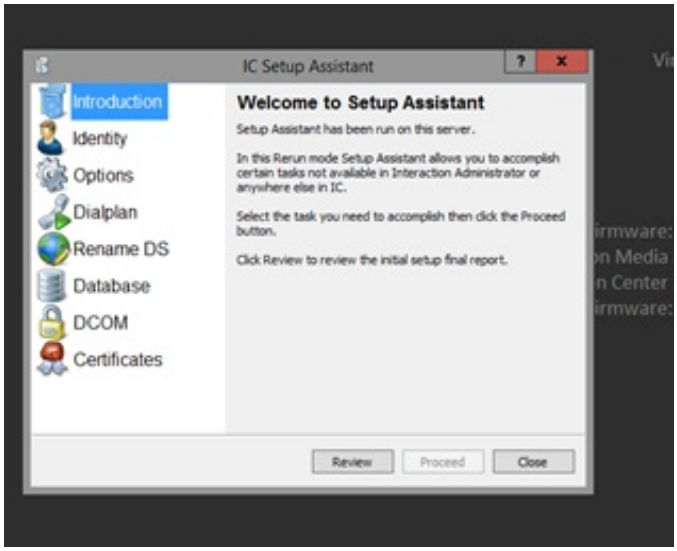
  AG-ASE365R primary (Availability Group)



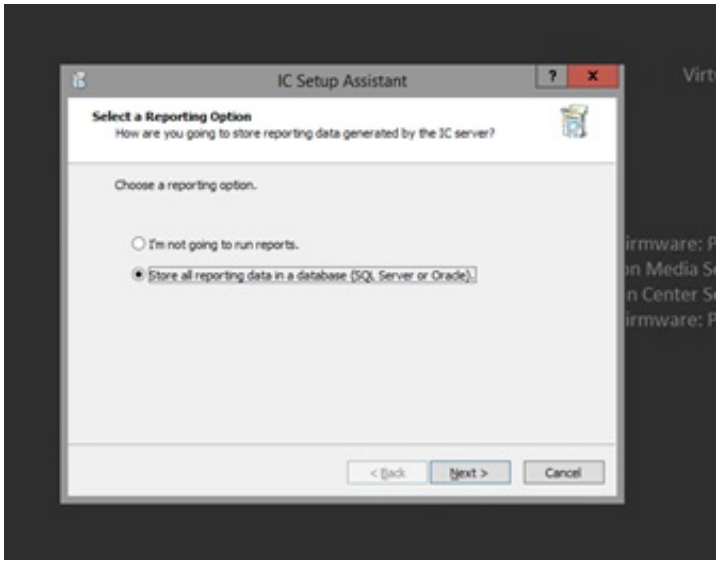
Setup Assistance with SQL Server 2019 High Availability

To create the database, do the following:

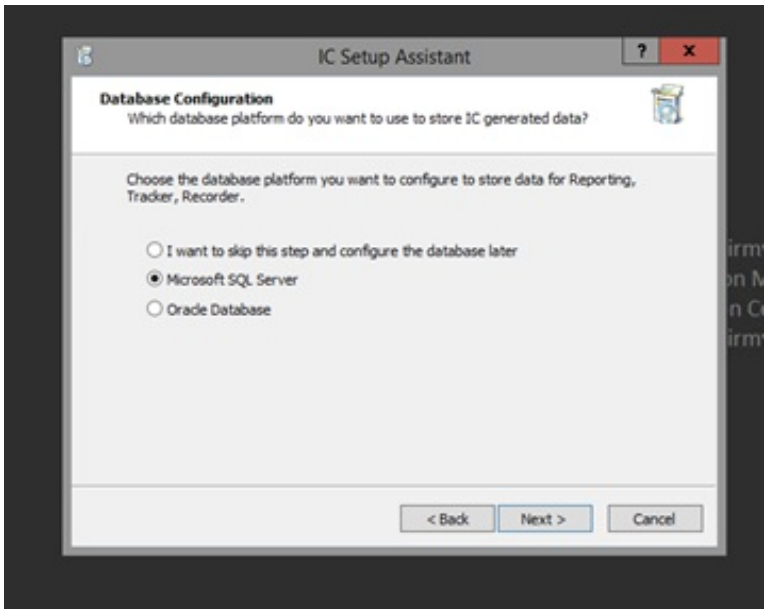
1. Run the setup assistant application.



2. Select the SQL server reporting option.



3. Select the Microsoft SQL server database configuration.

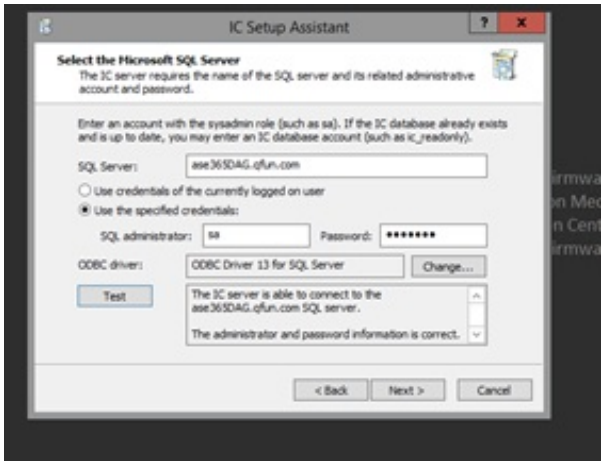


4. Edit the Microsoft SQL server name and select the ODBC driver and click **Test** button.
5. **Server name:** The name of DAG (Distributed Availability Group), which is the alias for the Primary Listener.

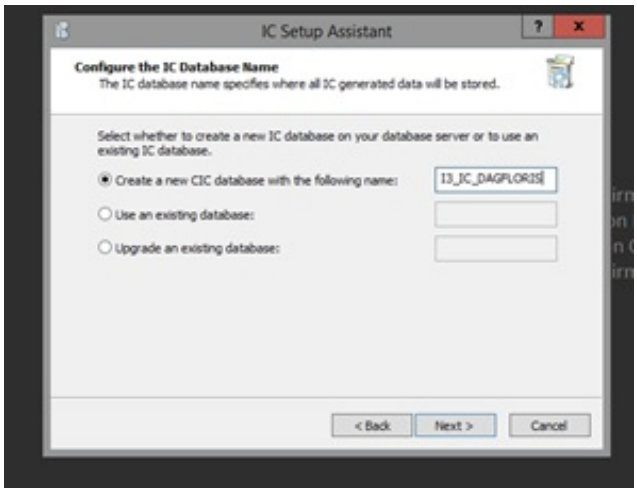
   For example, DAG: ase365DAG.qfun.com which is the alias for the primary listener: ase365listener.qfun.com
6. **ODBC driver:** ODBC driver 13 for SQL server

   Note: Download and install SQL ODBC Driver 13.1 on the PureConnect Server
7. The URL https://docs.microsoft.com/en-us/sql/connect/odbc/windows/system-requirements-installation-and-driver-files?view=sql-server-ver15 discusses the ODBC drivers that connect to SQL Server.
8. The driver name that you specify in a connection string is ODBC Driver 11 for SQL Server or ODBC Driver 13 for SQL Server (for both 13 and 13.1) or ODBC Driver 17 for SQL Server.
9. The driver is installed when you run **msodbcsql.msi** from the URL https://docs.microsoft.com/en-us/sql/connect/odbc/windows/release-notes-odbc-sql-server-windows?view=sql-server-ver15#131

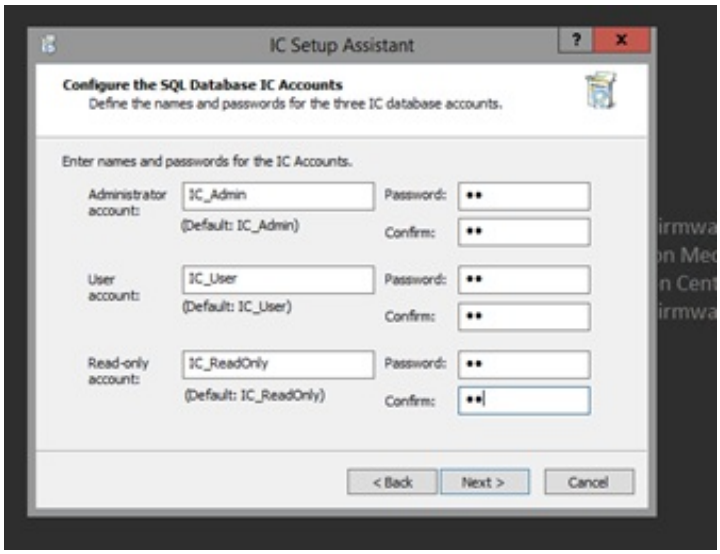
   - **SQL Server:** ase365DAG.qfun.com
   - **SQL administrator:** sa
   - **Password:** testing
10. Click **Test** to make sure it works.



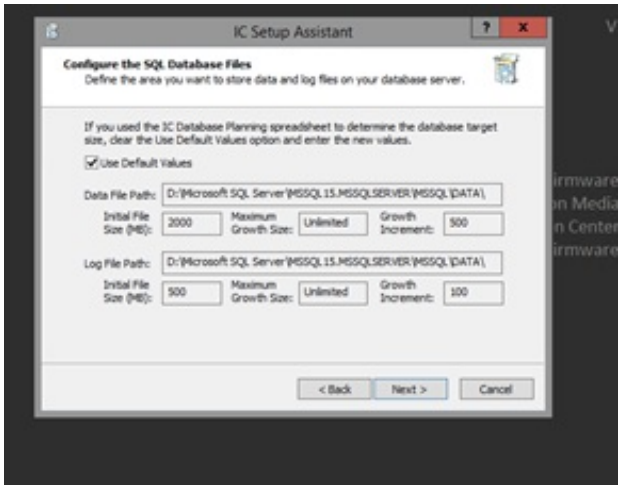
11. Edit the CIC database name. Enter name in this format **I3_IC_MachineNameWithoutPrefix**.



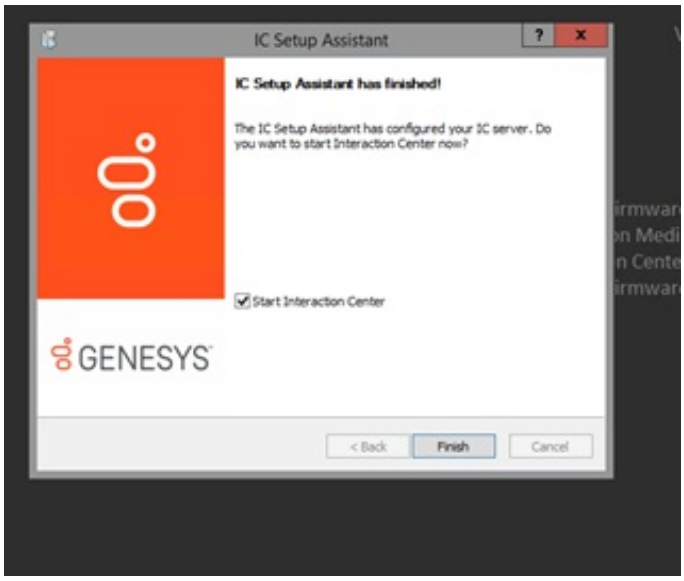
12. Enter **i3** in all **Password** and **Confirm** boxes.



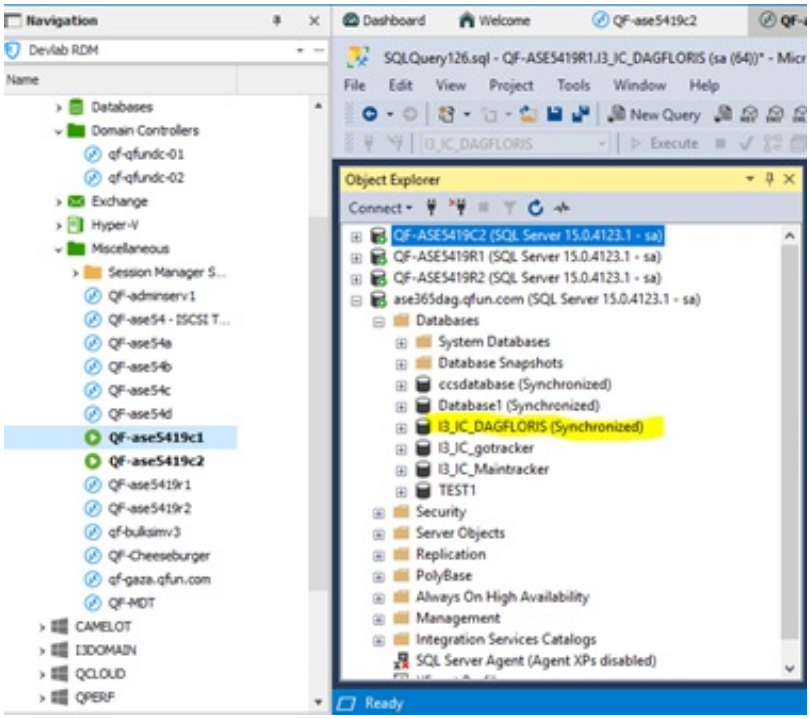
13. Configure the database files (use the default setting)



14. Click **Commit** and wait for completion followed by completion of new database creation.



15. Open the RDM and navigate to Devlab setup, open the object explorer and navigate to I3_IC_DAGFLORIS database under ase365dag.qfun.com VM tree structure.


Navigation
Devlab RDM
Name
Databases
Domain Controllers
qf-qfundc-01
qf-qfundc-02
Exchange
Hyper-V
Miscellaneous
Session Manager S...
QF-adminserv1
QF-ase54 - ISCSI T...
QF-ase54a
QF-ase54b
QF-ase54c
QF-ase54d
QF-ase5419c1
QF-ase5419c2
QF-ase5419r1
QF-ase5419r2
qf-bulksimv3
QF-Cheeseburger
qf-gaza.qfun.com
QF-MDT
CAMELOT
I3DOMAIN
QCLOUD
QPERF
Dashboard
Welcome
QF-ase5419c2
SQLQuery126.sql - QF-ASE5419R1.I3_IC_DAGFLORIS (sa (64))* - Micr
File
Edit
View
Project
Tools
Window
Help
New Query
I3_IC_DAGFLORIS
Execute
Object Explorer
Connect
QF-ASE5419C2 (SQL Server 15.0.4123.1 - sa)
QF-ASE5419R1 (SQL Server 15.0.4123.1 - sa)
QF-ASE5419R2 (SQL Server 15.0.4123.1 - sa)
ase365dag.qfun.com (SQL Server 15.0.4123.1 - sa)
Databases
System Databases
Database Snapshots
ccsdatabase (Synchronized)
Database1 (Synchronized)
I3_IC_DAGFLORIS (Synchronized)
I3_IC_gotracker
I3_IC_Maintracker
TEST1
Security
Server Objects
Replication
PolyBase
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent (Agent XPs disabled)
Ready

Database Maintenance Plans

Create a Database Maintenance Plan to maintain databases. Database integrity options include **Check database integrity** and **Include indexes**, which test the data and index page allocations in the databases for any errors. These tests are resource intensive and impair the performance of the server during the tests. These tests should only be run during off hours.

The database maintenance plan screen also has an option called **Perform these tests before backing up the database or transaction log**. This is not a good choice to make from a performance standpoint. If chosen, every time the Maintenance Plan is used to perform a database or transaction log backup–without exception–the integrity tests are automatically run. If the databases are large, and/or if frequent database or transaction log backups occur, running these tests this often can significantly degrade the server's performance. The recommended configuration consists of creating separate scheduled SQL tasks (in SQL Agent) to perform database/log backups and database integrity checks.

It is imperative to run backups, `DBCC CHECKDB`, and index rebuilds or reorgs regularly! It is equally imperative that indexes rebuilds and reorgs are done selectively. All indexes should not be rebuilt or reorged during the same job!

Third-party (Application) Access to Database

Be wary of allowing users to directly access the databases (especially OLTP databases) with third-party database access tools, such as Microsoft Excel or Access. Many of these tools can wreak havoc with database performance. Here are some reasons why:

- Often these users aren't experienced with these tools, and create overly complex queries that eat up server resources. At the other extreme, their queries may not be complex enough (such as lacking effective WHERE clauses) and return thousands, if not millions, of unnecessary rows of data.
- This reporting activity can often lock rows, pages, or tables, creating user contention for data and reducing database performance.
- These tools are often file-based. This means that even if an effective query is written, the query is not performed at the server. Instead, the entire table (or multiple tables in the case of joins) must be returned to the client software where the query is actually performed. This leads to excessive server activity, and can play havoc on your network.

If users must be allowed access to the data, limit hits on the production OLTP databases by pointing them to a "reporting" server that is replicated or in the form of a datamart or data warehouse.

For tools to help performance tune the operating system, see Sysinternals. The site has tools to defrag the server's swap file, among many others. And best of all, most are free.

Change Log

The following table lists the changes to the *CIC Database Configuration and Maintenance for SQL Server Technical Reference* since it was distributed for IC 4.0 GA.

Date	Change
01-August-2014	Updated documentation to reflect changes required in the transition from version 4.0 SU# to CIC 2015 R1, such as updates to product version numbers, system requirements, installation procedures, references to Interactive Intelligence Product Information site URLs, and copyright and trademark information.
4-September-2015	Updated documentation to reflect rebranding in CIC 2016 R1.
17-January-2017	Updated Copyright and Trademark information for 2017
17-January-2017	Updated location for the database space planning spreadsheet, in the CIC Resource Center
03-April-2017	Removed obsolete content from "Networking" topic.
11-May-2018	Rebranded from Interactive Intelligence to Genesys
11-June-2019	Reorganized the content only, which included combining some topics and deleting others that just had an introductory sentence such as, "In this section...".