AlwaysON in SQL 2022, what's new? Contained AG!

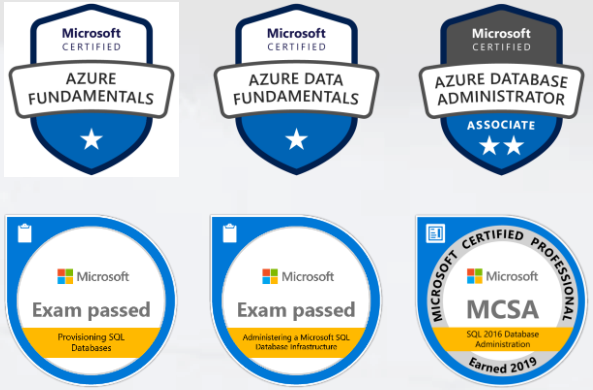
Massimiliano Buschi
mbuschi@lucient.com

SQL START! 2023

Sponsors & Organizers

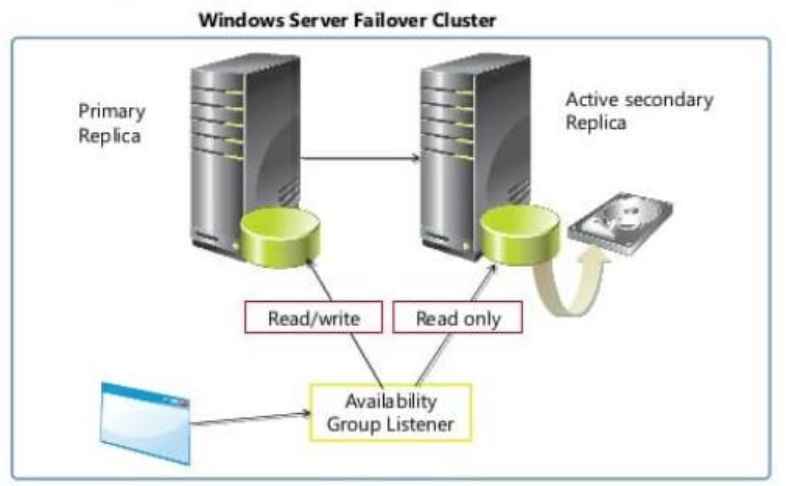
About me

- Massimiliano Buschi mbuschi@lucient.com
- Data Platform Architect @ Lucient Italia One4 Group
- 15 + years of work with SQL Server
- SQL & Data Saturday lurker
- DBA Stack Exchange Community Member



Always On availability groups: a high-availability and disaster-recovery solution

- **High-availability** and **disaster-recovery solution (AG)**
- Introduced in SQL Server 2012
- Maximizes the **availability** of **a set of user databases**
- Supports a set of read-write primary databases and one to eight sets of corresponding secondary databases
- AlwayOn AG is **not** a Backup Solution (like true HA&DR solutions)
- Requires a Windows Server Failover Cluster (**WSFC**)
  - a cluster role is created for every availability group that you create



Always On availability groups: benefits

- Supports **up to nine** availability replicas
- 2 replication modes:
  - **Asynchronous**-commit
  - **Synchronous**-commit (5 replicas since 2019, before only 3)
- Several forms of availability-group failovers:
  - **Automatic** failover
  - **Planned** manual
  - **Forced** manual
- You can configure a given availability replica to support the following active-**secondary capabilities**:
  - **Read-only** connections
  - **Backup** operations
- Supports an availability group **listener** for each availability group
- Supports **automatic page repair** for protection against page corruption
- Supports **encryption** and **compression**


STANDARD

Always On availability groups: can we have them in SQL Server Standard Edition?

- Sadly, NO or better you can have a simpler version called **Basic Availability group (BAG)**

- These are the **differences**:
  - **Only two replicas**
    - The primary and the secondary replica
  - You can **use only the primary replica**
  - Support for one availability database
    - the ‘group’ can be composed by only **a single database**

Always On availability groups: how to manage them?

- We have a set of tools to simplify deployment and management of availability groups, including:
  - Transact-SQL DDL statements
  - **SQL Server Management Studio** (for SQL server 2022 use SSMS 19.0.1)
    - The New Availability Group Wizard
    - The Add Database to Availability Group Wizard
    - The Add Replica to Availability Group Wizard
    - The Fail Over Availability Group Wizard
  - The Always On Dashboard
  - PowerShell and DBATools cmdlets

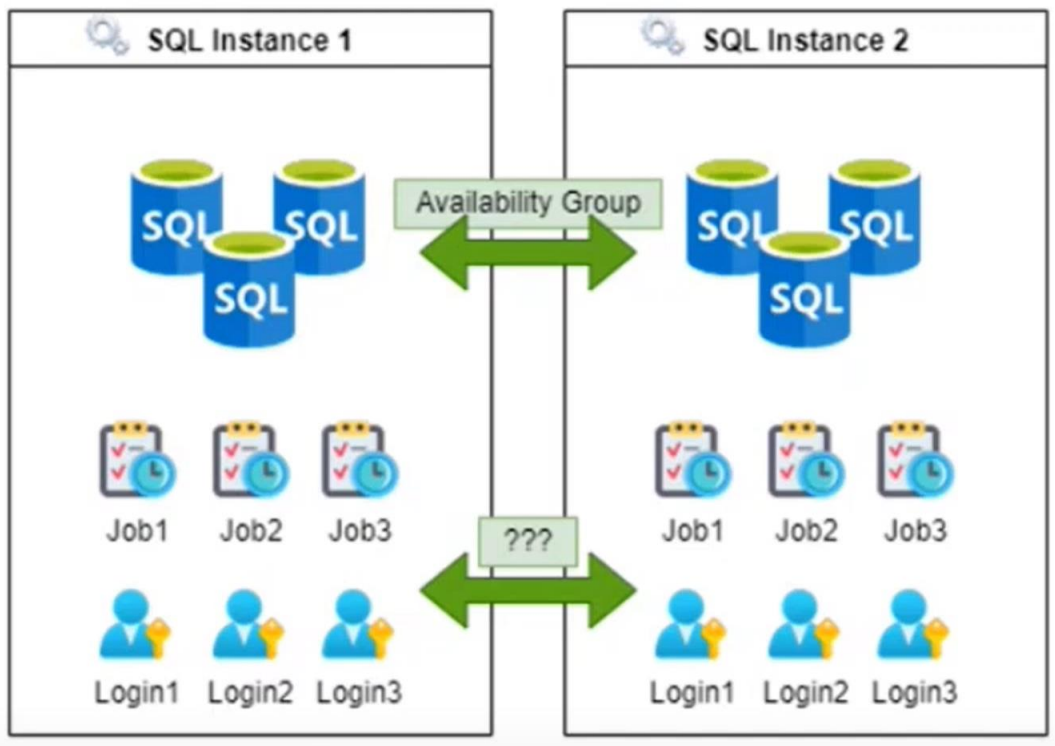
Always On availability groups: client connectivity with availability group listener

- An availability group **listener** is **a server name** (and an IP number) to which **clients can connect** in order to access a database in a primary or secondary replica of an Always On availability group.
- Availability group listeners **direct** incoming connections **to the primary** replica or **to a read-only secondary** replica.
- The listener provides **fast application failover** after an availability group fails over.
- They rely on **failover cluster services role switch** to work

Alwayson AG DEMO1

- AG Setup
- Failover
- Connectivity to listener

Alwayson AG: before sql 2022



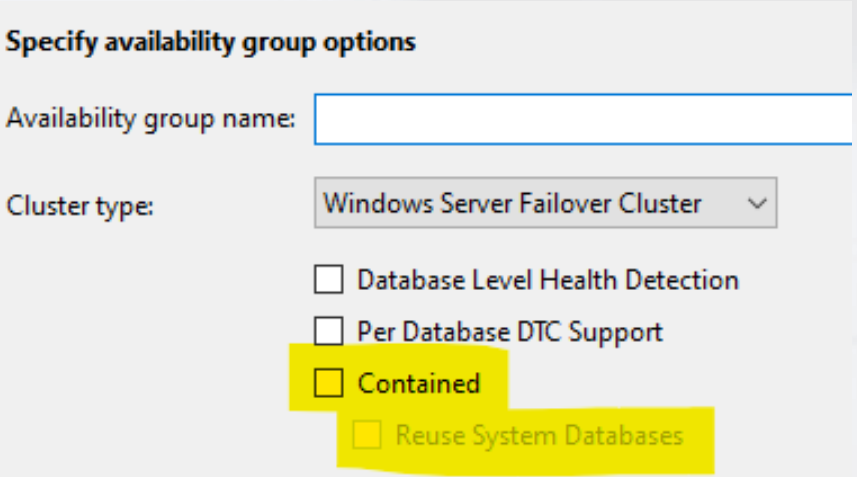
- We have some issues
  - A few types of objects are outside AG replication as the are instance level objects
  - Login
  - Jobs
  - Linked server
  - Anything in the master database
  - Anything in the msdb database
    - Alerts
    - Operators

Always On contained availability groups

- A **contained availability group (CAG)** is an Always On availability group that supports:
  - **managing metadata** objects (logins, permissions, SQL Agent jobs etc.) **at the availability group level in addition to the instance level**.
  - specialized **contained system databases within the availability group**.

```
SQL                                                    Copy

<with_option_spec>::=
CONTAINED |
REUSE_SYSTEM_DATABASES
```



Always On contained availability groups: after creation

During CAG creation sysadmin logins at instance level are copied to the contained ag master database

- And other logins?
  - **They are not copied**
- What will happen next ?
  - **NOTHING will be copied automatically.**

Always On contained availability groups: differences

Differences between connecting to the instance and connecting to the contained availability group:

- When connected to contained AG, **users will only see databases in the contained AG, plus tempdb**. 👍
- At instance level, contained AG master and msdb names will be **[AGName]_master**, and **[AGName]_msdb**. Inside contained AG, their names are master and msdb. 👍
- **Database ID for contained AG master is 1** from inside contained AG, but something else when connected to the instance. 👍
- While **users will not see databases outside of the contained AG** in sys.databases when connected in a contained AG connection, **they will be able to access those databases by three part name** or through the use command. 👎
- Server configuration through sp_configure can be read from contained AG connection but **can only be written from instance level**. 👍
- From contained AG connections, **sysadmin is able to perform instance level operations**, such as shutting down SQL Server. 👎
- Most DB level, end point level, or AG level operations **can only be performed from instance connections**, not contained AG connections. 👍

Alwayson AG DEMO2

- Contained AG Setup
- Failover
- Connectivity to listener
  - Differences
    - Logins
    - Jobs
    - Linked server

Always On contained availability groups: recap

Good things:

- **You don't need to keep login/jobs in sync** between nodes and manage job running on secondaries
- **Filtered database list in SSMS** gives and idea of AG boundaries.
- Query store for secondary replicas (preview)

Things that need improvements:

- TDE: you have to **manually install the DMK** into the contained master databases

Things **not** currently **supported**:

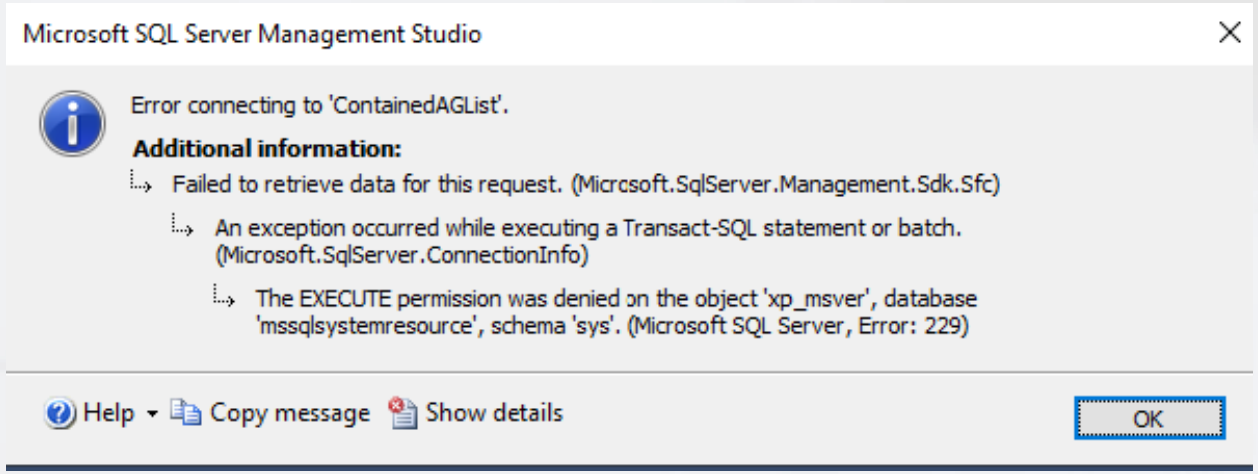
- SQL Server Replication
- Distributed AG
- Log shipping where the target database is in the contained availability group

Things to pay attention to:

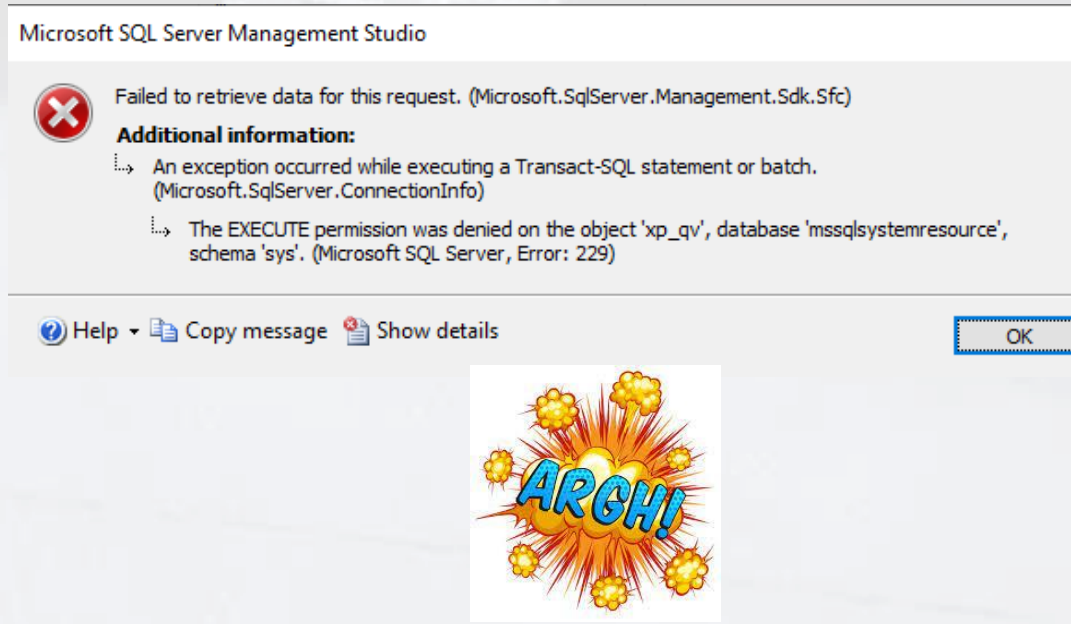
- Define a **contained objects naming convention**; something like **agname_jobname**, **agname_loginname** etc.. as the contained system databases.
- Remember to administer contained objects connecting to the listener only, **anything else connecting only to instances**
- **Do not rely** on the possibility of **querying databases outside the ag**, especially for data. If you need them, move them to the group or merge the two groups.
- No instance level object are synchronized between contained ag 'world' and instance 'world' after creation. **You have to do it manually**.

Last minute errors!

- The execute permission was denied on the object xp_msver. Database mssqlsystemresource, schema sys. (Microsoft SQL Server, Error 229)




```
USE master;
GO
GRANT EXECUTE ON sys.xp_msver TO [Public]
GO
```



Thank You All

May the forced plan be with you