# DBAChecks Service

## Setup

* Build solution
* Deploy database from dacpac (DBAChecksDB project)
* Copy DBAChecksService binaries from bin folder to server where you want to run the agent on.
* Run DBAChecksServiceConfig app.
* Set a destination (SQL connection string to the DBAChecksDB deployed earlier)
* Click source tab. Add source connections you want to monitor
* Click “Service Tab”. Click “Install”. Enter an account to run as.
* Click “Start” to start the service.
* Deploy DBAChecksReports project to a SSRS instance for reporting and configure data sources.

## Upgrade

* Deploy database changes from dacpac
* Stop DBAChecks agent
* Replace DBAChecks agent binaries

Notes:  
Multiple agents can be used.

The destination can be set to a S3 bucket if there isn’t direct access to the DBAChecks database. In this case another agent with access to the DBAChecksDB would have the source set to this S3 Bucket location. Credentials can optionally be specified in the AWSCredentials tab.

Passwords (Connection string and AWS secret key) are encrypted, but consider this a layer of obfuscation rather than security. Integrated authentication is recommended.

## Architecture

DBAChecks project is where the interesting stuff happens. Collect and Import are two separate operations. The collection part is handled by the DBCollector class and this returns a dataset with the data collected. There are a number of defined collection types which you can use to define what to collect. Normally these will be “General” or “Performance” which are categories that contain queries that should be run frequently (performance) or hourly (general).

The DataSet would then be passed to DBImporter class for import into DBAChecksDB. Or the DataSet could be serialized and copied to a S3 bucket or local path. These serialized datasets can then be de-serialized and imported into the DBAChecksDB by another agent.

To add a new collection, the query is added as a resource in the DBAChecks project. The collection type is added to the CollectionType enum in DBCollector (Use the name of the resource minus SQL prefix). Configure CollectionType.General or CollectionType.Performance to collect this new type. Unless the type requires any special handling, additional code changes are not required for collection. The DBImporter class needs updating to include the new table (Any additional tables not in the list configured will be ignored). A stored procedure needs to be created to handle the import as well as a user defined table type to match the data collected.

The DBAChecksService project is what produces the binaries for the agent. This requires a ServiceConfig.json file which you configure using the DBAChecksServiceConfig tool. The DBAChecksService uses Quartz scheduler to setup collections for each source at specified intervals.