ControlPoint

Software Version 5.9.0

Database Conversion Guide

Document Release Date: August 2020
Software Release Date: August 2020

Legal notices

Copyright notice

The only warranties for products and services of Micro Focus and its affiliates and licensors (“Micro Focus”) are as may be set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Documentation updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for updated documentation, visit https://www.microfocus.com/support-and-services/documentation/.

Support

Visit the MySupport portal to access contact information and details about the products, services, and support that Micro Focus offers.

This portal also provides customer self-solve capabilities. It gives you a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the MySupport portal to:

- Search for knowledge documents of interest
- Access product documentation
- View software vulnerability alerts
- Enter into discussions with other software customers
- Download software patches
- Manage software licenses, downloads, and support contracts
- Submit and track service requests
- Contact customer support
- View information about all services that Support offers

Many areas of the portal require you to sign in. If you need an account, you can create one when prompted to sign in. To learn about the different access levels the portal uses, see the Access Levels descriptions.

Contents

Chapter 1: Overview

Over the lifetime and growth of a ControlPoint system, particularly at the large enterprise level, overall database resource utilization and performance degrades as the quantity of managed data continues to grow. The ControlPoint database conversion package provides a set of SQL scripts to improve scalability at high volumes by decreasing the rate of degradation. It also adds mechanisms to all ControlPoint databases to enable support for advanced database maintenance.

Specifically, the conversion package provides the following benefits to all ControlPoint databases, regardless of size:

- **Reduces the size of the ControlPointMetaStore database**

  This is one of the largest of the ControlPoint databases. The reduction, in particular to the index component, results in an up to 33% reduction in total database size. In addition, the new storage structure of the databases is in smaller, more manageable files. This allows a systems operator to make use of smaller, more independent logical volumes.

- **Reduces the storage throughput required for ControlPoint operations**

  This is achieved by taking advantage of the concurrent storage channels/volumes usually available to production servers. This results in an up to 66% reduction in required storage throughput for a given workload.

- **Separates the structure of the database storage into multiple discrete files**

  This separation allows you to more accurately monitor your server for I/O hotspots while under load and to easily relocate component files to additional volumes. It also allows you to preserve a standard logical internal structure and it facilitates future upgrades, even if you performed custom reorganization of the storage files.

- **Reduces SQL Server memory utilization**

  This achieves up to 90% reduction in downward memory pressure under load.

- **Adds SQL table and index partitioning**

  This provides some small (< 10%) query performance improvement in specific query types. However, the major gain in this area is a reduction in necessary SQL index maintenance windows; allowing for more processing hours in a given day.

  By using the conversion package, you can optimize the databases. Although the upgrade process creates filegroups across the filesystem to ensure new data will be adequately distributed, it does not evenly redistribute existing data to these filegroups. Nor does it convert objects to use SQL partitioning when appropriate. The conversion process moves and redistributes the ControlPoint table and index data from its original storage to its optimal placement. To accomplish this, the conversion package does the following:

  1. Drops all indexes and views, etc. on the database objects
  2. Moves the data for all tables to backup tables
  3. Creates replacement tables on the appropriate filegroups

4. Intelligently distributes the data back into the replacement objects
5. Recreates the indexes and other data dependent objects on the appropriate filegroups

It is in steps 4 and 5 that the package uses SQL partitioning to evenly segments objects to files in the filegroups and across the configured partitions appropriate for the data.

Database naming conventions

By default, the database names in your ControlPoint environment conform to the following naming conventions:

- ControlPoint
- ControlPointAudit
- ControlPointMetaStore
- ControlPointMetaStoreTags
- ControlPointTracking

Requirements

The following details the requirements, in hardware, software and environment resources and in user impact or downtime, necessary to complete the execution of this conversion package.

SQL Server software

One of the versions of SQL Server that ControlPoint supports. Both the Standard and Enterprise edition of all supported versions have SQL partitioning, a feature that the conversion package requires. For the list of supported versions, see the *ControlPoint Support Matrix*.

SQL Server Management Studio

This process requires access to SQL Server Management Studio either on the server itself (recommended) or remotely from a Windows workstation.

> **IMPORTANT:** Do not use Powershell or SQLCMD to complete the tasks in this document.

Permissions

- **SA** or another SQL user account with the **sysadmin** server role assigned, must be used for the conversion.

In addition, this account (and the account being used to execute the MS SQL Server process) must have the ability to read and write files to the local server disk storage.

This includes directories chosen for placement of new files and file groups as well as a temporary location to be used for MS SQL Bulk Copy (BCP) workspace.

Bulk Copy (BCP) workspace

IMPORTANT: The location chosen for the BCP workspace must be of sufficient size to hold the entire data portion (in text form) of any single table. This workspace is emptied during the execution and may be removed after conversion.

You can calculate the maximum size necessary by choosing the widest table, summing the maximum byte count for the column types, and then multiplying by the number of rows. The information is simplified in the following example (see TSQL example: BCP expected size calculation, on page 17 for an example of how to perform this calculation).

Example: ControlPointMetaStore database, Metadata.Document table

Average size per row	2,752 bytes
Current row count	347,641,154
Necessary temporary BCP workspace to convert this database:	892 GB

Sufficient additional SQL disk storage resources up to 100% of the current database's data portion size will be utilized in the final production configuration.

Micro Focus recommends that the additional space reside on the additional, smaller performant logical volumes targeted for converted database storage.

This information can be retrieved by SQL Server Management Studio (SSMS) by running the `sp_spaceused` command. This space can be returned to the OS (and then removed) after the database conversion is run and data integrity is verified.

Example: ControlPointMetaStore database

database_name	ControlPointMetaStore
database_size	1,007,723.32 MB
unallocated space	14,598.63 MB
reserved	1,016,566,592 KB
data	470,683,552 KB
index_size	545,817,792 KB
unused	65,248 KB

In this example, an additional 450 GB of storage should be available for use during the conversion process.

User impact and downtime

The ControlPoint environment must be completely offline for the duration of this conversion.

CAUTION: All ControlPoint and Micro Focus IDOL services must be stopped before proceeding with the database conversion steps.

Failure to stop all services in your ControlPoint environment may result in failure to successfully convert the database structures using the scripts.

Component processes should be cleanly stopped and disabled in the Windows Services applet. This includes:

- All ControlPoint connectors
- All ControlPoint related Windows processes.
- IIS and W3SVC hosting the ControlPoint user interface

IMPORTANT: When calculating the necessary downtime, include all steps of the conversion process, including:

- The initial database backup
- The conversion itself
- The verification of converted data
- The cleanup of backed up data
- The initial execution of the maintenance task.

For large databases, this will be significant. For more information, see Conversion reference, on page 18.

Chapter 2: Convert the databases

This section details the steps required to convert the ControlPoint databases.

For enterprise customers with databases of significant size, run the following tasks on a user acceptance test environment (UAT) that closely matches the production equipment's configuration and capacity.

Most anticipated errors encountered during the process are safely handled or recoverable from, with the investment of extra time. Every attempt is provided to allow administrators to estimate the time and resources required. Micro Focus strongly recommends performing the complete process as a test run with your individual data.

IMPORTANT: Before starting the conversion process, make note of the following points:

- The conversion process requires significant time, resources, knowledge, and planning to complete successfully; it must be performed manually. For more information, see Requirements, on page 5.
- You must perform all tasks to completion.

Prepare for conversions

Before converting each database, you must prepare the environment for the conversion.

To prepare for conversions:

1. On the SQL Server machine, ensure all directories planned for use (for new data files, for BCP, etc.) have the appropriate permissions. This includes:
   a. Set read, write, and change permissions on each directory for the account being used to operate the SQL Server process.

      This is usually either `NT Service\MSSQLSERVER` or `SYSTEM`, but may be a different user account. Check the Services control panel to identify this.

   b. If you are connecting to SQL Server with a non-SQL user account (for example, a Windows account), ensure this account also has read, write, and change access to the new data file directories and the BCP workspace.
2. Allow any executing policy phases to complete.

   NOTE: Ensure all items in the existing policies are in the `executed` or `failed` status, before the upgrade.
3. In the ControlPoint Administration dashboard, disable the Assign Policies and Execute Policies

scheduled tasks using the Scheduled Tasks settings. This prevents new policies from being assigned to documents.

> **NOTE:** Be sure to disable all of the scheduled tasks: Normal, Low and High priority.

4. Ensure that all ingestion jobs are complete.

   > **NOTE:** If ingestion jobs are still running, wait for them to complete before proceeding.

5. Stop all ControlPoint-related processes on all servers that make up the environment.

   > **CAUTION:** All ControlPoint and IDOL services must be stopped before proceeding with the database conversion steps.
   >
   > Failure to stop all services in your ControlPoint environment may result in failure to successfully convert the database structures using the scripts.

   - MetaStore service
   - IDOL service
   - OGS service
   - DataAnalysis service
   - Engine service
   - License Server service
   - License Service service
   - Distributed Connector
   - Individual connectors and Connector Framework Services
   - IIS

6. Create full backups of all ControlPoint databases.

   > **NOTE:** Ensure that you have sufficient storage space for the database backups.

   > **IMPORTANT:** This is a critical step and serves as the necessary safety net for recovery in case of failure.

Convert the ControlPoint database

This section provides the specific steps to convert the component database, named **ControlPoint** by default.

To convert the ControlPoint database

1. In SQL Server Management Studio, connect to the SQL Server instance as SA, or equivalent.
2. Select the **ControlPoint** database.
3. Open the `\ControlPoint\01_Convert_ControlPoint_v1.sql` script.
4. Find and replace all instances of the following placeholders in the script as indicated:
   a. **`<sever_db_name>`**. Name of the database, typically **ControlPoint**.
   b. **`<bcpPath>`**. Storage location with high I/O performance and adequate space for large table data transfers.
   c. **`<username>`**. SQL user with select and insert permissions to all objects in this database.
   d. **`<password>`**. Plain-text password of the specified SQL user.
5. (Optional) Adjust the value of the `@largeRowCount` parameter.

   This parameter specifies the maximum number of rows in a table before the BCP backup and copy method is utilized. Default: 20,000,000.
6. Execute the modified script. Progress and results of the operation are returned in the **Messages** tab.

   > **TIP:** To preserve the record of this execution, save this content to a text file.
7. Open the `\ControlPoint\02_ControlPoint_ConversionCleanup.sql` script.
8. Find and replace all instances of the `<server_db_name>` placeholder with the name of the ControlPoint database (typically **ControlPoint**).

   > **CAUTION:** This script is data destructive. If row count verification tests pass, it permanently removes the backup tables created during the conversion process.
   >
   > If disk storage space is not immediately required to be returned to the OS, execution of this script may be delayed until full conversion and application function verification is complete.
9. Execute the modified script. Progress and results of the operation are returned in the **Messages** tab.

Convert the ControlPointAudit database

This section provides the specific steps to convert the component database, named **ControlPointAudit** by default.

To convert ControlPointAudit database

1. In SQL Server Management Studio, connect to the SQL Server instance as SA, or equivalent.
2. Select the **ControlPointAudit** database.
3. Open the `\ControlPointAudit\01_Convert_ControlPointAudit_v1.sql` script.
4. Find and replace all instances of the following placeholders in the script as indicated:
   a. **`<sever_db_name>`**. Name of the database, typically **ControlPointAudit**.
   b. **`<bcpPath>`**. Storage location with high I/O performance and adequate space for large table data transfers.
   c. **`<username>`**. SQL user with select and insert permissions to all objects in this database.
   d. **`<password>`**. Plain-text password of the specified SQL user.
5. (Optional) Adjust the value of the `@largeRowCount` parameter.

   This parameter specifies the maximum number of rows in a table before the BCP backup and copy method is utilized. Default: 20,000,000.
6. Execute the modified script. Progress and results of the operation are returned in the **Messages** tab.

   > **TIP:** To preserve the record of this execution, save this content to a text file.
7. Open the `\ControlPointAudit\02_ControlPointAudit_ConversionCleanup.sql` script.
8. Find and replace all instances of the `<server_db_name>` placeholder with the name of the ControlPoint database (typically **ControlPointAudit**).

   > **CAUTION:** This script is data destructive. If row count verification tests pass, it permanently removes the backup tables created during the conversion process.
   >
   > If disk storage space is not immediately required to be returned to the OS, execution of this script may be delayed until full conversion and application function verification is complete.
9. Execute the modified script. Progress and results of the operation are returned in the **Messages** tab.

Convert the ControlPointMetaStore database

This section provides the specific steps to convert the component database, named **ControlPointMetaStore** by default.

To convert the ControlPointMetaStore database

1. In SQL Server Management Studio, connect to the SQL Server instance as SA, or equivalent.
2. Open the `\ControlPointMetaStore\01_Convert_ControlPointMetaStore_v2.sql` script.
3. Find and replace all instances of the following placeholders in the script as indicated:
   a. **`<sever_db_name>`**. Name of the database, typically **ControlPointMetaStore**.
   b. **`<bcpPath>`**. Storage location with high I/O performance and adequate space for large table data transfers.
   c. **`<username>`**. SQL user with select and insert permissions to all objects in this database.
   d. **`<password>`**. Plain-text password of the specified SQL user.
4. (Optional) Adjust the value of the `@largeRowCount` parameter.

   This parameter specifies the maximum number of rows in a table before the BCP backup and copy method is utilized. Default: 20,000,000.
5. If your MetaStore database contains custom columns, update the conversion script with those custom columns so that it retains them when run.
6. Execute the modified script. Progress and results of the operation are returned in the **Messages** tab.

   IMPORTANT: This script generates a table named `ControlPointMetadata.Additional_backup`. If you need to re-run this script for any reason, such as if you forgot to update it with custom properties, manually delete this table first. Otherwise, the script fails because the table exists.

   TIP: To preserve the record of this execution, save this content to a text file.
7. Open the `\ControlPointMetaStore\02_ControlPointMetaStore_ConversionCleanup.sql` script.
8. Find and replace all instances of the `<server_db_name>` placeholder with the name of the database (typically **ControlPointMetaStore**).

   CAUTION: This script is data destructive. If row count verification tests pass, it permanently removes the backup tables created during the conversion process.

   If disk storage space is not immediately required to be returned to the OS, execution of this script may be delayed until full conversion and application function verification is complete.
9. Execute the modified script. Progress and results of the operation are returned in the **Messages** tab.

The **ControlPointMetaStore** database conversion is complete. Continue the conversion process with Convert the ControlPointTracking database, on page 15.

Update the conversion script with custom columns

If your **ControlPointMetaStore** database contains custom columns, update the `01_Convert_ControlPointMetaStore_v2.sql` script to include those columns so the conversion process retains them.

To update the conversion script

1. In the `01_Convert_ControlPointMetaStore_v2.sql` script, edit the **[ControlPointMetadata].[Additional]** table creation section by adding your custom columns.

   > **NOTE:** Micro Focus recommends that you get the definition for the new columns from SQL Server.

   Example

```
--Object is partitioned, need to check for scheme existence
IF EXISTS(SELECT * FROM sys.partition_schemes WHERE name = 'ps_binary_eight_fg_
ControlPointMetadata_data' )
BEGIN
     IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'
[ControlPointMetadata].[Additional]') AND type in (N'U'))
     BEGIN
             CREATE TABLE [ControlPointMetadata].[Additional](
                          [RepositoryId] [int] NOT NULL,
                          [DocKey] [binary](8) NOT NULL,
                          [HPRMDataSet] [nchar](2) NULL,
                          [ComparisonField] [nvarchar](64) NULL,
                          [MatchWithinArchive] [binary](8) NULL,
                          [MatchArchive] [binary](8) NULL,
                          [FileType] [nvarchar](32) NULL,
                          [HPRMClassification] [nvarchar](max) NULL,
                          [HPRMContainer] [nvarchar](max) NULL,
                          [SPUUID] [nvarchar](128) NULL,
                          [SPSiteURL] [nvarchar](max) NULL,
                          [SPListURL] [nvarchar](max) NULL,
                          [TrimURLLocationHash] [binary](8) NULL,
                          [ImportErrorCode] [int] NULL,
                          [DocumentDateCreated] [datetime] NULL,
                          [CustomColumnName] [datatype] DEFAULT if any,
            CONSTRAINT [ControlPointMetadata_Additional_PK] PRIMARY KEY
NONCLUSTERED
             (
                           [DocKey] ASC,
                           [RepositoryId] ASC
             )WITH (PAD_INDEX = OFF) ON ps_binary_eight_fg_
```

```
ControlPointMetadata_index(DocKey)
             ) ON ps_binary_eight_fg_ControlPointMetadata_data(DocKey)
             Print '    Table [ControlPointMetadata].[Additional] created.'
   END
END

--Regular version if PS doesn't exist.
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'
[ControlPointMetadata].[Additional]') AND type in (N'U'))
BEGIN
          CREATE TABLE [ControlPointMetadata].[Additional](
                         [RepositoryId] [int] NOT NULL,
                         [DocKey] [binary](8) NOT NULL,
                         [HPRMDataSet] [nchar](2) NULL,
                         [ComparisonField] [nvarchar](64) NULL,
                         [MatchWithinArchive] [binary](8) NULL,
                         [MatchArchive] [binary](8) NULL,
                         [FileType] [nvarchar](32) NULL,
                         [HPRMClassification] [nvarchar](max) NULL,
                         [HPRMContainer] [nvarchar](max) NULL,
                         [SPUUID] [nvarchar](128) NULL,
                         [SPSiteURL] [nvarchar](max) NULL,
                         [SPListURL] [nvarchar](max) NULL,
                         [TrimURLLocationHash] [binary](8) NULL,
                         [ImportErrorCode] [int] NULL,
                         [DocumentDateCreated] [datetime] NULL,
                         [CustomColumnName] [datatype] DEFAULT if any,
                CONSTRAINT [ControlPointMetadata_Additional_PK] PRIMARY KEY
NONCLUSTERED
                (
                         [DocKey] ASC,
                         [RepositoryId] ASC
                )WITH (PAD_INDEX = OFF) ON [fg_ControlPointMetadata_index]
                ) ON [fg_ControlPointMetadata_data]
                Print '    Table [ControlPointMetadata].[Additional] created.'
END
```

2. If you added any indexes for the new column in the **[ControlPointMetadata].[Additional]** table, add them to the index creation section.

   > **NOTE:** Replace the **bold** text with the name of the index and the appropriate column name.

   The index section begins with the following line:

   ```
   -- create indexes functions and views
   ```

   Example

   ```
   -- create indexes functions and views

   ...
   ```

```
IF NOT EXISTS (SELECT name FROM sys.indexes WHERE object_id = OBJECT_ID(N'
[ControlPointMetadata].[Additional]')
AND name = 'ControlPointMetadata_Additional_IDX_IndexName')
BEGIN
       CREATE NONCLUSTERED INDEX [ControlPointMetadata_Additional_IDX_
IndexName'] ON
       [ControlPointMetadata].[Additional]
        (
                        [CustomColumnName] ASC
        ) WITH (PAD_INDEX = OFF) ON [fg_ControlPointMetadata_index]
        Print '    Non Clustered Index ControlPointMetadata_Additional_IDX_
IndexName ON
       [ControlPointMetadata].[Additional] created.'
END
```

3. Continue with step 6 from the **ControlPointMetaStore** database conversion procedure.

Convert the ControlPointTracking database

This section provides the specific steps to convert the component database, named **ControlPointTracking** by default.

To convert the ControlPointTracking database

1. In SQL Server Management Studio, connect to the SQL Server instance as SA, or equivalent.
2. Select the **ControlPointTracking** database.
3. Open the `\ControlPointTracking\01_Convert_ControlPointTracking_v1.sql` script.
4. Find and replace all instances of the following placeholders in the script as indicated:
   a. **`<sever_db_name>`**. Name of the database, typically **ControlPointTracking**.
   b. **`<bcpPath>`**. Storage location with high I/O performance and adequate space for large table data transfers.
   c. **`<username>`**. SQL user with select and insert permissions to all objects in this database.
   d. **`<password>`**. Plain-text password of the specified SQL user.
5. (Optional) Adjust the value of the `@largeRowCount` parameter.

   This parameter specifies the maximum number of rows in a table before the BCP backup and copy method is utilized. Default: 20,000,000.
6. Execute the modified script. Progress and results of the operation are returned in the **Messages** tab.

   > **TIP:** To preserve the record of this execution, save this content to a text file.

7. Open the `\ControlPointTracking\02_ControlPointTracking_ConversionCleanup.sql` script.
8. Find and replace all instances of the `<server_db_name>` placeholder with the name of the database (typically **ControlPointTracking**).

   > **CAUTION:** This script is data destructive. If row count verification tests pass, it permanently removes the backup tables created during the conversion process.
   >
   > If disk storage space is not immediately required to be returned to the OS, execution of this script may be delayed until full conversion and application function verification is complete.

9. Execute the modified script. Progress and results of the operation are returned in the **Messages** tab.

Appendix A: TSQL example: BCP expected size calculation

The TSQL example in this section calculates the expected size, in megabytes, of a bulk copy (BCP) of the Document table. To determine this value, it calculates the average row size of the Document table and the estimated number of rows in that table, and then multiplies those values together. You can use this example to gather information from your ControlPoint environment.

```
declare
       @AvgRowSizeInBytes   int,
       @RowEstimate         bigint,
       @TableName           nvarchar(128) = 'Document',
       @SchemaName          nvarchar(128) = 'Metadata'


select @AvgRowSizeInBytes = sum(a.max_length)from
       ( select c.name,
               case
                       when c.max_length < 0 then 1024
                       -- Assume unlimited length fields of average length, 1KB in this case.
                       when ty.name like '%var%' then (c.max_length/4)
                       -- When a variable width field is found, take a fraction of the maximum
                       -- length (25% in this example) or to be more accurate, calculate the
                       -- current average length of data currently in the column.
                       else c.max_length
               end as max_length
       from sys.columns c (nolock)
       inner join sys.tables t (nolock) on c.object_id = t.object_id
       inner join sys.types ty (nolock) on c.user_type_id = ty.user_type_id
       where t.name = @TableName --Table Name
       ) as a


select @RowEstimate = sum(p.rows)
       from sys.tables t
       inner join sys.indexes i on i.object_id = t.object_id and i.index_id < 2
       inner join sys.partitions p on p.object_id = t.object_id and p.index_id = i.index_id
       inner join sys.schemas s on t.schema_id = s.schema_id
       where t.name = @TableName and s.name = @SchemaName


select ((@AvgRowSizeInBytes * @RowEstimate) /1024)/1024 as ExpectedOutputInMB
```

Appendix B: Conversion reference

This appendix provides reference information to support improvement claims and to provide data to facilitate conversion execution planning.

Conversion

Size reference information focuses exclusively on the **ControlPointMetaStore** database. The other databases' storage size, as well as the time and resources to convert them, are insubstantial in comparison.

Total number of stored documents: 347,641,154

Reference SQL Server

CPU	2x Intel Xeon E5 2660 @ 2.20GHz, Hyperthreading Enabled (Total: 16 core/32 thread)
RAM	128G DDR3 (SQL permitted to use 124G)
System Disk	1x 450G HDD
Tempdb Disk	7x 900G 2x HDD RAID0
Database Disk	3x 1500G SSD

Disk I/O reference (concurrently measured by CrystalDiskMark)

Report ID	Sequential Read	Sequential Write	Random Read 512K	Random Write 512K	Random Read 4K (QD=1)	Random Read 4K (QD=1) IOPS	Random Write 4K (QD=1)	Random Write 4K (QD=1) IOPS	Random Read 4K (QD=32)	Random Read 4K (QD=32) IOPS	Random Write 4K (QD=32)	Random Write 4K (QD=32) IOPS
system	189.633	151.879	79.424	142.545	0.672	164.1	5.313	1297.2	2.050	500.5	5.257	1283.4
tempdb	311.058	290.263	70.920	142.474	0.820	200.3	6.430	1569.9	4.116	1005	7.004	1710
database	384.987	448.749	329.368	494.759	14.115	3446	67.043	16367.9	85.023	20757.7	82.046	20030.8

5.9.0 database size pre-conversion

database_name	**database_size**	**unallocated space**	
ControlPointMetaStore	1007723.32 MB	14598.63 MB	
Reserved	**data**	**index_size**	**unused**
1016566592 KB	470683552 KB	545817792 KB	65248 KB

5.9.0 database size post-conversion

database_name	database_size	unallocated space	
ControlPointMetaStore	631867.57 MB	1423.22 MB	
Reserved	**data**	**index_size**	**unused**
645181920 KB	471001296 KB	172662280 KB	1518344 KB

Execution times

00:59:53	Drop all dependent objects, Rename to create backup tables, Create replacement tables, Create temporary indexes on backup tables
31:13:00	Restore all data to new tables (Filegroups and partitions, even distribution)
03:51:00	Recreate non-clustered indexes (Filegroups and partitions, even distribution)
00:47:00	Recreate all remaining constraints
00:01:00	Recreate all functions and views
36:50:53	Total

Performance

Metrics in this section normalized to the same data set, the same SQL Server hardware, etc. and are intended to be directly comparable and of substantial enough size to indicate likely real world performance.

The ControlPoint configuration driving this collection is also static and separate from the SQL Server.

Concurrent collections: 4

Collection size (documents): 1,000,000

Total documents per test iteration: 4,000,000

Reference SQL Server

CPU	2x Intel Xeon E5620 @ 2.40GHz, Hyperthreading Enabled (Total: 8 cores/16 threads)
RAM	32G DDR3 (SQL Server is permitted to use 16G)

Reference SQL Server, continued

System Disk	1x 300G 2x HDD RAID1
Tempdb & Database disks	3x 300G 2x HDD RAID1 1x 900G 6x HDD RAID0+1

Disk I/O reference (concurrently measured by CrystalDiskMark)

Report ID	Sequential Read	Sequential Write	Sequential Read (QD=32)	Sequential Write (QD=32)	Random Read 4K (QD=1)	Random Read 4K (QD=1) IOPS	Random Write 4K (QD=1)	Random Write 4K (QD=1) IOPS	Random Read 4K (QD=32)	Random Read 4K (QD=32) IOPS	Random Write 4K (QD=32)	Random Write 4K (QD=32) IOPS
System R1	165.477	195.864	245.840	189.610	1.232	300.8	12.695	3099.4	4.895	1195.1	11.495	2806.4
Database R0+1	467.257	440.631	510.922	370.264	1.748	426.8	19.772	4827.1	17.086	4171.4	20.711	5056.4
Database R1	272.000	241.378	250.913	215.737	1.924	469.7	19.153	4676.0	10.662	2603.0	16.249	3967.0

Detailed metric information

This section provides detailed information captured during the collection operation. Brief guidance for each report and a description of what it indicates are as follows:

- **Disk Performance**. The percentage of availability of the logical volume reported to Windows. Higher is better. The average value over the measurement window calculates both this value and the average read and write queue depth on the volume. Lower is better.

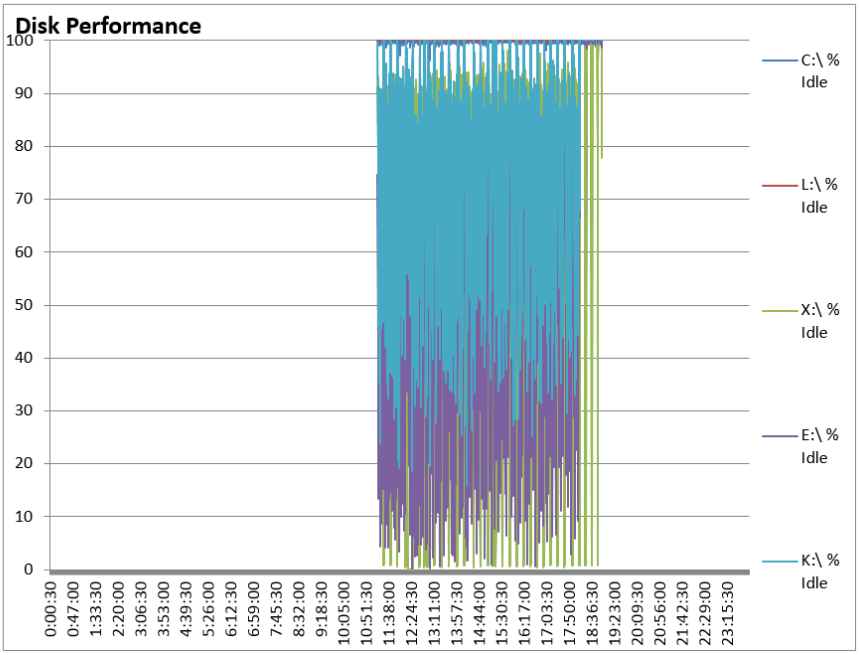
  A fully utilized disk will show queue depth approaching 1 when considered over time. A value above 1 indicates that improving I/O throughput of the volume would improve performance. The delta above 1 can be used as a relative measure of over-utilization of the logical volume.

- **Locks (ms)**. The time for SQL to acquire and hold necessary locks, of all kinds, to perform the requested operations. Lower is better.

- **Average Wait Time (ms)**. The amount of time required to acquire resources to perform requested operations. Lower is better. Displayed are times to acquire the necessary locks, the time to acquire the network socket for the data, and the time to acquire the necessary pages (disk).

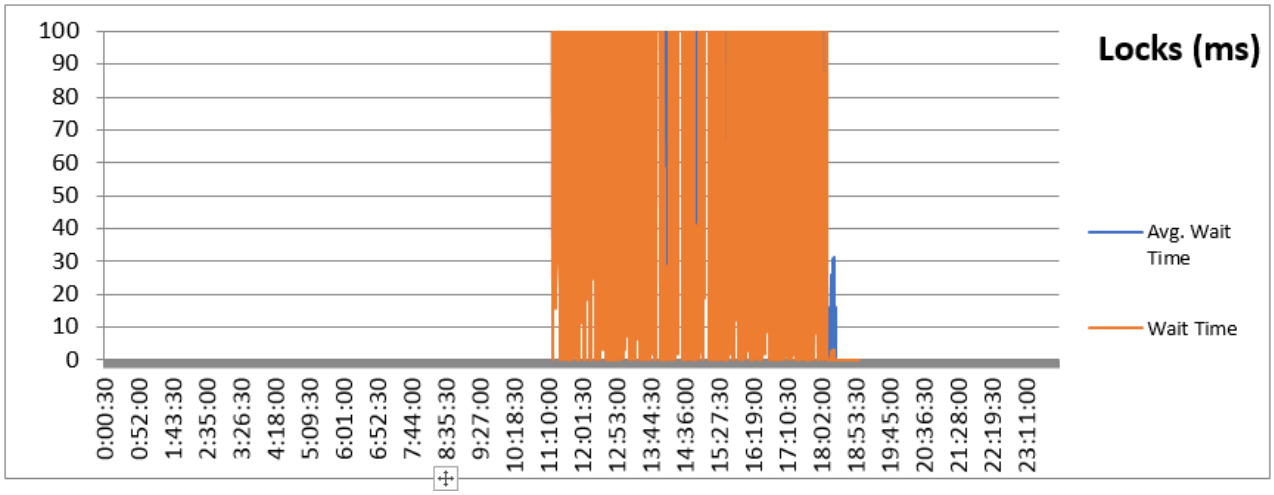
- **SQL Page Life Expectancy (s)**. The amount of time any given page can be expected to reside in memory. Higher is better. This measure is used to consider memory pressure in SQL. Anything below 600 indicates that memory available to SQL is significantly reducing performance.

5.9.0 database metrics pre-conversion



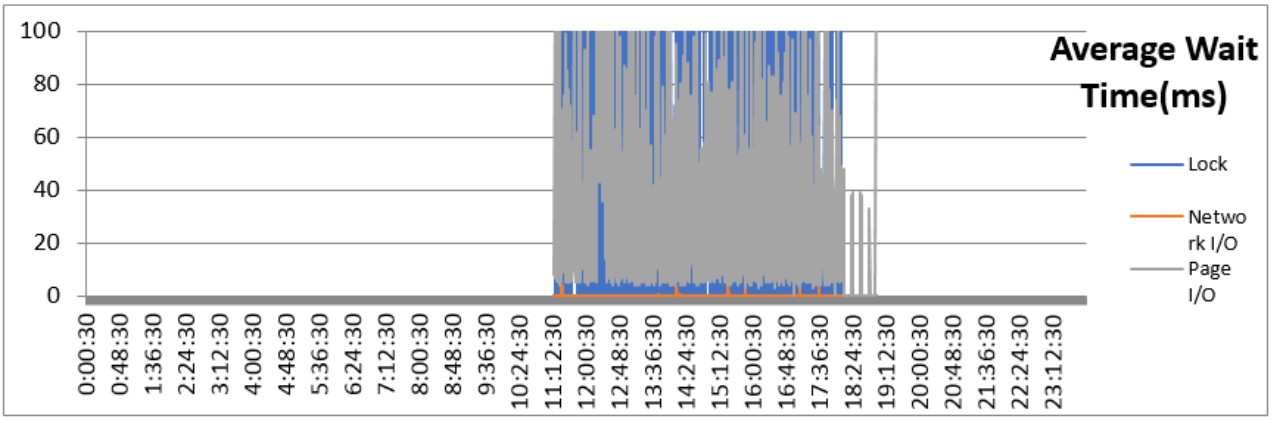
Averages

99.53 C:\ % Idle	0.01 C:\ QD
100.10 L:\ % Idle	0.00 L:\ QD
69.02 X:\ % Idle	25.26 X:\ QD
56.15 E:\ % Idle	18.18 E:\ QD
75.37 K:\ % Idle	11.88 K:\ QD



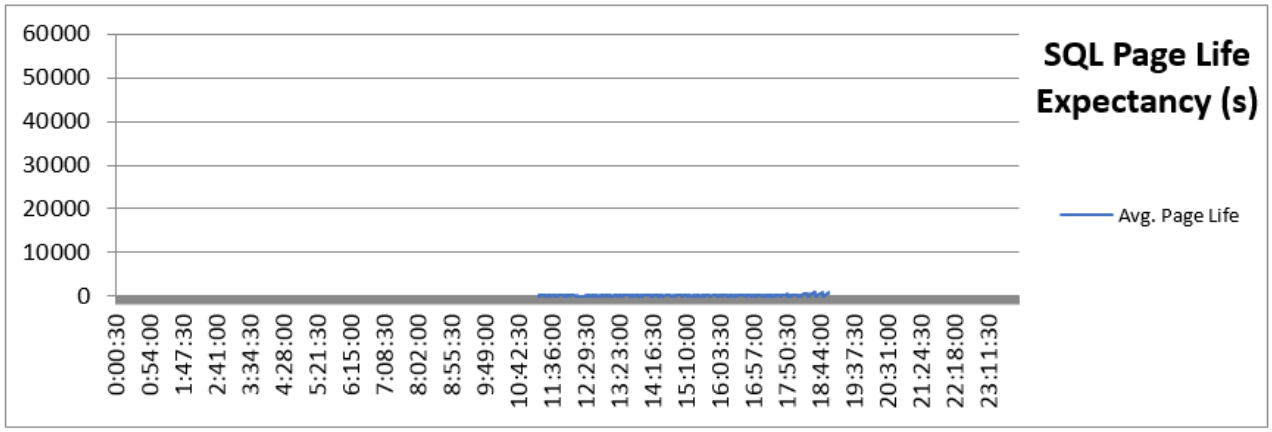
Averages

4433.80	Average wait time
2326.94	Wait time



Averages

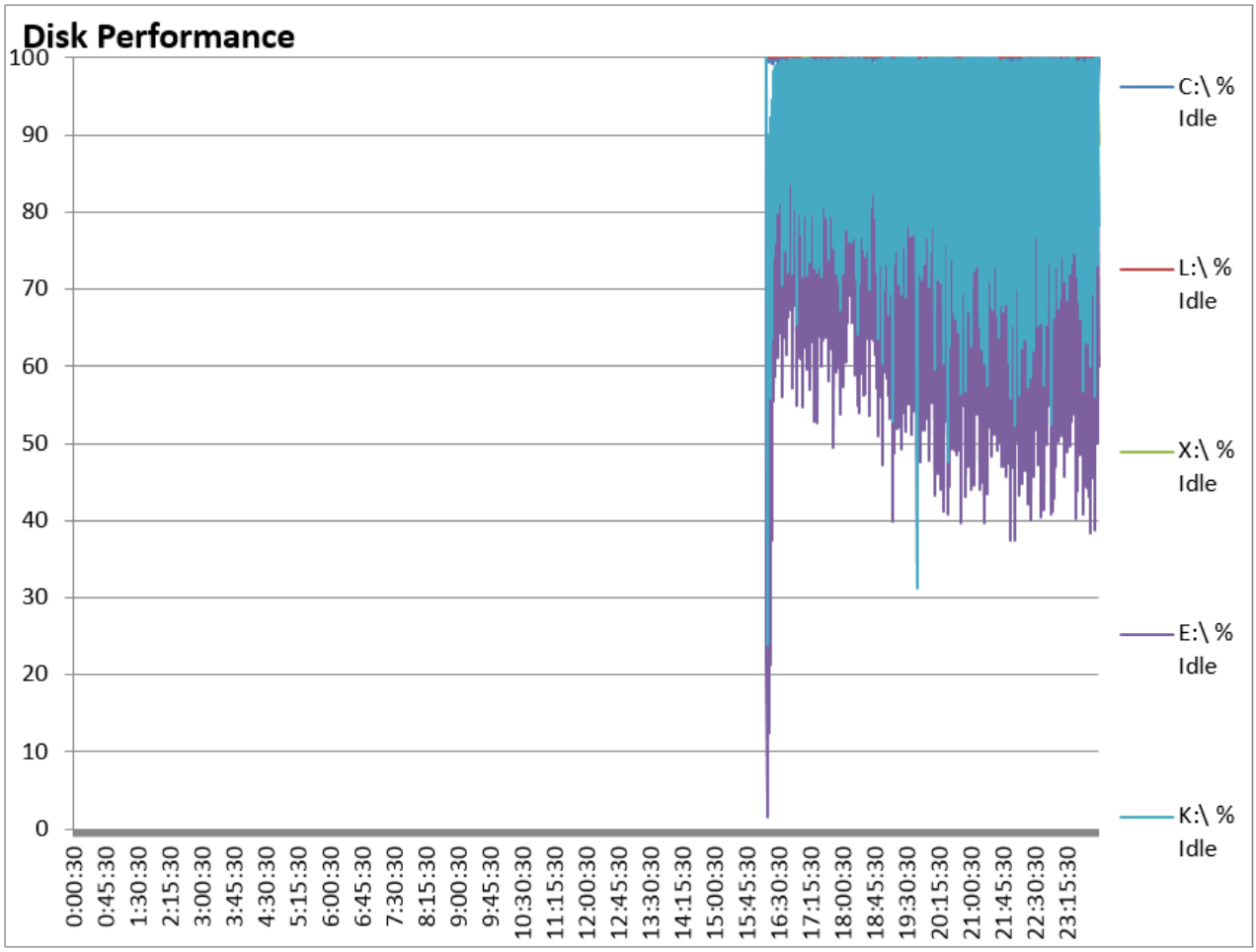
1117.45	Lock
0.03	Network I/O
33.51	Page I/O



Averages

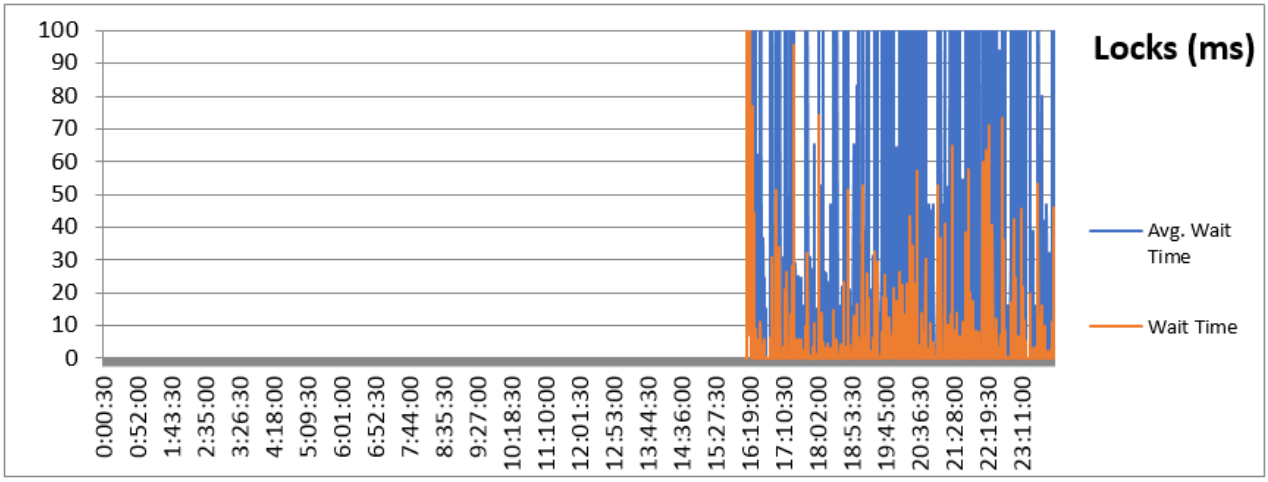
174.54	Average page life

5.9.0 database metrics post-conversion



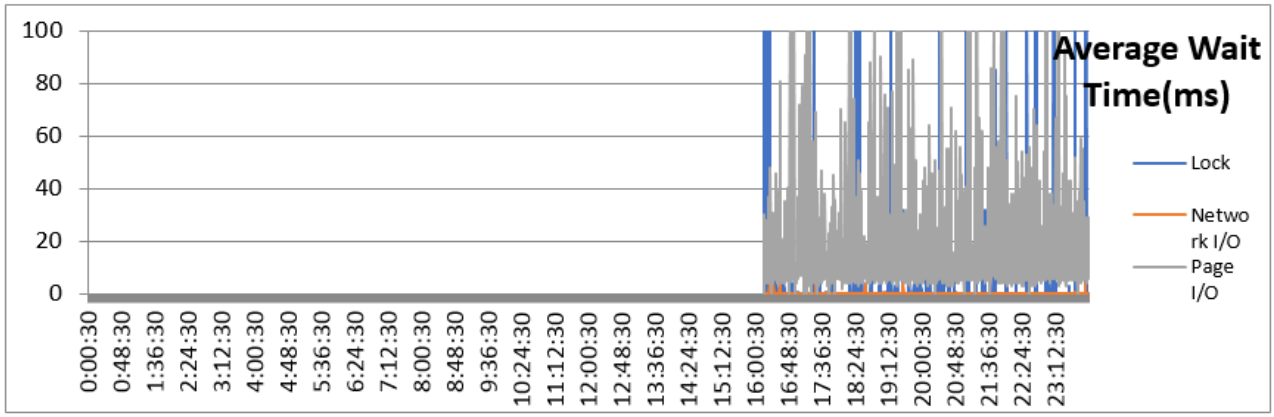
Averages

99.63 C:\ % Idle	0.00 C:\ QD
100.12 L:\ % Idle	0.00 L:\ QD
94.98 X:\ % Idle	0.21 X:\ QD
78.36 E:\ % Idle	19.50 E:\ QD
88.56 K:\ % Idle	10.16 K:\ QD



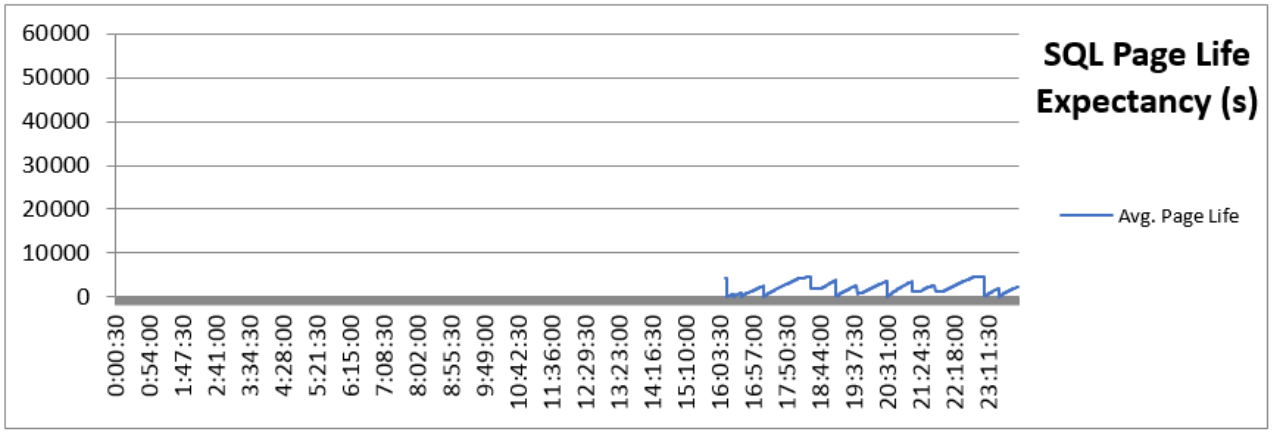
Averages

51.87	Average wait time
4.91	Wait time



Averages

15.85	Lock
0.03	Network I/O
17.08	Page I/O



Averages

2064.48	Average page life

Send documentation feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Database Conversion Guide (Micro Focus ControlPoint 5.9.0)

Add your feedback to the email and click **Send**.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to swpdl.controlpoint.docfeedback@microfocus.com.

We appreciate your feedback!