
San Francisco

VULKAN

A META DATA DRIVEN BACKUP RESTORE SYSTEM

VULKAN

INTRODUCTION - VULKAN

VULKAN - DISTRUBUTION OF SUBSETS OF DATA, BETWEEN DATABASE SYSTEMS, IN A LIGHTWEIGHT, CONFIGURABLE AND SCALABLE FASHION.

PRESENTATION CONTEXT

- DATAWAREHOUSING
- SQL SERVER
- BACKUP / RESTORE OF DATABASES
- FILEGROUPS = CONTAINERS OF DATA (TABLES / INDEXES)
- PARTITONING OF DATA
- META DATA DRIVEN SYSTEMS
- BIML (Business Intelligece Markup Language)
- INTEGRATION SERVICES (SSIS)

VULKAN

BACKGROUP – MISSION STATEMENT

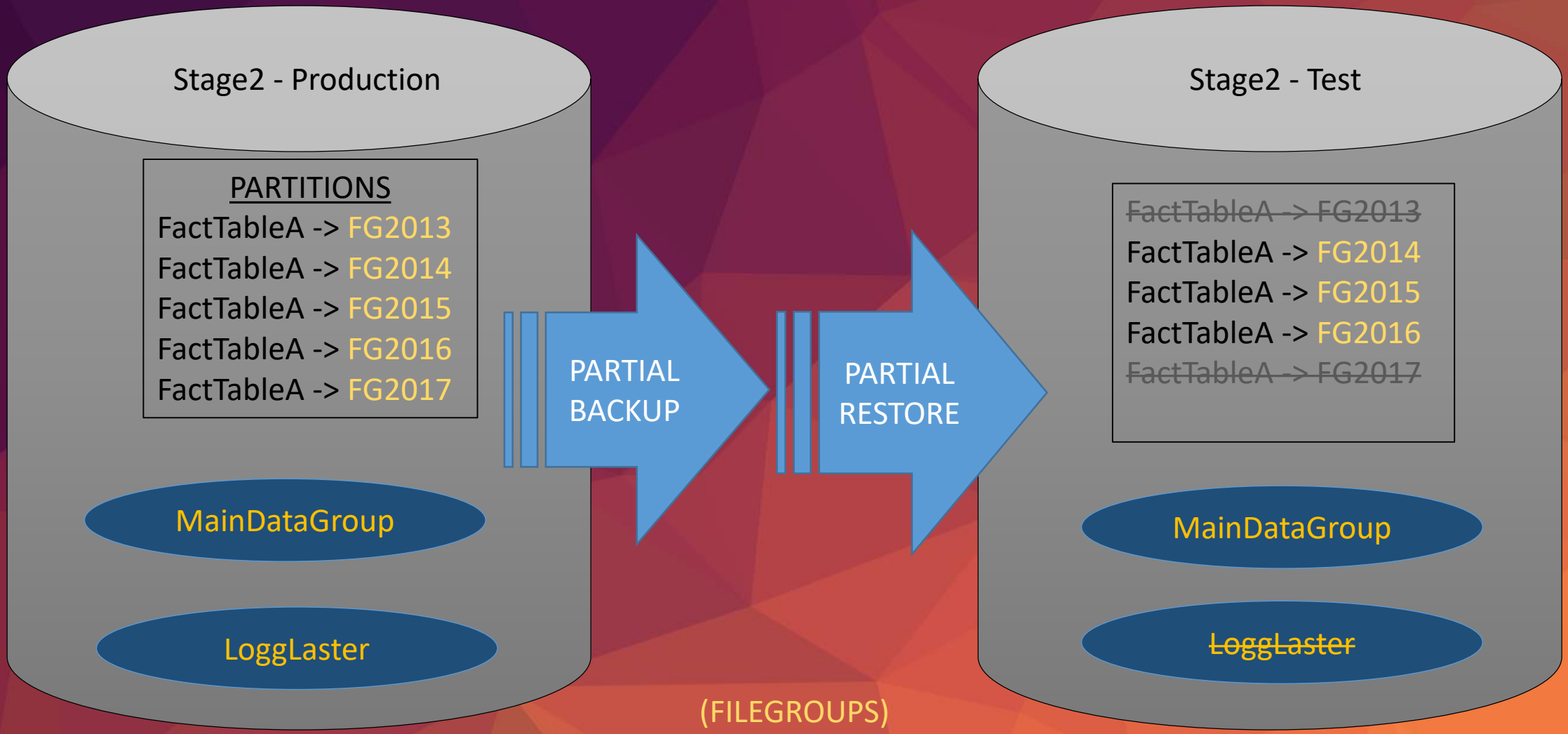
"I want us to be able to distribute subsets of production data, to other subsystems, in a light-weight, configurable and scalable fashion"

In the context of distrubiting subsets of data, we want:

- Ability to completly include/omit entire islands of data. (*For example, omit large logg tables*)
  - Vertical Scalability

- Ability to include/omit specific levels of data, i.e. Include/omit ranges of data, usually time ranges, within **individual** tables. (*For example include only last 3 years of data*)
  - Horizontal Scalability

VULKAN


Stage2 - Production
PARTITIONS
FactTableA -> FG2013
FactTableA -> FG2014
FactTableA -> FG2015
FactTableA -> FG2016
FactTableA -> FG2017
MainDataGroup
LoggLaster
PARTIAL BACKUP
PARTIAL RESTORE
Stage2 - Test
~~FactTableA -> FG2013~~
FactTableA -> FG2014
FactTableA -> FG2015
FactTableA -> FG2016
~~FactTableA -> FG2017~~
MainDataGroup
~~LoggLaster~~
(FILEGROUPS)

VULKAN

Vulkan – A Technical Overview

- Handles Full or Partial Distribution of subsets of data, to SubSystems (Typical Production Data to Test Systems)
- System Parameters in Configurable Meta Data SQL Tables
- SSIS packages to execute all parts of system workflow
- BIML Code, in conjunction with Metadata, to Auto generate SSIS Packages

VULKAN

TERMINOLOGY - DISTRUBUTION OF DATA USING SQL SERVER

BACKUP – Backup a full or partial database to a *.BAK-file
RESTORE – Restore a database from a *.BAK-file, Full of Partial
FILEGROPS – A way to group set of data, logical (and physical)
DATAFILE – The physical file of data, attached to a certain FILEGROUP
CLUSTRED INDEX – Instructions (Function) on how and where a table is stored om disk
PARTITIONS – A way to split tables into subsections, logical (and physical), based on fixed ranges
MERGE - Merging of Partitions
SWITCH – Switching of data, in and out of Partitions, using Shadow-tables

VULKAN

PREPARING FOR DATA DISTRUBITION – SET UP PARTITIONING

Sample table: dbo.FactTablePeriode (Partition Column: PERIOD (DataType INT))

Step 1. Create FileGroups and DataFiles, as containers for Partitioned Data (FG2014 -> FG2014.ndf…)

Step 2. Create Partition Function to support Range and DataType

```
CREATE PARTITION FUNCTION [IntRange_pf](INT) AS RANGE RIGHT FOR VALUES
(
20030101, 20040101, 20050101, 20060101, 20070101, 20080101,
20090101, 20100101, 20110101, 20120101, 20130101, 20140101,
20150101, 20160101, 20170101, 20180101, 20190101, 20200101,
20210101, 20220101, 20230101, 20240101,20250101
)
```

RANGE RIGHT; example: A row with PERIODE value 20060101, will be placed to the partition to the RIGHT (FG2006)
RANGE LEFT; example: A row with PERIODE value 20060101, will be placed to the partition to the LEFT (FG2006)

VULKAN

PREPARING FOR DATA DISTRUBITION – SET UP PARTITIONING

Sample table: dbo.FactTablePeriode (Partition Column: PERIOD (DataType INT))

Step 3. Create Partition Schema, to align Ranges with correct FileGroup (and Data File)

```
CREATE PARTITION SCHEME IntScheme_ps
AS PARTITION [IntRange_pf]
TO (

            FG2002,FG2003,FG2004,FG2005,FG2006,
            FG2007,FG2008,FG2009,FG2010,FG2011,
            FG2012,FG2013,FG2014,FG2015,FG2016,
            FG2017,FG2018,FG2019,FG2020,FG2021,
            FG2022,FG2023,FG2024,FG2025
 )
```

VULKAN

PREPARING FOR DATA DISTRUBITION – SET UP PARTITIONING

Sample table: dbo.FactTablePeriode (Partition Column: PERIOD (DataType INT))

Step 4. Move table data into Partitons with Aligned Clustred Index, with DROP_EXISTING on Partition Scheme

```
        CREATE UNIQUE CLUSTERED INDEX [ClusteredIndexPeriode] ON [dbo].[FactTablePeriod]
(
        [Period] ASC,
        [SalesOrderID] ASC,
        [SalesOrderDetailID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF,
DROP_EXISTING = ON, ONLINE = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
ON [IntScheme_ps]([Period])
```

VULKAN

BACKUP DATABASE PARTIAL

Sample table: dbo.FactTablePeriode (Partition Column: PERIOD (DataType INT))

Step 5. Backup Database WITH PARTIAL (in Full Recovery Mode)

```
BACKUP DATABASE [Stage2]

 FILEGROUP = 'PRIMARY',
 FILEGROUP = 'MainDataGroup',
 FILEGROUP = 'FG2014',
 FILEGROUP = 'MainIndexGroup',
 FILEGROUP = 'EVRYGroup',
 FILEGROUP = 'UDBGroup',
 FILEGROUP = 'FG2015',
 FILEGROUP = 'FG2016'

    TO  DISK = 'E:\SQLBackup\BackupFileforRestoreToTest\Stage2.bak'

    WITH COMPRESSION,
    COPY_ONLY,
    NOFORMAT,
    NOINIT,
    NAME = N'Stage2 Backup',
    SKIP,
    REWIND,
    NOUNLOAD;
```

VULKAN

RESTORE DATABASE PARTITAL

Sample table: dbo.FactTablePeriode (Partition Column: PERIOD (DataType INT))

Step 6. Restore Database on Destination Server WITH PARTIAL (in Full Recovery Mode)

```
RESTORE DATABASE [VulkanProd]

FILEGROUP = 'FG2015',
FILEGROUP = 'FG2016',
FILEGROUP = 'FG2017',
FILEGROUP = 'MainDataGroup',
FILEGROUP = 'MainIndexGroup',
FILEGROUP = 'PRIMARY'
                FROM DISK = 'W:\VulkanProd.bak'

    WITH
    PARTIAL,
    REPLACE,
    RECOVERY,

MOVE N'VulkanProd' TO N'F:\Data\VulkanProd.mdf',
MOVE N'VulkanProd_log' TO N'F:\Log\VulkanProd_log.ldf',
MOVE N'MainDataGroup' TO N'F:\Data\MainDataGroup.ndf',
MOVE N'MainIndexGroup' TO N'F:\Data\MainIndexGroup.ndf',
MOVE N'FG2016' TO N'F:\Data\VulkanProdFG2016.ndf',
MOVE N'FG2017' TO N'F:\Data\VulkanProdFG2017.ndf',
MOVE N'FG2015' TO N'F:\Data\VulkanProdFG2015.ndf'
```

VULKAN

PRE RESTORE

Sample table: dbo.FactTablePeriode (Partition Column: PERIOD (DataType INT))

Step 7. Test Select * dbo.FactTablePeriod

One of the partitions of index ‘ClusteredIndex-hiskonto' for table 'dbo.UDBHiskonto'(partition ID 72057594042843136) resides on a filegroup ("FG1994") that cannot be accessed because it is offline, restoring, or defunct. This may limit the query result.

Step 8. Test Select * dbo.FactTablePeriod WHERE PERIODE BETWEEN 201401 AND 201612

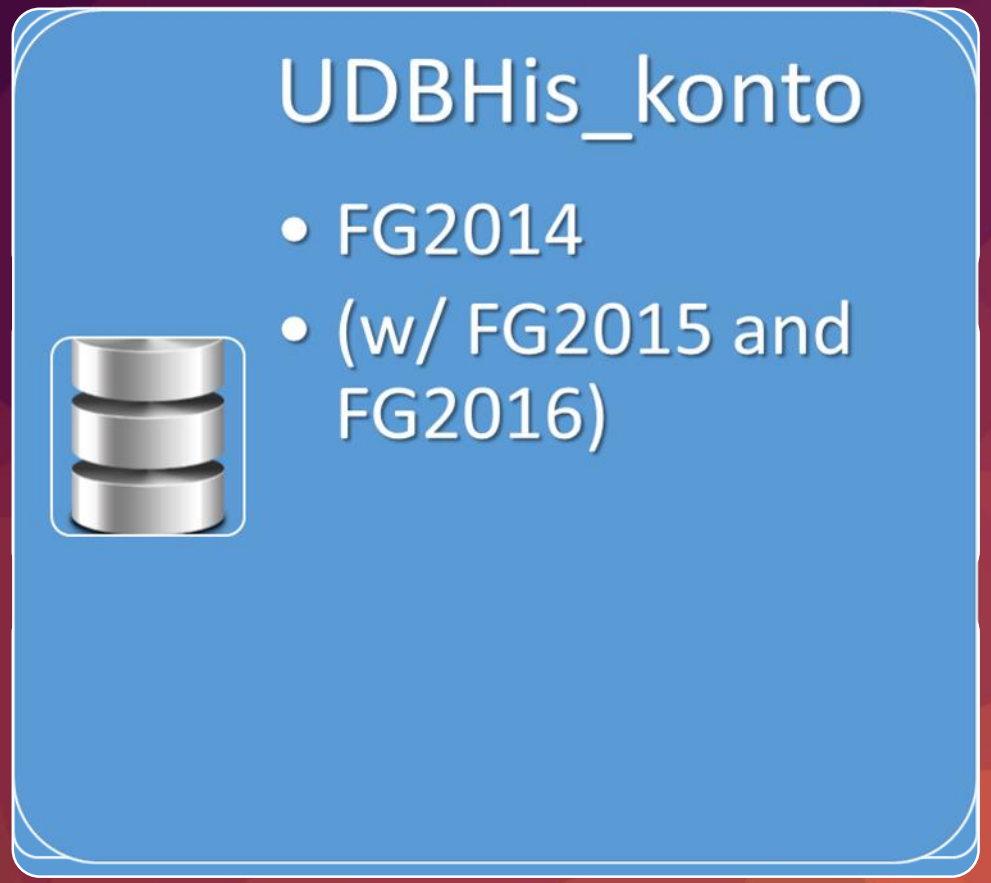
Produces rows!

Step 9. Fix Defuct Issue with **Merge**, **Switch** and **Rename**.

VULKAN

MERGE PARTITIONS

MERGE – Move Data from one Partition into a adjacent Partition.




```
ALTER PARTITION FUNCTION
[PeriodeFloatRange_pf]()

  MERGE RANGE (201601);
GO
```

Remember, 201601 is the lower boundry point (RANGE)

VULKAN

SWITCH PARTITIONS

CREATE SHADOW TABLE

1. Create “Shadow”-table of FactTablePeriod, on FG2015. (same filegroup av FactTablePeriod, after Merge)

```
CREATE TABLE [dbo].[FactTablePeriod](
[KON_KONTONR] [VARCHAR](11) NULL,
[...]
[Checksum] [INT] NULL
) ON FG2015
```

2. Create Index on Shadow-table, aligned with Partition.

```
CREATE UNIQUE CLUSTERED INDEX [ClusteredIndexPeriode] ON [dbo].[FactTablePeriod_shadpw]
(
        [Period] ASC,
        [SalesOrderID] ASC,
        [SalesOrderDetailID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF,
DROP_EXISTING = OFF, ONLINE = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
GO
```

VULKAN

SWITCH PARTITIONS

3. Switch Partition data between Main and Shadow Tables.

```
ALTER TABLE [dbo].[FactTablePeriod]
SWITCH PARTITION 21 TO [dbo].[FactTablePeriod_shadow]
GO
```

4. Rename Tables

```
EXEC sp_rename 'FactTablePeriod', 'FactTablePeriod_dummy';

EXEC sp_rename 'FactTablePeriod_Shadow', 'FactTablePeriod';
```

5. “Original” table name now works!

```
SELECT *
FROM dbo.FactTablePeriod
```

VULKAN

LIVE DEMO

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]
VULKAN

The Meta Data Configuration Tables

We have a total of 6 configuration tables that specifies the **core objects** and controls the **work flow** of the VULKAN system.

[vulkan].[BaseInformation] – Subsystem description
[vulkan].[BaseConfiguration] – The initial setup, per Subsystem / Server-to-server / Database
[vulkan].[DatabaseFilesConfiguration] – Information on Data- and Logfile, FileGroups, per database
[vulkan].[PartitionedTablesConfiguration] – Information concerning the partitioned tables
[vulkan].[MergeConfiguration] – Controls the workflow of the merging of Partitions on the Destination Server
[vulkan].[SwitchConfigurtation] - Controls the workflow of the Switching of Partitions on the Destination Server
[vulkan].[NonClustedIndexProcessConfiguration] – Handles the process of droping and recreating the (Non-aligned) NonClustered Indexes on the Destination database. The system uses this table in runtime, no need to set this manually

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]

VULKAN

The Meta Data Configuration Tables

[vulkan].[BaseConfiguration] – The initial setup, per Subsystem / Server-to-server / Database

[**SubSystemName**] - Grouping name, for a server to server backup/restore setup
[**SourceServer**] - Server where the Backup is created
[**DestinationServer**] - Server where the restore is done
[**SourceDatabase**] - Database to Backup
[**DestinationDatabase**] - Database to Restore
[**RecoverModeSourceServer**] - Recovery Mode of Source database (Full / Simple)
[**RecoverModeDestinationServer**]- Recovery Mode of Destination database (Full / Simple)
[**FullOrPartialBackup**] - Is it a Full (all filegroups) or Partial (selected Filegroups)
[**FilePathBakFileSourceServer**] - Path to folder on Source server, for .BAK file (F:\Backup\)
[**FilePathBakFileDestinationServer**] - Path to folder on Destination server, for .BAK file
[**FilePathBakFileDestinationServerRemoteAccess**] - Remote access to folder on destination server
[**DeleteBackupFileOnSourceServer**] - If you want to remove the BAK file on Source Server after Restore (y/n)
[**DeleteBackupFileOnDestinationServer**]- If you want to remove the BAK file on Destination Server after Restore (y/n)
[**Active**] - If the Row is Active and should be used. (y/n)

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]

VULKAN

The Meta Data Configuration Tables

[vulkan].[DatabaseFilesConfiguration] – Information on Data- and Logfile, FileGroups, per database

[**SubSystemName**] - Grouping name, for a server to server backup/restore setup
[**Database**] - Name of database
[**FileGroupsToBackupRestore**] - One rows for each FileGroup to Backup / Restore (for Log File – “Not Applicable”)
[**Type**] - Datafile / LogFile
[**FilePathDestination**] - Specifies where on the Destination server each File should be placed
[**LogicalName**] - Logical name of Datafile / LogFile
[**Active**] - If the Row is Active and should be used (y/n)

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]
VULKAN

The Meta Data Configuration Tables

[vulkan].[PartitionedTablesConfiguration] – Information concerning the partitioned tables

[**SubSystemName**] - Link to same SubSystem as above
[**SourceServer**] - Source server for the tables
[**SourceDatabase**] - Source database for the tables
[**DestinationServer**] - Destination server for the tables
[**DestinationDatabase**] - Destination database for the tables
[**Table**] - name of the table
[**Active**] - 'y' to use this table, 'n' to not use

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]

VULKAN

The Meta Data Configuration Tables

[vulkan].[MergeConfiguration] – Controls the workflow of the merging of Partitions on the Destination Server

[**SubSystemName**] - Grouping name, for a server to server backup/restore setup
[**DestinationServer**] - Server where we are doing the Merge
[**DestinationDatabase**] - Database where we are doing the Merge
[**PartitionFunction**] - Name of function to merge
[**Range**] - Name of the Range to merge
[**Order**] - The order in which the Merges should be done
[**Active**] - If the Row is Active and should be used (y/n)

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]

VULKAN

The Meta Data Configuration Tables

[vulkan].[SwitchConfigurtation] - Controls the workflow of the Switching of Partitions on the Destination Server

[**SubSystemName**] - Grouping name, for a server to server backup/restore setup
[**DestinationServer**] - Server where we are doing the Switch
[**DestinationDatabase**] - Database where we are doing the Switch
[**Table**] - The partitioned tables involved in the Switch
[**CurrentFileGroup**] - Which FileGroup are the partitioned tables Merged to? (See above)
[**Active**] - Active or not (y/n)

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]

VULKAN

The Meta Data Configuration Tables

[vulkan].[NonClustedIndexProcessConfiguration] – Handles the process of droping and recreating the (Non-aligned) NonClustered Indexes on the Destination database. The system uses this table in runtime, no need to set this manually

[SubSystem] - Grouping name, for a server to server backup/restore setup
[SourceServer] - Server where the Backup is created
[SourceDatabase] - Database to Backup
[DestinationServer] - Server where the restore is done
[DestinationDatabase] - Database to Restore
[CurrentSchema] – Schema for table
[CurrentTable] – Table
[RunDate] – Time of execution
[DropIndexScript] – SQL Script for Droping Index
[CreateIndexScript] – SQL Script for Creating Index

Lets look as the tables!

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]

VULKAN

The Meta Data Configuration Tables

Adding Rows to the Configuration Tables

“…..\Test_environment\VULKAN\VULKAN Dokumentasjon\VulkanAddConfigRowsProcScript.sql“

[partition].[PartitionConfiguration]
[partition].[PartitionConfiguration]

VULKAN

Putting it all together, BIML to the rescue!

We have six BIML scripts, that reads from Config tables and generates SSIS packages:

GeneratePreProcessPackages.biml - Generates the PreProcess Package -
Saves Drop and Create sql scripts for Non-Clustered Non-Partition-Aligned Indexes for the partitioned tables in the Index meta data in [vulkan].[NonClustedIndexProcessConfiguration] table. (**This is a runtime table, no need to fill in this manually.**)

GenerateBackupPackages.biml - Generates Backup SSIS packages

GenerateRestorePackages.biml - Generates Restore SSIS packages

GenerateMergePackages.biml - Generates Merge SSIS packages

GenerateSwitchPackages.biml - Generates Switch SSIS packages

GeneratePostProcessPackages.biml - Recreates the Non-Clustered Non-Partition-Aligned Indexes for the partitioned tables, on the Destination server.To generate the SSIS packages, aligned with the Config tables, right click each Biml-file and select: Generate SSIS packages. **Remember, if you add/edit rows in ConFig Tables, you have re-generate the affected SSIS package, and Deploy this to the SSIS server.**

VULKAN

LIVE DEMO