SQLintersection

Session: Thursday, 11:30 am - 12:45 pm

Temporal Tables In Depth

Ben Miller

@DBAduck - ben@benmiller.net

Ben Miller

- **Data Platform MVP since 2009**
- **Microsoft Certified Master in SQL Server**
- **Author – https://dbaduck.com**
  - MVP Deep Dives #2
  - PowerShell MVP Deep Dives
  - PowerShell Stairway on SQLServerCentral.com
- **Email : ben@benmiller.net**
- **Twitter: @DBAduck**
- **GitHub: github.com/dbaduck**

Overview

- What are Temporal Tables
- Reasons to Entertain Use
- How does it work?
- In Memory OLTP Use
- Anonymous Type, Default Type, Custom Type
- Adding to an Existing Table
- Testing For Performance

SQL intersection

Temporal Tables Introduction

- SQL Server 2016 Feature
- No triggers needed
- Many questions but answers are harder to get
- No restore needed to get the data back

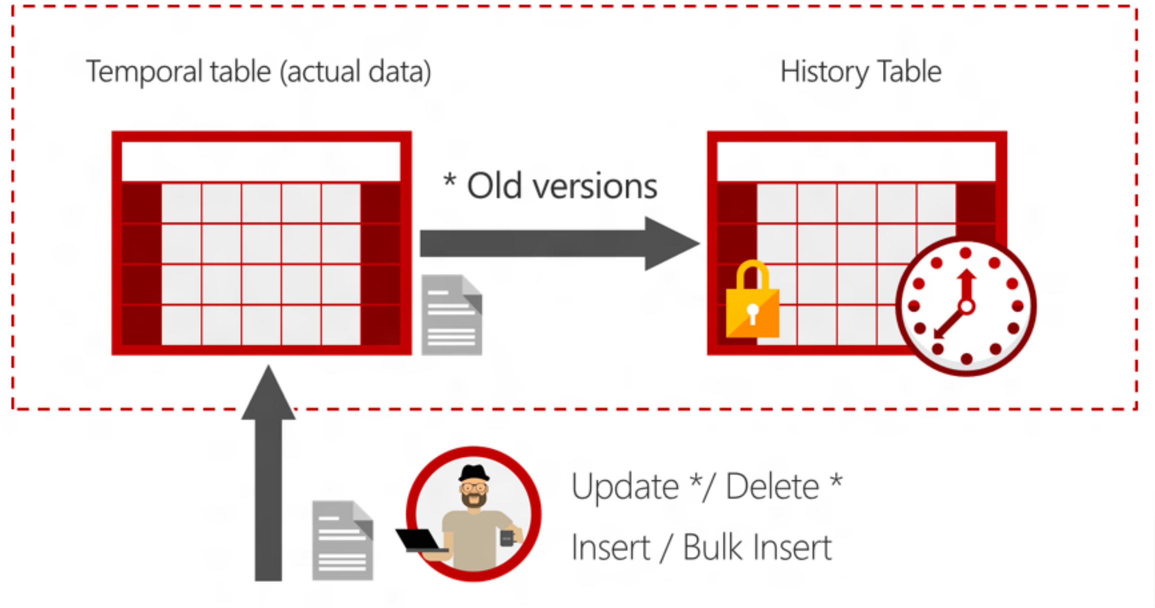
What are Temporal Tables

- **Classic Problem**
  - I have a table or tables
  - Need to know what changes have been made
  - Need to know what the data was a day ago
  - Need to have a log of changes made to the data over time
- **Solution**
  - System Versioned tables
  - Each modification is added to the history table and dated with Start and End times

Reasons to Entertain Use

- Auditing all data changes and performing data forensics when necessary
- Reconstructing state of the data as of any time in the past
- Calculating trends over time
- Maintaining a slowly changing dimension for decision support applications
- Recovering from accidental data changes and application errors

How Does it Work?



SQL intersection

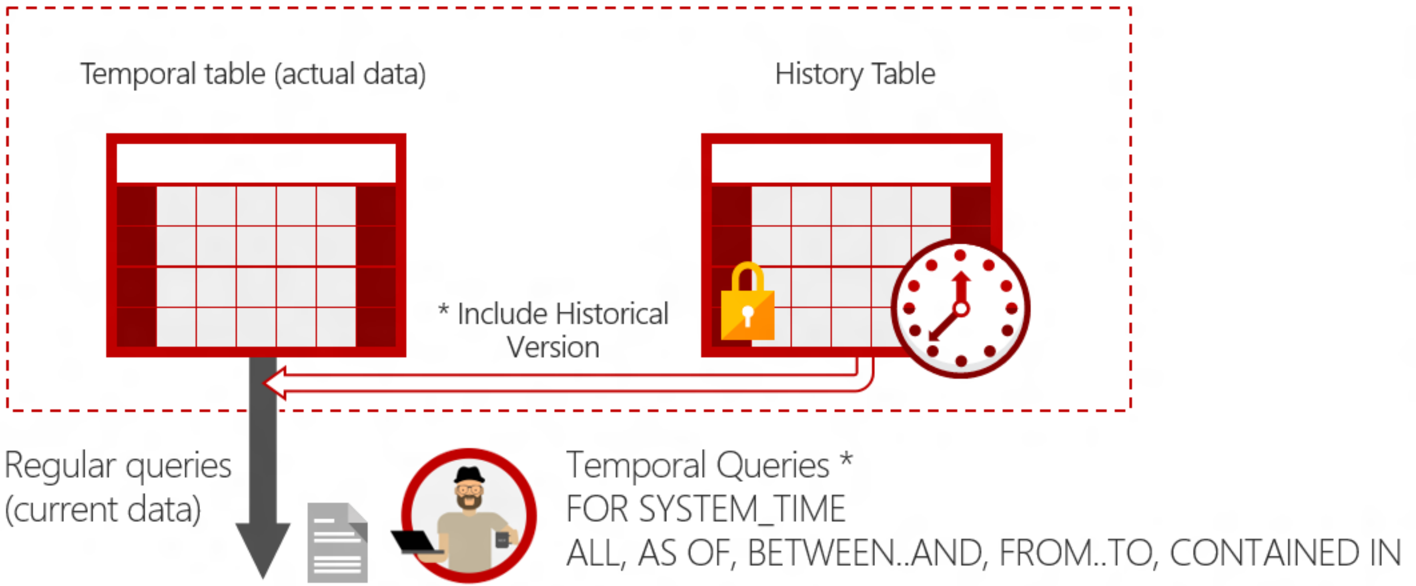
How Does it Work?

- **INSERTS**
  - Establish SysStartTime in the main table as the start time of the transaction
  - SysEndTime is assigned max value of 9999-12-31 which marks the row as Open
- **UPDATES**
  - System stores previous values of the row(s)
  - Sets SysEndTime as the start time of the transaction in UTC time (system clock) which marks the row as closed
  - Sets the main tables SysStartTime with the start time of the transaction in UTC time (System Clock)

How Does it Work?

- **DELETES**
  - Stores previous values of the row in the History table
  - Sets SysEndTime as the start time of the transaction in UTC time (system clock)
  - Marks the row as closed with a period of which the previous row was valid
  - Main table has the row removed
- **NOTE**
  - All SysStartTime, SysEndTime and Period are in DateTime2 column types
  - Dates are in UTC

Querying the History Table



Temporal table (actual data)

History Table

* Include Historical Version

Regular queries (current data)

Temporal Queries *
FOR SYSTEM_TIME
ALL, AS OF, BETWEEN..AND, FROM..TO, CONTAINED IN

Querying the History Table

- **AS OF start**
  - Beginning date to start and goes to now
- **FROM start TO end**
  - Does not include dates EXACTLY on the FROM or EXACTLY on the TO
- **BETWEEN start AND end**
  - Same as FROM/TO but INCLUDES those that are EXACTLY on the End

SQL intersection

Querying the History Table

- **CONTAINED IN (start, end)**
  - All rows Opened AND Closed in the boundaries including those EXACTLY on the boundary
- **ALL**
  - Returns UNION of rows that belong to the main table and the history table

SQL intersection

In Memory OLTP Tables

- **Only SCHEMA_AND_DATA tables supported**
- **FOR SYSTEM TIME not supported in Native Compiled SP**
- **History Table must be disk based**
- **Temporary Staging table created in memory and uses memory**
  - Data is flushed to the disk based History table asynchronously
  - sys.dm_xtp_memory_consumers
  - Can flush manually sp_xtp_flush_temporal_history @schema_name, @object_name
- **SYSTEM_VERSIONING=OFF or ALTER TABLE**
  - Staging table must be flushed to the disk based table

SQL intersection

How IMOLTP Staging Table Works

- **Table Name for the Staging table**
  - Memory_Optimized_History_Table_OBJECTID
- **Table replicates the schema in the current table + 1 BIGINT column**
  - Format for the bigint column (Change_ID[_suffix]) in case there is a column named Change_ID in the table
- **Max row size reduces by 8 bytes to 8052 because of the extra column**
- **You will NOT see this table in the Object Explorer**
- **Meta data about this is in the sys.internal_tables**

SQL intersection

Data Flush

- **Data movement starts when memory consumption of the internal staging table reaches 8% of memory consumption by the Main table**
- **Data Flush task is activated on a schedule**
  - Could be as often as 5 seconds and infrequent as 1 minute
- **One thread is used for each staging table that needs to be flushed**
- **Data flush deletes rows in staging table that are older than the oldest currently running transaction and moves them to the disk based history table**
- **Manually flush**
  - sp_xtp_flush_temporal_history @schema_name, @object_name

Data Retention Options (2017+)

- **Option to Set Retention**
  - HISTORY_RETENTION_PERIOD
- **Available Options**
  - DAYS, WEEKS, MONTHS, and YEARS
- **Retention Setting lost when setting System Versioning OFF**
- **Not specifying Retention Explicitly means INFINITE**
- **Finite Retention requires a Clustered Index on at least ValidTo**

Data Retention (2017+) Syntax

```
SELECT is_temporal_history_retention_enabled, name
FROM sys.databases

ALTER DATABASE <myDB>
SET TEMPORAL_HISTORY_RETENTION  ON
```

SQL intersection

```
CREATE TABLE dbo.WebsiteUserInfo
(
    [UserID] int NOT NULL PRIMARY KEY CLUSTERED
    ,[UserName] nvarchar(100) NOT NULL
    , [PagesVisited] int NOT NULL
    , [ValidFrom] datetime2 (0) GENERATED ALWAYS AS ROW START
    , [ValidTo] datetime2 (0) GENERATED ALWAYS AS ROW END
    , PERIOD FOR SYSTEM_TIME (ValidFrom, ValidTo)
 )
 WITH
 (
     SYSTEM_VERSIONING = ON
     (
        HISTORY_TABLE = dbo.WebsiteUserInfoHistory,
        HISTORY_RETENTION_PERIOD = 6 MONTHS
     )
 );
```

Important Notes

- A primary key must be defined.
- The table option SYSTEM_VERSIONING must be set to ON.
- Two DATETIME2 columns must be defined for the start and end date.
- LIMIT: Temporal and history table cannot be FILETABLE.
- LIMIT: INSTEAD OF triggers are not allowed. AFTER triggers are only allowed on the current table.
- LIMIT: The history table cannot have any constraints.
- LIMIT: Data in the history table cannot be modified.
- Full List: http://bit.ly/2pNEMMO

Ways to Enable Temporal History Tables

- Anonymous History Tables
- Default History with Specified Table
- Custom History Table
- Add Versioning to an Existing Table

End Result

- Table with automatic history
- No more triggers
- Data History for many use cases
- Solution Realized

Review

- What are Temporal Tables
- Reasons to Entertain Use
- How does it work?
- In Memory OLTP Use
- Anonymous Type, Default Type, Custom Type
- Adding to an Existing Table
- Testing For Performance

SQL intersection

Questions?

DBADuck
ben miller

Don’t forget to complete an online evaluation!

Temporal Tables In Depth

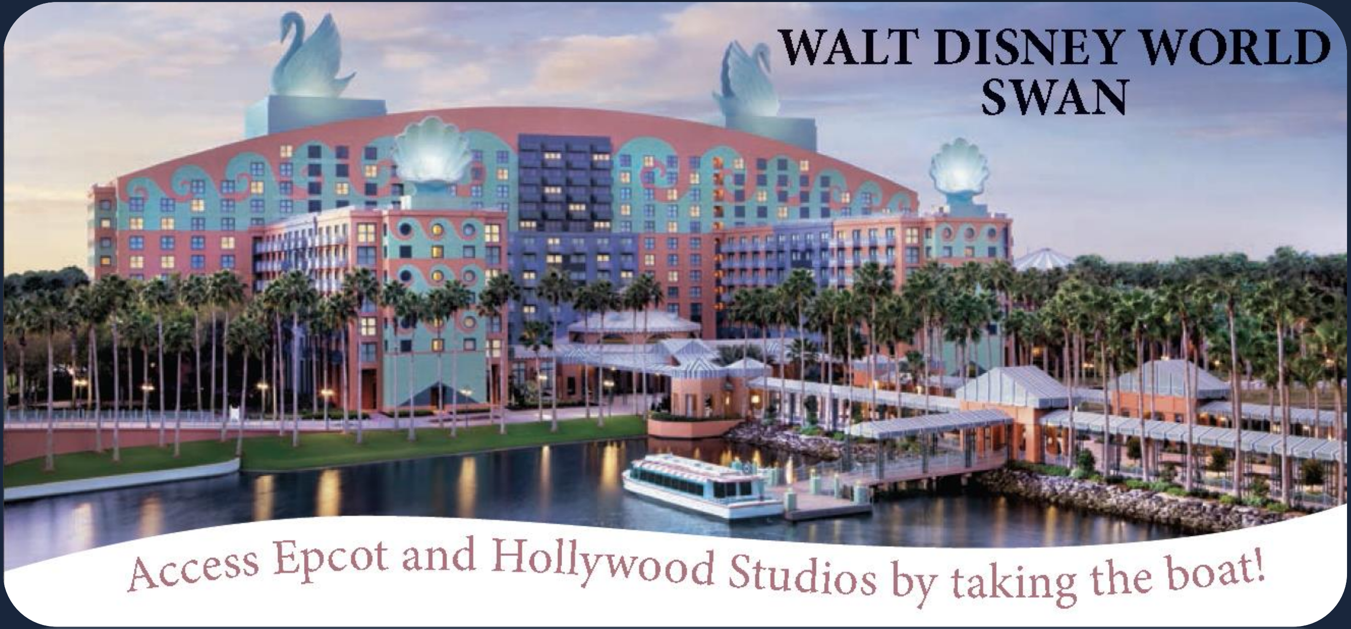
Your evaluation helps organizers build better conferences and helps speakers improve their sessions.

SQL
intersection

Thank you!

Save the Date!

www.SQLintersection.com



2019

June 9-14

We're back in Orlando!

SQL intersection

Leave the every day behind and enter a world of wonder and enchantment at the Walt Disney World® Resort. Located in the heart of the most magical place on earth, the Walt Disney World Swan and Dolphin Resort provides a truly extraordinary backdrop for our event! Beautiful tropical landscaping, tranquil waterways, and classic art and architecture work together to create a stunning landmark!