# Chad Green

Director of Software Development, ScholarRx


chadgreen@chadgreen.com

chadgreen.com

ChadGreen

ChadwickEGreen


Microsoft® MVP Most Valuable Professional

{Code PaLOUsa}

IT'S SOFTWARE DEVELOPMENT MADNESS

# Our Agenda

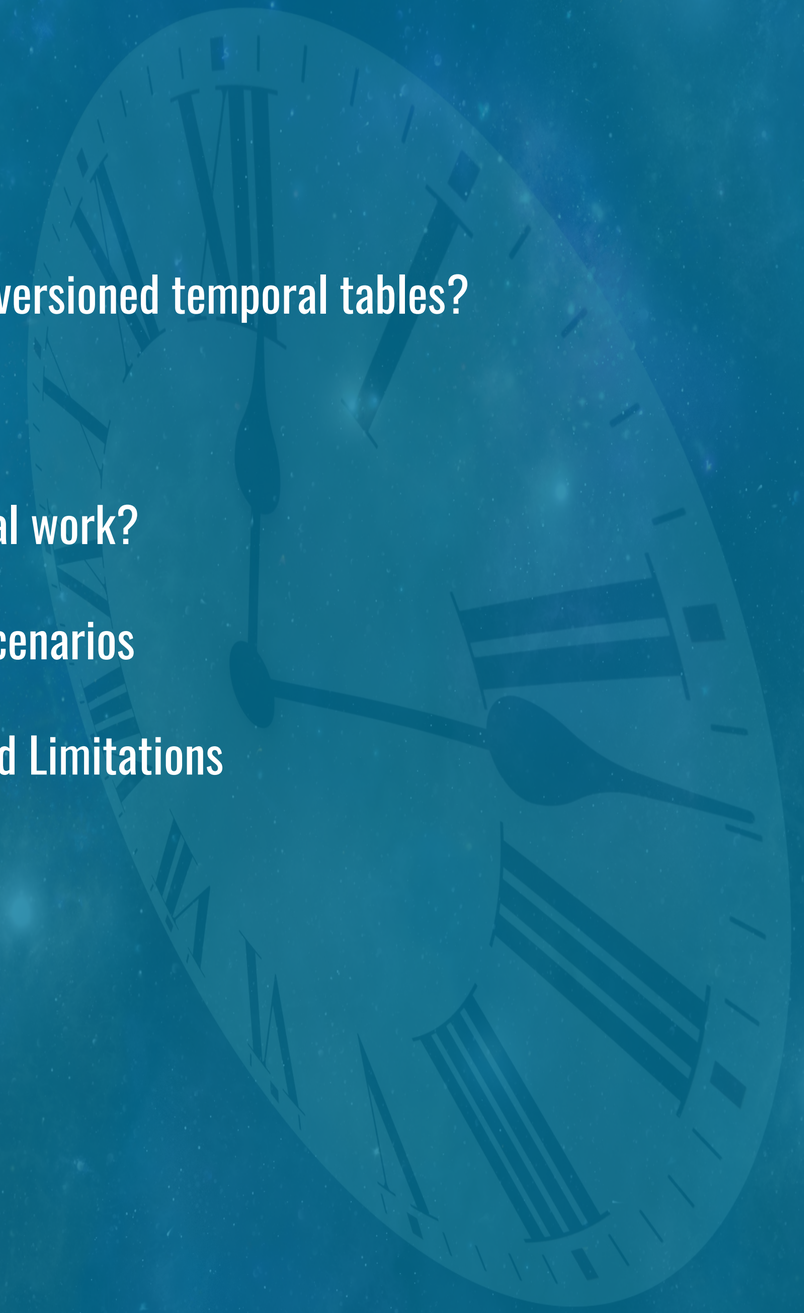
What are system-versioned temporal tables?

Why Temporal?

How does temporal work?

Common Usage Scenarios

Considerations and Limitations



# What are system-versioned temporal tables?

- Special kind of table used to track changes over time
- Called System-Versioned because the system handles them
- Rule of Two: 2 tables and 2 new columns
  - Two period columns of type datetime2 (SystemStartTime and SystemEndTime)
  - History table with same column definitions
- Availability: All versions of SQL Server starting with 2016

# Why temporal?

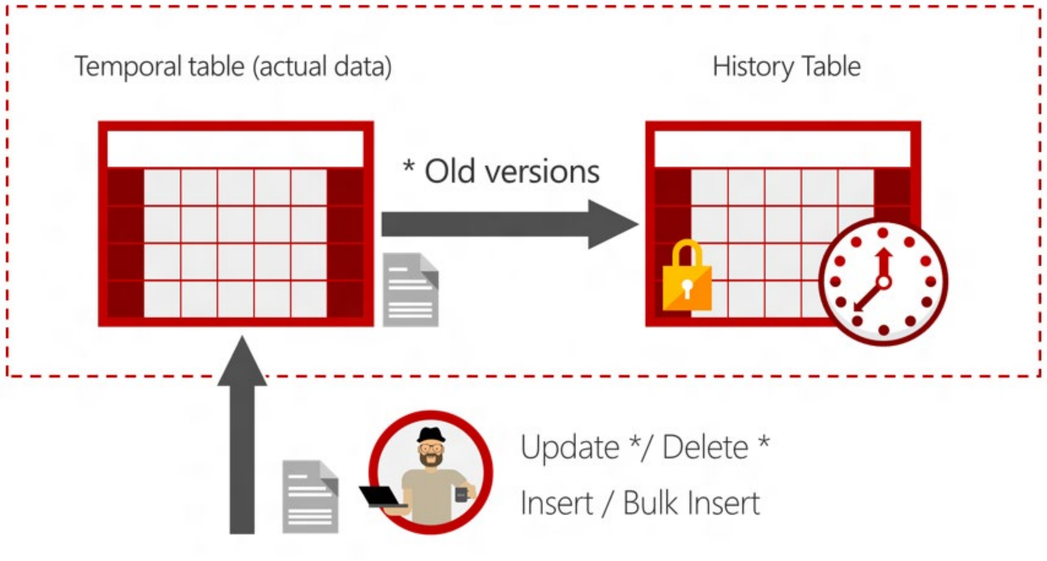
- Auditing all data changes and performing data forensics when necessary
- Reconstructing state of the data as of any time in the past
- Calculating trends over time
- Maintaining a slowly changing dimension for decision support applications
- Recovering from accidental data changes and application errors

# Why temporal?

- Built-in functionality better than roll-your-own solution:
  - Immutable history
  - New syntax for time-travelling queries
  - Better DML performance
  - Minimal maintenance costs

# How does temporal work?

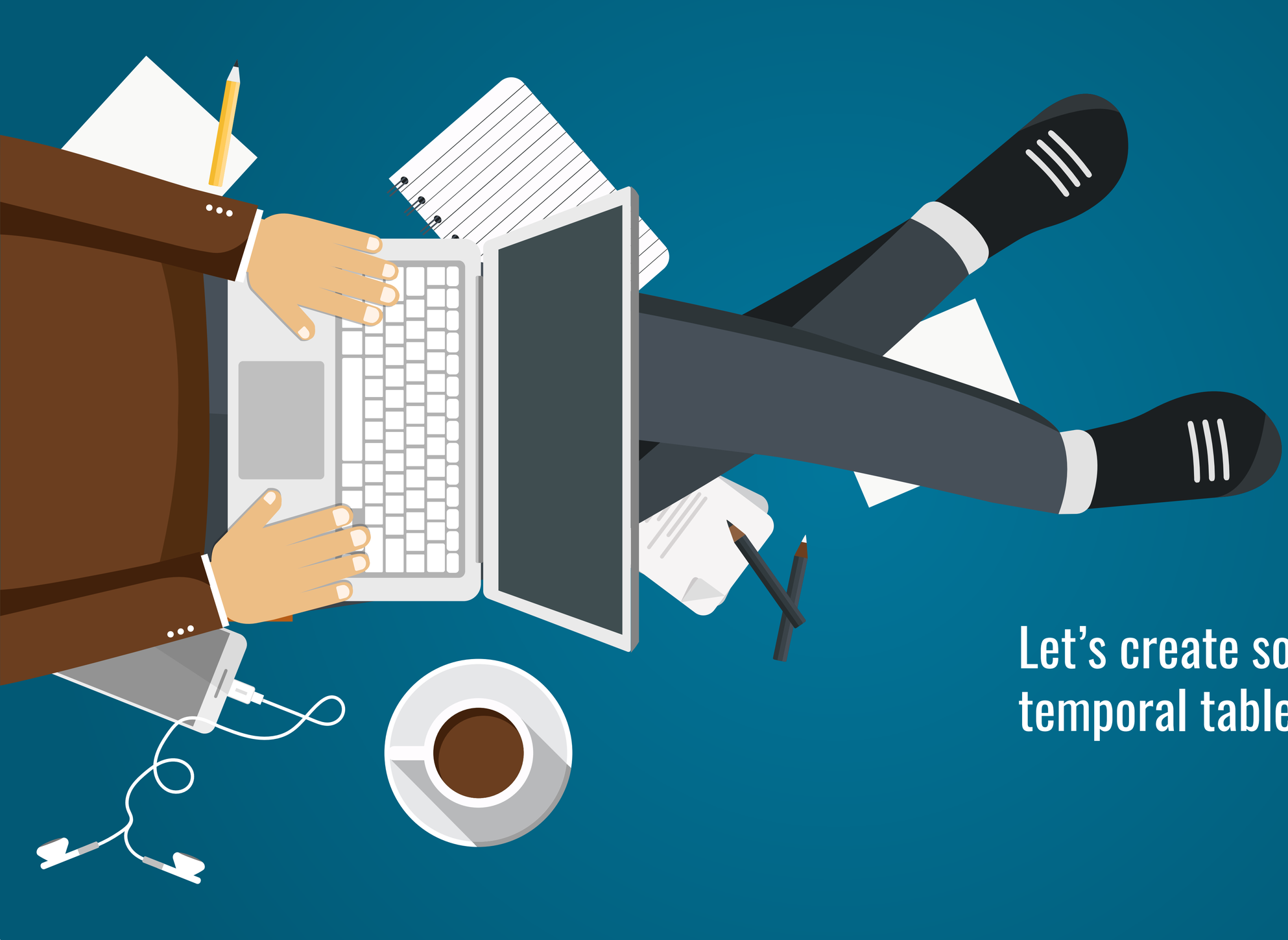
- Period start column (SysStartTime)
- Period end column (SysEndTime)
- The current table contains the current value for each row
- The history table contains each previous value for each row, if any, and the start time and time for the period for which it was valid



# How does temporal work?

- Three ways to create a system-versioned temporal table
  - Temporal table with an anonymous history table
  - Temporal table with a default history table
  - Temporal table with a user-defined history table created beforehand

- Things to consider
  - System-versioned temporal table must have:
    - Primary Key
    - PERIOD FOR SYSTEM_TIME with two datetime2 columns
  - Period columns are always non-nullable
  - History table is created as a rowstore table



Let’s create some temporal tables

# How does temporal work?

- INSERT
  - SysStartTime value set to begin time of current transaction time
  - SysEndTime value set to maximum value of 9999-12-31

# How does temporal work?

- UPDATE
  - History Table: Previous value stored & SysEndTime set to current transaction time
  - Current Table: Row is updated with its new value & SysStartTime set to current transaction time

# How does temporal work?

- DELETE
  - History Table: SysEndTime set to current transaction time
  - Current Table: Row is removed

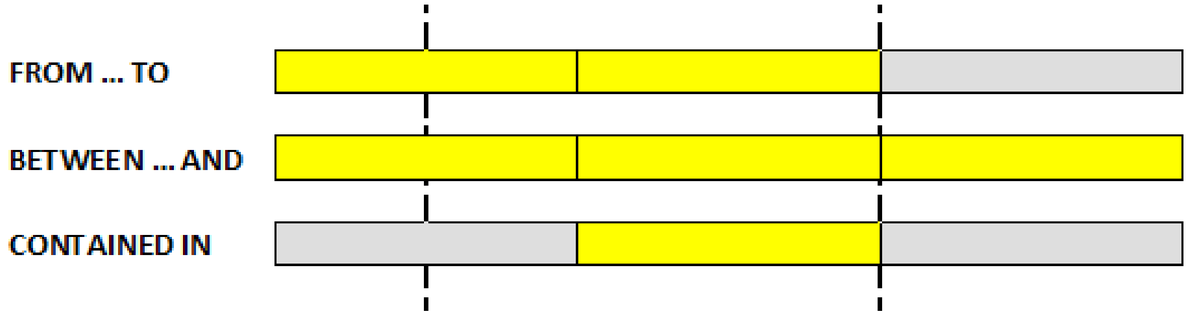
# How does temporal work?

- MERGE
  - Exactly as if up to 3 statements (INSERT, UPDATE, DELETE) executed, depending on what is specified in the MERGE statement

INSERTs, UPDATEs, and DELETEs; oh my!

# How do I query temporal data?

- New **FOR SYSTEM_TIME** clause added to the **FROM** clause
- For SYSTEM_TIME sub-clauses
  - AS OF <date_time>
    - Rows containing the values current at specified point in time
  - FROM <start_date_time> TO <end_date_time>
    - All row versions that were active within the specified time range
  - BETWEEN <start_date_time> AND <end_date_time>
    - All row versions that became active on the upper boundary
  - CONTAINED IN (<start_date_time>, <end_date_time>)
    - All row versions that were opened and closed within specified time range
  - ALL



Roads? Where we are going we don’t need roads.

# Common Usage Scenarios

- Data Audit
- Point in Time Analysis (Time Travel)
- Anomaly Detection
- Slowly-Changing Dimensions
- Repairing Row-Level Data Corruption

# Temporal Table Considerations and Limitations

- Current table must have primary key; history table cannot have primary key
- SysStartTime and SysEndTime columns must be of datetime2
- Must specify schema when specifying history table name
- By default, the history table is PAGE compressed
- No FILESTREAM data types
- Blob data types incur significant storage costs
- History tables must be created in same database

# Temporal Table Considerations and Limitations

- History tables cannot have constraints
- Indexed views not supported on top of temporal queries
- INSERT and UPDATE statements cannot reference the SYSTEM_TIME period columns
- TRUNCATE TABLE is not supported while SYSTEM_VERSIONING is ON
- Direct modification of the data in a history table is not permitted
- INSTEAD OF triggers are not permitted
- Regular queries only affect data in the current table

# Chad Green

Director of Software Development, ScholarRx


chadgreen@chadgreen.com

chadgreen.com

ChadGreen

ChadwickEGreen


Microsoft® MVP Most Valuable Professional