SQL Temporal Tables

```sql
CREATE TABLE dbo.Person (
  [Id] UNIQUEIDENTIFIER NOT NULL,
  [Name] NVARCHAR(100) NOT NULL,
  [ValidFrom] DATETIME2 GENERATED ALWAYS AS ROW START,
  [ValidTo] DATETIME2 GENERATED ALWAYS AS ROW END,
  PERIOD FOR SYSTEM_TIME(ValidFrom, ValidTo),
  CONSTRAINT [PK_Person] PRIMARY KEY CLUSTERED (Id)
)
WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = dbo.PersonHistory));
```

- A new record is **inserted** to the temporal table. The period of validity is now till 9999-12-31.
- When the record is **updated**, the original record is copied to the history table. The period of validity is updated in the temporal table and in the history table.
- When the record is **deleted**, the row is removed from the temporal table and its value is copied to the history table. The ValidTo value is de timestamp when the record got removed.

	Id	Name	ValidFrom	ValidTo	
✨ Insert	3FA9D3A3-...	Alice Original	2024-03-02	9999-12-31	
✨ Update	3FA9D3A3-...	Alice Updated	2024-03-20	9999-12-31	
✨ Delete					

Temporal (current) table

Id	Name	ValidFrom	ValidTo	
3FA9D3A3-...	Alice Original	2024-03-02	2024-03-20	
3FA9D3A3-...	Alice Updated	2024-03-20	2024-03-26	

History table

timdeschryver.dev/bits

SQL Temporal Tables: Creation

```sql
CREATE TABLE dbo.Person (
    [Id] UNIQUEIDENTIFIER NOT NULL,
    [Name] NVARCHAR(100) NOT NULL,
    [ValidFrom] DATETIME2 GENERATED ALWAYS AS ROW START,
    [ValidTo] DATETIME2 GENERATED ALWAYS AS ROW END,
    PERIOD FOR SYSTEM_TIME(ValidFrom, ValidTo),
    CONSTRAINT [PK_Person] PRIMARY KEY CLUSTERED (Id)
)
WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = dbo.PersonHistory));
```

Creates the temporal (current) table

Creates the history table

timdeschryver.dev/bits

SQL Temporal Tables: What

Temporal tables (also known as system-versioned temporal tables) are a database feature that brings built-in support for providing information about data stored in the table at any point in time, rather than only the data that is correct at the current moment in time.

Because it keeps a full history of data changes, it allows for easy point-in-time analysis.

This is useful for:

- Auditing all data changes and performing data forensics when necessary
- Reconstructing state of the data as of any time in the past
- Recovering from accidental data changes and application errors
- Insights to data trends

SQL Temporal Tables: How

- A new record is **inserted** to the temporal table. The period of validity is now till 9999-12-31.
- When the record is **updated**, the original record is copied to the history table.
  The period of validity is updated in the temporal table and in the history table.
- When the record is **deleted**, the row is removed from the temporal table and its value is copied to the history table. The ValidTo value is de timestamp when the record got removed.

Id	Name	ValidFrom	ValidTo
3FA9D3A3-...	Alice Original	2024-03-02	9999-12-31
3FA9D3A3-...	Alice Updated	2024-03-20	9999-12-31

Temporal (current) table

Id	Name	ValidFrom	ValidTo
3FA9D3A3-...	Alice Original	2024-03-02	2024-03-20
3FA9D3A3-...	Alice Updated	2024-03-20	2024-03-26

History table

timdeschryver.dev/bits

SQL Temporal Tables: Query

```
SELECT *
FROM dbo.Person
FOR SYSTEM_TIME [ALL] | [AS OF] | [FROM-TO] | [BETWEEN] | [CONTAINED IN]
```

Query the tables manually, or use the `FOR SYSTEM_TIME` clause the query both tables with a single query.
Use the subclause to specify the range.

timdeschryver.dev/bits