À la recherche du temps passé…

SQL et les tables temporelles

Frédéric Brouard / Arian Papillon

MVP Data Platform

sqlpro@sqlspot.com / a.papillon@datafly.fr

Microsoft® MVP Most Valuable Professional

MS Cloud Summit Paris

Agile.Net – aOS – AZUG FR – CMD - GUSS

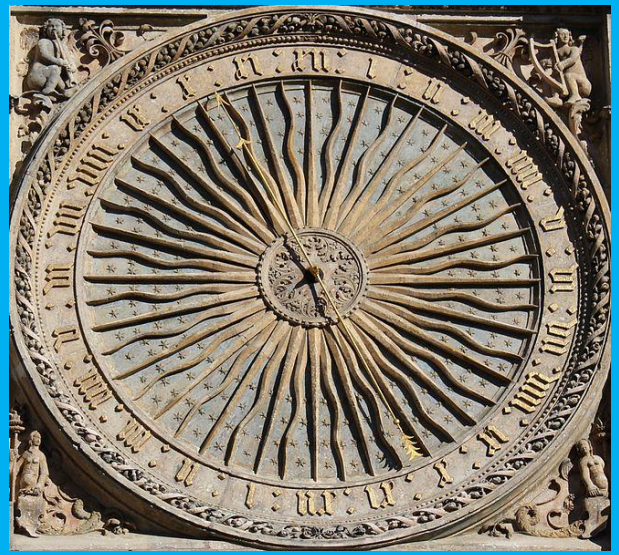
- Frédéric Brouard
  sqlpro@sqlspot.com

- Arian Papillon
  a.papillon@datafly.fr

ORSYS formation

MS Cloud Summit Paris 2017

Agile.Net – aOS – AZUG FR – CMD - GUSS



LE CONCEPT

Normes et définitions

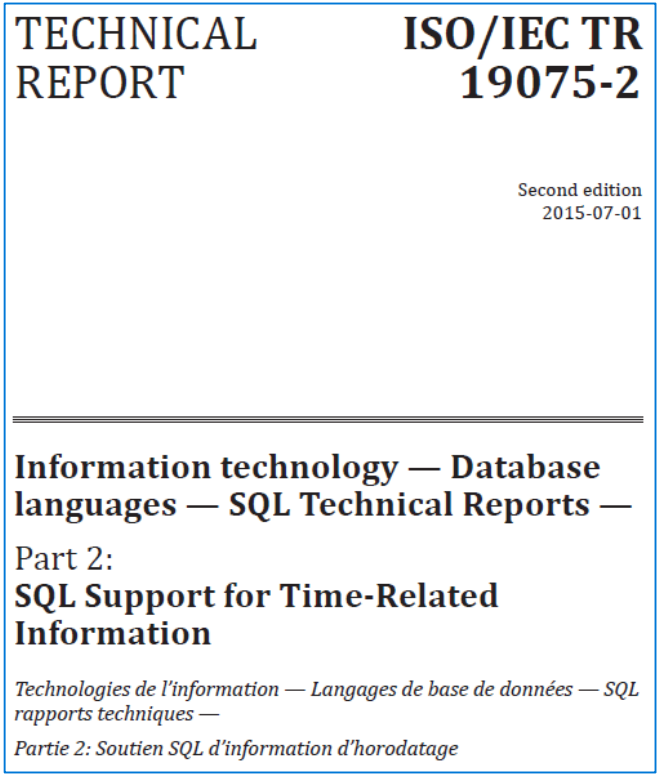
Tables temporelles avec SQL Server

Norme

ISO/IEC TR 19075-2:2015

Information technology -- Database languages -- SQL Technical Reports

Part 2: SQL Support for Time-Related Information


TECHNICAL REPORT

ISO/IEC TR 19075-2

Second edition
2015-07-01

Information technology — Database languages — SQL Technical Reports —

Part 2:
SQL Support for Time-Related Information

Technologies de l'information — Langages de base de données — SQL rapports techniques —

Partie 2: Soutien SQL d'information d'horodatage

Norme, définitions

- **Valid Time** : période de validité de la donné, que nous traduirons par "temps fonctionnel"
- **Transaction Time** : période de stockage de la donnée dans la base, que nous traduirons par "temps opérationnel"
- **Bitemporal data** : combine les deux temporalités

Convention

Par distinction, nous parlerons de :

- **Table d'historisation** : la table technique, ne contenant que les données passées, associée à une table utilisateur
- **Table temporalisée** : la table utilisateur, contenant les données actuelle, ayant une table d'historisation associée
- *Le terme "**table temporelle**" parlant du concept, rassemble les deux tables…*

Définitions

Périodes : exprimées par des intervalles de temps (PERIOD), définis par deux colonnes :

- une date/heure de début,
- une date/heure de fin,

valorisé en temps universel (UTC).

Pour SQL Server, ces deux colonnes sont de type DATETIME2(n).

La précision n est celle que vous souhaitez.

Périodes

ATTENTION : SQL considère que les intervalles temporels sont fermé à droite et ouvert à gauche :

[début, fin [

Afin qu'il y ait continuité sans recoupement.

La fonction OVERLAPS procède ainsi.

Définitions

SQL Server automatise le temps opérationnel (Transaction Time).

C'est à vous d'ajouter le temps fonctionnel si vous le désirez !

Par convention pour les dates inconnues, les valeurs sont les suivantes :

- dans le futur, valeur 9999-12-31 23:59:59.9999999
- dans le passé, valeur 0001-01-01 00:00:00

Conclusion

La table contient toutes les versions de toutes les lignes insérées, modifiées et supprimées à chaque mise à jour.

Vous pouvez revenir à tout moment sur les valeurs telles qu'elles étaient à un moment ou une période du passé

Plus besoin de sauvegarde !

MÉCANISME

La mise en œuvre

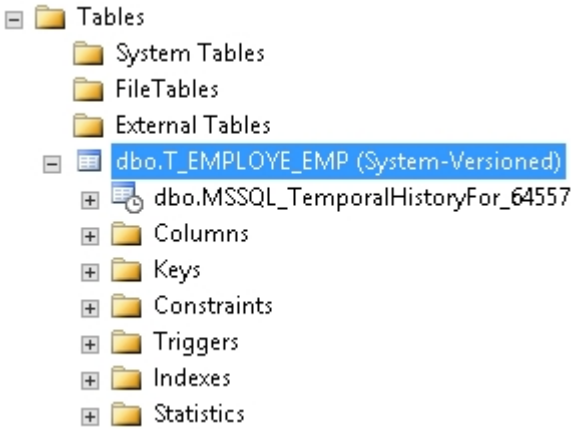
Tables temporelles avec SQL Server

Mise en place

Dès mise en place d'une table temporelle, il y a création de la table d'historisation.

Nom par défaut : **`MSSQL_TemporalHistoryFor_`** suivi de l'id de la table maître

- Lecture seule
- Données compressées.
- Aucune contrainte



Mise en place

1 index clustered (fin, début) pour la table d'historique

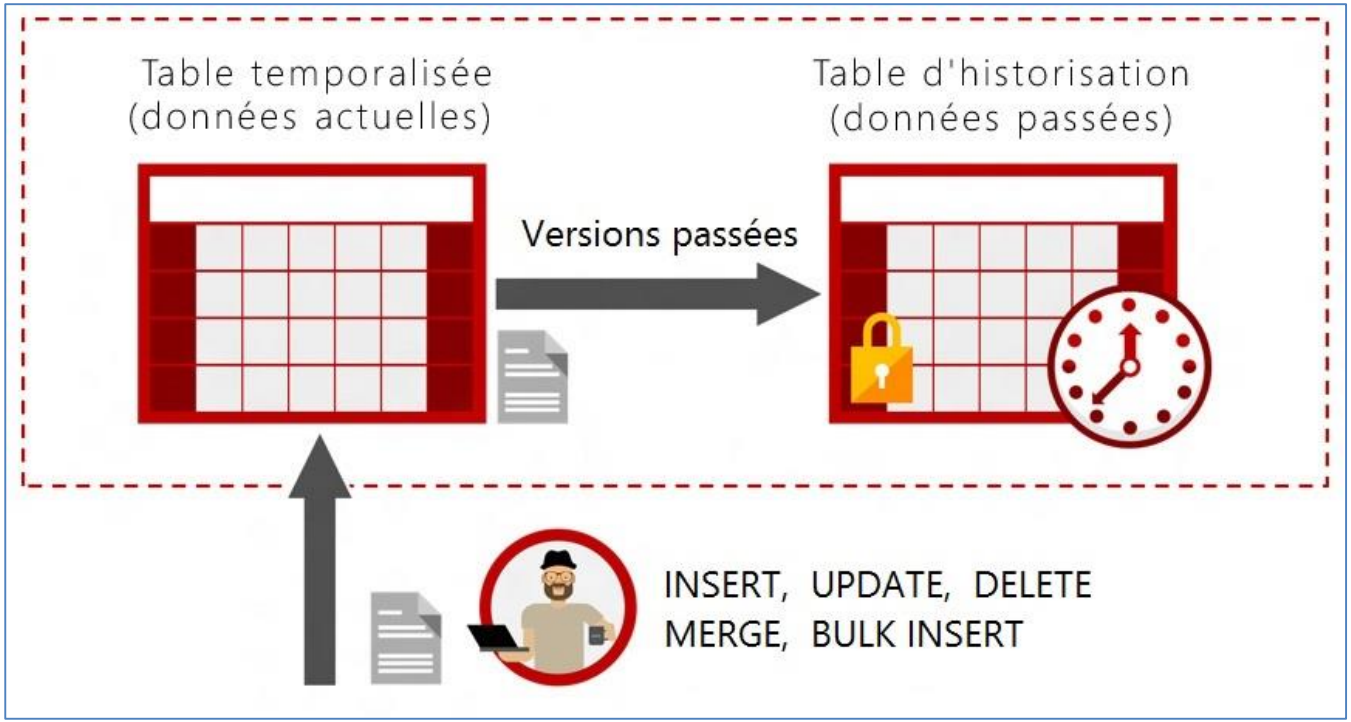
Ce dernier peut être transformé en index "columnstore" !

Vous pouvez ajouter des index.

La table peut être mise en cloud ("stretch table")

On peut la renommer, changer son schéma.

Mise à jour et historisation



Mise à jour et historisation

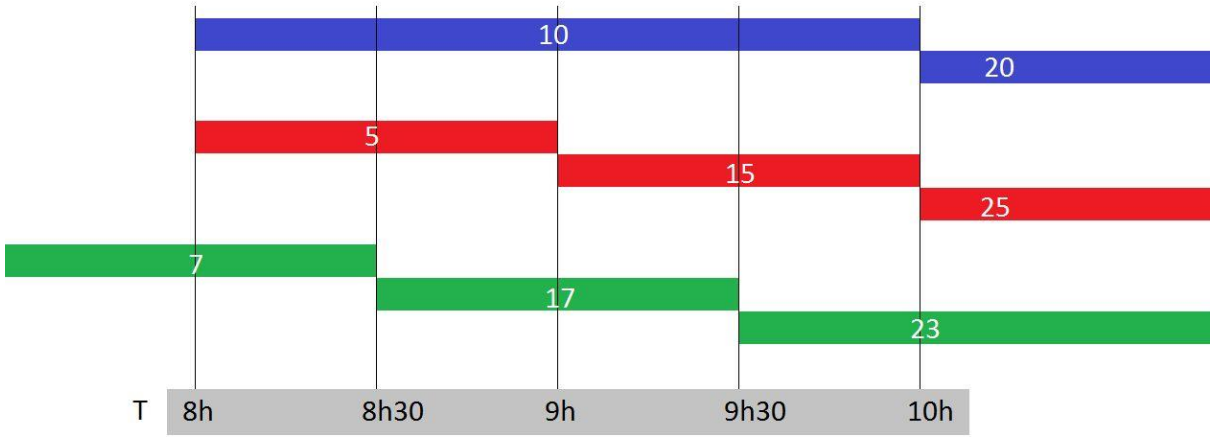
Commande	Table temporalisée	Table d'historisation
INSERT	début = date heure UTC INSERT fin = 9999-12-31	
UPDATE	début = date heure UTC UPDATE fin = 9999-12-31	ajout d'une nouvelle ligne avec les anciennes valeur début = valeur originale du début fin = date heure UTC actuelle
DELETE	*La ligne est supprimée*	ajout d'une nouvelle ligne avec les anciennes valeur début = valeur originale du début fin = date heure UTC DELETE

Interrogation temporelle

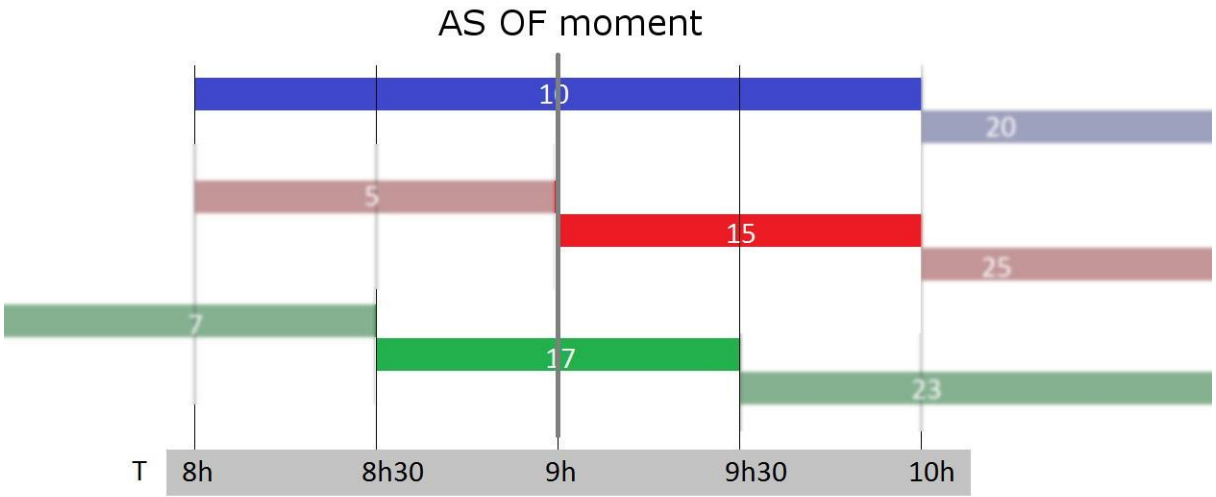
Opérateur	Paramètre	Description
AS OF …	<date_heure>	données telles qu'elles étaient à ce moment
FROM …	<debut_date_heure> **TO** <fin_date_heure>	Période située en tout ou partie dans l'intervalle : […] (fermé)
BETWEEN	<debut_date_heure> **AND** <fin_date_heure>	Période située en tout ou partie dans l'intervalle [… [(ouvert à droite)
CONTAINED IN	(<debut_date_heure> **,** <fin_date_heure>)	Période située en totalité dans l'intervalle […] (fermé)
ALL		Toutes les données

Introduit par FOR SYSTEM_TIME (clause FROM)

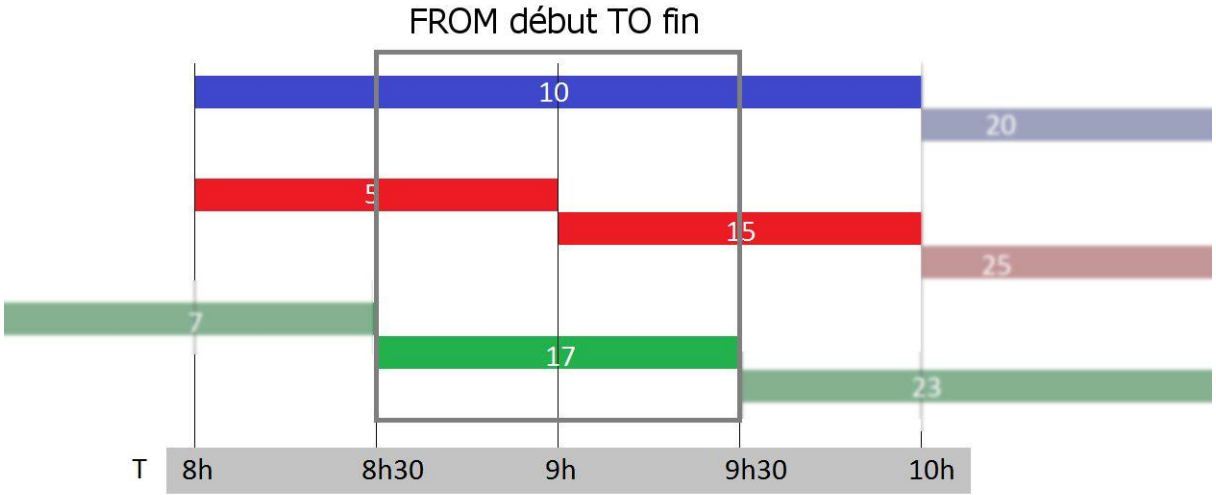
ALL



AS OF 9h

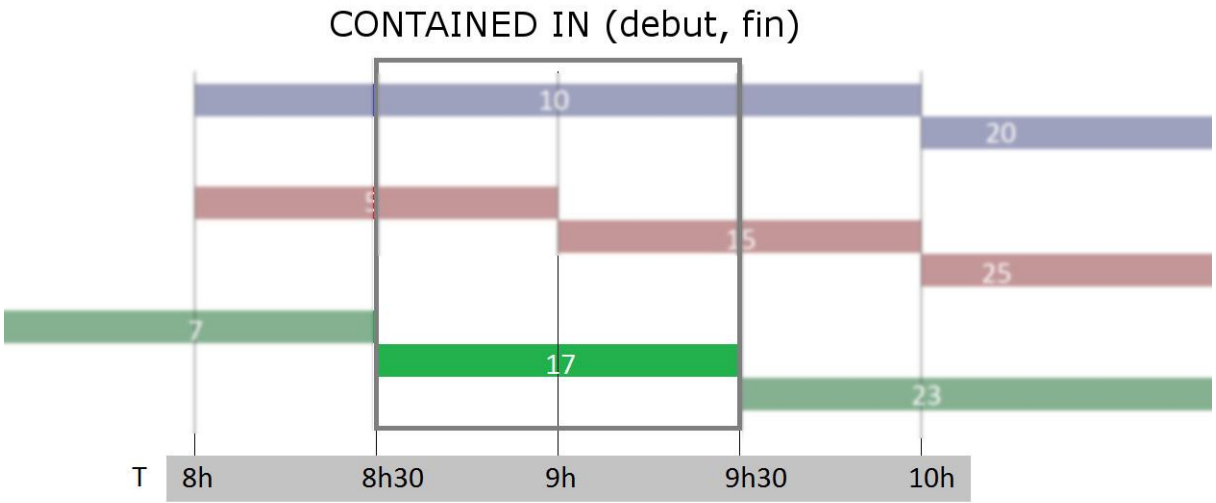


FROM 8h30 TO 9h30



BETWEEN 8h30 AND 9h30

CONTAINED (8h30, 9h30)



SYNTAXES

Les commandes SQL

Tables temporelles avec SQL Server

Colonnes

```
nom_col_début DATETIME2
    GENERATED ALWAYS AS ROW START HIDDEN NOT NULL,

nom_col_fin DATETIME2
    GENERATED ALWAYS AS ROW END HIDDEN NOT NULL,

PERIOD FOR SYSTEM_TIME (nom_col_début, nom_col_fin)
```

HIDDEN : (facultatif) pour ne pas exposer la colonne

Colonnes

nom_col_début :

- valeur par défaut => SYSUTCDATETIME()

nom_col_fin :

- Valeur par défaut => '9999-12-31T23:59:59.9999999'

Vous pouvez spécifier ces valeurs par défaut explicitement dans la définition des colonnes de temporalisation

Historisation

Après la définition de table :

```
WITH ( SYSTEM_VERSIONING = ON
[ ( HISTORY_TABLE = nom_table_historisation
   [, DATA_CONSISTENCY_CHECK = { ON | OFF } ] ) ] )
```

- La table temporalisée doit avoir une clef primaire
- La table d'historisation peut résider dans un schéma SQL different
- DATA_CONSISTENCY_CHECK peut être utilisé pour verifier le chainage des temps si l'on récupère une table d'historique

Suppression table temporelle

DROP TABLE : ne fonctionne pas (car deux tables sont liées).

Procéder comme suit :

- Arrêter la temporalisation
- Supprimer la table temporalisée
- Supprimer la table d'historique

Suppression table temporelle

```
ALTER TABLE nom_table_temporalisée
    SET (SYSTEM_VERSIONING  = OFF);
DROP TABLE nom_table_temporalisée;
DROP TABLE nom_table_historique;
```

Si vous ne supprimez que l'historique et ne voulez plus les colonnes de temporalisation :

```
ALTER TABLE nom_table_temporalisée
    DROP PERIOD FOR SYSTEM_TIME;
ALTER TABLE nom_table_temporalisée
    DROP COLUMN col_debut, col_end
```

Indexation table historique

La table d'historique est un index clustered bâtit sur les colonnes début et fin (1).

Transformation de l'index clustered en columnstore clustered :

```
CREATE CLUSTERED COLUMNSTORE INDEX nom_index
ON nom_table_historisation
WITH (DROP_EXISTING = ON,
      COMPRESSION_DELAY = 0);
```

Migration table historique

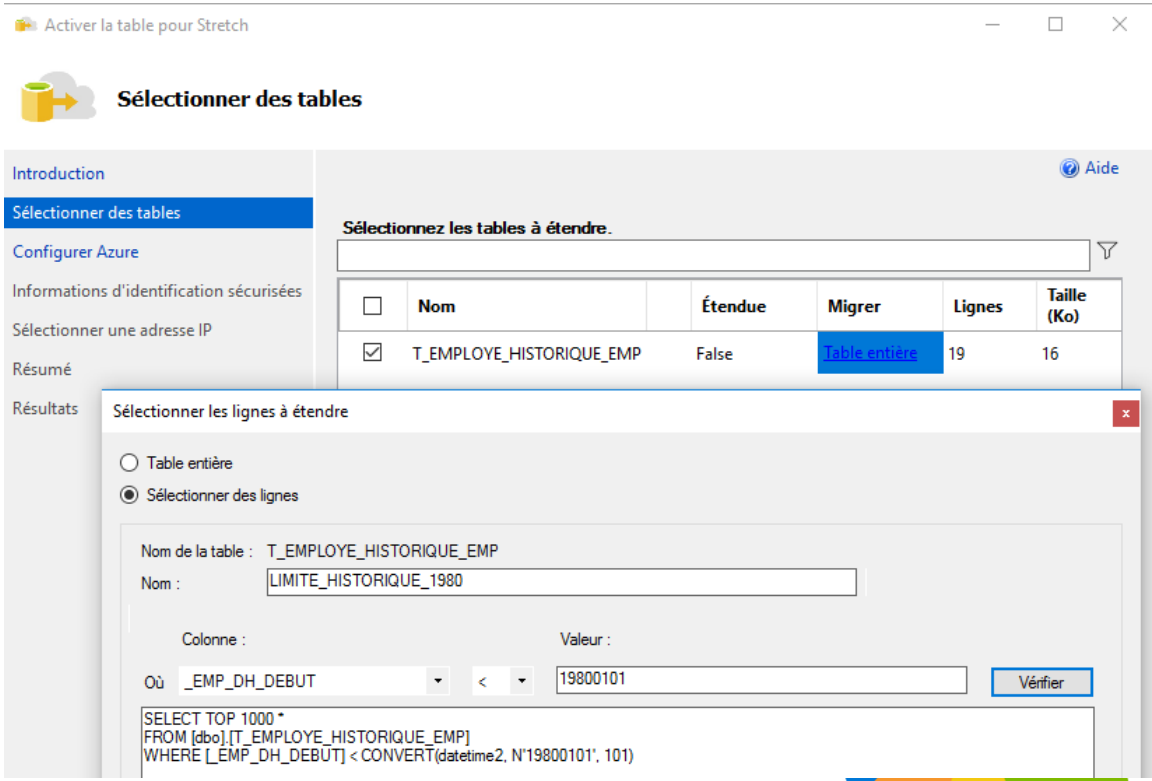
L'index clustered peut être déplacé sur un « storage » différent :

```
CREATE CLUSTERED INDEX nom_index_historique
    ON nom_table_historique( <liste_colonne_clef> )
    WITH (DROP_EXISTING = ON)
    ON nom_filegroup;
GO
```

Table historique en « stretch »

Vous pouvez placer une partie de l'historique dans le cloud Azure via le concept de « stretch table »…

Utilisez l'assistant pour ce faire

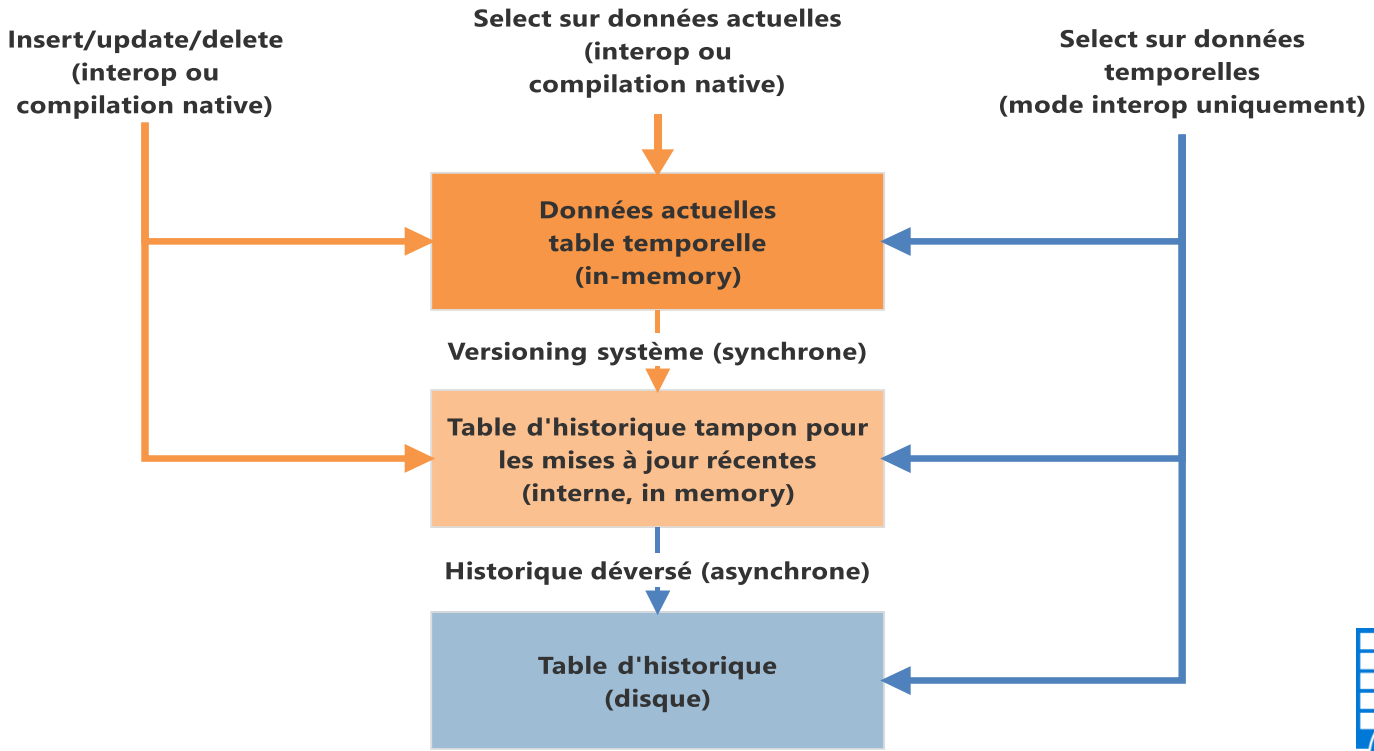


Historisation et table « in memory »

Une table « in memory » durable peut être une table temporelle

- La table d'historique est sur disque mais SQL Server crée une table interne in-memory qui sert de tampon (*staging*) pour les mises à jour
- Cette table tampon est déversée automatiquement dans la table d'historique de manière asynchrone

Historisation et table « in memory »



Historisation et table « in memory »

Flush de la table tampon :

- Dès que la consommation de mémoire atteint 8% de celle de la table temporelle (maintien < 10%)
- On peut forcer avec la procédure sp_xtp_flush_temporal_history

On peut ajouter des index à la table tampon in-memory : traceflag 10316

Gérer la rétention

Manuel :

- Désactiver le versionnement;
- Purger la table d'historisation;
- Réactiver le versionnement;

Possibilité d'utiliser le partitionnement.

DEMO !

C’est parti… … mon kiki !

Un peu plus loin

Norme SQL *Support for Time-Related Information* :

- Tables « bi-temporelles » (+ Valid Time)
- Opérateurs de comparaison d'intervalles
- Contraintes de non chevauchement
- UPDATE de « PORTION » de période

Non encore supportées par SQL Server…

Mais parfois nécessaires !

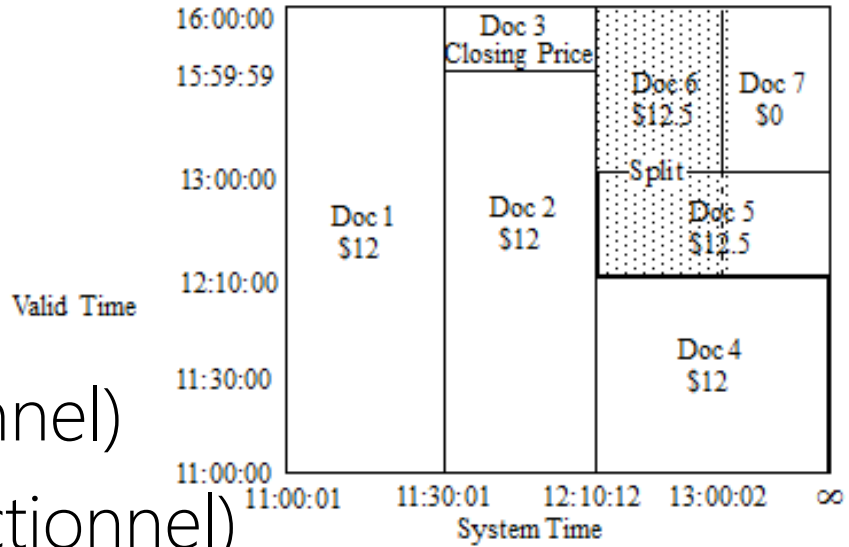
Un peu plus loin

Tables « bi-temporelles »

Deux axes de temps :

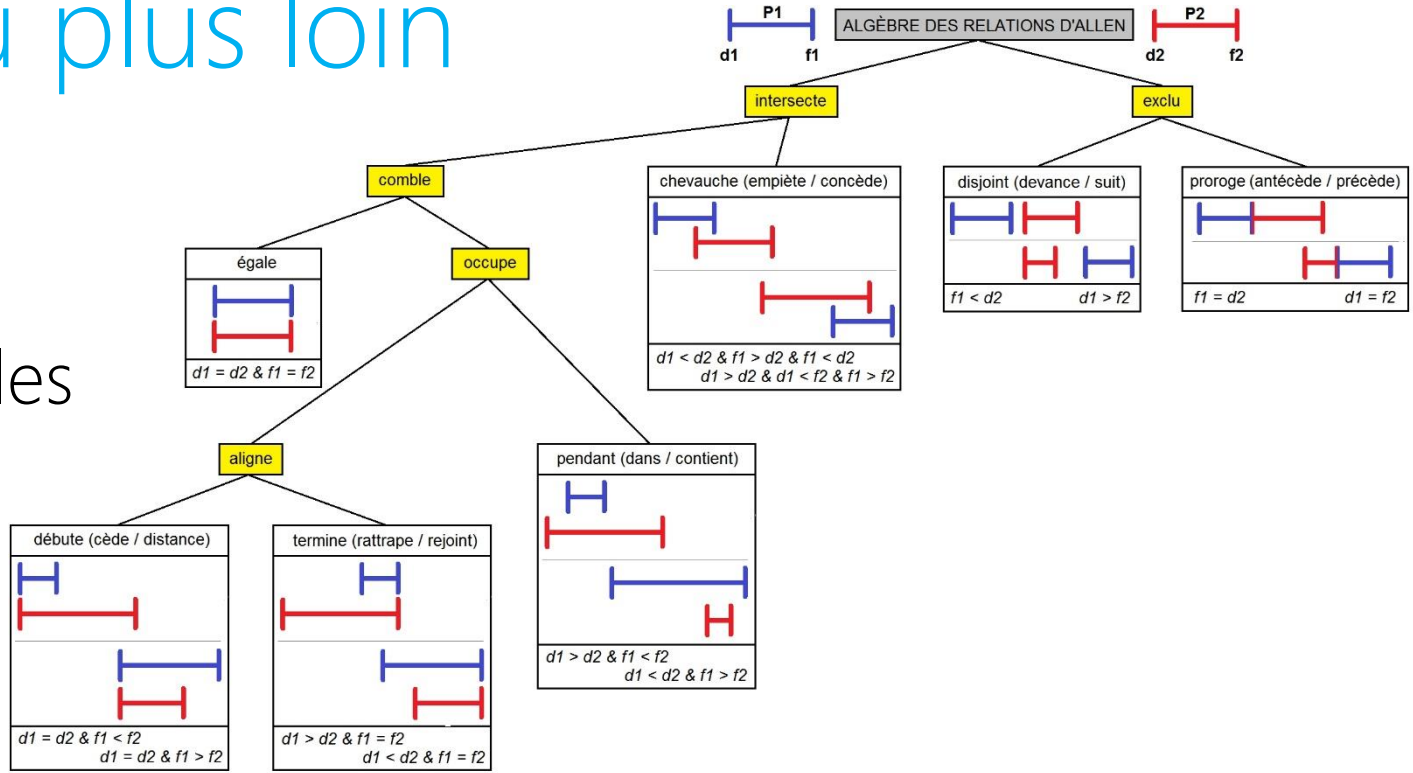
- Valid time (temps opérationnel)
- System time (temps transactionnel)

Rajouter les colonnes et les contraintes



Un peu plus loin

Algèbre d'Allen sur les intervalles



Un peu plus loin

Algèbre des intervalles : la norme SQL propose les opérateurs :

- OVERLAPS
- EQUALS
- CONTAINS
- PRECEDES
- SUCCEED
- IMMEDIATLY PRECEDES
- IMMEDIALTLY SUCCEED

Facile à implémente en Transact SQL

Un peu plus loin

Clef primaire avec non chevauchement relatif :

```
CREATE TABLE T_EMPLOYE_EMP
(EMP_ID       INTEGER NOT NULL,
 EMP_NOM      VARCHAR(30),
 EMP_SALAIRE DECIMAL(5,2),
 EMP_DEBUT    DATE NOT NULL,EMP_FIN      DATE NOT NULL,
 PERIOD FOR EMP_PERIOD_OPERATION (EMP_DEBUT, EMP_FIN),
 PRIMARY KEY (EMP_ID,
              EMP_PERIOD_OPERATION WITHOUT OVERLAPS));
```

Utilisez un trigger.

Un peu plus loin

Certaines périodes peuvent nécessiter une contrainte de chaînage.

- Pas de recoupement
- Pas de trou

Rend les mises à jours complexes

Faire à l'aide de triggers et/ou procédure

Un peu plus loin

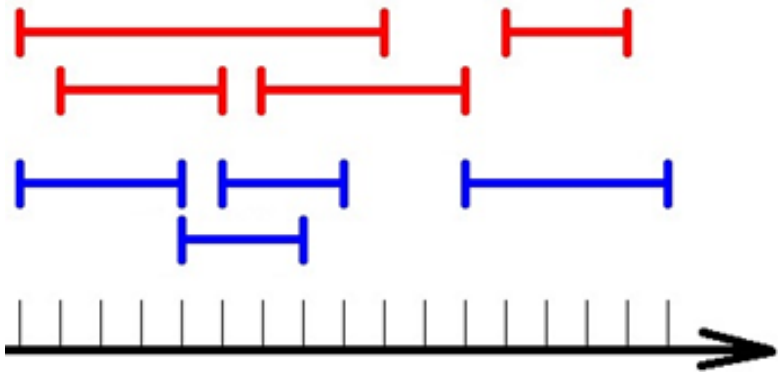
Modification dans une période

```
UPDATE T_EMPLOYE_EMP
    FOR PORTION OF EMP_PERIOD_OPERATION
        FROM DATE '2002-01-01' TO DATE
'2003-01-01'
    SET EMP_SALAIRE = 3000
    WHERE EMP ID = 100;
```

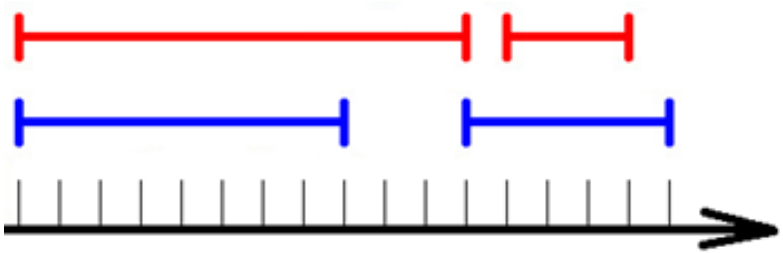
Ceci rajoute une période opérationnelle et raboute…

Opérateur COLLAPSE

COLLAPSE permet de synthétiser, fusionner des périodes recouvrantes ou chainées (raboutantes).
C'est une fonction d'agrégation.

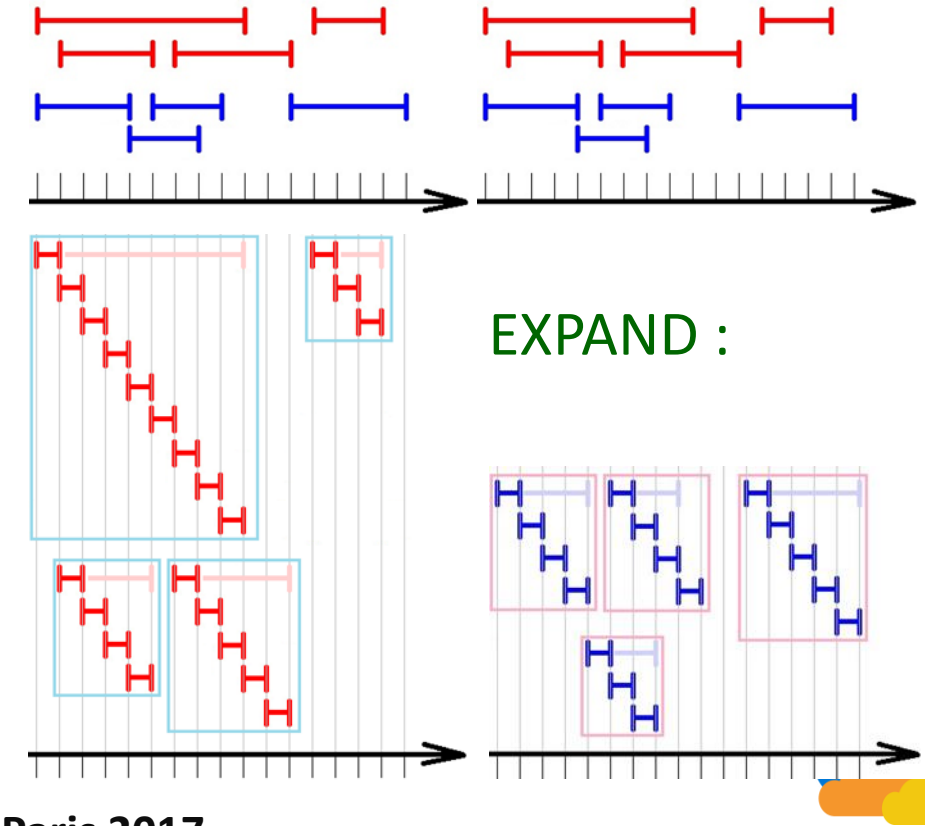


COLLAPSE :



Opérateur EXPAND

EXPAND permet de créer des nouvelles périodes à la granularité temporelle voulue.



Opérateur COLLAPSE & EXPAND

Ces opérateurs ne sont pas implémentés.

Possibilité de les programmer sous forme

- Transact SQL
- SQL CLR.

Bibliographie

Norme ISO :

Information technology — Database languages — SQL Technical Reports — Part 2: SQL Support for Time-Related Information.

Référence ISO/IEC TR 19075-2.

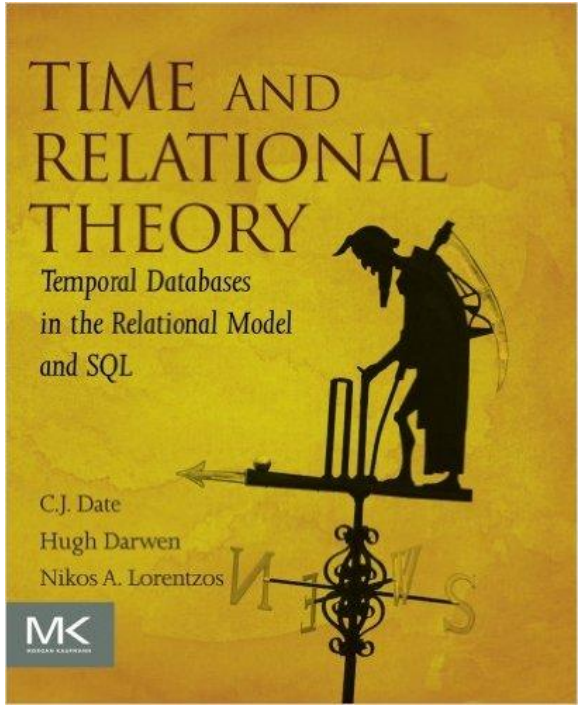
2e édition 2015-07-01.

Bibliographie

Time and Relational Theory

- Chris J. Date
- Hugh Darwen
- Nikos Lorentzos

Morgan Kaufman éditeur 2014

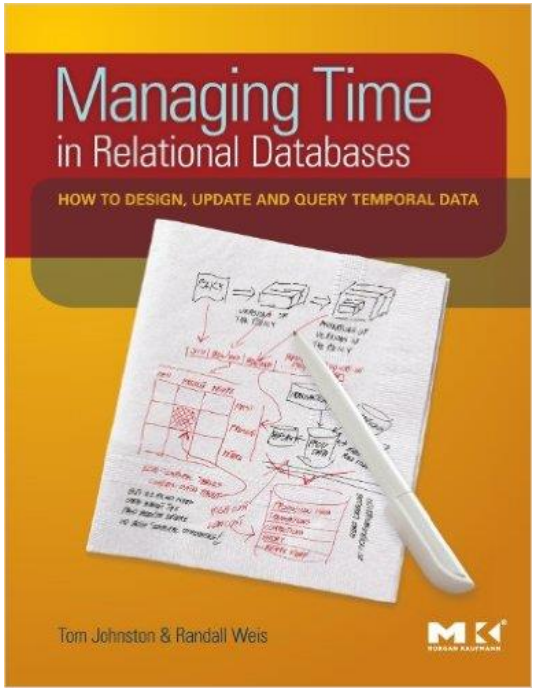


Bibliographie

Managing Time in Relationnal Databases

- Tom Johnston
- Randall Weis

Morgan Kaufman éditeur 2010

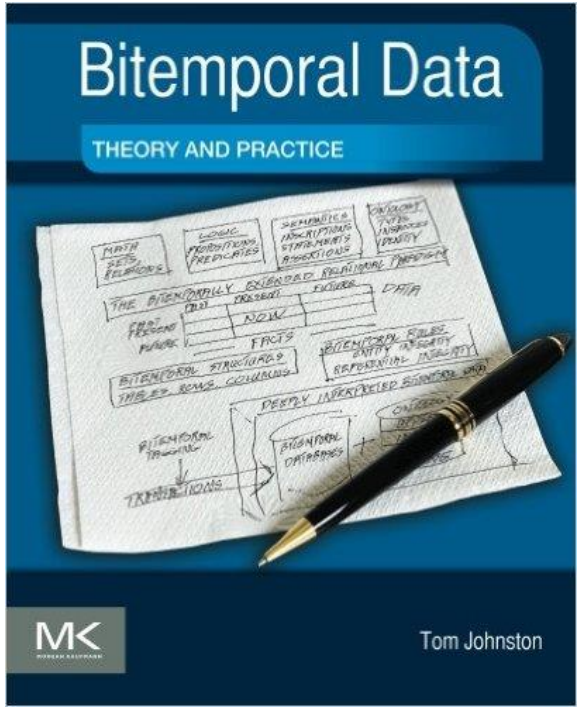


Bibliographie

Bitemporal Data

- Tom Johnston

Morgan Kaufman éditeur 2014

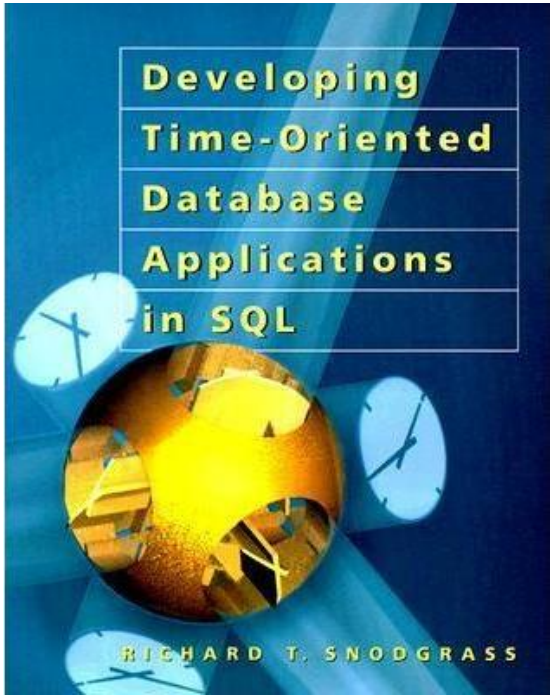


Bibliographie

Developping Time-Oriented Database Application in SQL

- Richard T. SNODGRASS

Morgan Kaufman éditeur 2001



Téléchargement libre à :

ftp://ftp.uwc.ac.za/users/DMS/Delete/M4ry-J4n3'S%20d0cS/HELP%20books/Developing%20Time%20-%20Oriented%20Database%20Applications%20in%20SQL.pdf

Webographie

Temporal Tables (MS) : https://msdn.microsoft.com/en-us/library/dn935015.aspx

Getting Started with System-Versioned Temporal Tables (MS) : https://msdn.microsoft.com/en-US/library/mt604462.aspx

Getting Started with Temporal Tables in Azure SQL Database (MS) : https://docs.microsoft.com/en-us/azure/sql-database/sql-database-temporal-tables

First Look at System-Versioned Temporal Tables : http://sqlmag.com/sql-server/first-look-system-versioned-temporal-tables-part-1-creating-tables-and-modifying-data

Time Traveling with Temporal Tables on SQL Server 2016 : http://www.sqlservercentral.com/articles/SQL+Server+2016/147087/

Webographie

SQL Server 2016 T-SQL Syntax to Query Temporal Tables :

https://www.mssqltips.com/sqlservertip/3682/sql-server-2016-tsql-syntax-to-query-temporal-tables/

SQL Server 2016 Temporal Table Query Plan Behaviour :

https://sqlperformance.com/2016/06/sql-server-2016/temporal-table-query-plan-behaviour

Stretch Databases and Temporal Tables in SQL Server 2016 :

https://www.vertabelo.com/blog/technical-articles/stretch-databases-and-temporal-tables-in-sql-server-2016

Improve query performance on memory optimized tables with Temporal using new index creation enhancement in SP1 :

https://blogs.msdn.microsoft.com/sqlcat/2016/12/08/improve-query-performance-on-memory-optimized-tables-with-temporal-using-new-index-creation-enhancement-in-sp1/

Merci beaucoup à nos sponsors!
Thank you to all our sponsors!

Join the conversation

#MSCloudSummit
@MSCloudSummit

Un événement proposé par Agile.Net, aOS, AZUG FR, CMD, GUSS

MS Cloud Summit Paris

Agile.Net – aOS – AZUG FR – CMD - GUSS

Merci Beaucoup! Thank you!

Join the conversation

#MSCloudSummit
@MSCloudSummit