

# Modélisation, conception et exploitation de données

## *Modélisation DDLM*

TEM\_03b  
112a

2025-09-25

Christina.Khnaisser@USherbrooke.ca  
Luc.Lavoie@USherbrooke.ca

© 2018-2021, Μητρώ (http://info.usherbrooke.ca/lavoie)  
CC BY-NC-SA 4.0 (https://creativecommons.org/licenses/by-nc-sa/4.0/)

1

## Plan

- Temporalisation TRM-TT
- Temporalisation TRM-VT
- Temporalisation TRM-BT
- Construction
- Invariants
  - définition
  - programmation
- Exemple
- Synthèse

2025-09-25

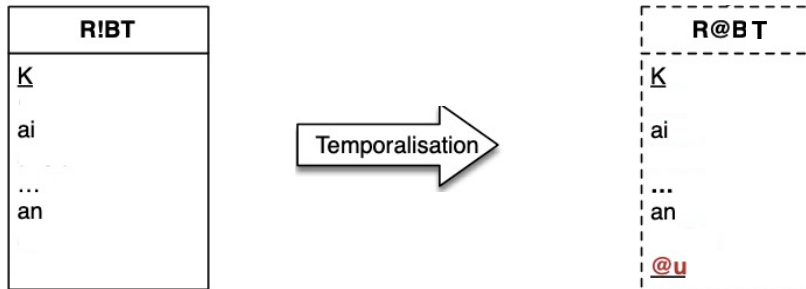
MCOE\_TEM\_03b - TRM - Modélisation DDLM (112a) — © 2018-2025 Μητρώ — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

2

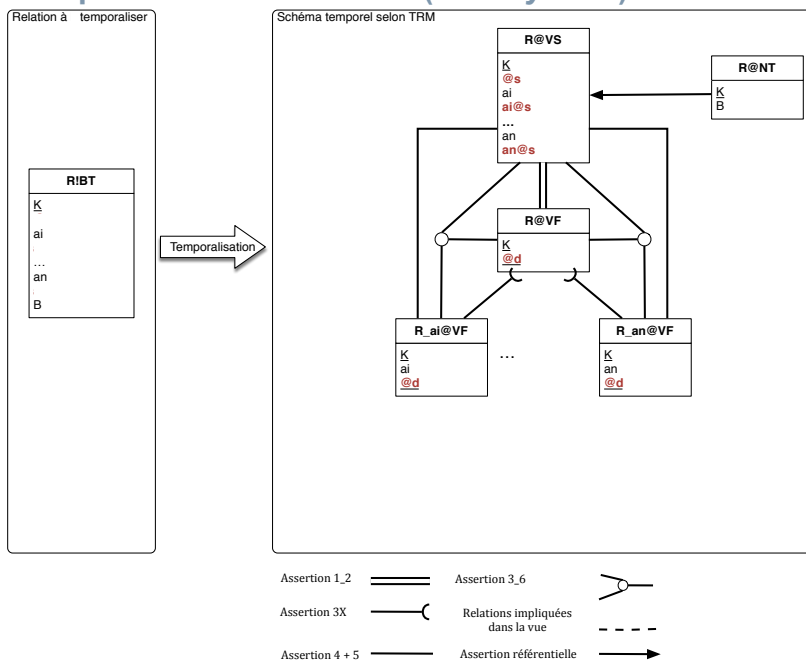
2

### Temporalisation TRM – TT (*transaction time*)

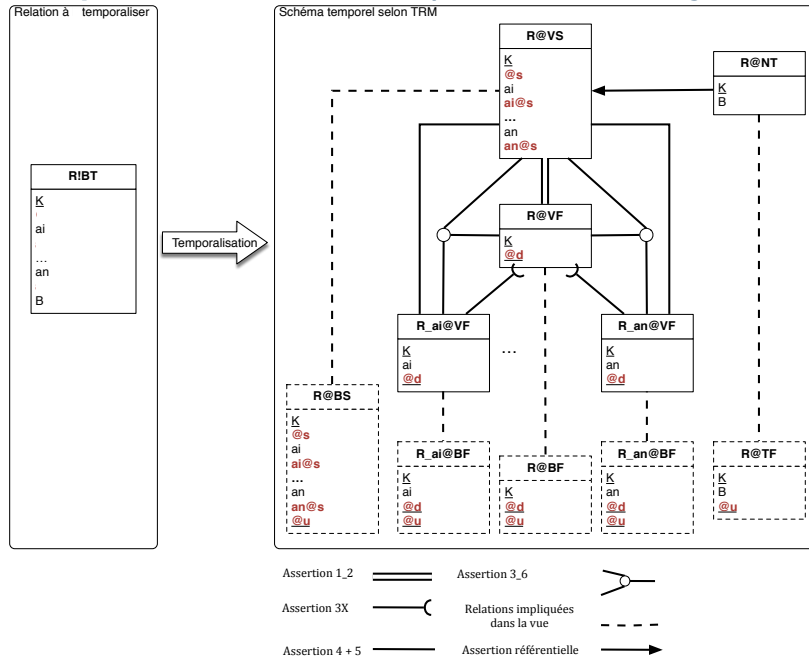
Il suffit de définir une vue sur le journal (déjà existant) de la table !



### Temporalisation TRM – VT (*validity time*)



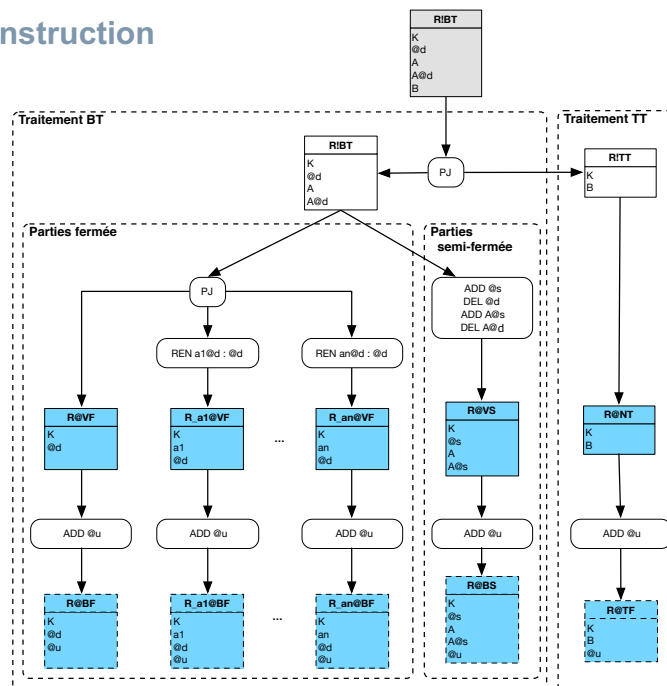
## Temporalisation TRM – BT (both time, validity and transaction)



5

5

## Temporalisation TRM – Construction



6

6

## Temporalisation TRM

### Exigences sur les attributs clés

#### EX.1 Non-redondance et non-contradiction

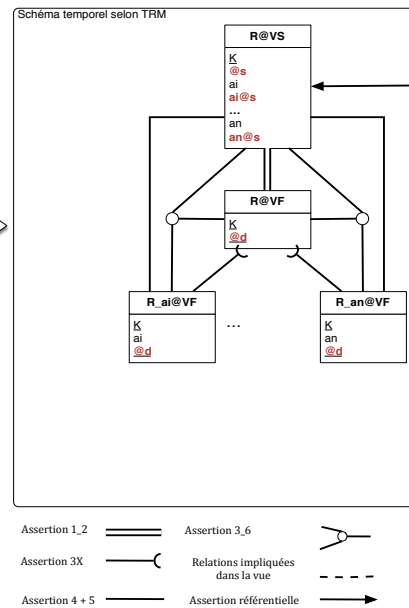
Une seule proposition (tuple) doit représenter le fait que la valeur d'un attribut clé est valide à un moment « t ».

#### EX.2 Non-circonlocution

Une même proposition (tuple) doit représenter le fait que la valeur d'un attribut clé est valide au moment « t » et au moment « t+1 ».

#### EX.3 Compacité (des parties d'une partition)

Si la valeur d'un attribut clé est valide à un moment « t », alors chacun des attributs non clés associés à cette clé doit avoir une (et une seule) valeur valide au même moment.



7

## Temporalisation TRM

### Exigences sur les attributs non clés

#### EX.4 Non-redondance et non-contradiction

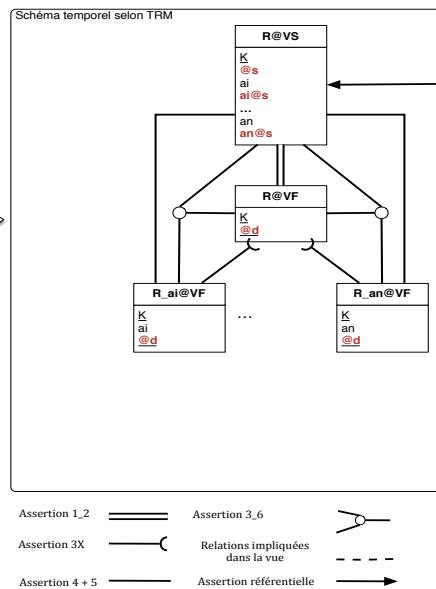
Une seule proposition (tuple) doit représenter le fait que la valeur d'un attribut non clé est valide à un moment « t ».

#### EX.5 Non-circonlocution

Une seule proposition (tuple) doit représenter le fait que la valeur d'un attribut non clé est valide au moment « t » et au « t+1 ».

#### EX.6 Compacité (des parties d'une partition)

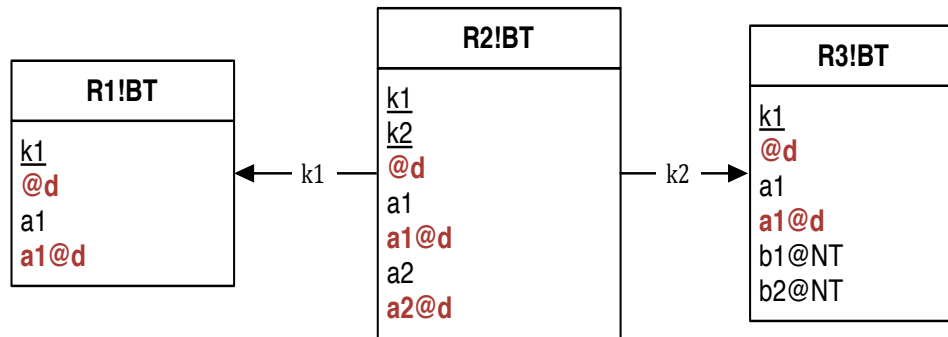
Si la valeur d'un attribut non clé est valide à un moment « t », alors chacun des attributs clés associés à cet attribut non clé doit avoir une (et une seule) valeur au même moment (l'inverse de EX.3).



8

## Temporalisation TRM

### Exigences d'une assertion référentielle



## Temporalisation TRM

### Exigences d'une assertion référentielle

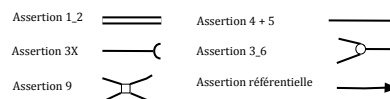
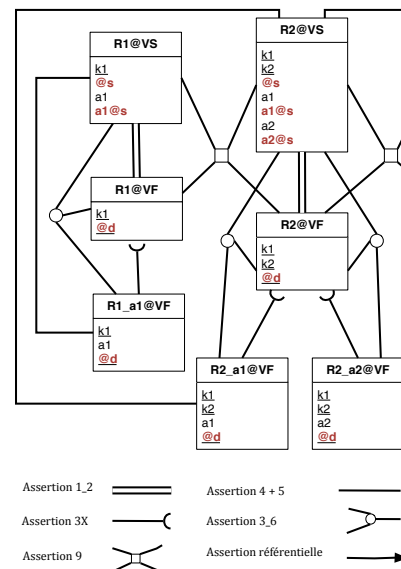
- EX.7 Non-redondance et non-contradiction

Une seule proposition (tuple) doit représenter le fait que la valeur d'un attribut d'origine est valide à un moment « t ».

- EX.8 Non-circonlocution
- Une même proposition (tuple) doit représenter le fait que la valeur d'un attribut d'origine est valide au moment « t » et au « t+1 ».

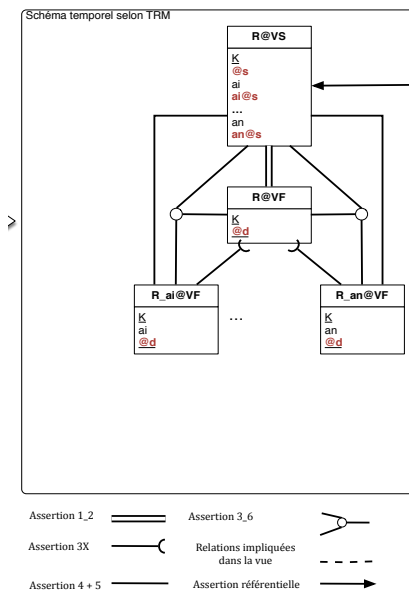
- EX.9 Compacité (entre partitions)

Si la valeur d'un attribut origine est valide à un moment « t », la valeur de l'attribut destination associée doit être valide au même moment.



## Temporalisation TRM Exigence 1\_2

- La période de validité de la partie R@VS ne doit pas inférieure ou égale au successeur de la période de validité de la partie R@VF.
- [EX1\_2]  
CONSTRAINT R\_ex1\_2  
IS\_EMPTY  
((R@VS JOIN R@VF)  
WHERE @s ≤ POST(@d));



2025-09-25

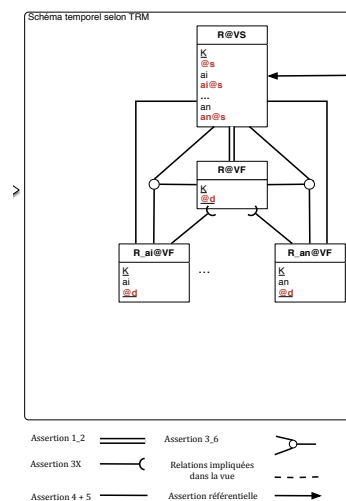
MCOE\_TEM\_03b - TRM - Modélisation DOLM (v111a) — © 2018-2025 MFTS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

11

11

## Temporalisation TRM Exigence 3x

- Toute clé non temporalisée de la partie R@VF doit apparaître dans chacune des partie R\_ai@VF.
- [EX3x]  
CONSTRAINT R\_ai\_ex3x  
( R@VF {K} ⊆  
R\_ai@VF {K} );



2025-09-25

MCOE\_TEM\_03b - TRM - Modélisation DOLM (v111a) — © 2018-2025 MFTS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

12

12

## Temporalisation TRM

### Exigence 3\_6

- La contrainte 3\_6 garantit la compacité des tuples entre les parties  $R@VS$ ,  $R@VF$  et une  $R_{a_i}@VF$ .

#### [EX3\_6]

CONSTRAINT  $R_{a_i\_ex3\_6}$

WITH (

$t1 := \text{EXTEND } R@VS :$   
 $\{ @d := \text{INTERVALLE}[@s:UFN] \},$

$t2 := t1 \{K, @d\},$

$t3 := t2 \text{ UNION } R@VF,$

$t4 := \text{EXTEND } R@VS :$

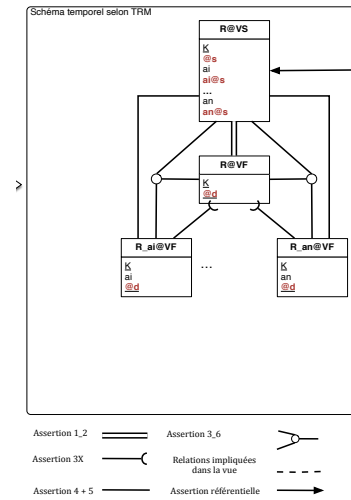
$\{ @d := \text{INTERVALLE}[ai@s:UFN] \},$

$t5 := t4 \{K, ai, @d\},$

$t6 := t5 \text{ UNION } R_{a_i}@VF,$

$t7 := t6 \{K, @d\} ) :$

USING( $@d$ ) :  $t3 = t7$ ;



2025-09-25

MGED\_TEM\_03b - TRM - Modélisation DOLM (v111a) — © 2018-2025 M.F.T. — CC BY-NC-SA 4.0  
 Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

13

13

## Temporalisation TRM

### Exigences 4 et 5

- La période de validité de l'attribut  $a_i$  de la partie  $R@VS$  ne doit pas être inférieure au successeur de la période de validité de la partie  $R_{a_i}@VF$ .

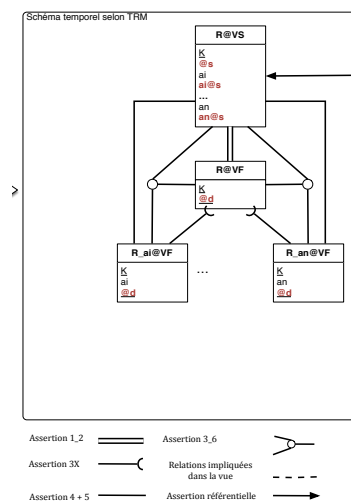
#### [EX4]

CONSTRAINT  $R_{a_i\_ex4}$  IS EMPTY  
 $(( R@VS \text{ JOIN } R_{a_i}@VF \{K, @d\} )$   
 $\text{WHERE } ai@s < \text{POST}(@d) );$

- La période de validité de l'attribut  $a_i$  de la partie  $R@VS$  ne doit pas être égale au successeur de la période de validité de la partie  $R_{a_i}@VF$  (vérification des périodes consécutives).

#### [EX5]

CONSTRAINT  $R_{a_i\_ex5}$  IS EMPTY  
 $(( R@VS \text{ JOIN } R_{a_i}@VF )$   
 $\text{WHERE } ai@s = \text{POST}(@d) );$



2025-09-25

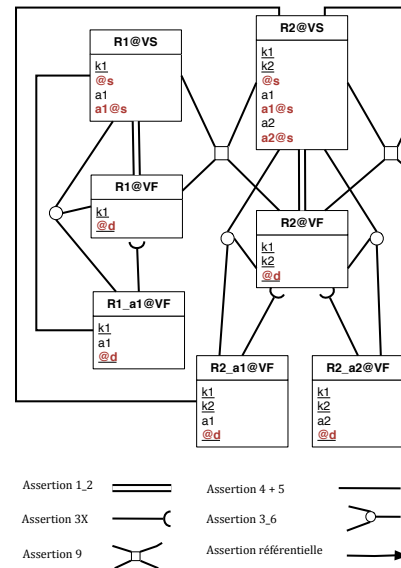
MGED\_TEM\_03b - TRM - Modélisation DOLM (v111a) — © 2018-2025 M.F.T. — CC BY-NC-SA 4.0  
 Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

14

14

### Temporalisation TRM Exigences 7 et 8

- Si l'attribut d'origine fait partie d'une relation-clé alors 1 et 2 garantissent déjà 7 et 8.
- Si l'attribut d'origine fait partie d'une relation non-clé alors 4 et 5 garantissent déjà 7 et 8.



### Temporalisation TRM Exigence 9 – *analogue à 3\_6*

- L'exigence 9 traite les assertions référentielles. Elle garantit que la période de validité de chaque tuple de la relation d'origine est incluse dans la période de validité du tuple correspondant dans la relation de destination.
- Remplacer chaque assertion référentielle  $Ro_{\{Xo\}} \rightarrow Rd$  par l'assertion  $Ro\_Rd\_ex9$ .
- [EX9]  

$$\text{CONSTRAINT } Ro\_Rd\_ex9$$

$$\text{WITH (}$$

$$t1 := \text{EXTEND } Rd@VS:$$

$$\quad \{ @d := \text{INTERVAL}[@s:UFN] \},$$

$$t2 := t1 \{ K, @d \},$$

$$t3 := t2 \text{ UNION } Rd@VF,$$

$$t4 := \text{EXTEND } Ro@VS:$$

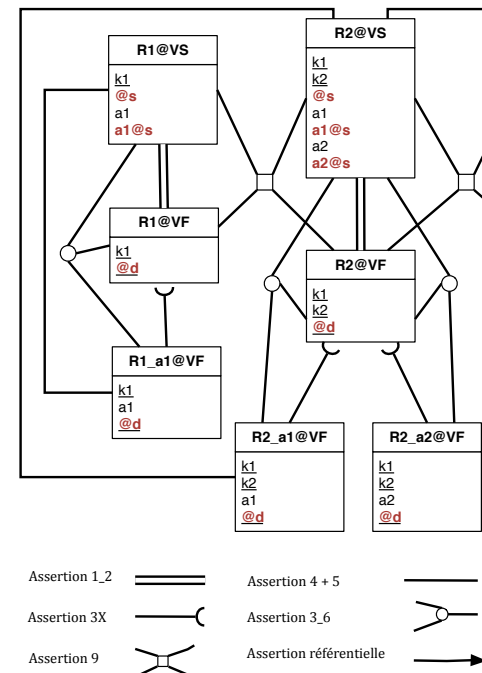
$$\quad \{ @d := \text{INTERVAL}[@s:UFN] \},$$

$$t5 := t4 \{ K, @d \},$$

$$t6 := Ro@VF \{ K, @d \},$$

$$t7 := t5 \text{ UNION } t6 \text{ )};$$

$$\text{USING}(@d) : t7 \subseteq t3;$$





## Temporalisation TRM

### Exigence 9 – avec explicitations

- L'exigence 9 traite les assertions référentielles. Elle garantit que la période de validité de chaque tuple de la relation d'origine est incluse dans la période de validité du tuple correspondant dans la relation de destination.
- Remplacer l'assertion  $REF R_o \{X_o\} \rightarrow R_d$  par l'assertion  $R_o R_d_{ex9}$ .
- Note 1 :  $X_o \equiv K_d$ .
- Note 2 : la période de validité de  $X_o$  dans  $R_o$  est la même que celle de la clé  $K_o$  dans  $R_o$  en raison de l'assertion  $R_o X_o_{3\_6}$ .

○ [EX9]

CONSTRAINT  $R_o R_d_{ex9}$

WITH (

vo1 := EXTEND  $R_o @VS$  : { $@d := INTERVAL[@s:UFN]$ },

vo2 := vo1 { $K, @d$ },

vo3 :=  $R_o @VF$  { $K, @d$ },

vo := vo2 UNION vo3,

vd1 := EXTEND  $R_d @VS$  : { $@d := INTERVAL[@s:UFN]$ },

vd2 := vd1 { $K, @d$ },

vd := vd2 UNION  $R_d @VF$  ) :

USING( $@d$ ) : vo  $\subseteq$  vd;

2025-09-25

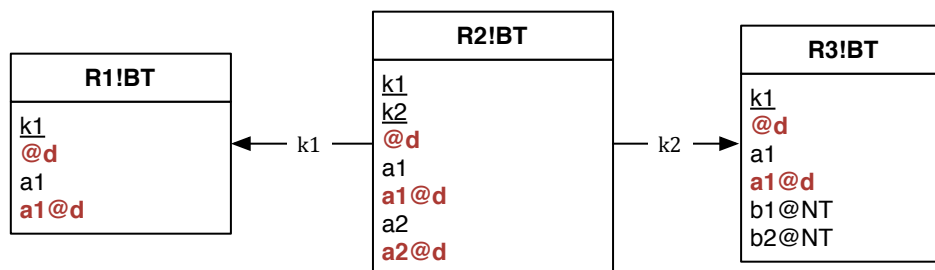
MCED\_TEM\_03b - TRM - Modélisation DDM (v111a) — © 2018-2025 MFTS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

17

17

## Temporalisation TRM

### Exemple – schéma source



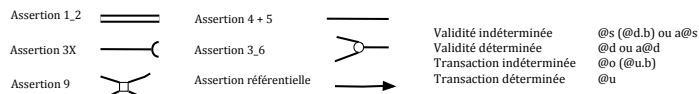
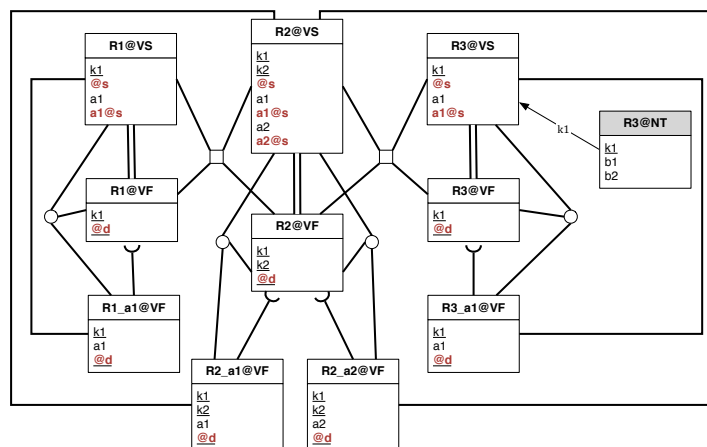
2025-09-25

MCED\_TEM\_03b - TRM - Modélisation DDM (v111a) — © 2018-2025 MFTS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

18

18

## TRM



2025-09-25

MCOE\_TEM\_03b - TRM - Modélisation DOLM (v111a) — © 2018-2025 MFC — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

19

19

## Exemple Fate, Darwen, Lorentzos p. 399

S\_SINCE

SNO	SNO_SINCE	STATUS	STATUS_SINCE
S1	d04	20	d06
S2	d07	10	d07
S3	d03	30	d03
S4	d04	20	d08
S5	d02	30	d02

SP\_SINCE

SNO	PNO	SINCE
S1	P1	d04
S1	P2	d05
S1	P3	d09
S1	P4	d05
S1	P5	d04
S1	P6	d06
S2	P1	d08
S2	P2	d09
S3	P2	d08
S4	P5	d05

S\_DURING

SNO	DURING
S2	[d02:d04]
S6	[d03:d05]

SP\_DURING

SNO	PNO	DURING
S2	P1	[d02:d04]
S2	P2	[d03:d03]
S3	P5	[d05:d07]
S4	P2	[d06:d09]
S4	P4	[d04:d08]
S6	P3	[d03:d03]
S6	P3	[d05:d05]

S\_STATUS\_DURING

SNO	STATUS	DURING
S1	15	[d04:d05]
S2	5	[d02:d02]
S2	10	[d03:d04]
S4	10	[d04:d04]
S4	25	[d05:d07]
S6	5	[d03:d04]
S6	7	[d05:d05]

2025-09-25

MCOE\_TEM\_03b - TRM - Modélisation DOLM (v111a) — © 2018-2025 MFC — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

20

20

- Supplier Sx has been able to supply part Py since day d'

```

WITH (
  temp := SP_DURING WHERE SNO = Sx AND PNO = Py AND d ≤ POST (DURING)
):
CASE
  WHEN IS_EMPTY (temp) THEN
    UPDATE SP_SINCE WHERE SNO = Sx AND PNO = Py : { SINCE := d' }
  ELSE
    DELETE SP_DURING temp ,
    UPDATE SP_SINCE WHERE SNO = Sx AND PNO = Py :
      { SINCE := MIN (temp , BEGIN (DURING)) }
END CASE ;

```

## Synthèse

### Relation courante

- Une relation courante nommée SINCE (@VS), contient les données qui reflètent l'état courant. Autrement dit, les données couramment valides dans le monde réel ou celles qui seront valides dans le futur. La relation représente les données dont la date de début est connue, mais pas la date de fin. Elle est composée de l'ensemble d'attributs à temporalisé et de l'ensemble d'attributs épisodique. La relation est normalisée en 5FN.

### Relation historique

- Une relation historique nommée DURING (@VF), contient les données du présent, du passé et du futur. La relation représente les données dont la date de début et la date de fin sont connues. Elle est composée de l'ensemble d'attributs à temporalisé et de l'ensemble d'attributs de période. La relation est normalisée en 6FN.

### Relation de trace

- La relation de trace nommée LOG (@BS, @BF), contient les traces de mise à jour des données par l'intermédiaire d'un attribut de trace. Elle est associée à chaque relation du schéma.

## Synthèse

- À partir d'un schéma non temporel initial  
 $S \{ \langle v_1, \dots, v_n \rangle, \langle cr, \dots, cr_m \rangle \}$ , normalisé en 5FN,  
 pour chaque variable de relation  $v_j$ 
  - **Construire les relations courantes**
    - Ajouter un attribut d'incidence  $t_0$  de type estampille associé à la clé  $K$ .
    - Pour chaque attribut  $a_i$ , ajouter un attribut d'incidence  $t_i$  de type estampille.
  - **Construire les relations historiques**
    - Ajouter un attribut de période  $t_0$  de type intervalle d'estampille associé à la clé  $K$ .
    - Pour chaque attribut  $a_i$ , ajouter un attribut de période  $t_i$  de type intervalle d'estampille [1].
    - Ajouter une dépendance de jointure DJ formé par une partition courante et historique clé  $\{K, t_0\}$  et  $n$  partition courante et historique  $\{K, a_i, t_i\}$  où  $n = |A|$  et une partition pour les attributs de B  $\{K, B\}$ .
    - Normaliser le schéma en 6FN.

[1] L'attribut  $t_i$  sera renommé  $t_0$  lors de la normalisation 6FN.

