BASE DE DONNÉES
TEMPORALITÉ

TRM
Modélisation

BD212
v110a
2016-12-02

Christina KHNAISSER et Luc LAVOIE
Département d'informatique
Faculté des sciences

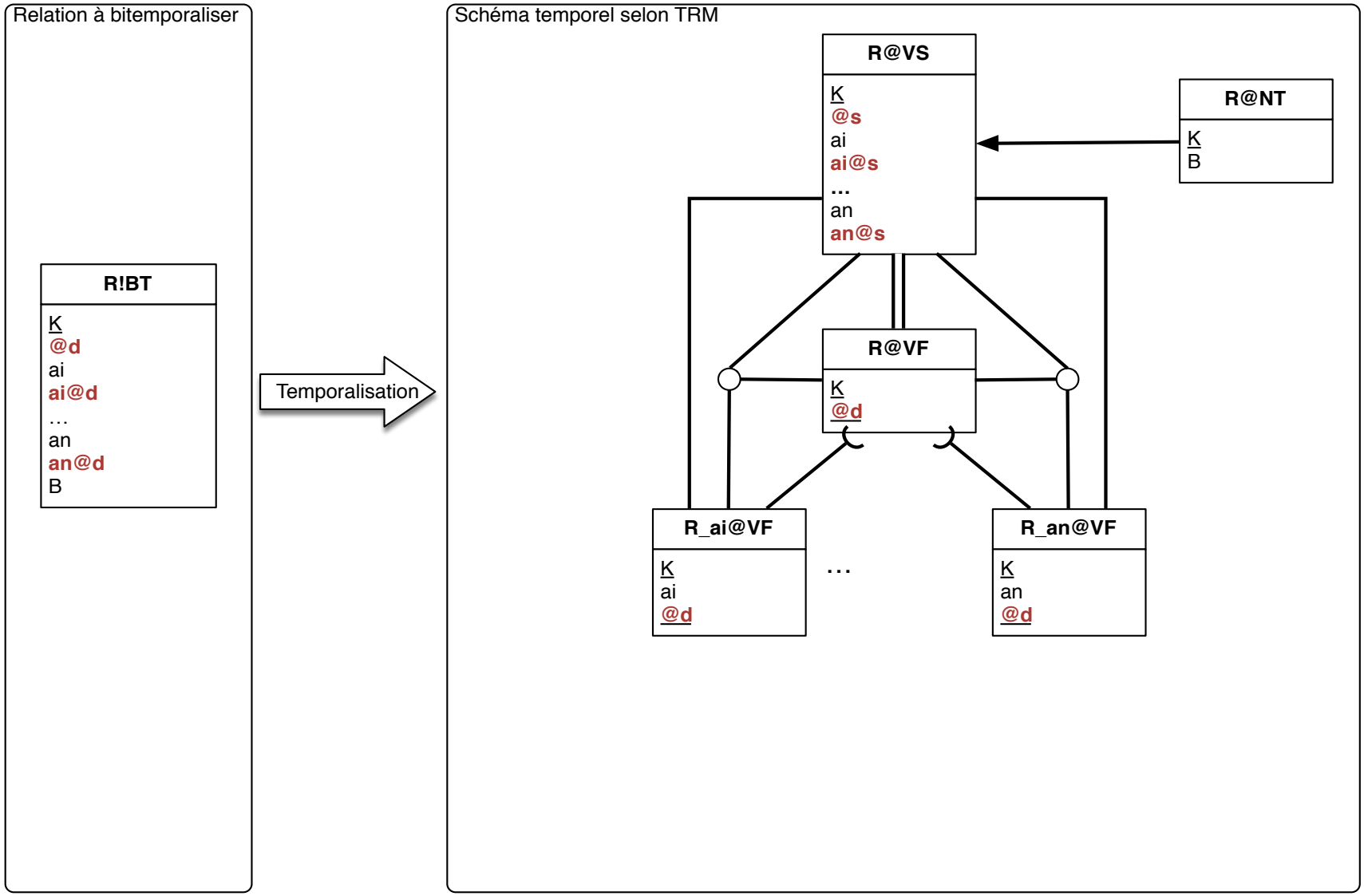
UNIVERSITÉ DE SHERBROOKE

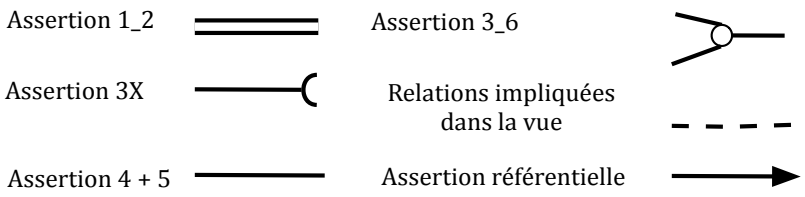
Christina.Khnaisser@usherbrooke.ca
Luc.Lavoie@usherbrooke.ca
http://info.usherbrooke.ca/llavoie

Plan

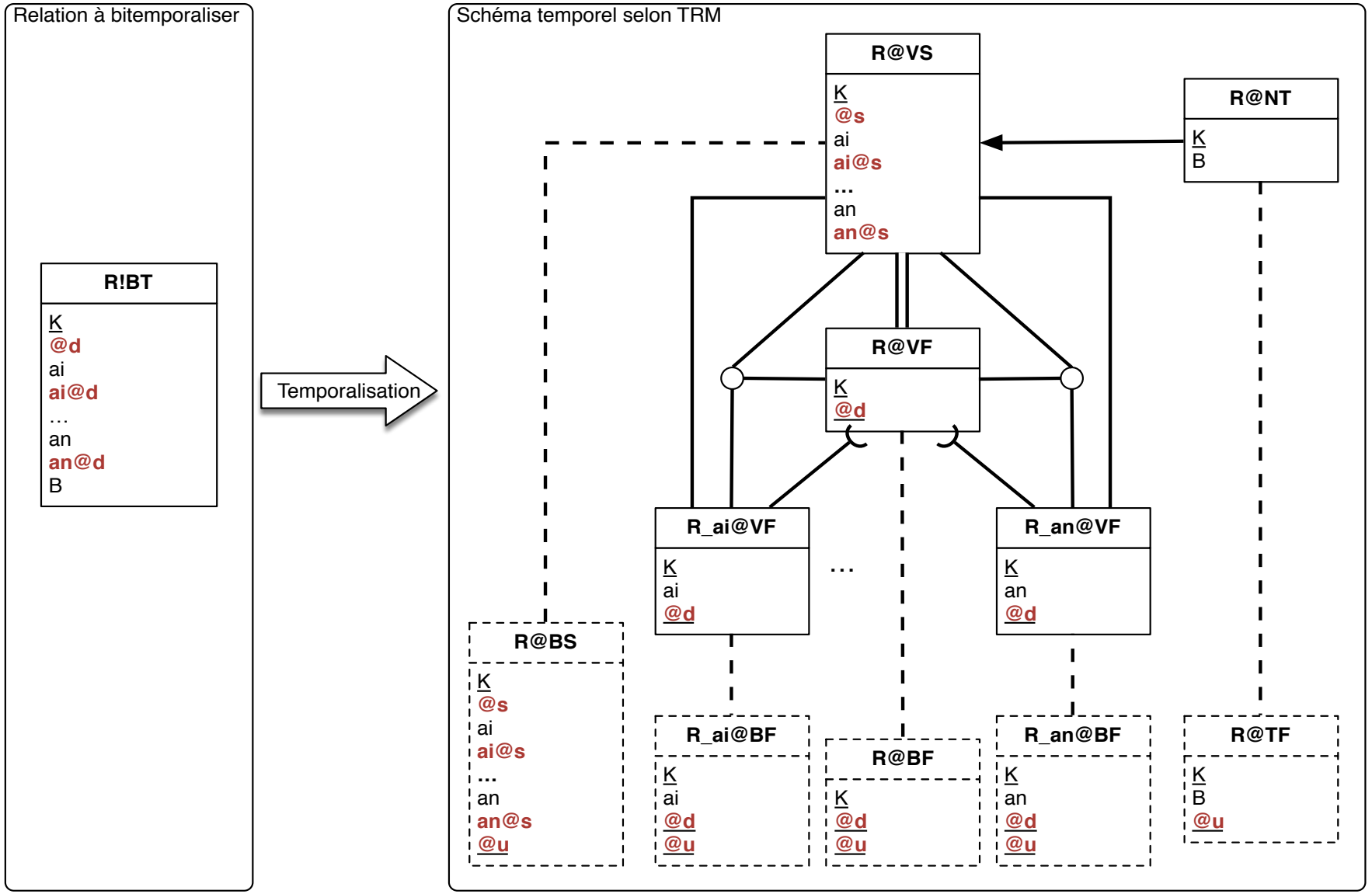
- Temporalisation TRM-VT
- Temporalisation TRM-BT
- Invariants
- Synthèse

TEMPORALISATION TRM – VT

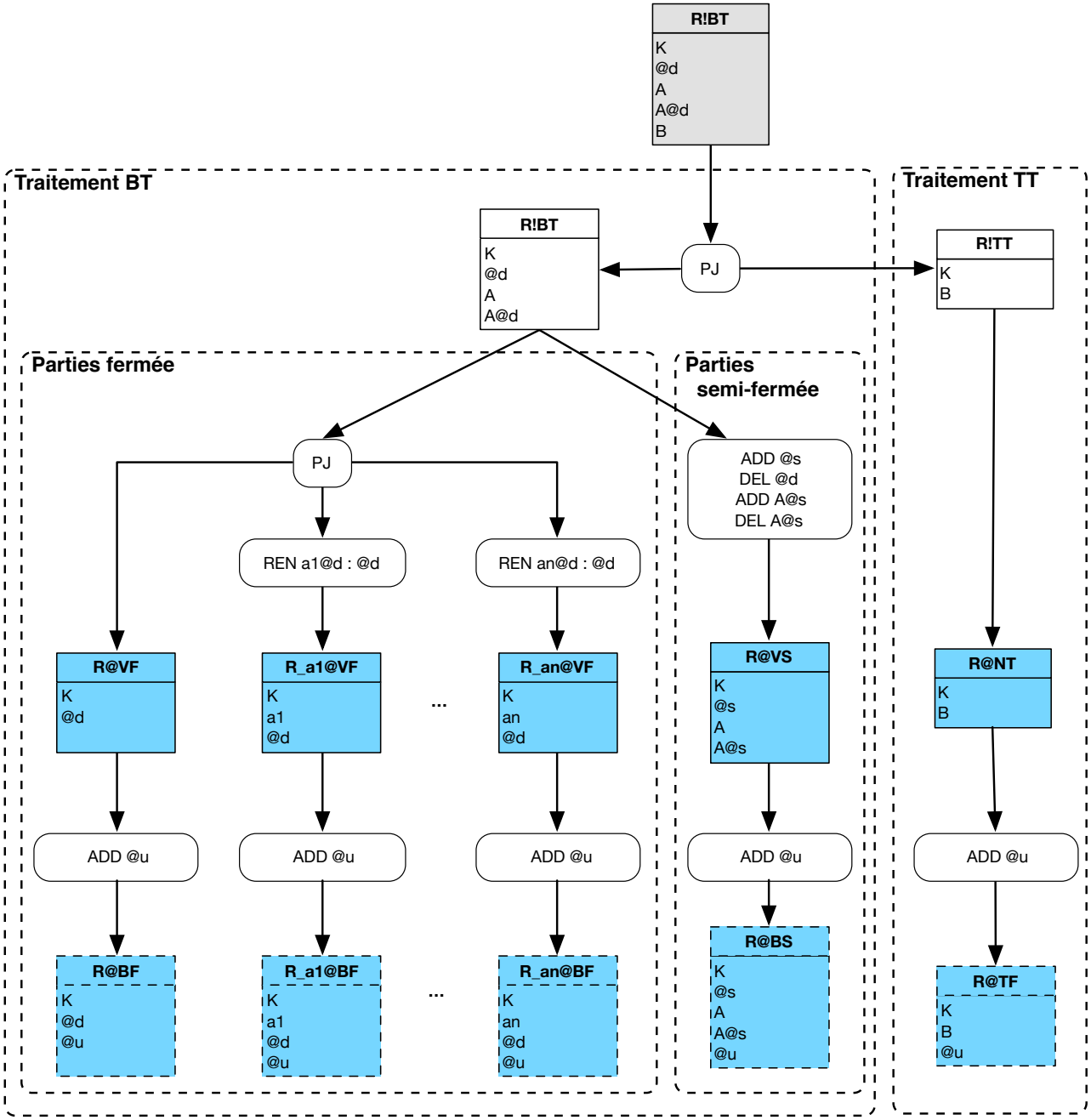




TEMPORALISATION TRM – BT



TEMPORALISATION TRM – CONSTRUCTION



TEMPORALISATION TRM

EXIGENCES SUR LES ATTRIBUTS CLÉS

- EX.1 Non-redondance
  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 associées à cette clé doit avoir une (et une seule) valeur valide au même moment.

TEMPORALISATION TRM

EXIGENCES SUR LES ATTRIBUTS NON CLÉ

- 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).

TEMPORALISATION TRM

EXIGENCES D'UNE ASSERTION RÉFÉRENTIELLE

- EX.7 Non-redondance
  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));

Temporalisation TRM
Exigence 3x

- Toute clé non temporalisée de la partie R@VF doit apparaître dans chacune des partie R_a_i@VF.
- [EX3x]
  CONSTRAINT R_a_i_ex3x
  (R@VF {K} ⊆ R_a_i@VF {K});

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 := EXTEND R@VS : {@d := INTERVALLE[@s:UFN]},
  t2 := t1 {K, @d},
  t3 := t2 UNION R@VF,
  t4 := EXTEND R@VS : {@d := INTERVALLE[ai@s:UFN]},
  t5 := t4 {K, a_i, @d},
  t6 := t5 UNION R_a_i@VF,
  t7 := t6 {K, @d}) :
USING(@d) : t3 = t7;

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 JOIN R_a_i@VF {K, @d})
  WHERE a_i@s < 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 JOIN R_a_i@VF)
  WHERE a_i@s = POST(@d));

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.

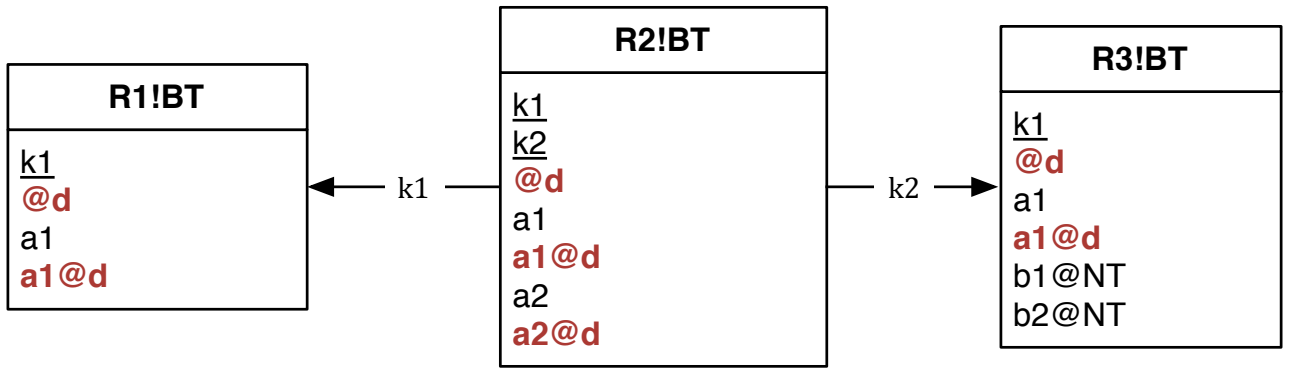
TEMPORALISATION TRM
EXIGENCE 9 – *ANALOGUE À 3_6*

- 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} → Rd*** par l'assertion ***Ro_Rd_ex9.***
- [EX9]
  CONSTRAINT Ro_Rd_ex9
  WITH (
  t1 := EXTEND Rd@VS: {@d := INTERVAL[@s:UFN]},
  t2 := t1 {K, @d},
  t3 := t2 UNION Rd@VF,
  t4 := EXTEND Ro@VS : {@d:= INTERVAL[@s:UFN]},
  t5 := t4 {K, @d},
  t6 := Ro@VF {K, @d},
  t7 := t5 UNION t6) :
  USING(@d) : t7 ⊆ t3;

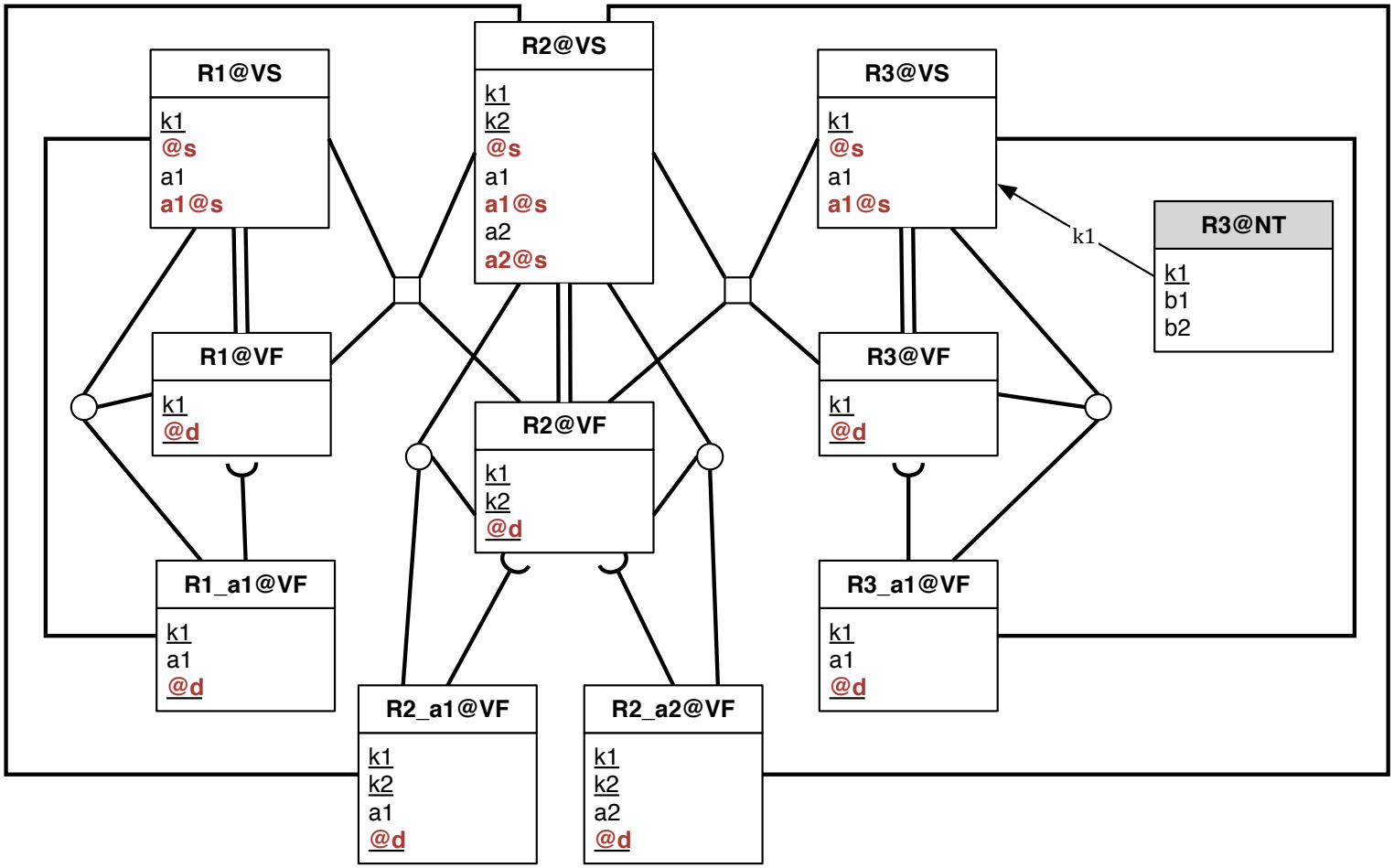
TEMPORALISATION TRM
EXIGENCE 9 – AVEC EXPLICITATIONS

- 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 ⊆ vd;

TEMPORALISATION TRM
EXEMPLE – SCHÉMA SOURCE



TRM



Assertion 1_2

Assertion 3X

Assertion 9

Assertion 4 + 5

Assertion 3_6

Assertion référentielle

Validité indéterminée	@s (@d.b) ou a@s
Validité déterminée	@d ou a@d
Transaction indéterminée	@o (@u.b)
Transaction déterminée	@u

QUESTIONS ?

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 valident dans le futur. La relation représente les données dont la date de début est connue, mais pas la date de fin. Elle composé de l'ensemble d'attributs a 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 composé de l'ensemble d'attributs a 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 {<v_1, …, v_n>, <cr , …, cr_m>}, normalisé en 5FN,
  pour chaque variable de relation $\boldsymbol{v_j}$

  - **Construire les relations courantes**
    - Ajouter un attribut d'épisode $\boldsymbol{t_0}$ de type estampille associé à la clé $\boldsymbol{K}$.
    - Pour chaque attribut $\boldsymbol{a_i}$, ajouter un attribut d'épisode $\boldsymbol{t_i}$ de type estampille.

  - **Construire les relations courantes et historiques**
    - Ajouter un attribut de période $\boldsymbol{t_0}$ de type intervalle d'estampille associé à la clé $\boldsymbol{K}$.
    - Pour chaque attribut $\boldsymbol{a_i}$, ajouter un attribut de période $\boldsymbol{t_i}$ 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.