L'ALGEBRE RELATIONNELLE

L'expression algébrique inclut, en plus des tables relationnelles, d'autres paramètres constitués de formules logiques construites à partir de :

constantes

attributs des tables

opérateur de comparaison ($=, \neq, <, \leq, >, \geq$)

connecteurs logiques ($\neg, \vee, \wedge$)

$$\Pi_{\text{numAvion, localisation}} (\sigma_{\text{localisation} \neq \text{'Nice'} \wedge \text{capacite} > 200} (\text{Avion}))$$

L'algèbre relationnel

L'algèbre relationnelle (AR) est un système formel constitué d'opérations sur les relations.

Les opérations de l'AR satisfont la propriété de fermeture car, appliquées à une ou plusieurs relations, elles produisent une nouvelle relation.

Utilisation : démontrer l'équivalence de requêtes + écrire des optimiseurs de requêtes

Classification des opérateurs

Les opérations de base qui constituent un ensemble minimal d'opérations dans le sens où aucune d'entre elles ne peut s'écrire par combinaison des autres : union, différence, produit cartésien, projection, sélection, renommage.

Les opérations dérivées qui peuvent être exprimées à partir des opérations de base : intersection, théta-jointure, jointure naturelle et division.

Les opérations de groupage et agrégation qui constituent une extension des opérations de base sans pouvoir être exprimées à l'aide de celles-ci : compte, somme, min, max, moyenne...

L'AR comme langage de requêtes

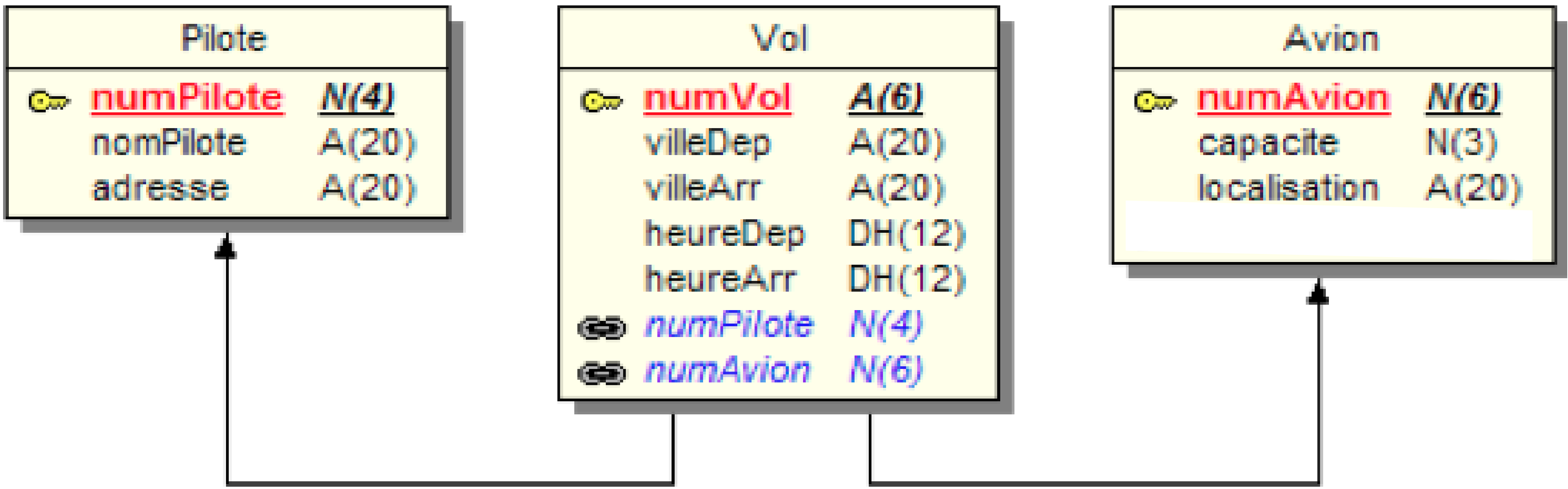
L'apprentissage de l'AR sert d'algorithmique préalable au codage des requêtes en SQL et permet d'améliorer l'écriture des requêtes.

$$\Pi_{\textbf{numAvion, localisation}} (\sigma_{\textbf{localisation} \neq \textbf{'Nice'} \wedge \textbf{capacite} > \textbf{200}} (\text{Avion}))$$

Select numAvion,localisation from Avion where localisation<>'nice' and capacite>200

Formuler en algèbre relationnelle une requête relative à une base de données relationnelle consiste à écrire une expression algébrique en composant des opérations de l'algèbre relationnelle ; chaque opération étant appliquée à des tables de la base de données.

Fil rouge du cours



Les opérateurs de base : la projection $\Pi_{a1,...,an}(R)$

$$\left\{u \,\middle|\, \exists v \in \mathcal{R},\ \forall i \in \{1,\ldots,n\},\ u_{A_i} = v_{A_i}\right\}$$

Relation obtenue à partir de R en ne gardant des éléments de R que les colonnes a1,...,an.

Attention : notion ensembliste donc pas de doublons ce qui explique la présence que d'une seule occurrence de (c,d) dans la projection.

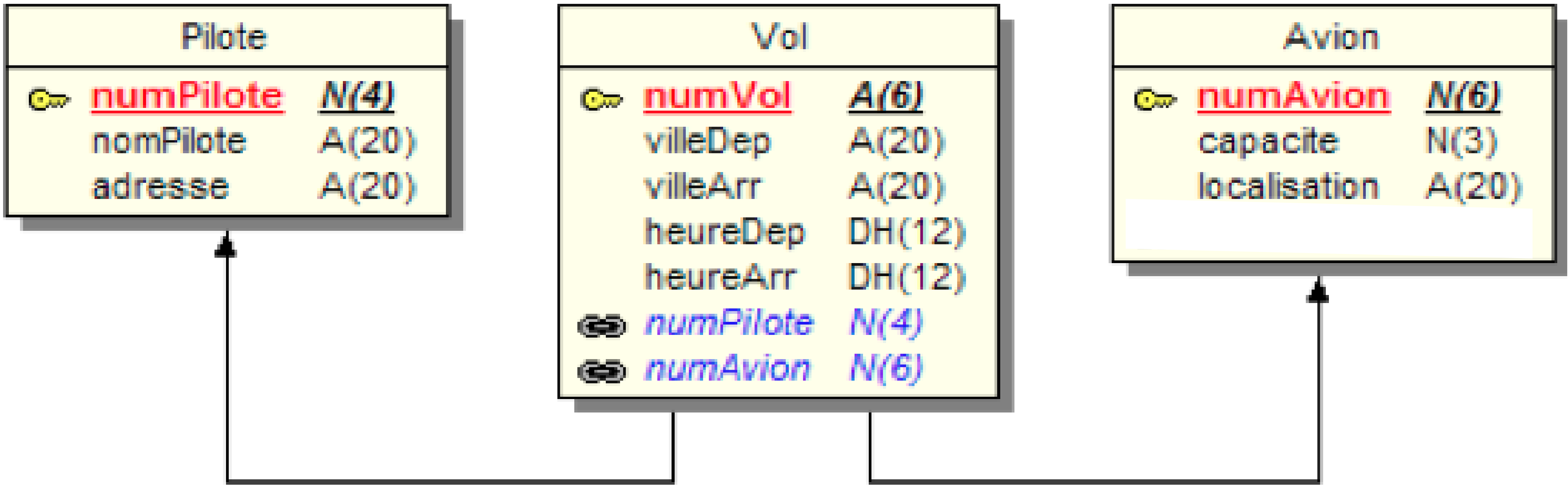
r

A	B	C
a	b	c
d	a	f
c	b	d
c	a	d

$\pi_{A,C}(r)$

A	C
a	c
d	f
c	d

Exemple de projection

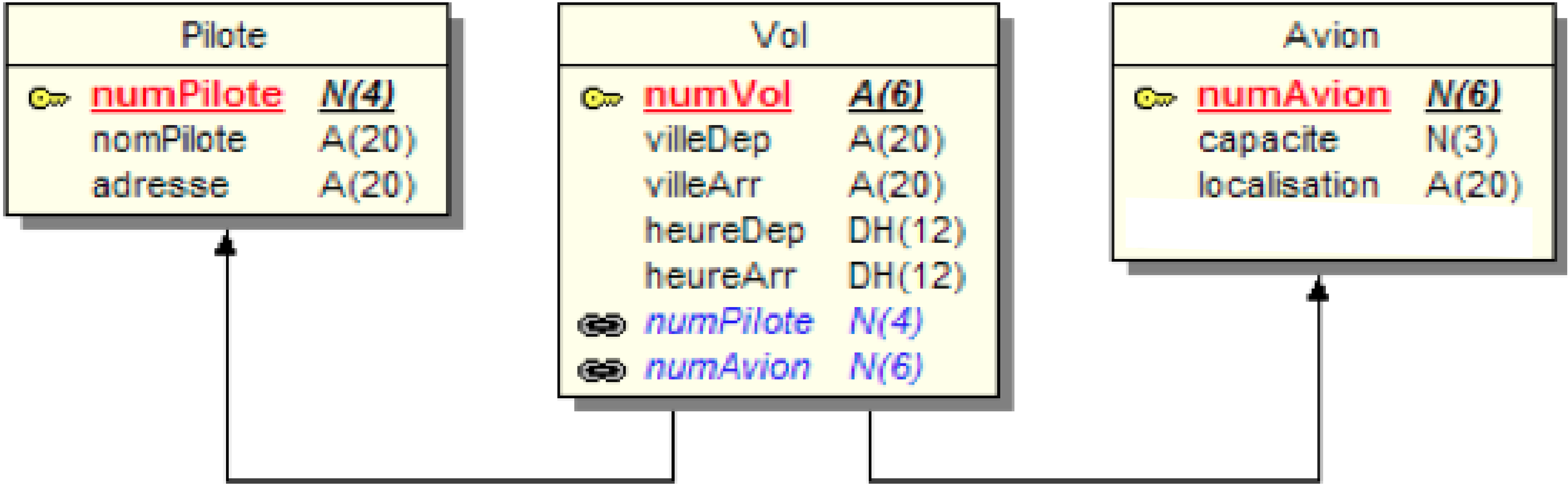


Quelles sont les différentes villes de départ de la table vol ?

$$\Pi_{\textbf{villeDep}} (\text{Vol})$$

Quels sont les noms et adresses des pilotes ?

Exemple de projection



Quelles sont les différentes villes de départ de la table vol ?

$$\Pi_{\textbf{villeDep}} (\text{Vol})$$

Quels sont les noms et adresses des pilotes ?

$$\Pi_{\textbf{nomPilote,adresse}} (\text{Pilote})$$

Les opérations de base : la sélection σ_C (R)

Ensemble des éléments de R qui satisfont la condition de sélection (dite qualification) C.

$$\{u \mid u \in \mathcal{R} \wedge C(u)\}$$

Une qualification est une formule logique construite à partir des règles suivantes

- (Ai Op Aj) ou (Ai Op constante) avec Op $\in \{=, \neq, <, \leq, >, \geq\}$ (Ai,Aj des attributs de R) est une qualification,

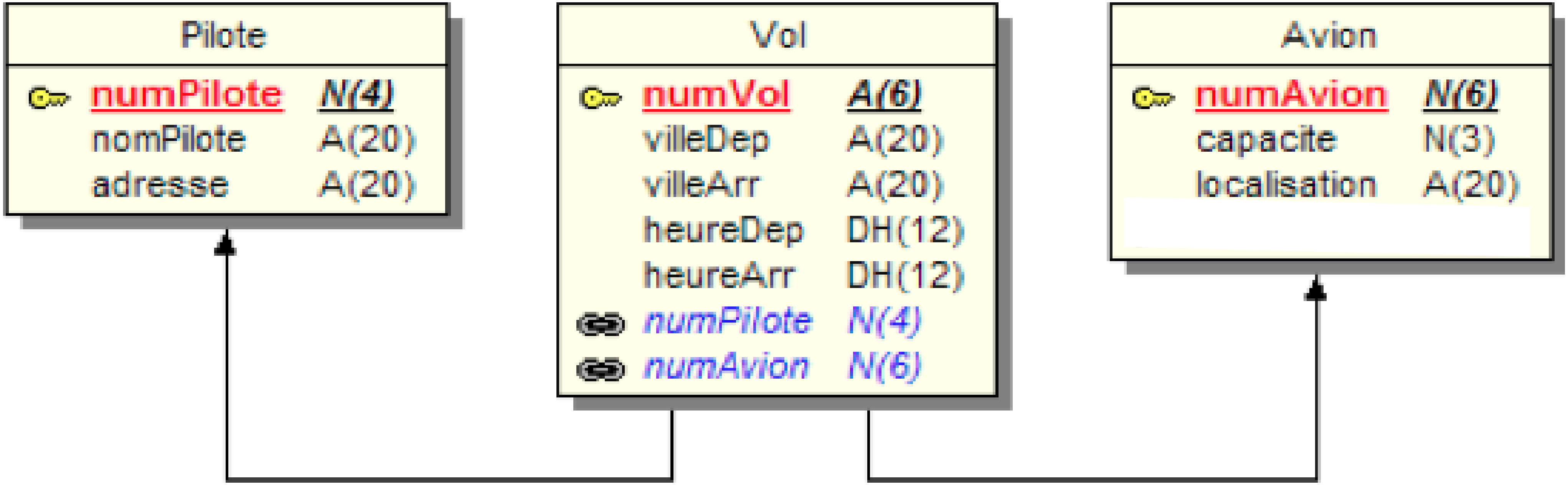
- si Q1 et Q2 sont des qualifications, alors (Q1), $\neg$ Q1, Q1 $\wedge$ Q2, Q1 $\vee$ Q2 sont des qualifications.

r		
A	**B**	**C**
a	b	c
d	a	f
c	b	d

$\sigma_{B = 'b'}$(**r**)		
A	**B**	**C**
a	b	c
c	b	d

Les opérations de base : la sélection σ_C (R)

Exemple : quels sont les avions (numéro et localisation) non localisés à Nice ayant une capacité supérieure à 200 places.



Les opérations de base : la sélection σ_C (R)

Exemple : quels sont les avions (numéro et localisation) non localisés à Nice ayant une capacité supérieure à 200 places.

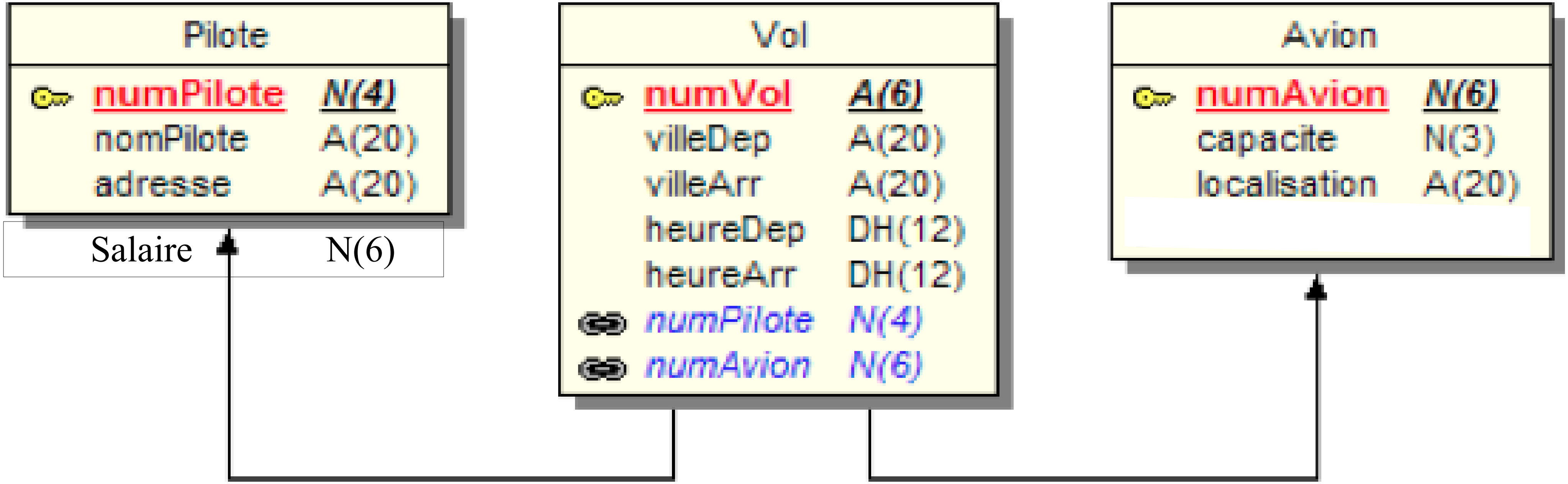
$$\Pi_{\textbf{numAvion, localisation}} (\sigma_{\textbf{localisation} \neq \textbf{'Nice'} \wedge \textbf{capacite} > \textbf{200}} (\text{Avion}))$$

Remarque : en cas d'ambiguïté sur le nom d'un attribut un renommage et/ou préfixage par le nom de la relation peut s'avérer indispensable.

Les opérations de base : la sélection σ_C (R)

Exemple : Rajoutons une colonne Salaire dans la table Pilote.

Quels sont les noms des pilotes domiciliés à Paris dont le salaire est supérieur à 3000



Les opérations de base : la sélection σ_C (R)

Exemple : Quels sont les noms des pilotes domiciliés à Paris dont le salaire est supérieur à 3000

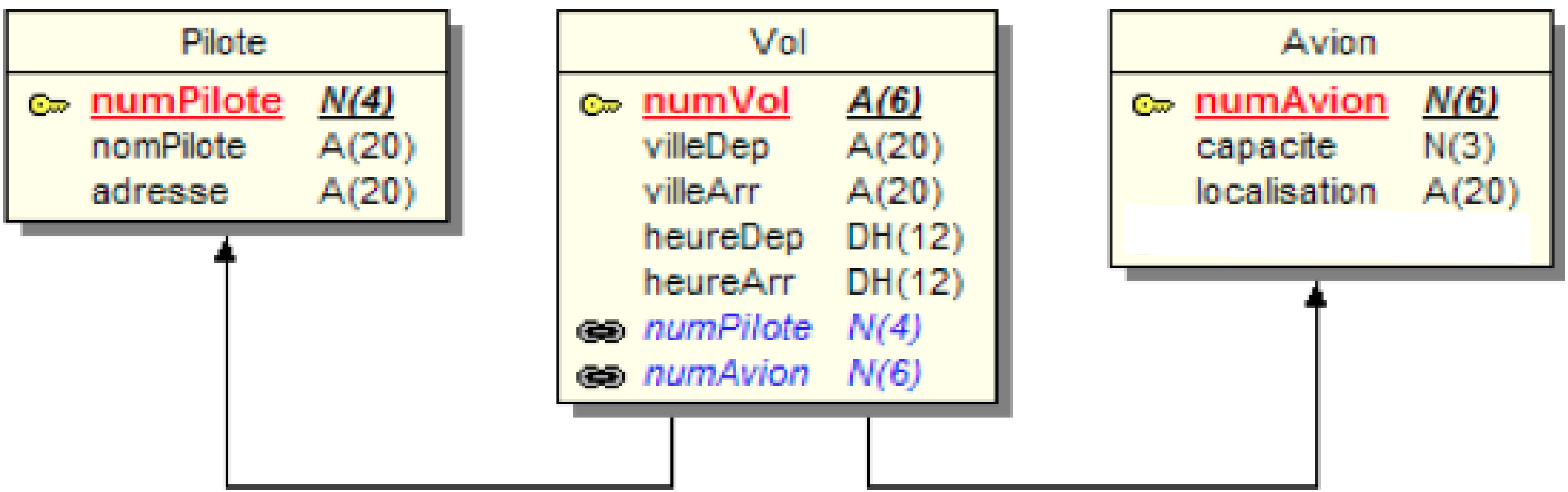
$$\Pi_{\textbf{nompilote}} (\sigma_{\textbf{adresse= 'Paris'} \wedge \textbf{salaire > 3000}} (\text{Pilote}))$$

Les opérations ensemblistes de base : l'union R∪S

Ensemble des éléments apparaissant dans R ou dans S.

$$\{u \mid u \in \mathcal{R} \vee u \in \mathcal{S}\}$$

Exemple : Quels sont les avions (numéro) localisés à Nice ou dont la capacité est inférieure à 350

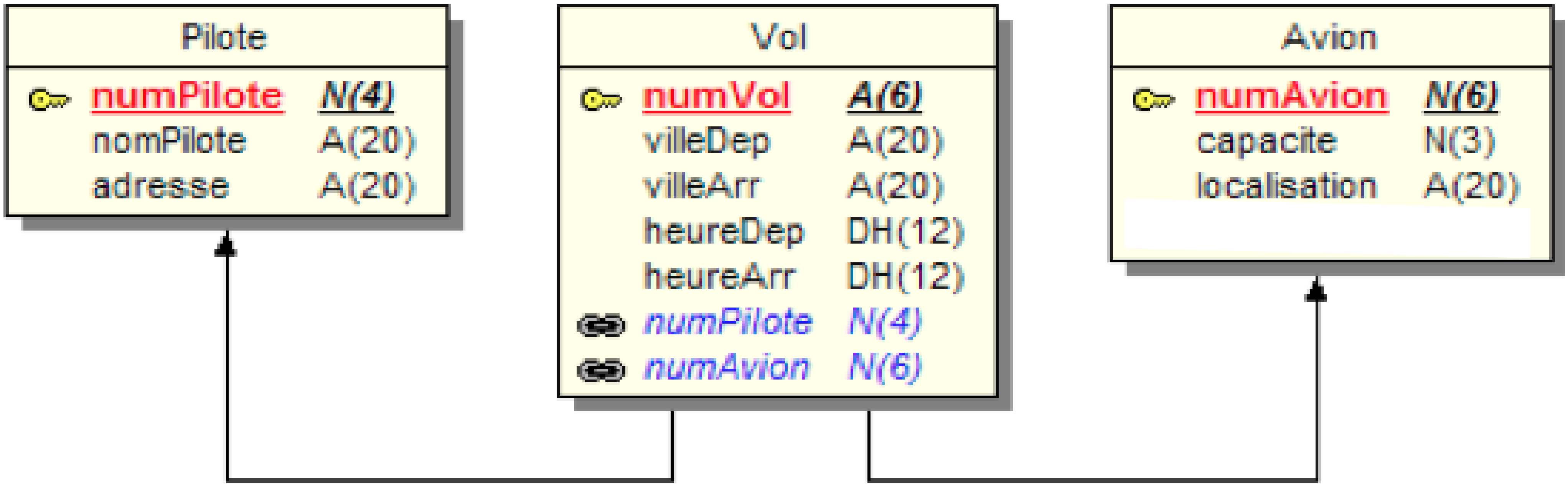


Les opérations ensemblistes de base : l'union R∪S

Exemple : Quels sont les avions (numéro) localisés à Nice ou dont la capacité est inférieure à 350

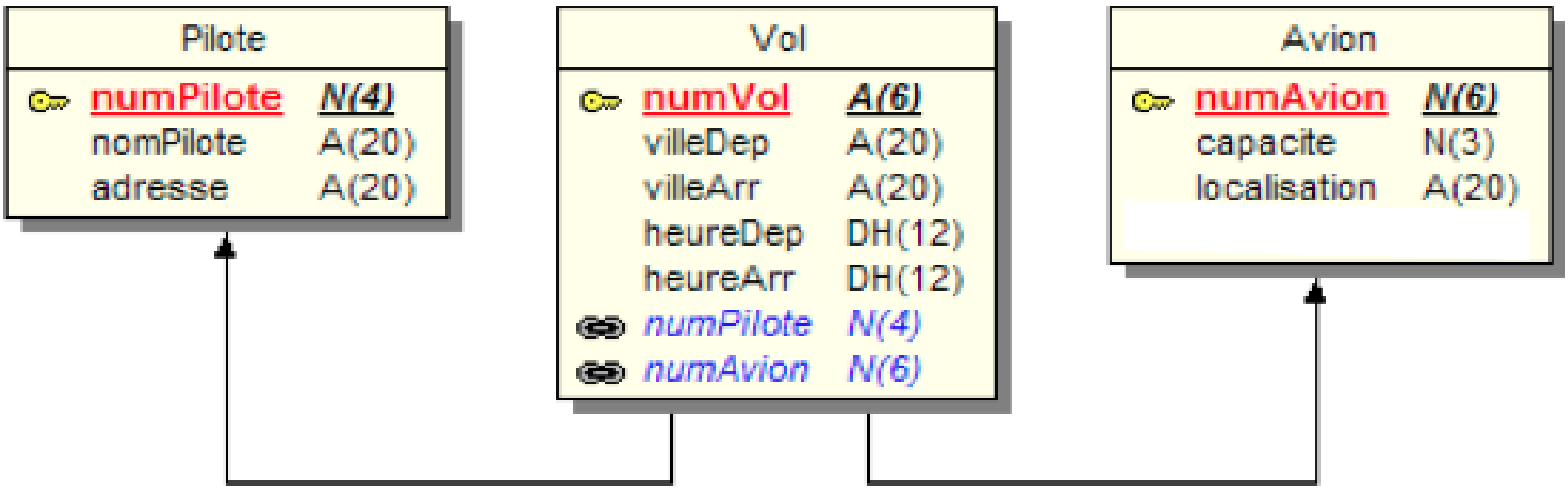
$$\Pi_{\textbf{numavion}}\ (\sigma_{\textbf{localisation= 'Nice'}}\ (\text{Avion}))\quad \cup$$

$$\Pi_{\textbf{numavion}}\ (\sigma_{\textbf{capacité<350}}\ (\text{Avion}))$$



Les opérations ensemblistes de base : l'union R∪S

Exemple : Quels sont les villes de départ, soit des vols à destination de Melbourne, soit des vols à destination de Sydney mais dont l'heure d'arrivée est prévue avant 14H.

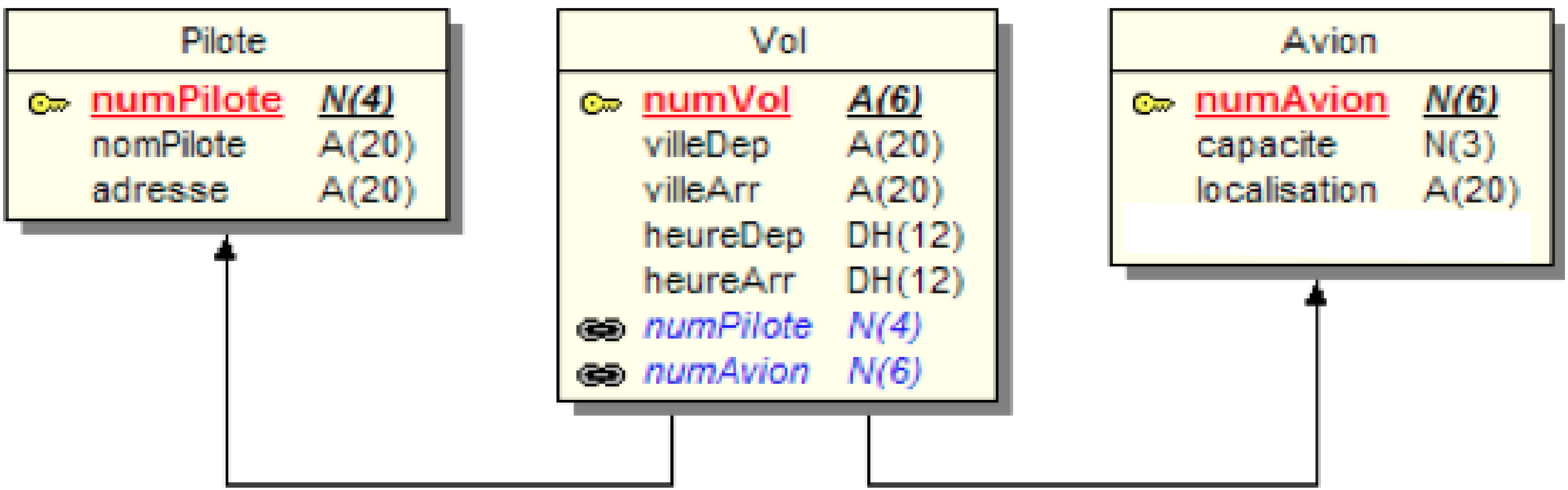


Les opérations ensemblistes de base : l'union R∪S

Exemple : Quels sont les villes de départ, soit des vols à destination de Melbourne, soit des vols à destination de Sydney mais dont l'heure d'arrivée est prévue avant 14H.

$$\Pi_{villeDep}\left(\sigma_{villeArr='Melbourne'}(Vol)\right) \cup$$

$$\Pi_{villeDep}\left(\sigma_{villeArr='Sydney' \wedge heureArr<14:00}(Vol)\right)$$



Les opérations ensemblistes de base : l'union $R \cup S$

Remarque : il est conseillé que les tables R et S soient compatibles, c'est-à-dire :

(1) Cohérence entre le nombre d'attributs :

Contre exemple : R(a1,a2) et S(a3) → $R \cup S$?

Solution : $\Pi_{a1}(R) \cup \Pi_{a3}(S)$

(2) Même groupe type du domaine (data type group) respectif :

Contre exemple : $\Pi_{a1}(R) \cup \Pi_{a2}(S)$ avec a1 caractère et a2 numérique !!!

En effet lors du passage vers ORACLE ces deux contraintes doivent être respectées.

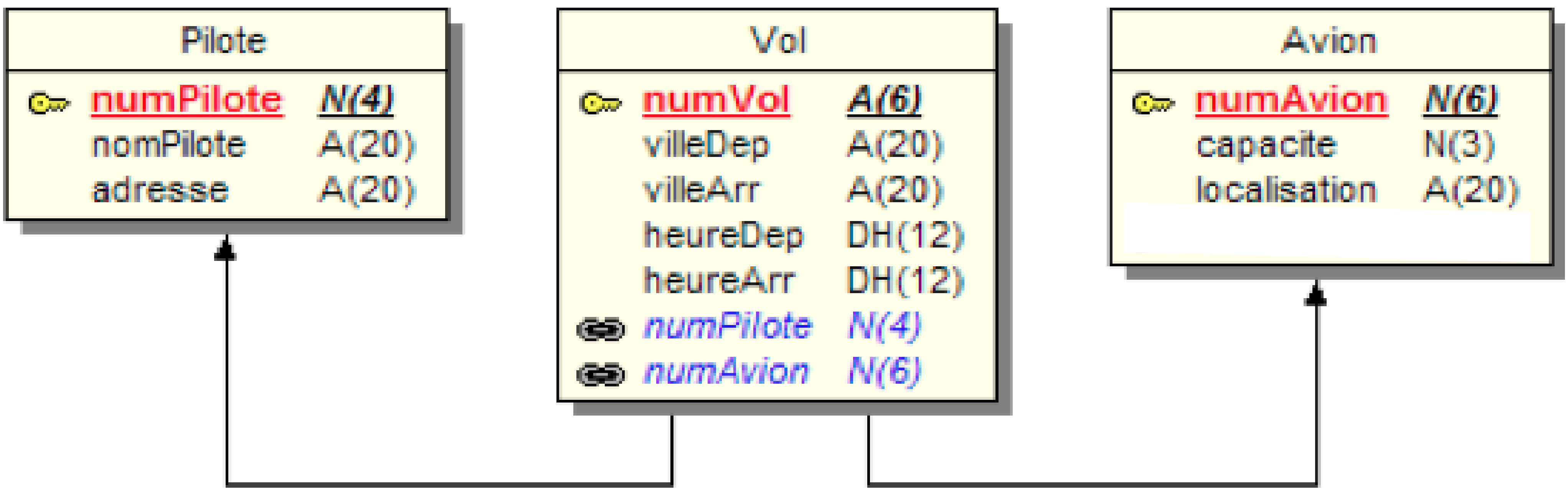
Même remarque pour l'intersection et la différence.

Les opérations ensemblistes de base : la différence R \ S

Ensemble des éléments apparaissant dans R mais pas dans S.

$$\{u \mid u \in \mathcal{R} \wedge u \notin S\}$$

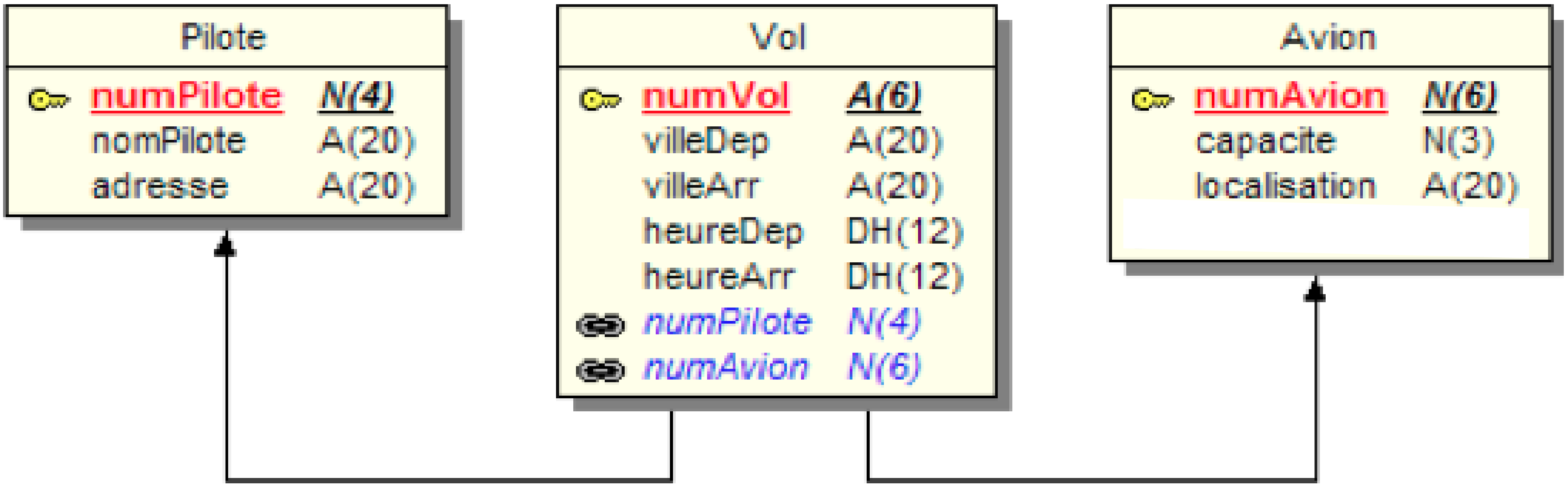
Très pratique pour les question avec « aucun » : quels sont les pilotes (numéro) qui n'assurent aucun vol ?



Les opérations ensemblistes de base : la différence R \ S

Quels sont les pilotes (numéro) qui n'assurent aucun vol ?

$$\Pi_{\textbf{numPilote}} (\text{Pilote}) \setminus \Pi_{\textbf{numPilote}} (\text{Vol})$$



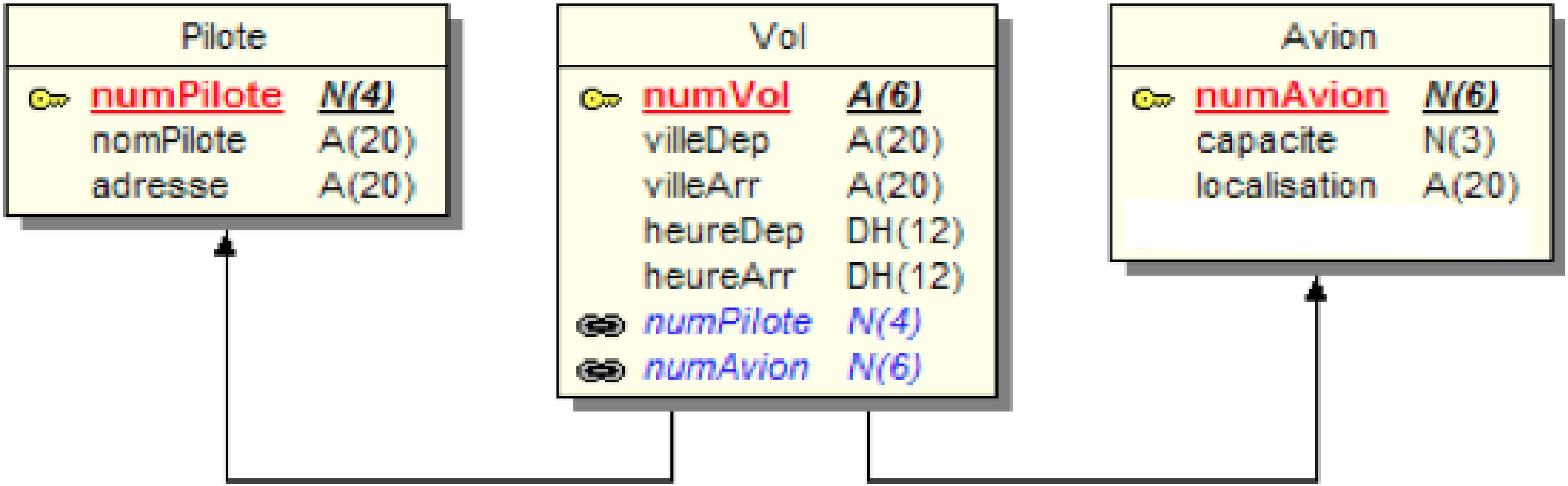
Les opérations ensemblistes de base : la différence R \ S

Donnez les numéros de pilotes qui ont fait au moins un vol mais qui n'ont jamais utilisé l'avion numéro 320.

On serait tenter de faire :

$$\Pi_{\textbf{numpilote}} (\sigma_{\textbf{numavion} \neq \textbf{320}} (\text{Vol}))$$

Mais pourquoi c'est faux ?



Les opérations ensemblistes de base : la différence R \ S

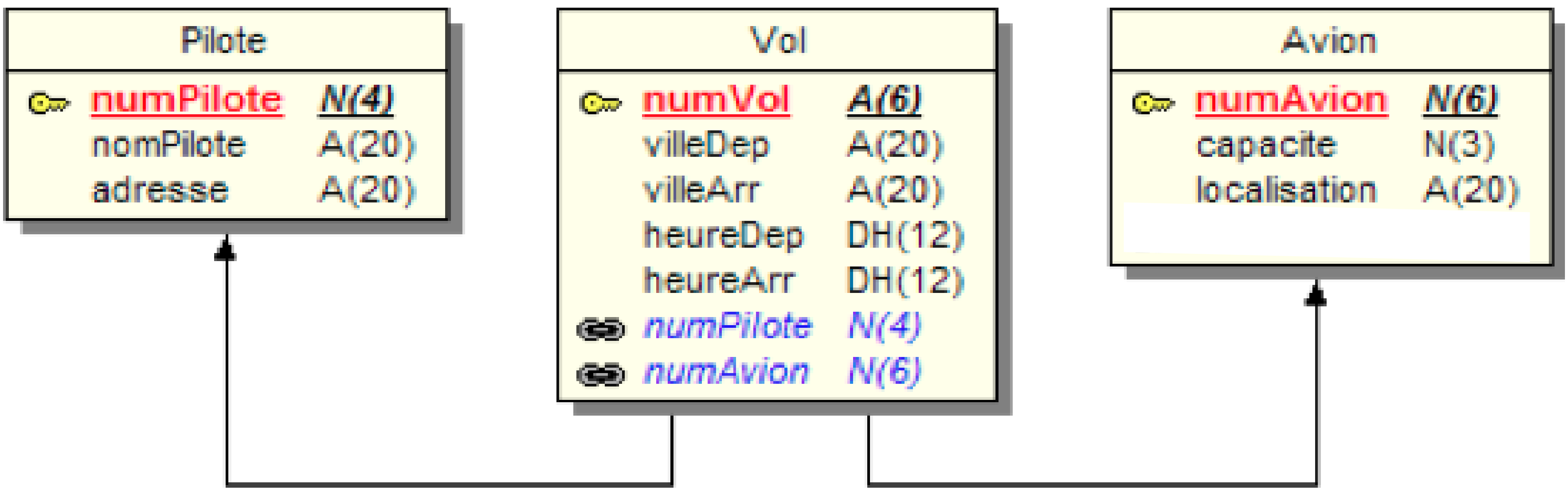
$$\Pi_{\textbf{numpilote}} (\sigma_{\textbf{numavion} \neq \textbf{320}} (\text{Vol}))$$

Supposons que le pilote 1 a fait le vol AF1 dans l'avion 320 ainsi qu'un vol AF2 dans l'avion 400

On a donc deux lignes dans la table Vol avec le pilote 1.

Notre requête va supprimer la première ligne car le numéro de l'avion est 320 mais elle va retenir la seconde ligne car le numéro de l'avion est différent de 320. Le pilote 1 sera donc affiché !

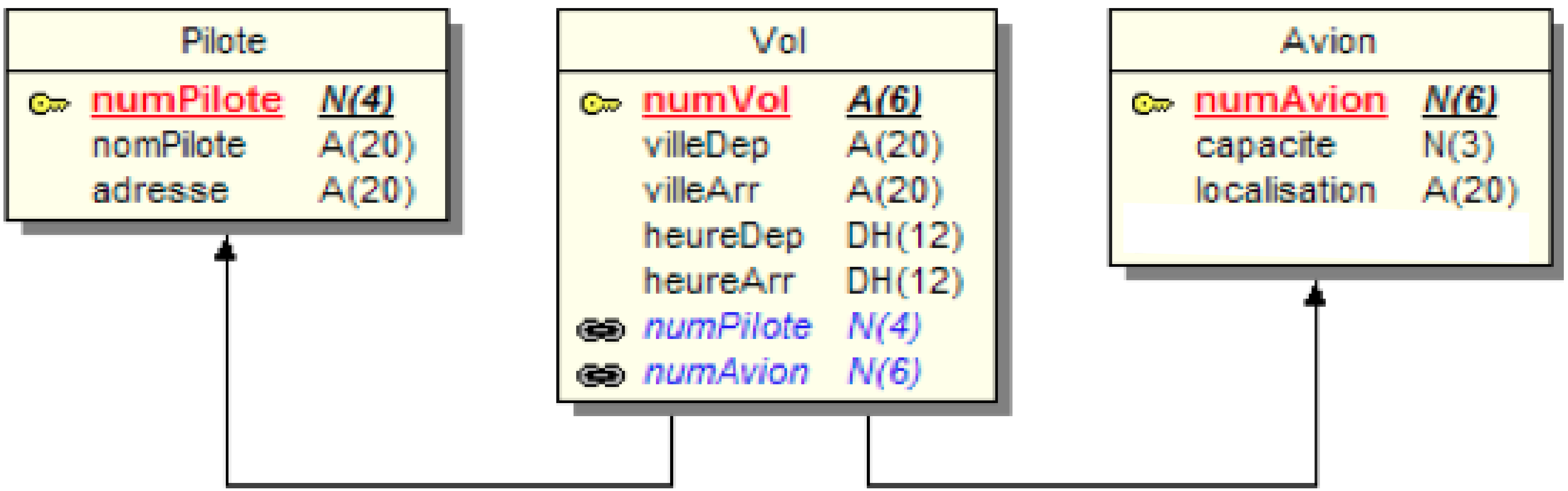
Corrigez cette requete et méfiez vous des question avec une négation dans l'énoncé !



Les opérations ensemblistes de base : la différence R \ S

Donnez les numéros de pilotes qui ont fait au moins un vol mais qui n'ont jamais utilisé l'avion numéro 320.

$$\Pi_{numpilote}(Vol) \setminus \Pi_{numpilote}(\sigma_{numavion=320}(Vol))$$



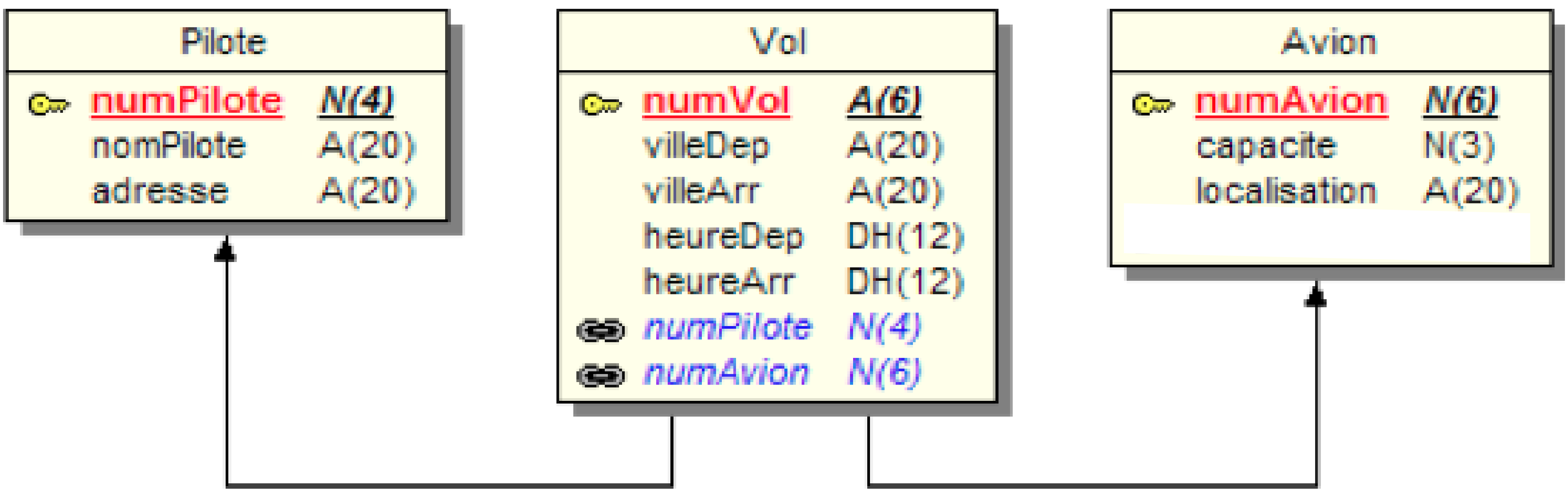
Les opérations dérivées : l'intersection R∩S

Ensemble des éléments apparaissant à la fois dans R et dans S.

$$\{u \mid u \in \mathcal{R} \wedge u \in \mathcal{S}\}$$

R ∩ s=r \ (r \ s) =s \ (s\ r)

Exemple : quels sont les heures de départ des vols à destination de Melbourne qui sont également des heures de départ de vols de Paris ou de Nice.

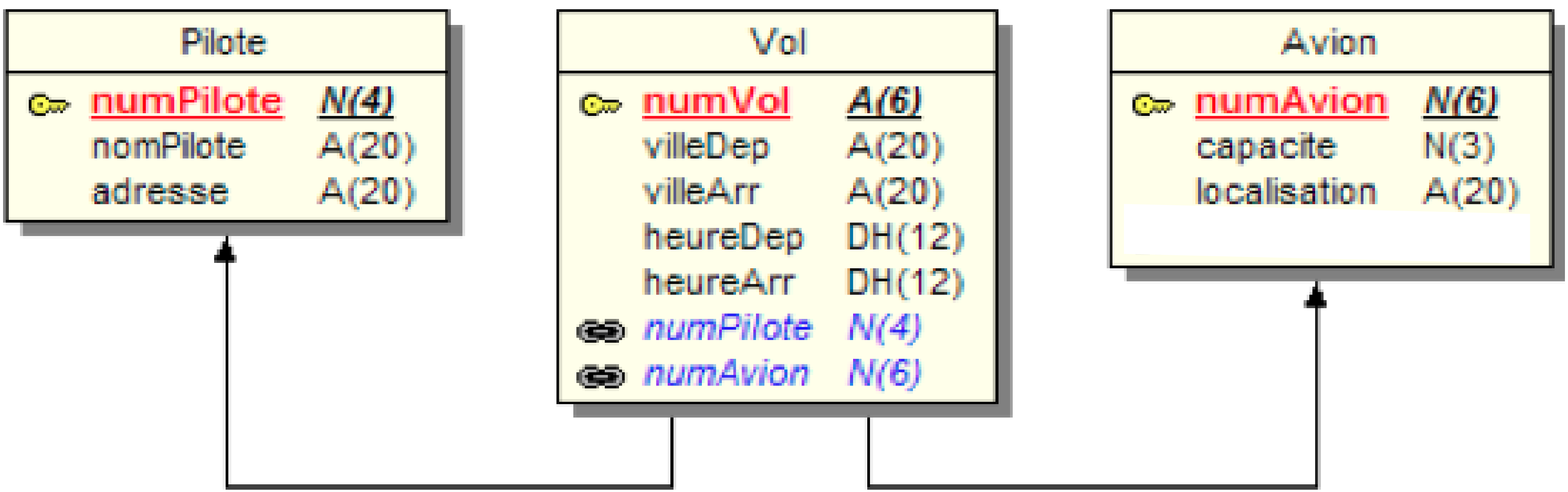


Les opérations dérivées : l'intersection R∩S

Exemple : quels sont les heures de départ des vols à destination de Melbourne qui sont également des heures de départ de vols de Paris ou de Nice.

$$\Pi_{\text{heureDep}} (\sigma_{\text{villeArr= 'Melbourne'}} (\text{Vol})) \quad \cap$$

$$\Pi_{\text{heureDep}} (\sigma_{\text{villeDep='Paris'} \vee \text{villeDep='Nice'}} (\text{Vol}))$$



Les opérations dérivées : l'intersection R∩S

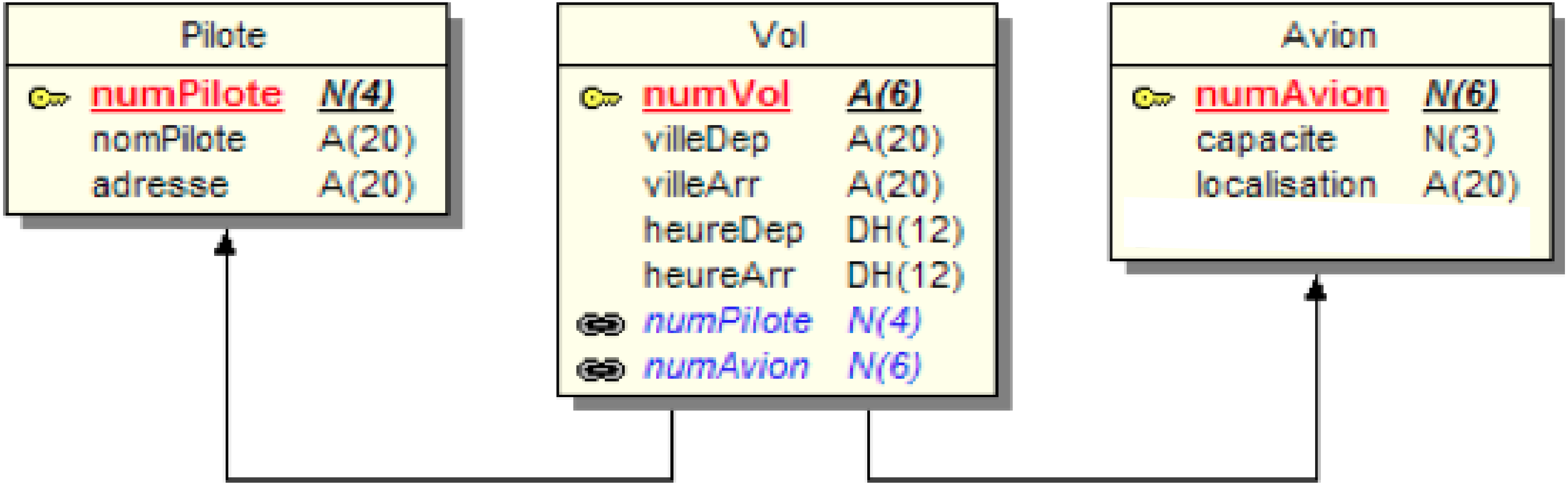
Supposons que vous avez déjà écrit les requêtes suivantes :

Req1 : liste des vols qui partent de Paris.

Req 2 : liste des vols qui arrivent à Marseille

Req 3 : liste des vols qui utilisent l'avion numéro 320

On souhaite en utilisant uniquement les requêtes 1, 2 et 3 obtenir la liste des vols qui partent de Paris et arrivent à Marseille en utilisant l'avion 320.



Les opérations dérivées : l'intersection R∩S

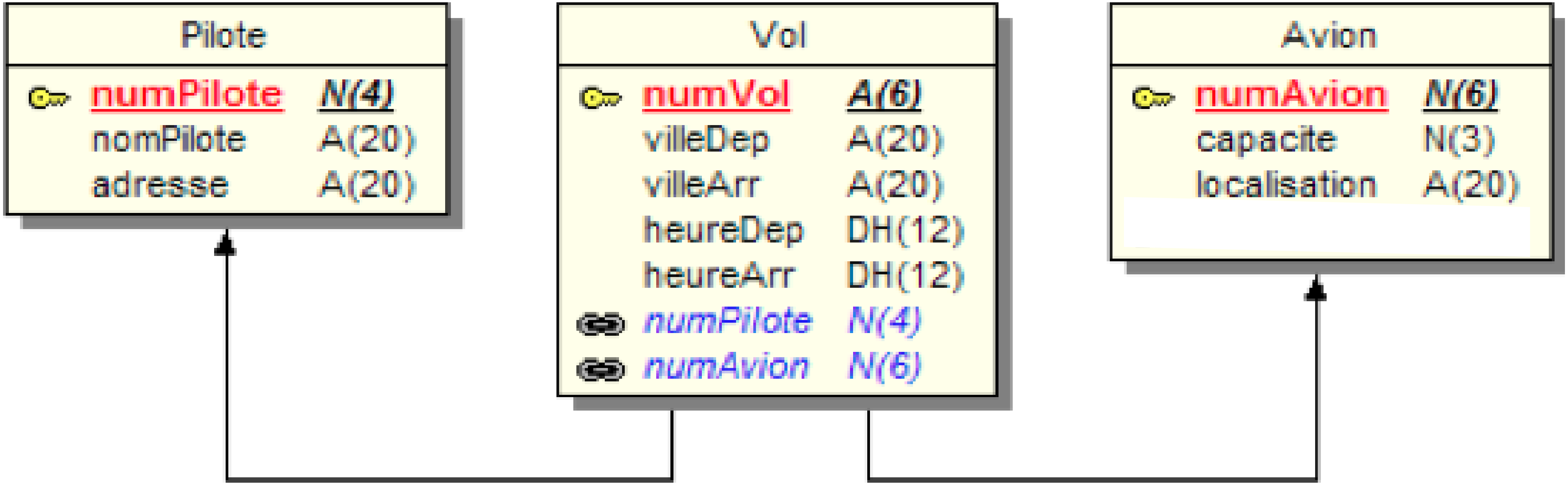
Req1 : numéros des vols qui partent de Paris.

Req 2 : numéros des vols qui arrivent à Marseille

Req 3 : numéros des vols qui utilisent l’avion numéro 320

On souhaite en utilisant uniquement les requêtes 1, 2 et 3 obtenir la liste des vols qui partent de Paris et arrivent à Marseille en utilisant l’avion 320.

Req1 ∩ Req1 ∩ Req1



Les opérations ensemblistes de base : le produit cartésien R × S

Ensemble des tuples constitués à partir d'un élément de R et d'un élément de S.

$$\{u | \exists v \in \mathcal{R}, \exists w \in \mathcal{S}, u = (v_1, \ldots, v_n, w_1, \ldots, w_m)\}$$

Attention au coût de cette opération !

r

A	B	C
a	b	c
d	a	f
c	b	d

s

A	B	C
b	g	a
d	a	f

r × s

r.A	r.B	r.C	s.A	s.B	s.C
a	b	c	b	g	a
a	b	c	d	a	f
d	a	f	b	g	a
d	a	f	d	a	f
c	b	d	b	g	a
c	b	d	d	a	f

Les opérations dérivées : la jointure

La jointure sous toute ses formes est une opération essentielle pour les BDD relationnelles car elle permet une utilisation raisonnable du produit cartésien et permet d'assurer très simplement la navigation dans les BDD relationnelles.

La théta-jointure $R \bowtie_C S$

Constituée des n-uplets de $R \times S$ qui satisfont la condition C.

$R \bowtie_C S = \sigma_C (R \times S)$

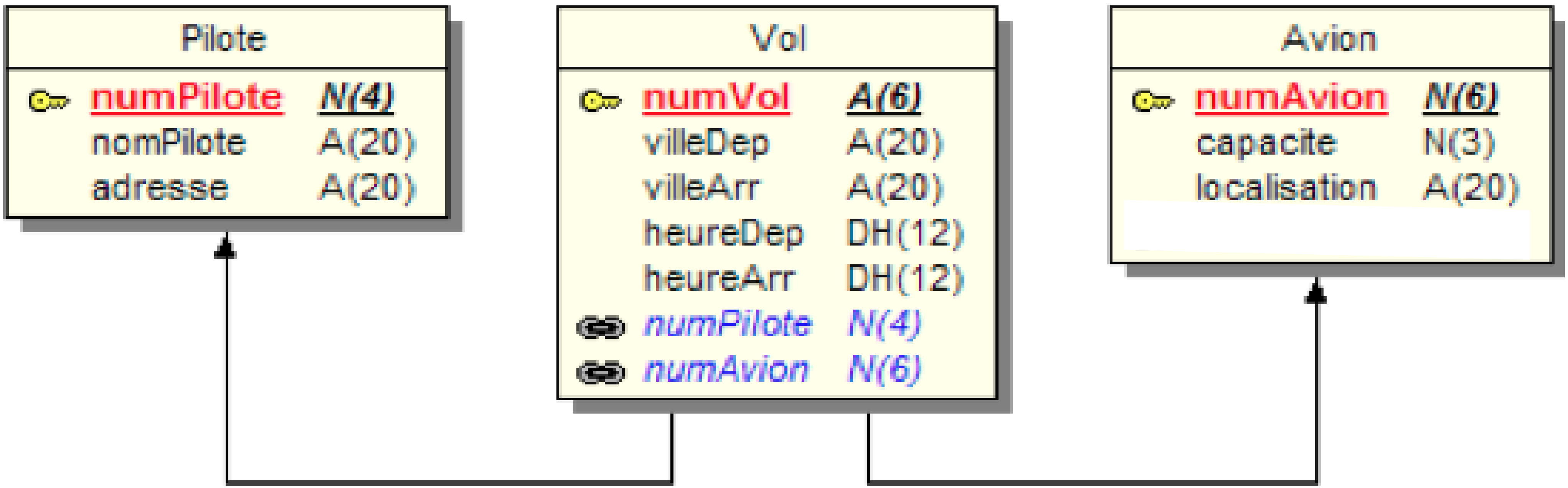
r		
A	**B**	**C**
1	2	3
4	5	6
7	8	9

s	
D	**E**
3	1
6	2

$r \bowtie_{B<D} s$				
A	**B**	**C**	**D**	**E**
1	2	3	3	1
1	2	3	6	2
4	5	6	6	2

La théta-jointure R $\bowtie_c$ S

Exemple : quels sont les numéros des vols au départ de Londres et à destination de Paris dont l'heure d'arrivée prévue est antérieure à l'heure de départ de vol au départ de Paris et à destination de Melbourne.

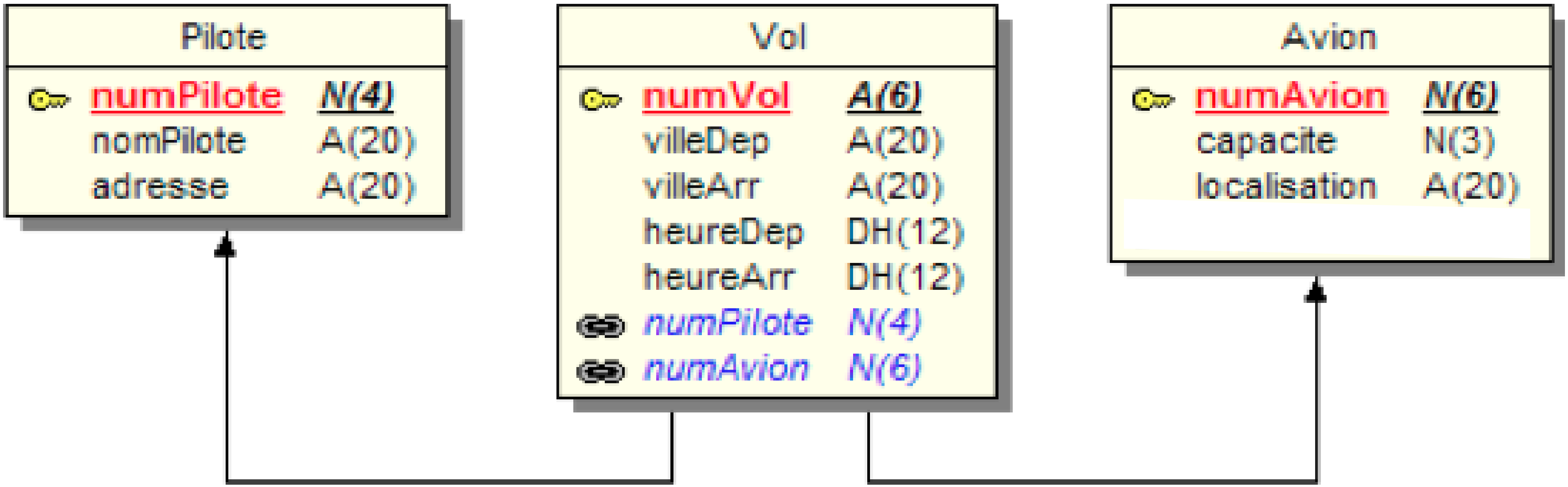


Indication : on doit utiliser une jointure de la table vol sur elle même

Problème : ambiguité des noms des attributs → renommage.

La théta-jointure R $\bowtie_c$ S

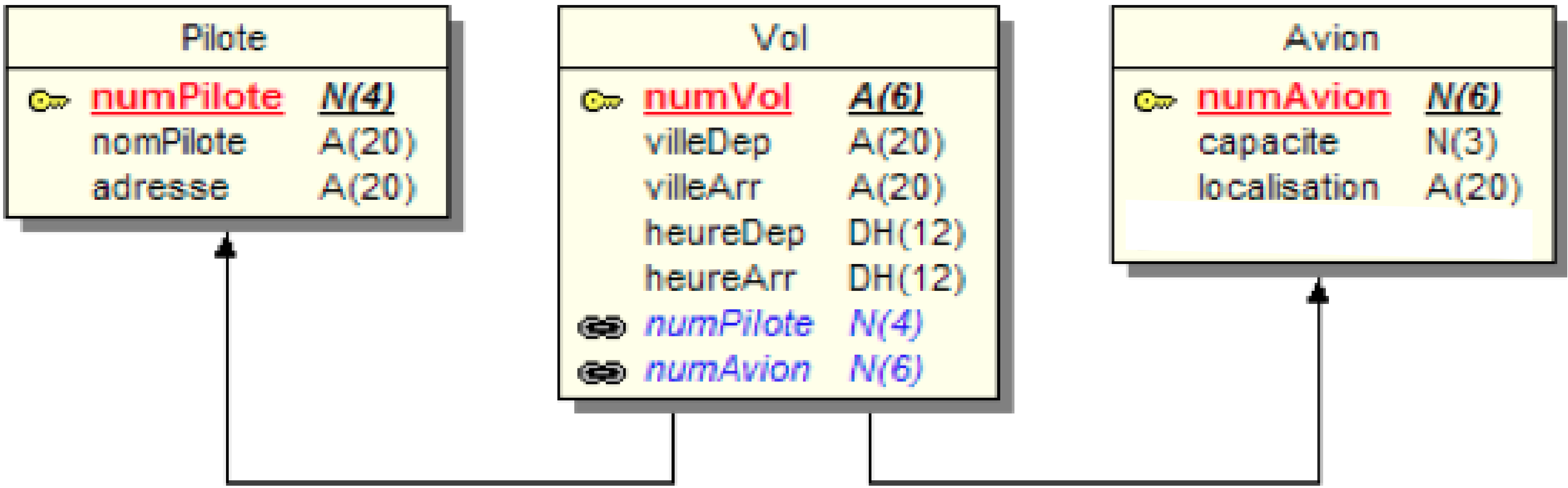
Exemple : quels sont les numéros des vols au départ de Londres et à destination de Paris dont l'heure d'arrivée prévue est antérieure à l'heure de départ de vol au départ de Paris et à destination de Melbourne.



Solution 1 : suffixer les noms de tables par des entiers → vol qui devient vol_1 et vol_2

La théta-jointure R $\bowtie_c$ S

Exemple : quels sont les numéros des vols au départ de Londres et à destination de Paris dont l'heure d'arrivée prévue est antérieure à l'heure de départ de vol au départ de Paris et à destination de Melbourne.



Solution 1 : suffixer les noms de tables par des entiers → vol qui devient vol_1 et vol_2

$$\Pi_{numVol\_1}\left(\sigma_{villeDep\_1=\text{'Londre'}\ \wedge\ villeArr\_1=\text{'Paris'}}(Vol\_1)\ \bowtie_{heureArr\_1<heureDep\_2}\ \sigma_{villeDep\_2=\text{'Paris'}\ \wedge\ villeArr\_2=\text{'Melbourne'}}(Vol\_2)\right)$$

Solution 2 : créer une opération de base dédiée au renommage → $\rho_{S\,(A1\,,\ldots,An\,)}\ (R)$

Il en existe deux formes :

$\rho_{S\,(A1\,,\ldots,An\,)}\ (R)$: relation obtenue en renommant R en S et ses attributs en A1, . . . , An.

$\rho_{S}\ (R)$: relation obtenue en renommant R en S. Les attributs sont accessibles via S.nomAttribut

Reprenons l'exemple : quels sont les numéros des vols au départ de Londres et à destination de Paris dont l'heure d'arrivée prévue est antérieure à l'heure de départ de vol au départ de Paris et à destination de Melbourne.

Solution 2 : renommage via ρ_{Vol_1} (Vol) et ρ_{Vol_2} (Vol)

L'exemple précédent s'écrit plus élégamment :

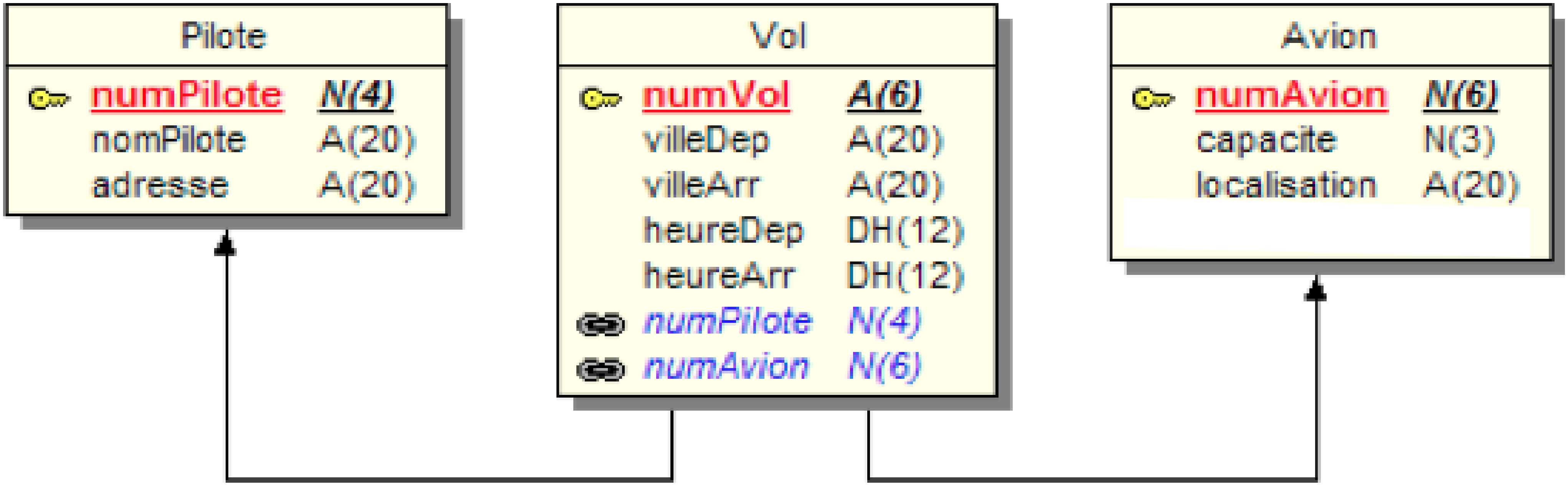
$$\Pi_{vol\_1.numVol} \Big(\sigma_{vol\_1.villeDep='Londre' \wedge vol\_1.villeArr='Paris'} (\rho_{Vol\_1}(Vol))$$

$$\bowtie_{vol\_1.heureArr<vol\_2.heureDep}$$

$$\sigma_{vol\_2.villeDep='Paris' \wedge vol\_2.villeArr='Melbourne'} (\rho_{Vol\_2}(Vol)) \Big)$$

La théta-jointure R $\bowtie_C$ S

Exemple : Quels sont les pilotes (numéro et nom) habitant dans la même ville que le pilote Dupont



La théta-jointure R $\bowtie_C$ S

Exemple : Quels sont les pilotes (numéro et nom) habitant dans la même ville que le pilote Dupont

$$\Pi_{\text{pilote\_2.numPilote,pilote\_2.nomPilote}} \Big(\sigma_{\text{pilote\_1.nomPilote='Dupont'}} (\rho_{\text{pilote\_1}}(\text{Pilote})) \bowtie_{\text{pilote\_1.adresse=pilote\_2.adresse}} \sigma_{\text{pilote2.nomPilote}\neq\text{'Dupont'}} (\rho_{\text{pilote\_2}}(\text{Pilote})) \Big)$$

On utilise $\sigma_{\text{adresse}\neq\text{'Dupont'}}$ pour ne pas avoir Dupont dans la réponse (dupont habite dans la même ville que lui même)

La théta-jointure R $\bowtie_C$ S

Exemple : Quels sont les pilotes (numéro et nom) habitant dans la même ville que le pilote Dupont

Pourquoi cette autre solution est bonne mais trop lente !

$$\Pi_{\text{pilote\_2.numPilote, pilote\_2.nomPilote}} \Big(\sigma_{\text{pilote1.nomPilote='Dupont'}} \big(\rho_{\text{pilote\_1}}(\text{Pilote}) \bowtie_{\text{pilote\_1.adresse=pilote\_2.adresse}} \sigma_{\text{pilote2.nomPilote}\neq\text{'Dupont'}} (\rho_{\text{pilote\_2}}(\text{Pilote})) \big) \Big)$$

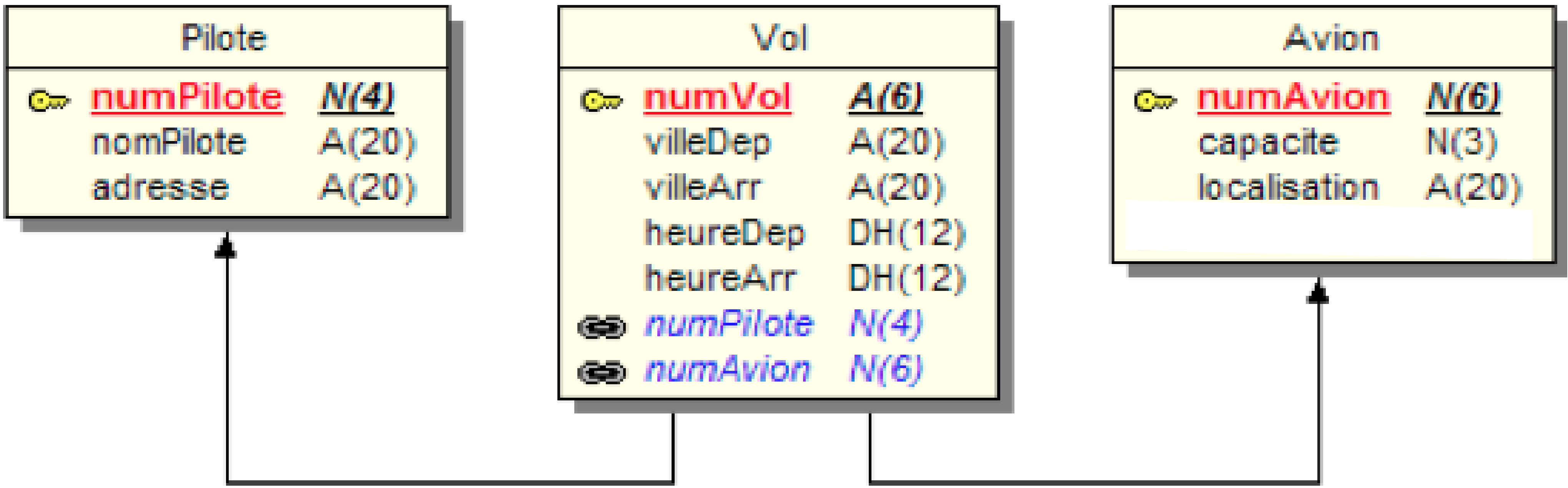
La théta-jointure R $\bowtie_c$ S

Pourquoi cette autre solution est bonne mais trop lente : la restriction ne se fait qu'une fois toutes les combinaisons générées : ces combinaisons ne nous intéressent pas toutes vu que seules celles qui contiennent l'adresse de Dupont nous intéressent. Dans la diapo précédente les combinaisons ne sont crées qu'une fois l'adresse de Dupont calculée.

$$\Pi_{\text{pilote\_2.numPilote, pilote\_2.nomPilote}} \Big(\sigma_{\text{pilote1.nomPilote='Dupont'}} \Big(\rho_{\text{pilote\_1}}(\text{Pilote}) \bowtie_{\text{pilote\_1.adresse=pilote\_2.adresse}} \sigma_{\text{pilote2.nomPilote} \neq \text{'Dupont'}} (\rho_{\text{pilote\_2}}(\text{Pilote})) \Big) \Big)$$

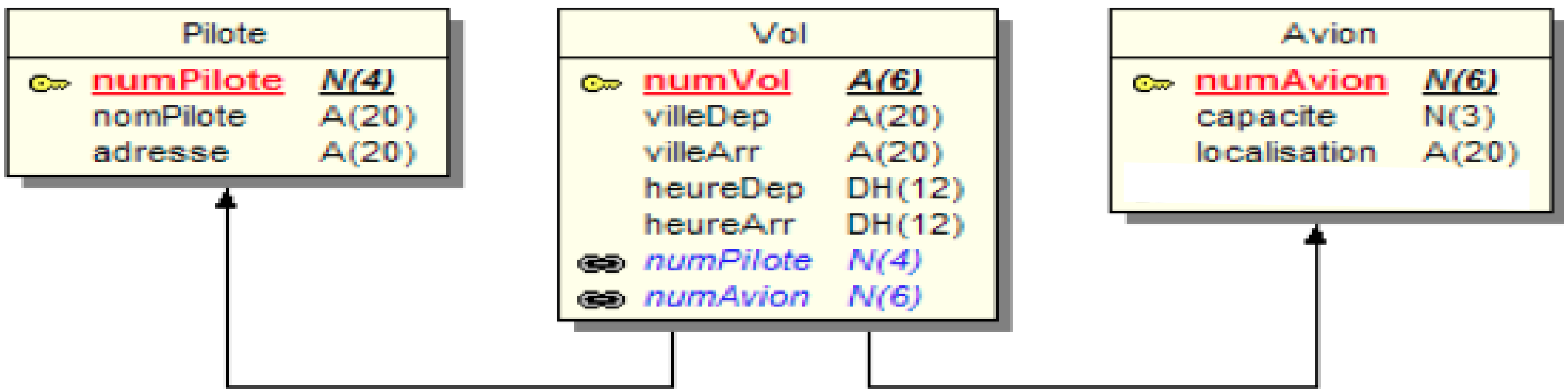
La théta-jointure R $\bowtie_c$ S

Exemple : Y a-t-il des homonymes parmi les pilote. Si oui, donner leur numéro et nom.



La théta-jointure R $\bowtie_C$ S

Exemple : Y a-t-il des homonymes parmi les pilote. Si oui, donner leur numéro et nom.



$$\Pi_{\text{pilote\_1.numPilote, pilote\_1.nomPilote}} \Big(\rho_{\text{pilote\_1}}(\text{Pilote}) \bowtie_{\text{pilote\_1.nomPilote=pilote\_2.nomPilote} \wedge \text{pilote\_1.numPilote} \neq \text{pilote\_2.numPilote}} \rho_{\text{pilote\_2}}(\text{Pilote}) \Big)$$

La condition $\text{pilote\_1.numPilote} \neq \text{pilote\_2.numPilote}$ empeche la jointure de produire un pilote avec lui même.

La jointure naturelle R ⋈ S

Obtenue par une theta jointure de R et S selon l'égalité de valeur de tous les attributs communs mais on ne garde qu'une seul attribut (colonne) pour chaque couple d'attributs communs.

$$R \bowtie S = \{u \mid \exists v \in R(A_1,\ldots,A_k,B_{k+1},\ldots,B_n),$$

$$\exists w \in S(A_1,\ldots,A_k,C_{k+1},\ldots,C_m),$$

$$(v_1,\ldots,v_k) = (w_1,\ldots,w_k) \wedge$$

$$u = (v_1, \ldots, v_n, w_{k+1}, \ldots, w_m)\}$$

r		
A	B	C
a	b	c
d	b	c
b	b	f
c	a	d

s		
B	C	D
b	c	d
b	c	e
a	d	b

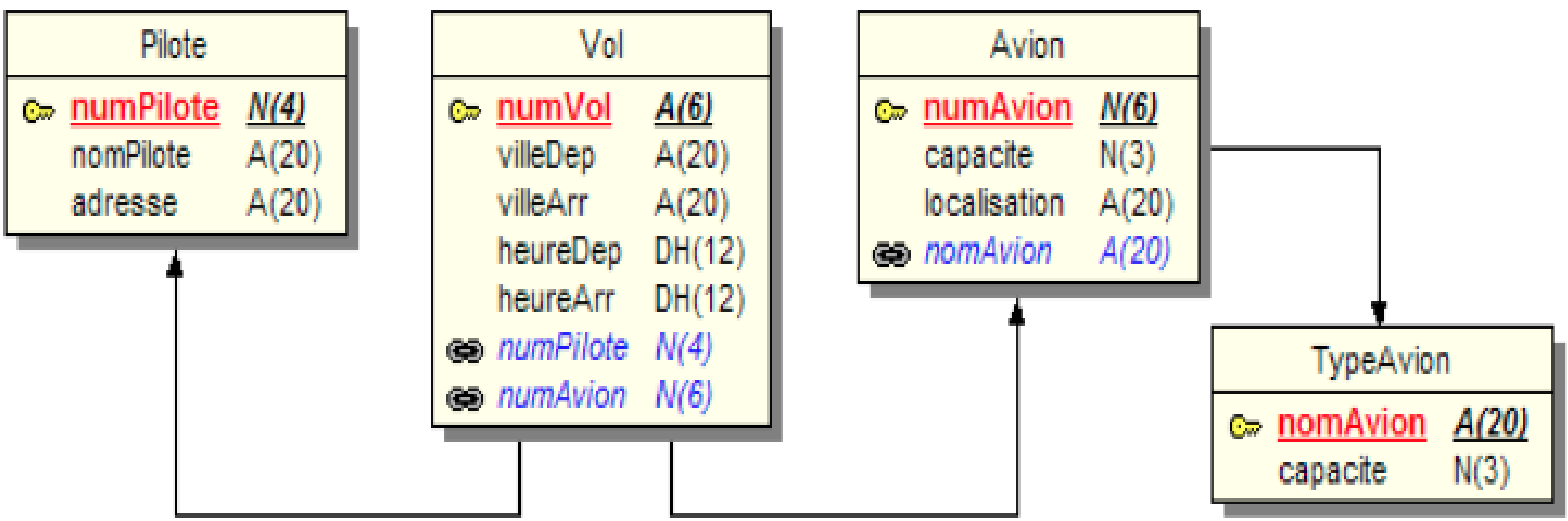
r ⋈ s			
A	B	C	D
a	b	c	d
a	b	c	e
d	b	c	d
d	b	c	e
c	a	d	b

On voit bien que les colonnes communes B et C sont présentent qu'une seul fois dans la table R ⋈ S

La jointure naturelle R ⋈ S

Revenons à l'exemple : quels sont les noms des pilotes qui ont déjà fait un vol ?

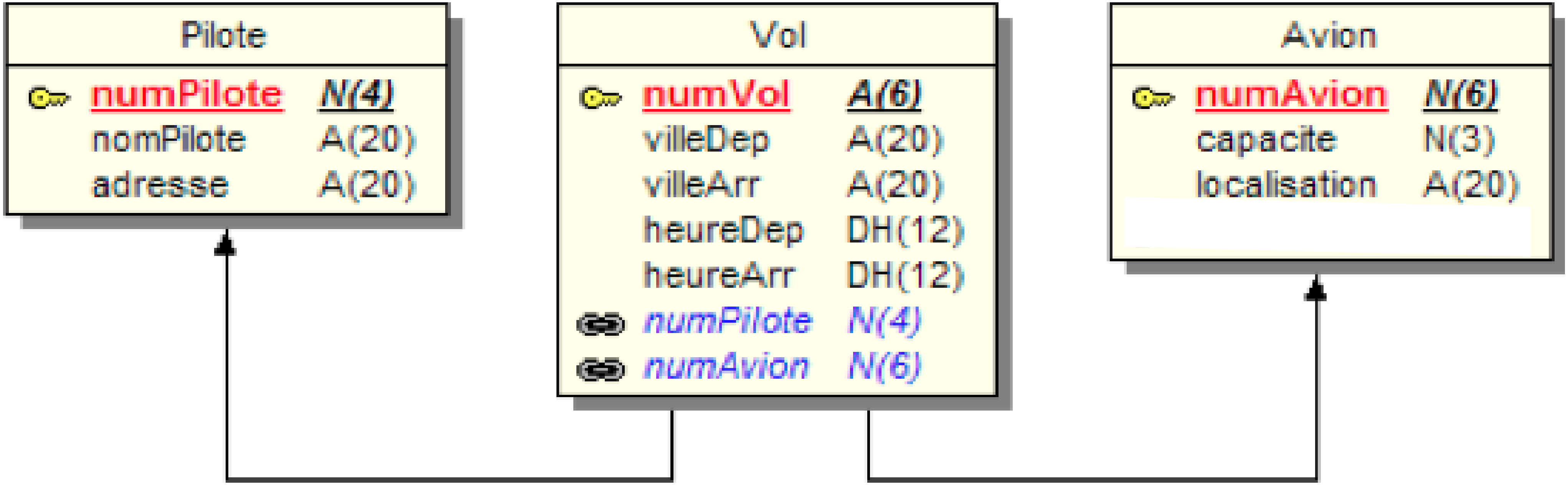
L'information est dispatchée sur deux tables : Vol et Pilote



La jointure naturelle R ⋈ S

Revenons à l'exemple : quels sont les noms des pilotes qui ont déjà fait un vol ?

L'information est dispatchée sur deux tables : Vol et Pilote



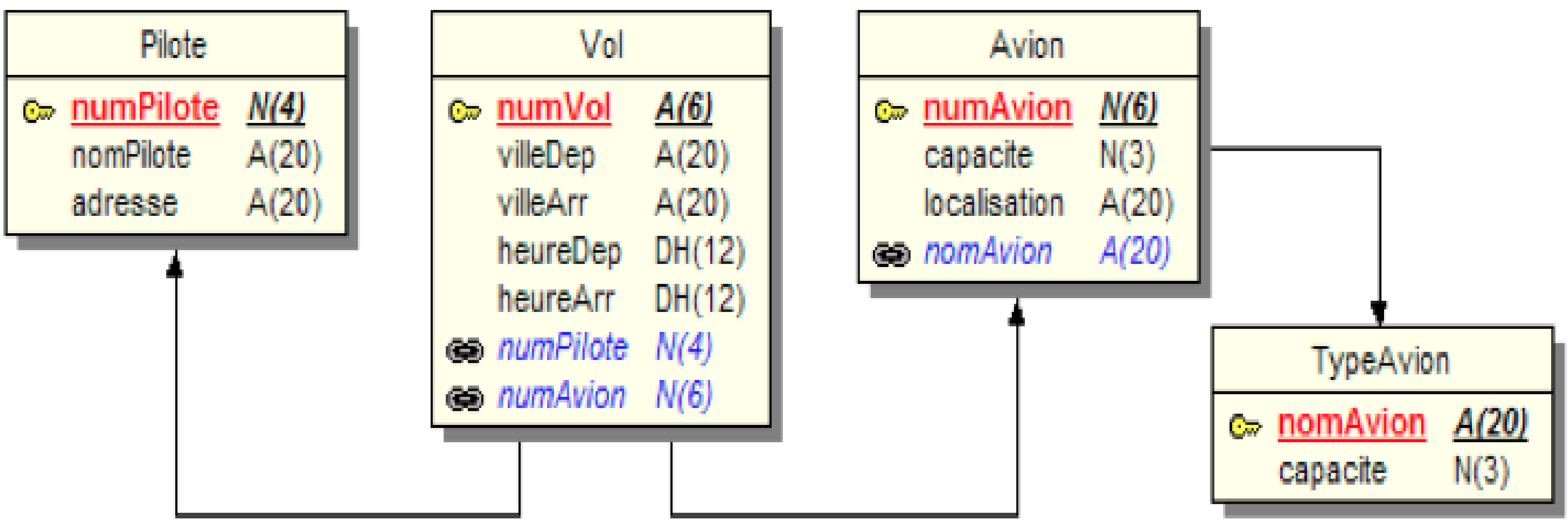
$$\Pi_{\text{nomPilote}} (\text{Pilote} \bowtie \text{Vol})$$

La jointure naturelle R ⋈ S

Revenons à l'exemple : quels sont les pilotes (numéro de pilote) qui n'assurent aucun vol ? Nous avions répondu à cette question en donnant uniquement le numéro de pilote :

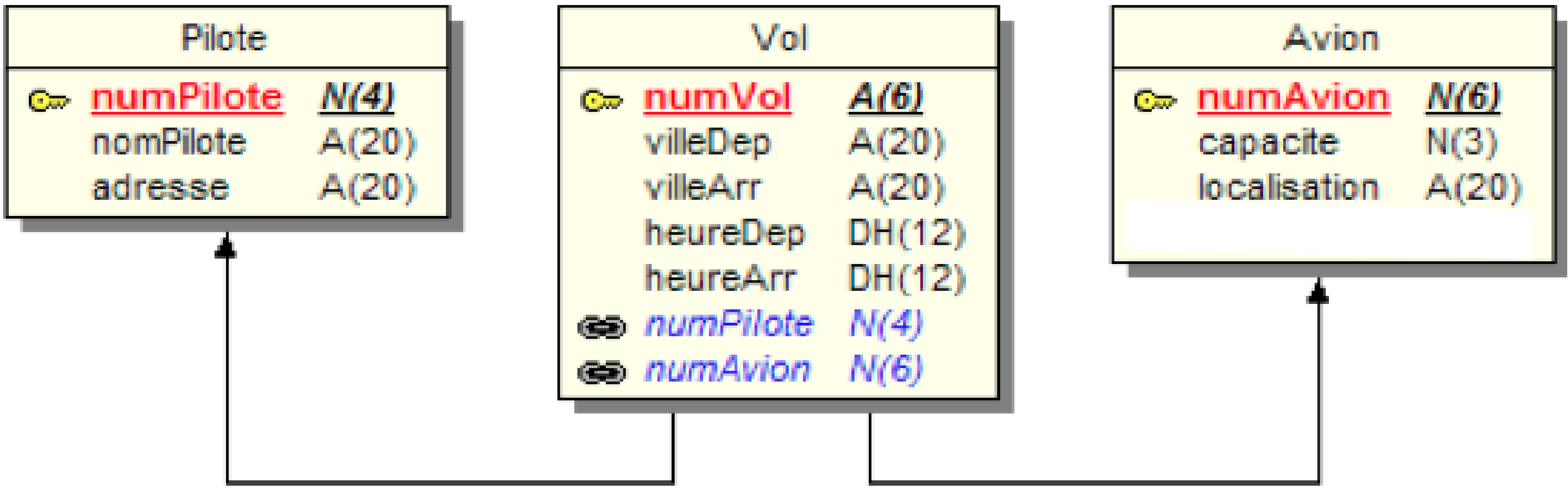
$$\Pi_{\mathbf{numPilote}} (\text{Pilote}) \setminus \Pi_{\mathbf{numPilote}} (\text{Vol})$$

Comment l'étendre pour avoir le nom et l'adresse ?



La jointure naturelle R ⋈ S

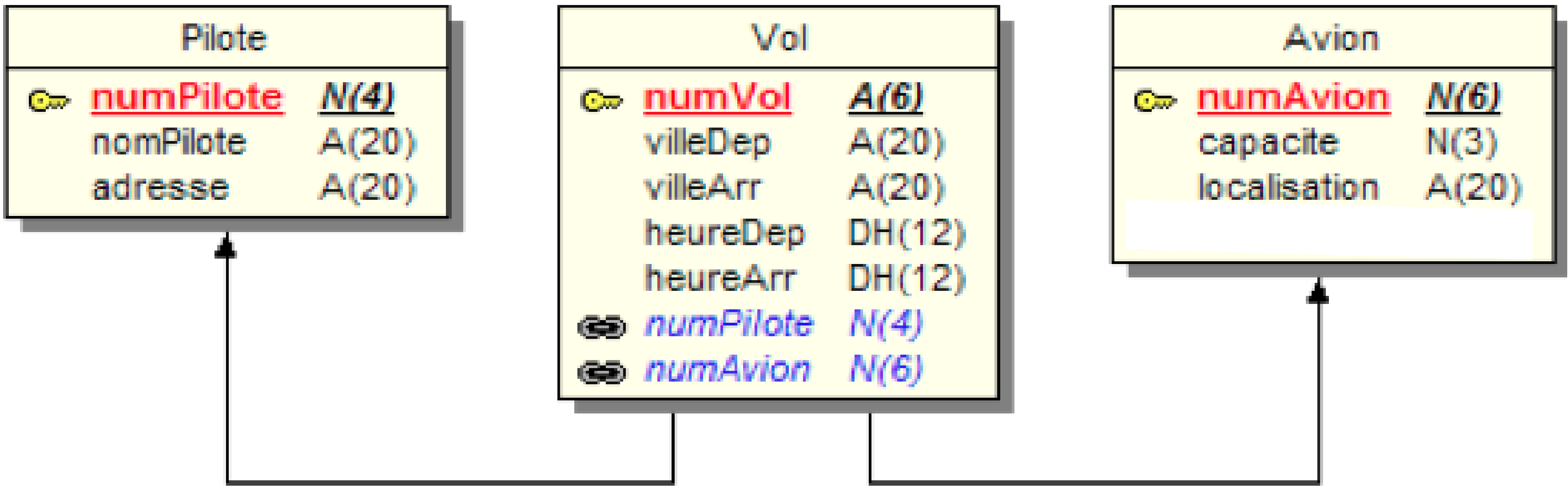
Exemple : quels sont les pilotes (nom et adresse) qui n'assurent aucun vol ?



$$\Pi_{\text{nomPilote,adresse}} \left(\text{Pilote} \bowtie \left(\Pi_{\text{numPilote}} (\text{Pilote}) \setminus \Pi_{\text{numPilote}} (\text{Vol}) \right) \right)$$

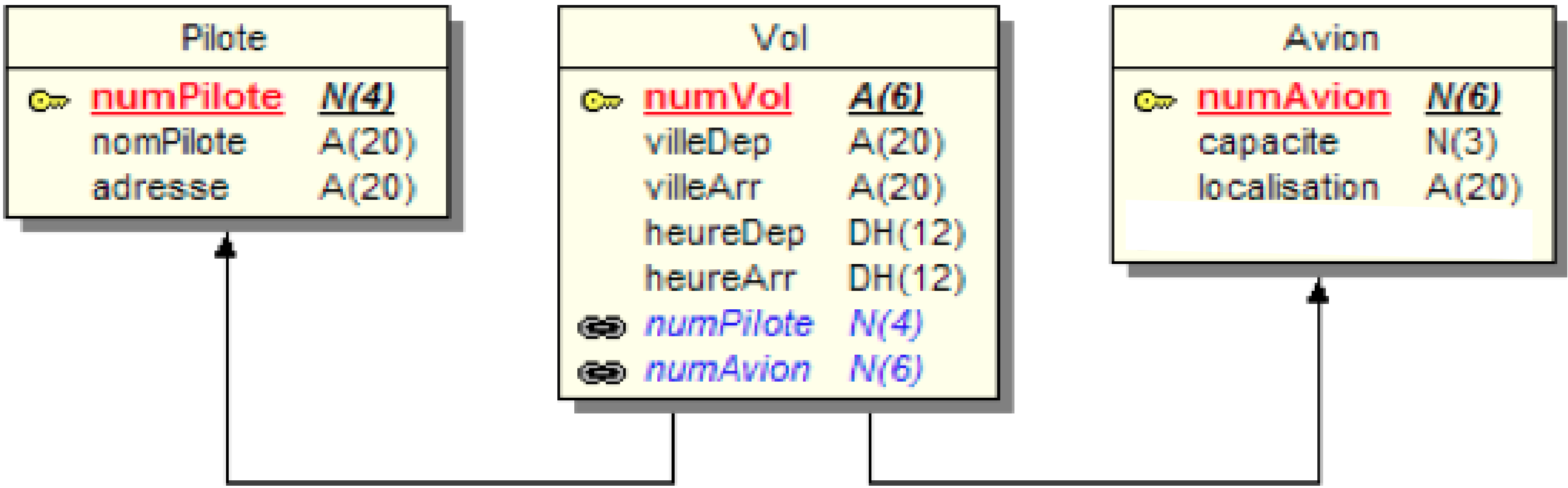
La jointure naturelle R ⋈ S

Exemple : Donnez le numéro des vols effectués par des pilotes dont l'adresse est Nice ?



La jointure naturelle R $\bowtie$ S

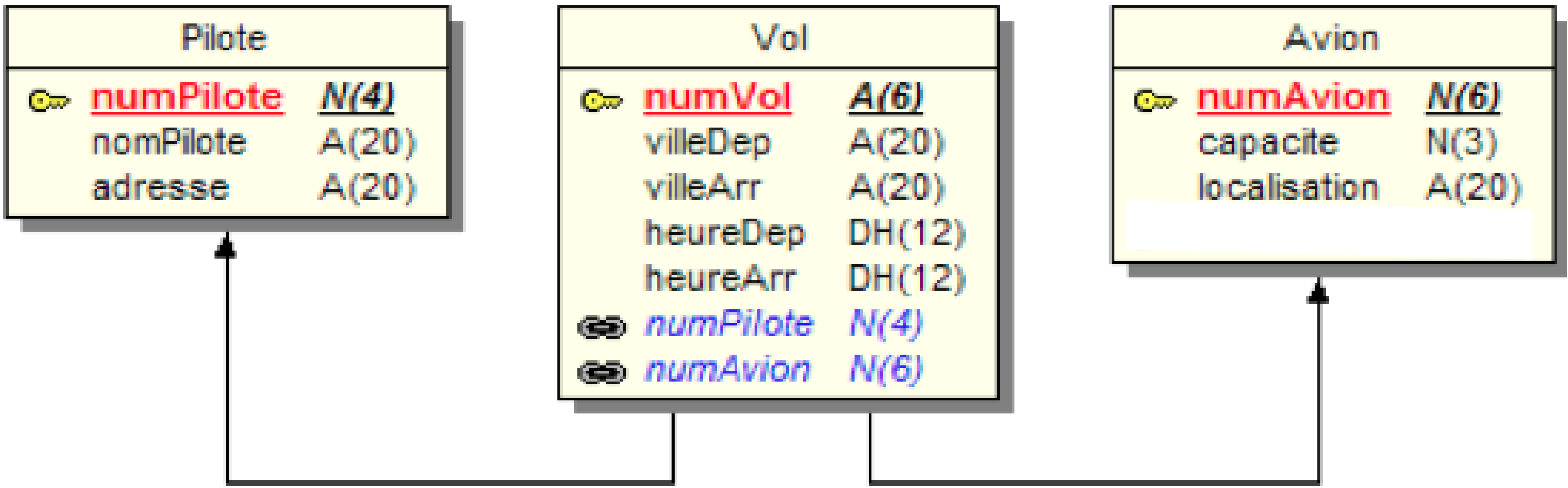
Exemple : Donnez le numéro des vols effectués par des pilotes dont l'adresse est Nice ?



$$\Pi_{\textbf{numvol}} \left(\sigma_{\text{adresse='Nice'}} (\text{Pilote}) \bowtie \text{Vol} \right)$$

La jointure naturelle R ⋈ S

Exemple : Donnez le numéro des vols effectués par des pilotes dont l'adresse est Nice ?

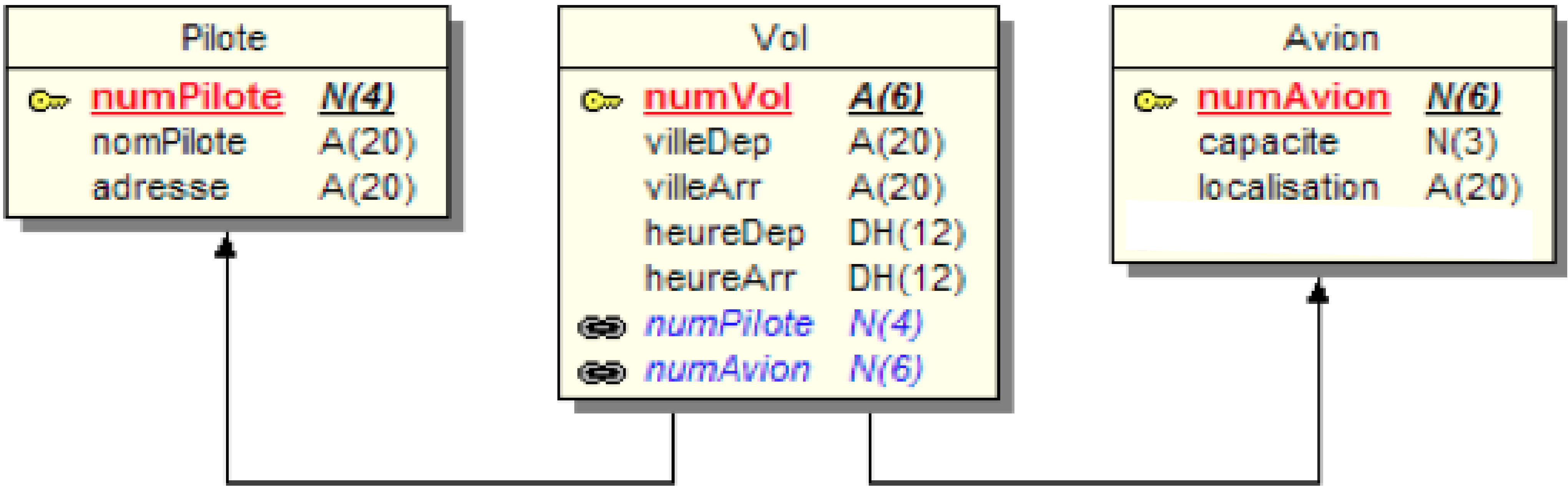


Attention à ne pas faire : $\Pi_{\textbf{numvol}} (\sigma_{\text{adresse='Nice'}} (\text{Pilote} \bowtie \text{Vol}))$

Qui effectue d'abord la jointure puis filtre les lignes : pertes de temps et espace

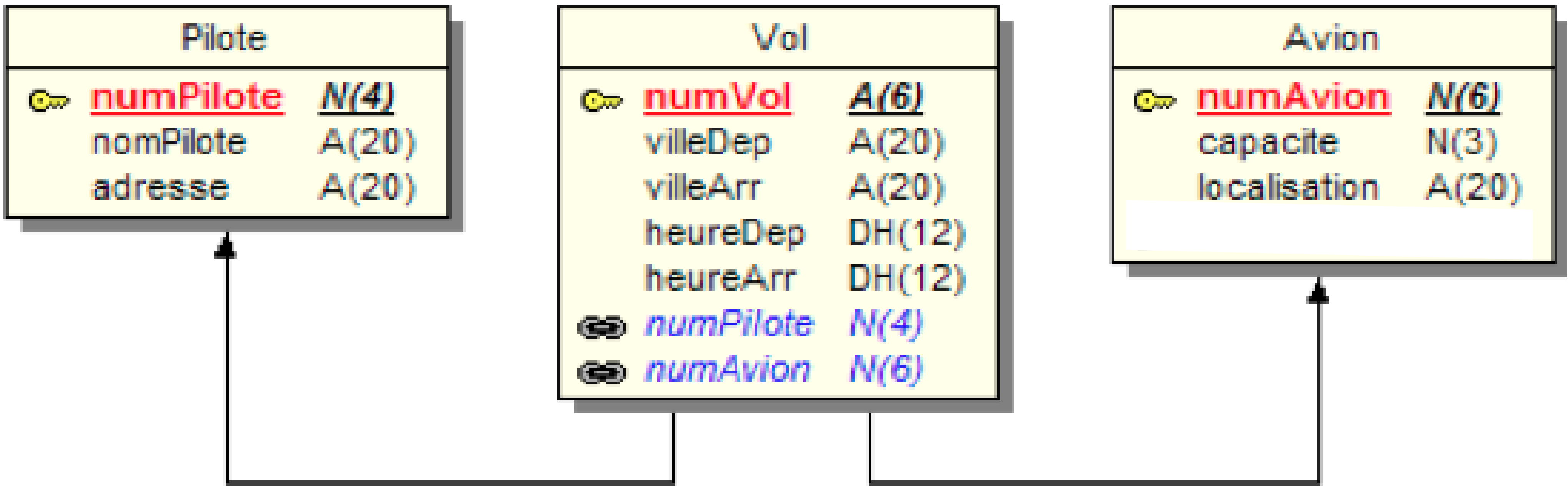
La jointure naturelle R ⋈ S

Rajoutons une contrainte sur les vols par rapport à la requête précédente :
Donnez le numéro des vols effectués **au départ de Nice** par des pilotes Niçois ?



La jointure naturelle R $\bowtie$ S

Rajoutons une contrainte sur les vols par rapport à la requête précédente :
Donnez le numéro des vols effectués **au départ de Nice** par des pilotes Niçois ?



$$\Pi_{\textbf{numvol}} \left(\sigma_{adresse='Nice'} (Pilote) \bowtie \sigma_{villeDep='Nice'} (Vol) \right)$$

De même attention à ne pas faire

$$\Pi_{\textbf{numvol}} \left(\sigma_{adresse='Nice' \ \wedge \ villeDep='Nice'} (Pilote \bowtie Vol) \right)$$

qui effectue d'abord la jointure et ensuite filtre → perte de temps et espace

La jointure externe naturelle R $\overset{\circ}{\bowtie}$ S

Parfois, perdre les tuples défaillants de la jointure n'est pas convenable...

Ci-dessus, les attributs en communs qui conditionnent la jointure sont B et C. On voit que nous avons perdu de R1 entre autres le tuple (6,7,8)

R_1 =

A	B	C
1	2	5
3	4	6
2	5	12
6	7	8

R_2 =

B	C	D	E
2	5	6	3
4	6	8	10
5	7	10	3
4	6	7	8
3	4	6	8

R1⋈R2 :

A	B	C	D	E
1	2	5	6	3
3	4	6	8	10
3	4	6	7	8

La jointure externe naturelle est une opération de jointure naturelle qui permet de retrouver aussi les tuple défaillants, en mettant des valeurs NULL (ou ⊥) pour les attributs qui ne sont pas définis dans les résultats.

Deux variantes : jointure naturelle externe gauche et droite

La jointure externe naturelle gauche $R \overset{\circ}{\bowtie}_L S$

$R_1 =$

A	B	C
1	2	5
3	4	6
2	5	12
6	7	8

$R_2 =$

B	C	D	E
2	5	6	3
4	6	8	10
5	7	10	3
4	6	7	8
3	4	6	8

R1⋈R2 =

A	B	C	D	E
1	2	5	6	3
3	4	6	8	10
3	4	6	7	8

$R_1 \overset{\circ}{\bowtie}_L R_2 =$

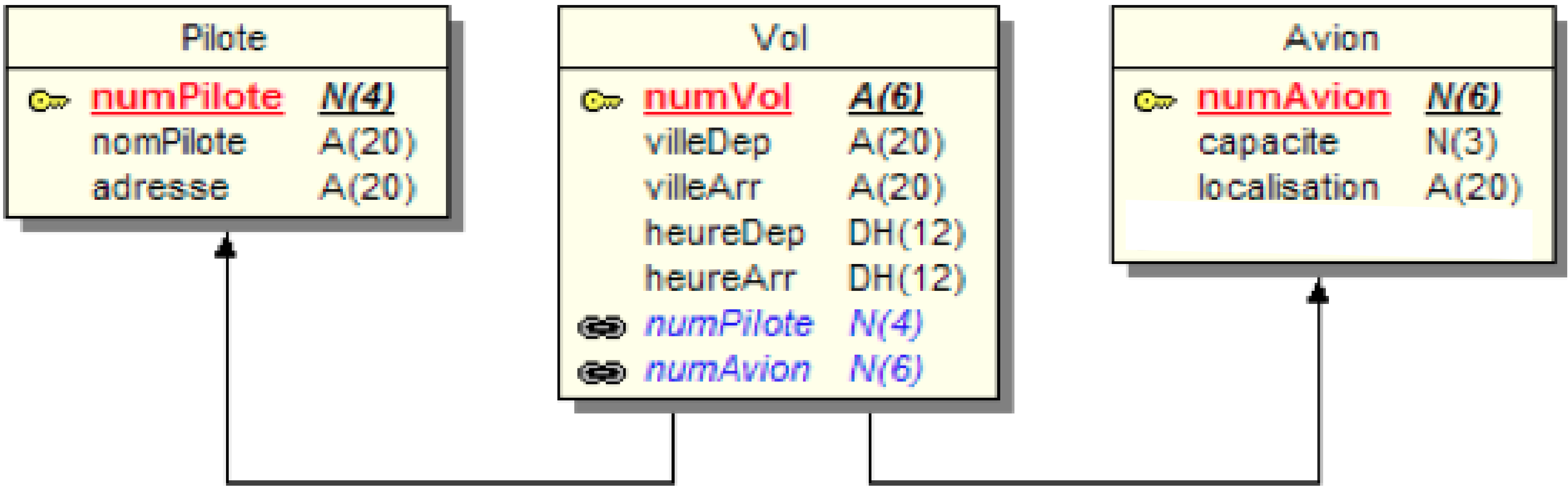
A	B	C	D	E
1	2	5	6	3
3	4	6	8	10
3	4	6	7	8
2	5	12	⊥	⊥
6	7	8	⊥	⊥

Sur l'exemple ci-dessus, les attributs en communs qui conditionnent la jointure sont B et C.

Pour la jointure naturelle externe gauche on remarque le tuple (2,5,12) de R1 ne peut pas se joindre aux tuples de R2 sur BC mais du fait que la jointure soit externe gauche alors on l'ajoute au résultat en mettant à ⊥ toutes les valeurs des attributs provenant de R2 c'est-à-dire D et E ce qui explique la ligne (2,5,12,⊥,⊥) de la figure de droite.

La jointure externe naturelle gauche $R \overset{\circ}{\bowtie}_{L} S$

Donnez pour chaque avion de la table Avion la liste de tous les vols qui ont utilisé cette avion. Si un avion n'a fait aucun vol alors on l'affiche quand même avec NULL comme numéro de vol.



Attention : ce sont des couples (numAvion,capacité, numVol) que l'on souhaite afficher :

(1, 300, AF01)
(1, 300, AF03)
(2, 150, NULL)
(3, 300, AF04)
(3, 300, AF05)

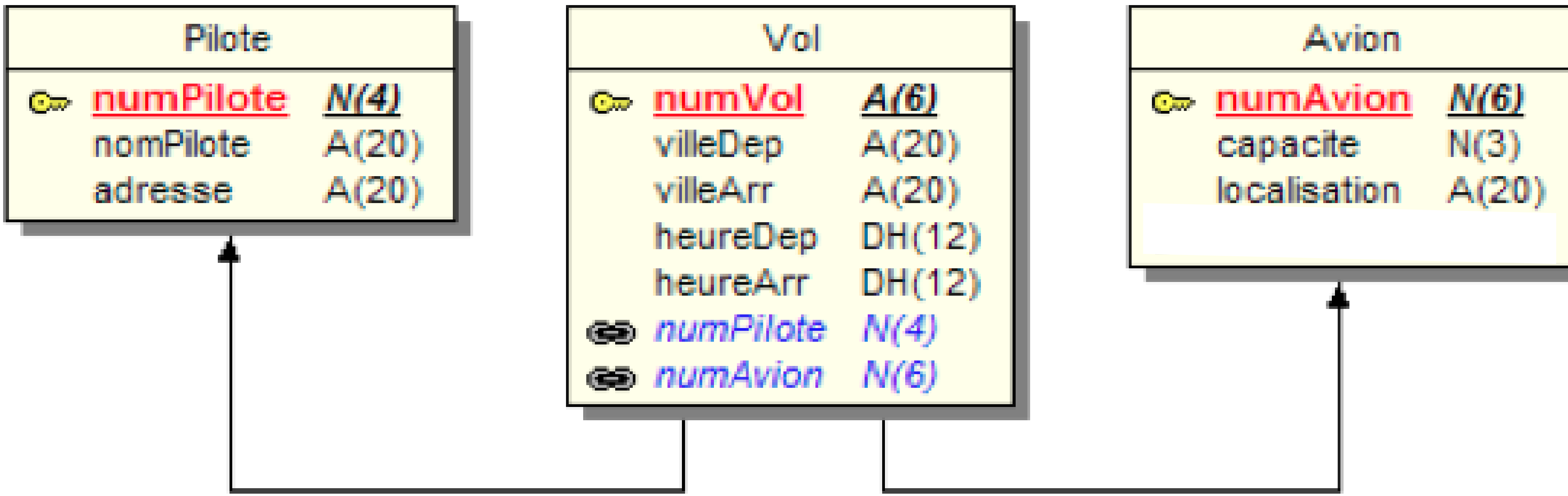
L'avion 1 de capacité 300 a fait les vols AF01 et AF03

L'avion 2 de capacité 150 n'a fait aucun vol

L'avion 3 de capacité 300 a fait les vols AF04 et AF05

La jointure externe naturelle gauche $R \overset{\circ}{\bowtie}_{L} S$

Donnez pour chaque avion de la table Avion la liste de tous les vols qui ont utilisé cette avion. Si un avion n'a fait aucun vol alors on l'affiche quand même avec NULL comme numéro de vol.



$$\Pi_{\text{numAvion, capacité, numvol}} \left(\text{avion} \overset{\circ}{\bowtie}_{L} \text{Vol} \right)$$

Si on fait une jointure naturelle on aura pas les avions qui ont fait aucun vol.

La jointure externe naturelle $R \overset{\circ}{\bowtie} S$

Sur l'exemple ci-dessus, les attributs en communs qui conditionnent la jointure sont B et C.

Pour la jointure naturelle externe on garde tous les éléments de R1 mais aussi tous ceux de R2 et on ajoute selon le cas des $\perp$ soit sur (D,E) de R2 soit sur (A) de R1.

$R_1 =$

A	B	C
1	2	5
3	4	6
2	5	12
6	7	8

$R_2 =$

B	C	D	E
2	5	6	3
4	6	8	10
5	7	10	3
4	6	7	8
3	4	6	8

$R1 \bowtie R2 =$

A	B	C	D	E
1	2	5	6	3
3	4	6	8	10
3	4	6	7	8

$R_1 \overset{\circ}{\bowtie} R_2 =$

A	B	C	D	E
1	2	5	6	3
3	4	6	8	10
3	4	6	7	8
2	5	12	$\perp$	$\perp$
6	7	8	$\perp$	$\perp$
$\perp$	5	7	10	3
$\perp$	3	4	6	8

On peut également faire la même chose dans un contexte de théta-jointure. On parle alors de théta-jointure externe et ces variantes gauche et droite.

La division R / S

La division permet de rechercher l'ensemble de tous les sous-n-uplets d'une relation satisfaisant une sous-relation décrite par la relation diviseur.

On l'utilise pour répondre à des questions de la forme « trouver y, quel que soit x » ou « donner les n-uplets vérifiant tous x ».

La division (ou le quotient) de la relation R de schéma R(U,V) par la relation S de schéma S(V) est une relation T ,notée R/T de schéma T(U) formée de tous les (sous-) n-uplets de R qui, concaténés à chaque n-uplet de S, donnent toujours un n-uplet de R.

On a : $u \in R / S \Leftrightarrow \forall v \in S \; ((u,v) \in R)$

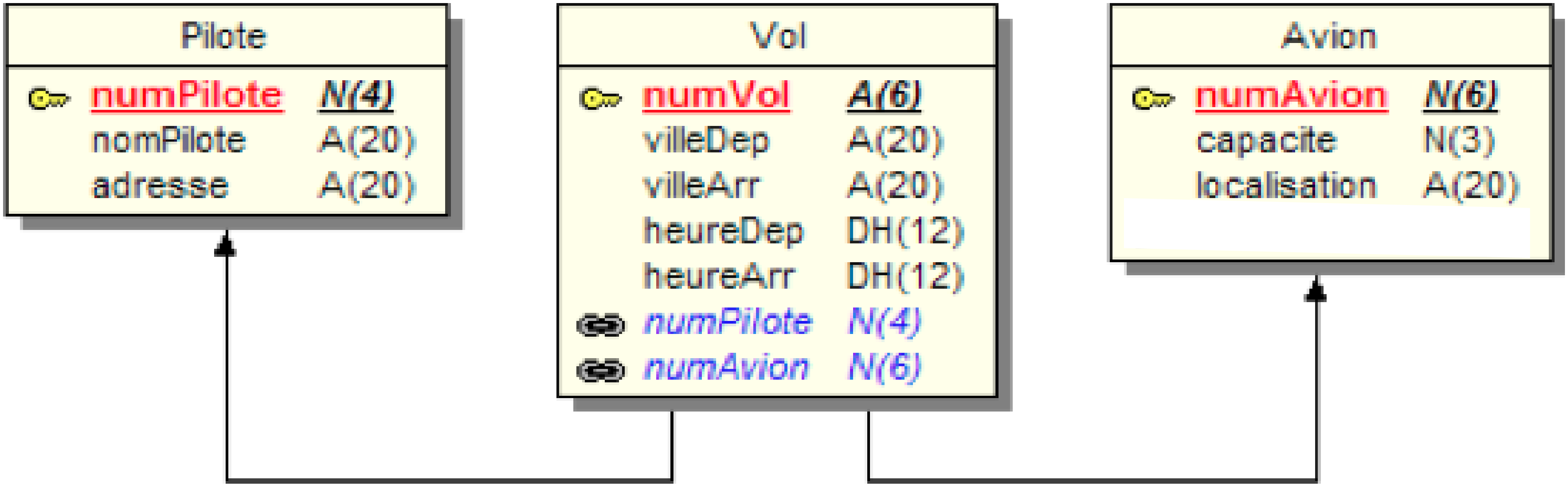
r			
A	**B**	**C**	**D**
a1	b1	c1	d1
a1	b1	c2	d2
a2	b2	c2	d2
a3	b3	c1	d1
a3	b3	c2	d2
a1	b1	c3	d3

s	
C	**D**
c1	d1
c2	d2

r ÷ s	
A	**B**
a1	b1
a3	b3

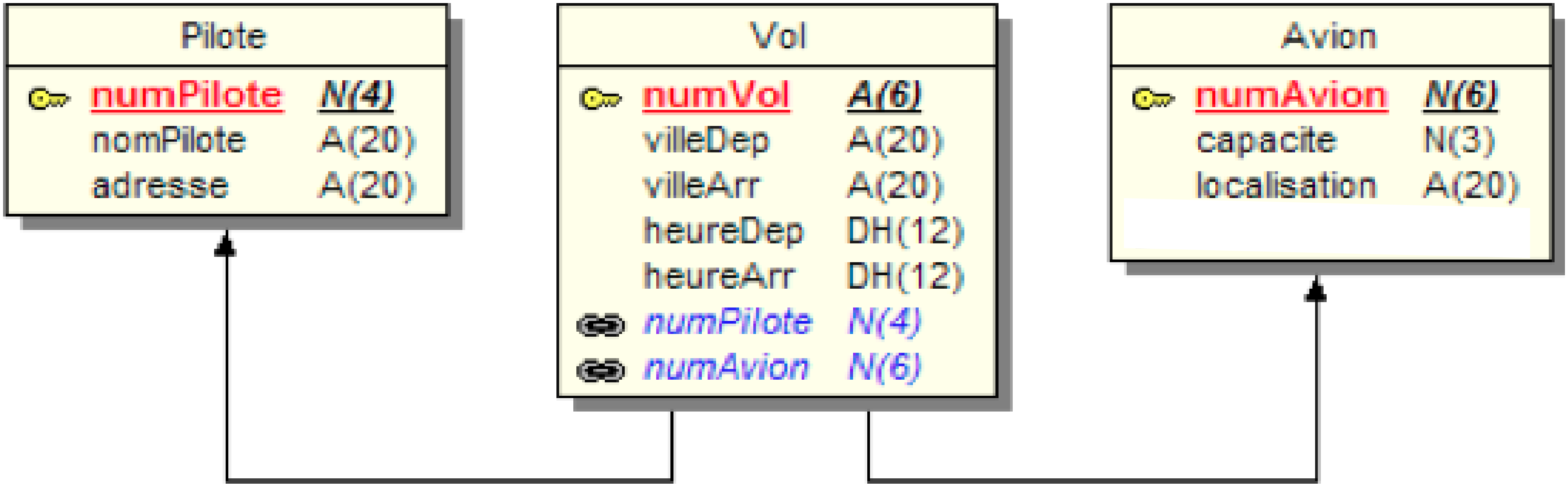
La division R/S

Exemple : quels sont les numéros d'avions qui ont été utilisés par tous les pilotes qui ont fait au moins un vol.



La division R/S

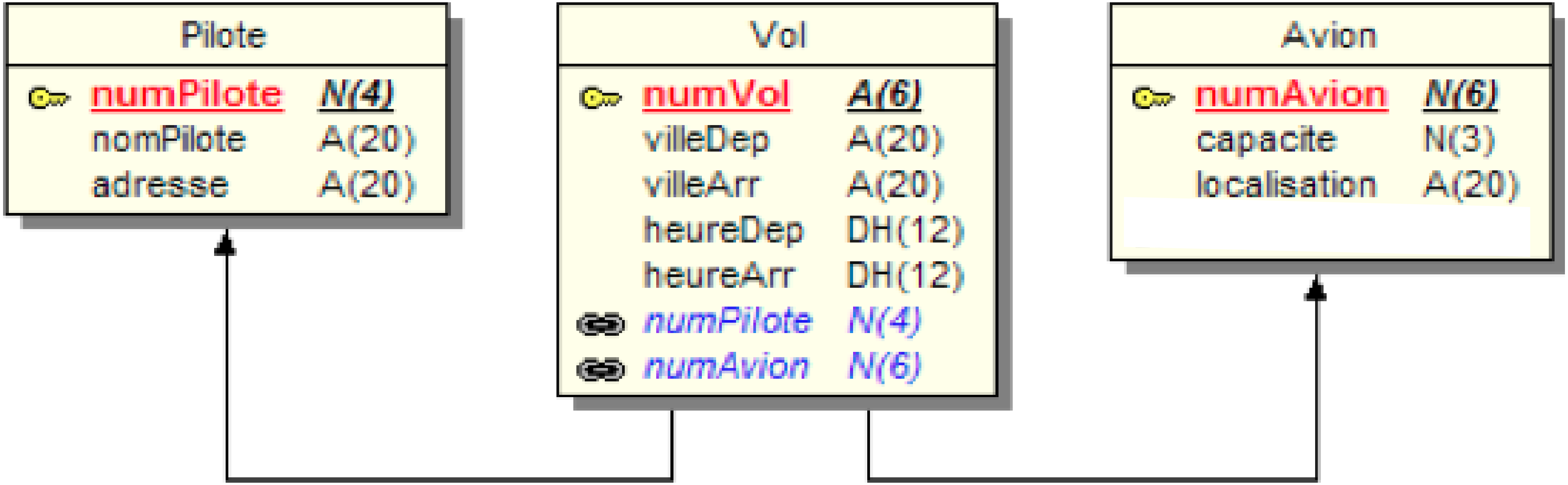
Exemple : quels sont les numéros d'avions qui ont été utilisés par tous les pilotes qui ont fait au moins un vol.



$$\Pi_{\text{numPilote,numAvion}} (\text{Vol}) \,/\, \Pi_{\text{numPilote}} (\text{Vol})$$

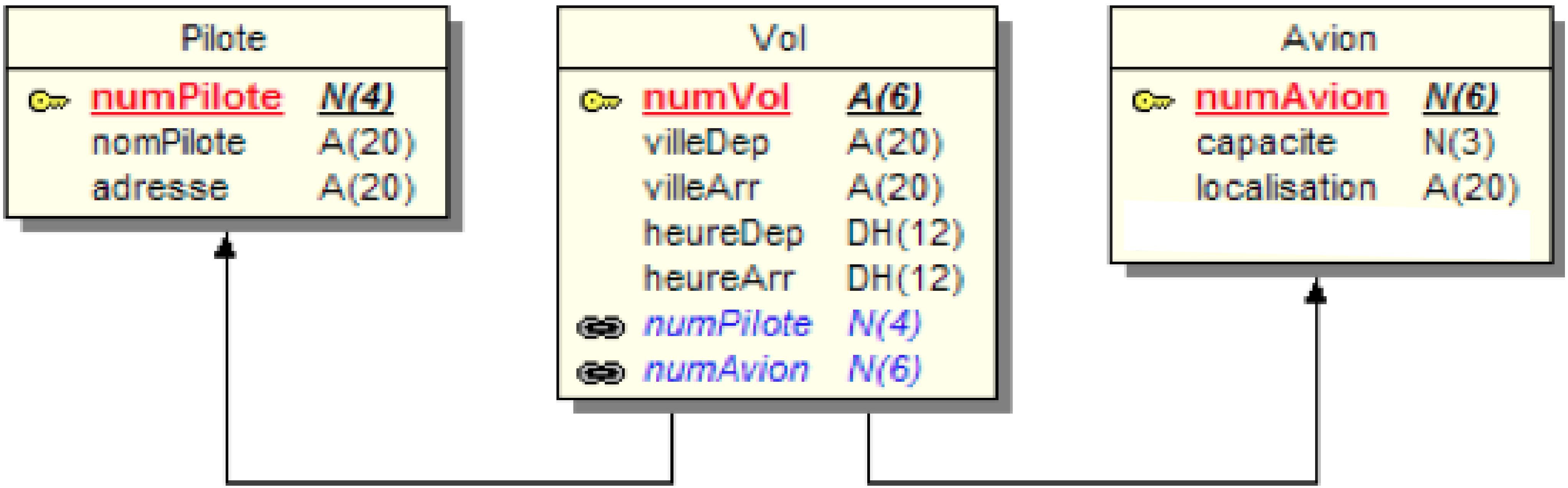
La division R/S

Exemple : quels sont les numéros de pilotes dont on a prévu des vols avec tous les avions localisés à nice.



La division R/S

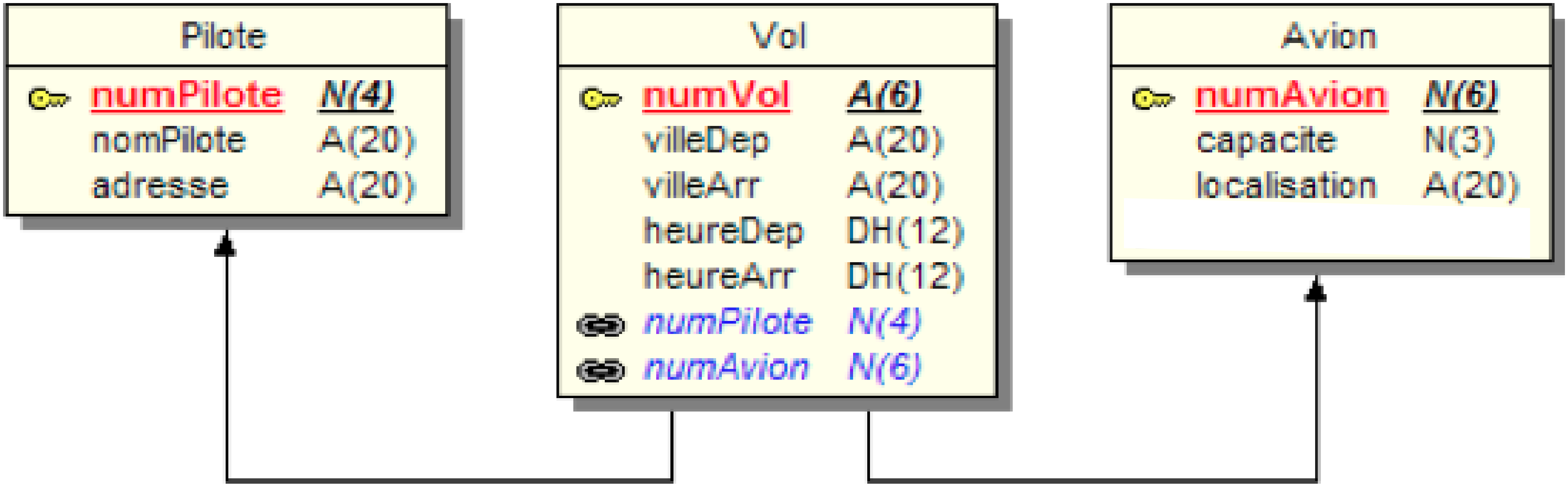
Exemple : quels sont les numéros de pilotes dont on a prévu des vols avec tous les avions localisés à nice.



$$\Pi_{\text{numPilote,numAvion}}(\text{Vol}) \;/\; \Pi_{\text{numAvion}}(\sigma_{\text{localisation='nice'}}(\text{Avion}))$$

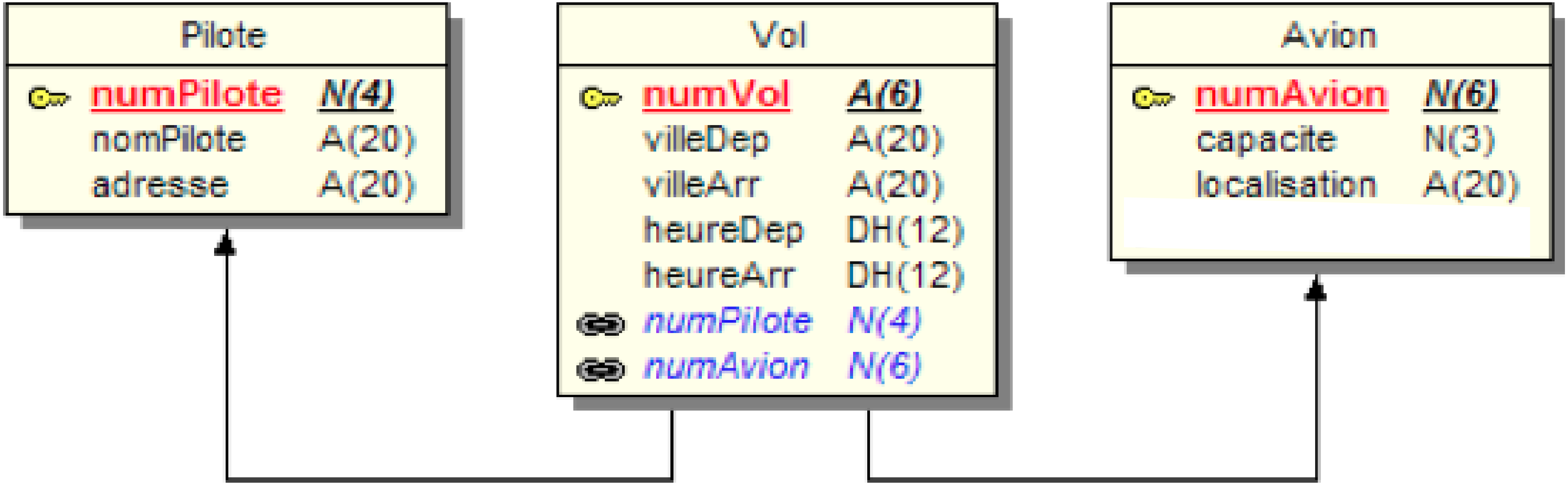
La division R/S

Exemple : quels sont les numéros d'avions qui ont fait des vols avec tous les pilotes qui habitent Nice.



La division R/S

Exemple : quels sont les numéros d'avions qui ont fait des vols avec tous les pilotes qui habitent Nice.



$$\Pi_{\text{numPilote,numAvion}} (\text{Vol}) \ / \ \Pi_{\text{numPilote}} (\sigma_{\text{adresse='nice'}} (\text{Pilote}))$$

L'AR étendue : tri, projection étendue

Les opérateurs classiques de l'algèbre relationnelle décrits jusqu'ici sont le fondement des moteurs de requêtes modernes.

ORACLE SQL propose d'autres opérations sur les tables qui correspondent aux opérateurs étendus ci-après.

L'opérateurs de tri $\tau_L(R)$ pour une liste d'attributs L de R : trie les tuples de R selon les valeurs des attributs dans L, considérés dans l'ordre de leur apparition dans L.

Projection étendue $\Pi_L(R)$: certains attributs peuvent changer de nom, d'autres nouveaux peuvent être construits. Les éléments de la liste L des attributs résultat peuvent contenir :

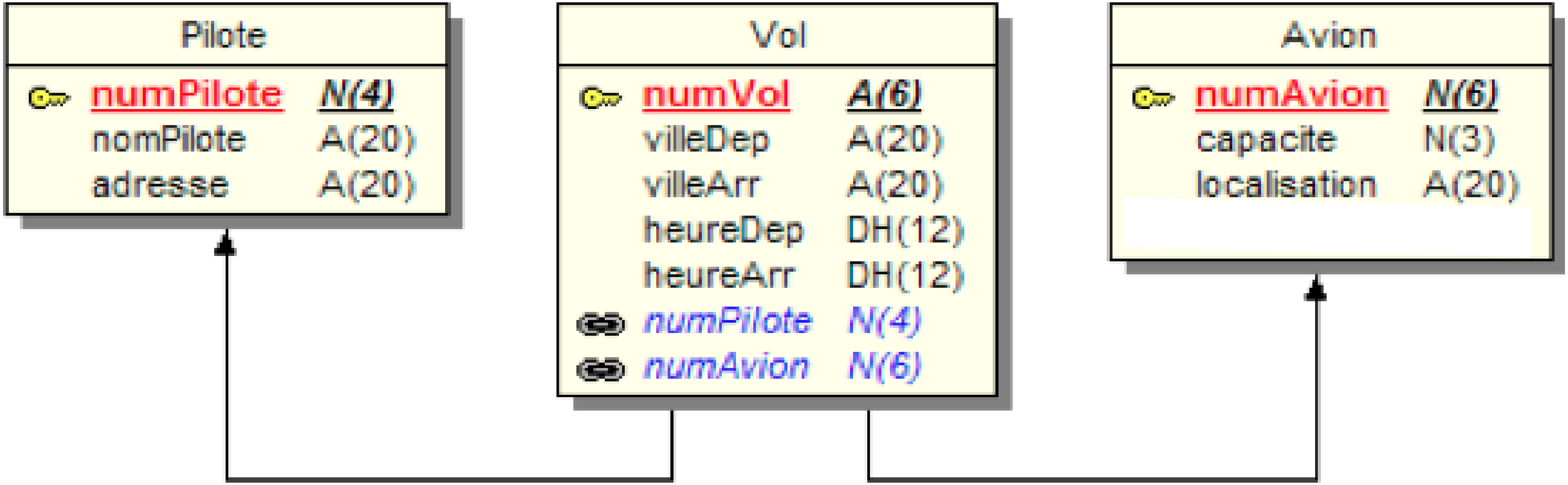
un attribut A

un renommage $A \rightarrow B$

une expression $E \rightarrow B$ où E combine des attributs et des opérateurs sur ces attributs.

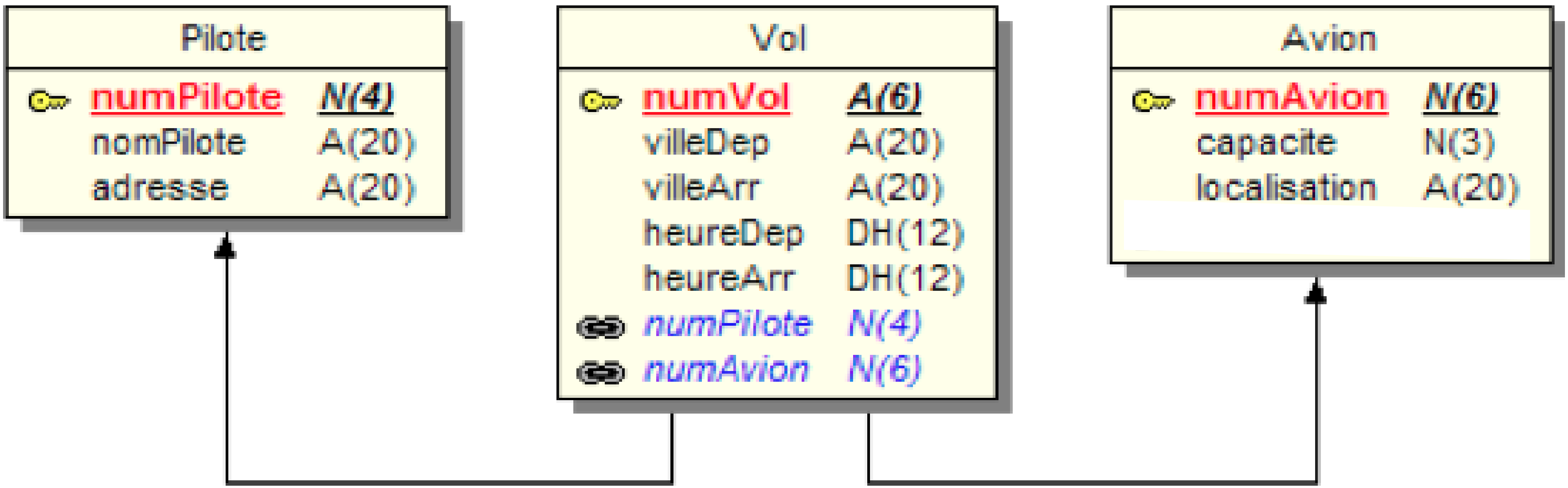
L'AR étendue : tri, projection étendue

Exemple : donnez pour chaque avion de la table Avion le nombre de sièges de première classe sachant que cela représente 10 % de la capacité de l'avion pour cette compagnie.



L'AR étendue : tri, projection étendue

Exemple : donnez pour chaque avion de la table Avion le nombre de sièges de première classe sachant que cela représente 10 % de la capacité de l'avion pour cette compagnie. Cette colonne se nommera « premiereClasse ».



$$\Pi_{\textbf{numAvion, capacité * 0,1} \rightarrow \textbf{premiereClasse}} \text{(Avion)}$$

L'AR étendue : groupement et agrégation

Opération de groupement $\gamma_L(R)$: représente l'ensemble de tuples construits comme suit :

Regrouper d'abord les tuples de R en un ensemble de groupes, chaque groupe ayant une seule combinaison de valeurs pour les attributs dans L

Pour chaque élément de l'ensemble de groupes, produire un seul tuple dans le résultat.

Exemple :

$R =$

A	B	C
1	2	3
1	2	4
1	1	3
2	1	3
2	1	2

$\gamma_{A,B}(R) =$

A	B
1	2
1	1
2	1

Sur l'exemple ci-dessus, les groupes sont faits selon les attributs (A,B) : on a donc trois groupes

(1,2)
(1,1)
(2,1)

Chaque groupe n'est représenté qu'une seule fois dans le résultat final.

On peut également effectuer des calculs en introduisant les opérations agrégées : SUM,AVG (moyenne), MIN, MAX, COUNT,..

Opération de groupement avec calcul $\gamma_{L,Op \rightarrow NvAttr}(R)$: se construit comme suit :

Regrouper d'abord les tuples de R en un ensemble de groupes, chaque groupe ayant une seule combinaison de valeurs pour les attributs dans « L »

A l'intérieur de chaque groupe, appliquer l'opérateur d'agrégation « op » et produire, avec les résultats, une nouvelle colonne NvAttr contenant le résultat.

Enfin, produire un tuple pour chaque combinaison de valeurs présente pour les attributs dans la liste L plus NvAttr.

Remrque : il est possible d'omettre la liste « L » : $\gamma_{Op \rightarrow NvAttr}(R)$ dans ce cas l'opération s'appliquera sans calcul de groupe. Par exemple pour calculer la somme d'une colonne de toute une table ou compter le nombre d'éléments dans une table.

Exemples de groupage et agrégation

$\gamma_{A,SUM(B) \rightarrow SB}(R)$ établie des regroupements sur « A » et somme B à l'intérieur de chacun d'eux.

$R =$

A	B	C
1	2	3
1	2	4
1	1	3
2	1	3
2	1	2

$\gamma_{A,SUM(B) \rightarrow SB}(R) =$

a	SB
1	5
2	2

$\gamma_{SUM(B) \rightarrow total}$ somme toute la colonne B (sans effectuer de groupes).

$\gamma_{SUM(B) \rightarrow total}(R) =$

total
7

$\gamma_{COUNT(A) \rightarrow nbElementA}$ calcule le nombre de lignes dans R pour lesquelles l'attribut « A » a une valeur différente de NULL (sans faire de groupes).

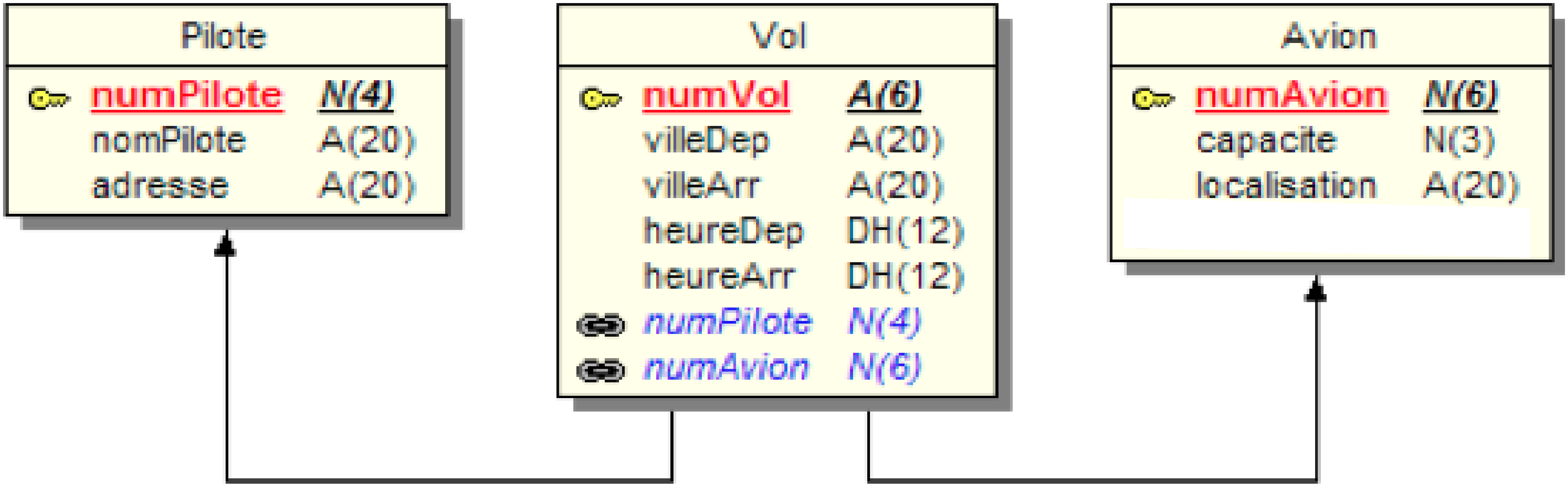
$\gamma_{COUNT(A) \rightarrow nbElementA}(R) =$

nbElementA
5

COUNT(*) : compte le nombre de ligne même s'il y a des valeurs NULL
COUNT(DISTINCT A) : similaire à COUNT(A) mais éjecte les doublons des paquets formés

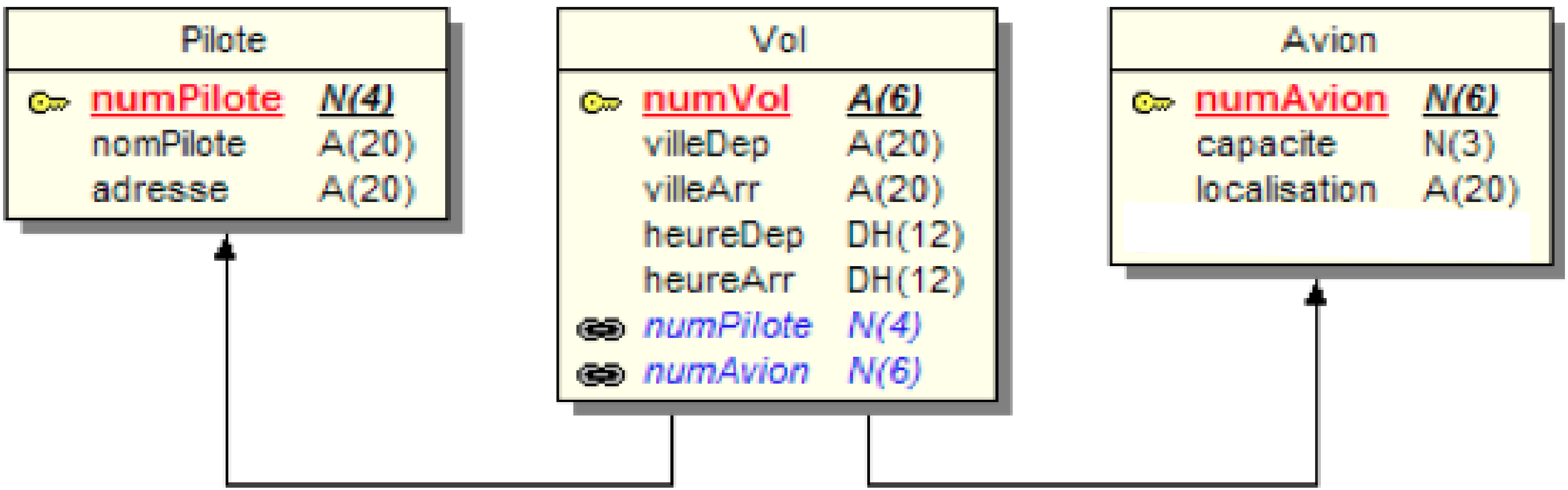
Exemples de groupage et agrégation

Pour chaque numPilote, donnez le nombre de vols de chaque pilote qui a effectué au moins un vol. Cette nouvelle colonne sera nommé nbVol



Exemples de groupage et agrégation

Pour chaque numPilote, donnez le nombre de vols de chaque pilote qui a effectué au moins un vol. Cette nouvelle colonne sera nommé nbVol

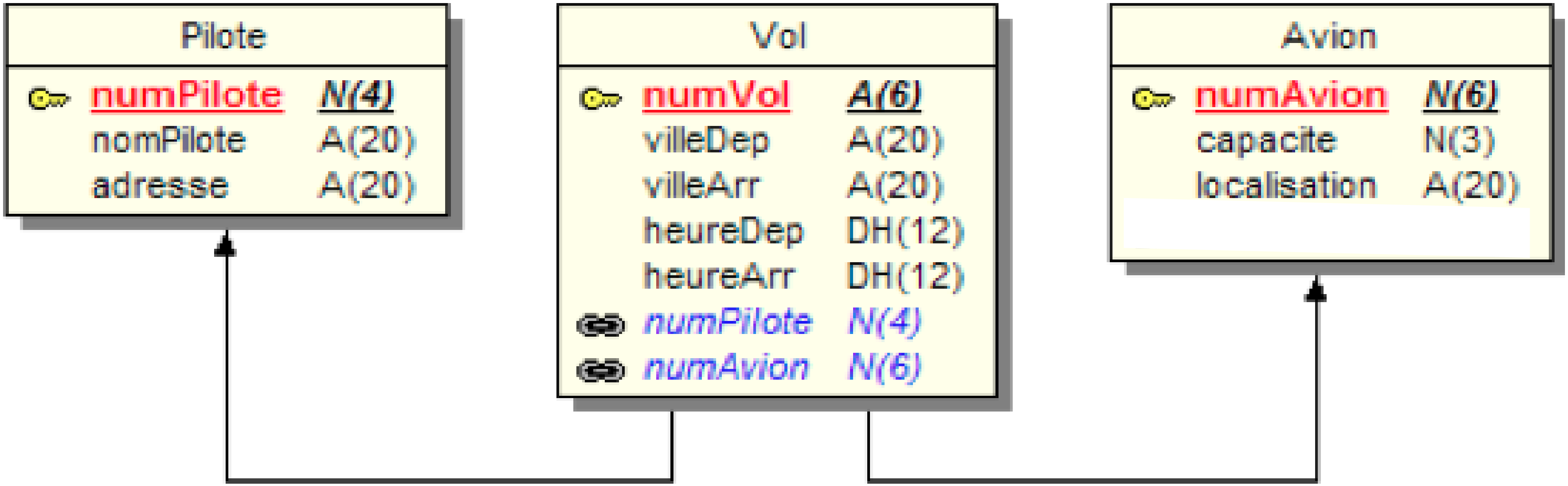


$\gamma_{\text{numPilote},\text{Count(numVol)} \rightarrow \text{nbVol}}(\text{Vol}) \longrightarrow$ On obtient la relation (numPilote,nbVol)

On fait des groupes de numPilote et on compte le nombre de numVol dans chaque groupe

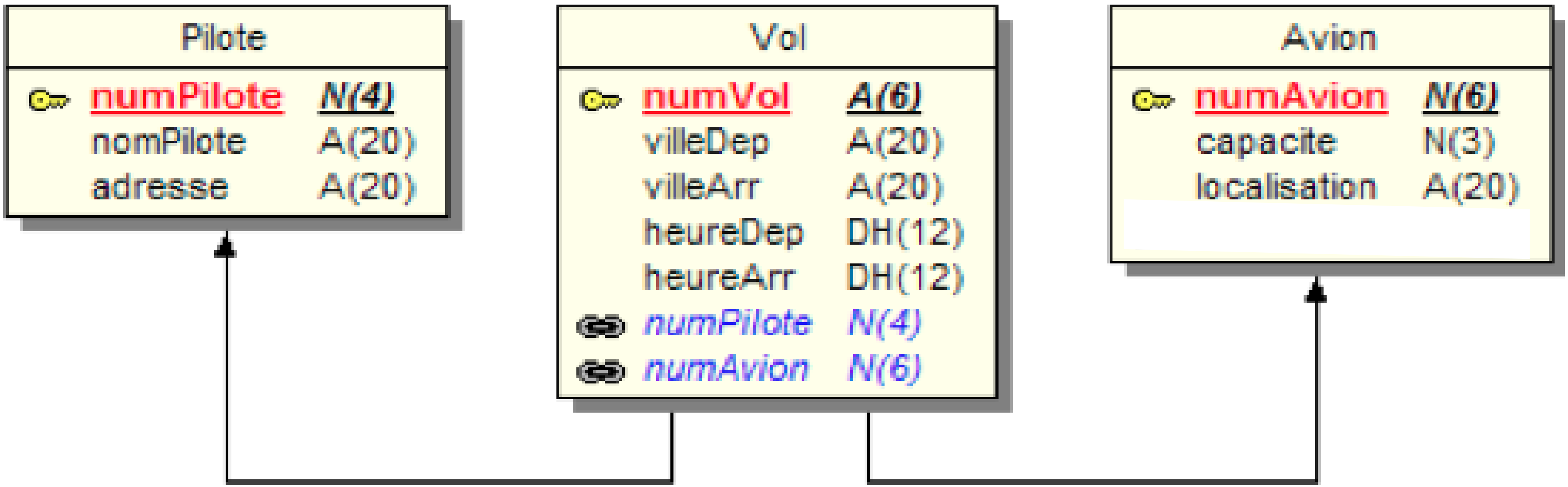
Exemples de groupage et agrégation

Afficher les numéros de pilotes qui ont pris le même avion deux fois



Exemples de groupage et agrégation

Afficher les numéros de pilotes qui ont pris le même avion deux fois



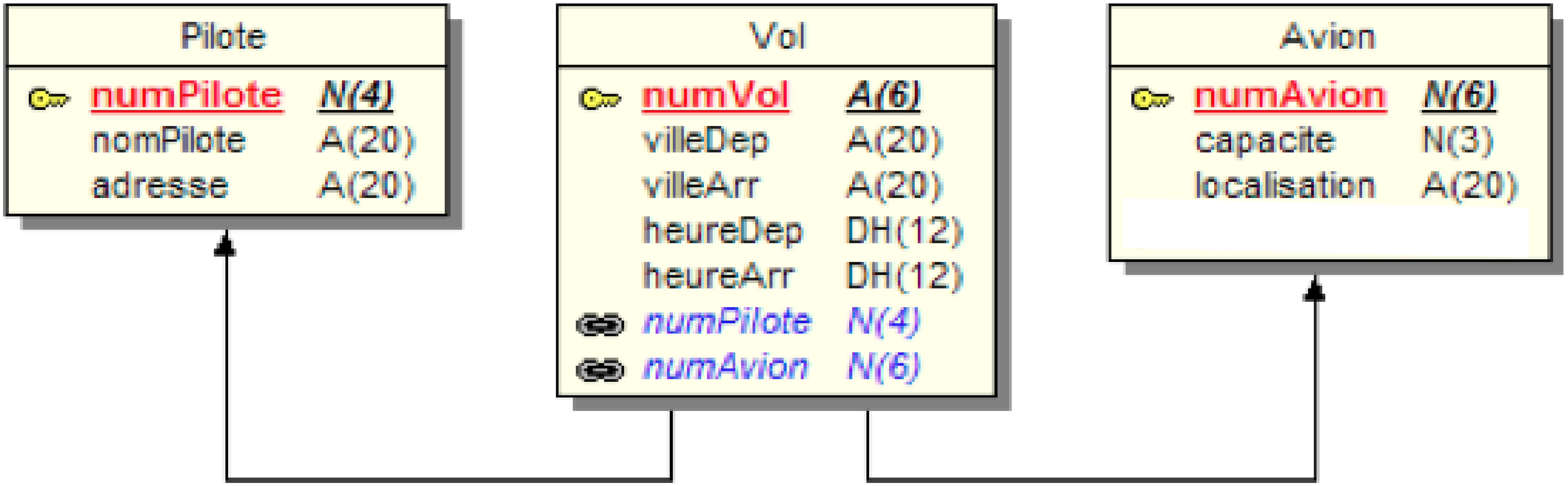
$$\Pi_{\text{numPilote}} \left(\sigma_{\text{nbVol}=2} \left(\gamma_{(\text{numPilote},\text{numAvion}),\text{Count}(\text{numVol}) \rightarrow \text{nbVol}} (\text{Vol}) \right) \right)$$

la relation (numPilote,numAvion,nbVol)

On fait des groupes de couples (numPilote,numAvion) et on compte le nombre de numVol dans chaque groupe

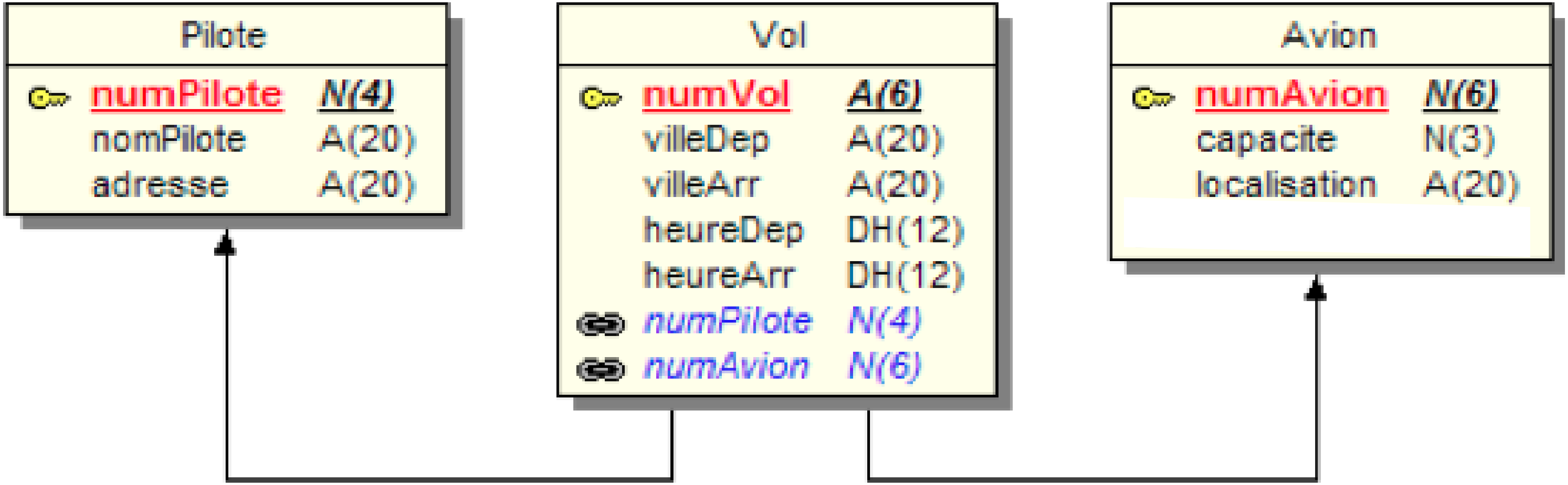
Exemples de groupage et agrégation

Exemple : donnez pour chaque couple (villeDep,villeArrivée) le nombre de vols qui lui est associé. Cette nouvelle colonne sera nommée nbVol. Puis ne gardez que ceux pour lesquels on a plus de 2 vols.



Exemples de groupage et agrégation

Exemple : donnez pour chaque couple (villeDep,villeArrivée) le nombre de vols qui lui est associé. Cette nouvelle colonne sera nommée nbVol. Puis ne gardez que ceux pour lesquels on a plus de 2 vols.



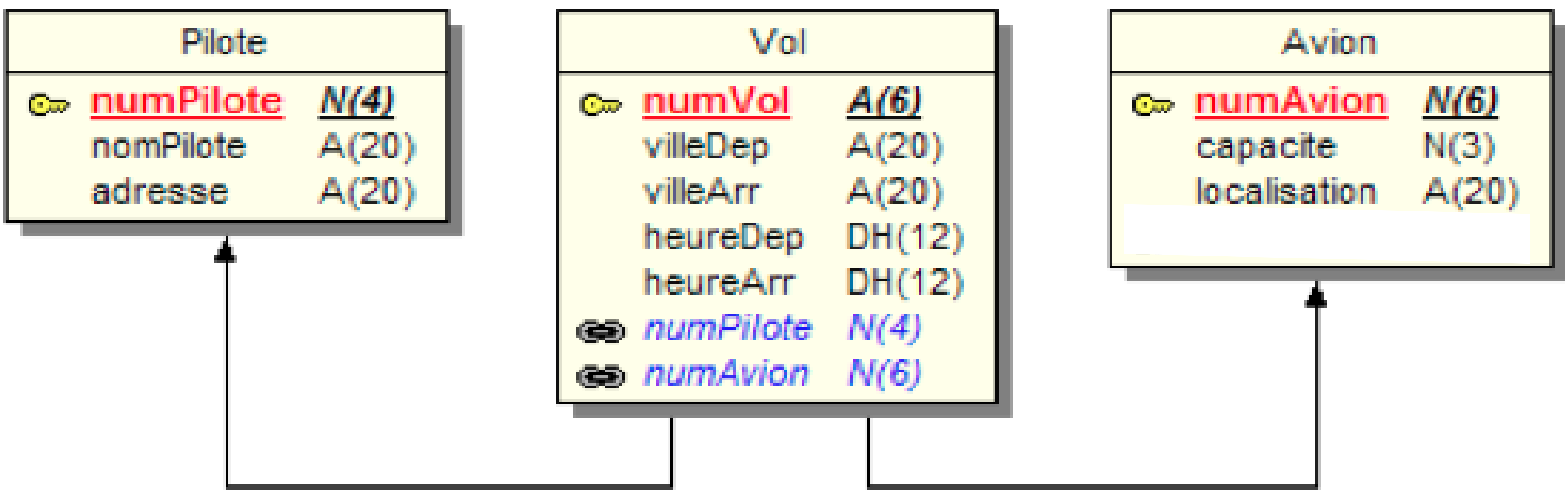
$$\sigma_{nbVol>2}\left(\gamma_{(heureDep,heureArr),Count(numVol)\rightarrow nbVol}(Vol)\right)$$

L'AR pour la formulation de contraintes

Une contrainte est une expression qui doit s’évaluer à l’ensemble vide avant et après toute opération : $R = \emptyset$.

Exemple 1 : exprimez le fait que numPilote de la table Vol est une clé étrangère qui fait référence au numPilote de la table Pilote.

$\Pi_?(\ ?\ )\ \setminus\ \Pi_?(\ ?\ ) = \emptyset$

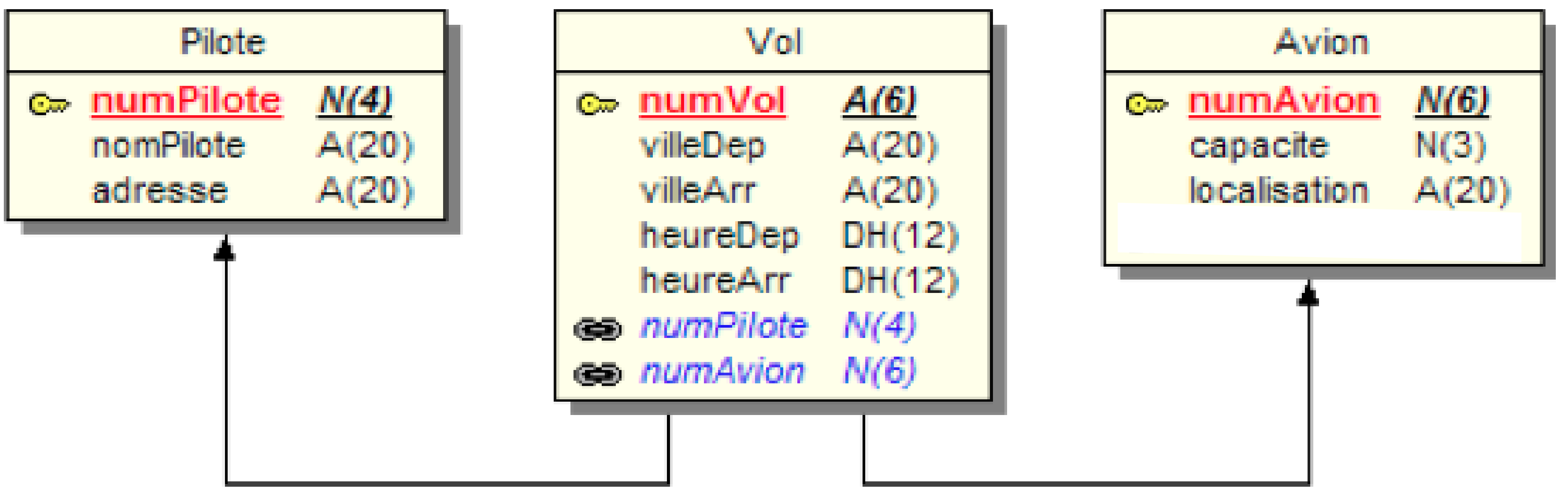


L'AR pour la formulation de contraintes

Exemple 1 : exprimez le fait que numPilote de la table Vol est une clé étrangère qui fait référence au numPilote de la table Pilote.

$\Pi_{numPilote}(Vol) \setminus \Pi_{numPilote}(Pilote) = \varnothing$

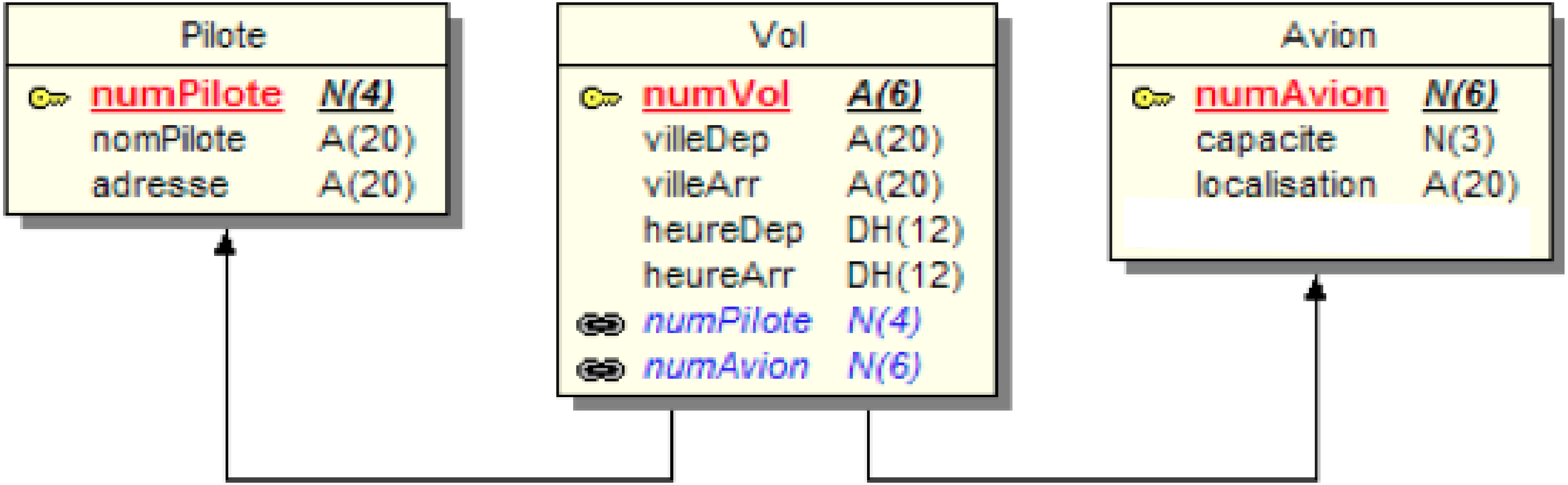
Si le résultat est différent de l'ensemble vide alors il existe un numPilote qui figure dans la table Vol mais pas dans la table Pilote ce qui contredit la contrainte de clé étrangère.



L'AR pour la formulation de contraintes

Exemple 2 : exprimez le fait que numAvion est une clé primaire dans Avion.

$$\sigma_{?} \; (? \times ?) = \varnothing$$

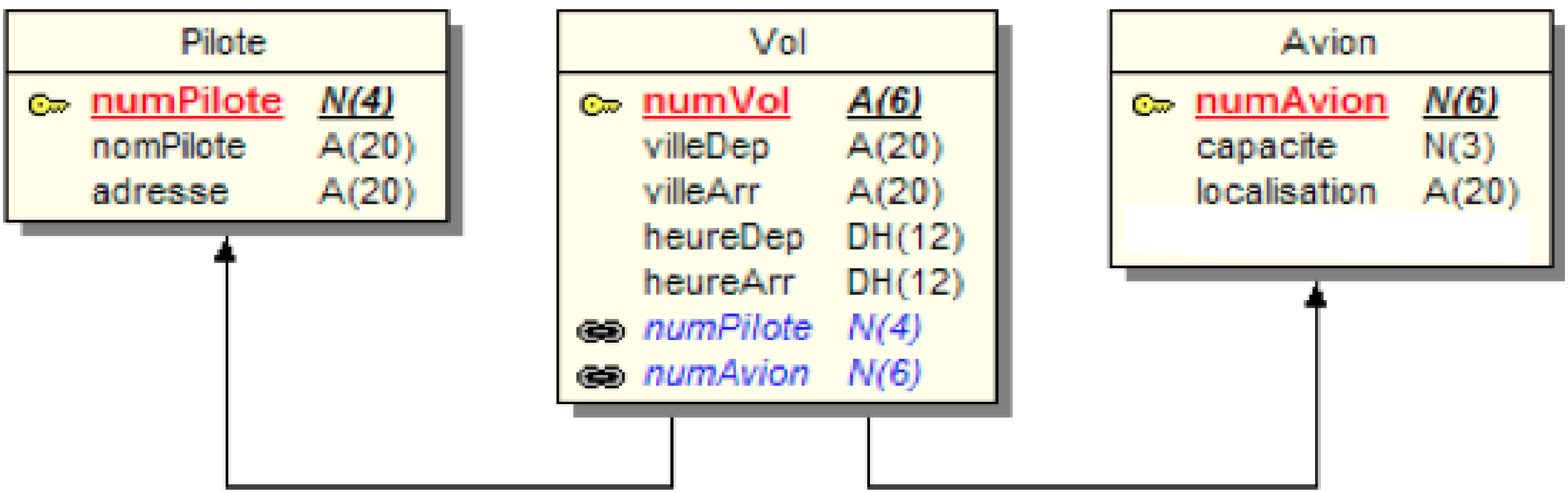


L'AR pour la formulation de contraintes

Exemple 2 : exprimez le fait que numAvion est une clé primaire dans Avion.

$$\sigma_{\text{Avion1.numAvion} = \text{Avion2.numAvion} \,\wedge\, \text{Avion1.capacité} \neq \text{Avion1.capacité}} (\rho_{\text{Avion1}}(\text{Avion}) \times \rho_{\text{Avion2}}(\text{Avion})) = \emptyset$$

Pourquoi ? Décomposons la requêtes



L'AR pour la formulation de contraintes

$$\sigma_{\text{Avion1.numAvion} = \text{Avion2.numAvion} \wedge \text{Avion1.capacité} \neq \text{Avion1.capacité}}$$

$$(\rho_{\text{Avion1}}(\text{Avion}) \times \rho_{\text{Avion2}}(\text{Avion})) = \varnothing$$

numAvion	capacité
AV1	350
AV2	250

Etape 1

$$\rho_{\text{Avion1}}(\text{Avion}) \times \rho_{\text{Avion2}}(\text{Avion})$$

Avion1		Avion2	
AV1	350	AV1	350
AV1	350	AV2	250
AV2	250	AV1	350
AV2	250	AV2	250

L'AR pour la formulation de contraintes

$$\sigma_{Avion1.numAvion = Avion2.numAvion \wedge Avion1.capacité \neq Avion1.capacité} (\rho_{Avion1}(Avion) \times \rho_{Avion2}(Avion)) = \emptyset$$

numAvion	capacité
AV1	350
AV2	250

Etape 2

$$\rho_{Avion1}(Avion) \times \rho_{Avion2}(Avion)$$

Avion1.numAvion=Avion2.numAvion

AV1	350	AV1	350
~~AV1~~	~~350~~	~~AV2~~	~~250~~
~~AV2~~	~~250~~	~~AV1~~	~~350~~
AV2	250	AV2	250

L'AR pour la formulation de contraintes

$$\sigma_{\text{Avion1.numAvion} = \text{Avion2.numAvion} \wedge \text{Avion1.capacité} \neq \text{Avion1.capacité}}$$

$$(\rho_{\text{Avion1}}(\text{Avion}) \times \rho_{\text{Avion2}}(\text{Avion})) = \emptyset$$

numAvion	capacité
AV1	350
AV2	250

Etape 3

~~AV1~~	~~350~~	~~AV1~~	~~350~~
~~AV1~~	~~350~~	~~AV2~~	~~250~~
~~AV2~~	~~250~~	~~AV1~~	~~350~~
~~AV2~~	~~250~~	~~AV2~~	~~250~~