|  |  |
| --- | --- |
|  | **Bases de données – Correction TD1**  **Modélisation, Optimisation de modèles, LDD** |

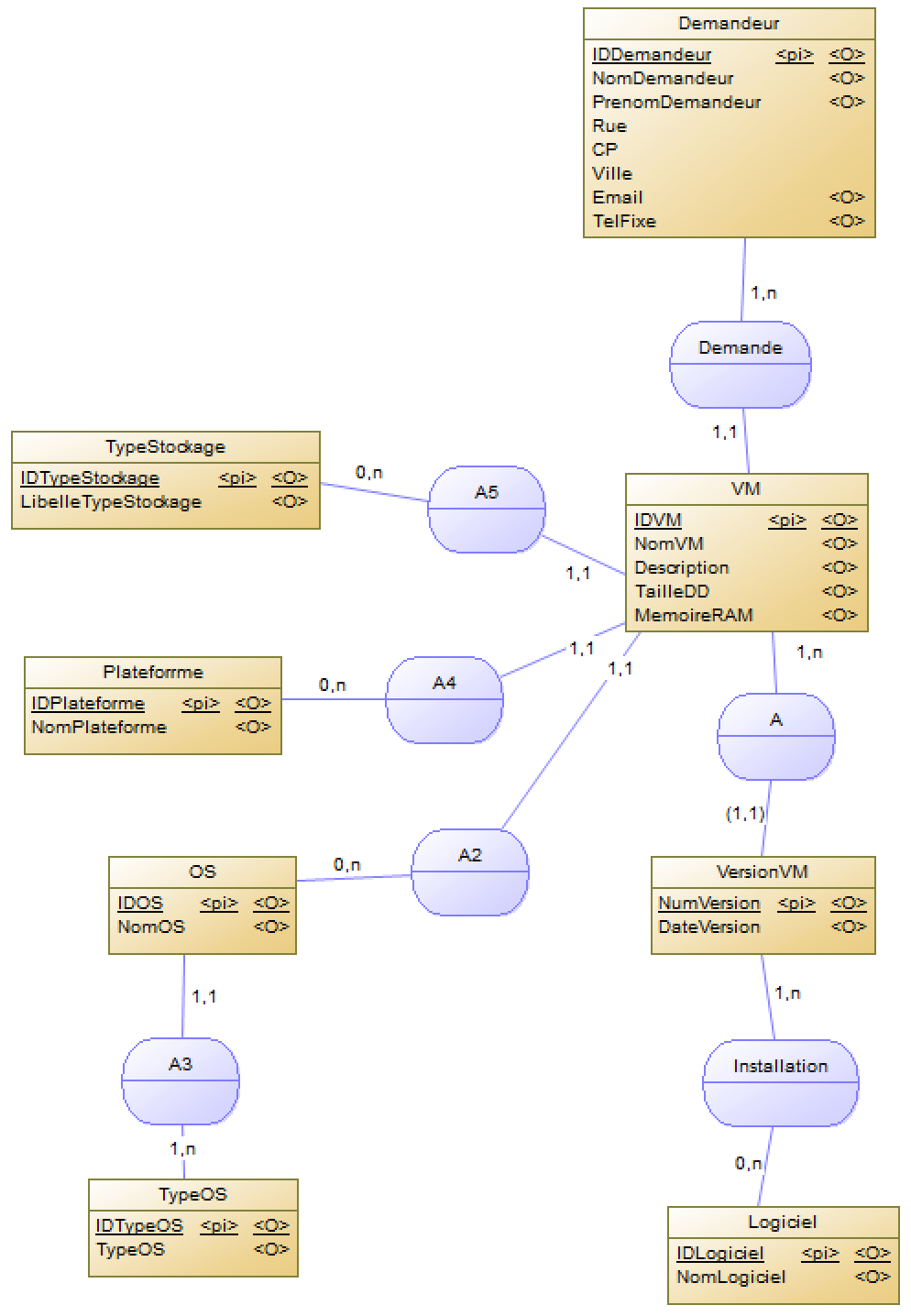
**Question 1 :**

**Liste des propriétés et entités :**

|  |  |  |
| --- | --- | --- |
| Terme | Type | Commentaire |
| Machine virtuelle | Entité | = VM |
| Version VM | Entité |  |
| Plateforme virtualisation | Entité | = Logiciel de virtualisation |
| OS | Entité | = Système d’exploitation |
| Nom VM | Propriété |  |
| Description VM | Propriété |  |
| Demandeur VM | Entité |  |
| Taille DD VM | Propriété |  |
| Type de stockage VM | Entité |  |
| Taille mémoire RAM VM | Propriété |  |
| N° VM | Propriété |  |
| Nom demandeur | Propriété |  |
| Type OS | Entité |  |
| Prénom demandeur | Propriété |  |
| Adresse demandeur | Propriété | A décomposer |
| Mail demandeur | Propriété |  |
| Tel. Fixe demandeur | Propriété |  |
| Date version VM | Propriété |  |
| Logiciel | Entité |  |
| N° version VM | Propriété |  |

**Dans un MCD, on sort tout !**

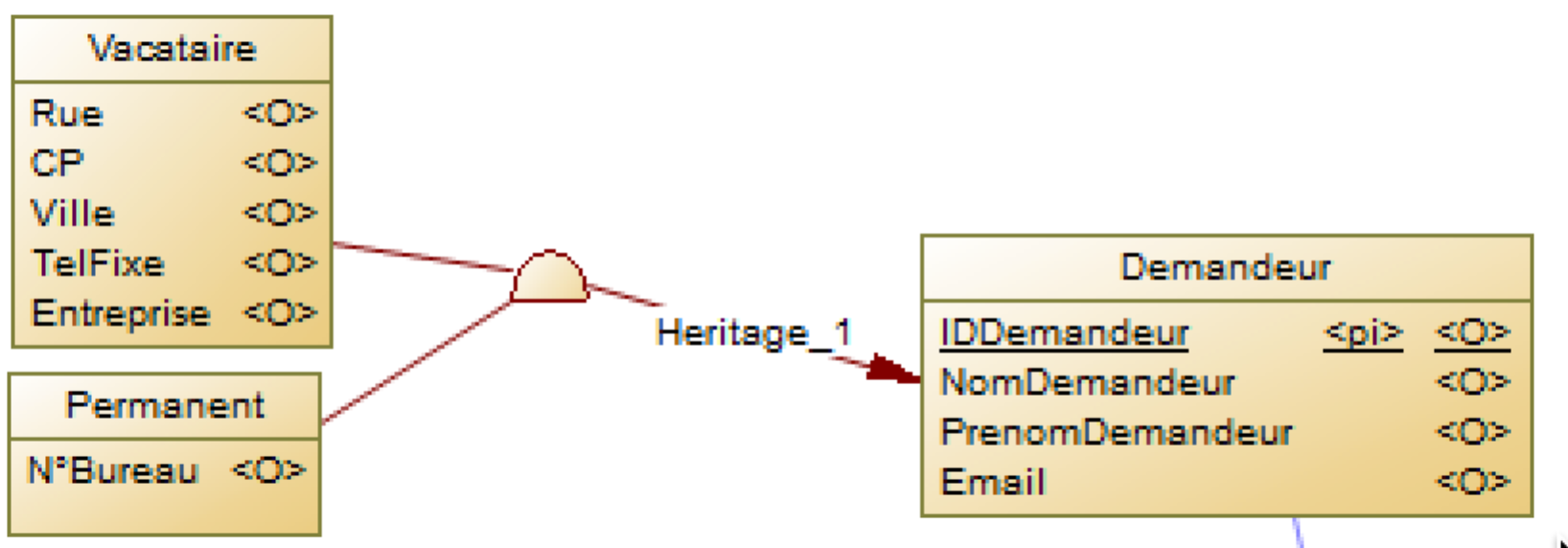
**MCD :**

****

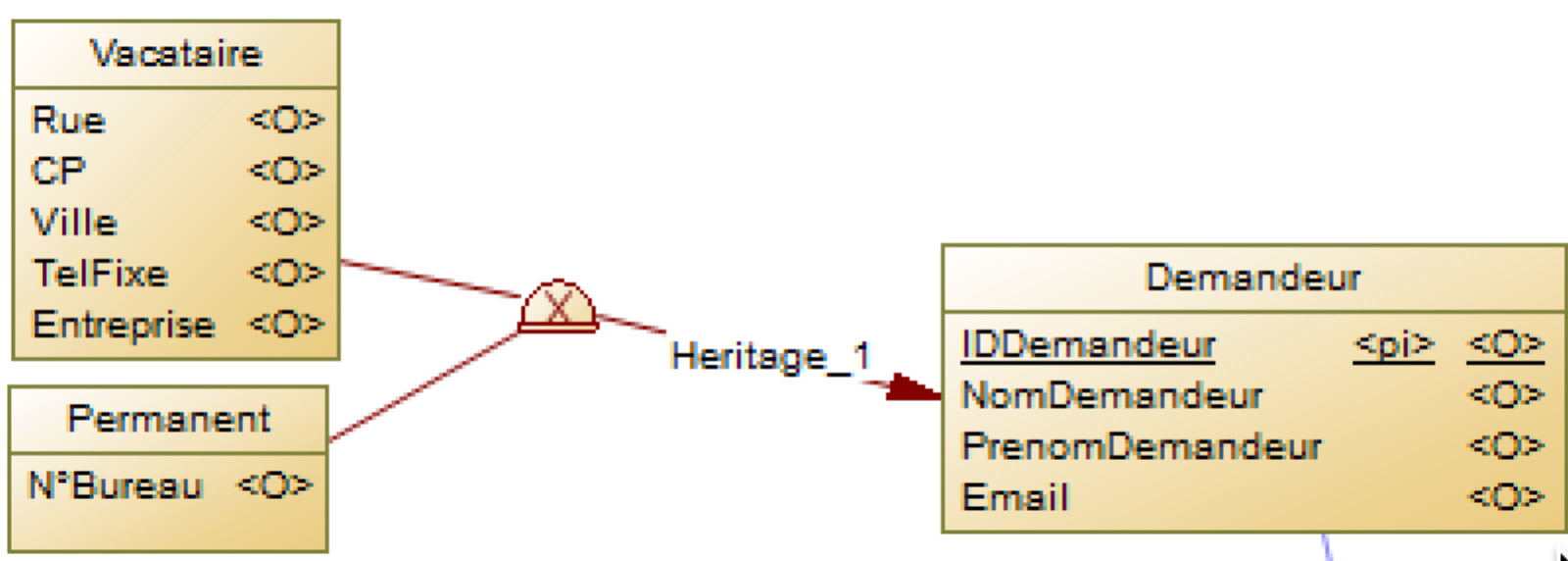
(1,1) : identifiant relatif. L’entité « VersionVM » est dépendante de l’entité « VM ». Le n° de version est relatif au numéro de VM.

Il faut toujours décomposer une adresse en rue, CP, ville, voire pays car une propriété doit être atomique (non décomposable)

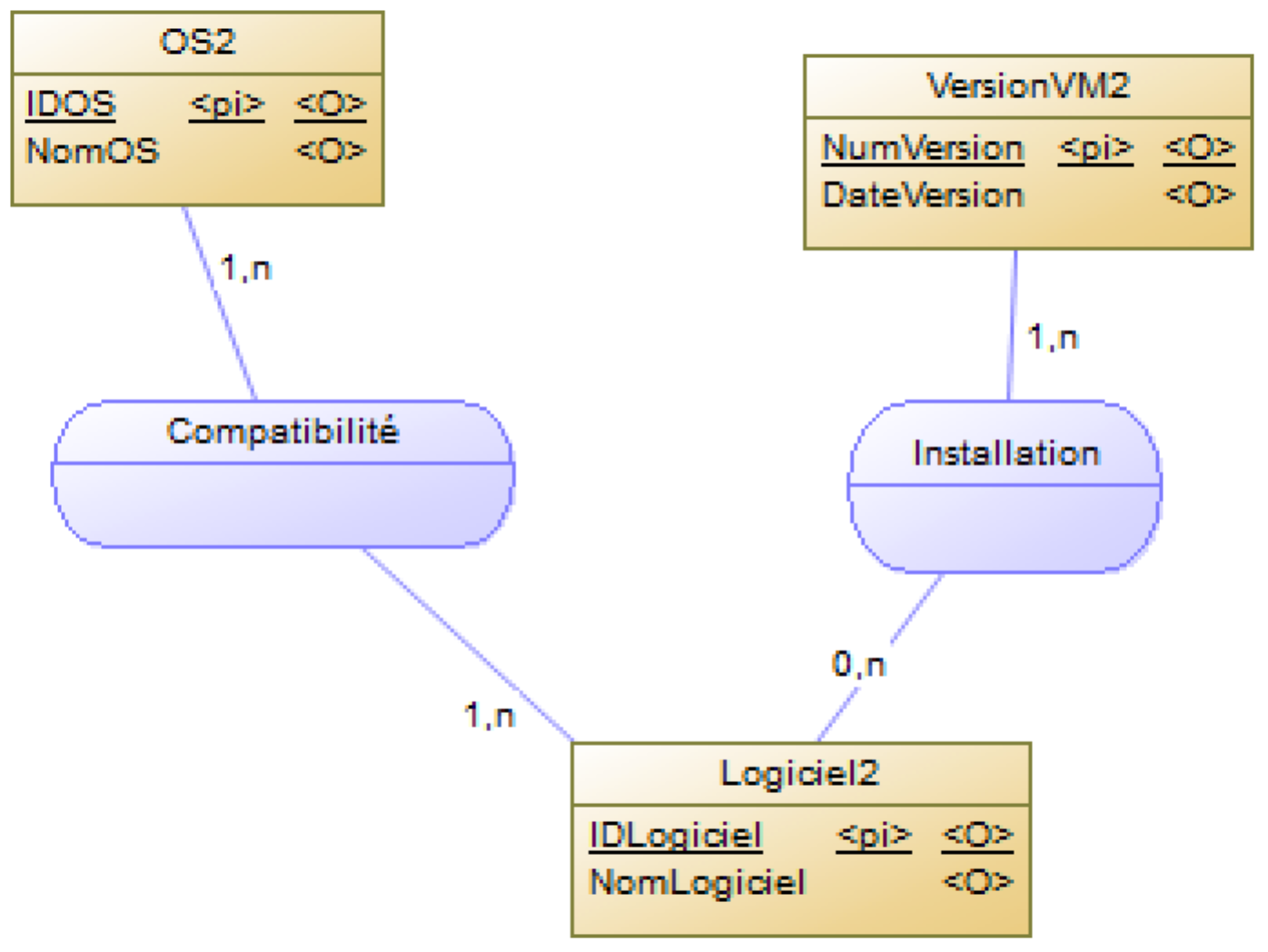
* **La gestion des demandeurs est plus complexe qu’il n’y paraît. En effet, il y a 2 types de demandeurs : vacataire et permanent. L’adresse et le téléphone fixe ne sont saisis que pour les vacataires. On souhaite également enregistrer le nom de leur entreprise. Pour les permanents, on souhaite enregistrer le numéro de bureau. Proposer une modification au modèle.**

****

OU MIEUX car il s’agit d’un héritage par partition (exclusion – totalité), représenté par XT ou X (X souligné) dans PowerAMC. En effet, un demandeur ne peut être que permanent ou vacataire (et pas autre chose) :

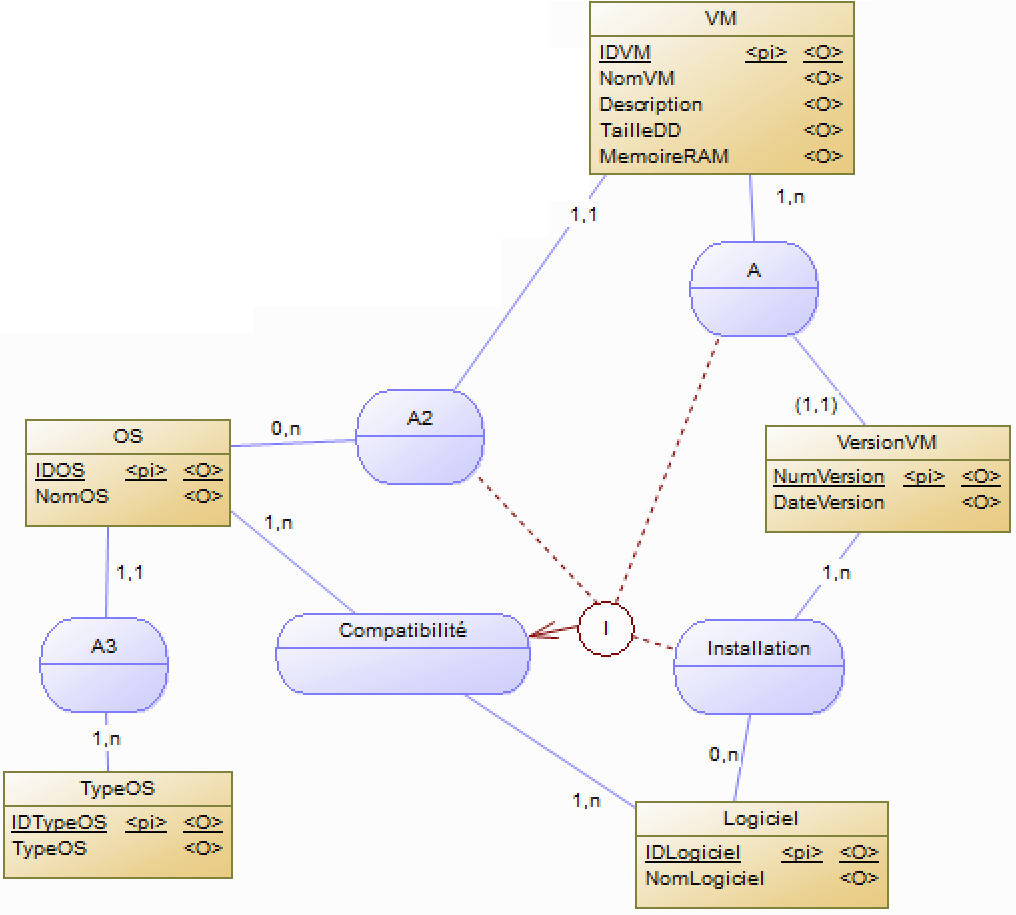
****

* **Un logiciel n’est compatible que sur certains OS. On souhaite gérer cette compatibilité afin de ne pas saisir d’erreur dans les logiciels installés.**

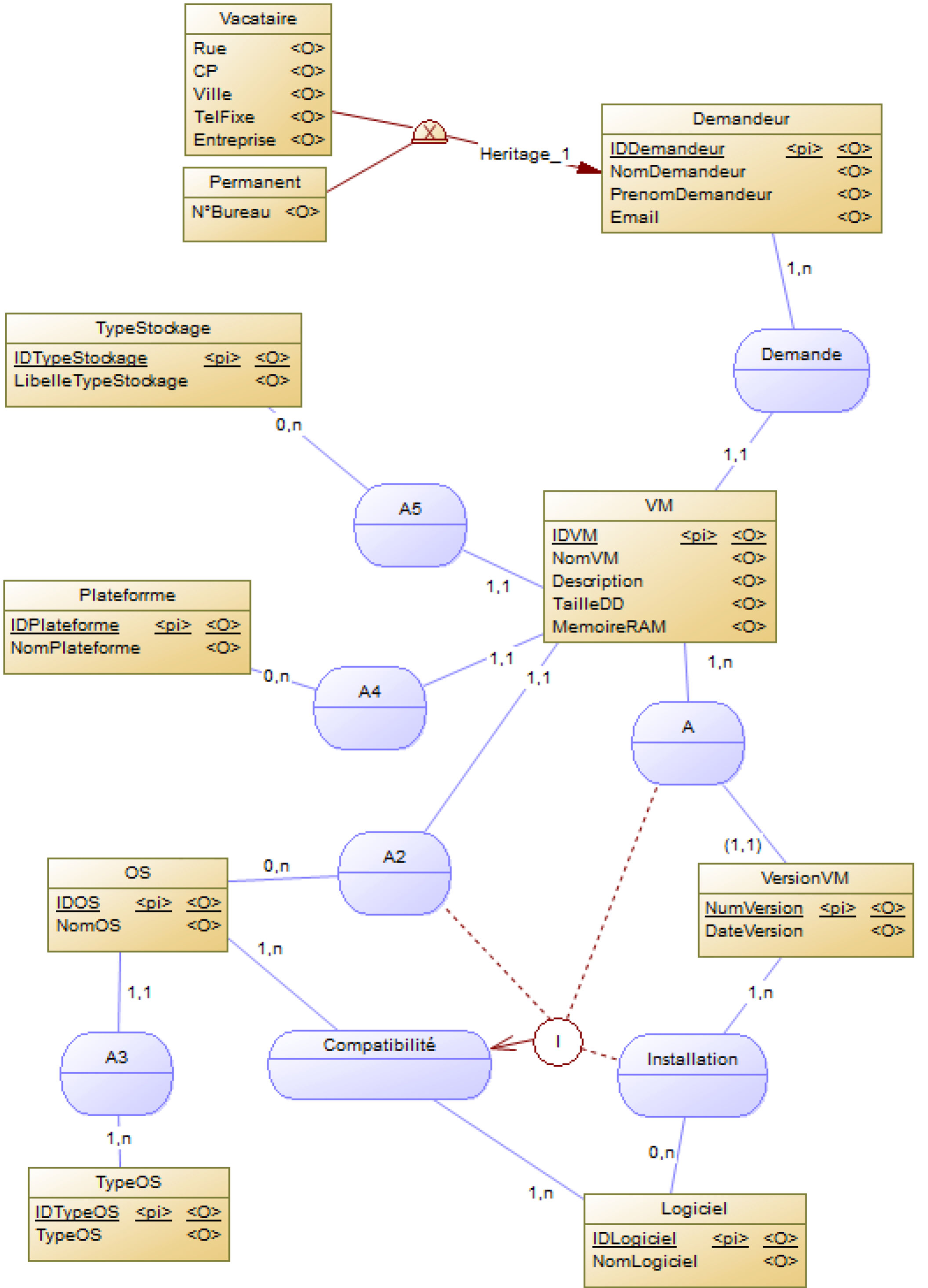
****

Il existe une contrainte d’inclusion entre les associations Compatibilité et Installation. Le couple (IDLogiciel, IDOS de la VM) de l’installation doit être dans l’association Compatibilité. Pour accéder à l’IDOS de la VM (qui n’est pas dans l’association « Installation »), les associations « A » et « A2 » sont nécessaires également.

*Remarque : Il n’est pas possible de modéliser cette contrainte dans PowerAMC, sauf à utiliser des symboles libres.*

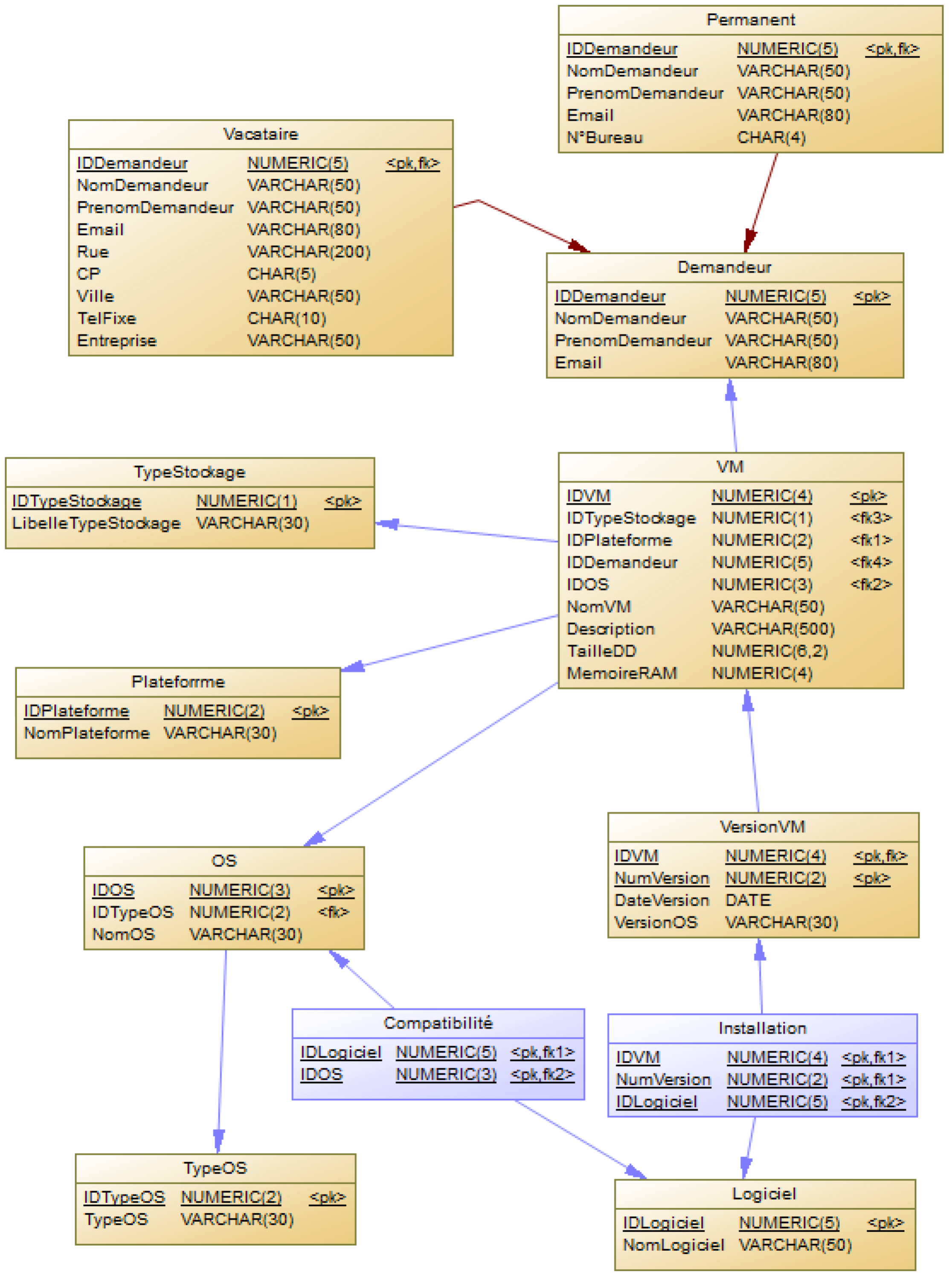
****

**MCD final :**

****

**MLD :**

Consiste à créer le modèle relationnel.

****

**Questions à (se) poser :**

* **Pourquoi le téléphone est-il de type CHAR ?** Sinon le 0 de début disparaît (idem pour un code postal français).
* **Pourquoi le CP est-il de type CHAR et non VARCHAR ?** Un caractère de longueur fixe tient moins de place mémoire qu’un caractère de longueur variable. En outre, un CP ne peut contenir que 5 caractères (en l’occurrence 5 chiffres) ni plus ni moins.
* **Combien y a-t-il de clés primaires dans la table VersionVM ?** 1 seule composée de 2 champs !

**MPD :**

**Actions possibles :**

* Renommage des tables et/ou colonnes (peut également être fait à l’étape précédente) en respectant les règles de nommage
* Conservation ou non des tables de référence ?

Ici, tables de référence : TypeOS, Plateforme, TypeStockage. Leurs données vont-elles changer ou non à l’avenir ? Si l’on hésite, il est préférable de garder la table de référence.

* + TypeStockage : est-ce que cette liste est susceptible d'évoluer ou bien est-elle figée ? Non => contrainte de validation.
  + TypeOS, Plateforme : peuvent évoluer => conservation des tables. Par exemple, le type d’OS « Mobile » n’existait pas il y a de cela quelques années, de même certains peuvent disparaître.

*Remarque : dans PowerAMC, pour ne pas générer une entité, il suffit au niveau du MCD de décocher « Générer » (propriété de l’entité).*

**Questions à (se) poser :**

* **Pourquoi doit-on toujours conserver une table de référence si les valeurs de ses champs ne sont pas figées ?**
  + - Pour éviter les doublons sur des champs de type caractère variable (ou caractère) dans la table contenant la clé étrangère **=>** gain en volumétrie (seul le n° ou le code est doublonné, et non le libellé, en général plus long).
    - Pour éviter les erreurs de saisie (une seule saisie réalisée dans la table de référence).
    - Pour avoir la possibilité de disposer d’une liste déroulante dans l’application cliente et donc de sélectionner la valeur à partir de la table de référence sans avoir à la saisir…
* Table Installation : il est préférable de limiter les clés primaires portant sur plusieurs champs à 2 champs au maximum. On créé alors une clé artificielle ID (INS\_ID). Pour assurer l’unicité des enregistrements sur (IDVM, NumVersion, IDLogiciel), on créé une contrainte de clé unique portant sur ces champs.

**ATTENTION : une clé primaire ne peut jamais être NULL contrairement à une clé unique => penser à forcer à NOT NULL l’ensemble des champs composant la clé unique.**

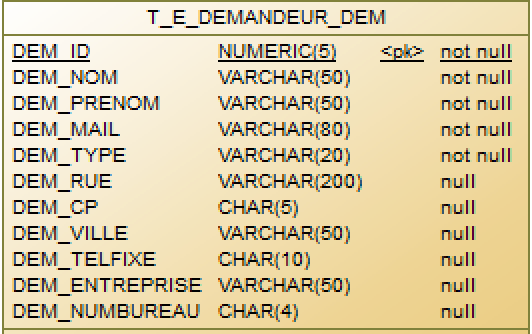
* HERITAGE : le MLD implique une duplication des données (nom, prénom, mail). C’est une perte de place. 3 solutions possibles (en simplifiant) :
  + Solution 1 : on aplatit le modèle et on créée une seule table. On ajoute du code (application cliente et/ou trigger) pour assurer la cohérence des données.
  + Solution 2 : on garde toutes les tables, mais on supprime les attributs dupliqués dans les sous-types.
  + Solution 3 : on ne conserve que les tables sous-types. Les champs du sur-type sont intégrés aux sous-types.

Cette optimisation dépend du contexte. Généralement, on procède ainsi :

|  |  |
| --- | --- |
| CAS 1 : Les associations ne portent que sur le sur-type  heritage1 | * S’il y a peu de sous-types, on applique la solution 1 (1 seule table) et on créée du code de cohérence des données. * S’il y a beaucoup de sous-types et/ou si beaucoup d’attributs dans les sous-types, le code assurant la cohérence des données sera complexe et long => solution 2 (toutes les tables sont conservées). |
| CAS 2 : Des associations existent aux niveaux sur-type et sous-types  heritage2 | - On garde les 2 niveaux => solution 2 afin d’assurer la cohérence des jointures (même si on peut faire des jointures conditionnelles (pas vu en DUT)) |
| CAS 3 : Des associations n’existent qu’au niveau sous-type  heritage3 | - Le sur-type ne sera pas conservé => solution 3. |

Nous sommes dans le CAS 1, donc 2 solutions possibles :

SOLUTION 1 :

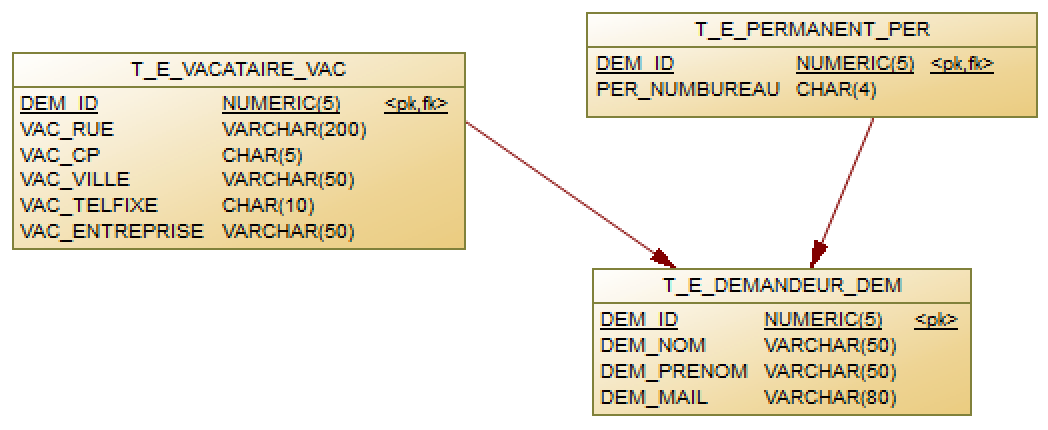


On ajoute un champ obligatoire DEM\_TYPE (TypeDemandeur) qui aura pour valeur « Permanent » ou « Vacataire » (contrainte CHECK). Les champs issus des sous-types sont mis en non obligatoires et leur saisie dépendra du type de demandeur sélectionné :

* Si Permanent : NumBureau est saisi ; Rue, CP, Ville, TelFixe, Entreprise sont vides.
* Si Vacataire : NumBureau est vide; Rue, CP, Ville, TelFixe, Entreprise sont saisis.

Nous ajouterons un trigger dans un prochain TP qui permettra de vérifier si ces contraintes de saisie sont bien respectées.

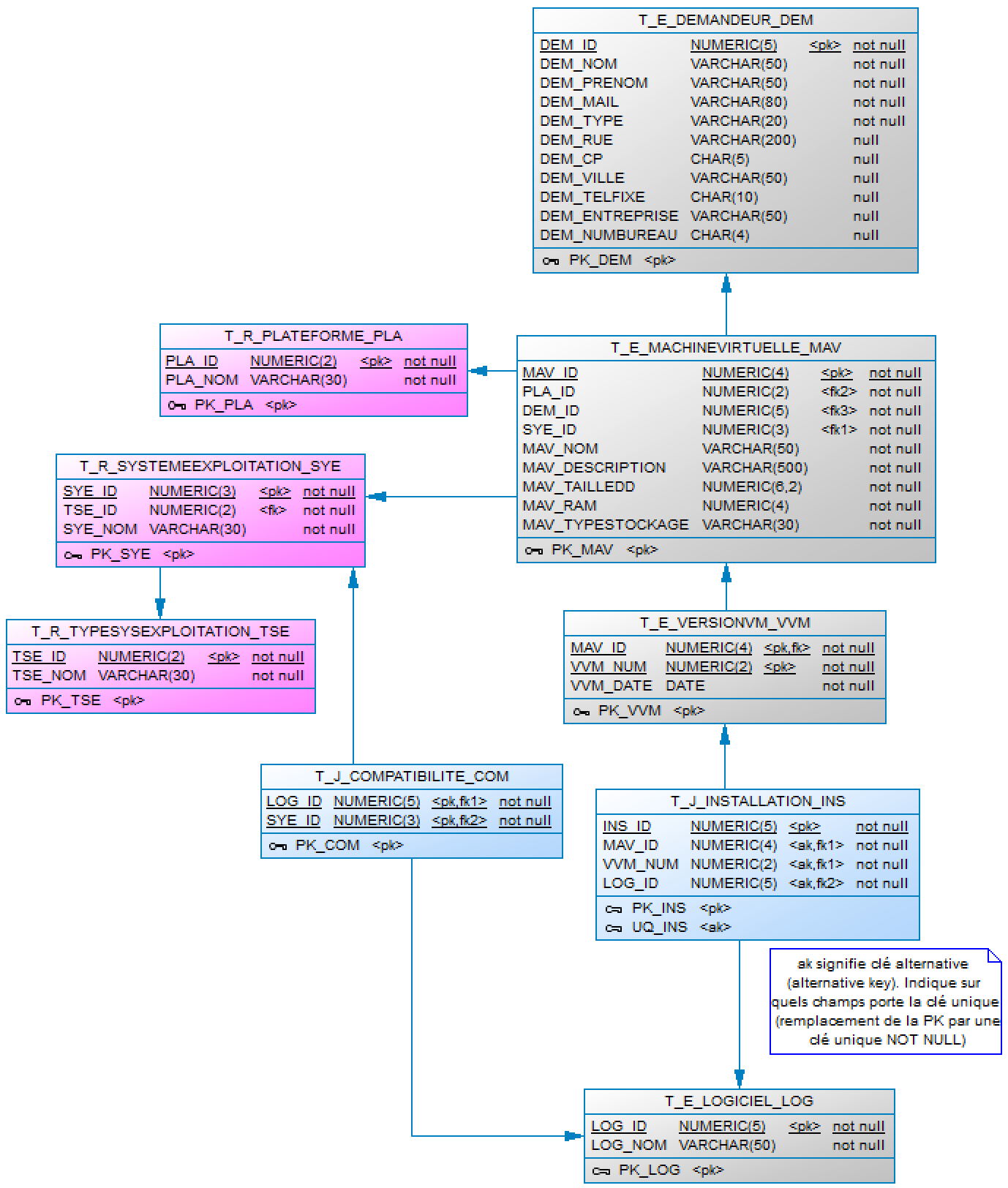
SOLUTION 2 :



L’insertion des données se fera d’abord dans Demandeur, puis en fonction du type de demandeur (choix dans une liste déroulante par exemple), l’insertion des autres données se fera dans le sous-type adéquat.

**NOUS AVONS CHOISI LA SOLUTION 1 => MPD :**

*Remarque : Oracle, comme PostgreSQL, est un SGBD Relationnel-Objet et sait parfaitement gérer l’héritage entre tables. On n’est donc pas obligé de transformer l’héritage en des contraintes d’intégrité référentielle (FK) classiques.*



**Tables :**

Table **T\_J\_COMPATIBILITE\_COM :**

|  |  |  |
| --- | --- | --- |
| LOG\_ID | numeric(5) | Clé étrangère qui référence la table Logiciel.  Participe à la clé primaire. |
| SYE\_ID | numeric(3) | Clé étrangère qui référence la table SystemeExploitation.  Participe à la clé primaire. |

Tous les champs sont obligatoires.

Table **T\_E\_DEMANDEUR\_DEM :**

|  |  |  |
| --- | --- | --- |
| DEM\_ID | numeric(5) | Clé primaire |
| DEM\_NOM | varchar(50) |  |
| DEM\_PRENOM | varchar(50) |  |
| DEM\_MAIL | varchar(80) |  |
| DEM\_TYPE | varchar(20) | Contrainte de validation. Valeurs possibles : Permanent, Vacataire |
| DEM\_RUE | varchar(200) |  |
| DEM\_CP | char(5) |  |
| DEM\_VILLE | varchar(50) |  |
| DEM\_TELFIXE | char(10) |  |
| DEM\_ENTREPRISE | varchar(50) |  |
| DEM\_NUMBUREAU | char(4) |  |

Tous les champs sont obligatoires sauf l’adresse (rue, CP, ville), TelFixe, Entreprise et NumBureau.

Table **T\_J\_INSTALLATION\_INS :**

|  |  |  |
| --- | --- | --- |
| INS\_ID | numeric(6) | Clé primaire |
| MAV\_ID | numeric(4) | Participe à la clé étrangère qui référence la table VersionVM.  Participe à la clé unique.  ouvrage |
| VVM\_NUM | numeric(2) | Participe à la clé étrangère qui référence la table VersionVM.  Participe à la clé unique. |
| LOG\_ID | numeric(5) | Clé étrangère qui référence la table Logiciel.  Participe à la clé unique. |

Tous les champs sont obligatoires.

Table **T\_E\_LOGICIEL\_LOG :**

|  |  |  |  |
| --- | --- | --- | --- |
| LOG\_ID | | numeric(5) | Clé primaire |
| LOG\_NOM | varchar(50) | |  |

Tous les champs sont obligatoires.

Table **T\_R\_SYSTEMEEXPLOITATION\_SYE :**

|  |  |  |
| --- | --- | --- |
| SYE\_ID | numeric(3) | Clé primaire |
| TSE\_ID | numeric(2) | Clé étrangère qui référence la table TypeSysExploitation. |
| SYE\_NOM | varchar(30) |  |

Tous les champs sont obligatoires.

Table **T\_R\_PLATEFORME\_PLA :**

|  |  |  |
| --- | --- | --- |
| PLA\_ID | numeric(2) | Clé primaire |
| PLA\_NOM | varchar(30) |  |

Tous les champs sont obligatoires.

Table **T\_R\_TYPESYSEXPLOITATION\_TSE :**

|  |  |  |
| --- | --- | --- |
| TSE\_ID | numeric(2) | Clé primaire |
| TSE\_NOM | varchar(30) |  |

Tous les champs sont obligatoires.

Table **T\_E\_VERSIONVM\_VVM :**

|  |  |  |
| --- | --- | --- |
| MAV\_ID | numeric(4) | Participe à la clé primaire.  Clé étrangère qui référence la table MachineVirtuelle. |
| VVM\_NUM | numeric(2) | Participe à la clé primaire |
| VVM\_DATE | date | Par défaut date du jour (current\_date) |

Tous les champs sont obligatoires.

Table **T\_E\_MACHINEVIRTUELLE\_MAV :**

|  |  |  |
| --- | --- | --- |
| MAV\_ID | numeric(4) | Clé primaire |
| PLA\_ID | numeric(2) | Clé étrangère qui référence la table Plateforme |
| DEM\_ID | numeric(5) | Clé étrangère qui référence la table Demandeur |
| SYE\_ID | numeric(3) | Clé étrangère qui référence la table SystemeExploitation. |
| MAV\_NOM | varchar(50) |  |
| MAV\_DESCRIPTION | varchar(500) |  |
| MAV\_TAILLEDD | numeric(6,2) |  |
| MAV\_RAM | numeric(4) | Contrainte de validation. Doit être inférieur à 6144 Mo. |
| MAV\_TYPESTOCKAGE | varchar(30) | Contrainte de validation. Valeurs possibles : Dynamiquement alloué, Taille fixe. |

Tous les champs sont obligatoires.

**Question 2 :**

**drop table if exists T\_E\_DEMANDEUR\_DEM cascade;**

**drop table if exists T\_E\_LOGICIEL\_LOG cascade;**

**drop table if exists T\_E\_MACHINEVIRTUELLE\_MAV cascade;**

**drop table if exists T\_E\_VERSIONVM\_VVM cascade;**

**drop table if exists T\_J\_COMPATIBILITE\_COM cascade;**

**drop table if exists T\_J\_INSTALLATION\_INS cascade;**

**drop table if exists T\_R\_PLATEFORME\_PLA cascade;**

**drop table if exists T\_R\_SYSTEMEEXPLOITATION\_SYE cascade;**

**drop table if exists T\_R\_TYPESYSEXPLOITATION\_TSE cascade;**

**/\*==============================================================\*/**

**/\* Table : T\_E\_DEMANDEUR\_DEM \*/**

**/\*==============================================================\*/**

**create table T\_E\_DEMANDEUR\_DEM (**

**DEM\_ID NUMERIC(5) not null,**

**DEM\_NOM VARCHAR(50) not null,**

**DEM\_PRENOM VARCHAR(50) not null,**

**DEM\_MAIL VARCHAR(80) not null,**

**DEM\_TYPE VARCHAR(20) not null**

**constraint CK\_DEM\_TYPE check(DEM\_TYPE in ('Permanent', 'Vacataire')),**

**DEM\_RUE VARCHAR(200) null,**

**DEM\_CP CHAR(5) null,**

**DEM\_VILLE VARCHAR(50) null,**

**DEM\_TELFIXE CHAR(10) null,**

**DEM\_ENTREPRISE VARCHAR(50) null,**

**DEM\_NUMBUREAU CHAR(4) null,**

**constraint PK\_DEM primary key (DEM\_ID)**

**);**

**/\*==============================================================\*/**

**/\* Table : T\_E\_LOGICIEL\_LOG \*/**

**/\*==============================================================\*/**

**create table T\_E\_LOGICIEL\_LOG (**

**LOG\_ID NUMERIC(5) not null,**

**LOG\_NOM VARCHAR(50) not null,**

**constraint PK\_LOG primary key (LOG\_ID)**

**);**

**/\*==============================================================\*/**

**/\* Table : T\_E\_MACHINEVIRTUELLE\_MAV \*/**

**/\*==============================================================\*/**

**create table T\_E\_MACHINEVIRTUELLE\_MAV (**

**MAV\_ID NUMERIC(4) not null,**

**PLA\_ID NUMERIC(2) not null,**

**DEM\_ID NUMERIC(5) not null,**

**SYE\_ID NUMERIC(3) not null,**

**MAV\_NOM VARCHAR(50) not null,**

**MAV\_DESCRIPTION VARCHAR(500) not null,**

**MAV\_TAILLEDD NUMERIC(6,2) not null,**

**MAV\_RAM NUMERIC(4) not null**

**constraint CK\_MAV\_RAM check(MAV\_RAM <= 6144),**

**MAV\_TYPESTOCKAGE VARCHAR(30) not null**

**constraint CK\_MAV\_TYPESTOCKAGE check(MAV\_TYPESTOCKAGE in ('Dynamiquement alloué', 'Taille fixe')),**

**constraint PK\_MAV primary key (MAV\_ID)**

**);**

**/\*==============================================================\*/**

**/\* Table : T\_E\_VERSIONVM\_VVM \*/**

**/\*==============================================================\*/**

**create table T\_E\_VERSIONVM\_VVM (**

**MAV\_ID NUMERIC(4) not null,**

**VVM\_NUM NUMERIC(2) not null,**

**VVM\_DATE DATE default current\_date not null,**

**constraint PK\_VVM primary key (MAV\_ID, VVM\_NUM)**

**);**

**/\*==============================================================\*/**

**/\* Table : T\_J\_COMPATIBILITE\_COM \*/**

**/\*==============================================================\*/**

**create table T\_J\_COMPATIBILITE\_COM (**

**LOG\_ID NUMERIC(5) not null,**

**SYE\_ID NUMERIC(3) not null,**

**constraint PK\_COM primary key (LOG\_ID, SYE\_ID)**

**);**

**/\*==============================================================\*/**

**/\* Table : T\_J\_INSTALLATION\_INS \*/**

**/\*==============================================================\*/**

**create table T\_J\_INSTALLATION\_INS (**

**INS\_ID NUMERIC(5) not null,**

**MAV\_ID NUMERIC(4) not null,**

**VVM\_NUM NUMERIC(2) not null,**

**LOG\_ID NUMERIC(5) not null,**

**constraint PK\_INS primary key (INS\_ID),**

**constraint UQ\_INS unique (MAV\_ID, VVM\_NUM, LOG\_ID)**

**);**

**/\*==============================================================\*/**

**/\* Table : T\_R\_PLATEFORME\_PLA \*/**

**/\*==============================================================\*/**

**create table T\_R\_PLATEFORME\_PLA (**

**PLA\_ID NUMERIC(2) not null,**

**PLA\_NOM VARCHAR(30) not null,**

**constraint PK\_PLA primary key (PLA\_ID)**

**);**

**/\*==============================================================\*/**

**/\* Table : T\_R\_SYSTEMEEXPLOITATION\_SYE \*/**

**/\*==============================================================\*/**

**create table T\_R\_SYSTEMEEXPLOITATION\_SYE (**

**SYE\_ID NUMERIC(3) not null,**

**TSE\_ID NUMERIC(2) not null,**

**SYE\_NOM VARCHAR(30) not null,**

**constraint PK\_SYE primary key (SYE\_ID)**

**);**

**/\*==============================================================\*/**

**/\* Table : T\_R\_TYPESYSEXPLOITATION\_TSE \*/**

**/\*==============================================================\*/**

**create table T\_R\_TYPESYSEXPLOITATION\_TSE (**

**TSE\_ID NUMERIC(2) not null,**

**TSE\_NOM VARCHAR(30) not null,**

**constraint PK\_TSE primary key (TSE\_ID)**

**);**

**alter table T\_E\_MACHINEVIRTUELLE\_MAV**

**add constraint FK\_MAV\_SYE foreign key (SYE\_ID)**

**references T\_R\_SYSTEMEEXPLOITATION\_SYE (SYE\_ID);**

**alter table T\_E\_MACHINEVIRTUELLE\_MAV**

**add constraint FK\_MAV\_PLA foreign key (PLA\_ID)**

**references T\_R\_PLATEFORME\_PLA (PLA\_ID);**

**alter table T\_E\_MACHINEVIRTUELLE\_MAV**

**add constraint FK\_MAV\_DEM foreign key (DEM\_ID)**

**references T\_E\_DEMANDEUR\_DEM (DEM\_ID);**

**alter table T\_E\_VERSIONVM\_VVM**

**add constraint FK\_VVM\_MAV foreign key (MAV\_ID)**

**references T\_E\_MACHINEVIRTUELLE\_MAV (MAV\_ID);**

**alter table T\_J\_COMPATIBILITE\_COM**

**add constraint FK\_COM\_LOG foreign key (LOG\_ID)**

**references T\_E\_LOGICIEL\_LOG (LOG\_ID);**

**alter table T\_J\_COMPATIBILITE\_COM**

**add constraint FK\_COM\_SYE foreign key (SYE\_ID)**

**references T\_R\_SYSTEMEEXPLOITATION\_SYE (SYE\_ID);**

**alter table T\_J\_INSTALLATION\_INS**

**add constraint FK\_INS\_VVM foreign key (MAV\_ID, VVM\_NUM)**

**references T\_E\_VERSIONVM\_VVM (MAV\_ID, VVM\_NUM);**

**alter table T\_J\_INSTALLATION\_INS**

**add constraint FK\_INS\_LOG foreign key (LOG\_ID)**

**references T\_E\_LOGICIEL\_LOG (LOG\_ID);**

**alter table T\_R\_SYSTEMEEXPLOITATION\_SYE**

**add constraint FK\_SYE\_TSE foreign key (TSE\_ID)**

**references T\_R\_TYPESYSEXPLOITATION\_TSE (TSE\_ID);**

**Question 5 :**

Create sequence SEQ\_DEM start with 1 increment by 1 owned by T\_E\_DEMANDEUR\_DEM.DEM\_ID;

-- OU Create sequence SEQ\_DEM owned by T\_E\_DEMANDEUR\_DEM.DEM\_ID;

**Question 6 :**

Alter table T\_E\_DEMANDEUR\_DEM add constraint UQ\_DEM unique (DEM\_NOM, DEM\_PRENOM, DEM\_TELFIXE);

**Question 7 :**

Alter table T\_E\_DEMANDEUR\_DEM add DEM\_MOBILE char(10);

Alter table T\_E\_DEMANDEUR\_DEM add constraint CK\_DEM\_MOBILE check (DEM\_MOBILE like '06%' OR DEM\_MOBILE like '07%' OR DEM\_MOBILE IS NULL); -- et non (DEM\_MOBILE like '06%' ~~OR '07%'~~)

**Question 8 :**

Il faut créer des index sur les clés étrangères.

Les FK correspondent aux flèches du MPD reliant les tables => 9 FK => 9 index à créer :

- Table T\_E\_COMPATIBILITE\_COM : 1 index sur LOG\_ID et 1 index sur SYE\_ID

- Table T\_E\_INSTALLATION\_INS : 1 index sur LOG\_ID et 1 index sur 2 champs (MAV\_ID, VVM\_NUM)

- Table T\_R\_SYSTEMEEXPLOITATION\_SYE: colonne TSE\_ID

- Table T\_E\_VERSIONVM\_VVM : colonne MAV\_ID

- Table T\_E\_MACHINEVIRTUELLE\_MAV : 3 index sur SYE\_ID, PLA\_ID, DEM\_ID

create index IDX\_COM\_LOG\_ID on T\_J\_COMPATIBILITE\_COM (LOG\_ID);

create index IDX\_COM\_SYE\_ID on T\_J\_COMPATIBILITE\_COM (SYE\_ID);

**create index IDX\_INS\_MAV\_ID\_VVM\_NUM on T\_J\_INSTALLATION\_INS (MAV\_ID, VVM\_NUM);**

create index IDX\_INS\_LOG\_ID on T\_J\_INSTALLATION\_INS (LOG\_ID);

create index IDX\_SYE\_TSE\_ID on T\_R\_SYSTEMEEXPLOITATION\_SYE (TSE\_ID);

create index IDX\_VVM\_MAV\_ID on T\_E\_VERSIONVM\_VVM(MAV\_ID);

create index IDX\_MAV\_SYE\_ID on T\_E\_MACHINEVIRTUELLE\_MAV (SYE\_ID);

create index IDX\_MAV\_PLA\_ID on T\_E\_MACHINEVIRTUELLE\_MAV (PLA\_ID);

create index IDX\_MAV\_DEM\_ID on T\_E\_MACHINEVIRTUELLE\_MAV (DEM\_ID);

**Question 9 :**

Alter table T\_E\_DEMANDEUR\_DEM alter column DEM\_TYPE set default 'Permanent';

**Question 10 :**

Insert into T\_E\_DEMANDEUR\_DEM (DEM\_ID, DEM\_NOM, DEM\_PRENOM, DEM\_MAIL, DEM\_TYPE, DEM\_RUE, DEM\_CP, DEM\_VILLE, DEM\_TELFIXE, DEM\_ENTREPRISE, DEM\_NUMBUREAU) values (nextval('**SEQ\_DEM')**, 'ALBERT', 'Anne', 'anne.albert@gmail.com', 'Vacataire', '13 rue de la Gare', '69002', 'LYON', '0450505055', 'EDF', null);

Insert into T\_E\_DEMANDEUR\_DEM (DEM\_ID, DEM\_NOM, DEM\_PRENOM, DEM\_MAIL, DEM\_TYPE, DEM\_RUE, DEM\_CP, DEM\_VILLE, DEM\_TELFIXE, DEM\_ENTREPRISE, DEM\_NUMBUREAU) values (nextval('**SEQ\_DEM')**, 'BERNAUD', 'Barnabé', 'barnabe.bernaud@cpe.fr', 'Permanent', NULL, NULL, NULL, NULL, NULL, 'G102');

**Question 12 :**

select dem.dem\_nom as "Nom demandeur", dem.dem\_prenom as "Prénom demandeur",

count(mav.mav\_id) as "nb vm"

from T\_E\_DEMANDEUR\_DEM dem

left join T\_E\_MACHINEVIRTUELLE\_MAV mav on dem.DEM\_ID=mav.DEM\_ID

group by dem.dem\_nom, dem.dem\_prenom

order by 1,2;

-- Un COUNT sur un champ NULL renvoie 0. Ici si un demandeur n’a pas de VM, tous les champs de T\_E\_MACHINEVIRTUELLE\_MAV seront NULL.

**Question 13 :**

SELECT tse.TSE\_NOM as "Type OS", sye.SYE\_NOM as "OS", COUNT(mav.MAV\_ID) AS "nb VMs"

FROM T\_E\_MACHINEVIRTUELLE\_MAV mav

JOIN T\_R\_SYSTEMEEXPLOITATION\_SYE sye ON mav.sye\_id=sye.sye\_id

JOIN T\_R\_TYPESYSEXPLOITATION\_TSE tse ON sye.tse\_id=tse.tse\_id

GROUP BY tse.TSE\_NOM, sye.SYE\_NOM

ORDER BY 1, 2;

**Question 14 :**

drop view v\_derversionvm;

CREATE VIEW v\_derversionvm AS

SELECT mav.mav\_id as "ID VM", mav.mav\_nom as "Nom VM", mav.mav\_description as description, max(vvm.vvm\_date) as "date der version"

FROM T\_E\_MACHINEVIRTUELLE\_MAV mav

JOIN T\_E\_VERSIONVM\_VVM vvm ON mav.mav\_id=vvm.mav\_id

GROUP BY mav.mav\_id, mav.mav\_nom, mav.mav\_description

HAVING to\_char(max(vvm.vvm\_date), 'YYYY')=to\_char(current\_date, 'YYYY')

ORDER BY 1;

SELECT \* FROM v\_derversionvm;

**Question 15 :**

SELECT vvm.mav\_id as "ID VM", vvm.vvm\_num as "Num version", count(ins.log\_id) as "Nb logiciels",

CASE count(ins.log\_id)

WHEN 0 THEN 'OS nu'

WHEN 1 THEN 'Peu'

WHEN 2 THEN 'Peu'

WHEN 3 THEN 'Normal'

WHEN 4 THEN 'Normal'

WHEN 5 THEN 'Normal'

ELSE 'Beaucoup de travail !'

END as Qualification

FROM T\_E\_VERSIONVM\_VVM vvm

LEFT JOIN T\_J\_INSTALLATION\_INS ins ON vvm.mav\_id=ins.mav\_id **AND vvm.vvm\_num=ins.vvm\_num**

GROUP BY vvm.mav\_id, vvm.vvm\_num

ORDER BY 1, 2

-- Jointure sur les 2 champs de la FK, sinon produit cartésien

OU

SELECT vvm.mav\_id as "ID VM", vvm.vvm\_num as "Num version", count(ins.log\_id) as "Nb logiciels",

CASE

WHEN count(ins.log\_id)=0 THEN 'OS nu'

WHEN count(ins.log\_id) between 1 AND 2 THEN 'Peu'

WHEN count(ins.log\_id) between 3 AND 5 THEN 'Normal'

ELSE 'Beaucoup de travail !'

END as Qualification

FROM T\_E\_VERSIONVM\_VVM vvm

LEFT JOIN T\_J\_INSTALLATION\_INS ins ON vvm.mav\_id=ins.mav\_id **AND vvm.vvm\_num=ins.vvm\_num**

GROUP BY vvm.mav\_id, vvm.vvm\_num

ORDER BY 1, 2

**Question 16 :**

CREATE TABLE INFOSVM AS

SELECT mav.mav\_id, mav.mav\_nom, mav.mav\_description, mav.mav\_tailledd, mav.mav\_ram, mav.mav\_typestockage, dem.dem\_nom, dem.dem\_prenom

FROM T\_E\_MACHINEVIRTUELLE\_MAV mav

JOIN T\_E\_DEMANDEUR\_DEM dem ON mav.dem\_id=dem.dem\_id;

SELECT \* FROM INFOSVM;

Pas de contraintes (FK, PK, CHECK, etc.) hormis not null sur la table car les contraintes ne sont pas créées lors d’un CREATE TABLE AS…

**Question 17 :**

* Première étape : créez une table temporaire, dans laquelle les informations vont être ajoutées au fur et à mesure de leur calcul. Pour une telle table, il est possible de ne pas définir de contrainte de clé pri­maire car c'est une table de "travail" qui sera vidée dès que le résultat attendu aura été fourni.

CREATE TEMPORARY TABLE SyntheseInstallations (

idvm NUMERIC(4),

nomvm VARCHAR(50),

numversion NUMERIC (2),

nbinstallationsversion NUMERIC(10),

nbinstallationsvm NUMERIC(10)

)

ON COMMIT PRESERVE ROWS;

* 2ème étape : Ajoutez les informations relatives à chaque version (insert)

INSERT INTO SyntheseInstallations(idvm, nomvm, numversion, nbinstallationsversion)

SELECT mav.mav\_id, mav.mav\_nom, vvm.vvm\_num, COUNT(\*)

FROM T\_E\_MACHINEVIRTUELLE\_MAV mav

JOIN T\_E\_VERSIONVM\_VVM vvm ON mav.mav\_id=vvm.mav\_id

JOIN T\_J\_INSTALLATION\_INS ins ON vvm.mav\_id=ins.mav\_id **AND vvm.vvm\_num=ins.vvm\_num**

GROUP BY mav.mav\_id, mav.mav\_nom, vvm.vvm\_num;

* 3ème étape : Ajoutez les informations relatives à chaque ouvrage (update)

UPDATE SyntheseInstallations S SET nbinstallationsvm=(

SELECT COUNT(\*) FROM T\_J\_INSTALLATION\_INS ins

WHERE ins.mav\_id = S.idvm

);

OU

UPDATE SyntheseInstallations S SET nbinstallationsvm=(

SELECT SUM(nbinstallationsversion) FROM SyntheseInstallations S2

WHERE S2.idvm = S.idvm

);

select \* from SyntheseInstallations

order by 1, 3;

* Puis supprimez la table.

Drop TABLE SyntheseInstallations;

**Question 18 :**

SELECT mav.mav\_id as "ID VM", mav.mav\_nom As Nom

FROM T\_E\_MACHINEVIRTUELLE\_MAV mav

JOIN T\_E\_VERSIONVM\_VVM vvm ON mav.mav\_id=vvm.mav\_id

WHERE vvm.vvm\_date > current\_date - interval '36 months'

GROUP BY mav.mav\_id, mav.mav\_nom

HAVING count(\*)>=2;

**Question 19 :**

SELECT log\_id as "ID Logiciel", log\_nom as "Nom"

FROM T\_E\_LOGICIEL\_LOG

WHERE log\_id NOT IN (

SELECT log\_id

FROM T\_E\_VERSIONVM\_VVM vvm

JOIN T\_J\_INSTALLATION\_INS ins ON **vvm.mav\_id=ins.mav\_id AND vvm.vvm\_num=ins.vvm\_num**

WHERE vvm.vvm\_date > current\_date - interval '70 months'

);

**Question 20 :**

SELECT log\_id as "ID Logiciel", log\_nom as "Nom"

FROM T\_E\_LOGICIEL\_LOG log

WHERE NOT EXISTS (

SELECT 'X'

FROM T\_R\_SYSTEMEEXPLOITATION\_SYE sye

WHERE NOT EXISTS (

SELECT 'X'

FROM T\_J\_COMPATIBILITE\_COM com

WHERE com.log\_id=log.log\_id AND com.sye\_id=sye.sye\_id

)

);

OU

SELECT log.log\_id as "ID Logiciel", log.log\_nom as "Nom"

FROM T\_J\_COMPATIBILITE\_COM com

JOIN T\_E\_LOGICIEL\_LOG log ON com.log\_id=log.log\_id

GROUP BY log.log\_id, log.log\_nom

HAVING COUNT(DISTINCT com.sye\_id) = (

SELECT COUNT(\*) FROM T\_R\_SYSTEMEEXPLOITATION\_SYE

)

ORDER BY 1;

**Question 21 :**

SELECT log.log\_id as "ID Logiciel", log.log\_nom as "Nom"

FROM T\_J\_COMPATIBILITE\_COM com

JOIN T\_E\_LOGICIEL\_LOG log ON com.log\_id=log.log\_id

JOIN T\_R\_SYSTEMEEXPLOITATION\_SYE sye ON com.sye\_id=sye.sye\_id

JOIN T\_R\_TYPESYSEXPLOITATION\_TSE tse ON sye.tse\_id=tse.tse\_id

WHERE tse.tse\_nom='Mac OS/X'

GROUP BY log.log\_id, log.log\_nom

HAVING COUNT(DISTINCT com.sye\_id) = (

SELECT COUNT(\*)

FROM T\_R\_SYSTEMEEXPLOITATION\_SYE sye

JOIN T\_R\_TYPESYSEXPLOITATION\_TSE tse ON sye.tse\_id=tse.tse\_id

WHERE tse.tse\_nom ='Mac OS/X'

)

ORDER BY 1;

OU

SELECT log\_id as "ID Logiciel", log\_nom as "Nom"

FROM T\_E\_LOGICIEL\_LOG log

WHERE NOT EXISTS

(

SELECT 'X'

FROM T\_R\_SYSTEMEEXPLOITATION\_SYE sye

JOIN T\_R\_TYPESYSEXPLOITATION\_TSE tse ON sye.tse\_id=tse.tse\_id

WHERE tse.tse\_nom ='Mac OS/X'

AND NOT EXISTS (

SELECT 'X'

FROM T\_J\_COMPATIBILITE\_COM com

WHERE com.log\_id=log.log\_id AND com.sye\_id=sye.sye\_id

)

);