Le modèle de données de Rose (suite)
Exemples de création d'éléments directement dans le modèle de données

Table des matières

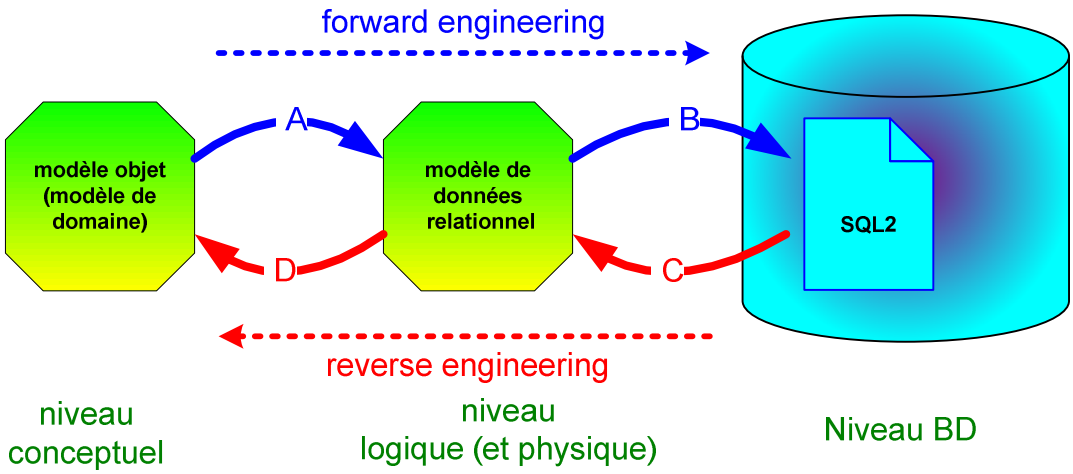
1 Introduction

On suppose ici que vous avez créé la base de données et son schéma (voir documents précédents).
Pour représenter le schéma, on utilise un modèle de données qui n'est qu'une fenêtre sur ce schéma. Ce modèle de données utilise le formalisme UML avec, en plus, le profile « UML profile for Data Modeling »

Vous pouvez remplir un schéma par une transformation automatique (forward-engineering ou backward- engineering) ou vous pouvez créer des éléments directement dans le schéma en travaillant dans un modèle de données.

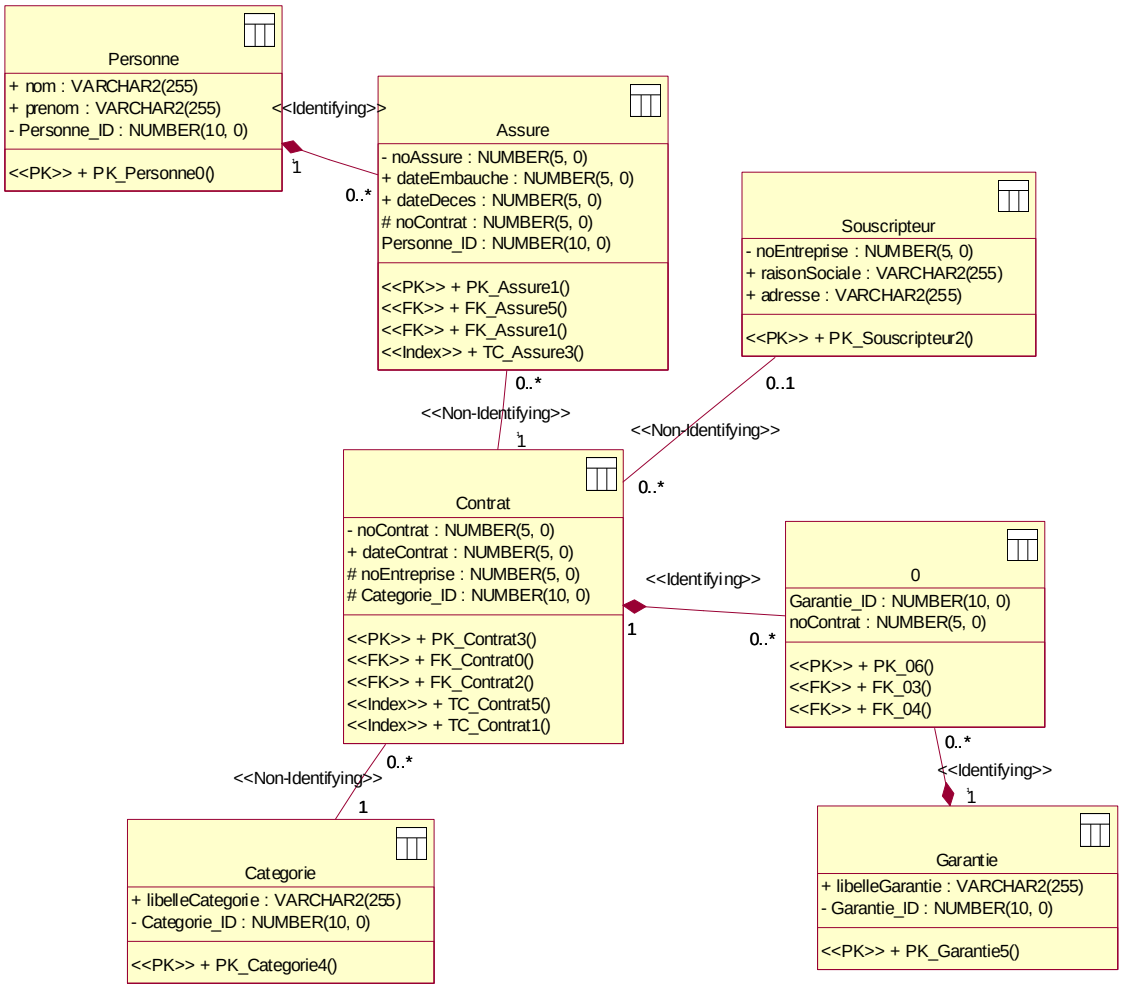
C'est ce dernier cas que nous traitons dans ce document.
Vous pouvez :

- créer des tables, colonnes, clés (primaires et uniques), associations entre tables, contraintes de type CHECK et index ;
- créer des déclencheurs, des domaines, des procédures cataloguées et des vues ;
- programmer l'intégrité référentielle (cohérence père-fils) avec ou sans déclencheurs ;



Par la suite, le schéma peut générer un script SQL permettant de créer les tables du SGBD.

Voici un exemple de modèle de données:



2 Création d'un domaine de valeurs

Créez un domaine en vous aidant des renseignements suivants.

Il n'est pas nécessaire d'avoir un schéma pour pouvoir créer un domaine de valeurs.

2.1 Le domaine

L'utilisation des domaines de valeurs (<<datatype>>) est nécessaire pour affecter à chaque attribut d'une classe UML le type exact de la colonne de la table qui sera générée. Un domaine de valeur est créé au niveau du compartiment Global Data Types. Pour créer un domaine, effectuez un clic droit sur Global Data Types, DataModeler/New/Domain Package.

2.2 Contrainte de type CHECK au niveau du domaine

On peut définir des contraintes de type CHECK au niveau du domaine de valeur. Ces contraintes seront ainsi communes à plusieurs classes disposant d'un attribut de même domaine. On peut aussi définir de telles contraintes directement au niveau d'un attribut de classe.
Pour définir une contrainte au niveau d'un domaine de valeur, il faut sélectionner le domaine (clic droit sur le domaine, Open Spécification) et l'écrire dans le langage SQL du SGBD cible.
Pour définir une contrainte au niveau d'un attribut du modèle de données, il faut le sélectionner dans le compartiment de gauche sous le répertoire des domaines puis faire un clic droit, option Open Specification et écrire la contrainte dans l'onglet associé.

3 Création d'éléments dans le schéma

Le schéma est composant.
Vous pouvez créer un schéma et un modèle de données (Voir la création du schéma MUTIN sur le document précédent). Le diagramme qui contiendra le modèle logique de données doit être créé dans le schéma associé à la base (clic droit dans le répertoire du schéma, option DataModeler/New/Data Model Diagram).
Dans le schéma, on insère des classes UML ayant le stéréotype «Table» (prédéfini dans Rose). On définit les attributs des classes puis on relie entre elles les classes par des associations.

3.1 Création d'une classe de stéréotype «Table»

Créez une classe de stéréotype «Table».

3.1.1 Visibilité des attributs

Avec le profil du Data Modeler, la visibilité des attributs a la signification suivante:

visibilité	signification
"-"	clé primaire
"#"	clé étrangère
"+"	champs sans contrainte
(rien)	champ clé primaire et clé étrangère

3.1.2 Identifiant d'une classe

La définition d'un identifiant par classe est fortement conseillé pour maîtriser les clés primaires dans le cycle de développement (traduction sous SQL2 de contraintes de type PRIMARY KEY).
Pour définir une clé primaire dans une classe de stéréotype «Table», il faut faire un clic droit sur la classe, Open Spécification, onglet Columns, puis sélectionner l'attribut en côchant la case Primary Key.

3.1.3 Une contrainte de type CHECK

On peut également déclarer des contraintes qui se traduiront en contraintes SQL2 de type UNIQUE, NOT NULL ou CHECK.

3.2 Associations

Une fois que toutes les classes de stéréotype «Table» sont créées avec leurs contraintes, il faut les relier entre elles à l'aide d'associations. L'outil propose deux types d'associations : Identifying et Non-Identifying. Le premier cas correspond à des associations fortes, le second à des associations plus faibles.
On crée une association via le menu Tools/Create/Identifying Relationship ou Non-Identifying Relationship.

- Une association Identifying spécifie qu'un enregistrement d'une table fils ne peut pas exister sans un enregistrement père associé. L'équivalent dans le modèle UML est l'association d'agrégation.
- Une association Non-Identifying ne spécifie rien de particulier entre la table fils et la table père. L'équivalent dans le modèle UML est l'association simple.

Il est important de noter que la création des clés étrangères est automatique. Il faut, lors de la création d'une association, connecter la table père à la table fils et non l'inverse.
Les associations Identifying permettent de modéliser les associations N-N, les associations n-aires ou les compositions.

Les cas particuliers concernent :

- les associations réflexives qui se traitent pareillement aux associations binaires ;
- les associations qui relient une classe d'intersection. Une classe d'intersection (ou d'association) de Rose serait par exemple la classe générée par ce que nous appelons, au niveau conceptuel, une association N-N ou n-aire. Nous étudierons ce type d'associations;
- les associations d'héritage .

3.2.1 Création d'une association «Non-Identifying»

Créez une association «Non-Identifying».

3.2.2 Création d'une association «Identifying»

Créez une association «Identifying».
Comme nous l'avons dit, cette option du profil UML permet de modéliser les associations N-N et n-aires du modèle conceptuel ou des compositions UML.

3.3 Création d'un déclencheur (trigger)

Créez un trigger avec le mode opératoire suivant.

On peut déclarer des déclencheurs et même d'écrire le code source sur chaque classe de stéréotype «Table». Pour ce faire, il faut effectuer un clic droit sur la classe, Open Spécification, onglet Triggers, puis définir ses caractéristiques comme la nature du déclencheur, de l'événement déclencheur, son code et des commentaires.
Si la base de données cible est de type Oracle, il faudra insérer du code PL/SQL.

3.4 Création d'une vue

Créez une vue avec le mode opératoire suivant.

La vue est créée par un clic droit sur le schéma puis sur l'option DataModeler/New/View Effectuez un clic droit sur la vue, puis choisissez l'option <Open Spécification. Plusieurs onglets sont proposés.

- Le premier permet de nommer la vue et de la rendre modifiable.
- Le deuxième onglet (From) permet de sélectionner la ou les table(s) source(s) de la vue et de définir la condition de la requête SQL de définition.

4 Génération du script SQL2

(Transformation B)
Une fois le schéma créé, on peut générer le script SQL2. Il faut sélectionner le schéma dans le compartiment de gauche puis faire un clic droit en choisissant DataModeler/Forward Engineer. Un assistant est alors lancé. Vous devrez y sélectionner différentes options (chemin et nom du script SQL à générer, création de commentaires, de tablespaces, de directives DROP, etc.). Il est en outre possible d'exécuter directement le script via une connexion au SGBD.

Dans les scripts SQL2 générés, on retrouve les contraintes avec des noms générés par Rose.

4.1 Exemple de SQL pour une association Non-Identifying

L'association 1-N entre Compagnie et Avion est de type Non-Identifying Relationship et se traduit par l'apparition de la clé étrangère dans Avion (en gras dans le script).

1 CREATE TABLE Compagnie (

,s comp VARCHAR2 (5) MOT NULL,

```
 i; nomComp VARCHAR2 (30),
| CQNSTRAINT PK_Compagniel PRIMARY KEY (comp) )
1 /

 : CREATE TABLE Avion (

 ,; immat VARCHAR2 ( 6 ) NOT NULL,
f typeAvion VARCHAR2 ( 10 ) NOT NULL,
'•' comp VARCHAR2 ( 5 ) NOT NULL,
y CONSTRAINT PK_AvionO PRIMARY KEY (immat),
i CONSTRAINT TC_Avionll CHECK (typeAvion IN ('A319', •A3201, •A321')) )
' /
| ALTER TABLE Avion ADD ( CONSTRAINT FK_Avion2 FOREIGN KEY (comp)

 —.REFERENCES Compagnie (comp))
! /
t CREATE TRIGGER TRIG_AvionO AFTER INSERT OM Avion
:! FOR EACH ROW
1|| -- code PL/SQL à insérer là
1 /
```

4.2 Exemple de SQL pour une association Identifying

L'association N-N entre Compagnie et Avion se traduit par l'apparition de la troisième table Affréter ayant sa clé primaire composée et deux clés étrangères (en gras dans le script).

```
; CREATE TABLE Compagnie (
"" comp VARCHAR2 ( 5 ) NOT NULL,
; nomComp VARCHAR2 ( 30 ),
f CONSTRAINT PK_Compagniel PRIMARY KEY (comp) )
•li /
| CREATE TABLE Affréter (
! datèVol DATE NOT NULL,
```