TD N°4 : Merise 2 + SQL

Thème : modéliser, requêter, modéliser, requêter…

Exercice 1 : select, select, select et toujours select

Sujet :

Soit le schéma relationnel suivant :
ARTICLES (NOART, LIBELLE, STOCK, PRIXINVENT)
FOURNISSEURS (NOFOUR, NOMFOUR, ADRFOUR, VILLEFOUR)
ACHETER (NOFOUR#, NOART#, PRIXACHAT, DELAI)

Il vous est demandé de rédiger les requêtes SQL correspondant aux questions qui suivent. Leur difficulté est *grosso modo* croissante.

Questions :

1. Numéros et libellés des articles dont le stock est inférieur à 10 ?

```
SELECT NOART, LIBELLE
FROM ARTICLES
WHERE STOCK<10;
```

2. Liste des articles dont le prix d'inventaire est compris entre 100 et 300 ?

```
SELECT *
FROM ARTICLES
WHERE PRIXINVENT BETWEEN 100 AND 300;
```

3. Liste des fournisseurs dont on ne connaît pas l'adresse ?

```
SELECT *
FROM FOURNISSEURS
WHERE ADRFOUR IS NULL;
```

4. Liste des fournisseurs dont le nom commence par "STE" ?

```
SELECT *
FROM FOURNISSEURS
WHERE NOMFOUR LIKE 'STE%';
```

5. Noms et adresses des fournisseurs qui proposent des articles pour lesquels le délai d'approvisionnement est supérieur à 20 jours ?

```
SELECT DISTINCT NOMFOUR, ADRFOUR, VILLEFOUR
FROM FOURNISSEURS AS F
INNER JOIN ACHETER AS A ON F.NOFOUR=A.NOFOUR
WHERE DELAI>20;
```

6. Nombre d'articles référencés ?

```
SELECT COUNT(*) AS NbArticles
FROM ARTICLES;
```

7. Valeur du stock ?

```
SELECT SUM(STOCK*PRIXINVENT) AS ValeurStock
FROM ARTICLES;
```

8. Numéros et libellés des articles triés dans l'ordre décroissant des stocks ?

```
SELECT NOART, LIBELLE, STOCK
FROM ARTICLES
ORDER BY STOCK DESC;
```

9. Liste pour chaque article (numéro et libellé) du prix d'achat maximum, minimum et moyen ?

```
SELECT A.NOART, LIBELLE, MAX(PRIXACHAT) AS PMAX, MIN(PRIXACHAT) AS PMIN, AVG(PRIXACHAT) AS PMOY
FROM ACHETER AS A
INNER JOIN ARTICLES B ON A.NOART = B.NOART
GROUP BY A.NOART, LIBELLE;
```

10. Délai moyen pour chaque fournisseur proposant au moins 2 articles ?

```
SELECT A.NOFOUR, NOMFOUR, AVG(DELAI) AS DelaiMoyen
FROM ACHETER AS A
INNER JOIN FOURNISSEURS AS F
ON A.NOFOUR = F.NOFOUR
GROUP BY A.NOFOUR, NOMFOUR
HAVING COUNT(*) >=2;
```

11. Prix minimum de chaque article avec le fournisseur correspondant

```
SELECT A.NOART, A.LBELLE, A1.PRIXACHAT, F.NOMFOUR
FROM ACHETER AS A1
INNER JOIN ARTICLES AS A ON A1.NOART = A.NOART
INNER JOIN FOURNISSEURS AS F ON A1.NOFOUR = F.NOFOUR
WHERE A1.PRIXACHAT = (
        SELECT min(A2.PRIXACHAT)
        FROM ACHETER AS A2
        WHERE A2.NOART = A.NOART
) ;
```

12. Listes des articles pouvant être achetés chez plusieurs fournisseurs, avec le prix d'achat et le délai correspondants

```
SELECT AR.NOART, AR.LIBELLE
FROM ARTICLES AS AR
INNER JOIN ACHETER AS AC
GROUP BY AR.NOART, AR.LIBELLE
HAVING count(AR.NOART) > 1
```

13. Quels sont le ou les fournisseurs qui fournissent le plus de produits ?

```
SELECT NOFOUR, NOMFOUR, count(*) AS nombre_produits
FROM FOURNISSEUR AS F, ACHETER AS A
WHERE  F.NOFOUR = A.NOFOUR
GROUP BY NOFOUR, NOMFOUR
HAVING count(*)=(
        SELECT MAX(nb_prod) FROM (
                SELECT count(*) AS nb_prod
                FROM ACHETER
                GROUP BY NOFOUR
        ) AS B
);
```

Exercice 2 : parc immobilier

Sujet :

- Le parc immobilier de la SCI (Société Civile Immobilière) EasyLocation est constitué d'appartements faisant partie d'immeubles situé à une adresse (n°, voie, code postal, ville) ;
- L'immeuble dispose ou ne dispose pas de la fibre optique, il dispose ou non d'un parking privatif ;
- L'appartement a une valeur locative (loyer mensuel), il a une superficie totale, il a une description, il possède ou non une terrasse, il a une classe de consommation d'énergie (« A », « B », « C », …), il est chauffé à l'électricité (« E ») ou au gaz (« G »), il a ou n'a pas de place de place de parking, s'il a une place de parking, son prix est indiqué ;
- L'appartement est constitué d'une à plusieurs pièces ayant une superficie et une fonction (exemple : « salle d'eau », « cuisine », etc.) ;
- A l'appartement peuvent être attachées des photos.

Questions :

1. Etablir le MCD correspondant à la situation décrite (voir ci-après).

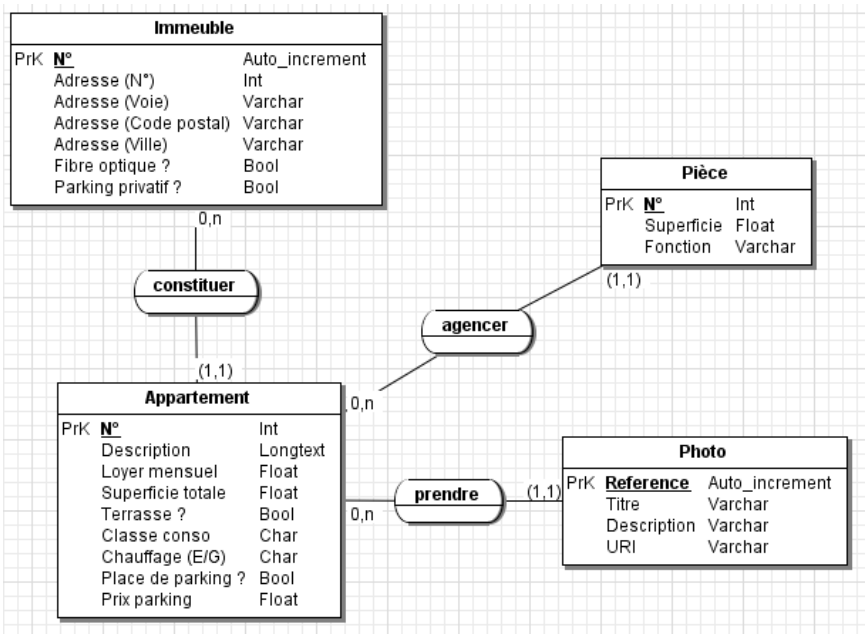
2. Etablir le MLD correspondant à la situation décrite.

Immeuble(id, adrNum, adrVoie, adrCodePostal, adrVille, fibreOptique, parkingPrivatif)
Clef primaire : id

Appartement(#immeuble, num, description, loyer, superficie, terrasse, classeConso, chauffage, placeParking, prixParking)
Clef primaire : immeuble, num
Clef étrangère : immeuble en référence à Immeuble(id)

Piece(#(immeuble, appartement), num, superficie, fonction)
Clef primaire : immeuble, appartement, num
Clefs étrangères : (immeuble, appartement) en référence à Appartement(immeuble, num)

Photo(num, titre, description, uri, #(immeuble, appartement))
Clef primaire : num
Clef étrangère : (immeuble, appartement) en référence à Appartement(immeuble, num)



3. Préciser les contraintes qui ne figurent pas sur le MCD ?

Les contraintes suivantes ne figurent pas sur le MCD :

- Le prix de la place de parking d'un appartement peut et doit être NULL si l'appartement ne possède pas de place de parking ;
- Un appartement ne peut avoir de place de parking si l'immeuble n'a pas de parking privatif ;
- La superficie totale d'un appartement doit être égale à la somme de la superficie de chacune de ses pièces.

4. Rédiger le script SQL de création des tables correspondant à votre MCD et/ou MLD.

```
CREATE TABLE Immeuble(
    Id INT(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    adrNum VARCHAR(7) NOT NULL, adrVoie VARCHAR(100) NOT NULL,
    adrCodePostal VARCHAR(5) NOT NULL,  adrVille VARCHAR(30) NOT NULL,
    fibreOptique TINYINT  NOT NULL,  parkingPrivatif TINYINT  NOT NULL
);
```

```
CREATE TABLE Appartement(
    immeuble INT(11), num INT(3) NOT NULL, description LONGTEXT,
    loyer DECIMAL(5,2) NOT NULL, superficie DECIMAL(3,2) NOT NULL,
    terrasse TINYINT NOT NULL, classeConso CHAR(1) NOT NULL, chauffage CHAR(1) NOT NULL,
    placeParking TINYINT NOT NULL, prixParking DECIMAL(3,2),
    CONSTRAINT pk_appartement PRIMARY KEY (immeuble, num),
    CONSTRAINT fk_immeuble FOREIGN KEY (immeuble) REFERENCES Immeuble(id)
);

CREATE TABLE Photo(
    immeuble INT(11), appartement INT(3), reference INT(11) NOT NULL,
    titre VARCHAR(75), description VARCHAR(255), uri VARCHAR(120) NOT NULL,
    CONSTRAINT pk_photo PRIMARY KEY (immeuble, appartement, reference),
    CONSTRAINT fk_appartement_photo
        FOREIGN KEY (immeuble, appartement) REFERENCES Appartement(immeuble, num)
);

CREATE TABLE Piece(
    immeuble INT(11), appartement INT(3), num INT(2) NOT NULL,
    superficie DECIMAL(3, 1), fonction VARCHAR(30),
    CONSTRAINT pk_piece PRIMARY KEY (immeuble, appartement, num),
    CONSTRAINT fk_appartement_piece
        FOREIGN KEY (immeuble, appartement) REFERENCES Appartement(immeuble, num)
);
```

5. Rédiger la requête SQL qui calcule la superficie totale de chacun des appartements à partir de celle de ses pièces.

```
SELECT immeuble, appartement, SUM(superficie)
FROM Piece
GROUP BY immeuble, appartement
```

6. Quelle notion évoquée en cours vous permettrait de vérifier les contraintes pour le moment non vérifiées ? Quelles serait les opérations de vérification à effectuer et à quelle(s) moment(s) ces vérifications interviendraient-elles ?

Un trigger nous permettrait de vérifier les contraintes pour le moment non vérifiées, à savoir celles listées en réponse à la question 3 (voir question **3.**).

Les vérifications à effectuer sont les suivantes :

- A l'insertion d'un appartement n'ayant aucune place de parking (placeParking vaut 0), il faut vérifier que le prix du parking est bien nul (prixParking vaut NULL). Si le prix n'est pas nul, il faut empêcher l'insertion ;
- A la mise à jour d'un appartement, s'il n'y a plus de place de parking (placeParking passe à 0), il faut forcer le passage du prix du parking à nul ;
- Toujours à la mise à jour d'un appartement, on doit empêcher le passage du prix à nul si l'appartement possède une place de parking ;
- Finalement, il importe de vérifier que la superficie totale de l'appartement soit bien égale à la somme de celle de ses pièces. A l'insertion comme à la mise à jour d'une pièce, on doit recalculer

et mettre à jour la superficie totale de l'appartement. A la mise à jour, on vérifiera que la superficie totale de l'appartement est bien égale à la somme de celle de ses pièces. Si tel est le cas tout va bien. Dans le cas contraire, il faut empêcher la mise à jour.

- Lors de l'insertion d'un appartement, ce dernier ne comporte encore aucune pièce. De ce fait, sa superficie doit pour le moment être égale à 0. Il convient de forcer cette valeur, i.e. de passer automatiquement la superficie à 0.

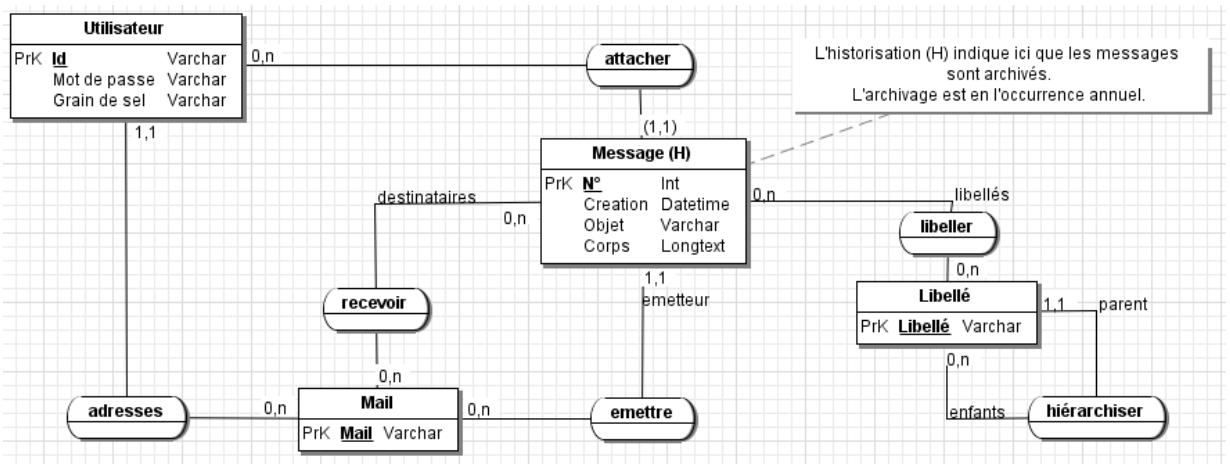
Exercice 3 : archivage de mails

Sujet :

- Une *webmail* (alias une messagerie web ou encore boîte de messagerie) permet de conserver les mails reçus et envoyés. Elle est propre à un utilisateur ;
- A un mail, il est possible d'affecter un ou plusieurs libellés ;
- Les libellés peuvent être hiérarchisés ;
- Les anciens mails de la *webmail* finissent par être archivés (exemple : une fois par an).

Questions :

1. Proposer un MCD représentant la situation décrite.



2. Etablir le MLD correspondant à votre MCD.

Mail(mail)
Clef primaire : mail

Utilisateur(id, motDePasse, grainDeSel, #mail)
Clef primaire : id
Clef étrangère : mail en référence à Mail

Message(#utilisateur, num, creation, objet, corps, #emetteur)
Clef primaire : utilisateur, num
Clef étrangère : utilisateur en référence à Utilisateur

MessageDestinataire(#(utilisateur, num), #destinataire)

Clef primaire : utilisateur, num, destinataire
Clefs étrangères : (utilisateur, num) en référence à Message, destinateur en référence à Mail

MessageLibelle(#(utilisateur, num), #libelle)
Clef primaire : utilisateur, num, libelle
Clefs étrangères : (utilisateur, num) en référence à Message, libelle en référence à Libelle

Libelle(libelle, #parent)
Clef primaire : libelle
Clef étrangère : parent en référence à Libelle

MessageArchive(#utilisateur, num, creation, objet, corps, #emetteur)
Clef primaire : utilisateur, num
Clef étrangère : utilisateur en référence à Utilisateur

N.B. : on notera qu'en pratique on se passera très volontiers de la table Mail.

3. Rédiger la requête SQL qui réponde à la question suivante : combien de mail ont été envoyés par mois, sur l'année 2016, par l'utilisateur dont l'adresse mail est john.doe@gmal.com ?

```
SELECT month(creation) AS 'Mois', count(*) AS 'Nombre de mails'
FROM Message AS M
INNER JOIN Utilisateur AS U ON M.utilisateur = U.id
WHERE U.mail = 'john.doe@gmal.com'
AND year(creation) = 2016
GROUP BY month(creation)
```

Exercice 4 : des produits au prix variant au fil du temps

Sujet :

- La société Wissenschaft est spécialisée dans la formation en langue en ligne. Elle commercialise ses produits au moyen d'un site e-commerce/e-learning. Ces produits sont des biens matériels (livres) ou des biens immatériels, à savoir des prestations (formations en ligne) ;
- Tous les produits ont un prix mais seuls les biens matériels ont un poids et font l'objet d'une livraison ;
- Une commande auprès de Wissenschaft porte soit sur une prestation soit sur un ou plusieurs biens matériels. On notera que le prix des produits de la société varie au fil du temps ;
- Une commande peut ou non avoir un mode de livraison. De même, la commande intègre ou non des frais de port.

Questions :

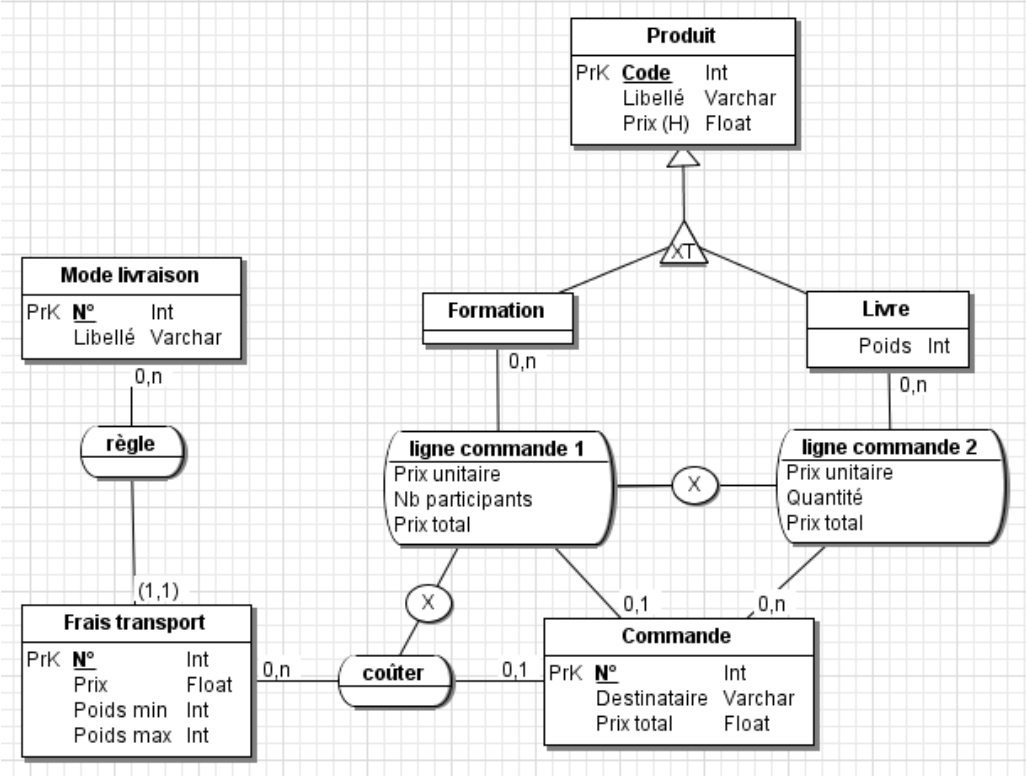
1. A votre avis, dans quel cas la commande n'a ni mode de livraison ni frais de port ?

La délivrance de prestations de formation ne requiert pas de livraison physique. De ce fait, une commande de formation ne nécessite ni mode de livraison ni frais port.

2. Le prix figurant sur une commande varient-ils dans le temps ?

Si le prix d'un produit est susceptible de varier dans le temps, une commande et le bon de commande associé sont des éléments commerciaux établit une fois pour toute. Pour chaque produit commandé (i.e. pour chaque ligne de commande), le prix appliqué est celui valable au jour de la commande.
En conséquence, une fois une commande validée, les prix sont figés et ne varient pas dans le temps.

3. Proposer un MCD représentant la situation décrite.



Justifications :
- on a bien précisé que le prix des produits est historisé : **(H)**. Par conséquent, pour retrouver simplement et efficacement le prix du produit valable lors d'une commande déjà passée, on a utilisé les propriétés portées « prix unitaire » ;
- pour retrouver simplement et efficacement les informations tarifaires propres à une commande, on a choisi de recourir aux champs calculés « prix total ». Ils ne sont pas nécessaires mais permettent de retrouver les données tarifaires sans passer par des agrégats, les agrégats devant être calculés à la création (insert) et recalculés lors d'une éventuelle modification (update) ;
- un produit est soit une formation, soit un livre. Dès lors, la spécialisation est bien de type « XT » : exclusivité + totalité ;
- on commande soit une formation soit un ou plusieurs livres, d'où la contrainte d'exclusion entre les associations « ligne de commande » ;
- si on commande une formation, il n'y a pas de frais de transport, d'où la seconde contrainte d'exclusion ;
- on a choisi que les frais de transport soient relatifs à un mode de livraison de sorte qu'on connaît le mode de livraison d'une commande par l'intermédiaire de ses frais de port ;
- on a très naturellement choisi que les frais de transport dépendent du poids total de la commande. Ainsi, une fois les produits et les quantités commandées choisis, on peut calculer le poids de la commande. Puis, une fois le mode de livraison choisi, on peut appliquer les frais de transport, à savoir les frais de port du mode de livraison choisi tels que $poids\ min \leq poids\ commande < poids$.

4. Justifier la présence d'éventuelles contraintes d'associations ? Pour chacune d'entre elles, expliquer quelles sont les vérifications à effectuer afin de préserver l'intégrité des données ? Préciser à quel(s) moment(s) ces vérifications doivent être effectuées.

> Voir correction de la question **3**.

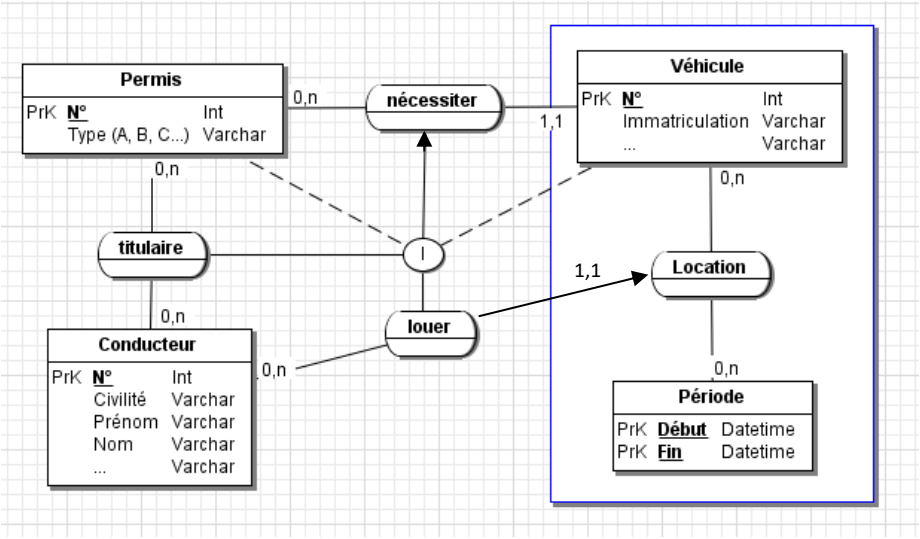
Exercice 5 : des produits au prix variant au fil du temps

Sujet :

- Un véhicule est loué pour une période donnée ;
- La location est effectuée par une personne disposant d'un permis (exemple : permis poids-lourd) ;
- Le véhicule requiert lui-même un permis particulier.

Questions :

1. Proposer un MCD représentant la situation décrite.



2. Justifier la présence d'une éventuelle contrainte d'association ? De quel type de contrainte d'association s'agit-il ?

> Une contrainte d'inclusion est nécessaire pour modéliser la contrainte d'intégrité suivante : un conducteur ne peut louer un véhicule que s'il est titulaire du permis requis.

3. Quelles sont les opérations à effectuer afin que la contrainte d'association soit vérifiée ? A quel(s) moment(s) ces vérifications doivent-elles intervenir ?

> Il convient de mettre en place un *trigger* :
> - à l'insertion et à la modification d'une Location (INSERT ou UPDATE sur la table « Location ») ;
> - qui vérifiera que le conducteur spécifié correspond à un conducteur ayant parmi ses permis celui requis pour conduire le véhicule ;

- autrement dit, le résultat retourné par la requête suivant doit être égal à 1 :

```
SELECT count(*)
FROM Location
INNER JOIN Titulaire ON Inserted.Conducteur = Titulaire.Conducteur
INNER JOIN Véhicule ON Inserted.Vehicule = Vehicule.Numero
WHERE Titulaire.Permis = Vehicule.Permis
```

N.B. : où Inserted correspond à la ligne insérée.

- si le résultat de la requête n'est pas égale à 1 (i.e. la vérification a échoué), l'insertion doit être annulée (*rollback*).