

Base de données :

**Sécurisation des mises à jour d'un bloc de requêtes
avec le mécanisme de transaction**

Transaction: principe

Il est fréquent de devoir grouper des mises à jour de tables dans une même séquence logique.

Dans cette séquence logique, chaque mise à jour est liée aux autres de telle sorte qu'une exécution partielle de ces mises à jour entraînerait une incohérence des données dans la base de données.

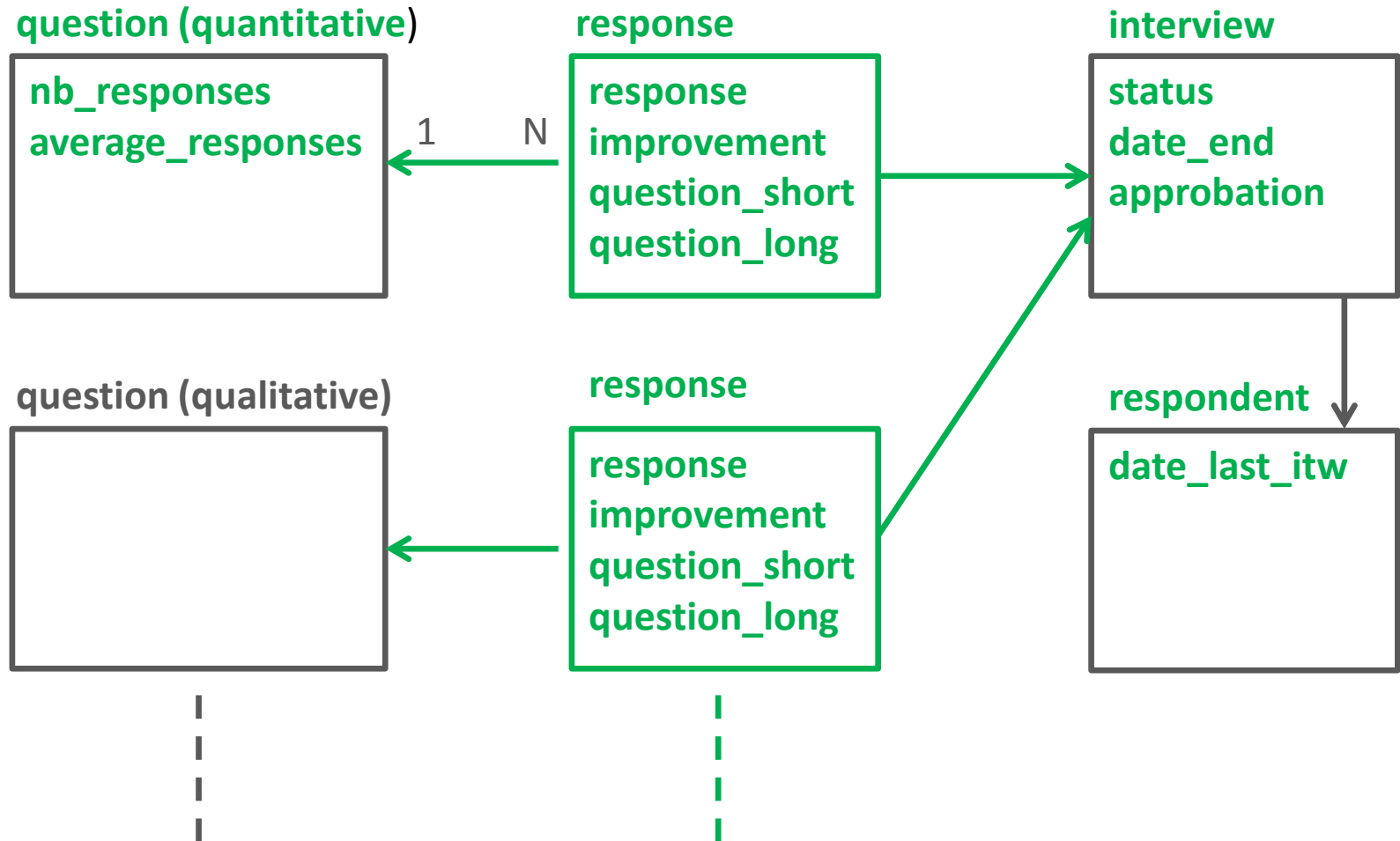
Pour garantir qu'une exécution de cette séquence ne sera jamais partielle, il faut utiliser le mécanisme de transaction de MySQL.

En exécutant cette séquence dans le cadre d'une transaction :

- S'il n'y a aucune erreur à l'issue de toutes les mises à jour, ces mises à jour seront définitivement enregistrées dans les tables.
- À la première erreur rencontrée sur l'une des mises à jour, la séquence est interrompue et les tables se retrouvent dans l'état initial avant l'exécution de la séquence.

Transaction: exemple, transaction d'enregistrement d'une interview

Les mises à jour sont en vert



Transaction: exemple, transaction d'enregistrement d'une interview

commandes SQL

InnoDB uniquement (MyISAM exclu) <https://dev.mysql.com/doc/refman/8.0/en/mysql-acid.html>

START TRANSACTION;

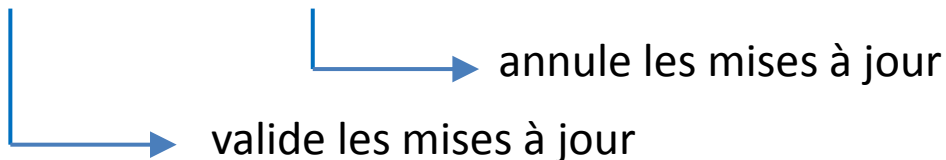
```
INSERT INTO response SET id_question = ... ;  
INSERT INTO response SET id_question = ... ;  
...
```

```
UPDATE question SET nb_responses = nb_responses+1, ... ;
```

```
UPDATE interview SET status = "COMPLETED", ... ;
```

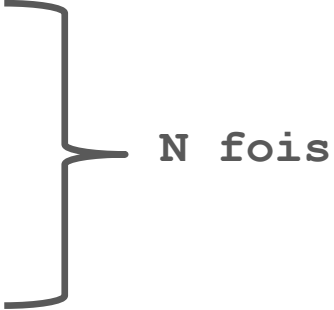
```
UPDATE respondent SET date_last_itw = ... ;
```

COMMIT; ou **ROLLBACK;**



Transaction: exemple, transaction d'enregistrement d'une interview commandes SQL via la programmation PHP/MySQLi

```
try {  
    $oConn->begin_transaction();  
  
    $oStmt = $oConn->prepare(  
        "INSERT/UPDATE/DELETE...");  
    $oStmt->bind_param (...);  
    $oStmt->execute();  
  
    // aucune erreur  
    $oConn->commit();  
  
} catch (Exception $e) { // traitement d'erreur  
    $oConn->rollBack();  
    ...  
}
```



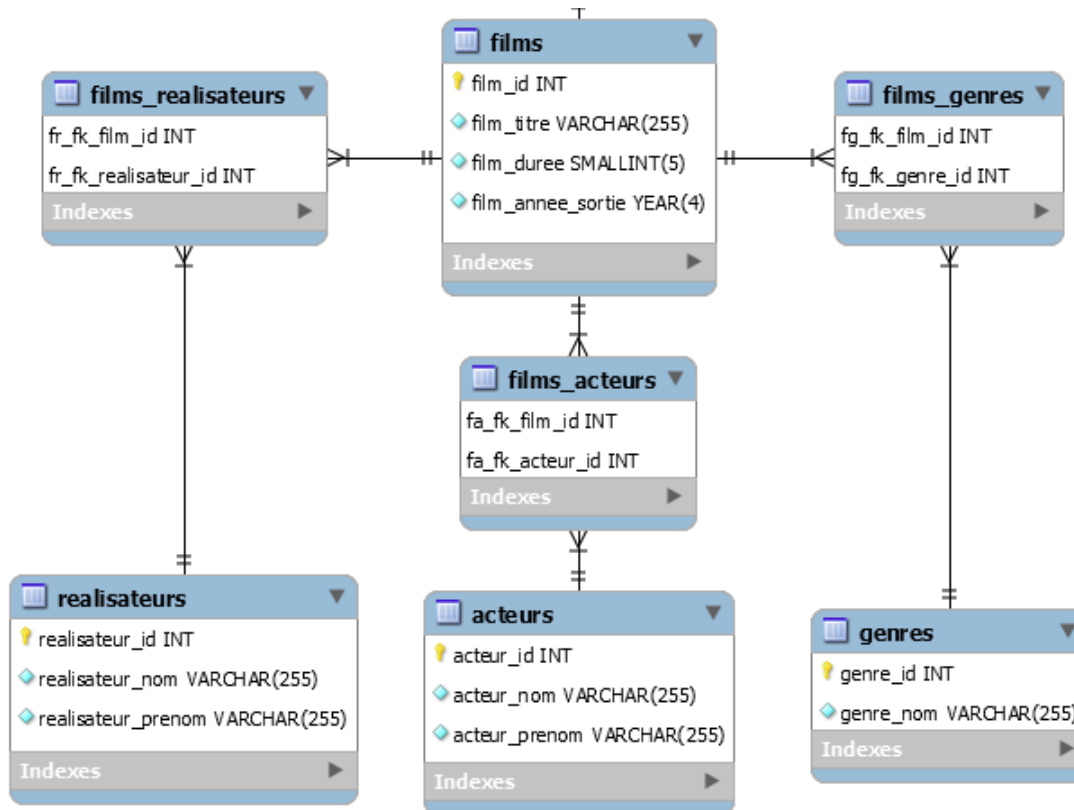
N fois

Transaction: pratique, enregistrement d'un film dans la base "club_video"

Enregistrement du film "Les mauvaises herbes" du réalisateur Louis Bélanger.

1- Ajouter au préalable le réalisateur et les acteurs principaux dans les tables :

```
INSERT INTO realisateurs SET realisateur_nom="Bélanger", realisateur_prenom="Louis";  
INSERT INTO acteurs      SET acteur_nom="Martin",  acteur_prenom="Alexis";  
INSERT INTO acteurs      SET acteur_nom="Renaud",  acteur_prenom="Gilles";
```



Transaction: pratique, enregistrement d'un film dans la base "club_video"

Enregistrement du film "Les mauvaises herbes" du réalisateur Louis Bélanger.

2- Créer la requête de consultation du films et de toute ses caractéristiques :

```
SELECT
  CONCAT (U.film_id, " - ", U.film_titre), U.film_annee_sortie, U.film_duree,
  CONCAT (R.realisateur_id, " - ", R.realisateur_nom, " ", R.realisateur_prenom) AS réalisateur,
  CONCAT(A.acteur_id, " - ", A.acteur_nom, " ", A.acteur_prenom) AS acteur,
  G.genre_nom
FROM films as U
INNER JOIN films_realisateurs AS FR ON FR.fr_fk_film_id = U.film_id
INNER JOIN realisateurs      AS R  ON R.realisateur_id = FR.fr_fk_realisateur_id
INNER JOIN films_acteurs     AS FA ON FA.fa_fk_film_id = U.film_id
INNER JOIN acteurs           AS A  ON A.acteur_id      = FA.fa_fk_acteur_id
INNER JOIN films_genres      AS FG ON FG.fg_fk_film_id = U.film_id
INNER JOIN genres            AS G  ON G.genre_id       = FG.fg_fk_genre_id
WHERE U.film_titre="Les mauvaises herbes";
```

Cette requête sera utilisée dans les points suivants pour vérifier si les mises à jour sont effectives ou pas.

Transaction: pratique, enregistrement d'un film dans la base "club_video"

Enregistrement du film "Les mauvaises herbes" du réalisateur Louis Bélanger.

3- Exécuter la transaction dans une session MySQL sans faire de COMMIT

```
USE club_video;

START TRANSACTION;

SET @realisateur_id = (SELECT realisateur_id FROM realisateurs
                        WHERE realisateur_nom="Bélanger" AND realisateur_prenom="Louis");
SET @acteur_id1      = (SELECT acteur_id FROM acteurs
                        WHERE acteur_nom="Martin" AND acteur_prenom="Alexis");
SET @acteur_id2      = (SELECT acteur_id FROM acteurs
                        WHERE acteur_nom="Renaud" AND acteur_prenom="Gilles");

SELECT @genre_id := genre_id FROM genres WHERE genre_nom="Comédie dramatique";

INSERT INTO films SET film_titre="Les mauvaises herbes", film_annee_sortie=2016, film_duree=108;

SET @film_id = LAST_INSERT_ID();

INSERT INTO films_realisateurs SET fr_fk_film_id=@film_id, fr_fk_realisateur_id=@realisateur_id;
INSERT INTO films_acteurs VALUES(@film_id, @acteur_id1), (@film_id, @acteur_id2);
INSERT INTO films_genres SET fg_fk_film_id=@film_id, fg_fk_genre_id=@genre_id;
```


Transaction: pratique, enregistrement d'un film dans la base "club_video"

Enregistrement du film "Les mauvaises herbes" du réalisateur Louis Bélanger.

3- Exécuter la transaction dans une session MySQL sans faire de COMMIT (suite)

Si vous exécutez la requête de consultation créée au point 2, vous visualisez le film avec toute ses caractéristiques, mais les données ne sont pas enregistrées dans la base de données, elles sont présentes uniquement dans un espace mémoire temporaire.

Si vous ouvrez une autre session MySQL avec une autre console MySQL , la même requête de consultation ne retourne aucune donnée.

Si vous déconnectez la session de la transaction puis vous reconnectez, la requête de consultation ne retourne alors aucune donnée, c'est comme si vous aviez effectué un ROLLBACK explicite.

Transaction: pratique, enregistrement d'un film dans la base "club_video"

Enregistrement du film "Les mauvaises herbes" du réalisateur Louis Bélanger.

4- Exécuter la transaction dans une session MySQL en terminant par un ROLLBACK

Ajoutez la commande ROLLBACK au script du point 3 et exécutez ce script.

Dans ce cas les requêtes de la transaction sont définitivement annulées.

Si vous exécutez ensuite la requête de consultation créée au point 2, cette requête ne retourne aucune donnée.

Si vous ouvrez une autre session MySQL avec une autre console MySQL , la même requête de consultation ne retourne aucune donnée.

Transaction: pratique, enregistrement d'un film dans la base "club_video"

Enregistrement du film "Les mauvaises herbes" du réalisateur Louis Bélanger.

5- Exécuter la transaction dans une session MySQL en terminant par un COMMIT

Ajoutez la commande COMMIT au script du point 3 et exécutez ce script.

Dans ce cas les requêtes de la transaction sont définitivement enregistrées dans la base de données.

Si vous exécutez ensuite la requête de consultation créée au point 2, cette requête visualise le film avec toutes ses caractéristiques.

Si vous ouvrez une autre session MySQL avec une autre console MySQL, la même requête de consultation visualise également le film avec toutes ses caractéristiques.

Transaction: utilisation d'un verrou de ligne

Il est possible de verrouiller une ligne d'une table, avant d'enchaîner une séquence de requêtes de mises à jour dont la mise à jour éventuelle de cette ligne.

On utilise pour cela la clause FOR UPDATE dans la commande SELECT de cette ligne (ou de plusieurs lignes si besoin).

```
START TRANSACTION;
```

```
-- Avec FOR UPDATE la SELECT est en attente  
-- de la fin d'une transaction similaire en cours
```

```
SELECT ... FROM ... WHERE ... FOR UPDATE;
```

```
INSERT ... ;
```

```
...
```

```
UPDATE ... ;
```

```
...
```

```
DELETE ...;
```

```
...
```

```
COMMIT;
```