Module : Bases de données

Les vues en SQL

Pr. M. RAHMOUNI
md.rahmouni@yahoo.fr

Les vues

- Les vues en deux mots : des tables virtuelles
- Les vues en une phrase : une vue est une table qui est le résultat d'une requête (**SELECT**) à laquelle on a donné un nom
- Le nom d'une vue peut être utilisé partout où on peut mettre le nom d'une table : **SELECT**, **UPDATE**, **DELETE**, **INSERT**, **GRANT**

Création d'une vue : syntaxe

CREATE [OR REPLACE]
[FORCE | NOFORCE] VIEW
nom-de-vue **[(*attr*$_1$, …, *attr*$_n$)]**
AS *requête*
[WITH CHECK OPTION
[CONSTRAINT *nom-contrainte*]]
[WITH READ ONLY]

3

Utilisation d'une vue : comme si elle était une table

SELECT ...
FROM *nom-de-vue*
WHERE ...

4

Suppression d'une vue

DROP VIEW *nom-de-vue*

- La suppression d'une vue n'entraîne pas la suppression des données
- Les vues figurent dans les tables systèmes **ALL_CATALOG**, **USER_VIEWS** et **ALL_VIEWS**

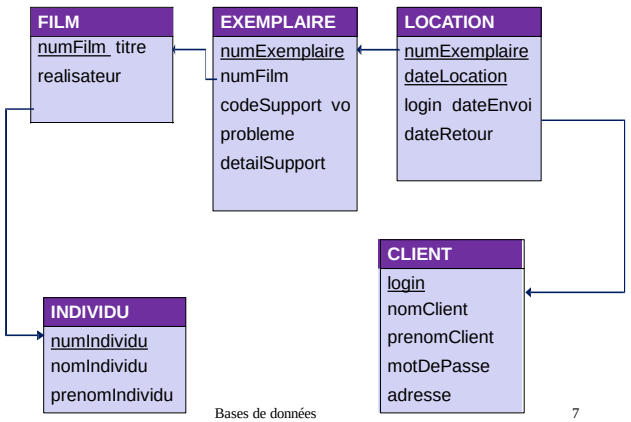
5

Renommer une vue

RENAME *ancien-nom* TO *nouveau-nom*

6

Exemple : vidéothèque



Exemple (2)

```
CREATE OR REPLACE VIEW
exemplairePlus (num, vo, titre, real,
support)  AS SELECT numExemplaire,
vo, titre,  nomIndividu, codesupport
FROM Exemplaire E, Film F, Individu
WHERE E.numFilm = F.numFilm  AND
realisateur = numIndividu
AND probleme IS NULL;
```

Exemple (3)

```
SELECT num, titre,
dateLocation, login  FROM
exemplairePlus, Location
WHERE num = numExemplaire
AND real = 'SAUTET'
AND dateRetour IS NULL;
```

Exemple (4)

```
INSERT INTO exemplairePlus (num,
support)  VALUES (150346, 'DVD');
```

```
DROP VIEW exemplairePlus;
```

Quatre raisons d'utiliser des vues

1) Effet macro : remplacer une requête compliquée par des requêtes plus simples
2) Confidentialité
3) Contraintes d'intégrité
4) Augmenter l'indépendance logique

1) Effet macro

- Remplacer une requête compliquée par des requêtes plus simples

2) Confidentialité : exemple

```
CREATE VIEW emprunteurRestreint
AS SELECT login, nomClient,
prenomClient  FROM client
```

3) Contraintes d'intégrité (CHECK OPTION) : exemple

```
CREATE VIEW anciensExemplaires
AS SELECT * FROM Exemplaire
WHERE numExemplaire < 2000
WITH CHECK OPTION;
UPDATE anciensExemplaires  SET
numExemplaire = 3812  WHERE
numExemplaire = 1318;
```

Sans **'WITH CHECK OPTION'**, c'est possible. Avec **'WITH CHECK OPTION'**, c'est impossible.

4) Augmenter l'indépendance logique

- Les applications utilisant les tables de la base ne doivent pas être modifiées si on change le schéma de la base

Conditions de mise à jour pour les vues

- Pour **UPDATE**, **DELETE**, **INSERT** la vue ne doit pas contenir :
  - Un opérateur ensembliste (**UNION**, **MINUS**, **INTERSECT**)
  - Un opérateur **DISTINCT**
  - Une fonction d'agrégation comme attribut
  - Une clause **GROUP BY**
  - Une jointure (la vue doit être construite sur une seule table)

Conditions de mise à jour pour les vues (2)

- Pour **UPDATE**, **DELETE**, **INSERT** :
  - Les colonnes résultats de l'ordre **SELECT** doivent être des colonnes réelles d'une table de la base et non des expressions
  - Si la vue est construite à partir d'une autre vue, cette dernière doit elle-même vérifier les conditions ci-dessus

Cas particulier d'Oracle

- Une table *T préserve la clé* dans une vue *V* si la clé primaire de la table *T* désigne une seule ligne de la vue (elle pourrait être une clé de la vue si celle-ci était une vraie table)
- On peut faire une mise à jour si les colonnes référencées dans la mise à jour **appartiennent toutes à la même table** et si cette table **préserve la clé** dans la vue

Cas particulier d'Oracle : exemple

```
CREATE VIEW locationBis AS SELECT
E.numExemplaire, nomClient, prenomClient,
dateEnvoi, dateRetour, numFilm
FROM Exemplaire E, Location L, Client C  WHERE
E.numExemplaire = L.numExemplaire  AND
L.login = C.login;
```

- La table **location** préserve la clé dans la vue **locationBis**
- Les tables **client** et **exemplaire** ne préservent pas la clé dans la vue **locationBis**

Cas particulier d'Oracle : exemple (2)

```
UPDATE locationBis
SET dateRetour = SYSDATE
WHERE nomClient = 'Martin'
AND numFilm = 1;
```

- Mise à jour possible

Cas particulier d'Oracle : exemple (3)

```
UPDATE locationBis  SET
numFilm = 4
WHERE nomClient = 'Duval';
```

- Mise à jour impossible

Cas particulier d'Oracle : exemple (4)

```
INSERT INTO locationBis
(  numExemplaire,
dateEnvoi,  nomClient,
prenomClient)
VALUES(4578, SYSDATE,
'FIORENZI',
'FRANCESCA');
```

- Insertion impossible

Rappels

- La définition de la vue est enregistrée dans la base de données, mais les lignes correspondant à la vue ne le sont pas
- A deux instants distincts, le « contenu » d'une vue peut changer (si le contenu des tables qui entrent dans la description de la vue a évolué). En effet, le contenu d'une vue est recalculé à chaque utilisation de la vue par SQL

Les séquences

- Générer des valeurs (numériques)
  - Par exemple de clés primaires
- Coordonner les valeurs de clés dans plusieurs lignes ou tables

Création : syntaxe

CREATE SEQUENCE *nom-séquence*
[INCREMENT BY (1 | *valeur*)]
[START WITH *valeur*]
[MAXVALUE *valeur* | NOMAXVALUE]
[MINVALUE *valeur* | NOMINVALUE]
[CYCLE | NOCYCLE]
[CACHE (*valeur* | 20) | NOCACHE]

Prégénération des valeurs (nombre des valeurs stockées en mémoire)

25

Suppression : syntaxe

DROP SEQUENCE *nom-séquence*

Exemple

```
CREATE SEQUENCE masequence
START WITH 1000
INCREMENT BY 30  NOMAXVALUE
NOCYCLE
```

Exemple (suite)

```
INSERT INTO film (numFilm, titre)
VALUES(masequence.NEXTVAL, 'Kill Bill')

INSERT INTO exemplaire (numExemplaire,
numFilm)
VALUES (290870, masequence.CURRVAL)
```