SQL

Marc Plantevit

marc.plantevit@liris.cnrs.fr

`EMP(EmpNo, Enom, Job, DirNo, Sal, Prime, DeptNo)`
`DEPT(DeptNo, Dnom, Adr)`
avec

- $EmpNo \rightarrow Enom, Job, DirNo, Sal, Prime, DeptNo$
- $DeptNo \rightarrow Dnom, Adr$
- $EMP(DeptNo) \subseteq DEPT(DeptNo)$

En SQL

```
CREATE TABLE DEPT (
Deptno NUMBER(3) PRIMARY KEY,
Dnom VARCHAR2(15),
Adr VARCHAR2(15));
```

```
CREATE TABLE EMP(
EmpNo NUMBER(5) PRIMARY KEY,
Enom VARCHAR2 (15) NOT NULL,
Job VARCHAR2(10),
DirNo NUMBER(5),
Sal NUMBER(7, 2),
Prime NUMBER(5,2),
DeptNo NUMBER(3) CONSTRAINT dept_cle_etr REFERENCES DEPT);
```

Clés Primaires

- Les mots clés `PRIMARY KEY` déclarent la clé primaire d'une table.
- La déclaration de la clé primaire peut être séparée (particulièrement adapté pour les clés composées de plusieurs attributs).

```
CREATE TABLE DEPT (
Deptno NUMBER(3),
Dnom VARCHAR2(15),
Adr VARCHAR2(15) PRIMARY KEY(Deptno));
```

Clés Uniques

- SQL d'Oracle permet une autre sorte de clés appelées clés uniques.
- Chaque table peut avoir une seule clé primaire, et **zéro ou plusieurs clés uniques**.
- Une clé unique impose la contrainte d'unicité de chaque tuple dans une table.
- Les différences entre la clé primaire et une clé unique consistent en deux points :
  - Une table peut avoir une seule clé primaire, mais peut avoir plusieurs clés uniques.
  - Un tuple peut avoir des valeurs NULL sur les attributs composants d'une clé unique tandis que ceci est interdit pour la clé primaire.
- On peut modifier la table DEPT pour ajouter une clé unique composée de deux attributs Dnom et Adr :
  `ALTER TABLE DEPT ADD CONSTRAINT dnom_cle_u UNIQUE(Dnom, Adr) ;`

Clés étrangères - Contraintes Référentielles

```
DeptNo NUMBER(3) CONSTRAINT dept_cle_etr REFERENCES DEPT
```

- Définit une contrainte référentielle.
- DEPT est appelée table mère (maître, ou référencée), et la table EMP est appelée table fille (esclave, ou dépendante).
- L'ensemble des attributs référencées doit être la clé primaire ou une clé unique de la table référencée.
- On peut expliciter les attributs référencés dans la définition : `DeptNo NUMBER(3) CONSTRAINT dept_cle_etr REFERENCES DEPT(DeptNo)`
- La forme explicite est nécessaire lorsque les attributs référencés ne sont pas les mêmes que les attributs référençants.

Une contrainte référentielle restreint les mises à jour sur les tables référencées et les tables dépendantes de la manière suivante :

- Soit t un tuple à insérer ou résultant d'une modification sur la table dépendante. Alors t sera rejeté si la valeur de t sur une clé etrangère ne réfère pas à la valeur sur la clé primaire (ou une clé unique) d'un tuple dans la table référencée.
- La modification ou suppression d'un tuple t' dans une table référencée sera rejetée s'il existe dans une table dépendante des tuples qui se réfère à t'.
- Cependant, si l'option DELETE CASCADE est ajoutée à la fin de la définition de la contrainte de chaque table dépendante, alors la suppression sera acceptée, avec la suppression dans les tables dépendantes des lignes qui se réfèrent à t'.
  `CONSTRAINT dept_cle_etr FOREIGN KEY(DeptNo) REFERENCES DEPT(DeptNo) ON DELETE CASCADE`

Contraintes par conditions booléennes

SQL d'Oracle supporte les contraintes spécifiées par des conditions booléenes sur les valeurs dans les domaines des attributs.
Sur la table EMP, on peut imposer :

Contraintes par conditions booléennes

SQL d'Oracle supporte les contraintes spécifiées par des conditions booléenes sur les valeurs dans les domaines des attributs.
Sur la table EMP, on peut imposer :

- les salaires doivent être positif, les primes doivent être supérieures ou égales à zéro.
  ```
  ALTER TABLE EMP ADD CONSTRAINT sal_prime_pos CHECK (Sal > 0 AND Prime >= 0) ;
  ```

Contraintes par conditions booléennes

SQL d'Oracle supporte les contraintes spécifiées par des conditions booléenes sur les valeurs dans les domaines des attributs.
Sur la table EMP, on peut imposer :

- les salaires doivent être positif, les primes doivent être supérieures ou égales à zéro.
  ```
  ALTER TABLE EMP ADD CONSTRAINT sal_prime_pos CHECK (Sal > 0 AND Prime >= 0) ;
  ```
- le salaire d'un employé doit être supérieur à sa prime :
  ```
  ALTER TABLE EMP ADD CONSTRAINT sal_sup_prime CHECK (Sal > Prime) ;
  ```

Un tuple dans la table EMP viole une telle contrainte si la condition booléenne spécifiée est évaluée à False, avec les valeurs sur Sal et Prime du tuple.

- Dans ce cas, le tuple est rejeté. L'action de m.à.j. sera annulée.
- Si la condition est évaluée à True ou Unknown (si valeur NULL) alors le tuple est considéré comme satisfaisant la contrainte.
- Si on veut éviter Unknown il faut imposer les contraintes NOT NULL ou :
  `ALTER TABLE DEPT ADD CONSTRAINT Adr_chk CHECK (Adr IN ('Lyon', 'Villeurbanne', 'Bron'));`

Restrictions sur les contraintes par conditions booléennes :

- Pour chaque tuple inséré ou m.à.j., la condition doit être évaluable avec les valeurs de la ligne.
- Un condition booléenne ne peut pas avoir de sous-requêtes SQL.

Suppression des contraintes

```
Alter table DEPT drop unique(dname, loc) ;
Alter table EMP drop primary key, drop constraint
dept_fkey ;
Drop table EMP cascade constraints ;
```

La dernière commande supprime la table et toutes les contraintes associées directement ou indirectement.

Information sur les contraintes

Le dictionnaire de données conserve les informations sur les contraintes définies sur la base. On peut les retrouver dans les vues : `CONSTRAINT_DEFS`, `USER_CONSTRAINTS`, `USERS_CONS_COLUMNS`, `USER_CROSS_REFS`,

Ex :

```
SELECT constraint_name, constraint_type, table_name,
r_constraint_name FROM user_constraints;
```

Dans la réponse de cette requête, on trouvera les abréviations :

- P pour primary,
- U pour unique,
- C pour check ou NOT NULL,
- R pour foreign key,
- et V pour le type de contraints créées par WITH CHECK OPTION pour les vues.