Modifications comportementales

Nous étudions dans cette section les mécanismes d'ajout, de suppression, d'activation et de désactivation des contraintes.

Faisons évoluer le schéma suivant. Les clés primaires sont nommées `pk_Compagnie` pour la table `Compagnie` et `pk_Avion` pour la table `Avion`.

Figure 3-4 *Schéma à faire évoluer*

Compagnie

comp	nrue	rue	ville	nomComp
AF	124	Port Royal	Paris	Air France
SING	7	Camparols	Singapour	Singapore AL

Affreter

compAff	immat	dateAff	nbPax
AF	F-WTSS	13-05-2003	85
SING	F-GAFU	05-02-2003	155
AF	F-WTSS	15-05-2003	82

Avion

immat	typeAvion	nbHVol	proprio
F-WTSS	Concorde	6570	SING
F-GAFU	A320	3500	AF
F-GLFS	TB-20	2000	SING

Ajout de contraintes

Jusqu'à présent, nous avons créé des tables en même temps que les contraintes. Il est possible de créer des tables seules (dans ce cas l'ordre de création n'est pas important et on peut même les créer par ordre alphabétique), puis d'ajouter les contraintes. Les outils de conception (*Win'Design*, *Designer* ou *PowerAMC*) adoptent cette démarche lors de la génération automatique de scripts SQL.

La directive `ADD CONSTRAINT` de l'instruction `ALTER TABLE` permet d'ajouter une contrainte à une table. La syntaxe générale est la suivante :

```
ALTER TABLE [schéma.]nomTable
ADD [CONSTRAINT nomContrainte] typeContrainte;
```

Comme pour l'instruction `CREATE TABLE`, quatre types de contraintes sont possibles :

- `UNIQUE (colonne1 [,colonne2]…)`
- `PRIMARY KEY (colonne1 [,colonne2]…)`

  `FOREIGN KEY (colonne1 [,colonne2]…)`

  `REFERENCES [schéma.]nomTablePère (colonne1 [,colonne2]…)`

  `[ON DELETE { CASCADE | SET NULL }]`
- `CHECK (condition)`

Clé étrangère

Ajoutons la clé étrangère à la table `Avion` au niveau de la colonne `proprio` en lui assignant une contrainte `NOT NULL` :

```
ALTER TABLE Avion
  ADD (CONSTRAINT nn_proprio CHECK (proprio IS NOT NULL),
       CONSTRAINT fk_Avion_comp_Compag FOREIGN KEY(proprio)
                        REFERENCES Compagnie(comp));
```

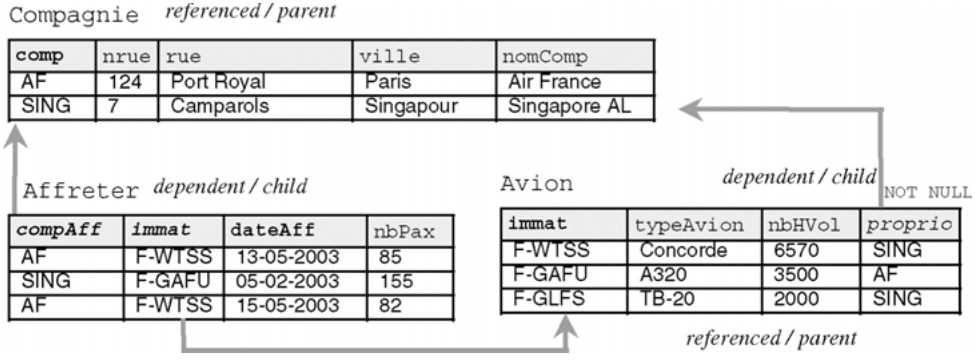
Clé primaire

Ajoutons la clé primaire de la table `Affreter` et deux clés étrangères (vers les tables `Avion` et `Compagnie`) :

```
ALTER TABLE Affrete ADD (
  CONSTRAINT pk_Affreter          PRIMARY KEY (compAff, immat, dateAff),
  CONSTRAINT fk_Aff_na_Avion      FOREIGN KEY(immat) REFERENCES
                                  Avion(immat),
  CONSTRAINT fk_Aff_comp_Compag FOREIGN KEY(compAff)
                REFERENCES Compagnie(comp));
```

Pour que l'ajout d'une contrainte soit possible, il faut que les données présentes dans la table respectent la nouvelle contrainte (nous étudierons plus tard les moyens de pallier ce problème). Les tables contiennent les contraintes suivantes :

Figure 3-5 *Après ajout de contraintes*



Suppression de contraintes

La directive `DROP CONSTRAINT` de l'instruction `ALTER TABLE` permet d'enlever une contrainte d'une table. La syntaxe générale est la suivante :

```
ALTER TABLE [schéma.]nomTable DROP CONSTRAINT nomContrainte [CASCADE];
```

La directive CASCADE supprime les contraintes référentielles des tables « pères ». On comprend mieux maintenant pourquoi il est si intéressant de nommer les contraintes plutôt que d'utiliser les noms automatiquement générés.

Supprimons la contrainte NOT NULL qui porte sur la colonne proprio de la table Avion :

```
ALTER TABLE Avion DROP CONSTRAINT nn_proprio;
```

Clé étrangère

Supprimons la clé étrangère de la colonne proprio. Il n'est pas besoin de spécifier CASCADE, car il s'agit d'une table « fils » pour cette contrainte d'intégrité référentielle.

```
ALTER TABLE Avion DROP CONSTRAINT fk_Avion_comp_Compag;
```

Clé primaire (ou candidate)

Supprimons la clé primaire de la table Avion. Il faut préciser CASCADE, car cette table est référencée par une clé étrangère dans la table Affreter. Cette commande supprime à la fois la clé primaire de la table Avion mais aussi les contraintes clés étrangères des tables dépendantes (ici seule la clé étrangère de la table Affreter est supprimée).

```
ALTER TABLE Avion DROP CONSTRAINT pk_Avion CASCADE;
```

Si l'option CASCADE n'avait pas été spécifiée, Oracle aurait renvoyé l'erreur « ORA-02273 : cette clé unique/primaire est référencée par des clés étrangères ».

La figure suivante illustre les trois contraintes qui restent : les clés primaires des tables Compagnie et Affreter et la clé étrangère de la table Affreter.

Figure 3-6 *Après suppression de contraintes*

Compagnie *referenced / parent*

comp	nrue	rue	ville	nomComp
AF	124	Port Royal	Paris	Air France
SING	7	Camparols	Singapour	Singapore AL

Affreter *dependent / child*

compAff	immat	dateAff	nbPax
AF	F-WTSS	13-05-2003	85
SING	F-GAFU	05-02-2003	155
AF	F-WTSS	15-05-2003	82

Avion

immat	typeAvion	nbHVol	proprio
F-WTSS	Concorde	6570	SING
F-GAFU	A320	3500	AF
F-GLFS	TB-20	2000	SING

Les deux possibilités pour supprimer ces trois contraintes sont décrites dans le tableau suivant. La deuxième écriture est plus rigoureuse car elle prévient des effets de bord. Il suffit, pour les

éviter, de détruire les contraintes dans l'ordre inverse d'apparition dans le script de création (tables « fils » puis « pères »).

Tableau 3-3 Suppression de contraintes

Avec CASCADE	**Sans** CASCADE

`ALTER TABLE Compagnie`
`  DROP CONSTRAINT pk_Compagnie` **`CASCADE`**`;`
`ALTER TABLE Affreter`
`  DROP CONSTRAINT pk_Affreter;`
	`ALTER TABLE Affreter`
	`  DROP CONSTRAINT fk_Aff_comp_Compag;`
	`ALTER TABLE Compagnie`
	`  DROP CONSTRAINT pk_Compagnie;`
	`ALTER TABLE Affreter`
	`  DROP CONSTRAINT pk_Affreter;`

Désactivation de contraintes

La désactivation de contraintes peut être intéressante pour accélérer des procédures de chargement (importation par SQL*Loader) et d'exportation massive de données. Ce mécanisme améliore aussi les performances de programmes *batchs* qui ne modifient pas des données concernées par l'intégrité référentielle ou pour lesquelles on vérifie la cohérence de la base à la fin.

La directive `DISABLE CONSTRAINT` de l'instruction `ALTER TABLE` permet de désactiver temporairement (jusqu'à la réactivation) une contrainte existante.

Syntaxe

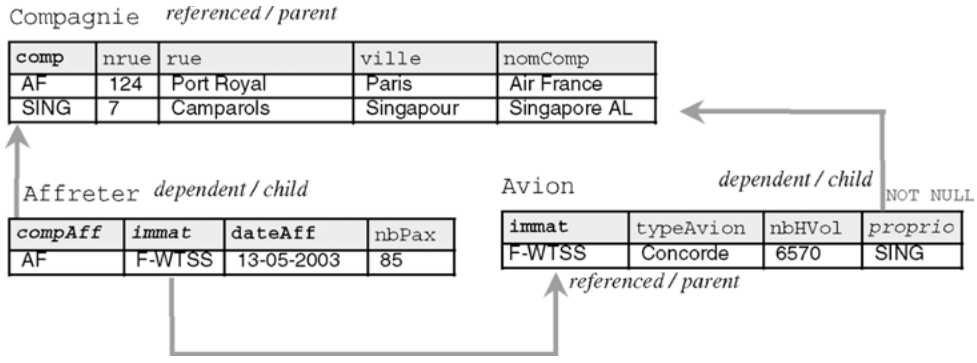
La syntaxe générale est la suivante :

```
ALTER TABLE [schéma.]nomTable
   DISABLE [ VALIDATE | NOVALIDATE ] CONSTRAINT nomContrainte
   [CASCADE] [ { KEEP | DROP } INDEX ] ;
```

- `CASCADE` répercute la désactivation des clés étrangères des tables « fils » dépendantes. Si vous voulez désactiver une clé primaire référencée par une clé étrangère sans cette option, le message d'Oracle renvoyé est : « `ORA-02297: impossible désactiver contrainte… - les dépendences existent` ».
- Les options `KEEP INDEX` et `DROP INDEX` permettent de préserver ou de détruire l'index dans le cas de la désactivation d'une clé primaire.
- Nous verrons plus loin l'explication des options `VALIDATE` et `NOVALIDATE`.

En considérant l'exemple suivant, désactivons quelques contraintes et insérons des enregistrements ne respectant pas les contraintes désactivées.

Figure 3-7 *Avant la désactivation de contraintes*



Contrainte de vérification

Désactivons la contrainte `NOT NULL` qui porte sur la colonne `proprio` de la table `Avion` et insérons un avion qui n'est rattaché à aucune compagnie :

```
ALTER TABLE Avion DISABLE CONSTRAINT nn_proprio;
INSERT INTO Avion VALUES ('Bidon1', 'TB-20', 2000, NULL);
```

Clé étrangère

Désactivons la contrainte de clé étrangère qui porte sur la colonne `proprio` de la table `Avion` et insérons un avion rattaché à une compagnie inexistante :

```
ALTER TABLE Avion DISABLE CONSTRAINT fk_Avion_comp_Compag;
INSERT INTO Avion VALUES ('F-GLFS', 'TB-22', 500, 'Toto');
```

Clé primaire

Désactivons la contrainte de clé primaire de la table `Avion`, en supprimant en même temps l'index, et insérons un avion ne respectant plus la clé primaire :

```
ALTER TABLE Avion DISABLE CONSTRAINT pk_Avion CASCADE DROP INDEX;
INSERT INTO Avion VALUES ('Bidon1', 'TB-21', 1000, 'AF');
```

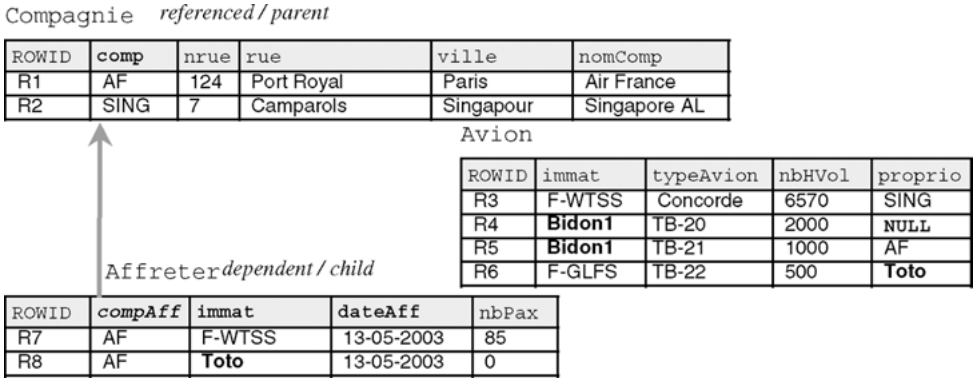
La désactivation de cette contrainte par `CASCADE` supprime aussi une des clés étrangères de la table `Affreter`. Insérons un affrètement qui référence un avion inexistant :

```
INSERT INTO Affreter VALUES ('AF', 'Toto', '13-05-2003', 0);
```

L'état de la base est désormais comme suit. Les *rowids* sont précisés pour illustrer les options de réactivation.

Web Bien qu'il semble incohérent de réactiver les contraintes sans modifier les valeurs ne respectant pas les contraintes (notées en gras), nous verrons que plusieurs alternatives sont possibles.

Figure 3-8 *Après désactivation de contraintes*



ROWID	comp	nrue	rue	ville	nomComp
R1	AF	124	Port Royal	Paris	Air France
R2	SING	7	Camparols	Singapour	Singapore AL

ROWID	immat	typeAvion	nbHVol	proprio
R3	F-WTSS	Concorde	6570	SING
R4	**Bidon1**	TB-20	2000	**NULL**
R5	**Bidon1**	TB-21	1000	AF
R6	F-GLFS	TB-22	500	**Toto**

ROWID	compAff	immat	dateAff	nbPax
R7	AF	F-WTSS	13-05-2003	85
R8	AF	**Toto**	13-05-2003	0

Réactivation de contraintes

La directive `ENABLE CONSTRAINT` de l'instruction `ALTER TABLE` permet de réactiver une contrainte.

Syntaxe

La syntaxe générale est la suivante :

```
ALTER TABLE [schéma.]nomTable
   ENABLE [ VALIDATE | NOVALIDATE ] CONSTRAINT nomContrainte
   [USING INDEX ClauseIndex] [EXCEPTIONS INTO tableErreurs];
```

- La clause d'index permet, dans le cas des clés primaires ou candidates (`UNIQUE`), de pouvoir recréer l'index associé.
- La clause d'exceptions permet de retrouver les enregistrements ne vérifiant pas la nouvelle contrainte (cas étudié au paragraphe suivant).

Il n'est pas possible de réactiver une clé étrangère tant que la contrainte de clé primaire référencée n'est pas active.

En supposant que les tables contiennent des données qui respectent les contraintes à réutiliser, la réactivation de la clé primaire (en recréant l'index) et d'une contrainte `NOT NULL` de la table `Avion` se programmerait ainsi :

```
ALTER TABLE Avion ENABLE CONSTRAINT pk_Avion
   USING INDEX (CREATE UNIQUE INDEX pk_Avion ON Avion (immat));

ALTER TABLE Avion ENABLE CONSTRAINT nn_proprio;
```