Le Langage SQL
Création de tables

Le Langage de Définition de Données (DDL)

Laurent Carmignac

Introduction

- Le langage de définition de données permet de créer la structure des tables qui composent votre schéma de données.
- Un schéma de données est issu d'un modèle conceptuel de données ou de tout autre modèle ou langage (UML par exemple) permettant la modélisation des données d'une application.

La création de tables (1)

- **La syntaxe générale**

```
CREATE TABLE [IF NOT EXISTS] nom_table
(
nom_colonne { type } [ DEFAULT valeur_default ][
  contrainte.. ],
…… ,
CONSTRAINT nom_contrainte {   UNIQUE |
                  PRIMARY KEY ( liste_colonne )|
                  FOREIGN KEY liste colonne
                        REFERENCES nom_table
  (liste_colonne)

) options de table;
```

La création de tables (2)

Quelques exemples

```
CREATE TABLE CLIENT
(
   CLI_NOM         CHAR(32),
   CLI_PRENOM   VARCHAR(32)
);

CREATE TABLE VOITURE
(
   VTR_ID                    INTEGER       NOT NULL PRIMARY KEY,
   VTR_MARQUE                CHAR(32)      NOT NULL,
   VTR_MODELE                VARCHAR(16),
   VTR_IMMATRICULATION       CHAR(10)      NOT NULL UNIQUE,
   VTR_COULEUR               VARCHAR(15)
   constraint C_color
            check (VTR_COULEUR IN
                        ('BLANC', 'NOIR', 'ROUGE', 'VERT', 'BLEU'))
);
```

La création de tables (3)

A sa création, la table peut être remplie par une requête SELECT.
Par défaut, une table est vide à sa création.

```
Exemple :
-- changement de nom des colonnes sélectionnées, même type
    CREATE TABLE Marque_Voiture
    (
       fabriquant,
       Modele
    ) AS  SELECT VTR_MARQUE , VTR_MODELE FROM VOITURE;

-- même nom des colonnes sélectionnées, même type
    CREATE TABLE Marque_Voiture_1
       AS SELECT VTR_MARQUE , VTR_MODELE FROM VOITURE;

-- Création de la copie d'une table
    CREATE TABLE Save_Voiture
       AS SELECT * FROM VOITURE;
```

Le type des colonnes(1)

- il existe plusieurs types pour définir les colonnes d'une table :
  - Les types numériques
  - le type auto_incrémenter
  - Les types de dates
  - Les types chaînes de caractères

Le type des colonnes(2)

- Les types numériques (1)
  - Les entiers

smallint	2 octets	-32768 à +32767
integer	4 octets	-2147483648 à +2147483647
bigint	8 octets	-9223372036854775808 à 9223372036854775807
decimal	variable	pas de limite
numeric	variable	pas de limite
real	4 octets	précision de 6 décimales
double precision	8 octets	précision de 15 décimales
serial	4 octets	1 à 2147483647
bigserial	8 octets	1 à 9223372036854775807

Le type des colonnes(3)

- **Les types numériques (2)**
  - **Les réels**

DECIMAL (length,decimals) **[UNSIGNED] [ZEROFILL]**
NUMERIC (length,decimals) **[UNSIGNED] [ZEROFILL]**

Salaire decimal(5,2)

Standard SQL = > -999.99 et 999.99

Le type des colonnes(4)

- **Les types numériques (3)**
  - **Le type auto_incrémenter (SERIAL ou BIGSERIAL)**

```
CREATE TABLE personne
(    id    SERIAL NOT NULL PRIMARY KEY ,
     name   VARCHAR(60) NOT NULL
);
```

```
INSERT INTO personne VALUES (default, 'Antonio Paz');
INSERT INTO personne VALUES (default, 'Lilliana Angelovska');
INSERT INTO personne VALUES (default, 'André Dopund');
INSERT INTO personne VALUES (default, 'René Dunp');
```

id	name
1	Antonio Paz
2	Lilliana Angelovska
3	André Dopund
4	René Dunp

Le type des colonnes(5)

- Les types numériques (4)
  - Le type auto_incrémenter (SERIAL ou BIGSERIAL)
    - Quelques fonctions sur les séquences

Un objet séquence est créé dans le schéma de la base de données il est nommé par défaut : NOMTABLE_NOMDUCHAMP_SEQ

Dans notre exemple : *Personne_id_seq*

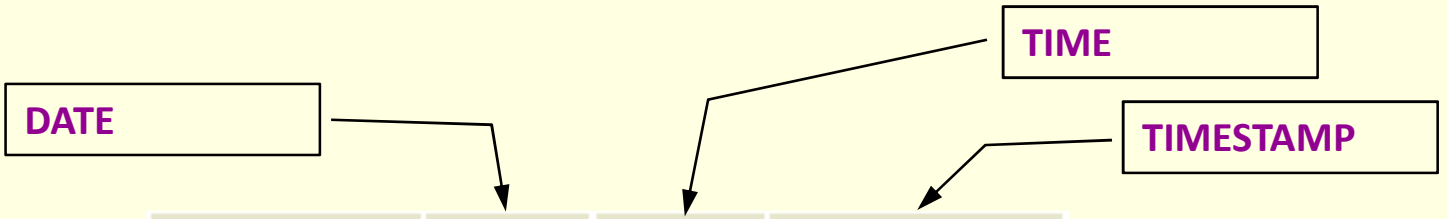
Fonction	Type de retour	Description
nextval('NomSeq')	bigint	Incrémente la valeur du champ auto incrémenté de la séquence spécifiée
currval('NomSeq')	bigint	Valeur de retour obtenue le plus récemment avec nextval pour la séquence spécifiée
Select setval('NomSeq', bigint)	bigint	Initialise la valeur courante de la séquence

```
INSERT INTO personne VALUES (nextval('personne_id_seq'), 'didier');
SELECT currval('personne_id_seq'); => retournera 4
```

Le type des colonnes(6)

- **Les types date et heure**

Nom	Description
DATE	dates seulement
INTERVAL	intervalle de temps
TIMESTAMP	date et heure
TIME	heures seulement



name	date_nais	heure_nais	dateetheure_nais
Antonio Paz	1968-01-15	22:30:12	1968-01-15 22:30:12
René Dunp	1978-04-23	12:45:00	1978-04-23 12:45:00
André Dopund	2000-12-24	24:00:00	2000-12-25 00:00:00
Lilliana Angelovska	1977-08-23	08:34:45	1977-08-23 08:34:45

Le type des colonnes(7)

Les types chaînes de caractères (1)

Nom	longueur
character(*n*) ou char(*n*)	longueur fixe, comblé avec des espaces
varying(n) ou varchar(n)	Longueur variable avec limite
text	longueur variable illimitée

Valeur	CHAR(4)	Espace requis	VARCHAR(4)	Espace requis
''	' '	4 octets	''	1 octet
'ab'	'ab '	4 octets	'ab'	3 octets

taille maximale possible pour une chaîne de caractères est de l'ordre 1 Go.

Le type des colonnes(8)

- **Les types boolean**

Le type booléen ne peut avoir que deux états:
TRUE (vrai) et FALSE (faux)

Les valeurs littérales valides pour l'état vrai sont :
TRUE | 't' | 'true' | 'y' | 'yes' | ' 1'

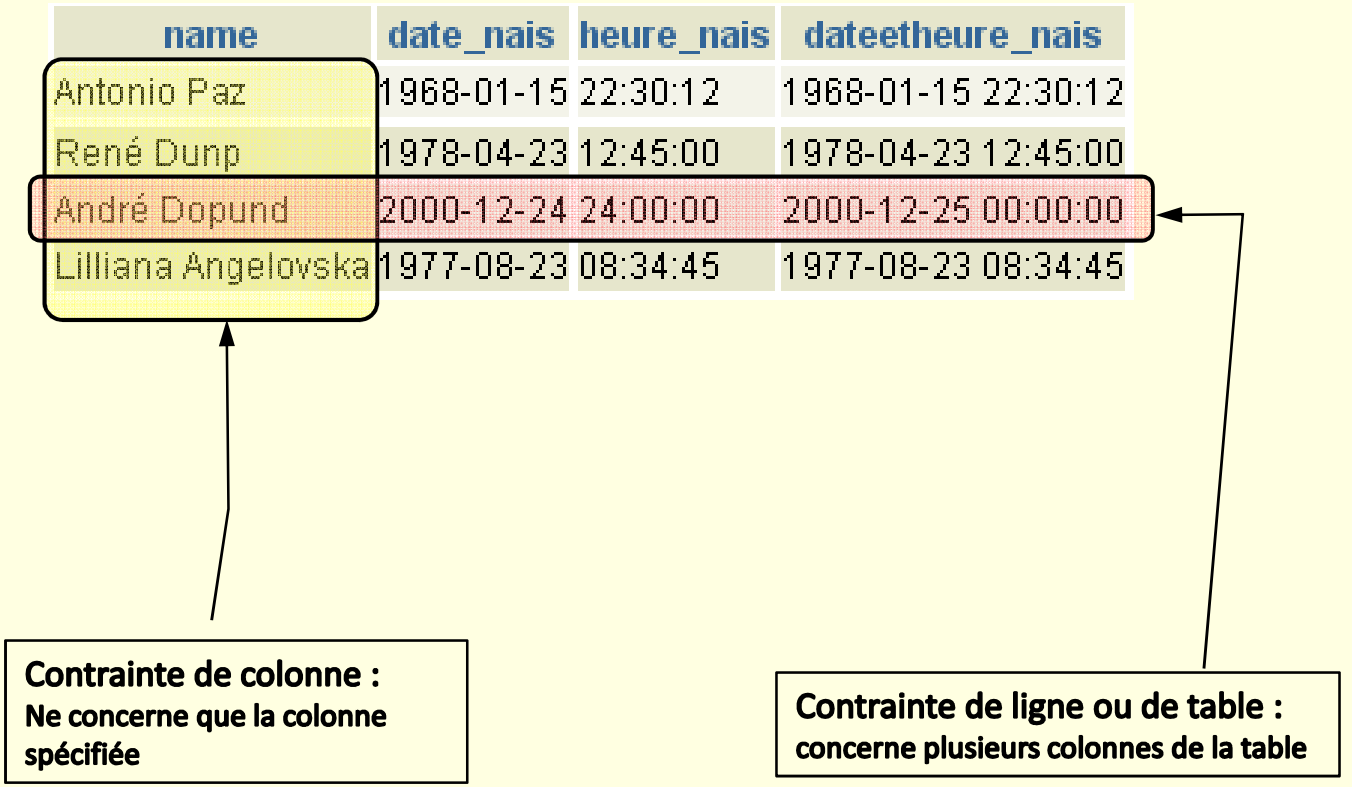
Pour l'état faux, les valeurs suivantes peuvent être utilisées:
FALSE | ' f' | ' false' | 'n' | 'no' | 'o'

Il est recommandé d'utiliser TRUE et FALSE (qui sont compatibles avec la norme SQL).

Il existe d'autres types sous postgres, pour en savoir plus :
http://docs.postgresqlfr.org/7.4/datatype.html

Les contraintes (1)

- **Introduction**



Les contraintes (2)

- **Les contraintes de colonnes (1)**

  - **NULL / NOT NULL**
  - **DEFAULT valeur**
  - **PRIMARY KEY**
  - **UNIQUE**
  - **CHECK**

Les contraintes (3)

- Les contraintes de colonnes (2)
  - Valeur obligatoire NOT NULL - NULL

```
CREATE TABLE PERSONNE
(
  PRS_ID                INTEGER      NOT NULL,
  PRS_NOM               VARCHAR(32)  NOT NULL,
  PRS_PRENOM            VARCHAR(32)  NULL,
  PRS_DATE_NAISSANCE    DATE
);
```

```
insert into PERSONNE values (null, null, null, now() );
```

Erreur SQL :

ERROR: null value in column "prs_id" violates not-null constraint

Les contraintes (4)

- Les contraintes de colonnes (3)
  - Valeur par défaut (DEFAULT)

```
CREATE TABLE PERSONNE
(
   PRS_ID              INTEGER,
   PRS_NOM             VARCHAR(32),
   PRS_PRENOM          VARCHAR(32),
   PRS_SEXE            CHAR(1)     DEFAULT 'M',
   PRS_DATE_NAISSANCE  TIMESTAMP   DEFAULT CURRENT_TIMESTAMP
);
```

```
insert into PERSONNE
       values(10, 'Duchemin', 'Paul',default, default);
```

prs_id	prs_nom	prs_prenom	prs_sexe	prs_date_naissance
10	Duchemin	Paul	M	2009-02-01 10:17:10.203

Les contraintes (5)

- Les contraintes de colonnes (4)
  - Clé primaire (PRIMARY KEY) obligatoire et unique

```
CREATE TABLE PERSONNE
(
  PRS_ID       INTEGER      PRIMARY KEY,
  PRS_NOM      VARCHAR(32),
  PRS_PRENOM   VARCHAR(32)
);
```

```
insert into PERSONNE values (NULL, 'Duchemin', 'Paul');
```

Erreur SQL :

ERROR: null value in column "prs_id" violates not-null constraint

```
insert into PERSONNE values (1, 'Duchemin', 'Paul');
insert into PERSONNE values (1, 'Duchemol', 'Pierre');
```

Erreur SQL :

ERROR: duplicate key violates unique constraint "t_personne1_pkey"

Les contraintes (6)

- **Les contraintes de colonnes (5)**
  - **Unicité (UNIQUE) (Si non renseigné = NULL)**

```
CREATE TABLE PERSONNE
(
   PRS_NOM          VARCHAR(32),
   PRS_PRENOM       VARCHAR(32),
   PRS_TELEPHONE    CHAR(14)  UNIQUE
);
```

```
INSERT INTO PERSONNE VALUES('DUPONT','MARCEL', '01 44 21 57 18');
INSERT INTO PERSONNE VALUES ('DUVAL', 'ANDRÉ',NULL);
INSERT INTO PERSONNE VALUES ('DURAND','JEAN', '06 11 86 46 69');
INSERT INTO PERSONNE VALUES ('DUGLAND','ALFRED','06 11 86 46 69');
```

Erreur SQL :

ERROR: duplicate key violates unique constraint "t_personne_prs_telephone_key"

Les contraintes (7)

- Les contraintes de colonnes (6)
  - La verification : CHECK

```
CREATE TABLE PERSONNE
(
  PRS_NOM          VARCHAR(32),
  PRS_PRENOM       VARCHAR(32),
  PRS_AGE    INTEGER
  constraint c_age check (PRS_AGE between 0 and 125)
);
```

```
INSERT INTO PERSONNE  VALUES('DUPONT', 'MARCEL', 52); -- OK
INSERT INTO PERSONNE  VALUES('DUVAL',  'ANDRÉ', 126); -- erreur
INSERT INTO PERSONNE  VALUES('DURAND', 'JEAN', -25);  -- erreur
```

Erreur SQL :

ERROR: new row for relation "t_personne" violates check constraint "c_age"

Les contraintes (7)

- **Les contraintes de table (1)**

  Les contraintes de table mettent en jeux plusieurs colonnes de la table

  - **PRIMARY KEY**
  - **UNIQUE**
  - **FOREIGN KEY**
  - **CHECK**

Les contraintes (8)

- **Les contraintes de table (2)**
  - **Les clés primaires (PRIMARY KEY)**

Exemple de clé composée de 2 champs

```
CREATE TABLE PERSONNE
(
 PRS_NOM            VARCHAR(32),
 PRS_PRENOM         VARCHAR(32),
 PRS_TELEPHONE      CHAR(14),
 CONSTRAINT PK_PRS PRIMARY KEY (PRS_NOM, PRS_PRENOM)
);
```

Les contraintes (9)

- **Les contraintes de table (3)**
  - **Unicité(UNIQUE)**

```
CREATE TABLE  PERSONNE
(
 PRS_ID                  INTEGER,
 PRS_NOM                 VARCHAR(32),
 PRS_PRENOM              VARCHAR(32),
 CONSTRAINT UNI_N_P UNIQUE (PRS_NOM, PRS_PRENOM)
);
```

Les contraintes (10)

- **Les contraintes de table (4)**
  - **La vérification CHECK**

```
CREATE TABLE FACTURE
(
FTC_ID          INTEGER,
FCT_DATE        DATE,
FCT_MT_MIN      DECIMAL(16,2),
FCT_MONTANT     DECIMAL(16,2),
Constraint C_ValMt     CHECK (FCT_MONTANT > FCT_MT_MIN)
);

INSERT INTO FACTURE values (10, '2007-06-04',10, 100); -- ok
INSERT INTO FACTURE values (10, '2007-06-04',10, 8); --erreur
```

Erreur SQL :

ERROR: new row for relation "facture" violates check constraint "c_valmt"

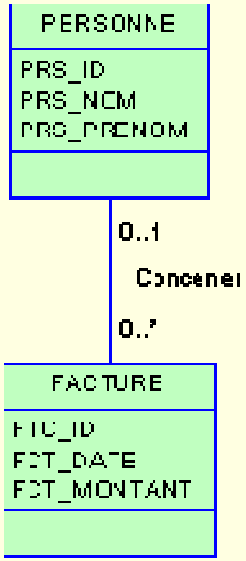
Les contraintes (11)

- Les contraintes de table (5)
  - Les clés étrangéres (FOREIGN KEY)

```
CONSTRAINT NOM_CONTRAINTE FOREIGN KEY (LISTE_COLONNE)
   REFERENCES NOM_TABLE_REF (LISTE_COLONNE_REF);
```

Les contraintes (12)

- **Les contraintes de table (6)**
  - **Les clés étrangères (FOREIGN KEY)**




```
CREATE TABLE PERSONNE
( PRS_ID                INTEGER  PRIMARY KEY,
  PRS_NOM               VARCHAR(32),
  PRS_PRENOM            VARCHAR(32)
);

CREATE TABLE FACTURE
( FTC_ID                INTEGER PRIMARY KEY,
  PRS_ID                INTEGER ,
  FCT_DATE              DATE,
  FCT_MONTANT           DECIMAL(16,2),
  FOREIGN KEY (PRS_ID) REFERENCES PERSONNE (PRS_ID)
);
```

Les contraintes (13)

- **Les contraintes de table (7)**
  - **La gestion de l'intégrité référentielle**

Le mode de gestion de l'intégrité consiste à se poser la question de ce que la machine doit faire dans le cas ou l'on tente de briser une intégrité référentielle.

Clé étrangère => clé primaire

```
CONSTRAINT nom_contrainte FOREIGN KEY (liste_colonne_table)
                  REFERENCES table_réf(liste_colonne_référencées)
[ON DELETE {CASCADE | SET NULL | NO ACTION | RESTRICT}]
[ON UPDATE {CASCADE | SET NULL | NO ACTION | RESTRICT}]
```

Les contraintes (14)

- **Les contraintes de table (8)**
  - **La gestion de l'intégrité référentielle**
  - **ON DELETE NO ACTION / ON UPDATE NO ACTION**

    La contrainte sera vérifiée et une erreur générée si les valuers violent la conrainte, c'est le comportement pas défaut

```
CREATE TABLE FACTURE
(
FTC_ID          INTEGER primary key,
PRS_ID          INTEGER ,
FCT_DATE        DATE,
FCT_MONTANT     DECIMAL(16,2),
FOREIGN KEY (PRS_ID) REFERENCES PERSONNE (PRS_ID)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);
```

Les contraintes (14)

Les contraintes de table (8)

- La gestion de l'intégrité référentielle
- ON DELETE NO ACTION / ON UPDATE NO ACTION
  - Exemple

facture

ftc_id	prs_id	fct_date	fct_montant
100	1	2009-03-01	2300.00
200	1	2009-02-23	1890.00
300	2	2009-01-15	590.00
400	2	2009-02-28	890.00

personne

prs_id	prs_nom	prs_prenom
1	Carmignac	laurent
2	Dupond	Jacques

```
-- tentative de suppression d'une personne ayant une facture
delete from personne
where prs_id=1;
```

Erreur SQL :

ERREUR: UPDATE ou DELETE sur la table « personne » viole la contrainte de clé étrangère « facture_prs_id_fkey » de la table « facture »
DETAIL: La clé (prs_id)=(1) est toujours référencée à partir de la table « facture ».

Les contraintes (15)

- Les contraintes de table (9)
  - La gestion de l'intégrité référentielle
  - ON DELETE CASCADE / ON UPDATE CASCADE

    La contrainte NE SERA PAS vérifiée et le SGDB supprimera ou modifiera toutes les valeurs de la clé étrangère qui sont liées à la valeur de la clé primaire supprimée ou modifiée

```
CREATE TABLE FACTURE
(
FTC_ID          INTEGER,
PRS_ID          INTEGER ,
FCT_DATE        DATE,
FCT_MONTANT     DECIMAL(16,2),
FOREIGN KEY (PRS_ID) REFERENCES PERSONNE (PRS_ID)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);
```

Les contraintes (15)

- Les contraintes de table (9)
  - La gestion de l'intégrité référentielle
  - ON DELETE CASCADE / ON UPDATE CASCADE
    - Exemple

personne

prs_id	prs_nom	prs_prenom
1	Carmignac	laurent
2	Dupond	Jacques

facture

ftc_id	prs_id	fct_date	fct_montant
100	1	2009-03-01	2300.00
200	1	2009-02-23	1890.00
300	2	2009-01-15	590.00
400	2	2009-02-28	890.00

```
-- Modification de la valeur de la clé d'une personne ayant une facture
Update personne
Set prs_id=10
where prs_id=1;
```

personne

prs_id	prs_nom	prs_prenom
10	Carmignac	laurent
2	Dupond	Jacques

facture

ftc_id	prs_id	fct_date	fct_montant
100	10	2009-03-01	2300.00
200	10	2009-02-23	1890.00
300	2	2009-01-15	590.00
400	2	2009-02-28	890.00

Les contraintes (15)

- Les contraintes de table (9)
  - La gestion de l'intégrité référentielle
  - ON DELETE CASCADE / ON UPDATE CASCADE
    - Exemple 2

personne

prs_id	prs_nom	prs_prenom
10	Carmignac	laurent
2	Dupond	Jacques

facture

ftc_id	prs_id	fct_date	fct_montant
100	10	2009-03-01	2300.00
200	10	2009-02-23	1890.00
300	2	2009-01-15	590.00
400	2	2009-02-28	890.00

```
-- suppression d'une paersonne ayant une facture
Delete from personne
where prs_id=10;
```

personne

prs_id	prs_nom	prs_prenom
2	Dupond	Jacques

facture

ftc_id	prs_id	fct_date	fct_montant
300	2	2009-01-15	590.00
400	2	2009-02-28	890.00

Les contraintes (16)

- **Les contraintes de table (10)**
  - **La gestion de l'intégrité référentielle**
  - **ON DELETE SET NULL / ON UPDATE SET NULL**

    La contrainte NE SERA PAS vérifiée et le SGDB modifiera toutes les valeurs de la clé étrangère qui sont liées à la valeur de la clé primaire supprimée ou modifiée en plaçant la valeur NULL dans la clé étrangére

```
CREATE TABLE FACTURE
(
FTC_ID          INTEGER,
PRS_ID          INTEGER ,
FCT_DATE        DATE,
FCT_MONTANT     DECIMAL(16,2),
FOREIGN KEY (PRS_ID) REFERENCES PERSONNE (PRS_ID)
        ON DELETE SET NULL
        ON UPDATE SET NULL
);
```

Les contraintes (16)

Les contraintes de table (10)

- La gestion de l'intégrité référentielle
- ON DELETE SET NULL / ON UPDATE SET NULL
  - Exemple

personne

prs_id	prs_nom	prs_prenom
1	Carmignac	laurent
2	Dupond	Jacques

facture

ftc_id	prs_id	fct_date	fct_montant
100	1	2009-03-01	2300.00
200	1	2009-02-23	1890.00
300	2	2009-01-15	590.00
400	2	2009-02-28	890.00

```
-- Modification de la valeur de la clé d'une personne ayant une facture
Update personne
Set prs_id=10
where prs_id=1;
```

personne

prs_id	prs_nom	prs_prenom
2	Dupond	Jacques
10	Carmignac	Laurent

facture

ftc_id	prs_id	fct_date	fct_montant
300	2	2009-01-15	590.00
400	2	2009-02-28	890.00
100	NULL	2009-02-23	1890.00
200	NULL	2009-03-01	2300.00

La suppression d'une personne aura le même effet sur la table facture

Modification et Suppression (1)

- **Ajout d'une colonne**

Syntaxe :

```
ALTER TABLE relation ADD COLUMN nom_colonne type [contrainte]
ou
ALTER TABLE relation ADD nom_colonne type [contrainte]
```

Ajoutons l'attribut *fax* et l'attribut ville:

```
ALTER TABLE T_PERSONNE
    ADD prs_fax DECIMAL(10,0),
    ADD prs_Ville varchar(10)  ;
```

Les nouvelles colonnes sont placées à la fin de la table

Modification et Suppression (2)

Supprimer une colonne

Attention, supprimer un attribut implique la suppression des valeurs qui se trouvent dans la colonne qui correspond à cet attribut.

Syntaxe :

```
ALTER TABLE relation DROP COLUMN attribut;
ou
ALTER TABLE relation DROP attribut;
```

Exemple :

```
ALTER TABLE T_PERSONNE DROP COLUMN prs_Ville;
```

Modification et Suppression (3)

- **Ajout/suppression d'une contrainte de clé étrangère**

Ajout :

```
ALTER TABLE T_TELEPHONE
        ADD CONSTRAINT FK_TEL_CLI FOREIGN KEY (CLI_ID)
                REFERENCES T_CLIENT(CLI_ID);
```

Suppression :

```
ALTER TABLE T_TELEPHONE
        DROP CONSTRAINT FK_TEL_CLI;
```

Modification et Suppression (4)

- **Ajout/suppression d'une clé primaire**

Ajout :

```
ALTER TABLE Personnes ADD PRIMARY KEY (nom,prenom)
```

Supression :

```
ALTER TABLE Personnes DROP constraint Personnes_pkey ;
```

ATTENTION : Vous devez chercher dans le schéma de la base de données le nom de la contrainte qui défini la clé primaire.

Par défaut ce nom est composé du nom de la table suivi de '_pkey'

Exemple : si la clé est dans la tables t_personne alors la contrainte de clé primaire ce nomme t_personne_pkey

Modification et Suppression (5)

- Changer le nom et/ou le type d'une colonne

Changer de nom de la colonne CLI_PRENOM de la table T_CLIENT

```
ALTER TABLE T_CLIENT
                RENAME COLUMN CLI_PRENOM TO PRENOM;
```

Changer le type de la colonne PRENOM sans la renommer

```
ALTER TABLE  T_CLIENT
                ALTER COLUMN PRENOM TYPE VARCHAR(20);
```

Modification et Suppression (6)

- **Modifier/Supprimer la valeur par défaut d'une colonne**

Modifie la valeur par défaut de la colonne CLI_ENSEIGNE de la table T_CLIENT

```
ALTER TABLE T_CLIENT
        ALTER CLI_ENSEIGNE SET DEFAULT  'PARTICULIER';
```

Supprime la valeur par défaut de la colonne CLI_ENSEIGNE

```
ALTER TABLE T_CLIENT
        ALTER CLI_ENSEIGNE DROP DEFAULT ;
```

Modification et Suppression (7)

- **Renommer une table**

```
ALTER TABLE T_CLIENT
        RENAME TO CLIENT ;
```

Pour en savoir plus sur la modification de la structure d'une table :

http://docs.postgresqlfr.org/7.4/sql-altertable.html

Modification et Suppression (8)

- **Supprimer une table**

```
DROP {TABLE | VIEW } NOM_OBJET ;
```

Exemple :

```
DROP TABLE personne ;
```

Les vues (1)

Introduction

- **Les vues permettent :**
  - **de simplifier le schéma relationnel ;**
  - **D'affiner les privilèges ;**
  - **De cacher certaines colonnes d'une table ;**
- **Syntaxe de création :**

```
CREATE [OR REPLACE] VIEW NOM_VUE [ ( NOM_COL1, [, NOM_COL2 ... ] ) ]
  AS
     REQUÊTE_SELECT
```

Les vues (2)

- **Création d'une vue**

trf_id	trf_date	prd_id	trf_valeur
1	1996-01-01	53	123.45
2	1998-09-15	53	128.52
3	1999-12-31	53	147.28
4	1997-01-01	89	254.89
5	1999-12-31	89	259.99
6	1996-01-01	97	589.52

```
CREATE VIEW V_TARIF
AS
SELECT TRF_ID,  PRD_ID, TRF_DATE AS TRF_DATE_DEBUT,
       (SELECT COALESCE (MIN(TRF_DATE) - INTERVAL '1 DAY',   CURRENT_DATE)
        FROM   T_TARIF T2
        WHERE  T2.PRD_ID = T1.PRD_ID
          AND T2.TRF_DATE > T1.TRF_DATE) AS TRF_DATE_FIN, TRF_VALEUR
FROM   T_TARIF T1
ORDER BY PRD_ID;
```

```
SELECT * FROM V_TARIF ;
```

trf_id	prd_id	trf_date_debut	trf_date_fin	trf_valeur
1	53	1996-01-01	1998-09-14 00:00:00	123.45
2	53	1998-09-15	1999-12-30 00:00:00	128.52
3	53	1999-12-31	2007-03-23 00:00:00	147.28
4	89	1997-01-01	1999-12-30 00:00:00	254.89
5	89	1999-12-31	2007-03-23 00:00:00	259.99
6	97	1996-01-01	2007-03-23 00:00:00	589.52

Les vues (3)

- **Supprimer une vue**

```
DROP VIEW  VIEW_NAME [, VIEW_NAME] ...
```

Exemple :

```
DROP VIEW V_TARIF;
```