LANGAGE DE DEFINITION DES DONNEES LDD

SOMMAIRE

I. TABLE

1. CREATE TABLE [schema.]table (colonne type [default expr], …)

2. Pour créer une table il faut avoir :

- Le privilège CREATE TABLE
- Un espace de stockage

Exemple

Create table Etudiants
(Netudiant number, nom nvarchar2(10), prenom nvarchar2(10))

Describe etudiants

Table	Column	Type De Données	Longueur	Précision	Echelle	Clé Primaire	Valeur Nullable	Valeur Par Défaut	Commentaire
ETUDIANTS	NETUDIANT	Number	-	-	-	-	✓	-	-
	NOM	Nvarchar2	20	-	-	-	✓	-	-
	PRENOM	Nvarchar2	20	-	-	-	✓	-	-
									1 - 3

1.TYPES DE DONNEES

Types de données	Description
CHAR [(*size [BYTE / CHAR])]*	Taille fixe comprise entre 1 et 2000
NCHAR [(*size)]*	Taille fixe comprise entre 1 et 2000
VARCHAR2 (size)	Taille Variable comprise entre 1 et 4000
NVARCHAR2 (size)	Taille Variable comprise entre 1 et 4000
NUMBER[(*precision [, scale]])*	Nombre ayant une précision p et une échelle s. La précision est comprise entre 1 et 38. L'échelle varie de -84 à 127
BINARY_FLOAT	32-bit nombre avec virgule flottante. C etype nécessite 5 octets
BINARY_DOUBLE	64-bit nombre avec virgule flottante. C etype nécessite 9 octets
LONG	Données caractères ayant une taille <= 2GO
DATE	Date comprise entre 1/1/4712 AJC et 31/12/999 APJC
TIMESTAMP	Année, mois, jour , heure, minute et seconde, fraction de seconde

ALTER TABLE

1. ALTER TABLE [schema.]table
   ADD (col1 type [default expr], ……),
   MODIFY (col1 type [default expr], ……),
   Drop(col1,col2, ….coln)
   ADD(Constraint nom_contrainte1 type de contrainte,
   Constraint nom_contrainte2 type de contarinte,
   ….
)
   Drop Constraint nom_contrainte

Exemples

alter table emp1 add(age number)
modify (ename varchar2(15), lname varchar2(15))

alter table emp1 drop (age)

DROP TABLE

1. DROP TABLE [schema.]table

- Exemple
- DROP TABLE emp1

TRUNCATE TABLE

1. TRUNCATE TABLE [schema.]table

- Exemple
- Truncate table emp1

RENAME TABLE

1. **RENAME TABLE [schema.]table1 TO [schema.]table2**

- **Exemple**
- **RENAME dept TO departement**

LES CONTRAINTES

1. CREATE table [schema.]table
   (col1 type [Default expr] [contrainte_colonne] , ….
   [Contrainte_table]
)

Contrainte au niveau colonne : contrainte d'intégrité incluse dans la définition de la colonne

Contrainte au niveau table : contrainte d'intégrité incluse dans la définition de la table

LES CONTRAINTES

1. DEFAULT
2. NOT NULL
3. CHECK

   UNIQUE

   PRIMARY KEY

   FOREIGN KEY

1.CONTRAINTE NOT NULL

1. La contrainte NOT NULL ne peut être définie qu'au niveau de la colonne, pas au niveau de la table

EXEMPLES

```
CREATE TABLE Fournisseurs1
(fournisseur_id numeric(10) not null PRIMARY KEY ,
 nom varchar2(50) not null,
contact varchar2(50)
)
CREATE TABLE Fournisseurs1
(fournisseur_id numeric(10) CONSTRAINT nn_fourn not null constraint
pk_fourn PRIMARY KEY ,
nom varchar2(50) not null,
 contact varchar2(50)
)
```

2.CONTRAINTE CHECK

1. La contrainte Check définit une condition que chaque ligne doit vérifier

Exemple

Constraint ck_emp_deptno CHECK(deptno between 10 AND 200)

3.CONTRAINTE UNIQUE

1. Une contrainte d'intégrité de type clé unique exige que chaque valeur dans une colonne ou dans un ensemble de colonnes constituant une clé soit unique.
2. Une contrainte unique autorise la valeur NULL à moins que vous définissiez des contraintes NOT NULL

Exemple

```
Constraint  uq_dept_dname UNIQUE(dname)
```

4.CONTRAINTE PRIMARY KEY

1. Une contrainte clé primaire crée une clé primaire pour la table. Une seule clé primaire peut être créée par table.
2. La clé primaire peut être constituée d'une ou plusieurs colonnes.
3. Aucune des colonnes faisant partie de la clé ne peut être NULL

Exemple

```
CONSTRAINT pk_dept_deptno PRIMARY KEY(deptno)
```

5.CONTRAINTE FOREIGN KEY

CREATE TABLE nom_table
(col1 type null/not null,col2 type null/not null,...
CONSTRAINT fk_table_colonne FOREIGN KEY (col1, col2, coln) REFERENCES table_parente (col1, col2, ... coln) ON DELETE {CASCADE|SET NULL}

NB: Créer les tables parentes avant les tables filles
Supprimer les tables filles avant les tables parentes

5.CONTRAINTE FOREIGN KEY

ALTER TABLE nom_table ADD CONSTRAINT fk_table_colonne FOREIGN KEY (col1, col2, ... coln) REFERENCES table_parente (col1, col2, ... coln) ON DELETE {CASCADE|SET NULL}
ALTER TABLE nom_table DROP CONSTRAINT fk_table_colonne

NB: Créer les tables parentes avant les tables filles
Supprimer les tables filles avant les tables parentes

5.CONTRAINTE FOREIGN KEY EXEMPLES

EXEMPLE 1 : Contrainte définit au niveau colonne avec clé simple

CREATE TABLE Fournisseurs (fournisseur_id numeric(10) not null PRIMARY KEY , nom varchar2(50) not null, contact varchar2(50))

CREATE TABLE Produits (produit_id numeric(10) not null PRIMARY KEY , fournisseur_id numeric(10) not null REFERENCES Fournisseurs (fournisseur_id), nom varchar2(50) not null)

5.CONTRAINTE FOREIGN KEY EXEMPLES

EXEMPLE 2 : Contrainte définit au niveau table avec clé simple

```
CREATE TABLE Fournisseurs
( fournisseur_id numeric(10) not null, nom varchar2(50) not null,
 contact varchar2(50),
 CONSTRAINT  pk_fournisseur PRIMARY KEY (fournisseur_id )
)

CREATE TABLE Produits
(produit_id numeric(10) not null,
fournisseur_id numeric(10) not null,
CONSTRAINT  fk_fournisseur  FOREIGN KEY (fournisseur_id)  REFERENCES
Fournisseurs (fournisseur_id ))
```

5.CONTRAINTE FOREIGN KEY EXEMPLES

EXEMPLE 3 : Contrainte définit au niveau table avec clé composée

```
CREATE TABLE Fournisseurs ( fournisseur_id numeric(10) not null, nom varchar2(50)
not null, contact varchar2(50),
CONSTRAINT  pk_fournisseur PRIMARY KEY (fournisseur_id ,nom))

CREATE TABLE Produits (produit_id numeric(10) not null,
fournisseur_id numeric(10) not null, nom varchar2(50) not null,
CONSTRAINT  fk_fournisseur  FOREIGN KEY (fournisseur_id , nom) REFERENCES
Fournisseurs (fournisseur_id ,nom))
```

5.CONTRAINTE FOREIGN KEY EXEMPLES

EXEMPLE 4 : Contrainte définit avec l'instruction Alter

```
CREATE TABLE Fournisseurs ( fournisseur_id numeric(10) not null, nom varchar2(50)
not null, contact varchar2(50),
CONSTRAINT  pk_fournisseur PRIMARY KEY (fournisseur_id ));

CREATE TABLE Produits (produit_id numeric(10) not null,
fournisseur_id numeric(10) not null, nom varchar2(50) not null);

ALTER TABLE  Produits
ADD CONSTRAINT  fk_fournisseur  FOREIGN KEY (fournisseur_id , nom)
REFERENCES Fournisseurs (fournisseur_id))
```

6.ACTIVATION / DESACTIVATION DES CONTRAINTES

ALTER TABLE nom_table {ENABLE|DISABLE } CONSTRAINT nom_contrainte

CREATION D'UNE TABLE A PARTIR D'UNE SOUS -INTERROGATION

1. CREATE TABLE [schema.]table (colonne type [default expr],)
AS subquery

Exemple

```
create table emp1 as select * from emp where deptno=20
select * from emp1
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DNAME
7566	JONES	MANAGER	7839	02/04/81	2975	-	20	RESEARCH
7788	SCOTT	ANALYST	7566	09/12/82	3000	-	20	RESEARCH
7902	FORD	ANALYST	7566	03/12/81	3000	-	20	RESEARCH
7369	SMITH	CLERK	7902	17/12/80	800	-	20	RESEARCH
7876	ADAMS	CLERK	7788	12/01/83	1100	-	20	RESEARCH

6.UTILISATION DES TABLES ET VUES SYSTEMES

AFFICHAGE DES COLONNES ASSOCIEES AUX NOMS DE CONTRAINTES

SELECT * from user_cons_columns

OWNER	CONSTRAINT_NAME	TABLE_NAME	COLUMN_NAME	POSITION
HR	REGION_ID_NN	REGIONS	REGION_ID	-
HR	REG_ID_PK	REGIONS	REGION_ID	1
HR	COUNTRY_ID_NN	COUNTRIES	COUNTRY_ID	-
HR	COUNTRY_C_ID_PK	COUNTRIES	COUNTRY_ID	1
HR	COUNTR_REG_FK	COUNTRIES	REGION_ID	1
HR	LOC_ID_PK	LOCATIONS	LOCATION_ID	1
HR	LOC_CITY_NN	LOCATIONS	CITY	-
HR	LOC_C_ID_FK	LOCATIONS	COUNTRY_ID	1
HR	DEPT_ID_PK	DEPARTMENTS	DEPARTMENT_ID	1
HR	DEPT_NAME_NN	DEPARTMENTS	DEPARTMENT_NAME	-

7.UTILISATION DES TABLES ET VUES SYSTEMES (suite)

AFFICHAGE DES COLONNES ASSOCIEES AUX NOMS DE CONTRAINTES

SELECT * FROM USER_CONSTRAINTS

CONSTRAINT_NAME	CONSTRAINT_TYPE	TABLE_NAME	SEARCH_CONDITION	R_OWNER	R_CONSTRAINT_NAME	DELETE_RULE	STATUS	DEFERRABLE	DEFERRED	VALIDATED
SYS_C003999	C	PERFORMANCES	"BORNEINF" IS NOT NULL	-	-	-	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED
SYS_C003998	C	INTERVENTIONS	"AFFAIRE_A_SUIVRE" IS NOT NULL	-	-	-	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED
SYS_C003997	C	INTERVENTIONS	"IDINTERVENTION" IS NOT NULL	-	-	-	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED
SYS_C003996	C	INTERVENANTS	"PRENOM" IS NOT NULL	-	-	-	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED
SYS_C003995	C	INTERVENANTS	"NOM" IS NOT NULL	-	-	-	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED
SYS_C003994	C	INTERVENANTS	"IDINTERVENANT" IS NOT NULL	-	-	-	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED
SYS_C003108	O	EMP_DETAILS_VIEW	-	-	-	-	ENABLED	NOT DEFERRABLE	IMMEDIATE	NOT VALIDATED
JHIST_DEPT_FK	R	JOB_HISTORY	-	HR	DEPT_ID_PK	NO ACTION	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED
JHIST_EMP_FK	R	JOB_HISTORY	-	HR	EMP_EMP_ID_PK	NO ACTION	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED
JHIST_JOB_FK	R	JOB_HISTORY	-	HR	JOB_ID_PK	NO ACTION	ENABLED	NOT DEFERRABLE	IMMEDIATE	VALIDATED

II.VUE

1. UNE VUE EST UNE TABLE LOGIQUE BASEE SUR UNE TABLE OU UNE VUE
2. LIMITE L'ACCES A LA BASE
3. FACILITE LA CREATION DE REQUETES COMPLEXES
4. GARANTIT L'INDEPENDANCE DES DONNEES
5. PRESENTE LES MEMES DONNEES SOUS DIFFERENTES FORMES

VUE SIMPLE COMPLEXE

1. VUE SIMPLE
   - UTILISE UNE SEULE TABLE
   - NE CONTIENT NI FONCTION NI GROUPE DE DONNEES
   - PERMET D'EXECUTER DES OPERATIONS DU LMD
2. VUE COMPLEXE
   - UTILISE PLUSIEURS TABLES
   - CONTIENT DES FONCTIONS OU DES GROUPES DE DONNEES
   - NE PERMET PAS TOUJOURS DES OPERATIONS DU LMD

CREATION D'UN VUE

CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW vue
[(alias [, alias], …)]
AS SUBQUERY [WITH CHECK OPTION [CONSTRAINT constrainte]]
[WITH READ ONLY]

FORCE : crée la vue que les tables existent ou non

Alias : indique les noms des expressions sélectionnées par la requête de la vue. Le nombre d'alias doit être égal au nombre d'expressions sélectionnées.

With check option : n'autorise l'insertion et la mise à jour des lignes que pour les lignes auxquelles la vue peut accéder.

With read only : aucune opération LMD ne peut être exécutée.

EXEMPLES DE VUES

EXEMPLES

- **CREATE VIEW empvue**
  AS SELECT employee_id matricule, last_name nom, first_name prénom, job_id grade FROM employees where department_id =30
- **DESC empvue**
- **create view empvue1(matricule, nom, prénom, grade) as**
  SELECT employee_id matricule, last_name , first_name , job_id FROM employees where department_id =30
  DESC empvue1
- **create view empvue2(matricule, nom, prénom, grade) as**
  SELECT employee_id matricule, last_name , first_name , job_id FROM employees where department_id =30
  WITH CHECK OPTION
- **create view empvue3(matricule, nom, prénom, grade) as**

SELECT employee_id matricule, last_name , first_name , job_id FROM employees where department_id =30
WITH READ ONLY

OPERATIONS SUR LES VUES

1. **MODIFICATION D'UNE VUE**

   CREATE OR REPLACE VIEW

2. **SUPPRESSION D'UNE VUE**

   DROP VIEW vue

OPERATIONS SUR LES VUES

1. **MODIFICATION D'UNE VUE**

   CREATE OR REPLACE VIEW

2. **SUPPRESSION D'UNE VUE**

   DROP VIEW vue

III.INDEX

1. DEFINITION D’UN INDEX
2. CREATION D’UN INDEX
3. OPERATIONS SUR LES INDEX
4. REGLES DE CREATION D’UN INDEX
5. VERIFICATION DES INDEX

1. DEFINTION D'UN INDEX

1. Un index est objet de la base de données qui permet d'accélérer la recherche des lignes
2. Les index sont indépendants logiquement et physiquement des tables qu'ils indexent.

2. CREATION D'UN INDEX

1. **Automatiquement**

   Un index unique est crée automatiquement lors de la définition d'une contrainte Clé Primaire

2. **Manuellement**

   Un index non unique peut être crée manuellement

3. OPERATIONS SUR LES INDEX

1.CREATE [UNIQUE] INDEX index ON table (col1, col2, …)

EXEMPLE

CREATE INDEX idx_emp_name ON emp(ename)

2.DROP INDEX index

EXEMPLE

DROP INDEX idx_emp_name

4. REGLES DE CREATION D'UN INDEX

1. La colonne doit être souvent utilisée dans la clause WHERE ou une condition de jointure
2. La colonne contient un grand nombre de valeurs NULL
3. Deux ou plusieurs colonnes sont souvent utilisées conjointement dans une clause WHERE ou une condition de jointure
4. La table est de grande taille et la plupart des requêtes doivent extraire moins de 2 à 4% des lignes

4. REGLES DE CREATION D'UN INDEX

Ne pas créez d'index si :

1. **La table est de petite taille**
2. **Les colonnes ne sont pas souvent utilisées comme condition dans une requête**
3. **La plupart des requêtes sont prévues pour extraire un trop grand pourcentage de lignes**
4. **La table est souvent mise à jour**

5. VERIFICATION DES INDEX

1. La vue USER_INDEXES contient les noms d'index et leur unicité
2. La vue USER_IND_COLUMNS contient les noms d'index , des tables et des colonnes

INDEX_NAME	INDEX_TYPE	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	UNIQUENESS	COMPRESSION	PREFIX_LENGTH	TABLESPACE_NAME	INI_TRANS	MAX_TRANS
IDX_C2	NORMAL	HR	T1	TABLE	NONUNIQUE	DISABLED	-	USERS	2	255
JHIST_EMP_ID_ST_DATE_PK	NORMAL	HR	JOB_HISTORY	TABLE	UNIQUE	DISABLED	-	USERS	2	255
JHIST_JOB_IX	NORMAL	HR	JOB_HISTORY	TABLE	NONUNIQUE	DISABLED	-	USERS	2	255
JHIST_EMPLOYEE_IX	NORMAL	HR	JOB_HISTORY	TABLE	NONUNIQUE	DISABLED	-	USERS	2	255
JHIST_DEPARTMENT_IX	NORMAL	HR	JOB_HISTORY	TABLE	NONUNIQUE	DISABLED	-	USERS	2	255
EMP_EMAIL_UK	NORMAL	HR	EMPLOYEES	TABLE	UNIQUE	DISABLED	-	USERS	2	255
EMP_EMP_ID_PK	NORMAL	HR	EMPLOYEES	TABLE	UNIQUE	DISABLED	-	USERS	2	255
EMP_DEPARTMENT_IX	NORMAL	HR	EMPLOYEES	TABLE	NONUNIQUE	DISABLED	-	USERS	2	255
EMP_JOB_IX	NORMAL	HR	EMPLOYEES	TABLE	NONUNIQUE	DISABLED	-	USERS	2	255

IV.SEQUENCE

1. DEFINITION D’UNE SEQUENCE
2. CREATION D’UNE SEQUENCE
3. VERIFICATION DES SEQUENCES
4. PSEUDOCOLONNES NEXVAL ET CURRVAL
5. UTILISATION D’UNE SEQUENCE
6. MODIFICATION D’UNE SEQUENCE
7. SUPPRESSION D’UNE SEQUENCE

1. DEFINITION D'UNE SEQUENCE

1. Génère automatiquement des numéros uniques
2. Est Partageable entre plusieurs utilisateurs et éventuellement entre plusieurs tables
3. Permet de créer une valeur de clé primaire

2. CREATION D'UNE SEQUENCE

```
CREATE  SEQUENCE
    [INCREMENT BY n]
    [START WITH n]
    [{MAXVALUE n | NOMAXVALUE}]
    [{MINVALUE n | NOMINVALUE}]
    [{CYCLE | NOCYCLE}]
    [{CACHE n | NOCACHE}]
```

INCREMENT BY : définit l'intervalle entre les numéros
START WITH n : premier numéro de la séquence
{MAXVALUE n | NOMAXVALUE} : valeur maximale
{MINVALUE n | NOMINVALUE} : valeur minimale
{CYCLE | NOCYCLE} : la séquence peut continuer à générer ou non des valeurs
{CACHE n | NOCACHE} : nombre de valeurs pré-allouées et conservées en mémoire.

2. CREATION D'UNE SEQUENCE

EXEMPLES

```
CREATE SEQUENCE sequence1 increment  by 1 start with 1 maxvalue 5

CREATE SEQUENCE sequence2 increment  by 5 start with 10 maxvalue 100
NOCACHE  NOCYCLE

CREATE SEQUENCE sequence3 increment  by 1 start with 1 maxvalue 5
CACHE  CYCLE
```

3. VERIFICATION DES SEQUENCES

Utiliser la table USER_SEQUENCES

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	CYCLE_FLAG	ORDER_FLAG	CACHE_SIZE	LAST_NUMBER
LOCATIONS_SEQ	1	9900	100	N	N	0	3300
DEPARTMENTS_SEQ	1	9990	10	N	N	0	280
EMPLOYEES_SEQ	1	999999999999999999999999999	1	N	N	0	207
SEQUENCE1	1	7	1	Y	N	3	7

4. PSEUDOCOLONNES NEXTVAL ET CURRVAL

NEXTVAL retourne la prochaine valeur de séquence disponible

CURRVAL : renvoie la valeur courante de la séquence

5. UTILISATION D'UNE SEQUENCE

INSERT INTO table VALUES (sequence1.NEXTVAL, val1, val2, …)

La valeur actuelle de la sequence1 est :

SEELECT sequence1.CURRVAL FROM DUAL

6. MODIFICATION D'UNE SEQUENCE

ALTER SEQUENCE sequence

[INCREMENT BY n]

[START WITH n]

[{MAXVALUE n | NOMAXVALUE}]

[{MINVALUE n | NOMINVALUE}]

[{CYCLE | NOCYCLE}]

[{CACHE n | NOCACHE}]

7. SUPPRESION D'UNE SEQUENCE

DROP SEQUENCE sequence

V.SYNONYM

1. Simplifie l’accès aux objets
2. Référencie une table appartenant à un autre utilisateur
3. CREATE [PUBLIC] SYNONYM synonym FOR object
4. DROP SYNONYM synonym