

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

Le Langage de Manipulation de Données (LMD) pour l'insertion, la mise à jour et la suppression de données. Le LMD est basé sur les opérateurs relationnels, auxquels sont ajoutés des fonctions de calcul d'agrégats et des instructions pour réaliser les opérations d'insertion, mise à jour et suppression.

Les 3 commandes de manipulation des données en SQL sont:

- ▶ INSERT
- ▶ UPDATE
- ▶ DELETE

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

INSERT (1/3)

L'ajout de tuples se fait à l'aide de la commande **INSERT**. Il faut fournir le nom d'une relation et une liste de valeurs pour le tuple. Les valeurs doivent être écrites dans le même ordre que celui dans lequel ont été spécifiés les attributs auxquels elles correspondent lors de la création de la table avec **CREATE TABLE**.

Pour insérer des données dans une base, il y a 2 syntaxes principales :

1. Insérer une ligne en indiquant les informations pour chaque colonne existante (en respectant l'ordre)

```
INSERT INTO table_name  
VALUES (value_list);
```

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

INSERT (2/3)

2. Si vous excluez une ou plusieurs colonnes de la commande **INSERT**, vous devez spécifier la liste de colonnes .

```
INSERT INTO table_name (column_list)  
VALUES( value_list);
```

Exemple:


Créez une nouvelle table nommée 'discounts' pour insérer des données :

```
CREATE TABLE discounts (  
    discount_id NUMBER,  
    discount_name VARCHAR2(255) NOT NULL,  
    amount NUMBER(3,1) NOT NULL,  
    start_date DATE NOT NULL,  
    expired_date DATE NOT NULL  
);
```

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

INSERT (3/3)

- Insérez une nouvelle ligne dans la table 'discounts'

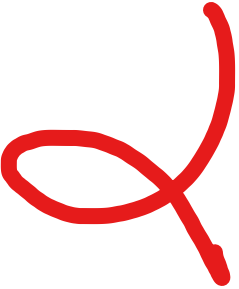


```
INSERT INTO discounts  
      (discount_name, amount, start_date, expired_date)  
VALUES  
      ('Summer Promotion', 9.5, '2017-05-01',  
      '2017-08-31');
```

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

UPDATE (1/4)

- La commande **UPDATE** permet d'effectuer des modifications sur des lignes existantes. Très souvent cette commande est utilisée avec **WHERE** pour spécifier sur quelles lignes doivent porter la ou les modifications.



```
UPDATE  
    table_name  
SET  
    column1 = value1,  
    column2 = value2,  
    column3 = value3,  
    ...  
WHERE  
    condition;
```

Remarque

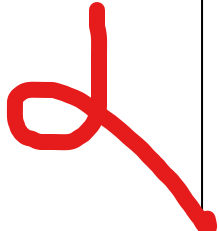
- La clause **WHERE** sélectionne les tuples à modifier.
- La clause **SET** spécifie les attributs à modifier et leurs nouvelles valeurs.

L'absence de clause **WHERE** signifie que les changements doivent être appliqués à toutes les lignes de la table cible.

LANGAGE DE MANIPULATION DE DONNÉES (LMD)


UPDATE (2/4)

- Créez une nouvelle table nommée 'parts'



```
CREATE TABLE parts (  
  part_id NUMBER NOT NULL,  
  part_name VARCHAR2(50) NOT NULL,  
  lead_time NUMBER(2,0) NOT NULL,  
  cost NUMBER(9,2) NOT NULL,  
  status NUMBER(1,0) NOT NULL,  
  PRIMARY KEY (part_id)  
);
```

- Inserez 3 nouvelles lignes dans la table 'parts'



```
INSERT INTO parts (part_name,lead_time,cost,status)  
VALUES ('seddictum',5,134,0);  
INSERT INTO parts (part_name,lead_time,cost,status)  
VALUES ('tristique',3,62,1);  
INSERT INTO parts (part_name,lead_time,cost,status)  
VALUES ('dolorquam,',16,82,1);
```

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

UPDATE (3/4)

- La commande UPDATE suivante modifie le `cost=130` de la table 'parts' pour le `part_id=1` :



```
UPDATE
  parts
SET
  cost = 130
WHERE
  part_id = 1;
```

- La commande UPDATE suivante modifie le `cost=120` et le 'status=1' de la table 'parts' pour le `part_id=5` :

```
UPDATE
  parts
SET
  cost = 120,
  status = 1
WHERE
  part_id = 5;
```


LANGAGE DE MANIPULATION DE DONNÉES (LMD)

UPDATE (4/4)

- La commande UPDATE suivante augmente les coûts (`cost`) de 50% de toutes les tuples de la table 'parts' :



```
UPDATE
```

```
parts
```

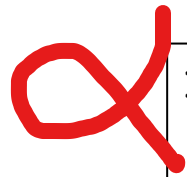
```
SET
```

```
cost = cost * 1,5;
```

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

DELETE (1/2)

- ▶ La commande **DELETE** permet de supprimer des lignes dans une table. En utilisant cette commande associé à **WHERE** il est possible de sélectionner les lignes concernées qui seront supprimées.



```
DELETE FROM table_name  
WHERE condition;
```

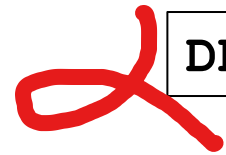
- ▶ Exemple: Supprimez de la table 'parts' les lignes dont l'identifiant (part_id) est 223 ou 335, et le coût 'cost' est supérieur à 110 :

```
DELETE FROM parts  
WHERE part_id IN(223,335) AND  
cost>110;
```

LANGAGE DE MANIPULATION DE DONNÉES (LMD)

DELETE (2/2)

- L'absence de clause WHERE signifie que toutes les lignes de la table cible sont enlevées



```
DELETE FROM table_name
```

- Exemple

```
DELETE FROM parts
```

LANGAGE DE CONTRÔLE DE TRANSACTIONS (LCT)

LANGAGE DE CONTRÔLE DE TRANSACTIONS (LCT)

Définition

- ▶ Les instructions de contrôle des transactions gèrent les modifications apportées par les instructions LMD.
- ▶ LCT permet de valider ou annuler des modifications de données dans la base de données.
- ▶ Pour contrôler les transactions, Oracle ne rend permanente aucune instruction LMD à moins que vous ne la validiez. Si vous ne validez pas la transaction et que l'alimentation est coupée ou que le système tombe en panne, la transaction est annulée.
- ▶ Les commandes LCT disponibles dans Oracle sont:
 - ▶ **COMMIT** : valide les modifications effectuées dans la transaction.
 - ▶ **ROLLBACK** : restaure l'état de la base de données jusqu'au dernier point de validation.
 - ▶ **SAVEPOINT** : utilisé pour spécifier un point dans la transaction auquel vous pourrez ultérieurement revenir en arrière.

LANGAGE DE CONTRÔLE DE TRANSACTIONS (LCT)

COMMIT


La commande **COMMIT** permet de valider les modifications apportées à une transaction et par conséquent de la rendre permanente.

Syntaxe:



```
COMMIT;
```

Exemple:



```
INSERT INTO discounts
    (discount_name, amount, start_date,
     expired_date)
VALUES
    ('Summer Promotion', 9.5, '2017-05-01',
     '2017-08-31');

COMMIT;
```

LANGAGE DE CONTRÔLE DE TRANSACTIONS (LCT)

ROLLBACK

La commande **ROLLBACK** permet d'annuler les modifications apportées à une transaction. **ROLLBACK** restaure l'état de la base de données au dernier point de validation.

Syntaxe:

```
ROLLBACK;
```

Exemple:



```
DELETE FROM discounts;  
ROLLBACK;
```

LANGAGE DE CONTRÔLE DE TRANSACTIONS (LCT)


SAVEPOINT

La commande **SAVEPOINT** permet de spécifier un point dans une transaction auquel vous pourrez revenir ultérieurement.

Syntaxe:

```
SAVEPOINT pointer_name;
```

Exemple:



```
INSERT INTO emp (empno,ename,sal) VALUES (109,'Sami',3000);  
SAVEPOINT a;  
INSERT INTO dept VALUES (10,'Sales','Hyd');  
SAVEPOINT b;  
INSERT INTO salgrade values ('III',9000,12000);
```


LANGAGE DE CONTRÔLE DE TRANSACTIONS (LCT)

SAVEPOINT

```
INSERT INTO emp (empno,ename,sal) VALUES (109,'Sami',3000);  
SAVEPOINT a;  
INSERT INTO dept VALUES (10,'Sales','Hyd');  
SAVEPOINT b;  
INSERT INTO salgrade VALUES ('III',9000,12000);
```

Si vous entrez:



```
ROLLBACK TO a;
```

Ensuite, la ligne de la table de **salgrade** et le **dept** seront annulés. À ce stade, vous pouvez valider la ligne insérée dans la table **emp** ou annuler la transaction.

LANGAGE DE CONTRÔLE DE TRANSACTIONS (LCT)

SAVEPOINT

```
INSERT INTO emp (empno,ename,sal) VALUES (109,'Sami',3000);  
SAVEPOINT a;  
INSERT INTO dept VALUES (10,'Sales','Hyd');  
SAVEPOINT b;  
INSERT INTO salgrade VALUES ('III',9000,12000);
```

Si vous entrez:

```
ROLLBACK TO b;
```

La ligne insérée dans la table **salgrade** sera annulée. À ce stade, vous pouvez valider la ligne insérée dans la table **dept** et la table **emp** ou annuler au point de **SAVEPOINT a** ou annuler (**ROLLBACK**) complètement la transaction.

LANGAGE DE CONTRÔLE DE TRANSACTIONS (LCT)

SAVEPOINT

```
INSERT INTO emp (empno,ename,sal) VALUES (109,'Sami',3000);  
SAVEPOINT a;  
INSERT INTO dept VALUES (10,'Sales','Hyd');  
SAVEPOINT b;  
INSERT INTO salgrade VALUES ('III',9000,12000);
```

- Si vous entrez:

ROLLBACK;

Ensuite, toutes les transactions sont annulées.

- Si vous entrez:

COMMIT;

Ensuite, toutes les transactions sont validées et tous les **SAVEPOINTS** sont **supprimés.**