Data Definition Language

Mirunalini P
Associate Professor
Department of Computer Science & Engineering
SSN College of Engineering

Overview

- Table
- CREATE, ALTER, TRUNCATE, DROP
- View
- CREATE, DROP

CREATE TABLE

- Syntax:

```
CREATE TABLE table
(column datatype [DEFAULT expr][, ...]);
```

- Example:

```
CREATE TABLE dept (deptno NUMBER(2),
               dname VARCHAR2(14),
               loc VARCHAR2(13)
               );
```

Constraints

- Constraints enforce rules at the table level.
- Constraints prevent the deletion of a table if there are dependencies.
- The following constraint types are valid:
- NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK

Constraints - Guidelines

- Name a constraint or the Oracle server generates a name by using the SYS_Cxxx format.
- Create a constraint either:
  - At the same time as the table is created, or
  - After the table has been created
  - Define a constraint at the column or table level.

CREATE TABLE with Constraints

- Syntax:

```
CREATE TABLE table (
column datatype [column_constraint],
…
[table_constraint]);
```

- Column constraint level – References a single column.
- Table constraint level – References one or more columns and is defined separately from column definitions.

CREATE TABLE with Constraints

- Column constraint level

```
column [CONSTRAINT constraint_name] constraint_type,
```

- Table constraint level

```
column,...

[CONSTRAINT constraint_name] constraint_type

(column, ...),
```

CREATE TABLE with Constraint – Not Null

- NOT NULL constraint ensures that the column contains no null values.
- Example:

```
CREATE TABLE departments(
    department_id NUMBER(4),
    department_name VARCHAR2(30)
    CONSTRAINT dept_name_nn NOT NULL );
```

CREATE TABLE with Constraint – Primary Key

- The PRIMARY KEY constraint is a *column or set of columns* that uniquely identifies each row in a table.
- This constraint enforces uniqueness of the column or column combination.
- Ensures that primary key column can NOT contain a null value.

CREATE TABLE with Constraint – Primary Key

- Example:

```
CREATE TABLE departments (
    department_id NUMBER(4),
    department_name VARCHAR2(30),
    manager_id NUMBER(6),
CONSTRAINT dept_id_pk PRIMARY KEY(department_id));
```

CREATE TABLE with Constraint – Foreign Key

- FOREIGN KEY is a referential integrity constraint.
- It designates a *column or combination of columns* as a foreign key and establishes a relationship between a *primary key* or a *unique key* in the *same or a different table*.
- A foreign key value must match an existing value in the parent table or be NULL.

CREATE TABLE with Constraint – Foreign Key

- Example:

```
CREATE TABLE employees( employee_id NUMBER(6),
    last_name VARCHAR2(25),
    email VARCHAR2(25),
    salary NUMBER(8,2),
    department_id NUMBER(4),
CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)
REFERENCES departments(department_id));
```

CREATE TABLE with Constraint – Foreign Key

- ON DELETE CASCADE: Deletes the dependent rows in the child table when a row in the parent table is deleted.
- Example:

```
CREATE TABLE employees( employee_id NUMBER(6),
    last_name VARCHAR2(25),
    salary NUMBER(8,2),
    department_id NUMBER(4) CONSTRAINT emp_dept_fk
REFERENCES departments(department_id) ON DELETE
CASCADE );
```

CREATE TABLE with Constraint – Unique

- A unique requires that no two rows of a table can have duplicate values in a specified column or set of column.
- UNIQUE constraints allow the input of nulls.
- Example:

```
CREATE TABLE employees(employee_id NUMBER(6),
    last_name VARCHAR2(25) NOT NULL,
    email VARCHAR2(25),
    salary NUMBER(8,2),
    CONSTRAINT emp_email_uk UNIQUE(email));
```

CREATE TABLE with Constraint – Check

- Defines a condition that each row must satisfy
- Example:

```
CREATE TABLE employees(employee_id NUMBER(6),
    last_name VARCHAR2(25) NOT NULL,
    salary NUMBER(2),
    CONSTRAINT emp_salary_min CHECK (salary > 0));
```

CREATE TABLE with Constraint – Check

- Check constraint can also be used with built-in functions.
- Example:

```
CHECK (gender in ('M','F'))

CHECK (length(reg_no)=8)

CHECK (class LIKE 'LH%')

CHECK ((extract(year from dob))>1990)

CHECK (salary BETWEEN 1000 AND 2000)
```

ALTER TABLE

To alter the table defintion and its constraints

ALTER TABLE

- Use the ALTER TABLE statement to add columns.

-
  ```
  ALTER TABLE table
  ADD         (column datatype [DEFAULT expr]
              [, column datatype]...);
  ```
- Example:
  ```
  ALTER TABLE department
  ADD (job_id VARCHAR2(9));
  ```

ALTER TABLE

- Use the ALTER TABLE statement to modify columns.

```
•ALTER TABLE table
MODIFY    (column datatype [DEFAULT expr]
          [, column datatype]...);
```

- Example:

```
ALTER TABLE department
MODIFY (last_name VARCHAR2(30));
```

ALTER TABLE

- Use the ALTER TABLE statement to drop columns.
- `ALTER TABLE table DROP  column;`
- Example:

```
ALTER TABLE department
DROP COLUMN job_id;
```

ALTER TABLE

- Use the ALTER TABLE statement to:
- Add or drop a constraint, but not modify its structure.
- Enable or disable constraints.
- Add a NOT NULL constraint by using the MODIFY clause.
- Syntax:

```
ALTER TABLE table
ADD [CONSTRAINT constraint] type (column)
```

ALTER TABLE

- Add constraint:

```
ALTER TABLE employees
ADD CONSTRAINT emp_manager_fk
FOREIGN KEY(manager_id)
REFERENCES employees(employee_id);
```

- Drop constraint:

```
ALTER TABLE employees
DROP CONSTRAINT emp_manager_fk;
```

TRUNCATE TABLE

- The TRUNCATE TABLE statement:
- Removes all rows from a table
- Releases the storage space used by that table
- Example:

```
TRUNCATE TABLE detail_dept;
```

- You cannot roll back row removal when using TRUNCATE.

DROP TABLE

- All data and structure in the table is deleted.
- Any pending transactions are committed.
- All indexes are dropped.
- You cannot roll back the DROP TABLE statement.
- Example:

```
DROP TABLE detail_dept;
```

References

- Introduction to Oracle9i: SQL