

## INCLUDING CONSTRAINTS -- CHAPTER 11 --

- What are 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 in ORACLE:
  - NOT NULL
    - Specifies that this column may not contain a null value.
  - UNIQUE
    - Specifies a column or combination of columns whose values must be unique for all rows in the table.
  - PRIMARY KEY
    - Uniquely identifies each row of the table.
  - FOREIGN KEY
    - Establishes and enforces a foreign key relationship between the column and the column of and a column of the referenced table.
  - CHECK
    - Specifies a condition that must be true.
- Note that if you do not give it a name, Oracle will generate a name with the format SYS\_Cn where *n* is an integer to create a unique constraint name.
- You can view the constraints defined for a specified table by looking at the USER\_CONSTRAINTS data dictionary table.

```
SQL> r
1 create table scott.victor3434
2 (numero NUMBER(4),
3 name VARCHAR2(10),
4* CONSTRAINT PK PRIMARY KEY (numero))

Table created.

SQL> select * from victor3434;

no rows selected

SQL> describe victor3434;
Name                                     Null?    Type
-----
NUMERO                                     NOT NULL NUMBER(4)
NAME                                       VARCHAR2(10)
```

- Note that constraints can be added after the table has been created. Note also that constraints may be defined at either the column or the table level.
- More on Foreign Keys: These designate a column or combination of columns as foreign key and establishes a relationship between a primary key or a unique key in the same table or a different table. A foreign key value must match an existing value in the parent table or be NULL. Foreign keys are based on data values and are purely logical, not physical; they are pointers.

```
SQL> r
1 CREATE TABLE emp(
```

```

2 empno NUMBER(4),
3 ename VARCHAR2(10) NOT NULL,
4 job VARCHAR2(9),
5 mgr NUMBER(4),
6 hiredate DATE,
7 sal NUMBER(7,2),
8 comm NUMBER(7,2),
9 soc NUMBER(4) NOT NULL,
10 CONSTRAINT emp_SOC_FK FOREIGN KEY (soc)
11* REFERENCES victor3434 (soc)

```

Table created.

```
SQL> describe emp;
```

Name	Null?	Type
EMPNO		NUMBER(4)
ENAME	NOT NULL	VARCHAR2(10)
JOB		VARCHAR2(9)
MGR		NUMBER(4)
HIREDATE		DATE
SAL		NUMBER(7,2)
COMM		NUMBER(7,2)
SOC	NOT NULL	NUMBER(4)

- Notice now that the victor3434 table, which contains the foreign key in emp prevents such table from being deleted as shown below...

```

SQL> DROP TABLE VICTOR3434
2 /
DROP TABLE VICTOR3434
*
ERROR at line 1:
ORA-02449: unique/primary keys in table referenced by foreign keys

```

- FOREIGN KEY Constraint Keywords:
  - FOREIGN KEY: Defines the column in the child table at the table constraint level
  - REFERENCES: Identifies the table and column in the parent table.
  - ON DELETE CASCADE: Allows deletion in the parent table and deletion of the dependent rows in the child table.
- The CHECK constraint defines a condition that each row must satisfy. The condition can use the same constructs as query conditions, with the following exceptions:
  - References to CURRVAL, NEXTVAL, LEVEL, and ROWNUM pseudocolumns
  - Calls to SYSDATE, UID, USER, and USERENV functions
  - Queries that refer to other values in other rows.
- Note that a single column can have multiple CHECK constraints that reference the column in its definition. There is no limit to the number to the limit of CHECK constraints that you can define on a column. Further, CHECK constraints can be defined at either the column or table levels.
- Adding a constraint to a table is performed as shown in the following example:

```

SQL> alter table victor3434
2 add CONSTRAINT boss_ck CHECK(boss = 'Rick');

```

Table altered.

```
SQL> select * from victor3434
2 /
```

SOC NAME	BOSS	SALARY
3434 Victor	Rick	3000
3535 Ryan	Rick	2500

```
SQL> insert into victor3434
2 values (3232, 'John', 'Rick', 3001);
```

1 row created.

```
SQL> insert into victor3434
2 values (3131, 'Jane', 'Jack', 3001);
```

```
insert into victor3434
*
```

ERROR at line 1:

ORA-02290: check constraint (VSANCHEZ.BOSS\_CK) violated

- Note in the previous example that we created a constraint that makes sure that all rows entered must contain the string 'Rick' under the BOSS column. If this is not the case, insertions are rejected and the appropriate error message is returned, i.e., **VSANCHEZ.BOSS\_CK**.
- Dropping a constraint is also easily accomplished by using the DROP keyword right before the CONSTRAINT keyword when ALTER-ing a table...

```
SQL> r
1 ALTER table victor3434
2* DROP CONSTRAINT boss_ck
```

Table altered.

- Notice that it is not necessary to include the table name ('victor3434.') before the name of the constraint ('boss\_ck').
- Note that the CASCADE option of the DROP clause causes any dependent constraints also to be dropped.

```
ALTER TABLE table
DROP PRIMARY KEY | UNIQUE (column) |
CONSTRAINT constraint [CASCADE];
```

- Execute the DISABLE clause of the ALTER TABLE statement to deactivate an integrity constraint. Apply the CASCADE option to disable dependent integrity constraints.

```
ALTER TABLE table
DISABLE CONSTRAINT constraint [CASCADE];
```

- Similarly, currently disabled constraints can be enabled using the ENABLE clause.

```
ALTER TABLE table
ENABLE CONSTRAINT constraint [CASCADE];
```

- Cascading CONSTRAINTS clause also drops all multicolumn constraints defined on the dropped columns.

```
SQL> ALTER TABLE victor3434 DROP (pk) CASCADE COSNTRAINS;
```

- Viewing constraints: After creating constraints, you can confirm its existence by issuing a DESCRIBE command. The only constraint that you can verify is the NOT NULL constraint. To view all constraints on your table, query the USER\_CONSTRAINTS table.

```
SELECT constraint_name, constraint_type, search_condition
FROM user_constraints
WHERE table_name = 'emp';
```

- You can also view the columns associated with constraints as follows:

```
SELECT constraint_name, column_name
FROM user_cons_columns
WHERE table_name = 'emp';
```

- this view is especially useful for constraints that use the system-assigned name.