DEFERRING CONSTRAINTS:

Constraint enforcement is necessary but it is sometimes required to defer or postpone a constraint check until the end of the transaction. This can be achieved using Deferring Constraints technique.

For example if we run a DML operation that violates constraints, the oracle server will return an error.

USE CASE:

Let's consider two tables, DEPARTMENTS and EMPLOYEES

The EMPLOYEES table has a DEPARTMENT_ID column which is a foreign key to the DEPARTMENT table DEPARTMENT_ID column.

If we want to update the DEPARTMENT_ID = 90 to DEPARTMENT_ID = 300, in the EMPLOYEE table (300 does not exist in the DEPARTMENT_ID column of DEPARTMENT table).

SOLUTION-1:

1) Create a new record of DEPARTMENT_ID = 300 in the DEPARTMENT table.
2) Update the records in the EMPLOYEE table, whose record has DEPARTMENT_ID = 90.
3) Delete the DEPARTMENT_ID = 90 in the DEPARTMENT table.

Other solution would be temporarily drop or disable the constraints and make the changes but this would create more problems in a real environment.

SOLUTION-2:

Constraints can have two states,

1) DEFERRABLE: Should not create unless it is necessary. They create non-unique indexes.
    1) INITIALLY IMMEDIATE: This is default.
    2) INITIALLY DEFERRED: Waits until commit.
2) NON DEFERRABLE: This is default. Can be changed to deferrable.

We can set the DEFERRED CONSTRAINTS for the current session (SET CONTRAINT) or for the entire session (USING ALTER SESSION SET CONSTRAINT - All constraints have to set in this case).

```
DROP TABLE departments_copy;
```

```
     CREATE TABLE departments_copy AS SELECT * FROM departments;

     ALTER TABLE departments_copy
     ADD CONSTRAINT dept_cpy_id_pk PRIMARY KEY (department_id)
DEFERRABLE INITIALLY DEFERRED;

     INSERT INTO departments_copy VALUES (10, 'Temp Department',
200, 1700);

     SET CONSTRAINT dept_cpy_id_pk IMMEDIATE;

     SET CONSTRAINT dept_cpy_id_pk DEFERRED;
```