UNIT-3

Integrity Constraints

Domain Constraints

Data constraints are rules created for entering and preventing users from entering invalid data into tables.

Oracle permits data constraints to be attached to table columns via SQL sentence that will check data for integrity. Once data constraints are attached, oracle engine will check data being entered into a table against the data constraints.

If the data is valid, it is stored in table, else data is rejected.

Even if the single column of the record being entered into the table fails constraints the entire record is rejected and not stored in the table.

The data constraints can be attached to the column using create table and alter table command. There are two types of data constraints-

- I/O constraints
- Business rule constraints

I/O Constraints:

I/O constraints are divided into following different categories.

a) Primary Key:

The primary key data constraints attached to the column ensures that the data entered in the table column is unique across the entire column and none of the cell belonging to the column are left empty.

b) Foreign Key:

This constraints establishes the relationship between records across the master and details table (parent and child). It ensures that records can not be inserted into detail or child table if corresponding record in the master or parent table does not exists and records of master table can not be deleted if corresponding records in the detail table exists.

In addition to primary key and foreign key Oracle has not null and unique as I/O constraints.

Business Rule Constraints:

- Business rule can be implemented into Oracle table by check constraints business rule validation checks are perform a right operation on the table i.e. insert / update.
- Any insert / update statement causes the relevant constraints to be evaluated. The constraints must be satisfied for the statement to be executed.

SQL Constraints

- SQL constraints are used to specify rules for the data in a table.
- Constraints are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the table. If there is any violation between the constraint and the data action, the action is aborted.
- Constraints can be column level or table level. Column level constraints apply to a column, and table level constraints apply to the whole table.
  The following constraints are commonly used in SQL:

NOT NULL :	- Ensures that a column cannot have a NULL value
UNIQUE :	- Ensures that all values in a column are different
PRIMARY KEY :	- A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
FOREIGN KEY :	- Uniquely identifies a row/record in another table
CHECK :	- Ensures that all values in a column satisfies a specific condition
DEFAULT :	- Sets a default value for a column when no value is specified
INDEX :	- Used to create and retrieve data from the database very quickly

SQL Create Constraints

Constraints can be specified when the table is created with the CREATE TABLE statement, or after the table is created with the ALTER TABLE statement.

Syntax

```
CREATE TABLE table_name (
  column1 datatype constraint,
  column2 datatype constraint,
  column3 datatype constraint,
  ....
);
```

Primary Key

- A primary key constraint is one or more columns in a table used to identify each row in the table. A table can have only one primary key. And it is combination of not null and unique constraint means the column which having primary key cannot be left blank or cannot use repeat value.
- A single column primary key is called as simple key and multiple column primary key called as composite key.

Ex. The primary key defined on column level as-

Syntax : column name data type (size) primary key

- e.g. create table student (Roll_no number (10) primary key, name varchar2 (20), address carchar2 (20));

The primary key constraint on table level:

Syntax- primary key (column name, column name)

- e.g. create table student (Roll_no number (3), name carchar2(20), Address varchar2(20), primary key (Rollno));

you can also use the alter table command to set the primary key.

SQL PRIMARY KEY Constraint

- The PRIMARY KEY constraint uniquely identifies each record in a table.
- Primary keys must contain UNIQUE values, and cannot contain NULL values.
- A table can have only one primary key, which may consist of single or multiple fields.

SQL PRIMARY KEY on CREATE TABLE

- The following SQL creates a PRIMARY KEY on the "ID" column when the "Persons" table is created:

```
CREATE TABLE Persons (
   ID int NOT NULL PRIMARY KEY,
   LastName varchar(255) NOT NULL,
   FirstName varchar(255),
   Age int
);
```

DROP a PRIMARY KEY Constraint

- To drop a PRIMARY KEY constraint, use the following SQL:

```
ALTER TABLE Persons
DROP CONSTRAINT PK_Person
```

Not null constraints:

- When column is defined as not null then that column becomes compulsory column. It implies that a value must be inserted into the column if record is accepted in the table.
- By default, a column can hold NULL values.
- The NOT NULL constraint enforces a column to NOT accept NULL values.
- This enforces a field to always contain a value, which means that you cannot insert a new record, or update a record without adding a value to this field.
-

Syntax : column_name data type (size) Not null)

```
e.g. create table student (Roll.no number (4), name varchar2(20) Not null, Address
carchar2(20));
```

Following is the example of Primary KEY, not NULL AND default:

```
SQL> create table stu (rolno number (3) primary key, Name varchar (10) not null, city varchar (10) d
efault 'Latur');

Table created.
```

In the following example you can observe that while inserting the values in table we are not assigning a value to city. Which is by default we set while creating a table so that default value will be displayed in city column. But if any one want to add another city name in that case he can insert the name of another city.

```
SQL> insert into stu (rolno, name) values(2,'amar');

1 row created.

SQL> select *FROM STU;

     ROLNO NAME       CITY
---------- ---------- ----------
         1 vimal
         2 amar       Latur
```

Unique Key

Unique constraints applied on a column when we want to store the unique value i.e. duplicate values are eliminated in column. A table can have more than one unique key which is not possible in primary key unique can combine upto 16 columns in a composite unique key.

- Column level unique key constraints

Syntax :

column_name data type (size) Unique)

e.g. create table student (Roll_no number(4) unique, name varchar (10), Address varchar2(10));

```
SQL> create table stu2 (SID number(2) not null unique,
  2  fname varchar (10),
  3  lname varchar (10));

Table created.

SQL> insert into stu2 values(11,'amar','shinde');

1 row created.
```

In above example we create a table stu2 and assign a unique constraint to SID column. As we insert the value into STU2 table notice that if we enter the same Number in SID field then it gives the error "Unique constraint violated". Unique constraints ensures that duplicate value does not entered into table while inserting values in that table.

```
SQL> insert into stu2 values (11,'amar','shinde');
insert into stu2 values (11,'amar','shinde')
*
ERROR at line 1:
ORA-00001: unique constraint (SCOTT.SYS_C005499) violated
```

Check constraint:

- The CHECK constraint is used to limit the value range that can be placed in a column.
- If you define a CHECK constraint on a single column it allows only certain values for this column.
- If you define a CHECK constraint on a table it can limit the values in certain columns based on values in other columns in the row.
- In the given example while creating table STU1 CHECK constraint is applied on ID column. Check constraint checks ID number should always greater than 0.

```
SQL> CREATE TABLE STU1 (ID NUMBER(2) NOT NULL PRIMARY KEY CHECK (ID>0), LNAME VARCHAR (10));

Table created.
```

If we insert the value 0 in ID column then error occurs that “Check Constraint is violated.”

```
SQL> INSERT INTO STU1 VALUES(1,'AMAR');

1 row created.

SQL> INSERT INTO STU1 VALUES(0,'ANANT'
  2  );
INSERT INTO STU1 VALUES(0,'ANANT'
*
ERROR at line 1:
ORA-02290: check constraint (SCOTT.SYS_C005496) violated
```

Foreign Key:

- Unique constraints applied on a column when we want to store the unique value i.e. duplicate values are eliminated in column. A table can have more than one unique key which is not possible in primary key unique can combine upto 16 columns in a composite unique key.
- It rejects of value if corresponding value does not currently exists in the master key table.
- A foreign key must refer a primary key or unique key column from primary table. It will reference to the primary key master table, if no column/group of column specified when creating a foreign key. It requires that foreign key column have matching data type.
- A FOREIGN KEY is a key used to link two tables together.
- A FOREIGN KEY is a field (or collection of fields) in one table that refers to the PRIMARY KEY in another table.
- The table containing the foreign key is called the child table, and the table containing the candidate key is called the referenced or parent table.
- Look at the following two tables:

e.g. : create master table

```
SQL>create table main  (Roll no number(3) primary key, name varchar(10), Address varchar (20));
```

Create child table with references of master

```
Create table child(Roll number(3) references main, Ph_no number(10), city varchar(20));
```

Consider following example:

Table Name: STU1

ID	Name
1	AMAR
3	MINAL

4	RAJESH

Table Name: STU12

ROLNO	MARKS	ID
1	20	1
3	19	3
4	18	4

- Note that ID column in STU12 table points to the ID column of STU1 table.
- ID column in the STU1 table is the PRIMARY KEY
- ID column in the STU12 table is a FOREIGN KEY.
- The FOREIGN KEY constraint also prevents that invalid data from being inserted into the foreign key column, because it has to be one of the values contained in the table it points to.

Creating a Master table:

```
SQL> CREATE TABLE STU1 (ID NUMBER(2) NOT NULL PRIMARY KEY CHECK (ID>0), LNAME VARCHAR (10));

Table created.
```

Creating a child table:

```
SQL> create table stu12
  2  (
  3  rolno number (2) primary key,
  4  marks number (2),
  5  ID number (2),
  6  foreign key (ID)
  7  references stu1(ID)
  8  );

Table created.
```

Insert the values into STU1 table and display all records.

```
SQL> insert into stu1 values(3,'Minal');

1 row created.

SQL> insert into stu1 values(4,'rajesh');

1 row created.

SQL> select *from stu1;

        ID LNAME
---------- ----------
         1 AMAR
         3 Minal
         4 rajesh
```

Now insert the values into STU12 table but note that the ID no present in Master table for only those ID numbers we can insert the values in child table.

```
SQL> insert into stu12 values(1,20,1);

1 row created.

SQL> insert into stu12 values(3,19,3);

1 row created.

SQL> insert into stu12 values(4,18,4);

1 row created.

SQL> select *from stu12;

     ROLNO      MARKS         ID
---------- ---------- ----------
         1         20          1
         3         19          3
         4         18          4

SQL>
```

If we are trying to insert the values for rol no 2 which is not present in master table it will gives error "Integrity constraint is violated by the user and specified parent key is not found in the master table".

```
SQL> select *from stu12;

    ROLNO      MARKS         ID
--------- ---------- ----------
        1         20          1
        3         19          3
        4         18          4

SQL> insert into stu12 values(2,11,2);
insert into stu12 values(2,11,2)
*
ERROR at line 1:
ORA-02291: integrity constraint (SCOTT.SYS_C005501) violated - parent key not
found
```

DROP a FOREIGN KEY Constraint

To drop a FOREIGN KEY constraint, use the following SQL:

```
ALTER TABLE Orders
DROP CONSTRAINT FK_PersonOrder;
```