|  |
| --- |
| Department of Software Engineering  Mehran University of Engineering and Technology, Jamshoro |

|  |  |  |  |
| --- | --- | --- | --- |
| Course: SW222 – Database Management & Administration | | | |
| Instructor | Ms Shafiya Qadeer | **Practical/Lab No.** | 10 |
| Date | 25-02-2021 | **CLOs** | 2 |
| Signature |  | **Assessment Score** | 2 Marks |

|  |  |
| --- | --- |
| Topic | To become familiar with constraints enforcement |
| Objectives | * To become familiar with Constraints enforcement * To become familiar with Constraint retrieval from Data Dictionary |

|  |
| --- |
| Lab Discussion: Theoretical concepts and Procedural steps |

**Constraints**

Oracle Database lets you create six types of constraints and lets you declare them in two ways.

You can define constraints syntactically in two ways:

1. As part of the definition of an individual column or attribute. This is called inline specification.
2. As part of the table definition. This is called out-of-line specification.
3. **NOT NULL**

A NOT NULL constraint prohibits a database value from being null.

**Restrictions on NOT NULL Constraints**

NOT NULL constraints are subject to the following restrictions:

* You cannot specify NULL or NOT NULL in a view constraint.
* You cannot specify NULL or NOT NULL for an attribute of an object. Instead, use a CHECK constraint with the IS [NOT] NULL condition.

**Example:**

1. CREATE TABLE students (col\_name col\_type NOT NULL);

2. ALTER TABLE Table\_name

MODIFY (column\_name CONSTRAINT constraint\_name NOT NULL);

1. **Unique Constraints**

A unique constraint prohibits multiple rows from having the same value in the same column or combination of columns but allows some values to be null.

**Restrictions on Unique Constraints**

Unique constraints are subject to the following restrictions:

* None of the columns in the unique key can be of LOB, LONG, LONG RAW, VARRAY, NESTED TABLE, OBJECT, REF, TIMESTAMP WITH TIME ZONE, or user-defined type. However, the unique key can contain a column of TIMESTAMP WITH LOCAL TIME ZONE.
* A composite unique key cannot have more than 32 columns.
* You cannot designate the same column or combination of columns as both a primary key and a unique key.
* You cannot specify a unique key when creating a subview in an inheritance hierarchy. The unique key can be specified only for the top-level (root) view.

**Example:**

1. CREATE TABLE students (col\_name col\_type UNIQUE);

2. ALTER TABLE Table\_name

MODIFY (column\_name CONSTRAINT constraint\_name UNIQUE);

1. **Primary Key Constraints**

A primary key constraint combines a NOT NULL constraint and a unique constraint in a single declaration. That is, it prohibits multiple rows from having the same value in the same column or combination of columns and prohibits values from being null.

A primary key constraint combines a NOT NULL and unique constraint in one declaration. Therefore, to satisfy a primary key constraint:

**Restrictions on Primary Key Constraints**

Primary constraints are subject to the following restrictions:

* A table or view can have only one primary key.
* None of the columns in the primary key can be LOB, LONG, LONG RAW, VARRAY, NESTED TABLE, BFILE, REF, TIMESTAMP WITH TIME ZONE, or user-defined type. However, the primary key can contain a column of TIMESTAMP WITH LOCAL TIME ZONE.
* The size of the primary key cannot exceed approximately one database block.
* A composite primary key cannot have more than 32 columns.
* You cannot designate the same column or combination of columns as both a primary key and a unique key.
* You cannot specify a primary key when creating a subview in an inheritance hierarchy. The primary key can be specified only for the top-level (root) view.

**Example:**

1. CREATE TABLE table\_name (col\_name col\_type PRIMARY KEY);
2. CREATE TABLE table\_name (col\_name col\_type, col\_name col\_type,…, CONSTRAINT constraint\_name PRIMARY KEY (column\_name));
3. ALTER TABLE table\_name

ADD CONSTRAINT constraint\_name PRIMARY KEY (col\_name, col\_name,…);

1. **Foreign Key Constraints**

A foreign key constraint requires values in one table to match values in another table.

A foreign key constraint (also called a referential integrity constraint) designates a column as the foreign key and establishes a relationship between that foreign key and a specified primary or unique key, called the referenced key. A composite foreign key designates a combination of columns as the foreign key.

The table or view containing the foreign key is called the child object, and the table or view containing the referenced key is called the parent object. The foreign key and the referenced key can be in the same table or view. In this case, the parent and child tables are the same. If you identify only the parent table or view and omit the column name, then the foreign key automatically references the primary key of the parent table or view. The corresponding column or columns of the foreign key and the referenced key must match in order and datatype.

You can define a foreign key constraint on a single key column either inline or out of line. You must specify a composite foreign key and a foreign key on an attribute out of line.

To satisfy a composite foreign key constraint, the composite foreign key must refer to a composite unique key or a composite primary key in the parent table or view, or the value of at least one of the columns of the foreign key must be null.

**Restrictions on Foreign Key Constraints**

Foreign key constraints are subject to the following restrictions:

* None of the columns in the foreign key can be of LOB, LONG, LONG RAW, VARRAY, NESTED TABLE, BFILE, REF, TIMESTAMP WITH TIME ZONE, or user-defined type. However, the primary key can contain a column of TIMESTAMP WITH LOCAL TIME ZONE.
* The referenced unique or primary key constraint on the parent table or view must already be defined.
* A composite foreign key cannot have more than 32 columns.
* If either the child or parent object is a view, then the constraint is subject to all restrictions on view constraints.
* You cannot define a foreign key constraint in a CREATE TABLE statement that contains an AS subquery clause. Instead, you must create the table without the constraint and then add it later with an ALTER TABLE statement.

**Example:**

CREATE TABLE table\_name (col\_name col\_type,

col\_name CONSTRAINT constraint\_name

REFERENCES table\_name\_which\_has\_that\_key\_as\_primary\_key(column\_name) );

1. **Check Constraints**

A check constraint requires a value in the database to comply with a specified condition.

A check constraint lets you specify a condition that each row in the table must satisfy. To satisfy the constraint, each row in the table must make the condition either TRUE or unknown (due to a null). When Oracle evaluates a check constraint condition for a particular row, any column names in the condition refer to the column values in that row.

The syntax for inline and out-of-line specification of check constraints is the same. However, inline specification can refer only to the column (or the attributes of the column if it is an object column) currently being defined, whereas out-of-line specification can refer to multiple columns or attributes.

Oracle does not verify that conditions of check constraints are not mutually exclusive. Therefore, if you create multiple check constraints for a column, design them carefully so their purposes do not conflict. Do not assume any particular order of evaluation of the conditions.

**Example:**

1. CREATE TABLE students (col\_name col\_type CONSTRAINT constraint\_name CHECK (Expression));

**Oracle Data Dictionary**

The Oracle data dictionary is one of the most important components of the Oracle DBMS. It contains all information about the structures and objects of the database such as

• tables,

• columns,

• users,

• data files etc.

The data stored in the data dictionary are also often called metadata. Although it is usually the domain of database administrators (DBAs), the data dictionary is a valuable source of information for end users and developers.

The data dictionary consists of two levels:

* The internal level contains all base tables that are used by the various DBMS software components and they are normally not accessible by end users.
* The external level provides numerous views on these base tables to access information about objects and structures at different levels of detail.

**Data Dictionary Contains**

The tables and views provided by the data dictionary contain information about

* users and their privileges,
* tables, table columns and their data types, integrity constraints, indexes,
* statistics about tables and indexes used by the optimizer,
* privileges granted on database objects,
* storage structures of the database.

For viewing all constraints

select \* from ALL\_CONSTRAINTS;

|  |
| --- |
| Lab Tasks |

1. Create a table named BOOKS that will have 4 columns as:
   1. Book\_ID with NUMBER data type. (Also PK)
   2. Book\_Name with varchar data type and length 30.
   3. Author\_ID with NUMBER data type.
   4. Book\_Isbn with varchar data type and length 30.
2. Add not null on Book\_Name
3. Add check on Book\_Isbn so that isbn is not null and unique
4. Create a table named AUTHORS that will have 3 columns as:
   1. Author\_ID with NUMBER data type. (Also PK)
   2. Author\_Name with varchar data type and length 30.
   3. Author\_Publications with varchar data type and length 20.
5. Add a foreign key between table BOOKS and table AUTHORS