Created by Ahsan Arif

Check Constraints

A **check constraint** allows you to specify a condition on each row in a table.

Note:

- A check constraint can NOT be defined on a VIEW.
- The check constraint defined on a table must refer to only columns in that table. It can not refer to columns in other tables.
- A check constraint can NOT include a SUBQUERY.

A check constraint can be defined in either a CREATE TABLE statement or an ALTER TABLE statement.

Using a CREATE TABLE statement

The syntax for creating a check constraint using a CREATE TABLE statement is:

```
CREATE TABLE table_name
(column1 datatype null/not null,
column2 datatype null/not null,
...
CONSTRAINT constraint_name CHECK (column_name condition) [DISABLE]
);
```

The DISABLE keyword is optional. If you create a check constraint using the DISABLE keyword, the constraint will be created, but the condition will not be enforced.

For example:

```
CREATE TABLE suppliers
( supplier_id      numeric(4),
  supplier_name varchar2(50),
  CONSTRAINT check_supplier_id
  CHECK (supplier_id BETWEEN 100 and 9999)
);
```

In this first example, we've created a check constraint on the suppliers table called check_supplier_id. This constraint ensures that the supplier_id field contains values between 100 and 9999.

```
CREATE TABLE suppliers
( supplier_id      numeric(4),
  supplier_name varchar2(50),
  CONSTRAINT check_supplier_name
  CHECK (supplier_name = upper(supplier_name))
);
```

In this second example, we've created a check constraint called check_supplier_name. This constraint ensures that the supplier_name column always contains uppercase characters.

Using an ALTER TABLE statement

The syntax for creating a check constraint in an ALTER TABLE statement is:

```
ALTER TABLE table_name
add CONSTRAINT constraint_name CHECK (column_name condition) [DISABLE];
```

The DISABLE keyword is optional. If you create a check constraint using the DISABLE keyword, the constraint will be created, but the condition will not be enforced.

For example:

```
ALTER TABLE suppliers
add CONSTRAINT check_supplier_name
   CHECK (supplier_name IN ('IBM', 'Microsoft', 'NVIDIA'));
```

In this example, we've created a check constraint on the existing suppliers table called check_supplier_name. It ensures that the supplier_name field only contains the following values: IBM, Microsoft, or NVIDIA.

Drop a Check Constraint

The syntax for dropping a check constraint is:

```
ALTER TABLE table_name
drop CONSTRAINT constraint_name;
```

For example:

```
ALTER TABLE suppliers
drop CONSTRAINT check_supplier_id;
```

In this example, we're dropping a check constraint on the suppliers table called check_supplier_id.

Enable a Check Constraint

The syntax for enabling a check constraint is:

```
ALTER TABLE table_name
enable CONSTRAINT constraint_name;
```

For example:

```
ALTER TABLE suppliers
enable CONSTRAINT check_supplier_id;
```

In this example, we're enabling a check constraint on the suppliers table called check_supplier_id.

Disable a Check Constraint

The syntax for disabling a check constraint is:

```
ALTER TABLE table_name
disable CONSTRAINT constraint_name;
```

For example:

```
ALTER TABLE suppliers
disable CONSTRAINT check_supplier_id;
```

In this example, we're disabling a check constraint on the suppliers table called check_supplier_id.