ITC 5104 RELATIONAL DATABASE DESIGN AND SQL

Lecture 4

Chapter 4 Oracle 12c: SQL

Constraints

Objectives

- Explain the purpose of constraints in a table
- Distinguish among PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK, and NOT NULL constraints and understand the appropriate use of each constraint
- Understand how constraints can be created when creating a table or when modifying an existing table
- When creating a table distinguish between creating constraints at the column level and at the table level
- Create PRIMARY KEY constraints for a single column and a composite primary key

Objectives

- Create a FOREIGN KEY constraint
- Create a UNIQUE constraint
- Create a NOT NULL constraint using the ALTER TABLE … MODIFY command
- Include constraints during table creation
- Use the ENABLE and DISABLE commands
- Use the DROP command

Introduction

- In the previous chapter, we learned how to create tables using SQL commands
- In this chapter, we will look at how to add constraints to existing tables and how to include constraints during the table creation process
- Constraints are rules used to enforce business rules, practices and policies
- Constraints can ensure the accuracy and integrity of data by preventing errors from being entered into a database. Constraints can specify rules; data cannot be added into tables if the it violates these rules

Introduction

CONSTRAINT	DESCRIPTION
PRIMARY KEY	Determines which column(s) uniquely identifies each record. The primary key cannot be NULL, and the data value(s) must be unique.
FOREIGN KEY	In a one-to-many or parent-child relationship, the constraint is added to the "many" table. The constraint ensures that if a value is entered into a specified column, it must already exist in the "one" table, or the record is not added.
UNIQUE	Ensures that all data values stored in a specified column are unique. The UNIQUE constraint differs from the PRIMARY KEY constraint in that it allows NULL values.
CHECK	Ensures that a specified condition is true before the data value is added to a table. For example, an order's ship date cannot be earlier than its order date.
NOT NULL	Ensures that a specified column cannot contain a NULL value. The NOT NULL constraint can be created *only* with the column-level approach to table creation.

FIGURE 4-1 List of constraint types

Creating Constraints

- Constraints can be added during table creation as part of the CREATE TABLE command
- Constraints can also be added after the table has been created using the ALTER TABLE command
- When creating a *constraint* you have two options:
  1. *Name the constraint using the same rules as for naming tables and columns*
  2. *Omit the constraint name and allow Oracle 10g to generate a name for the constraint*

Creating Constraints

- If Oracle 11g server names the constraint, it will follow the format of *SYS_Cn* where n is a unique numeric value assigned to make the name unique
- It is always good practice to provide your own name for a constraint
- It will allow you to identify it easily in the future
- Industry convention for creating a constraint name is:

  tablename_columnname_constraintype
- The constraint type is an abbreviation to identify the type of constraint as shown on the next slide

Creating Constraints

CONSTRAINT	ABBREVIATION
PRIMARY KEY	_pk
FOREIGN KEY	_fk
UNIQUE	_uk
CHECK	_ck
NOT NULL	_nn

FIGURE 4-2 Constraint abbreviations

There are two ways to create a constraint when creating a table; at the column level or at the table level

Creating the Constraint at the Column Level

- When you create *constraints* at the *column level,* the constraint being created applies to the specific column
- The optional CONSTRAINT keyword is used if you want to give the constraint a specific name
- The *constraint type* uses the uses the following keywords to identify the type of constraint being created:
  - PRIMARY KEY
  - FOREIGN KEY
  - UNIQUE
  - CHECK
  - NOT NULL

Creating the Constraint at the Column Level

- If the *constraint* applies to more than one column, the *constraint* must be created at the table level
- The general syntax for creation at the column level is shown on the next slide
- The NOT NULL *constraint* can only be created at the column level

Creating the Constraint at the Column Level

```
columnname [CONSTRAINT constraintname] constrainttype,
```

FIGURE 4-3 Syntax for creating a column-level constraint

Using the PRIMARY KEY Constraint

- A PRIMARY KEY constraint is used to enforce the primary key requirements for a table
- A table can be created, as we have seen, without specifying a primary key
- The PRIMARY KEY constraint will make certain that the *column(s) identified as the table's primary key is unique and does not contain any null values*

Primary Key Constraints – Column Level

Worksheet | Query Builder

```
1  CREATE TABLE location
2  ( loc_id       NUMBER(6) CONSTRAINT location_loc_id_pk PRIMARY KEY,
3  bldg_code      VARCHAR2(10),
4  room           VARCHAR2(10),
5  capacity       NUMBER(3));
```

Create a table called LOCATION and specify LOC_ID as the primary key, it will use a NUMBER datatype:

Script Output

Task completed in 0.094 seconds

```
table LOCATION created.
```

Creating the Constraint at the Table Level

- When a constraint is created at the table level at the same time the table is created, the constraint definition is separate from the column definitions. It is listed after all the columns definitions
- The two differences in the syntax of a *column-level constraint* and a *table-level constraint* are that the column name for a *table-level constraint* is :
  - At the end of the constraint definition rather than at the beginning
- Again, the only constraint that **must** be created at the column level is NOT NULL

Creating the Constraint at the Table Level

```
[CONSTRAINT constraintname] constrainttype
(columnname, ...),
```

FIGURE 4-4 Syntax for creating a table-level constraint

Primary Key Constraints – Table Level

Worksheet | Query Builder

```
CREATE TABLE location
( loc_id        NUMBER(6),
bldg_code       VARCHAR2(10),
room            VARCHAR2(10),
capacity        NUMBER(3),
CONSTRAINT location_loc_id_pk PRIMARY KEY (loc_id));
```

An alternate method to create the same table:

Script Output

Task completed in 0.062 seconds

```
table LOCATION created.
```

Now, we'll move on to the Alter Table technique, where we will add constraints to tables that already exist

Adding the PRIMARY KEY Constraint After Table is Created

```
ALTER TABLE tablename
ADD [CONSTRAINT constraintname] PRIMARY KEY (columnname);
```

FIGURE 4-5 Syntax of the ALTER TABLE command to add a PRIMARY KEY constraint

Adding the PRIMARY KEY Constraint After Table is Created

Worksheet | Query Builder

```
CREATE TABLE location
( loc_id        NUMBER(6),
bldg_code       VARCHAR2(10),
room            VARCHAR2(10),
capacity        NUMBER(3));

ALTER TABLE location
ADD CONSTRAINT location_loc_id_pk PRIMARY KEY (loc_id);
```

Script Output

Task completed in 0.141 seconds

```
table LOCATION created.
table LOCATION altered.
```

Table is created as a separate command

The CONSTRAINT is added to the table after the table has been created, with the ALTER TABLE command

Command to add a primary key constraint to the PROMOTION table. It was successful, as indicated above

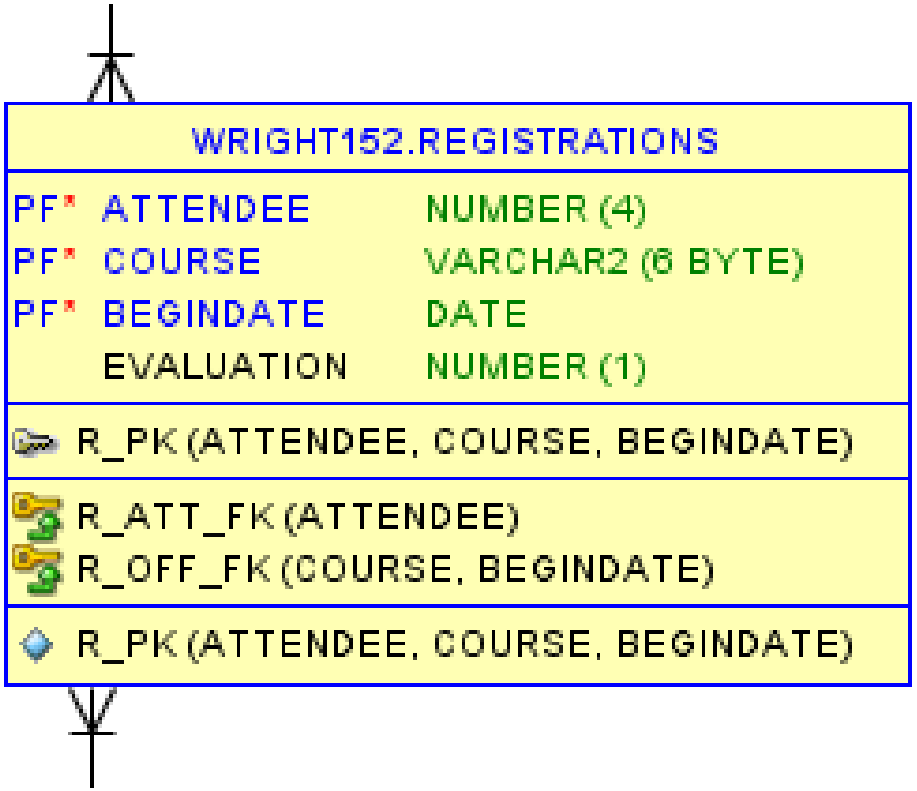
Adding the PRIMARY KEY Constraint After Table is Created

- The ADD CONSTRAINT clause instructs Oracle to *add a constraint* to the LOCATION table, this is part of the ALTER TABLE command
- The user has chosen the constraint name, location_loc_id_pk, rather than having it assigned by Oracle
- The final portion of the command identifies the type of *constraint* being created and the column it is being created on, PRIMARY KEY(loc_id)

The PRIMARY KEY Constraint

- Only one PRIMARY KEY constraint can be created for each table
- If the primary key is a *composite primary key,* it may only be created at the *table level*
- It may also be created with the ALTER TABLE command
- The columns used in the PRIMARY KEY must always be a *unique combination* in the table and neither value can be NULL

Composite Primary Key



It can be added at the table level or with the ALTER TABLE command

In some cases a single value cannot be used as a PRIMARY KEY

If we look at the REGISTRATIONS table we notice that it requires 3 columns to uniquely define each row

ATTENDEE, COURSE and BEGINDATE

Since there are three values it cannot be added at the column level

Composite Primary Key

Worksheet | Query Builder

```
CREATE TABLE registrations1
(attendee      NUMBER(4),
course         VARCHAR2(6),
BEGINDATE      DATE,
evaluation     NUMBER(1),
CONSTRAINT registrations1_pk PRIMARY KEY(attendee, course, begindate));
```

Script Output

Task completed in 0.063 seconds

```
table REGISTRATIONS1 created.
```

In this case the PRIMARY KEY is added as a table level constraint with all three columns that make up the primary key defined

Composite Primary Key

Worksheet | Query Builder

```
CREATE TABLE registrations1
(attendee      NUMBER(4),
course         VARCHAR2(6),
BEGINDATE      DATE,
evaluation     NUMBER(1));

ALTER TABLE registrations1
ADD CONSTRAINT registrations1_pk PRIMARY KEY(attendee, course, begindate);
```

Script Output

Task completed in 0.109 seconds

```
table REGISTRATIONS1 created.
table REGISTRATIONS1 altered.
```

The above command adds a composite primary key to the registrations1 table with the ALTER TABLE command

Displaying Constraints

- We can make sure that the two constraints we have created actually exist, by using the data dictionary
- The USER_CONSTRAINTS view will contain a list of all constraints that exist for your tables
- You will notice some have system defined names beginning with SYS_C, *what type of constraints are these?*
- What about the two we just created - *what type of constraints are they?*

Displaying Constraints

Worksheet | Query Builder

```
1  SELECT constraint_name
2  FROM user_constraints;
```

Query Result

SQL | All Rows Fetched: 79 in 4.025 seconds

	CONSTRAINT_NAME
1	H_EMPNO_FK
2	R_ATT_FK
3	O_TRAIN_FK
4	D_MGR_FK
5	E_MGR_FK
6	H_DEPT_FK
7	E_DEPT_FK
8	O_COURSE_FK
9	R_OFF_FK
10	SALES_CUST_ID_FK
11	CUSTOMERS_CUST_ID_PK
12	SYS_C0063718
13	SYS_C0063717
14	SYS_C0063716
15	SYS_C0063715
16	SYS_C0063714
17	SYS_C0063713
18	SYS_C0063712
19	SYS_C0063711
20	SYS_C0063710
21	SYS_C0063709

- The constraint names we have used easily explain the table, column and constraint type. The others, who knows???
- SYS_C0063718 or SYS_C0063717, I know they are constraints but I do not know anything else, the system ahs named these, I can find out what they are specifically used for but I have to search
- Using user-named constraints is much more effective

What Does the PRIMARY KEY Constraint DO?

Worksheet | Query Builder

```
INSERT INTO location
VALUES (1, 'E', '417', 50);

INSERT INTO location
VALUES (2, 'E', '422', 50);

INSERT INTO location
VALUES (3, 'J', '132', 50);

INSERT INTO location
VALUES (4, 'J', '140', 50);

INSERT INTO location
VALUES (5, 'J', '130', 40);

COMMIT;
```

Script Output

Task completed in 0.328 seconds

```
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.
committed.
```

- The following five rows were inserted to the LOCATION table
- The first value listed, 1 through 5 is the PRIMARY KEY value

What Does the PRIMARY KEY Constraint DO?

Worksheet | Query Builder

```
1  INSERT INTO location
2  VALUES (1, 'E', '417', 50);
3
4  COMMIT;
```

Script Output

Task completed in 0.203 seconds

```
Error starting at line : 1 in command -
INSERT INTO location
VALUES (1, 'E', '417', 50)
Error report -
SQL Error: ORA-00001: unique constraint (WRIGHT152.LOCATION_LOC_ID_PK) violated
00001. 00000 -  "unique constraint (%s.%s) violated"
*Cause:    An UPDATE or INSERT statement attempted to insert a duplicate key.
           For Trusted Oracle configured in DBMS MAC mode, you may see
           this message if a duplicate entry exists at a different level.
*Action:   Either remove the unique restriction or do not insert the key.
```

- You will notice that additional record is being inserted to the LOCATION table
- It fails, why?
- You are being given a constraint violation on the PRIMARY KEY, the value of 1 already exists in the
- LOCATION table

Using the FOREIGN KEY Constraint

- Suppose that a new faculty person has been added to the FACULTY table and the location specified does not exist in the LOCATION table
- Should a user be allowed to create a faculty record for a location that does not exist in the LOCATION table?
- This problem can be prevented using a FOREIGN KEY constraint

Foreign Key Constraints

- To prevent a user from entering an order from a customer who does not have a record in the CUSTOMERS table, you can *create a constraint that compares every entry made into the Customer# column of the ORDERS table with all the customer numbers existing in the CUSTOMER table*
- Foreign key constraints can be declared using the same method as before:
  - As part of the declaration of a column
  - As part of the declaration of the table

Foreign Key Constraints

- Constraint declaration syntax independent of column declaration:

```
CONSTRAINT constraint_name
FOREIGN KEY (constraint_fieldname)
REFERENCES table_where_field_is_primary_key
(name_of_field_in_table_where_it_is_the_primary_key)
```

- Constraint declaration syntax as part of column declaration

```
CONSTRAINT constraint_name
REFERENCES table_where_field_is_primary_key
(name_of_field_in_table_where_it_is_the_primary_key)
```

Foreign Key Constraint

- The keywords FOREIGN KEY are used to identify a column that, if it contains a value, must match data contained in another table
- The name of the column identified as the foreign key is contained in parentheses after the FOREIGN KEY keywords
- The REFERENCES keyword refers to *referential integrity.* It is used to *identify the table and column* that must already contain the data to be entered

Foreign Key Constraint

- Table declaration:

```
CONSTRAINT faculty_loc_id_fk FOREIGN KEY (loc_id)
    REFERENCES location (loc_id)
```

- Column declaration:

```
loc_id  NUMBER(6) CONSTRAINT
    faculty_loc_id_fk
    REFERENCES location (loc_id)
```

- When creating a foreign key constraint, the table being referenced (that contains the primary key) must already exist, in other words the LOCATION table must already exist and LOC_ID must already be defined as the primary key

Foreign Key Constraint – Table Level

Worksheet Query Builder

```
CREATE TABLE faculty
(f_id        NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last       VARCHAR2(15),
f_first      VARCHAR2(15),
f_mi         CHAR(1),
loc_id       NUMBER(6),
f_phone      CHAR(4),
f_rank       CHAR(4),
f_pin        NUMBER(4),
CONSTRAINT faculty_loc_id_fk FOREIGN KEY (loc_id) REFERENCES location (loc_id));
```

Script Output

Task completed in 0.093 seconds

```
table FACULTY created.
```

A FOREIGN KEY constraint must reference a column that has already been designated as the primary key for the referenced table, notice multiple constraints being defined

Foreign Key Constraint – Column Level

Worksheet | Query Builder

```
CREATE TABLE faculty
(f_id        NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last       VARCHAR2(15),
f_first      VARCHAR2(15),
f_mi         CHAR(1),
loc_id       NUMBER(6) CONSTRAINT faculty_loc_id_fk REFERENCES location (loc_id),
f_phone      CHAR(4),
f_rank       CHAR(4),
f_pin        NUMBER(4));
```

Script Output

Task completed in 0.078 seconds

```
table FACULTY created.
```

Foreign Key Constraint – ALTER TABLE Command

```
ALTER TABLE tablename
ADD [CONSTRAINT constraintname] FOREIGN KEY (columnname)
REFERENCES referencedtablename (referencedcolumnname);
```

FIGURE 4-11 Syntax of the ALTER TABLE command to add a FOREIGN KEY constraint

Foreign Key Constraint – ALTER TABLE Command

Worksheet | Query Builder

```
CREATE TABLE faculty
(f_id          NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last         VARCHAR2(15),
f_first        VARCHAR2(15),
f_mi           CHAR(1),
loc_id         NUMBER(6),
f_phone        CHAR(4),
f_rank         CHAR(4),
f_pin          NUMBER(4));

ALTER TABLE faculty
ADD CONSTRAINT faculty_loc_id_fk FOREIGN KEY (loc_id) REFERENCES location(loc_id);
```

Script Output

Task completed in 0.125 seconds

```
table FACULTY created.
table FACULTY altered.
```

Foreign Key Constraint – ALTER TABLE Command

- The command instructs Oracle to create a FOREIGN KEY constraint on the Customer# column of the FACULTY table
- The constraint is called faculty_loc_id_fk
- The constraint ensures that the location number entered for every faculty person has a corresponding value in the loc_id column of the LOCATION table
- The “table FACULTY altered” message confirms the creation of the constraint on the table specified

Foreign Key Constraint – ALTER TABLE Command

- The syntax of the FOREIGN KEY constraint is more complex than the PRIMARY KEY constraint since it involves both tables in the constraint
- The LOCATION table is the referenced table, it is on the *one side* of the one-to-many relationship and the FACULTY table is on the *many side* of the relationship
- The LOCATION table is the *parent table* and the FACULTY table is the *child table*

Foreign Key Constraint – ALTER TABLE Command

- When a FOREIGN KEY constraint exists between two tables, by default, a *record cannot be deleted from the parent table if a matching entry exists in the child table*
- In other words you *cannot delete a record* from the LOCATION table if there are records in the FACULTY table for that location

Foreign Key Constraint

Worksheet | Query Builder

```
INSERT INTO faculty (f_id, f_last, f_first, loc_id)
VALUES (1, 'Que', 'Barbie', 1);

INSERT INTO faculty (f_id, f_last, f_first, loc_id)
VALUES (2, 'Beech', 'Snadi', 2);

INSERT INTO faculty (f_id, f_last, f_first, loc_id)
VALUES (3, 'O''Shea', 'Rick', 3);

COMMIT;
```

Query Result | Script Output

Task completed in 0.205 seconds

```
1 rows inserted.
1 rows inserted.
1 rows inserted.
committed.
```

- In the INSERT statement the loc_id column is the Foreign Key
- As long as its value is found in the parent table the rows will be inserted with no issue

Foreign Key Constraint

Worksheet | Query Builder

```
1  INSERT INTO faculty (f_id, f_last, f_first, loc_id)
2  VALUES (6, 'Waters', 'Muddy', 6);
3
4  COMMIT;
```

Query Result | Script Output

Task completed in 0.202 seconds

```
Error starting at line : 1 in command -
INSERT INTO faculty (f_id, f_last, f_first, loc_id)
VALUES (6, 'Waters', 'Muddy', 6)
Error report -
SQL Error: ORA-02291: integrity constraint (WRIGHT152.FACULTY_LOC_ID_FK) violated - parent key not found
02291. 00000 - "integrity constraint (%s.%s) violated - parent key not found"
*Cause:    A foreign key value has no matching primary key value.
*Action:   Delete the foreign key or add a matching primary key.
committed.
```

In this case the INSERT fails, the value of 6 is not in the LOCATION table, so the error tells you about this – “parent key not found”

Foreign Key Constraint

Worksheet | Query Builder

```
1  DELETE FROM location
2  WHERE loc_id = 1;
```

Query Result | Script Output

Task completed in 0.156 seconds

```
Error starting at line : 1 in command -
DELETE FROM location
WHERE loc_id = 1
Error report -
SQL Error: ORA-02292: integrity constraint (WRIGHT152.FACULTY_LOC_ID_FK) violated - child record found
02292. 00000 - "integrity constraint (%s.%s) violated - child record found"
*Cause:    attempted to delete a parent key value that had a foreign
           dependency.
*Action:   delete dependencies first then parent or disable constraint.
```

In this case an attempt was made to remove a record from the LOCATION table, an error occurred since LOC_ID of 1 in LOCATION has records related to that value of 1 in the FACULTY table

That record has a child in the FACULTY table so it cannot be removed

Foreign Key Constraint – ALTER TABLE Command

- If you do need to delete a customer from the LOCATION table, you must first remove all related records from the FACULTY table (*the child table),* and then delete the record from the LOCATION table (*the parent table)*
- If the location occurs many times in various faculty records this process could take a little time

Foreign Key Constraint

Worksheet | Query Builder

```
1  DROP TABLE location;
```

Query Result | Script Output

Task completed in 0.109 seconds

```
Error starting at line : 1 in command -
DROP TABLE location
Error report -
SQL Error: ORA-02449: unique/primary keys in table referenced by foreign keys
02449. 00000 -  "unique/primary keys in table referenced by foreign keys"
*Cause:    An attempt was made to drop a table with unique or
           primary keys referenced by foreign keys in another table.
*Action:   Before performing the above operations the table, drop the
           foreign key constraints in other tables. You can see what
           constraints are referencing a table by issuing the following
           command:
           SELECT * FROM USER_CONSTRAINTS WHERE TABLE_NAME = "tabnam";
```

- In this case an attempt was made to drop the LOCATION table
- This failed since there are matching values in the FACULTY table
- It will not allow you to orphan the child records

Foreign Key Constraint – ALTER TABLE Command

- There is another solution - to add the keywords ON DELETE CASCADE to the constraint definition
- If the ON DELETE CASCADE keywords are *added to the constraint definition and a record is deleted from the parent table, then any corresponding records in the child table are automatically deleted as well*

Foreign Key Constraint – ALTER TABLE Command

Worksheet | Query Builder

```
ALTER TABLE faculty
DROP  CONSTRAINT faculty_loc_id_fk;

ALTER TABLE faculty
ADD CONSTRAINT faculty_loc_id_fk FOREIGN KEY (loc_id)
REFERENCES location (loc_id) ON DELETE CASCADE;
```

Query Result | Script Output

Task completed in 0.125 seconds

```
table FACULTY altered.
table FACULTY altered.
```

Foreign Key Constraint – ALTER TABLE Command

- If a location who has faculty records using that location the entries are now deleted from the FACULTY table, then all the location those faculty members will now be deleted
- The ON DELETE CASCADE option is very dangerous since a user may not be aware that this option has been set. If a customer was deleted, then all the customer's orders would be deleted as well

ON DELETE CASCADE Option

Worksheet | Query Builder

```
DELETE FROM location
WHERE loc_id = 1;

SELECT *
FROM location;
```

Script Output | Query Result

SQL | All Rows Fetched: 4 in 0.047 seconds

	LOC_ID	BLDG_CODE	ROOM	CAPACITY
1	2	E	422	50
2	3	J	132	50
3	4	J	140	50
4	5	J	130	40

When we tried this previously on slide 42 we were prevented from removing this row

We are now permitted to remove the row, you can see that it is gone

See the next slide to see what else is gone …

ON DELETE CASCADE Option

Worksheet | Query Builder

```
1  DELETE FROM location
2  WHERE loc_id = 1;
3
4  SELECT *
5  FROM faculty;
```

You will notice that the F_ID value of 1 is now gone it had a value for LOC_ID of 1

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.047 seconds

	F_ID	F_LAST	F_FIRST	F_MI	LOC_ID	F_PHONE	F_RANK	F_PIN
1	2	Beech	Snadi	(null)	2	(null)	(null)	(null)
2	3	O'Shea	Rick	(null)	3	(null)	(null)	(null)

ON DELETE CASCADE Option

- This is the danger with the ON DELETE CASCADE option
- If you are not familiar with the action that will take place you may not realize you have just remove the child and the related parent
- As I mentioned dangerous option so be careful if you choose to use it

Foreign Key Constraint – ALTER TABLE Command

- Remember, if a NULL value is entered into a column that has a FOREIGN KEY constraint, the record will be accepted, this would mean that if an order were placed and no customer number was given, the order would be created
- If you wanted customer number to be mandatory, a NOT NULL constraint would have to be added to the customer# column as well

Using the UNIQUE Constraint

- The purpose of the UNIQUE constraint is to ensure that two records do not have the same value stored in the same column
- A UNIQUE constraint *will allow* NULL values, which are not permitted with a PRIAMRY KEY constraint

UNIQUE Constraint – Column Level

Worksheet | Query Builder

```
CREATE TABLE faculty
(f_id        NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last       VARCHAR2(15),
f_first      VARCHAR2(15),
f_mi         CHAR(1),
loc_id       NUMBER(6) CONSTRAINT faculty_loc_id_fk REFERENCES location(loc_id),
f_phone      CHAR(4),
f_rank       CHAR(4),
f_pin        NUMBER(4) CONSTRAINT faculty_f_pin_UK UNIQUE);
```

Script Output

Task completed in 0.094 seconds

```
table FACULTY created.
```

The column f_pin will now only accept unique values, also showing here how multiple constraints are added to the same table at the same time

UNIQUE Constraint – Table Level

Worksheet Query Builder

```
CREATE TABLE faculty
(f_id        NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last       VARCHAR2(15),
f_first      VARCHAR2(15),
f_mi         CHAR(1),
loc_id       NUMBER(6) CONSTRAINT faculty_loc_id_fk REFERENCES location(loc_id),
f_phone      CHAR(4),
f_rank       CHAR(4),
f_pin        NUMBER(4),
CONSTRAINT faculty_f_pin_UK UNIQUE (f_pin));
```

Script Output

Task completed in 0.062 seconds

```
table FACULTY created.
```

Constraint is added at the table level

Using the UNIQUE Constraint with ALTER TABLE

```
ALTER TABLE tablename
ADD [CONSTRAINT constraintname] UNIQUE (columnname);
```

FIGURE 4-17 Syntax for adding a UNIQUE constraint to a table

Using the UNIQUE Constraint with ALTER TABLE

Worksheet | Query Builder

```
ALTER TABLE faculty
DROP CONSTRAINT faculty_f_pin_uk;

ALTER TABLE faculty
ADD CONSTRAINT faculty_pin_uk UNIQUE (f_pin);
```

Script Output | Query Result

Task completed in 0.219 seconds

```
table FACULTY altered.
table FACULTY altered.
```

- In this case the constraint was first dropped then recreated using the ALTER TABLE command

Using the UNIQUE Constraint

- Once the command is successfully executed, Oracle will not allow an entry that would duplicate an existing entry into the F_PIN column of the FACULTY table

Using the CHECK Constraint

- A CHECK constraint requires that a specific condition be met before a record can be added to a table
- With a CHECK constraint, you can make sure a book's price is greater than 0, its retail price is less than $200.00, or a seller's commission rate is less than 50%
- You could check to make the sure the order date is earlier or equal to the ship date

Using the CHECK Constraint

- However, the condition cannot reference built-in functions such as SYSDATE, USER, or ROWNUM or refer to values stored in other rows. It **can** be compared to values within the same row
- So, you could not add a CHECK constraint that requires the ship date for an order to be the same as the current system date (since this would need the SYSDATE function)

Check Constraints

- An appropriate use is a gender code such as M or F
- An inappropriate use would be a colour code, since it could contain many values and could change. It would be best to handle this with another technique, such as placing colour in a separate table
- Check constraints must evaluate to TRUE or FALSE
- Logical operators AND and OR can be used
- Ranges are also permitted

CHECK Constraint – Column Level

Worksheet | Query Builder

```
CREATE TABLE faculty
(f_id        NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last       VARCHAR2(15),
f_first      VARCHAR2(15),
f_mi         CHAR(1),
loc_id       NUMBER(6) CONSTRAINT faculty_loc_id_fk REFERENCES location(loc_id),
f_phone      CHAR(4),
f_rank       CHAR(4) CONSTRAINT faculty_f_rank_ck CHECK ((f_rank = 'ASSO') OR (f_rank = 'FULL') OR
                                                         (f_rank = 'ASST') OR (f_rank = 'INST')),
f_pin        NUMBER(4) CONSTRAINT faculty_f_pin_UK UNIQUE );
```

Script Output

Task completed in 0.062 seconds

```
table FACULTY created.
```

CHECK Constraint – Table Level

Worksheet | Query Builder

```
CREATE TABLE faculty
(f_id        NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last       VARCHAR2(15),
f_first      VARCHAR2(15),
f_mi         CHAR(1),
loc_id       NUMBER(6) CONSTRAINT faculty_loc_id_fk REFERENCES location(loc_id),
f_phone      CHAR(4),
f_rank       CHAR(4),
f_pin        NUMBER(4) CONSTRAINT faculty_f_pin_UK UNIQUE,
CONSTRAINT faculty_f_rank_ck CHECK ((f_rank = 'ASSO') OR (f_rank = 'FULL') OR
                                    (f_rank = 'ASST') OR (f_rank = 'INST')));
```

Script Output

Task completed in 0.078 seconds

```
table FACULTY created.
```

Check Constraints

- Another option for listing constraints is to use the IN keyword instead of several OR's
- Repeating one of the previous examples:
  - We will address the FACULTY table again, the f_rank field will contain the same values as previous only this time we will use the IN operator to replace the multiple OR operators

```
CONSTRAINT faculty_f_rank_ck CHECK
(f_rank IN ('ASSO', 'FULL', 'ASST',
'INST'))
```

Check Constraints

Worksheet | Query Builder

```
CREATE TABLE faculty
(f_id        NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last       VARCHAR2(15),
f_first      VARCHAR2(15),
f_mi         CHAR(1),
loc_id       NUMBER(6) CONSTRAINT faculty_loc_id_fk REFERENCES location(loc_id),
f_phone      CHAR(4),
f_rank       CHAR(4),
f_pin        NUMBER(4) CONSTRAINT faculty_f_pin_UK UNIQUE,
CONSTRAINT faculty_f_rank_ck CHECK (f_rank IN ('ASSO', 'FULL', 'ASST','INST')));
```

Script Output

Task completed in 0.078 seconds

```
table FACULTY created.
```

Using the CHECK Constraint

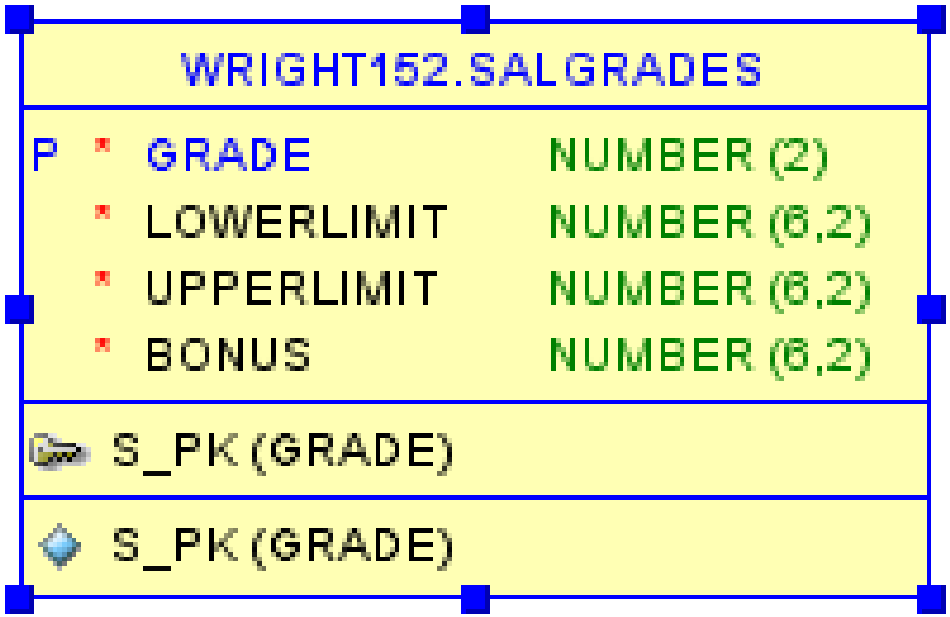
```
ALTER TABLE tablename
ADD [CONSTRAINT constraintname] CHECK (condition);
```

FIGURE 4-19 Syntax for adding a CHECK constraint to an existing table

Using the CHECK Constraint

- The syntax to add a CHECK constraint is the same as the syntax to add a PRIMARY KEY or UNIQUE constraint
- However, rather than simply list a column name for the constraint, the condition that must be satisfied is listed after the constraint type

Using the CHECK Constraint



In the SALGRADES table of our schema there are two columns LOWERLIMIT and UPPERLIMIT

If a row is inserted to the table we want ensure the value for the LOWERLIMIT is lees that the VALUE for the UPPERLIMIT

Using the CHECK Constraint

Worksheet | Query Builder

```
1  ALTER TABLE salgrades
2  ADD CONSTRAINT lower_upper_ck CHECK (upperlimit > lowerlimit);
```

Script Output | Query Result

Task completed in 0.078 seconds

```
table SALGRADES altered.
```

Using the CHECK Constraint

Worksheet | Query Builder

```
1  INSERT INTO salgrades
2  VALUES (6, 5000, 3000, 250);
```

Script Output | Query Result

Task completed in 0.156 seconds

```
Error starting at line : 1 in command -
INSERT INTO salgrades
VALUES (6, 5000, 3000, 250)
Error report -
SQL Error: ORA-02290: check constraint (WRIGHT152.LOWER_UPPER_CK) violated
02290. 00000 -  "check constraint (%s.%s) violated"
*Cause:    The values being inserted do not satisfy the named check

*Action:   do not insert values that violate the constraint.
```

Using the CHECK Constraint

- If any records already stored in the SALGRADES table violate the condition lowerlimit > upperlimit, Oracle will return an error message that the constraint has been violated and the ALTER TABLE command will fail
- If you get an error, you need to correct the records with the condition that violates the constraint
- How would you find the offending records?
  - SELECT * FROM orders WHERE lowerlimit > upperlimit;
- Fix the record(s) that appear, then reissue the ALTER TABLE command

Using the NOT NULL Constraint

- The NOT NULL constraint is actually a special CHECK constraint with the condition IS NOT NULL
- Basically, it prevents you from adding a row that contains a NULL value in the specified column
- The NOT NULL constraint, when created at the table creation time, can only be added as a column level constraint

Using the NOT NULL Constraint

- A field defined with a primary key automatically has a NOT NULL constraint applied to it
- If a field defined as a foreign key MUST have a value entered, then a NOT NULL constraint needs to be added to the table definition

NOT NULL Constraint – Column Level

Worksheet | Query Builder

```
CREATE TABLE faculty
(f_id        NUMBER(3) CONSTRAINT faculty_f_id PRIMARY KEY,
f_last       VARCHAR2(15) CONSTRAINT faculty_f_last_nn NOT NULL,
f_first      VARCHAR2(15) CONSTRAINT faculty_f_first_nn NOT NULL,
f_mi         CHAR(1),
loc_id       NUMBER(6) CONSTRAINT faculty_loc_id_fk REFERENCES location(loc_id),
f_phone      CHAR(4),
f_rank       CHAR(4) CONSTRAINT faculty_f_rank_ck CHECK (f_rank IN ('ASSO', 'FULL', 'ASST','INST')),
f_pin        NUMBER(4) CONSTRAINT faculty_f_pin_UK UNIQUE);
```

Script Output

Task completed in 0.078 seconds

```
table FACULTY created.
```

NOT NULL Constraint – Column Level

- When you do a DESCRIBE against the faculty table now, the two NOT NULL constraints that were just created show in the column NOT NULL
- This indicates that a value must be supplied to these columns when you insert a record

NOT NULL Constraint – Column Level

Worksheet | Query Builder

```
1  DESC faculty;
```

Script Output | Query Result

Task completed in 0.967 seconds

```
DESC faculty
Name     Null     Type
------- -------- ------------
F_ID     NOT NULL NUMBER(3)
F_LAST   NOT NULL VARCHAR2(15)
F_FIRST  NOT NULL VARCHAR2(15)
F_MI              CHAR(1)
LOC_ID            NUMBER(6)
F_PHONE           CHAR(4)
F_RANK            CHAR(4)
F_PIN             NUMBER(4)
```

ALTER TABLE to add the NOT NULL Constraint

- After the table has been created, you add a NOT NULL constraint using ALTER TABLE with the MODIFY command (rather than the ADD command)

ALTER TABLE to add the NOT NULL Constraint

```
ALTER TABLE tablename
MODIFY (columnname [CONSTRAINT constraintname]
NOT NULL);
```

FIGURE 4-22 Syntax for adding a NOT NULL constraint to an existing table

Using the NOT NULL Constraint

Worksheet | Query Builder

```
1  ALTER TABLE faculty
2  MODIFY (loc_id CONSTRAINT faculty_loc_id_nn NOT NULL);
```

Script Output | Query Result

Task completed in 0.078 seconds

```
table FACULTY altered.
```

A NOT NULL constraint is added to the loc_id column of the FACULTY table

This column had a previous FOREIGN KEY constraint on it, this will now make this a mandatory value when new data is inserted

Using the NOT NULL Constraint

Worksheet | Query Builder

```
INSERT INTO faculty (f_id, f_last, f_first, loc_id, f_rank)
VALUES  (100, 'Wright', 'Bill', NULL, 'FULL');
```

Script Output | Query Result

Task completed in 0.172 seconds

```
Error starting at line : 1 in command -
INSERT INTO faculty (f_id, f_last, f_first, loc_id, f_rank)
VALUES  (100, 'Wright', 'Bill', NULL, 'FULL')
Error report -
SQL Error: ORA-01400: cannot insert NULL into ("WRIGHT152"."FACULTY"."LOC_ID")
01400. 00000 -  "cannot insert NULL into (%s)"
*Cause:    An attempt was made to insert NULL into previously listed objects.
*Action:   These objects cannot accept NULL values.
```

LOC_ID can no longer be a NULL value, it will cause an error when you attempt to insert a NULL value

Constraints with No Name Specified

- A constraint can also be created without giving a proper name
- It is not advisable to do so
- It will complicate error messages since the name of your constraint appears in some error messages when you violate the conditions of a constraint
- We will create a table to demonstrate this

Using Unnamed Constraints

- You can add a NOT NULL constraint or a PRIMARY KEY constraint without specifying a name for the constraint
- Simply omit the CONSTRAINT keyword and the name and list the constraint type directly after the column name
- Actually all constraints could be created without constraint names

Using Unnamed Constraints

Worksheet | Query Builder

```
1 CREATE TABLE dept
2 (dept_id        NUMBER(2) PRIMARY KEY,
3 d_name          VARCHAR2(100) NOT NULL,
4 fax             VARCHAR2(12) UNIQUE);
```

Script Output x

Task completed in 0.078 seconds

```
table DEPT created.
```

The constraints are created without using any names. Oracle, in this case, would assign a name to the constraints (SYS_Cn)

Viewing Constraint Information

- In the data dictionary, Oracle stores information about constraints
- The data dictionary also includes information about objects such as tables, and information about users on the system
- We would like to use the data dictionary to view information on some of the constraints we have created

Viewing Constraint Information

- You can also click the table name in SQL Developer
- Then click the Constraints tab to see a listing of your constraints

Viewing Constraint Information

Columns | Data | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME	DELETE_RULE
1	FACULTY_F_FIRST_NN	Check	"F_FIRST" IS NOT NULL	(null)	(null)	(null)	(null)
2	FACULTY_F_ID	Primary_Key	(null)	(null)	(null)	(null)	(null)
3	FACULTY_F_LAST_NN	Check	"F_LAST" IS NOT NULL	(null)	(null)	(null)	(null)
4	FACULTY_F_PIN_UK	Unique	(null)	(null)	(null)	(null)	(null)
5	FACULTY_F_RANK_CK	Check	f_rank IN ('ASSO', 'FULL', 'ASST','INST')	(null)	(null)	(null)	(null)
6	FACULTY_LOC_ID_FK	Foreign_Key	(null)	WRIGHT152	LOCATION	LOCATION_LOC_ID_PK	NO ACTION
7	FACULTY_LOC_ID_NN	Check	"LOC_ID" IS NOT NULL	(null)	(null)	(null)	(null)

There are more columns that could not be shown effectively

This is the constraints we created on the FACULTY table

Notice the NOT NULL constraints appear as CHECK constraints with a SEARCH_CONDITION that IS NOT NULL

A FOREIGN KEY constraint will show the referencing table and the reference to the primary key of that table

Best Practices

- A NOT NULL constraint should not be assigned to a PRIMARY KEY column, it is a common error to do so, you will not receive an error but you are duplicating processing, so refrain from doing this
- A PRIMARY KEY enforces both NOT NULL and UNIQUE constraints
- CHECK, FOREIGN KRY and NOT NULL do allow NULL values to be inserted, a NOT NULL constraint must be used along with these constraints if you require a value to be inputted to the column
- If you assign a DEFAULT value to a column a NOT NULL constraint should not be used, since if no value is assigned the DEFAULT value is assigned

Viewing Constraints

Worksheet | Query Builder

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

Script Output | Query Result

SQL | All Rows Fetched: 7 in 1.903 seconds

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_CONSTRAINT_NAME
1	FACULTY_LOC_ID_FK	R	(null)	LOCATION_LOC_ID_PK
2	FACULTY_LOC_ID_NN	C	"LOC_ID" IS NOT NULL	(null)
3	FACULTY_F_LAST_NN	C	"F_LAST" IS NOT NULL	(null)
4	FACULTY_F_RANK_CK	C	f_rank IN ('ASSO', 'FULL', 'ASST','INST')	(null)
5	FACULTY_F_FIRST_NN	C	"F_FIRST" IS NOT NULL	(null)
6	FACULTY_F_PIN_UK	U	(null)	(null)
7	FACULTY_F_ID	P	(null)	(null)

Viewing Constraints

- On the previous slide, the constraint_name lists the names of the constraints on the FACULTY table
- The second column lists the type of constraint:
  - P – primary key
  - C – check or not null constraints
  - U – unique constraints
  - R – foreign key constraints (R stands for referential integrity, exhibited by foreign keys)
- The third column gives the conditions for check constraints, it is blank for a constraint that is not a check constraint
- The fourth column shows the referenced constraint for a foreign key, a foreign key must match the primary key of the related table
- The names are clear so you can actually see the table and column the constraint exists for and the type of constraint

Viewing Constraints

Worksheet | Query Builder

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

Script Output | Query Result

SQL | All Rows Fetched: 3 in 1.669 seconds

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_CONSTRAINT_NAME
1	SYS_C0064706	U	(null)	(null)
2	SYS_C0064705	P	(null)	(null)
3	SYS_C0064704	C	"D_NAME" IS NOT NULL	(null)

The constraint names shown in this case are not as clear since Oracle has named the constraints for you since you did not provide a name for them

Disabling and Dropping Constraints

- If a *constraint* exists for a column, every time an entry is made to that column, it must be evaluated to determine whether the value to be entered in the column violates the constraint or not
- If a large block of data is added to a table, the validation process can severely slow down the Oracle Server's processing speed
- If you are certain that the data being added adheres to the constraints, you can *disable the constraints* before you add the block of data to the table

Disabling and Dropping Constraints

- To disable a constraint, use the ALTER TABLE command with the DISABLE keyword
- At a later time, you can reverse this by using ALTER TABLE with the ENABLE keyword

Disabling and Dropping Constraints

Worksheet | Query Builder

```
ALTER TABLE faculty
DISABLE CONSTRAINT faculty_f_last_nn;

ALTER TABLE faculty
ENABLE CONSTRAINT faculty_f_last_nn;
```

Script Output

Task completed in 0.062 seconds

```
table FACULTY altered.
table FACULTY altered.
```

In order to disable or enable some constraints, I must know their name. Think about the SYS_Cn constraints if I provide no name!!!!!

You need to find the names

Disabling and Dropping Constraints

- If you create a constraint and then decide it is no longer required, or if you make an error during its initial creation and you wish to remove the constraint, you issue the DROP (constraint_name) command
- The DROP clause varies based on the type of constraint being dropped

Disabling and Dropping Constraints

- If the DROP references a PRIMARY KEY constraint for the table, then the key words PRIMARY KEY are sufficient since a table can only contain one primary key
- If the DROP references a UNIQUE constraint then only the column name affected by the constraint is required because a column can have only one UNIQUE constraint
- Any other type of constraint must be referenced by the constraint's actual name – *regardless of whether the constraint name was issued by the user or the Oracle Server*

Disabling and Dropping Constraints

Worksheet | Query Builder

```
1  ALTER TABLE dept
2  DROP PRIMARY KEY;
```

The name of the PRIMARY KEY constraint is not specified since a table can only have a single primary key.

Script Output x

Task completed in 0.078 seconds

```
table DEPT altered.
```

Disabling and Dropping Constraints

Worksheet | Query Builder

```
1  ALTER TABLE faculty
2  DROP UNIQUE (f_pin);
```

Script Output

Task completed in 0.078 seconds

```
table FACULTY altered.
```

The Constraint Name does not need to be provided when you drop a UNIQUE constraint

Disabling and Dropping Constraints

Worksheet | Query Builder

```
1  ALTER TABLE faculty
2  DROP CONSTRAINT faculty_f_first_nn
```

Script Output

Task completed in 0.047 seconds

```
table FACULTY altered.
```

In this case you need to know the name of the constraint, if you follow a naming convention you should not even have to search for the name of the constraint, think of the SYS_Cn constraint names????

Disabling and Dropping Constraints

- The last piece to look at is dropping a table with foreign key constraints
- If you attempt to drop a table that has a foreign key attached to it, the constraint will forbid you from removing the table
- The CASCADE option can be used to remove the table and any constraints attached to it
- This is seen on the next slide (LOCATION is used as a FOREIGN KEY in the FACULTY table)

Disabling and Dropping Constraints

Worksheet | Query Builder

```
1  ALTER TABLE location
2  DROP PRIMARY KEY;
```

Script Output

Task completed in 0.109 seconds

```
Error starting at line : 1 in command -
ALTER TABLE location
DROP PRIMARY KEY
Error report -
SQL Error: ORA-02273: this unique/primary key is referenced by some foreign keys
02273. 00000 -  "this unique/primary key is referenced by some foreign keys"
*Cause:    Self-evident.
*Action:   Remove all references to the key before the key is to be dropped.
```

The initial DROP failed since the LOCATION table has a FOREIGN KEY that is referencing it

The second was successful sine the CASCADE option was used to also drop the FOREIGN KEY that is referencing the PRIMARY KEY column

Disabling and Dropping Constraints

Worksheet | Query Builder

```
1  ALTER TABLE location
2  DROP PRIMARY KEY CASCADE;
```

Script Output

Task completed in 0.109 seconds

```
table LOCATION altered.
```

- The second was successful sine the CASCADE option was used to also drop the FOREIGN KEY that is referencing the PRIMARY KEY column

Disabling and Dropping Constraints

Worksheet | Query Builder

```
1  DROP TABLE location CASCADE CONSTRAINTS;
```

Script Output

Task completed in 0.063 seconds

```
table LOCATION dropped.
```

If you wish to drop a table that has foreign key constraints you can use the CASCADE CONSTRAINTS option with the DROP TABLE