ORACLE/PLSQL: ALTER TABLE STATEMENT

This Oracle tutorial explains how to use the Oracle **ALTER TABLE statement** to add a column, modify a column, drop a column, rename a column or rename a table (with syntax, examples and practice exercises).

DESCRIPTION

The Oracle ALTER TABLE statement is used to add, modify, or drop/delete columns in a table. The Oracle ALTER TABLE statement is also used to rename a table.

ADD COLUMN IN TABLE

Syntax

To ADD A COLUMN in a table, the Oracle ALTER TABLE syntax is:

```
ALTER TABLE table_name
  ADD column_name column-definition;
```

Example

Let's look at an example that shows how to add a column in an Oracle table using the ALTER TABLE statement.

For example:

```
ALTER TABLE customers
  ADD customer_name varchar2(45);
```

This Oracle ALTER TABLE example will add a column called *customer_name* to the *customers* table.

ADD MULTIPLE COLUMNS IN TABLE

Syntax

To ADD MULTIPLE COLUMNS to an existing table, the Oracle ALTER TABLE syntax is:

```
ALTER TABLE table_name
  ADD (column_1 column-definition,
       column_2 column-definition,
       ...
       column_n column_definition);
```

Example

Let's look at an example that shows how to add multiple columns in an Oracle table using the ALTER TABLE statement.

For example:

```
ALTER TABLE customers
  ADD (customer_name varchar2(45),
       city varchar2(40));
```

This Oracle ALTER TABLE example will add two columns, *customer_name* as a varchar2(45) field and *city* as a varchar2(40) field to the *customers* table.

MODIFY COLUMN IN TABLE

Syntax

To MODIFY A COLUMN in an existing table, the Oracle ALTER TABLE syntax is:

```
ALTER TABLE table_name
  MODIFY column_name column_type;
```

Example

Let's look at an example that shows how to modify a column in an Oracle table using the ALTER TABLE statement.

For example:

```
ALTER TABLE customers
  MODIFY customer_name varchar2(100) not null;
```

This Oracle ALTER TABLE example will modify the column called *customer_name* to be a data type of varchar2(100) and force the column to not allow null values.

MODIFY MULTIPLE COLUMNS IN TABLE

SyntaX

To MODIFY MULTIPLE COLUMNS in an existing table, the Oracle ALTER TABLE syntax is:

```
ALTER TABLE table_name
  MODIFY (column_1 column_type,
          column_2 column_type,
          ...
          column_n column_type);
```

Example

Let's look at an example that shows how to modify multiple columns in an Oracle table using the ALTER TABLE statement.

For example:

```
ALTER TABLE customers
  MODIFY (customer_name varchar2(100) not null,
          city varchar2(75));
```

This Oracle ALTER TABLE example will modify both the *customer_name* and *city* columns.

DROP COLUMN IN TABLE

Syntax

To DROP A COLUMN in an existing table, the Oracle ALTER TABLE syntax is:

```
ALTER TABLE table_name
  DROP COLUMN column_name;
```

Example

Let's look at an example that shows how to drop a column in an Oracle table using the ALTER TABLE statement.

For example:

```
ALTER TABLE customers
  DROP COLUMN customer_name;
```

This Oracle ALTER TABLE example will drop the column called *customer_name* from the table called *customers*.

RENAME COLUMN IN TABLE (NEW IN ORACLE 9I RELEASE 2)

Syntax

Starting in Oracle 9i Release 2, you can now rename a column.

To RENAME A COLUMN in an existing table, the Oracle ALTER TABLE syntax is:

```
ALTER TABLE table_name
  RENAME COLUMN old_name to new_name;
```

Example

Let's look at an example that shows how to rename a column in an Oracle table using the ALTER TABLE statement.

For example:

```
ALTER TABLE customers
  RENAME COLUMN customer_name to cname;
```

This Oracle ALTER TABLE example will rename the column called *customer_name* to *cname*.

RENAME TABLE

Syntax

To RENAME A TABLE, the Oracle ALTER TABLE syntax is:

```
ALTER TABLE table_name
  RENAME TO new_table_name;
```

Example

Let's look at an example that shows how to rename a table in Oracle using the ALTER TABLE statement.

For example:

```
ALTER TABLE customers
  RENAME TO contacts;
```

This Oracle ALTER TABLE example will rename the *customers* table to *contacts*.

PRACTICE EXERCISE #1:

Based on the *departments* table below, rename the *departments* table to *depts*.

```
CREATE TABLE departments
( department_id number(10) not null,
  department_name varchar2(50) not null,
  CONSTRAINT departments_pk PRIMARY KEY (department_id)
);
```

Solution for Practice Exercise #1:

The following Oracle ALTER TABLE statement would rename the *departments* table to *depts*:

```
ALTER TABLE departments
  RENAME TO depts;
```

PRACTICE EXERCISE #2:

Based on the *employees* table below, add a column called *bonus* that is a number(6) datatype.

```
CREATE TABLE employees
( employee_number number(10) not null,
  employee_name varchar2(50) not null,
  department_id number(10),
  CONSTRAINT employees_pk PRIMARY KEY (employee_number)
);
```

Solution for Practice Exercise #2:

The following Oracle ALTER TABLE statement would add a *bonus* column to the *employees* table:

```
ALTER TABLE employees
  ADD bonus number(6);
```

PRACTICE EXERCISE #3:

Based on the *customers* table below, add two columns - one column called *contact_name* that is a varchar2(50) datatype and one column called *last_contacted* that is a date datatype.

```
CREATE TABLE customers
( customer_id number(10) not null,
  customer_name varchar2(50) not null,
  address varchar2(50),
  city varchar2(50),
  state varchar2(25),
  zip_code varchar2(10),
  CONSTRAINT customers_pk PRIMARY KEY (customer_id)
);
```

Solution for Practice Exercise #3:

The following Oracle ALTER TABLE statement would add the *contact_name* and *last_contacted* columns to the *customers* table:

```
ALTER TABLE customers
  ADD (contact_name varchar2(50),
       last_contacted date);
```

PRACTICE EXERCISE #4:

Based on the *employees* table below, change the *employee_name* column to a varchar2(75) datatype.

```
CREATE TABLE employees
( employee_number number(10) not null,
  employee_name >varchar2(50) not null,
  department_id number(10),
  CONSTRAINT employees_pk PRIMARY KEY (employee_number)
);
```

Solution for Practice Exercise #4:

The following Oracle ALTER TABLE statement would change the datatype for the *employee_name* column to varchar2(75):

```
ALTER TABLE employees
  MODIFY employee_name varchar2(75);
```

PRACTICE EXERCISE #5:

Based on the *customers* table below, change the *customer_name* column to NOT allow null values and change the *state* column to a varchar2(2) datatype.

```
CREATE TABLE customers
( customer_id number(10) not null,
  customer_name varchar2(50),
  address varchar2(50),
  city varchar2(50),
```

```
  state varchar2(25),
  zip_code varchar2(10),
  CONSTRAINT customers_pk PRIMARY KEY (customer_id)
);
```

Solution for Practice Exercise #5:

The following Oracle ALTER TABLE statement would modify the *customer_name* and *state* columns accordingly in the *customers* table:

```
ALTER TABLE customers
  MODIFY (customer_name varchar2(50) not null,
          state varchar2(2));
```

PRACTICE EXERCISE #6:

Based on the *employees* table below, drop the *salary* column.

```
CREATE TABLE employees
( employee_number number(10) not null,
  employee_name varchar2(50) not null,
  department_id number(10),
  salary number(6),
  CONSTRAINT employees_pk PRIMARY KEY (employee_number)
);
```

Solution for Practice Exercise #6:

The following Oracle ALTER TABLE statement would drop the *salary* column from the *employees* table:

```
ALTER TABLE employees
  DROP COLUMN salary;
```

PRACTICE EXERCISE #7:

Based on the *departments* table below, rename the *department_name* column to *dept_name*.

```
CREATE TABLE departments
( department_id number(10) not null,
  department_name varchar2(50) not null,
  CONSTRAINT departments_pk PRIMARY KEY (department_id)
);
```

Solution for Practice Exercise #7:

The following Oracle ALTER TABLE statement would rename the *department_name* column to *dept_name* in the *departments* table:

```
ALTER TABLE departments
  RENAME COLUMN department_name to dept_name;
```