Created by Ahsan Arif

ALTER TABLE Statement

The ALTER TABLE statement allows you to rename an existing table. It can also be used to add, modify, or drop a column from an existing table.

Renaming a table

The basic syntax for renaming a table is:

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

For example:

```
ALTER TABLE suppliers
 RENAME TO vendors;
```

This will rename the *suppliers* table to *vendors*.

Adding column(s) to a table

Syntax #1

To add a column to an existing table, the ALTER TABLE syntax is:

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

For example:

```
ALTER TABLE supplier
 ADD supplier_name  varchar2(50);
```

This will add a column called *supplier_name* to the *supplier* table.

Syntax #2

To add multiple columns to an existing table, the ALTER TABLE syntax is:

```
ALTER TABLE table_name
ADD
(       column_1  column-definition,
```

```
    column_2  column-definition,
    ...
    column_n  column_definition );
```

For example:

```
ALTER TABLE supplier
ADD
(
    supplier_name  varchar2(50),
    city           varchar2(45) );
```

This will add two columns (*supplier_name* and *city*) to the *supplier* table.

Modifying column(s) in a table

Syntax #1

To modify a column in an existing table, the ALTER TABLE syntax is:

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

For example:

```
ALTER TABLE supplier
 MODIFY supplier_name   varchar2(100)     not null;
```

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

Syntax #2

To modify multiple columns in an existing table, the ALTER TABLE syntax is:

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

For example:

```
ALTER TABLE supplier
MODIFY
(
    supplier_name varchar2(100)  not null,
```

```
    city            varchar2(75)        );
```

This will modify both the *supplier_name* and *city* columns.

Drop column(s) in a table

Syntax #1

To drop a column in an existing table, the ALTER TABLE syntax is:

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

For example:

```
ALTER TABLE supplier
 DROP COLUMN supplier_name;
```

This will drop the column called *supplier_name* from the table called *supplier*.

Rename column(s) in a table (NEW in Oracle 9i Release 2)

Syntax #1

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

To rename a column in an existing table, the ALTER TABLE syntax is:

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

For example:

```
ALTER TABLE supplier
 RENAME COLUMN supplier_name to sname;
```

This will rename the column called *supplier_name* to *sname*.

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:

The following 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 *salary* 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:

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

```
ALTER TABLE employees
 ADD salary 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:

The following 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:

The following 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:

The following 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:

The following 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:

The following 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;
```