# Alias Syntax

In this lesson, we will learn about assigning aliases to tables and columns.

#### WE'LL COVER THE FOLLOWING ^

- Alias syntax
  - Syntax of a table alias
  - Example
  - Syntax for a column alias
  - EXAMPLE
  - Quick quiz!

## Alias syntax #

You can rename a table or a column temporarily by giving another name known as an **Alias**. The use of table aliases is to rename a table in a specific SQL statement. Column aliases are used to rename a table's columns for a particular SQL query. This renaming is a temporary change and the actual table/column name does not change in the database.

### Syntax of a table alias #

The basic syntax of a table alias is as follows:

```
SELECT column1, column2 ... columnN

FROM table_name AS alias_name

WHERE condition;
```

## Example #

For our example, we will be using the following tables:

#### **Customer Table**

#### **Orders Table**

ID	NA ME	AG E	AD DR ESS	SAL AR Y
1	Ma rk	32	Tex as	50, 000
2	Joh n	25	NY	65, 000
3	Emi ly	23	Ohi o	20, 000
4	Bill	25	Chi cag o	75, 000
5	To m	27	Wa shi ngt on	35, 000
6	Jan e	22	Tex as	45, 000

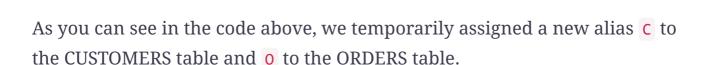
ORDE R_ID	DATE	CUST OME R_ID	AMO UNT
100	2019- 09-08	2	5000
101	2019- 08-20	5	3000
102	2019- 05-12	1	1000
103	2019- 02-02	2	2000

```
/* This is the same table we created in the previous lessons.*/
CREATE TABLE CUSTOMERS(

ID INT NOT NULL,
NAME VARCHAR (20) NOT NULL,
AGE INT NOT NULL,
ADDRESS CHAR (25),
SALARY DECIMAL (18, 2), /* The (18,2) simply means that we can have 18 digits with 2 of PRIMARY KEY (ID)
);

INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (1, 'Mark', 32, 'Texas', 50000.00 );
```

```
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (2, 'John', 25, 'NY', 65000.00 );
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (3, 'Emily', 23, 'Ohio', 20000.00 );
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (4, 'Bill', 25, 'Chicago', 75000.00 );
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (5, 'Tom', 27, 'Washington', 35000.00);
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (6, 'Jane', 22, 'Texas', 45000.00 );
/*We will now create the ORDERS table*/
CREATE TABLE ORDERS(
 ORDER_ID INT
                       NOT NULL,
 DATE VARCHAR (20)
                      NOT NULL,
 CUSTOMER ID INT
                      NOT NULL,
 AMOUNT INT,
 PRIMARY KEY (ORDER_ID),
  FOREIGN KEY (CUSTOMER_ID) REFERENCES CUSTOMERS(ID) /* We must specify the table to which the
);
INSERT INTO ORDERS (ORDER_ID, DATE, CUSTOMER_ID, AMOUNT)
VALUES (100, '2019-09-08', 2, 5000);
INSERT INTO ORDERS (ORDER ID, DATE, CUSTOMER ID, AMOUNT)
VALUES (101, '2019-08-20', 5, 3000);
INSERT INTO ORDERS (ORDER_ID, DATE, CUSTOMER_ID, AMOUNT)
VALUES (102, '2019-05-12', 1, 1000);
INSERT INTO ORDERS (ORDER_ID, DATE, CUSTOMER_ID, AMOUNT)
VALUES (103, '2019-02-02', 2, 2000 );
SELECT C.ID, C.NAME, C.AGE, O.AMOUNT
FROM CUSTOMERS AS C, ORDERS AS O
WHERE C.ID = O.CUSTOMER_ID;
```



Furthermore, when working with two tables, we need to specify the table name from which the column is derived, therefore having short aliases helps us to avoid writing large names before each column name. Also, sometimes tables can have the same column names so specifying the table name before the column name helps to avoid confusion regarding which table we are referring to.

have selected based on the condition that ID in the CUSTOMERS table is equal to ORDER\_ID in ORDERS table. So the result-set is the group of people who have placed orders.

### Syntax for a column alias #

The basic syntax of a column alias is as follows:

```
SELECT column_name AS alias_name
FROM table_name
WHERE condition;
```

### **EXAMPLE**

Following is the usage of a column alias.

```
/* This is the same table we created in the previous lessons.*/
                                                                                      (C)
CREATE TABLE CUSTOMERS(
  ID
     INT
                       NOT NULL,
 NAME VARCHAR (20) NOT NULL,
 AGE INT
                       NOT NULL,
 ADDRESS CHAR (25),
 SALARY DECIMAL (18, 2), /* The (18,2) simply means that we can have 18 digits with 2 of
 PRIMARY KEY (ID)
);
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (1, 'Mark', 32, 'Texas', 50000.00);
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (2, 'John', 25, 'NY', 65000.00);
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (3, 'Emily', 23, 'Ohio', 20000.00 );
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (4, 'Bill', 25, 'Chicago', 75000.00);
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (5, 'Tom', 27, 'Washington', 35000.00);
INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)
VALUES (6, 'Jane', 22, 'Texas', 45000.00 );
/*We will now create the ORDERS table*/
CREATE TABLE ORDERS(
 ORDER_ID INT
                       NOT NULL,
 DATE VARCHAR (20) NOT NULL,
                      NOT NULL,
 CUSTOMER_ID INT
  AMOUNT INT,
```

```
PRIMARY KEY (ORDER_ID),
FOREIGN KEY (CUSTOMER_ID) REFERENCES CUSTOMERS(ID) /* We must specify the table to which the specify that the specify the table to which the specify in the specify that the specify the table to which the specify in the specify that table to which the specify in the specify that table to which the specify in the specify that table to which the specify in the specific in the
```





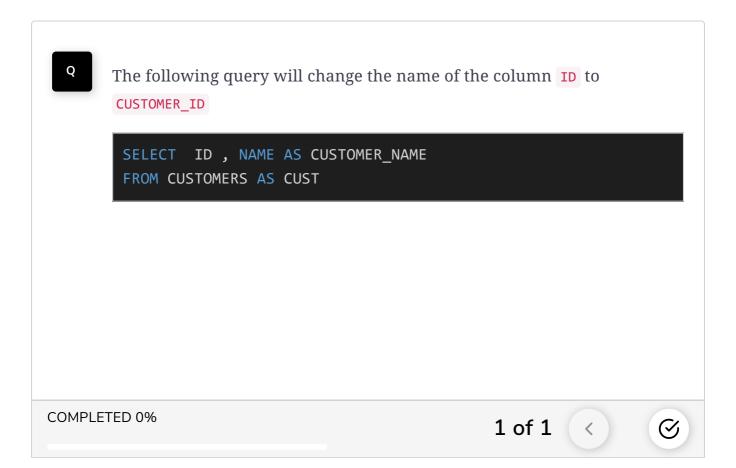


[]

As you can see in the output above, the column names have changed.

A column alias is particularly useful when we want to change the name of a column to one that is easier to understand for the user.

## Quick quiz! #



In the next lesson, we will learn to combine two different tables using joins.