**FOREIGN KEY**

A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table.

The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table.

CREATE TABLE customers(

id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(100),

last\_name VARCHAR(100),

email VARCHAR(100)

);

CREATE TABLE orders(

id INT AUTO\_INCREMENT PRIMARY KEY,

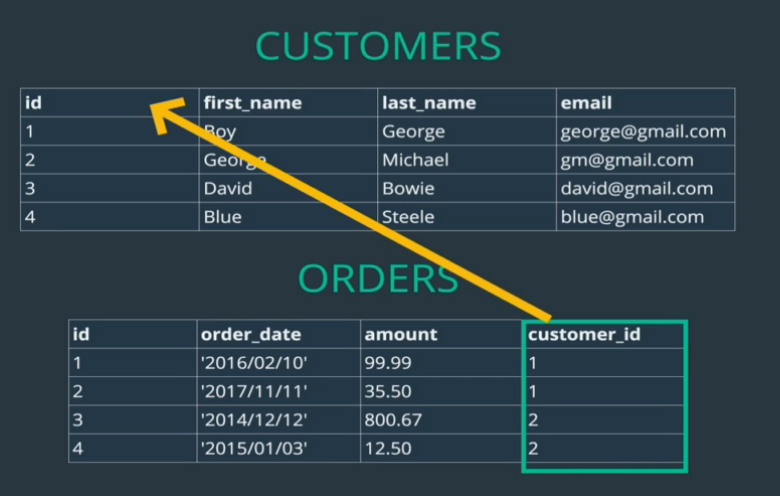
order\_date DATE,

amount DECIMAL(8,2),

customer\_id INT,

FOREIGN KEY(customer\_id) REFERENCES customers(id)

);



**JOINS**

**Finding Orders Placed By George: 2 Step Process**

1. SELECT id FROM customers WHERE last\_name='George';
2. SELECT \* FROM orders WHERE customer\_id = 1;

**Finding Orders Placed By George: Using a subquery**

SELECT \* FROM orders WHERE customer\_id =

(

SELECT id FROM customers

WHERE last\_name='George'

);

**Cross Join Craziness**

SELECT \* FROM customers, orders;

the CROSS JOIN is **used to combine each row of the first table with each row of the second table**. It is also known as the Cartesian join since it returns the Cartesian product of the sets of rows from the joined tables.

**Explicit Inner JOIN**



**The INNER JOIN keyword selects records that have matching values in both tables.**

mysql> SELECT first\_name, last\_name, order\_date, amount

-> FROM customers

-> JOIN orders

-> ON customers.id = orders.customer\_id;

+------------+-----------+------------+--------+

| first\_name | last\_name | order\_date | amount |

+------------+-----------+------------+--------+

| Boy | George | 2016-02-10 | 99.99 |

| Boy | George | 2017-11-11 | 35.50 |

| George | Michael | 2014-12-12 | 800.67 |

| George | Michael | 2015-01-03 | 12.50 |

| Bette | Davis | 1999-04-11 | 450.25 |

+------------+-----------+------------+--------+

5 rows in set (0.00 sec)

**LEFT JOIN**

The LEFT JOIN keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side, if there is no match.



mysql> **SELECT \* FROM CUSTOMERS LEFT JOIN ORDERS ON customers.id = orders.customer\_id;**

+----+------------+-----------+------------------+------+------------+--------+-------------+

| id | first\_name | last\_name | email | id | order\_date | amount | customer\_id |

+----+------------+-----------+------------------+------+------------+--------+-------------+

| 1 | Boy | George | george@gmail.com | 1 | 2016-02-10 | 99.99 | 1 |

| 1 | Boy | George | george@gmail.com | 2 | 2017-11-11 | 35.50 | 1 |

| 2 | George | Michael | gm@gmail.com | 3 | 2014-12-12 | 800.67 | 2 |

| 2 | George | Michael | gm@gmail.com | 4 | 2015-01-03 | 12.50 | 2 |

| 3 | David | Bowie | david@gmail.com | NULL | NULL | NULL | NULL |

| 4 | Blue | Steele | blue@gmail.com | NULL | NULL | NULL | NULL |

| 5 | Bette | Davis | bette@aol.com | 5 | 1999-04-11 | 450.25 | 5 |

+----+------------+-----------+------------------+------+------------+--------+-------------+

**RIGHT JOIN**

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match.



mysql> **SELECT \* FROM CUSTOMERS RIGHT JOIN ORDERS ON customers.id = orders.customer\_id;**

+------+------------+-----------+------------------+----+------------+--------+-------------+

| id | first\_name | last\_name | email | id | order\_date | amount | customer\_id |

+------+------------+-----------+------------------+----+------------+--------+-------------+

| 1 | Boy | George | george@gmail.com | 1 | 2016-02-10 | 99.99 | 1 |

| 1 | Boy | George | george@gmail.com | 2 | 2017-11-11 | 35.50 | 1 |

| 2 | George | Michael | gm@gmail.com | 3 | 2014-12-12 | 800.67 | 2 |

| 2 | George | Michael | gm@gmail.com | 4 | 2015-01-03 | 12.50 | 2 |

| 5 | Bette | Davis | bette@aol.com | 5 | 1999-04-11 | 450.25 | 5 |

+------+------------+-----------+------------------+----+------------+--------+-------------+

5 rows in set (0.00 sec)

**FULL JOIN**

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

**Tip:** FULL OUTER JOIN and FULL JOIN are the same.



The SQL UNION Operator

The UNION operator is used to combine the result-set of two or more SELECT statements.

* Every SELECT statement within UNION must have the same number of columns
* The columns must also have similar data types
* The columns in every SELECT statement must also be in the same order

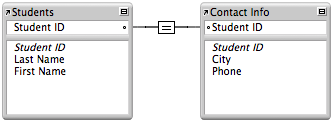
UNION Syntax

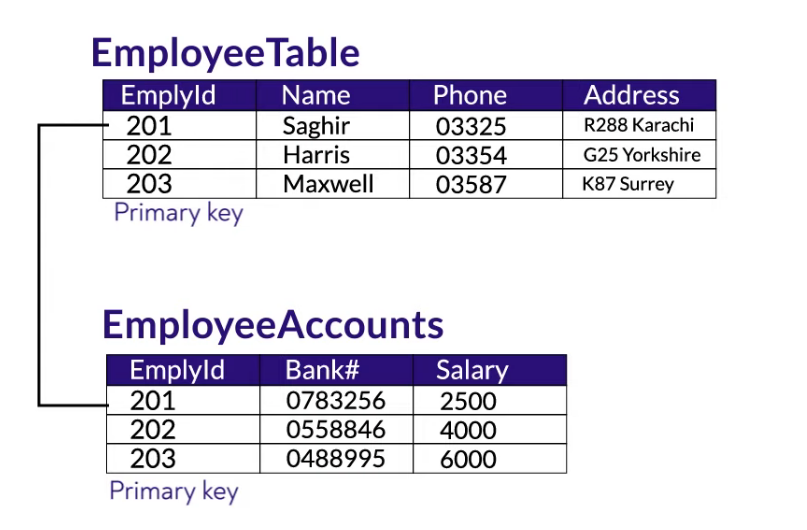
SELECT *column\_name(s)* FROM *table1*  
UNION  
SELECT *column\_name(s)* FROM *table2*;

MAPPING

**ONE TO ONE**

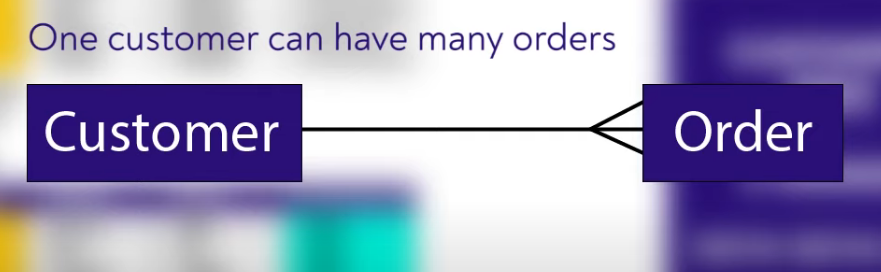
 A **one-to-one table relationship links two tables based on a Primary Key column in the child which is also a Foreign Key referencing the Primary Key of the parent table row**. Therefore, we can say that the child table shares the Primary Key with the parent table.

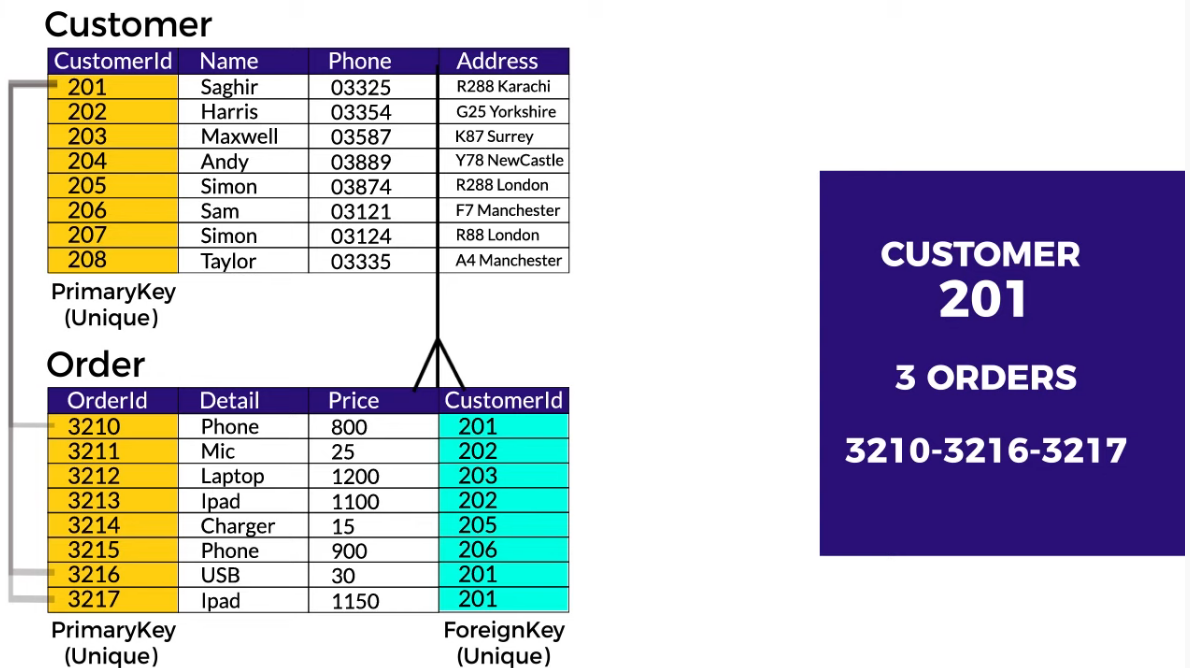




**ONE TO MANY**

A one-to-many relationship occurs when one record in table 1 is related to one or more records in table 2. However, one record in table 2 cannot be related to more than one record in table 1.





**MANY TO MANY**

