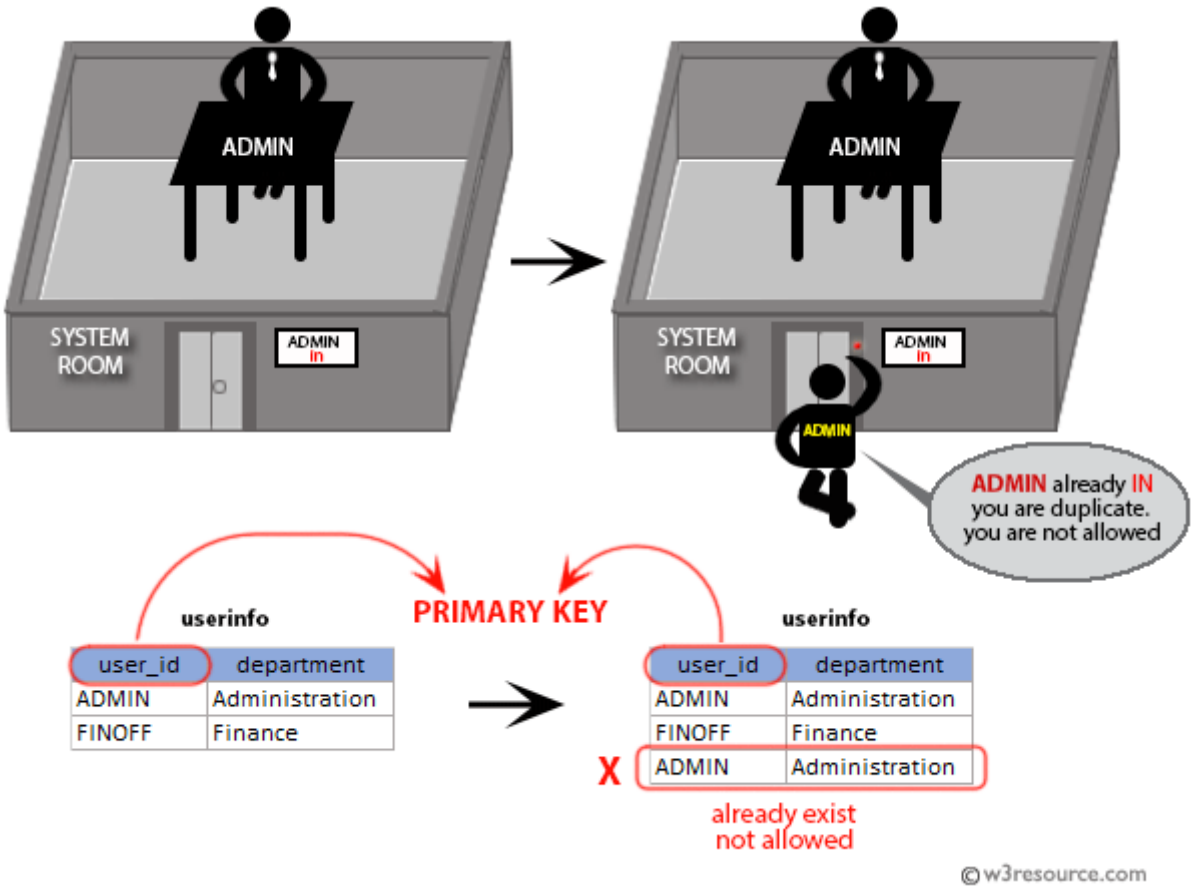
PRIMARY KEY

The SQL PRIMARY KEY is a column in a table which must contain a unique value which can be used to identify each and every row of a table uniquely.

However, SQL supports primary keys directly with the PRIMARY KEY constraint.



Functionally, it is the same as the UNIQUE constraint, except that only one PRIMARY KEY can be defined for a given table. PRIMARY KEY's will not allow NULL values.

A primary key is used to identify each row identically in a table. It may be a part of the actual record itself.

The SQL PRIMARY KEY can be made up by one or more fields on a table and when it happens, they are called a composite key.

Primary keys can be specified at the time of CREATING TABLE or the time of changing the structure of the existing table using ALTER TABLE statement.

This constraint is a combination of a NOT NULL constraint and a UNIQUE constraint. This constraint ensures that the specific column or combination of two or more columns for a table have a unique identity which helps to find a particular record in a table more easily and quickly.

Syntax:

```
CREATE TABLE <table_name>
column1    data_type[(size)]  NOT NULL  PRIMARY KEY,
column2    data_type[(size)],
...);
```

Parameters:

Name	Description
table_name	Name of the table where data is stored.
column1,column2	Name of the columns of a table.
data_type	Is char, varchar, integer, decimal, date and more.
size	Maximum length of the column of a table.

Good practice for primary keys in tables

- Primary keys should be as small as necessary. Prefer a numeric type because numeric types are stored in a much more compact format than character formats.
- Primary keys should never change.
- Do not use passport number, social security number, or employee contract number as "primary key" as these "primary key" can change for real world situations.

Example:

Suppose, we are going to create a table named 'agent1'. It contains the columns and data types which shown bellow. For each row of 'agent1' table, it is required to identify each agent with a unique code, because the name of two or more agents of a city of a country may be same.

So, it is not a good choice to create PRIMARY KEY on 'agent_name'. The 'agent_code' could be the only and exclusive choice for a PRIMARY KEY for this table.

Field Name	Data Type	Size	Decimal Places	NULL	Constraint
agent_code	char	6		No	PRIMARY KEY
agent_name	char	40		No	
working_area	char	35		Yes	
commission	decimal	10	2	Yes	
phone_no	char	17		Yes	

While creating a table you can include a primary key using column-level primary-key constraint or table-level constraint. Here are two examples on the said table :

Column-level primary-key constraint :

SQL Code:

```
CREATE TABLE agent1(
agent_code char(6) NOT NULL PRIMARY KEY,
agent_name char(40) NOT NULL,
working_area char(35),
commission decimal(10,2),
phone_no char(17)
) ;
```

Copy

Table-level primary-key constraint:

SQL Code:

```
CREATE TABLE agent1(
agent_code char(6) NOT NULL,
agent_name char(40) NOT NULL,
working_area char(35),
commission decimal(10,2),
phone_no char(17),
CONSTRAINT pk_agent_code PRIMARY KEY (agent_code)
) ;
```

SQL CREATE TABLE with PRIMARY KEY CONSTRAINT

SQL PRIMARY KEY CONSTRAINT is a combination of a NOT NULL constraint and a UNIQUE constraint. This constraint ensures that the specific column or combination of two or more columns for a table have an unique identity which helps to find a particular record in a table more easily and quickly.

Example :

The following example creates a table. Here is the field name and data types :

Field Name	Data Type	Size	Decimal Places	NULL	Constraint
agent_code	char	6		No	PRIMARY KEY
agent_name	char	40		No	
working_area	char	35		Yes	
commission	decimal	10	2	Yes	

phone_no	char	17		Yes	

the following SQL statement can be used :

SQL Code:

```
CREATE TABLE mytest(
agent_code char(6) NOT NULL PRIMARY KEY,
agent_name char(40) NOT NULL,
working_area char(35),
commission decimal(10,2),
phone_no char(17));
```
Copy

To see the structure of the created table :

SQL Code:

```
DESCRIBE mytest;
```
Copy

Output :

Object Type **TABLE** Object **MYTEST**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MYTEST	AGENT_CODE	Char	6	-	-	1	-	-	-
	AGENT_NAME	Char	40	-	-	-	-	-	-
	WORKING_AREA	Char	35	-	-	-	✓	-	-
	COMMISSION	Number	-	10	2	-	✓	-	-
	PHONE_NO	Char	17	-	-	-	✓	-	-
									1 - 5

SQL CREATE TABLE with PRIMARY KEY and UNIQUE CONSTRAINT

The following example creates a table. Here is the field name and data types:

Field Name	Data Type	Size	Decimal Places	NULL	Constraint
cust_code	char	6		No	PRIMARY KEY
cust_name	char	25		No	UNIQUE
cust_city	char	25		No	
grade	integer			Yes	

agent_code	char	6		No	

the following SQL statement can be used:

SQL Code:

```
CREATE TABLE mytest(
cust_code char(6) NOT NULL PRIMARY KEY,
cust_name      char(25) NOT NULL UNIQUE,
cust_city char(25) NOT NULL,
grade   integer,
agent_code char(6) NOT NULL);
```

Copy

To see the structure of the created table :

SQL Code:

```
DESCRIBE mytest;
```

Output:

Object Type **TABLE** Object **MYTEST**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MYTEST	CUST_CODE	Char	6	-	-	1	-	-	-
	CUST_NAME	Char	25	-	-	-	-	-	-
	CUST_CITY	Char	25	-	-	-	-	-	-
	GRADE	Number	-	-	0	-	✓	-	-
	AGENT_CODE	Char	6	-	-	-	-	-	-
									1 - 5

SQL CREATE TABLE with PRIMARY KEY CONSTRAINT on more columns

In the following topic, we are going to discuss the usage of SQL PRIMARY KEY CONSTRAINT along with the CREATE TABLE statement for two or more columns.

Example:

The following example creates a table. The table must contain a PRIMARY KEY with the combination of two columns 'cust_code' and 'cust_city'. Here is the field name and data types :

Field Name	Data Type	Size	Decimal Places	NULL	Constraint
cust_code	char	6		No	PRIMARY KEY
cust_name	char	25		No	
cust_city	char	25		No	PRIMARY KEY

Grade	integer			Yes	
agent_code	char	6		No	

the following SQL statement can be used:

SQL Code:

```
CREATE TABLE mytest(
cust_code char(6) NOT NULL,
cust_name char(25) NOT  NULL,
cust_city char(25),
grade integer,
agent_code char(6) NOT NULL,
PRIMARY KEY (cust_code,cust_city));
```

To see the structure of the created table:

SQL Code:

```
DESCRIBE mytest;
```

Output:

Object Type **TABLE** Object **MYTEST**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MYTEST	CUST_CODE	Char	6	-	-	1	-	-	-
	CUST_NAME	Char	25	-	-	-	-	-	-
	CUST_CITY	Char	25	-	-	2	-	-	-
	GRADE	Number	-	-	0	-	✓	-	-
	AGENT_CODE	Char	6	-	-	-	-	-	-
									1 - 5