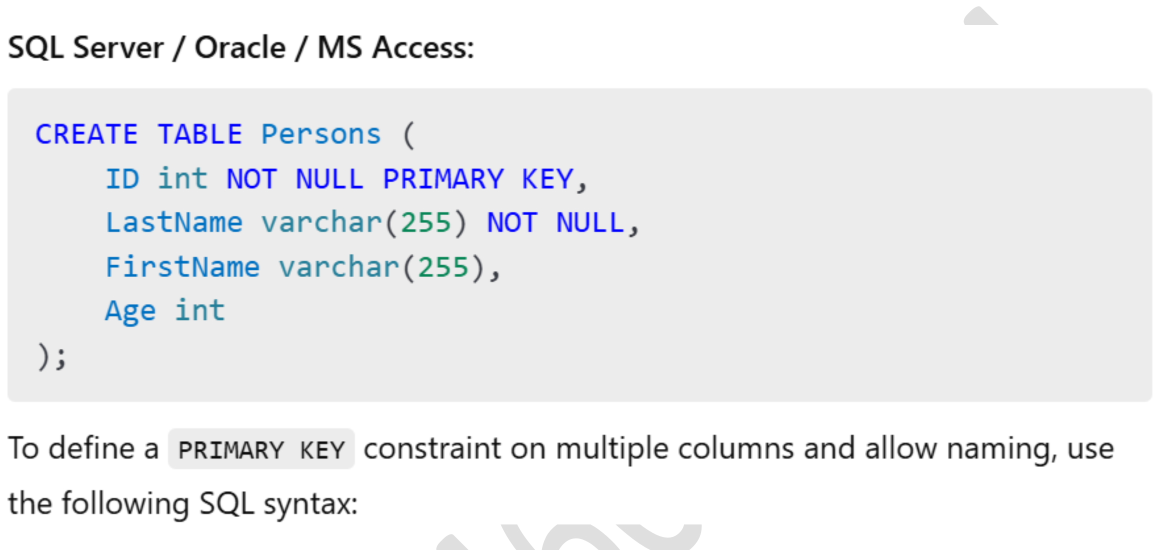
SQL PRIMARY KEY Constraint

The `PRIMARY KEY` constraint uniquely identifies each record in a table. Primary keys must contain unique values and cannot contain `NULL` values. A table can have only one primary key, which may consist of a single column or multiple columns.

SQL PRIMARY KEY on CREATE TABLE

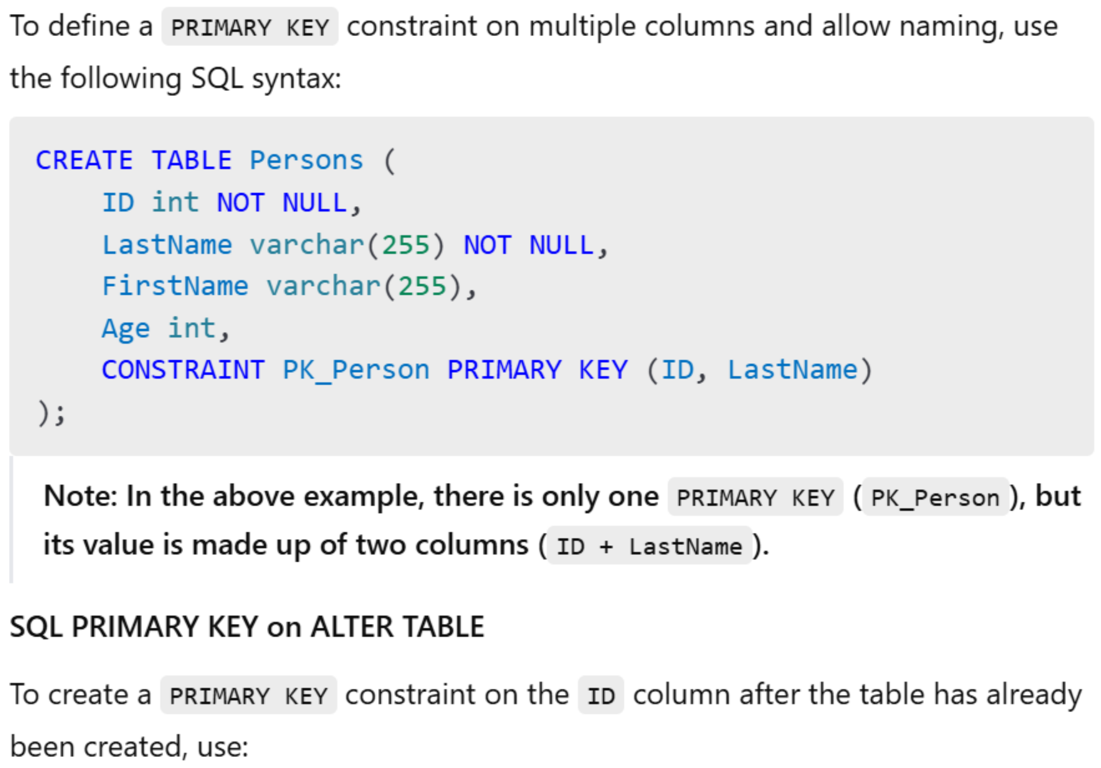
The following SQL creates a `PRIMARY KEY` on the `ID` column when the `Persons` table is created



SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons (
    ID int NOT NULL PRIMARY KEY,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int
);
```

To define a `PRIMARY KEY` constraint on multiple columns and allow naming, use the following SQL syntax:



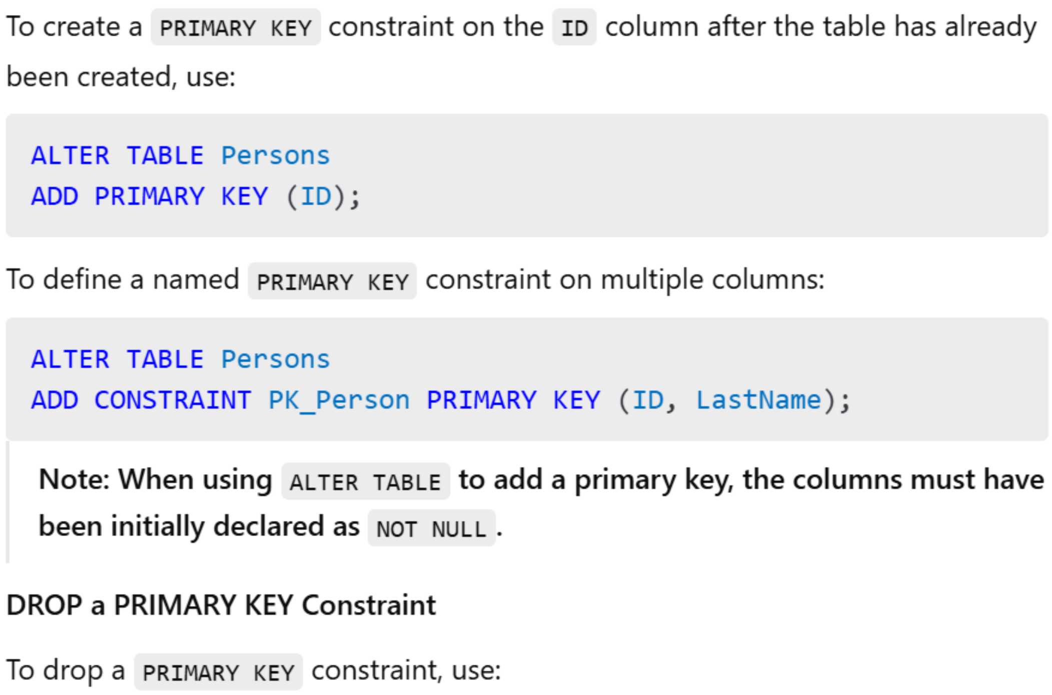
To define a `PRIMARY KEY` constraint on multiple columns and allow naming, use the following SQL syntax:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CONSTRAINT PK_Person PRIMARY KEY (ID, LastName)
);
```

> **Note: In the above example, there is only one `PRIMARY KEY` (`PK_Person`), but its value is made up of two columns (`ID + LastName`).**

SQL PRIMARY KEY on ALTER TABLE

To create a `PRIMARY KEY` constraint on the `ID` column after the table has already been created, use:



To create a `PRIMARY KEY` constraint on the `ID` column after the table has already been created, use:

```
ALTER TABLE Persons
ADD PRIMARY KEY (ID);
```

To define a named `PRIMARY KEY` constraint on multiple columns:

```
ALTER TABLE Persons
ADD CONSTRAINT PK_Person PRIMARY KEY (ID, LastName);
```

> **Note: When using `ALTER TABLE` to add a primary key, the columns must have been initially declared as `NOT NULL`.**

DROP a PRIMARY KEY Constraint

To drop a `PRIMARY KEY` constraint, use: