... continue

This lesson continues the discussion on creating tables and how to add restrictions when defining them.

> In the previous lesson we worked with a very simple create table statement. In this lesson we'll gradually apply more restrictions on what our table can accept.

> Connect to the terminal below by clicking in the widget. Once connected, the command line prompt will show up. Enter or copy and paste the command **./DataJek/Lessons/4lesson.sh** and wait for the mysql prompt to start-up.

```
-- The lesson queries are reproduced below for convenient copy/paste into the terminal.

-- Query 1
CREATE TABLE Actors (
Id INT AUTO_INCREMENT,
FirstName VARCHAR(20) NOT NULL,
SecondName VARCHAR(20) NOT NULL,
DoB DATE NOT NULL,
Gender ENUM('Male','Female','Transgender') NOT NULL,
MaritalStatus ENUM('Married', 'Divorced', 'Single', 'Unknown') DEFAULT "Unknown",
NetWorthInMillions DECIMAL NOT NULL,
PRIMARY KEY (Id));

-- Query 2
CREATE TABLE IF NOT EXISTS Actors (
Id INT AUTO_INCREMENT,
FirstName VARCHAR(20) NOT NULL,
SecondName VARCHAR(20) NOT NULL,
DoB DATE NOT NULL,
Gender ENUM('Male','Female','Transgender') NOT NULL,
MaritalStatus ENUM('Married', 'Divorced', 'Single', 'Unknown') DEFAULT "Unknown",
NetWorthInMillions DECIMAL NOT NULL,
PRIMARY KEY (Id));
```

Terminal

1. We'll want to restrict a user from adding duplicate rows. In a relational database we can uniquely identify a row by designating a single column or a set of columns as the primary key. We discuss the concept of a primary key in depth in a later lesson but for now, it suffices to know that a column or multiple columns acting as the primary key serve to uniquely identify each row in a table. In the case of our **Actors** table, we may be tempted to choose the combination of the first name and the second name as the primary key. However, there is a remote possibility that two actors have the same first and last names. In fact, even if we choose all the columns to form the primary key, an obviously bad design choice, we'll not be assured each row can be uniquely identified. One way out of this predicament is to add a column that assigns a numeric ID starting from one that is incremented each time we add a new row. Since every row will be associated with a different integer, we can be assured that every row can be uniquely identified using the ID column.

2. If we create a new column ID and designate it as the primary key for the **Actors** table, then we must know the highest value of the ID column in the table, add one to it and insert the result as the Id of a new row. MySQL provides a feature **AUTO_INCREMENT** that automatically assigns the next integer in the sequence to the ID column of a new row.

   The auto increment sequence begins at 1 for an empty table. The following restrictions exist when using **AUTO_INCREMENT** feature:

   - There can be only one column marked as **AUTO_INCREMENT** in a table.
   - The **AUTO_INCREMENT** column can’t have a default value.
   - The **AUTO_INCREMENT** column must be indexed.

   The **AUTO_INCREMENT** feature isn’t portable to other databases and the counter is reset when we truncate or drop a table.

3. The next restriction we want to put on the users of **Actors** table is to disallow inserting null as a column value. We can mark a column as **NOT NULL** to achieve this purpose.

4. Imagine a situation for the **Actors** table where you want to add a record to the table but aren't aware of the marital status of the Actor. In such a scenario, we can set a default value for a particular column using the **DEFAULT** keyword. The default value specified must be a constant. The default value will be used whenever a user omits a value for a column that has a default specified. We'll recreate the table **Actors** and add another enum value "Unknown" to the list. We'll also mark other columns as **NOT NULL** so that users are forced to provide valid values for other columns.

> We can also use **NOT NULL** and **DEFAULT** together.

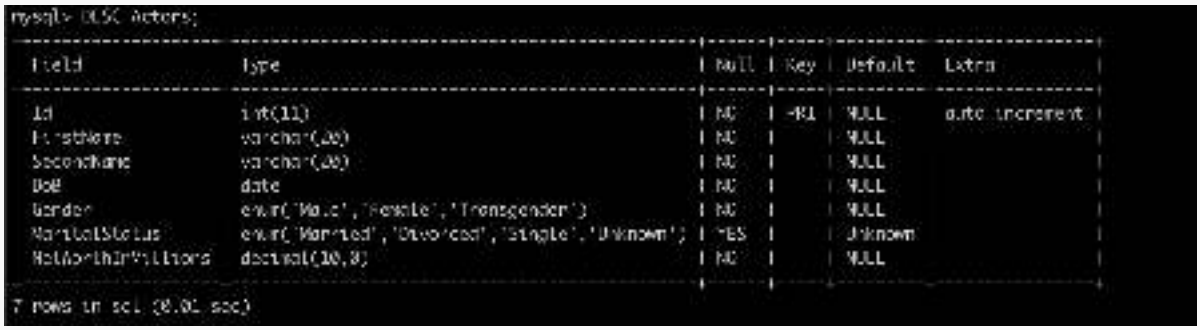
5. The following MYSQL statement adds all the restrictions we just talked about:

```
CREATE TABLE Actors (
Id INT AUTO_INCREMENT,
FirstName VARCHAR(20) NOT NULL,
SecondName VARCHAR(20) NOT NULL,
DoB DATE NOT NULL,
Gender ENUM('Male','Female','Transgender') NOT NULL,
MaritalStatus ENUM('Married', 'Divorced', 'Single', 'Unknown') DE
FAULT "Unknown",
NetWorthInMillions DECIMAL NOT NULL,
PRIMARY KEY (Id));
```

6. Contrast the output of the **DESC** command from the previous lesson below:

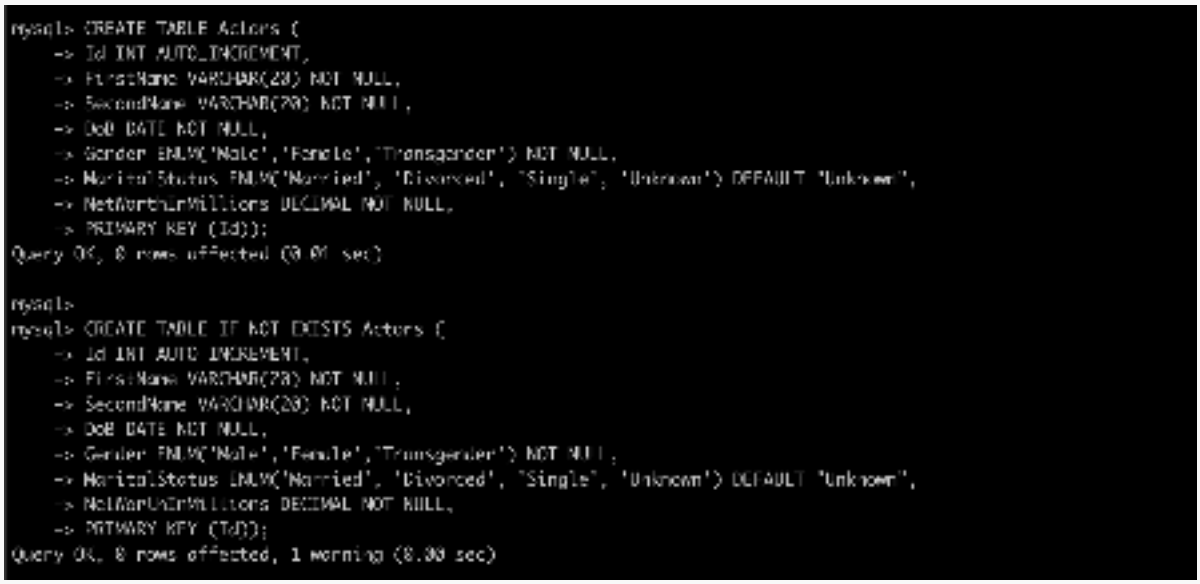
```
mysql> DESC Actors;
+--------------------+-----------------------------------------+------+-----+---------+-------+
| Field              | Type                                    | Null | Key | Default | Extra |
+--------------------+-----------------------------------------+------+-----+---------+-------+
| FirstName          | varchar(20)                             | YES  |     | NULL    |       |
| SecondName         | varchar(20)                             | YES  |     | NULL    |       |
| DoB                | date                                    | YES  |     | NULL    |       |
| Gender             | enum('Male','Female','Transgender')     | YES  |     | NULL    |       |
| MaritalStatus      | enum('Married','Divorced','Single')     | YES  |     | NULL    |       |
| NetWorthInMillions | decimal(10,0)                           | YES  |     | NULL    |       |
+--------------------+-----------------------------------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

vs



7. We can also use the **IF EXISTS** clause when creating a table, similar to the way we did when creating a database. We can execute the following query without any side-effects as the **Actors** table already exists.

```
CREATE TABLE IF NOT EXISTS Actors (
Id INT AUTO_INCREMENT,
FirstName VARCHAR(20) NOT NULL,
SecondName VARCHAR(20) NOT NULL,
DoB DATE NOT NULL,
Gender ENUM('Male','Female','Transgender') NOT NULL,
MaritalStatus ENUM('Married', 'Divorced', 'Single', 'Unknown') DE
FAULT "Unknown",
NetWorthInMillions DECIMAL NOT NULL,
PRIMARY KEY (Id));
```


mysql> CREATE TABLE Actors (
-> Id INT AUTO_INCREMENT,
-> FirstName VARCHAR(20) NOT NULL,
-> SecondName VARCHAR(20) NOT NULL,
-> DoB DATE NOT NULL,
-> Gender ENUM('Male','Female','Transgender') NOT NULL,
-> MaritalStatus ENUM('Married', 'Divorced', 'Single', 'Unknown') DEFAULT 'Unknown',
-> NetWorthInMillions DECIMAL NOT NULL,
-> PRIMARY KEY (Id));
Query OK, 0 rows affected (0.01 sec)
mysql>
mysql> CREATE TABLE IF NOT EXISTS Actors (
-> Id INT AUTO_INCREMENT,
-> FirstName VARCHAR(20) NOT NULL,
-> SecondName VARCHAR(20) NOT NULL,
-> DoB DATE NOT NULL,
-> Gender ENUM('Male','Female','Transgender') NOT NULL,
-> MaritalStatus ENUM('Married', 'Divorced', 'Single', 'Unknown') DEFAULT 'Unknown',
-> NetWorthInMillions DECIMAL NOT NULL,
-> PRIMARY KEY (Id));
Query OK, 0 rows affected, 1 warning (0.00 sec)