Exercise 4 – Creating a new table Code breakdown

----------------------------------------------------

Table `autoclub`.`user`

-----------------------------------------------------

* The first three lines are comments to show what the code is doing

CREATE TABLE IF NOT EXISTS `autoclub`.`user`

* **CREATE TABLE** is telling the server we want to create a table, **IF NOT EXISTS** is telling the server to only create the table if it does not already exist. **`autoclub`.`user`** is the name of the table to create and what database to put it in. If the database name is left off, the table will be created in the current default database, including the database name ensures it is created in the intended database. Both the database name and the table name are surrounded by **back ticks**.

(

* Next comes the field names and their definition. The field names are enclosed in brackets, they have been placed on a separate line to make this breakdown more readable.

`ID` INT NOT NULL AUTO\_INCREMENT,

* This is the ID field
  + All field names are enclosed in back ticks, they appear first on the line followed by the datatype.
  + Next is either **NULL** or **NOT NULL** to define if the field can take a null or not
  + Then other options if applicable.
  + All field definitions are separated by a comma
  + The auto\_increment command will ensure that a value is always entered into this field by the system.

The ID line, in plain English is, the fields name is **ID**, it can hold an **INT**eger value only, it can**NOT** contain a **NULL** and the value in it will **AUTO**matically\_**INCREMENT**ed for each record because it is also designated as the Primary Key in the last statement of the block.

`username` VARCHAR(16) NOT NULL,

* + This will hold a username of up to 16 characters in length, the field is set to **NOT NULL** to ensure a username is entered, it is a required field. There are no further options so the line is terminated with a comma. If an attempt is made to create or save a record without data in any NOT NULL fields, be it a new record or the value is removed from an existing record, the server will not create or save the record.

`email` VARCHAR(255) NULL,

* Emails can be long so up to 255 characters are permitted, not everyone has an email, really, so the field is permitted to take a **NULL**

`password` VARCHAR(32) NOT NULL,

* The password is a required field, it can be up to 32 characters long and is set to **NOT NULL** to force an entry

`Active` BIT NOT NULL DEFAULT 1,

* This field is to indicate if the user account is active or not and will be used by any application developed for the database, the **BIT** type can be used as a simple True/False, the field should never contain a NULL and a default value of 1 (True) is set for any new records.

`WhenAdded` TIMESTAMP NULL DEFAULT CURRENT\_TIMESTAMP,

* Indicates when the records was added, the datatype is a **TIMESTAMP** and it can be a **NULL** however the default value of **CURRENT\_TIMESTAMP** will ensure a value, the current date/time will be put in the field. This field cannot and should not be modified after the record is created.

`LastModified` TIMESTAMP NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

* Similar to the ‘WhenAdded’ field however the default of **CURRENT\_TIMESTAMP ON UPDATE** will allow the field to be updated with the current date/time by the system whenever the records data is modified

PRIMARY KEY (`ID`)

* This command indicates which field will be set to be the **PRIMARY KEY** for the table. The Primary Key is usually an INT datatype and is usually set to Auto Increment as was done for the ID field’s definition earlier. This is the last command in the field definition so it does NOT end with a comma, the last command in the block, whether it be a field definition or other command does not have a trailing comma.

);

* When all of the fields are defined, the **closing bracket** is added. The **semicolon** is also added here to indicate this is the end of the SQL statement block. The semicolon tells the server that everything before it (and after the semicolon of any preceding statement blocks) is to be treated as one statement. This is a single statement block script, the semicolon could have been left off but if you were creating several tables or running other statements in the same script, you need to separate them with the semicolon, leaving the semicolon out may cause issues with multiple statement block scripts.