# CSC1023 Databases

NOTICE to Student Route / Tier 4 Students only:

For today, 7th March 2022, you are required to participate in the Student Route / Tier 4 Student engagement monitoring process. A QR code will be posted around the laboratory (at the entrance door and with your Demonstrator) and you must scan the QR while you are at the laboratory.

Practical 6 - SQL CREATE, Saving

Schema and Referential Integrity

# Section A. Create and Update tables

## Perform the following tasks for all the questions listed in this section:

* **write** an **SQL query**,
* **execute** the query,
* **Check** your output **table** gives the expected result,
* **paste** the SQL **query** statement into this document.

Here is a data dictionary for a table:

**Table:** doggo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Nullable** | **Other** |
| id | INT | Y | N | PRIMARY KEY, AUTO INCREMENT |
| Name | VARCHAR(255) | N | N |  |
| Breed | VARCHAR(255) | N | N |  |
| Birth | DATE | N | N |  |
| Floof | BOOLEAN | N | Y |  |
| Hooman | VARCHAR(32) | N | Y |  |

***A1.*** Write and execute SQL statements to create the above table.  
CREATE TABLE doggo (id int not null PRIMARY KEY AUTO\_INCREMENT, Name varchar(255) not null, Breed varchar(255) not null, Birth date not null, Floof boolean, Hooman varchar(32), INDEX(id))

***A2.*** It turns out Hooman can have names longer than 32 characters - update the Hooman field to be a VARCHAR of length 255.  
ALTER TABLE doggo

MODIFY COLUMN Hooman varchar(255)

***A3.*** Modify the table to give the 'Floof' field a default value of 1 (true), as all doggo records are stored as floof until proven otherwise.  
UPDATE doggo

SET Floof = '1'

***A4.*** In the table, the attribute 'Birth' has the datatype of DATE. Modify the table so that

'Birth' has the datatype of DATETIME.  
ALTER TABLE doggo

MODIFY COLUMN Birth datetime

***A5.*** Add a "Notes" field of type TEXT which can be NULL.  
 ALTER TABLE doggo

ADD COLUMN Notes text

***A6.*** Add a BOOLEAN field 'Hungry' with a default of 1 (true).

ALTER TABLE doggo

ADD COLUMN Hungry boolean

DEFAULT 1

***A7.*** We want to be able to quickly search on Name, add this field as an index  
 ALTER TABLE doggo

ADD index(Name)

***A8.*** We don't always know what Breed a doggo is, so update this field so it can be NULL.  
 UPDATE doggo

set Breed = null

***A9.*** Extensive research has shown that doggos will always claim they are hungry - get rid of the Hungry field from the table.  
ALTER TABLE doggo

DROP COLUMN Hungry

# Section B. Saving Schema, Deleting Tables, Restoring Schema

The **doggo** table is now created in your MySQL account. Next, we will learn to export relational tables, so we can install it on another database or to store it as a backup, or just to keep a copy.

phpMyAdmin can export the entire database or an individual table - with or without data. We only want to export the doggo table and without data (just the schema).

## Perform the following tasks:

***B1.*** Under your database tree (the left-hand pane of your phpMyAdmin screen), select your doggo table.



***B2.*** On the right-hand pane of phpMyAdmin, select the "Export" tab at the top of the main

pane.



***B3.*** Then, under 'Export method:', select "Custom" to show all export options.



***B4.*** Under '**Format**:' make sure it shows SQL



***B5.*** Under '**Output**':

1. Select 'Save output to a file'
2. **Do NOT Select** "Save on server."

(Note: only applicable to EEECS LAMP host, not applicable for local host on your own PC)

***B6.*** Under '**Format-specific options**' (and scroll down to near the end of this section):

1. make sure that "structure" is selected,
2. **do NOT select** 'data'
3. **do NOT select** 'structure and data'

***B7.*** Click 'Go'.

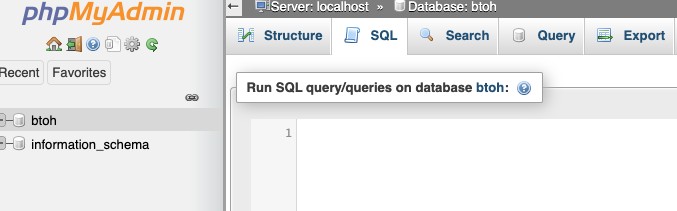
A file named doggo.sql should be downloaded to your PC. Open this file using a text editor of your choice (e.g. Notepad++ or TextEdit) and have a look at the contents (paste a copy here).

Is this what you expected to see? Note the various options and comments which are there in addition to the standard table creation SQL.

|  |
| --- |
| -- phpMyAdmin SQL Dump  -- version 5.1.1  -- https://www.phpmyadmin.net/  --  -- Host: localhost  -- Generation Time: Mar 07, 2022 at 10:26 AM  -- Server version: 10.4.21-MariaDB  -- PHP Version: 7.4.23  SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";  START TRANSACTION;  SET time\_zone = "+00:00";  /\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;  /\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;  /\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;  /\*!40101 SET NAMES utf8mb4 \*/;  --  -- Database: `hlau05`  --  -- --------------------------------------------------------  --  -- Table structure for table `doggo`  --  CREATE TABLE `doggo` (  `id` int(11) NOT NULL,  `Name` varchar(255) NOT NULL,  `Breed` varchar(255) NOT NULL,  `Birth` datetime DEFAULT NULL,  `Floof` tinyint(1) DEFAULT NULL,  `Hooman` varchar(255) DEFAULT NULL,  `Notes` text DEFAULT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  --  -- Indexes for dumped tables  --  --  -- Indexes for table `doggo`  --  ALTER TABLE `doggo`  ADD PRIMARY KEY (`id`),  ADD KEY `id` (`id`),  ADD KEY `Name` (`Name`);  --  -- AUTO\_INCREMENT for dumped tables  --  --  -- AUTO\_INCREMENT for table `doggo`  --  ALTER TABLE `doggo`  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT;  COMMIT;  /\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;  /\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;  /\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/; |

***B8.*** Now we have a backup of our schema, write an SQL statement to **delete** doggo table from your database.  
DROP TABLE doggo

***B9.*** With your doggo.sql still opened in a text editor, copy and paste the entire content of the doggo.sql file on the SQL prompt in phpMyAdmin.



***B10.*** Click 'Go' to execute to recreate the doggo table. What will happen to your database?  
 my b is back

# Section C. Referential Integrity

## Perform the following tasks for all the questions listed in this section:

* **write** an **SQL query**,
* **execute** the query,
* **Check** your output **table** gives the expected result,
* **paste** the SQL **query** statement into this document.

Here is the schema for two tables:

Table: ship

Table: sailor

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Additional** |
| id | INT | AUTO INCREMENT, NOT NULL, PRIMARY KEY |
| name | VARCHAR(255) | NOT NULL |

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Additional** |
| id | INT | AUTO INCREMENT, NOT NULL, PRIMARY KEY |
| shipid | INT | FOREIGN KEY to ship.id, NOT NULL |
| name | VARCHAR(255) | NOT NULL |

***C1.*** Create the above two tables including the foreign key link (that should cascade on both delete and update).  
CREATE TABLE ship (id int AUTO\_INCREMENT not null primary KEY, name varchar(255) not null)  
CREATE TABLE sailor (id int AUTO\_INCREMENT not null primary KEY, shipid int not null, name varchar(255) not null, FOREIGN key(shipid) REFERENCES ship(id))

***C2.*** In the table sailor, add a new sailor named "Bob" who is stationed on a ship with a

ship identification of 1 (record the SQL and the result - what happened, and why?).

INSERT INTO sailor(Name, shipid)

VALUES ('Bob', 1)  
  
cannot la becoz there’s no record in sip and lke shipid is foreign key

***C3.*** Add a ship called "The Good Ship Queens"  
INSERT INTO ship(Name)

VALUES ('The Good Ship Queens')

***C4.*** Repeat ***C2*** (add Bob to ship 1) - what happens this time?  
ok la fin9lly coz have thigs let you reference

***C5.*** Add another ship called "SS EEECS".  
INSERT INTO ship(Name)

VALUES ('SS EEECS')

***C6.*** Add a sailor called "Jane" who serves on the SS EEECS.  
UPDATE sailor SET name = 'Jane' WHERE shipid = 2;

***C7.*** Delete "The Good Ship Queens" - what happens to Bob?

***C8.*** Delete the sailor table.

***C9.*** Recreate the table sailor, to include the foreign key linking to ship.id, but this time set it to ON DELETE NULL and ON UPDATE CASCADE (**note**: you may have to change the shipid field slightly as well).

***C10.*** Add "Jane" to serve on the SS EEECS.

***C11.*** Delete the ship 'SS EEECS'. What is the SQL query statement used? What happens to

Jane's ship?

***C12.*** Re-add the "SS EEECS" as a ship.

# Section D. Many-to-Many with Link Tables

## Perform the following tasks for all the questions listed in this section:

* **write** an **SQL query**,
* **execute** the query,
* **Check** your output **table** gives the expected result,
* **paste** the SQL **query** statement into this document.

The relationship modelled between ship and sailor is one to many (each ship may have one or more sailors and each sailor may have 0 or 1 ships) but often we want to model many-to- many relationships using link tables.

Let's consider the idea of mascots - a ship may have zero or more mascots and each mascot may have zero or more ships.



To model this relationship we need to create a link table.



The mascot table looks like this:

Table: mascot

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Additional** |
| id | INT | AUTO INCREMENT, NOT NULL, PRIMARY KEY |
| name | VARCHAR(255) |  |

And to link the ships to mascots we have another table:

Table: shipmascot

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Additional** |
| shipid | INT | FOREIGN KEY to ship.id, NOT NULL |
| mascotid | INT | FOREIGN KEY to mascot.id, NOT NULL |

**Updates and deletes should cascade for all foreign keys in shipmascot.**

***D1.*** Implement the two tables in SQL.

***D2.*** Add a mascot called "Fred the Cat".

***D3.*** Assign "Fred the Cat" to be the mascot of the ship "SS EEECS".

***D4.*** For the ship table created in Exercise C, add the new ship "USS Enterprise".

***D5.*** Assign "Fred the Cat" to also be a mascot of the ship "USS Enterprise".

***D6.*** Add another mascot "Daisy the Dolphin" who is also a mascot of the ship "USS Enterprise".

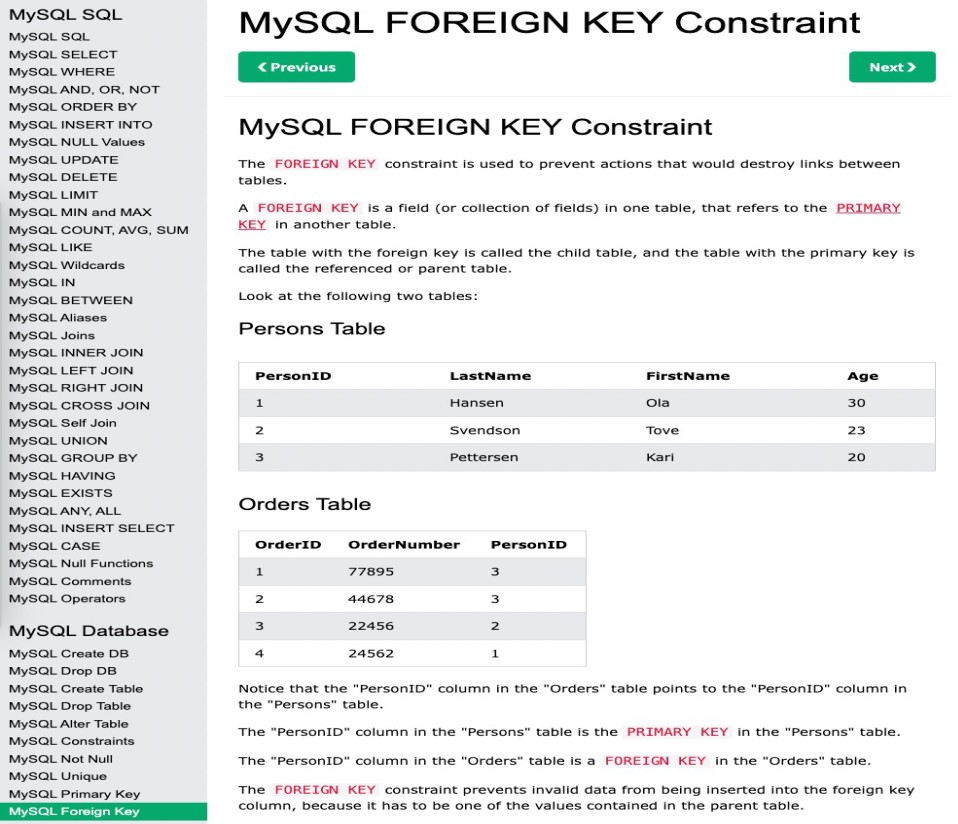
***D7.*** Write an SQL Join statement to output all the columns from the table ship, shipmascot and mascot.

***D8.*** Compare the output from D7 and the ER Diagram below. What are your observations?



# Section E. Further Practice on W3Schools website

Open up the SQL tutorial (link: <https://www.w3schools.com/mySQl/default.asp>) and scroll down to the "**SQL Database**" section and follow through the topics up to and including "**SQL Foreign Key**" - this will cover the work we've already done but in a fresh way.



# Section F. Preparing for the Project

The assignment has three parts, the largest of which is creating a schema (creating, inserting, and exploiting).

With the taught material up to and including this practical you should have covered everything needed to create the schema (note the exploitation [search queries] have not been covered yet - just the creation).

Take this time to look through the project and see if you think you are able to create the tables needed.

If not then raise a question on the discussion forum on CSC1023 CANVAS (<https://canvas.qub.ac.uk/courses/14922/discussion_topics/187483>) pointing out what area you are uncertain on - and we can have this clarified on the discussion forum or during further in teaching sessions.