Introduction to Databases

Author: Akhil Pramod

Coauthor: Ernest Edinboro

Date: Summer 2020

Question	Answers
What type of database type does Shodor use?	Relational
What DBMS does Shodor use?	mySQL

MySQL Building My First Table

SQL	MySQL Syntax	Results
Building Table and Table Info	Write proper SQL syntax to query the People table using Sequel Pro.	What Was the Result

CREATE TABLE
Step1:
	`CREATE TABLE People (`
	`  id int(11) unsigned NOT NULL auto_increment,`
	`  firstName varchar(255) default NULL,`
	`  lastName varchar(255) default NULL,`
	
	`  favColor varchar(255) default NULL,`
	`  shoeSize decimal (3,1)default NULL,`
	`  PRIMARY KEY  (id)`
	`) ENGINE=InnoDB DEFAULT CHARSET=utf8;`
		Made different column headers for a new table called People. There are no entries in the table. Set the parameters for the entries for each column. There is an ID for each entry, which increments by 1 for each entry

INSERT INTO
Step2:
	`INSERT INTO People (firstName, lastName,favColor, shoeSize)`
	`VALUES`
	`('Ernie','Edinboro','Orange',11.5),`
	`('Aras','Rose','Blue',9.5),`
	`('Inaayah','Galloway', 'Red', 10),`
	`('Ruby','Chung','Light Green', 9),`
	`('Amit','Lloyd', 'Purple', 10),`
	`('Jerry','Ford', 'Black', 4),`
	`('Miruna','Woods', 'Red', 10),`
	`('Jamie','Pemberton','Purple',9.5),`
	`('Avani','Lister', 'Blue', 12),`
	`('Hanifa ','Phelps', 'Blue', 7.5),`
	`('Madina','Lowe', 'Red', 10.5),`
	`('Elsie', 'Wise', 'Cupcake', 8.5),`
	`('Jacques','Oakley', 'Yellow', 3),`
	`('Andrea','Waller', 'Green', 13.5),`
	`('Hadi','Washington', 'Red', 12.5),`
	`('Keiron','Schofield', 'Blue', 9.5),`
	`('Farrell','Mosley', 'Orange', 8),`
	`('Markus','Collins', 'Blue', 12),`
	`('Ernie','Edinboro', 'Green', 11),`
	`('Ernie','Edinboro', 'Blue',6),`
	`('Fynley','Merce', 'Purple', 3.5);`
		This put in entries for the table, each having its own unique ID.

ALTER Table **Step3:**	`ALTER TABLE People` `ADD coinFlip VARCHAR(5) NOT NULL;`	**This added a new column called coinFlip to the table.**

Add Data to the newly created coinFlip field.
Step4:
	`INSERT INTO People (id, coinFlip)`
	`VALUES`
	`('1','Tails'),('2','Tails'),('3','Tails'),`
	`('4','Heads'),('5','Heads'),('6','Tails'),`
	`('7','Heads'),('8','Tails'),('9','Heads'),`
	`('10','Heads'),('11','Tails'),('12','Heads'),`
	`('13','Heads'),('14','Heads'),('15','Tails'),`
	`('16','Heads'),('17','Heads'),('18','Heads'),`
	`('19','Heads'),('20','Tails'),('21','Heads')`
	`ON DUPLICATE KEY UPDATE`
	`id=VALUES(id), coinFlip=VALUES(coinFlip);`
		Skip
		This added coin faces to each entry under the coinFlip column in the table.

MySQL Syntax Guide:
Day 1

SQL	MySQL Syntax	Results
Other usefully SQL for Building Table and Table Info	Write proper SQL syntax to query the People table using Sequel Pro.	What Was the Result
SHOW TABLES	`SHOW TABLES;`	Just shows the tables in the database I am in. In my case, it showed just People.
DESCRIBE TABLE	`DESCRIBE People;`	Shows the different fields in the table as well as the certain parameters for each individual field.
DROP TABLE Don't run unless you really want to get rid of a table and all its data. You will have to start over. DO NOT RUN QUERY!	`DROP TABLE People;`	DO NOT RUN this query unless you want to destroy a table and data. Once done it cannot be undone.

NOT NULL	`CREATE TABLE People (`
	`  id int(11) unsigned NOT NULL auto_increment`
	`}`
	`Note: this is a snippet of the code from the People table`
		This tells the column that it should not accept any values that equal NULL or 0.

PRIMARY KEY	`CREATE TABLE People (` `  id int(11) unsigned NOT NULL auto_increment,`	This makes it so that the records in the table can

	PRIMARY KEY (id) } Note: this is another snippet from the People table code	be identified. If the Primary Key is referenced, the entry will be pulled up. (This set ID to be primary key, so that we can refer to the ID)
AUTO_INCREMENT	CREATE TABLE People (id int(11) unsigned NOT NULL auto_increment } Note: this is a snippet of the code from the People table	This automatically incremented the value of ID up by 1 each time an entry is entered.
DELETE **Be careful! DO NOT RUN QUERY!**	DELETE FROM People WHERE favColor= 'Green';	**Do not run query**
Useful SQL Syntax		
UPDATE	UPDATE People, SET favColor = "Green" WHERE id = 3;	Goes to the table called People, then goes to the WHERE command and sees the id 3, and then changes favColor value to Green.
SELECT	SELECT * FROM People;	Selects the whole table called People, and then displays it above the SQL command text box.
WHERE	SELECT * FROM People; WHERE favColor = "Green";	Looks at the table, and then brings back all of

		the entries that have the favColor = Green. Displays the entries above the SQL command text box.

ORDER BY	`SELECT * FROM People`
	
	`ORDER BY shoeSize;`
		Orders the table People by the shoe size in ascending order, and then displays the results.

AND, OR, Not	`SELECT * FROM People`
	
	`WHERE shoeSize = 9.5 AND favColor= "Blue";`
	
	`SELECT * FROM People`
	
	`WHERE favColor = "Green" OR favColor= "Blue";`
	
	`SELECT * FROM People`
	
	`WHERE NOT firstName= "Ernie";`
		Locates entries with the shoeSize value of 9.5 and the favColor value of Blue. It then displays the entries. (Aras Rose and Keiron Schofield) ---<u>AND</u> Locates entries with favColor equal to either Blue or Green, and then displays them. ---<u>OR</u> Locates entries that have firstName = Ernie, and then DOES NOT display them. It displays all of the other entries. ---<u>NOT</u>

LIKE	`SELECT * FROM People`
	
	`WHERE firstName LIKE "a%";`
		Locates entries where firstName starts with the letter A, and displays them.

IN	`SELECT * FROM People`
	
	`WHERE favColor IN ("Blue", "Green", "Purple");`
		Locates entries where favColor is equal to Blue, Green, or Purple, and then displays them.

BETWEEN	`SELECT * FROM People`
	
	`WHERE shoeSize BETWEEN 5 AND 10;`
		Locates entries where the shoeSize value is between 5 and 10, and then displays those entries.

ALIAS	`SELECT id AS firstName`
	
	`FROM People;`
		Changes all of the first names into their respective id, and then returns the firstName column with just IDs.

SQL Operators	**Best to use Arithmetic and Comparison Operators on shoeSize field**	
Arithmetic Operators		
Add	`SELECT 30+20;`	`Returns the sum of 30 and 20. - 50`
Subtraction	`SELECT 30-20;`	`Returns the difference between 30 and 20. - 10`
Multiplication	`SELECT 30*20;`	`Returns the product of 30 times 20. - 600`
Division	`SELECT 40/20;`	`Returns 40 divided by 20. - 2`
Modulos	`SELECT 51%16;`	`Returns the remainder when 51 is divided by 16 - 3`

Comparison Operators		

Equal to	`SELECT * FROM People`
	
	`WHERE shoeSize = 10;`
		Locates entries where the shoeSize is equal to 10, then selects and displays them.

Greater than >	`SELECT * FROM People`
	
	`WHERE shoeSize > 8;`
		Locates entries where the shoeSize is greater than 8, then selects and displays them.

Less than <	`SELECT * FROM People`
	
	`WHERE shoeSize < 8;`
		Locates entries where the shoeSize is less than 8, then selects and displays them.

Greater than equal to >=	`SELECT * FROM People`
	
	`WHERE shoeSize >= 10;`
		Locates entries where the shoeSize is greater than or equal to 10, then selects and displays them.

Less than equal to <=	`SELECT * FROM People`
	
	`WHERE shoeSize <= 10;`
		Locates entries where the shoeSize is less than or equal to 10, then selects and displays them.

Not equal to <>	`SELECT * FROM People`
	
	`WHERE shoeSize <> 10;`
		Locates entries where the shoeSize is NOT EQUAL to 10, then selects and displays them.

Useful Math Functions		

Count	`SELECT COUNT(shoeSize)`
	
	`FROM People;`
		Counts the number of entries where the shoeSize is shown. - 21

Avg	`SELECT AVG(shoeSize)`
	
	`FROM People;`
		Takes the average shoe size among all of the different shoe sizes in the table, and displays it. - 9.09524

Sum	`SELECT SUM(shoeSize)`
	
	`FROM People;`
		Adds up all of the shoeSize values shown on the table and then displays it.

SQL Data Types

List 10 SQL Data Types **You cannot user INT or VARCHAR**	**What types of Data do they Store?**	
CHAR	It stores a fixed length string. This string can have numbers, letters, and special characters.	
BINARY	Stores binary byte strings. Specifies the column length in bytes.	
BIT	It stores a certain integer representing the number of bits per value. The size can be from 1 to 64.	
DATE	Stores the date, uses numbers and dashes (-)	
TEXT	Holds a string of text, such as letters. Max string is 65,525 bytes	
BLOB	Binary Large Objects. Can hold 65,535 bytes of data.	
ENUM	Stores different predefined values (up to 65,535). String object	
SET	Stores values like text, numbers, characters, and can have 0 or more values.	
BOOLEAN	Equal to BOOL; Stores true and false data for 0 and nonzero.	
FLOAT	Stores a floating point number (numerical data)	

Writing SQL queries

SQL MINI Challenge Cannot use INT or VARCHAR	Write SQL query to answer the questions	Result
What is Elsie's favorite color?	`SELECT favColor FROM People` `WHERE firstName = "Elsie";`	Cupcake
Whose shoe size is 3? Show First and last name	`SELECT firstName, lastName FROM People` `WHERE shoeSize = 3;`	Jacques Oakley
Whose favorite color is Cupcake? Show First Name only	`SELECT firstName FROM People` `WHERE favColor = "Cupcake";`	Elsie
What is the average shoe size?	`SELECT AVG(shoeSize)` `FROM People;`	9.09524

Whose favorite color is Red?
Show Last Name only ORDER BY ASCending order.
	`SELECT lastName FROM People`
	`WHERE favColor = "Red"`
	`ORDER BY lastName;`
		Lowe
		Washington
		Woods

What is the favorite color of people with shoe size 10?	`SELECT favColor FROM People` `WHERE shoeSize = 10;`	Green Purple Red

Select all data for user(s) id =13	`SELECT * FROM People`
	
	`WHERE id = 13;`
		Jacques Oakley, Yellow, 3.0, Heads

Select first name for user(s) where favorite color is red	`SELECT firstName FROM People`
	
	`WHERE favColor = "red";`
		Miruna
		Madina
		Hadi

Select favorite color where shoe size greater than or equal to 12	`SELECT favColor FROM People`
	
	`WHERE shoeSize >= 12;`
		Blue
		Green
		Red
		Blue

Select shoe size where first name = Farrell	`SELECT shoeSize FROM People`
	
	`WHERE firstName = "Farrell";`
		8.0

Select lastname where favorite color is black	`SELECT lastName FROM People`
	
	`WHERE favColor = "Black";`
		Ford

Insert into People Jimmy, Johns, Black, 11, Tails	`INSERT INTO People (firstName, lastName,favColor, shoeSize, coinFlip)`
	
	`VALUES`
	
	`('Jimmy','Johns','Black', 11, "Tails");`
		Created row with ID 22, first name Jimmy last name Johns favorite color black, shoe size 11, and coin flip face tails.

Update First name of id 1 to Ernest	`UPDATE People`
	
	`SET firstName = "Ernest"`
	
	`WHERE id = 1;`
		Changed the first name of entry 1 with id 1 from Ernie to Ernest.

STOP, you are done with Day 1

MySQL Syntax Guide:
Day 2 or Day 3

Database Interaction	MySQL Syntax using People table	What where the results?
PRIMARY KEY	`PRIMARY KEY  (id)`	This made it so that the entries in the table were identifiable by the id number entries.
FOREIGN KEY	`FOREIGN KEY (peopleId) REFERENCES exPeople (id)`	This made it so that the id from the exPeople table becomes the main reference point while referring to the entries in the exPeople table, which had names. This foreign key associates and relates the color and shoe size and coin flip to the first and last name.

JOIN		

INNER JOIN	`SELECT exColor.id, exPeople.fname, exColor.favColor`
	
	`FROM exColor`
	
	`INNER JOIN exPeople ON exColor.peopleID=exPeople.id;`
		Provided a mini table, with the first name and their corresponding favorite color with their id on the left of their name.

LEFT JOIN	`SELECT exPeople.fname, exColor.id`
	
	`FROM exPeople`
	
	`LEFT JOIN exColor ON exPeople.id = exColor.peopleid;`
		This displayed the id of the color from the exColor Table. This made it easier to associate the first name with the color id they had.
		Hadi 1
		Markus 2
		Farrell 3
		Avani 4
		Ernie 5

RIGHT JOIN	`SELECT exPeople.fname, exShoe.id`
	
	`FROM exPeople`
	
	`RIGHT JOIN exShoe ON exPeople.id = exShoe.peopleid;`
		This displayed the id of the shoeSize from the exShoe table next to the first name, similar to the previous one. Allows the first name to be easily associated with the shoeSize ID.

GROUP BY	`SELECT COUNT(id), favColor`
	
	`FROM People`
	
	`GROUP BY favColor`
		Provides the number of people that like certain colors. Produces a table with the count on the left side, and the favColor on the right side.

		<table><tr><th>COUNT(id)</th><th>favColor</th></tr><tr><td>2</td><td>Black</td></tr><tr><td>6</td><td>Blue</td></tr><tr><td>1</td><td>Cupcake</td></tr><tr><td>3</td><td>Green</td></tr><tr><td>1</td><td>Light Green</td></tr><tr><td>2</td><td>Orange</td></tr><tr><td>3</td><td>Purple</td></tr><tr><td>3</td><td>Red</td></tr><tr><td>1</td><td>Yellow</td></tr></table>

SQL Snippet for creating Relational DB	`CONSTRAINT CoinFlip_ibfk_1`
	`FOREIGN KEY (peopleId) REFERENCES People (id)`
	`ON DELETE CASCADE ON UPDATE CASCADE,`
		`Establishes a relation between 2 tables, and allows values from the main table to be referenced in other tables. In this, the id from exPeople became the peopleId in the other tables, which associated the first name with its respective values (favorite color, coin flip, shoe size) in the other tables.`

SQL MINI Challenge	**SQL Syntax used**	**Result**
? Guess you don't have to do anything		
? Guess you don't have to do anything		
? Guess you don't have to do anything		
? Guess you don't have to do anything		

? Guess you don't have to do anything		
? Guess you don't have to do anything		