**MY SQL CODE**

1. **CODE: Showing Databases**

To list available databases:

show databases;

1. **CODE: Creating Databases**

The general command for creating a database:

CREATE DATABASE <database\_name>;  
  
A specific example:

CREATE DATABASE soap\_store;

1. **CODE: Dropping and Using Databases**

To drop a database:

DROP DATABASE <database-name>;

To use a database:

USE <database-name>;

1. **CODE: Creating Tables**

Creating Tables:

CREATE TABLE cats (

name VARCHAR(50),

age INT

);

CREATE TABLE dogs (

name VARCHAR(50),

breed VARCHAR(50),

age INT

);

1. **CODE: How Do We Know It Worked?**

SHOW tables;

SHOW COLUMNS FROM cats;

DESC cats;

1. **CODE: Dropping Tables**

To drop a table:

DROP TABLE <table-name>;

To specifically drop the cats table:

DROP TABLE cats;

1. **Tables Basics Activity**

Create the table:

CREATE TABLE pastries

(

name VARCHAR(50),

quantity INT

);

View tables:

SHOW TABLES;

View details of pastries table:

DESC pastries;

Delete the whole pastries table:

DROP TABLE pastries;

1. **CODE: INSERT: The Basics**

-- Re-create the cats table (I dropped it in a previous video)

CREATE TABLE cats (

name VARCHAR(50),

age INT

);

Insert a cat:

INSERT INTO cats (name, age) VALUES ('Blue Steele', 5);

And another:

INSERT INTO cats (name, age) VALUES ('Jenkins', 7);

1. **CODE: A Quick Preview of SELECT**

To view all rows in our table:

SELECT \* FROM cats;

1. **CODE: Multi-inserts**

-- Single insert (switching order of name and age)

INSERT INTO cats (age, name) VALUES (2, 'Beth');

-- Multiple Insert:

INSERT INTO cats (name, age) VALUES ('Meatball', 5), ('Turkey', 1), ('Potato Face', 15);

1. **SOLUTION: INSERT Exercise**

-- INSERT Challenge Solution Code

CREATE TABLE people ( first\_name VARCHAR(20), last\_name VARCHAR(20),

age INT );

INSERT INTO people(first\_name, last\_name, age) VALUES ('Tina', 'Belcher', 13);

INSERT INTO people(age, last\_name, first\_name) VALUES (42, 'Belcher', 'Bob');

INSERT INTO people(first\_name, last\_name, age) VALUES ('Linda', 'Belcher', 45),  ('Phillip', 'Frond', 38), ('Calvin', 'Fischoeder', 70);

DROP TABLE people;

SELECT \* FROM people;

SHOW TABLES;

1. **CODE: Working With NOT NULL**

Using NOT NULL:

CREATE TABLE cats2 (name VARCHAR(100) NOT NULL,age INT NOT NULL);

1. **CODE: Adding DEFAULT Values**

Define a table with a DEFAULT name specified:

CREATE TABLE cats3 ( name VARCHAR(20) DEFAULT 'no name provided',

age INT DEFAULT 99 );

Notice the change when you describe the table:

DESC cats3;

Insert a cat without a name:

INSERT INTO cats3(age) VALUES(13);

Or a nameless, ageless cat:

INSERT INTO cats3() VALUES();

Combine NOT NULL and DEFAULT:

CREATE TABLE cats4 ( name VARCHAR(20) NOT NULL DEFAULT 'unnamed', age INT NOT NULL DEFAULT 99 );

1. **CODE: Introducing Primary Keys**

-- One way of specifying a PRIMARY KEY

CREATE TABLE unique\_cats (cat\_id INT PRIMARY KEY, name VARCHAR(100) NOT NULL, age INT NOT NULL);

-- Another option:

CREATE TABLE unique\_cats2 (cat\_id INT, name VARCHAR(100) NOT NULL,age INT NOT NULL,PRIMARY KEY (cat\_id));

1. **CODE: Working With AUTO\_INCREMENT**

--  AUTO\_INCREMENT

CREATE TABLE unique\_cats3 (cat\_id INT AUTO\_INCREMENT,name VARCHAR(100) NOT NULL,age INT NOT NULL,PRIMARY KEY (cat\_id));

1. **Create Table/Insert Exercise**

-- Defining employees table

CREATE TABLE employees (

id INT AUTO\_INCREMENT,

first\_name VARCHAR(255) NOT NULL,

last\_name VARCHAR(255) NOT NULL,

middle\_name VARCHAR(255),

age INT NOT NULL,

current\_status VARCHAR(255) NOT NULL DEFAULT 'employed',

PRIMARY KEY(id));

-- Another way of defining the primary key:

CREATE TABLE employees (

id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(255) NOT NULL,

last\_name VARCHAR(255) NOT NULL,

middle\_name VARCHAR(255),

age INT NOT NULL,

current\_status VARCHAR(255) NOT NULL DEFAULT 'employed');

-- A test INSERT:

INSERT INTO employees(first\_name, last\_name, age) VALUES

('Dora', 'Smith', 58);

1. **Drop the current cats table (if you have one)**

DROP TABLE cats;

-- Create the new cats table:

CREATE TABLE cats (

cat\_id INT AUTO\_INCREMENT,

name VARCHAR(100),

breed VARCHAR(100),

age INT,

PRIMARY KEY (cat\_id)

);

-- Insert some cats:

INSERT INTO cats(name, breed, age)

VALUES ('Ringo', 'Tabby', 4),

('Cindy', 'Maine Coon', 10),

('Dumbledore', 'Maine Coon', 11),

('Egg', 'Persian', 4),

('Misty', 'Tabby', 13),

('George Michael', 'Ragdoll', 9),

('Jackson', 'Sphynx', 7);

1. **To get all the columns:**

SELECT \* FROM cats;

-- To only get the age column:

SELECT age FROM cats;

-- To select multiple specific columns:

SELECT name, breed FROM cats;

1. **Where clause:**

-- Use where to specify a condition:

SELECT \* FROM cats WHERE age = 4;

 SELECT \* FROM cats WHERE name ='Egg';

1. **CRUD BASICS- EXERCISE SOLUTION CODE**

SELECT cat\_id FROM cats;

SELECT name, breed FROM cats;

SELECT name, age FROM cats WHERE breed='Tabby';

SELECT cat\_id, age FROM cats WHERE cat\_id=age;

SELECT \* FROM cats WHERE cat\_id=age;

1. **ALIASES**

-- Use 'AS' to alias a column in your results (it doesn't actually change the name of the column in the table)

SELECT cat\_id AS id, name FROM cats;

1. **UPDATE**

#### Updating Data

Change tabby cats to shorthair:

UPDATE cats SET breed='Shorthair' WHERE breed='Tabby';

Another update:

UPDATE cats SET age=14 WHERE name='Misty';

1. **UPDATE EXERCIES**

#### Update Challenges Solution

SELECT \* FROM cats WHERE name='Jackson';

UPDATE cats SET name='Jack' WHERE name='Jackson';

SELECT \* FROM cats WHERE name='Jackson';

SELECT \* FROM cats WHERE name='Jack';

SELECT \* FROM cats WHERE name='Ringo';

UPDATE cats SET breed='British Shorthair' WHERE name='Ringo';

SELECT \* FROM cats WHERE name='Ringo';

SELECT \* FROM cats;

SELECT \* FROM cats WHERE breed='Maine Coon';

UPDATE cats SET age=12 WHERE breed='Maine Coon';

SELECT \* FROM cats WHERE breed='Maine Coon';

1. **DELETE**

-- Delete all cats with name of 'Egg':

DELETE FROM cats WHERE name='Egg';

-- Delete all rows in the cats table:

DELETE FROM cats;

1. **DELETE EXERCISE**

-- Delete all 4 year old cats:

DELETE FROM cats WHERE age=4;

-- Delete cats where cat\_id is the same as their age:

DELETE FROM cats WHERE cat\_id=age;

-- Delete all cats:

DELETE FROM cats;

1. **CRUD CHALLENGE SOLUTION – CREATE**

CREATE DATABASE shirts\_db;

USE shirts\_db;

CREATE TABLE shirts (

shirt\_id INT AUTO\_INCREMENT PRIMARY KEY,

article VARCHAR(50),

color VARCHAR(50),

shirt\_size VARCHAR(5),

last\_worn INT

);

DESC shirts;

INSERT INTO shirts (article, color, shirt\_size, last\_worn)

VALUES

('t-shirt', 'white', 'S', 10),

('t-shirt', 'green', 'S', 200),

('polo shirt', 'black', 'M', 10),

('tank top', 'blue', 'S', 50),

('t-shirt', 'pink', 'S', 0),

('polo shirt', 'red', 'M', 5),

('tank top', 'white', 'S', 200),

('tank top', 'blue', 'M', 15);

INSERT INTO shirts (article, color, shirt\_size, last\_worn)

VALUES ('polo shirt','purple', 'M', 50);

1. **CRUD CHALLENGE SOLUTION – READ**

SELECT article, color FROM shirts;

SELECT \* FROM shirts WHERE shirt\_size='M';

SELECT article, color, shirt\_size, last\_worn FROM shirts WHERE shirt\_size='M';

1. **CRUD CHALLENGE SOLUTION – UPDATE**

UPDATE shirts

SET

shirt\_size = 'L'

WHERE

article = 'polo shirt';

UPDATE shirts

SET

last\_worn = 0

WHERE

last\_worn = 15;

UPDATE shirts

SET

color = 'off white',

shirt\_size = 'XS'

WHERE

color = 'white';

1. **CRUD CHALLENGE SOLUTION – DELETE**

SELECT \* FROM shirts WHERE last\_worn=200;

DELETE FROM shirts WHERE last\_worn=200;

SELECT \* FROM shirts WHERE article='tank top';

DELETE FROM shirts WHERE article='tank top';

SELECT \* FROM shirts;

DELETE FROM shirts;

DROP TABLE shirts;

show tables;

DESC shirts;

1. **STRING FUNCTIONS- LOAD BOOKS DATA**

CREATE TABLE books

(

book\_id INT AUTO\_INCREMENT,

title VARCHAR(100),

author\_fname VARCHAR(100),

author\_lname VARCHAR(100),

released\_year INT,

stock\_quantity INT,

pages INT,

PRIMARY KEY(book\_id)

);

INSERT INTO books (title, author\_fname, author\_lname, released\_year, stock\_quantity, pages)

VALUES

('The Namesake', 'Jhumpa', 'Lahiri', 2003, 32, 291),

('Norse Mythology', 'Neil', 'Gaiman',2016, 43, 304),

('American Gods', 'Neil', 'Gaiman', 2001, 12, 465),

('Interpreter of Maladies', 'Jhumpa', 'Lahiri', 1996, 97, 198),

('A Hologram for the King: A Novel', 'Dave', 'Eggers', 2012, 154, 352),

('The Circle', 'Dave', 'Eggers', 2013, 26, 504),

('The Amazing Adventures of Kavalier & Clay', 'Michael', 'Chabon', 2000, 68, 634),

('Just Kids', 'Patti', 'Smith', 2010, 55, 304),

('A Heartbreaking Work of Staggering Genius', 'Dave', 'Eggers', 2001, 104, 437),

('Coraline', 'Neil', 'Gaiman', 2003, 100, 208),

('What We Talk About When We Talk About Love: Stories', 'Raymond', 'Carver', 1981, 23, 176),

("Where I'm Calling From: Selected Stories", 'Raymond', 'Carver', 1989, 12, 526),

('White Noise', 'Don', 'DeLillo', 1985, 49, 320),

('Cannery Row', 'John', 'Steinbeck', 1945, 95, 181),

('Oblivion: Stories', 'David', 'Foster Wallace', 2004, 172, 329),

('Consider the Lobster', 'David', 'Foster Wallace', 2005, 92, 343);

1. **CONCAT FUNCTION**

SELECT CONCAT('pi', 'ckle');

SELECT CONCAT(author\_fname,' ', author\_lname) AS author\_name FROM books;

SELECT CONCAT\_WS('-',title, author\_fname, author\_lname) FROM books;

1. **SUBSTRING FUNCTION**

SELECT SUBSTRING('Hello World', 1, 4);

SELECT SUBSTRING('Hello World', 7);

SELECT SUBSTRING('Hello World', -3);

SELECT SUBSTRING(title, 1, 10) AS 'short title' FROM books;

SELECT SUBSTR(title, 1, 10) AS 'short title' FROM books;

1. **COMBINING STRING FUNCTIONS**

SELECT CONCAT

(

SUBSTRING(title, 1, 10),

'...'

) AS 'short title'

FROM books;

1. **REPLACE FUNCTIONS**

SELECT REPLACE('Hello World', 'Hell', '%$#@');

SELECT REPLACE('Hello World', 'l', '7');

SELECT REPLACE('Hello World', 'o', '0');

SELECT REPLACE('HellO World', 'o', '\*');

SELECT

REPLACE('cheese bread coffee milk', ' ', ' and ');

SELECT REPLACE(title, 'e ', '3') FROM books;

SELECT REPLACE(title, ' ', '-') FROM books;

1. **REVERSE FUNCTION**

SELECT REVERSE('Hello World');

SELECT REVERSE('meow meow');

SELECT REVERSE(author\_fname) FROM books;

SELECT CONCAT('woof', REVERSE('woof'));

SELECT CONCAT(author\_fname, REVERSE(author\_fname)) FROM books;

1. **CHAR\_LENGTH**

SELECT CHAR\_LENGTH('Hello World');

SELECT CHAR\_LENGTH(title) as length, title FROM books;

SELECT author\_lname, CHAR\_LENGTH(author\_lname) AS 'length' FROM books;

SELECT CONCAT(author\_lname, ' is ', CHAR\_LENGTH(author\_lname), ' characters long') FROM books;

1. **UPPER & LOWER**

SELECT UPPER('Hello World');

SELECT LOWER('Hello World');

SELECT UPPER(title) FROM books;

SELECT CONCAT('MY FAVORITE BOOK IS ', UPPER(title)) FROM books;

SELECT CONCAT('MY FAVORITE BOOK IS ', LOWER(title)) FROM books;

1. **OTHER STRING FUNCTIONS**

SELECT INSERT('Hello Bobby', 6, 0, 'There');

SELECT LEFT('omghahalol!', 3);

SELECT RIGHT('omghahalol!', 4);

SELECT REPEAT('ha', 4);

SELECT TRIM(' pickle ');

1. **STRING FUNCTION-EXERCISE**

SELECT REVERSE(UPPER('Why does my cat look at me with such hatred?'));

SELECT REPLACE(title, ' ', '->') AS title FROM books;

SELECT

author\_lname AS forwards, REVERSE(author\_lname) AS backwards

FROM

books;

SELECT UPPER(CONCAT(author\_fname, ' ', author\_lname)) AS 'full name in caps' FROM books;

SELECT CONCAT(title, ' was released in ', released\_year) AS blurb FROM books;

SELECT title, CHAR\_LENGTH(title) AS character\_count FROM books;

SELECT

CONCAT(SUBSTR(title, 1, 10), '...') AS short\_title,

CONCAT(author\_lname, ',', author\_fname) AS author,

CONCAT(stock\_quantity, ' in stock') AS quantity FROM books;

1. **REFINING SELECTIONS – Create Required Data**

INSERT INTO books

(title, author\_fname, author\_lname, released\_year, stock\_quantity, pages)

VALUES ('10% Happier', 'Dan', 'Harris', 2014, 29, 256),

('fake\_book', 'Freida', 'Harris', 2001, 287, 428),

('Lincoln In The Bardo', 'George', 'Saunders', 2017, 1000, 367);

1. **REFINING SELECTIONS – Distinct**

SELECT author\_lname FROM books;

SELECT DISTINCT author\_lname FROM books;

SELECT author\_fname, author\_lname FROM books;

SELECT DISTINCT CONCAT(author\_fname,' ', author\_lname) FROM books;

SELECT DISTINCT author\_fname, author\_lname FROM books;

1. **REFINING SELECTIONS - Order By**

SELECT \* FROM books

ORDER BY author\_lname;

SELECT \* FROM books

ORDER BY author\_lname DESC;

SELECT \* FROM books

ORDER BY released\_year;

1. **REFINING SELECTIONS – More On Order By**

SELECT book\_id, author\_fname, author\_lname, pages

FROM books ORDER BY 2 desc;

SELECT book\_id, author\_fname, author\_lname, pages

FROM books ORDER BY author\_lname, author\_fname;

1. **REFINING SELECTIONS – Limit**

SELECT title FROM books LIMIT 3;

SELECT title FROM books LIMIT 1;

SELECT title FROM books LIMIT 10;

SELECT \* FROM books LIMIT 1;

SELECT title, released\_year FROM books

ORDER BY released\_year DESC LIMIT 5;

SELECT title, released\_year FROM books

ORDER BY released\_year DESC LIMIT 1;

SELECT title, released\_year FROM books

ORDER BY released\_year DESC LIMIT 14;

SELECT title, released\_year FROM books

ORDER BY released\_year DESC LIMIT 0,5;

SELECT title, released\_year FROM books

ORDER BY released\_year DESC LIMIT 0,3;

SELECT title, released\_year FROM books

ORDER BY released\_year DESC LIMIT 1,3;

SELECT title, released\_year FROM books

ORDER BY released\_year DESC LIMIT 10,1;

SELECT \* FROM tbl LIMIT 95,18446744073709551615;

SELECT title FROM books LIMIT 5;

SELECT title FROM books LIMIT 5, 123219476457;

SELECT title FROM books LIMIT 5, 50; SELECT title FROM books LIMIT 3;

1. SELECT title FROM books LIMIT 1;
3. SELECT title FROM books LIMIT 10;
5. SELECT \* FROM books LIMIT 1;
7. SELECT title, released\_year FROM books
8. ORDER BY released\_year DESC LIMIT 5;
10. SELECT title, released\_year FROM books
11. ORDER BY released\_year DESC LIMIT 1;
13. SELECT title, released\_year FROM books
14. ORDER BY released\_year DESC LIMIT 14;
16. SELECT title, released\_year FROM books
17. ORDER BY released\_year DESC LIMIT 0,5;
19. SELECT title, released\_year FROM books
20. ORDER BY released\_year DESC LIMIT 0,3;
22. SELECT title, released\_year FROM books
23. ORDER BY released\_year DESC LIMIT 1,3;
25. SELECT title, released\_year FROM books
26. ORDER BY released\_year DESC LIMIT 10,1;
28. SELECT \* FROM tbl LIMIT 95,18446744073709551615;
30. SELECT title FROM books LIMIT 5;
32. SELECT title FROM books LIMIT 5, 123219476457;
34. SELECT title FROM books LIMIT 5, 50;
35. **REFINING SELECTIONS – Like**

SELECT title, author\_fname, author\_lname, pages

FROM books

WHERE author\_fname LIKE '%da%';

SELECT title, author\_fname, author\_lname, pages

FROM books

WHERE title LIKE '%:%';

SELECT \* FROM books

WHERE author\_fname LIKE '\_\_\_\_';

1. **REFINING SELECTIONS – Escaping Wildcards**

-- To select books with '%' in their title:

SELECT \* FROM books

WHERE title LIKE '%\%%';

-- To select books with an underscore '\_' in title:

SELECT \* FROM books

WHERE title LIKE '%\\_%';

1. **REFINING SELECTIONS – Exercise**

SELECT title FROM books WHERE title LIKE '%stories%';

SELECT title, pages FROM books ORDER BY pages DESC LIMIT 1;

SELECT

CONCAT(title, ' - ', released\_year) AS summary

FROM books ORDER BY released\_year DESC LIMIT 3;

SELECT title, author\_lname FROM books WHERE author\_lname LIKE '% %';

SELECT title, released\_year, stock\_quantity

FROM books ORDER BY stock\_quantity LIMIT 3;

SELECT title, author\_lname

FROM books ORDER BY author\_lname, title;

SELECT title, author\_lname

FROM books ORDER BY 2,1;

SELECT

CONCAT(

'MY FAVORITE AUTHOR IS ',

UPPER(author\_fname),

' ',

UPPER(author\_lname),

'!'

) AS yell

FROM books ORDER BY author\_lname;

1. **AGGREGATE FUNCTIONS : - Count Basics**

SELECT COUNT(\*) FROM books;

SELECT COUNT(author\_lname) FROM books;

SELECT COUNT(DISTINCT author\_lname) FROM books;

1. **AGGREGATE FUNCTIONS : - Group By**

SELECT author\_lname, COUNT(\*)

FROM books

GROUP BY author\_lname;

SELECT

author\_lname, COUNT(\*) AS books\_written

FROM

books

GROUP BY author\_lname

ORDER BY books\_written DESC;

1. **AGGREGATE FUNCTIONS : - Min and Max Basics**

SELECT MAX(pages) FROM books;

SELECT MIN(author\_lname) FROM books;

1. **AGGREGATE FUNCTIONS : - Subqueries**

SELECT title, pages FROM books

WHERE pages = (SELECT MAX(pages) FROM books);

SELECT MIN(released\_year) FROM books;

SELECT title, released\_year FROM books

WHERE released\_year = (SELECT MIN(released\_year) FROM books);

1. **AGGREGATE FUNCTIONS : - Grouping By Multiple Columns**

SELECT author\_fname, author\_lname, COUNT(\*)

FROM books

GROUP BY author\_lname, author\_fname;

SELECT CONCAT(author\_fname, ' ', author\_lname) AS author, COUNT(\*)

FROM books

GROUP BY author;

1. **AGGREGATE FUNCTIONS : - Min and Max with Group By**

SELECT author\_lname, MIN(released\_year) FROM books GROUP BY author\_lname;

SELECT author\_lname, MAX(released\_year), MIN(released\_year) FROM books GROUP BY author\_lname;

SELECT

author\_lname,

COUNT(\*) as books\_written,

MAX(released\_year) AS latest\_release,

MIN(released\_year) AS earliest\_release,

MAX(pages) AS longest\_page\_count

FROM books GROUP BY author\_lname;

SELECT

author\_lname,

author\_fname,

COUNT(\*) as books\_written,

MAX(released\_year) AS latest\_release,

MIN(released\_year) AS earliest\_release

FROM books GROUP BY author\_lname, author\_fname;

1. **AGGREGATE FUNCTIONS : - Sum**

SELECT SUM(pages) FROM books;

SELECT author\_lname, COUNT(\*), SUM(pages)

FROM books

GROUP BY author\_lname;

1. **AGGREGATE FUNCTIONS : - Avg**

SELECT AVG(pages) FROM books;

SELECT AVG(released\_year) FROM books;

SELECT

released\_year,

AVG(stock\_quantity),

COUNT(\*) FROM books

GROUP BY released\_year;

1. **AGGREGATE FUNCTIONS : - Exercise**

SELECT COUNT(\*) FROM books;

SELECT released\_year, COUNT(\*)

FROM books GROUP BY released\_year;

SELECT AVG(released\_year)

FROM books GROUP BY author\_lname, author\_fname;

SELECT CONCAT(author\_fname, ' ', author\_lname) AS author, pages FROM books

WHERE pages = (SELECT MAX(pages) FROM books);

SELECT CONCAT(author\_fname, ' ', author\_lname) AS author, pages FROM books

ORDER BY pages DESC LIMIT 1;

SELECT

released\_year AS year,

COUNT(\*) AS '# books',

AVG(pages) AS 'avg pages'

FROM books

GROUP BY released\_year

ORDER BY released\_year;

1. **: -**
2. **: -**