MYSQL

Reference site

https://www.mikedane.com/databases/sql/
https://www.youtube.com/watch?v=7S_tz1z_5bA

Create Table

```
CREATE TABLE student (
    student_id INT,
    name VARCHAR(20),
    major VARCHAR(20),
    PRIMARY KEY (student_id)
);
```

Alter Table

```
ALTER TABLE student ADD gpa DECIMAL(3,2);
ALTER TABLE student drop gpa;
DESCRIBE student;
```

Drop Table

```
DROP TABLE student;
```

Insert

```
INSERT INTO student values(1,'Ranjith','BE');
INSERT INTO student(name,student_id) values('Durai',2);
```

Table Constraints

```
CREATE TABLE student (
student_id INT,
name VARCHAR(20) NOT NULL,
major VARCHAR(20) UNIQUE,
PRIMARY KEY(student_id)
);

CREATE TABLE student (
student_id INT AUTO_INCREMENT,
name VARCHAR(20) NOT NULL DEFAULT 'No Name',
major VARCHAR(20) UNIQUE,
PRIMARY KEY(student_id)
);
```

Update

```
UPDATE student SET name = 'testing';

UPDATE student SET name = 'Hello' WHERE name = 'testing' OR major = 'BE';

UPDATE student SET name = 'hi', major = 'undecided' WHERE major = 'BE';

UPDATE student SET name = 'testing'  where student_id = (select student_id from another
table);
```

DELETE

```
DELETE FROM student WHERE name = 'hi' AND major = 'undecided';
DELETE FROM student;
```

Basic Queries

SELECT name, major FROM student;
SELECT name FROM student where major = 'BE';
SELECT student_id, name FROM student where major = 'BE' OR major = 'IT';
SELECT * FROM student ORDER BY name;
SELECT * FROM student ORDER BY name, major DESC;

<,>,<=,>=,<>,AND, OR

SELECT * FROM student WHERE student_id in (1,2) ORDER BY name DESC;

SELECT * FROM student LIMIT 1;

Company Database

Company Database

Employee

emp_id	first_name	last_name	birth_date	sex	salary	super_id	branch_id
100	David	Wallace	1967-11-17	M	250,000	NULL	1
101	Jan	Levinson	1961-05-11	F	110,000	100	1
102	Michael	Scott	1964-03-15	M	75,000	100	2
103	Angela	Martin	1971-06-25	F	63,000	102	2
104	Kelly	Kapoor	1980-02-05	F	55,000	102	2
105	Stanley	Hudson	1958-02-19	M	69,000	102	2
106	Josh	Porter	1969-09-05	M	78,000	100	3
107	Andy	Bernard	1973-07-22	M	65,000	106	3
108	Jim	Halpert	1978-10-01	M	71,000	106	3

Branch

branch_id	branch_name	mgr_id	mgr_start_date
1	Corporate	100	2006-02-09
2	Scranton	102	1992-04-06
3	Stamford	106	1998-02-13

Client

client_id	client_name	branch_id
400	Dunmore Highschool	2
401	Lackawana Country	2
402	FedEx	3
403	John Daly Law, LLC	3
404	Scranton Whitepages	2
405	Times Newspaper	3
406	FedEx	2

Works_With

emp_id	client_id	total_sales
105	400	55,000
102	401	267,000
108	402	22,500
107	403	5,000
108	403	12,000
105	404	33,000
107	405	26,000
102	406	15,000
105	406	130,000

Branch Supplier

branch_id	supplier_name	supply_type
2	Hammer Mill	Paper
2	Uni-ball	Writing Utensils
3	Patriot Paper	Paper
2	J.T. Forms & Labels	Custom Forms
3	Uni-ball	Writing Utensils
3	Hammer Mill	Paper
3	Stamford Lables	Custom Forms

Labels

	Primary Key
	Foreign Key
	Attribute

```
-- Find the names of all clients who have spent more than
100,000 dollars
```

Basic Queries

```
-- Find all employees
SELECT *
FROM employee;

-- Find all clients
SELECT *
FROM clients;

-- Find all employees ordered by salary
SELECT *
from employee
ORDER BY salary ASC/DESC;

-- Find all employees ordered by sex then name
SELECT *
from employee
ORDER BY sex, name;

-- Find the first 5 employees in the table
SELECT *
from employee
LIMIT 5;

-- Find the first and last names of all employees
SELECT first_name, employee.last_name
FROM employee;

-- Find the forename and surnames names of all employees
SELECT first_name AS forename, employee.last_name AS surname
FROM employee;

-- Find out all the different genders
SELECT DISCINCT sex
FROM employee;

-- Find all male employees
SELECT *
FROM employee
```

```
WHERE sex = 'M';

-- Find all employees at branch 2
SELECT *
FROM employee
WHERE branch_id = 2;

-- Find all employee's id's and names who were born after 1969
SELECT emp_id, first_name, last_name
FROM employee
WHERE birth_day >= ‘1970-01-01’;

-- Find all female employees at branch 2
SELECT *
FROM employee
WHERE branch_id = 2 AND sex = 'F';

-- Find all employees who are female & born after 1969 or who make over 80000
SELECT *
FROM employee
WHERE (birth_day >= '1970-01-01' AND sex = 'F') OR salary > 80000;

-- Find all employees born between 1970 and 1975
SELECT *
FROM employee
WHERE birth_day BETWEEN '1970-01-01' AND '1975-01-01';
SELECT *
FROM employee
WHERE birth_day >= '1970-01-01' AND birth_day <= '1975-01-01';



-- Find all employees named Jim, Michael, Johnny or David
SELECT *
FROM employee
WHERE first_name IN ('Jim', 'Michael', 'Johnny', 'David');
```

Functions

```
-- Find the number of employees
```

```sql
SELECT COUNT(super_id)
FROM employee;

-- Find the average of all employee's salaries
SELECT AVG(salary)
FROM employee;

-- Find the sum of all employee's salaries
SELECT SUM(salary)
FROM employee;

-- Find out how many males and females there are
SELECT COUNT(sex), sex
FROM employee
GROUP BY sex

-- Find the total sales of each salesman
SELECT SUM(total_sales), emp_id
FROM works_with
GROUP BY emp_id;

-- Find the total amount of money spent by each client
SELECT SUM(total_sales), client_id
FROM works_with
GROUP BY client_id;
```

Wildcards

-- % = any # characters, _ = one character

-- Find any client's who are an LLC
SELECT *
FROM client

```
WHERE client_name LIKE '%LLC';

-- Find any branch suppliers who are in the label business
SELECT *
FROM branch_supplier
WHERE supplier_name LIKE '% Label%';

-- Find any employee born on the 10th day of the month
SELECT *
FROM employee
WHERE birth_day LIKE '_____10%';

-- Find any clients who are schools
SELECT *
FROM client
WHERE client_name LIKE '%Highschool%';
```

UNION

```
-- Find a list of employee and branch names
SELECT employee.first_name AS Employee_Branch_Names
FROM employee
UNION
SELECT branch.branch_name
FROM branch;

-- Find a list of all clients & branch suppliers' names
SELECT client.client_name AS Non-Employee_Entities, client.branch_id AS
Branch_ID
FROM client
UNION
SELECT branch_supplier.supplier_name, branch_supplier.branch_id
```

```
FROM branch_supplier;
```

JOINS

```
SELECT employee.emp_id, employee.first_name, branch.branch_name
FROM employee
JOIN branch    -- LEFT JOIN, RIGHT JOIN
ON employee.emp_id = branch.mgr_id;
```

NESTED Queries

```
-- Find names of all employees who have sold over 50,000
SELECT employee.first_name, employee.last_name
FROM employee
WHERE employee.emp_id IN (SELECT works_with.emp_id
                    FROM works_with
                    WHERE works_with.total_sales > 50000);

-- Find all clients who are handles by the branch that Michael Scott manages
-- Assume you know Michael's ID
SELECT client.client_id, client.client_name
FROM client
WHERE client.branch_id = (SELECT branch.branch_id
                    FROM branch
                    WHERE branch.mgr_id = 102);

 -- Find all clients who are handles by the branch that Michael Scott manages
 -- Assume you DONT'T know Michael's ID
```

```
SELECT client.client_id, client.client_name
FROM client
WHERE client.branch_id = (SELECT branch.branch_id
                    FROM branch
                    WHERE branch.mgr_id = (SELECT employee.emp_id
                                        FROM employee
                                        WHERE employee.first_name = 'Michael' AND
employee.last_name ='Scott'
                                        LIMIT 1));


-- Find the names of employees who work with clients handled by the scranton branch
SELECT employee.first_name, employee.last_name
FROM employee
WHERE employee.emp_id IN (
                    SELECT works_with.emp_id
                    FROM works_with
                    )
AND employee.branch_id = 2;

-- Find the names of all clients who have spent more than 100,000 dollars
SELECT client.client_name
FROM client
WHERE client.client_id IN (
                    SELECT client_id
                    FROM (
                        SELECT SUM(works_with.total_sales) AS totals, client_id
                        FROM works_with
                        GROUP BY client_id) AS total_client_sales
                    WHERE totals > 100000
);
```

Select clause

```
-- SELECT first_name, last_name, points, points+10 FROM sql_store.customers;
-- SELECT first_name, last_name, points, points+10 as 'discount value' FROM
sql_store.customers;
-- SELECT name, unit_price AS old_price, unit_price*1.1  AS new_price FROM
products;
```

WHERE

```
SELECT * FROM sql_store.order_items where order_items.order_id = 6 AND
(order_items.quantity * order_items.unit_price) > 30;
```

BETWEEN

```
SELECT * FROM sql_store.customers where birth_date between '1990-01-01' AND
'2000-01-01'
```

LIKE

```
-- SELECT * FROM sql_store.customers where address LIKE '%trail%' OR address
LIKE  '%avenue%';
```

REGEXP

```
SELECT * FROM sql_store.customers where first_name regexp 'ELKA|AMBUR';
SELECT * FROM sql_store.customers where last_name regexp 'EY$|ON$';

SELECT * FROM sql_store.customers where last_name regexp '^MY|SE';
SELECT * FROM sql_store.customers where last_name regexp 'B[RU]';
```

IS NULL

```
 SELECT * FROM sql_store.orders WHERE shipped_date IS NULL;
```

ORDER BY

```
SELECT order_id, product_id, quantity, unit_price, quantity * unit_price as
total_price FROM sql_store.order_items
WHERE order_id = 2
ORDER BY total_price DESC;
```

LIMIT

```
SELECT * FROM sql_store.customers LIMIT 3,4;
SELECT * FROM sql_store.customers ORDER BY points DESC LIMIT 3;
```

INNER JOIN(JOIN)

```
SELECT o.order_id, c.customer_id, c.first_name, c.last_name FROM sql_store.orders
o
JOIN sql_store.customers c ON o.customer_id = c.customer_id;
```

Self Join

```
SELECT e.first_name, e.last_name, m.first_name AS Manager FROM employees e
JOIN employees m ON e.reports_to = m.employee_id;
```

Inner join(Multiple tables)

```
SELECT o.order_id, o.order_date, c.first_name, os.name as status FROM orders o
JOIN customers c ON o.customer_id = c.customer_id
JOIN order_statuses os ON o.status = os.order_status_id;
```

```
SELECT p.payment_id, p.amount, c.name, pm.name FROM payments p
JOIN clients c ON p.client_id = c.client_id
JOIN payment_methods pm ON p.payment_method = pm.payment_method_id;
```

OUTER Joins

Left join and right join

```
SELECT  c.first_name, o.order_date FROM customers c LEFT JOIN orders o ON
c.customer_id = o.customer_id ORDER BY c.first_name DESC;
```

LEFT JOIN multiple tables

```
SELECT o.order_date, o.order_id, c.first_name,
sh.name, st.name
FROM orders o
LEFT JOIN customers c ON o.customer_id = c.customer_id
LEFT JOIN shippers sh ON o.shipper_id = sh.shipper_id
LEFT JOIN order_statuses st ON o.status = st.order_status_id;
```

SELF OUTER JOIN

```
SELECT e.first_name, m.first_name as 'Manager name' FROM employees e
LEFT JOIN employees m ON e.reports_to = m.employee_id ORDER BY
m.first_name;
```

USING

Different key names

```
SELECT p.date,p.amount, c.name, pm.name FROM payments p
JOIN clients c USING(client_id)
JOIN payment_methods pm ON p.payment_method = pm.payment_method_id;
```

NATURAL JOINS

```
SELECT o.order_date, c.first_name
FROM orders o
NATURAL JOIN customers c;
```

INSERT

INSERT INTO products(name, quantity_in_stock, unit_price) values ('product 1',2,25),
('product 2',2,25),
('product 3',2,25);
