DROP DATABASE myDB;

CREATE DATABASE myDB;

USE myDB;

CREATE TABLE epmloyees(

employee\_id INT,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

hourly\_pay DECIMAL(5, 2),

hire\_date DATE

);

DROP TABLE epmloyees

SELECT \* FROM employees;

ALTER TABLE employees

add phone\_number VARCHAR(15);

ALTER TABLE employees

RENAME COLUMN phone\_number to email;

ALTER TABLE employees

MODIFY COLUMN phone\_number VARCHAR(100);

ALTER TABLE employees

MODIFY phone\_number VARCHAR(100)

FIRST;

SELECT \* FROM employees

ALTER TABLE employees

MODIFY phone\_number VARCHAR(100)

AFTER last\_name;

SELECT \* FROM employees

ALTER TABLE employees

DROP COLUMN phone\_number

add email VARCHAR (100);

INSERT INTO employees

VALUES (1, 'Eugene', 'krabs', 25.50, '2023-01-02', 0703);

INSERT INTO employees

VALUES (2, 'Squidward', 'Tentacles', 15.00, '2023-01-03', 0809),

(3, 'Spongebob', 'Squarepants', 12.50, '2023-01-04', 0908),

(4, 'Patrick', 'Star', 12.50, '2023-01-05', 0609),

(5, 'Sandy', 'cheeks', 17.25, '2023-01-06', 0206);

INSERT INTO employees (employee\_id, first\_name, last\_name)

VALUES (6, 'Sheldon', 'Plankton');

SELECT first\_name, last\_name

FROM employees;

SELECT \*

FROM employees

WHERE employee\_id = 1;

SELECT \*

FROM employees

WHERE first\_name = 'SpongeBob';

SELECT \*

FROM employees

WHERE hourly\_pay >= 15;

SELECT \*

FROM employees

WHERE hire\_date <= '2023-01-03'

SELECT \*

FROM employees

WHERE employee\_id != 1;

SELECT \*

FROM employees

WHERE hire\_date IS NULL;

SELECT \*

FROM employees

WHERE hire\_date IS NOT NULL;

UPDATE employees

SET hourly\_pay = 10.25

WHERE employee\_id = 6;

UPDATE employees

SET hourly\_pay = 10.50,

hire\_date = '2023-01-05'

WHERE employee\_id = 6;

UPDATE employees

SET hourly\_pay = 10.50,

hire\_date = '2023-01-05',

phone\_number = 0903

WHERE employee\_id = 6;

UPDATE employees

SET hire\_date = NULL

WHERE employee\_id = 6;

DELETE FROM employees

WHERE employee\_id = 6;

CREATE TABLE test(

my\_date DATE,

my\_time TIME,

My\_datetime DATETIME

);

CREATE TABLE products(

product\_id INT,

product\_name VARCHAR(25),

price DECIMAL(4, 2) NOT NULL

);

ALTER TABLE products

ALTER COLUMN price DECIMAL(4, 2) NOT NULL;

CREATE TABLE dynamics (

employee\_id INT,

first\_name VARCHAR(30),

middle\_name VARCHAR(30),

hourly\_pay DECIMAL(5, 2),

hire\_date DATE,

CONSTRAINT chk\_hour CHECK(hourly\_pay >= 10.00)

);

ALTER TABLE dynamics

ADD CONSTRAINT chk\_hire\_date CHECK(hire\_date >= '2022-02-02')

CREATE TABLE products (

product\_id INT,

product\_name VARCHAR(25),

price DECIMAL(4, 2) DEFAULT 0

);

INSERT INTO products (product\_id, product\_name)

VALUES (104, 'straw'),

(105, 'napkin'),

(106, 'fork'),

(107, 'spoon');

ALTER TABLE products

ALTER COLUMN price SET DEFAULT 1;

CREATE TABLE transactions (

transaction\_id INT PRIMARY KEY,

amount DECIMAL(5, 2),

);

CREATE TABLE transactions (

transaction\_id INT PRIMARY KEY AUTO\_INCREMENT,

amount DECIMAL(5, 2)

);

INSERT INTO transactions (amount)

VALUES (4.99)

ALTER TABLE dynamics

ADD CONSTRAINT fk\_employee\_id

FOREIGN KEY(employee\_id) REFERENCES employees(employee\_id);

SELECT \*

FROM customer LEFT JOIN orders

ON customer.customer\_id = orders.customer\_id

SELECT \*

FROM customer RIGHT JOIN orders

ON customer.customer\_id = orders.customer\_id

SELECT \*

FROM customer inner JOIN orders

ON customer.customer\_id = orders.customer\_id

SELECT COUNT(customer\_name) as NAME

FROM customer

SELECT MAX(customer\_id) as HIGHEST

FROM customer

SELECT MIN(customer\_id) as LOWEST

FROM customer

SELECT AVG(customer\_id) as AVERAGE

FROM customer

SELECT SUM(customer\_id) as SUM

FROM customer

SELECT CONCAT(first\_name, ' ' ,last\_name) AS full\_name

from employees;

SELECT \*

FROM employees

WHERE hire\_date < '2023-01-05' and job = 'cook';

SELECT \*

FROM employees

WHERE job = 'cook' OR job = 'asst.manager';

SELECT \*

FROM employees

WHERE NOT job = 'manager';

SELECT \*

FROM employees

WHERE NOT job = 'manager' AND not job = 'asst.manager';

SELECT \*

FROM employees

WHERE hire\_date BETWEEN '2023-01-02' AND '2023-01-06';

SELECT \*

FROM employees

WHERE job IN ('cook', 'cashier', 'janitor');

SELECT \*

FROM employees

WHERE first\_name Like 's%'

SELECT \*

FROM employees

WHERE hire\_date LIKE '2023%';

SELECT \* from employees

WHERE last\_name LIKE '%r';

SELECT \* FROM employees

WHERE job LIKE '\_ook';

SELECT \* FROM employees

WHERE hire\_date LIKE '\_\_\_\_-\_\_-04';

SELECT \* FROM employees

WHERE job LIKE '\_a%';

select \* from employees

ORDER BY last\_name;

select \* from employees

ORDER BY last\_name DESC;

select \* from employees

ORDER BY last\_name DESC, hire\_date ASC;

SELECT \* FROM INCOME

UNION

SELECT \* FROM expenses;

SELECT \*

FROM customer AS a

INNER JOIN customer AS b

on a.referral\_id = b.customer\_id;

SELECT a.customer\_id, a.customer\_name,

b.customer\_id, b.customer\_name

FROM customer AS a

INNER JOIN customer AS b

on a.referral\_id = b.customer\_id;

SELECT a.customer\_id, a.customer\_name,

CONCAT(a.customer\_name, ' ', b.customer\_name) AS 'REFERRED BY'

FROM customer AS a

INNER JOIN customer AS b

on a.referral\_id = b.customer\_id;

SELECT a.customer\_id, a.customer\_name,

CONCAT(a.customer\_name, ' ', b.customer\_name) AS 'REFERRED BY'

FROM customer AS a

LEFT JOIN customer AS b

on a.referral\_id = b.customer\_id;

SELECT a.customer\_id, a.customer\_name,

CONCAT(a.customer\_name, ' ', b.customer\_name) AS 'REFERRED BY'

FROM customer AS a

RIGHT JOIN customer AS b

on a.referral\_id = b.customer\_id;

SELECT a.first\_name, a.last\_name,

CONCAT(b.first\_name, ' ', b.last\_name) AS 'reports to'

FROM employees AS a

INNER JOIN employees AS b

ON a.supervisor\_id = b.employee\_id;

SELECT a.first\_name, a.last\_name,

CONCAT(b.first\_name, ' ', b.last\_name) AS 'reports to'

FROM employees AS a

RIGHT JOIN employees AS b

ON a.supervisor\_id = b.employee\_id;

CREATE VIEW employee\_attendance AS

SELECT first\_name, last\_name

FROM employees;

SELECT \* FROM employee\_attendance;

SELECT \* FROM employee\_attendance

ORDER BY last\_name DESC;

CREATE VIEW customer\_emails AS

SELECT email

FROM customer;

Select \*

FROM customer\_emails;

----INDEX

CREATE INDEX customer\_name\_idx

ON customer(customer\_name);

SELECT \* FROM customer

WHERE customer\_name = 'Emmanuel';

CREATE INDEX customer\_name\_city\_idx

on customer(customer\_name, city);

ALTER TABLE customer

DROP INDEX customer\_name\_idx;

SELECT \* FROM customer

where customer\_name = 'Emmanuel' AND city = 'oregun';

----SUBQUERY

SELECT first\_name, last\_name, hourly\_pay,

(SELECT AVG(hourly\_pay) from employees) AS avg\_pay

FROM employees;

SELECT first\_name, last\_name, hourly\_pay

FROM employees

WHERE hourly\_pay > (SELECT AVG(hourly\_pay) FROM employees);

SELECT customer\_name, city

FROM customer

WHERE customer\_id IN

(SELECT DISTINCT customer\_id

FROM customer

WHERE customer\_id IS NOT NULL);

SELECT customer\_name, city

FROM customer

WHERE customer\_id NOT IN

(SELECT DISTINCT customer\_id

FROM customer

WHERE customer\_id IS NOT NULL);

----GROUP BY, HAVING CLAUSE

SELECT sum(amount) as total\_per\_day, order\_date

from orders

GROUP BY order\_date;

SELECT MAX(amount) as total\_per\_day, order\_date

from orders

GROUP BY order\_date;

SELECT MIN(amount) as MIN\_per\_day, order\_date

from orders

GROUP BY order\_date;

SELECT avg(amount) as avg\_per\_day, order\_date

from orders

GROUP BY order\_date;

SELECT count(amount) as count\_per\_day, order\_date

from orders

GROUP BY order\_date;

SELECT count(amount) as count\_per\_day, order\_id

FROM orders

GROUP BY order\_id

HAVING COUNT(amount) < 1 AND order\_id IS NOT NULL;

----ROLL UP with aggregate

SELECT SUM(amount), order\_date

FROM orders

GROUP BY order\_date WITH ROLLUP;

SELECT SUM(order\_id), order\_date

FROM orders

GROUP BY order\_date WITH ROLLUP;

SELECT count(order\_id) as '# of orders', customer\_id

FROM orders

GROUP BY customer\_id with rollup;

SELECT SUM(hourly\_pay) AS 'hourly pay', employee\_id

FROM employees

GROUP BY employee\_id with rollup;

----on delete set null, on delete cascade

ALTER TABLE customer

ADD CONSTRAINT fk\_customer\_id

FOREIGN KEY(customer\_id) REFERENCES customer(customer\_id)

ON DELETE SET NULL;

ALTER TABLE customer

ADD CONSTRAINT fk\_customer\_id

FOREIGN KEY(customer\_id) REFERENCES customer(customer\_id)

ON DELETE cascade;

-----STORED PROCEDURE IN MY SQL

DELIMITER $$

CREATE PROCEDURE get\_customer()

BEGIN

SELECT \* from customer;

END $$

DELIMITER;

DELIMITER $$

CREATE PROCEDURE find\_customer(IN id INT)

BEGIN

SELECT \*

FROM customer

WHERE customer\_id = id;

END $$

DELIMITER;

CALL find\_customer();

DELIMITER $$

CREATE PROCEDURE find\_customer(IN f\_name VARCHAR(50),

IN l\_name VARCHAR(50))

BEGIN

SELECT \*

FROM customer

WHERE first\_name = f\_name AND last\_name = l\_name;

END $$

DELIMITER;

CALL find\_customer('larry', 'lobster');

----TRIGGER ON MYSQL when u insert, update or delete from a table

CREATE TRIGGER before\_hourly\_pay\_update

BEFORE UPDATE ON employees

FOR EACH ROW

SET NEW.salary = (NEW.hourly\_pay \* 2080);

CREATE TRIGGER before\_hourly\_pay\_insert

BEFORE INSERT ON employees

FOR EACH ROW

SET NEW.salary = (NEW.hourly\_pay \* 2080);

CREATE TRIGGER after\_salary\_delete

AFTER DELETE ON employees

FOR EACH ROW

UPDATE expenses\_number

SET expense\_total = expense\_total - OLD.salary

WHERE expense\_name = 'salaries';

DELETE FROM employees

WHERE employee\_id = 6;

CREATE TRIGGER after\_salary\_insert

AFTER INSERT ON employees

FOR EACH ROW

UPDATE expenses\_number

SET expense\_total = expense\_total + NEW.salary

WHERE expense\_name = 'salaries';

INSERT INTO employees

VALUES(6, 'SHEDOM', 'PLANKTON', 10, NULL, 'JANITOR', '2023-01-01');

CREATE TRIGGER after\_salary\_update

AFTER UPDATE ON employees

FOR EACH ROW

UPDATE expenses\_number

SET expense\_total = expense\_total + (NEW.salary - OLD.salary)

WHERE expense\_name = 'salaries';