[Como usar Node.js + MySQL – LuizTools](https://www.luiztools.com.br/post/como-usar-nodejs-mysql/)

-- Criando uma tabela de exemplo

CREATE TABLE Employees (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(50),

salary DECIMAL(10, 2)

);

-- Inserindo dados na tabela

INSERT INTO Employees (name, salary) VALUES ('João', 3000.00);

INSERT INTO Employees (name, salary) VALUES ('Maria', 3500.00);

INSERT INTO Employees (name, salary) VALUES ('Pedro', 3200.00);

-- Criando uma View para obter os funcionários com salário acima de 3200

CREATE VIEW HighSalaryEmployees AS

SELECT \* FROM Employees WHERE salary > 3200;

-- Criando uma Stored Procedure para aumentar o salário em 10%

DELIMITER $$

CREATE PROCEDURE IncreaseSalary()

BEGIN

UPDATE Employees SET salary = salary \* 1.10;

END$$

DELIMITER ;

-- Criando uma Function para calcular o salário líquido

DELIMITER $$

CREATE FUNCTION CalculateNetSalary(grossSalary DECIMAL(10, 2)) RETURNS DECIMAL(10, 2)

BEGIN

DECLARE tax DECIMAL(5, 2);

SET tax = 0.15; -- Supondo uma taxa de imposto de 15%

RETURN grossSalary \* (1 - tax);

END$$

DELIMITER ;

-- Criando um Trigger para registrar alterações de salário

CREATE TRIGGER SalaryChangeTrigger

AFTER UPDATE ON Employees

FOR EACH ROW

BEGIN

INSERT INTO SalaryChanges (employee\_id, old\_salary, new\_salary, change\_date)

VALUES (OLD.id, OLD.salary, NEW.salary, NOW());

END;

-- Exemplo de uso da Stored Procedure

CALL IncreaseSalary();

-- Exemplo de uso da Function

SELECT name, salary, CalculateNetSalary(salary) AS net\_salary FROM Employees;

-- Exemplo de uso da View

SELECT \* FROM HighSalaryEmployees;

-- Após criar a tabela SalaryChanges, podemos visualizar as mudanças de salário registradas pelo Trigger

-- Criando uma tabela de histórico de mudanças de salário

CREATE TABLE SalaryChanges (

id INT AUTO\_INCREMENT PRIMARY KEY,

employee\_id INT,

old\_salary DECIMAL(10, 2),

new\_salary DECIMAL(10, 2),

change\_date TIMESTAMP

);

-- Criando uma Stored Procedure para aumentar o salário em 10%, mas apenas para funcionários com salário inferior a 3500

DELIMITER $$

CREATE PROCEDURE IncreaseSalary()

BEGIN

DECLARE done INT DEFAULT 0;

DECLARE emp\_id INT;

DECLARE emp\_salary DECIMAL(10, 2);

DECLARE cur CURSOR FOR SELECT id, salary FROM Employees WHERE salary < 3500;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO emp\_id, emp\_salary;

IF done THEN

LEAVE read\_loop;

END IF;

-- Aumenta o salário apenas se o funcionário tiver um salário inferior a 3500

UPDATE Employees SET salary = salary \* 1.10 WHERE id = emp\_id;

-- Registra a mudança de salário

INSERT INTO SalaryChanges (employee\_id, old\_salary, new\_salary, change\_date)

VALUES (emp\_id, emp\_salary, emp\_salary \* 1.10, NOW());

END LOOP;

CLOSE cur;

END$$

DELIMITER ;

-- Exemplo de uso de um loop com IF na Stored Procedure

CALL IncreaseSalary();

-- Exemplo de uso de IF em uma Function para calcular o salário líquido com base no cargo

DELIMITER $$

CREATE FUNCTION CalculateNetSalary(grossSalary DECIMAL(10, 2), position VARCHAR(50)) RETURNS DECIMAL(10, 2)

BEGIN

DECLARE tax DECIMAL(5, 2);

SET tax = IF(position = 'Manager', 0.20, 0.15); -- Se o cargo for Gerente, a taxa de imposto é 20%, senão é 15%

RETURN grossSalary \* (1 - tax);

END$$

DELIMITER ;

-- Exemplo de uso da Function com IF

SELECT name, salary, CalculateNetSalary(salary, 'Manager') AS net\_salary FROM Employees;

-- Exemplo de uso de um loop para exibir os funcionários

DELIMITER $$

CREATE PROCEDURE DisplayEmployees()

BEGIN

DECLARE done INT DEFAULT 0;

DECLARE emp\_name VARCHAR(50);

DECLARE cur CURSOR FOR SELECT name FROM Employees;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO emp\_name;

IF done THEN

LEAVE read\_loop;

END IF;

SELECT emp\_name;

END LOOP;

CLOSE cur;

END$$

DELIMITER ;

-- Exemplo de uso do loop para exibir os funcionários

CALL DisplayEmployees();

DELIMITER $$

CREATE PROCEDURE registarUsuario(Varchar(1000))

BEGIN

UPDATE Employees SET salary = salary \* 1.10;

END$$

DELIMITER ;