Task1

SELECT \* FROM employee;

Task2

SELECT name,salary,

FROM employees

Task3

SELECT \* FROM employees

WHERE salary > 50000;

Task4

SELECT \* FROM employees

WHERE name LIKE 'J%';

TASK 5

SELECT \* FROM employees

WHERE department\_id IN (1, 2, 3);

TASK 6

SELECT \* FROM employees

ORDER BY hire\_date DESC;

TASK 7

SELECT COUNT(\*) AS employee\_count

FROM employees;

TASK8

SELECT SUM(salary) AS total\_salary

FROM employees;

TASK 9

SELECT AVG(salary) AS average\_salary

FROM employees;

TASK10

SELECT MAX(salary) AS max\_salary

FROM employees;

SELECT MIN(salary) AS min\_salary

FROM employees;

TASK11

SELECT \*

FROM employees

WHERE salary > 50000;

SELECT department\_id, AVG(salary) AS avg\_salary

FROM employees

GROUP BY department\_id

HAVING AVG(salary) > 50000;

TASK 12

CREATE TABLE employees (

employee\_id INT PRIMARY KEY,

name VARCHAR(100),

department\_id INT,

salary DECIMAL(10, 2),

hire\_date DATE,

FOREIGN KEY (department\_id) REFERENCES departments(department\_id)

);

TASK 13

CREATE TABLE departments (

department\_id INT PRIMARY KEY,

department\_name VARCHAR(100)

);

CREATE TABLE employees (

employee\_id INT PRIMARY KEY,

name VARCHAR(100),

department\_id INT,

salary DECIMAL(10, 2),

hire\_date DATE,

CONSTRAINT fk\_department\_id

FOREIGN KEY (department\_id)

REFERENCES departments(department\_id)

);

TASK14

CREATE TABLE students (

student\_id INT AUTO\_INCREMENT PRIMARY KEY,

student\_name VARCHAR(100) NOT NULL,

student\_email VARCHAR(100)

);

CREATE TABLE courses (

course\_id INT AUTO\_INCREMENT PRIMARY KEY,

course\_name VARCHAR(100) NOT NULL,

course\_description TEXT

); CREATE TABLE student\_courses (

student\_id INT,

course\_id INT,

PRIMARY KEY (student\_id, course\_id),

FOREIGN KEY (student\_id) REFERENCES students(student\_id),

FOREIGN KEY (course\_id) REFERENCES courses(course\_id)

);

TASK15

CREATE TABLE departments (

department\_id INT AUTO\_INCREMENT PRIMARY KEY,

department\_name VARCHAR(100) NOT NULL

);

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

employee\_name VARCHAR(100) NOT NULL,

department\_id INT, -- This will be the foreign key column

FOREIGN KEY (department\_id) REFERENCES departments(department\_id)

);

ALTER TABLE employees

ADD CONSTRAINT fk\_department\_id

FOREIGN KEY (department\_id) REFERENCES departments(department\_id);

TASK16

CREATE TABLE departments (

department\_id INT AUTO\_INCREMENT PRIMARY KEY,

department\_name VARCHAR(100) NOT NULL

);

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

employee\_name VARCHAR(100) NOT NULL,

department\_id INT, -- This will be the foreign key column

FOREIGN KEY (department\_id) REFERENCES departments(department\_id)

);

ALTER TABLE employees

ADD CONSTRAINT fk\_department\_id

FOREIGN KEY (department\_id) REFERENCES departments(department\_id);

TASK17

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

employee\_name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE, -- Adding unique constraint to email column

department\_id INT,

FOREIGN KEY (department\_id) REFERENCES departments(department\_id)

);

ALTER TABLE employees

ADD CONSTRAINT unique\_email

UNIQUE (email);

TASK18

ALTER TABLE employees

ADD CONSTRAINT chk\_salary\_gt\_zero CHECK (salary > 0);

TASK 19

ALTER TABLE employees

ALTER COLUMN status SET DEFAULT 'Active';

TASK 20

-- Departments table

CREATE TABLE departments (

department\_id INT AUTO\_INCREMENT PRIMARY KEY,

department\_name VARCHAR(100) NOT NULL,

location VARCHAR(100)

);

-- Employees table

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

employee\_name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE,

department\_id INT,

hire\_date DATE,

status VARCHAR(50) DEFAULT 'Active',

FOREIGN KEY (department\_id) REFERENCES departments(department\_id)

);

-- Projects table

CREATE TABLE projects (

project\_id INT AUTO\_INCREMENT PRIMARY KEY,

project\_name VARCHAR(100) NOT NULL,

start\_date DATE,

end\_date DATE,

budget DECIMAL(10, 2)

);

-- Employee\_Projects table (for many-to-many relationship)

CREATE TABLE employee\_projects (

employee\_id INT,

project\_id INT,

PRIMARY KEY (employee\_id, project\_id),

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

FOREIGN KEY (project\_id) REFERENCES projects(project\_id)

);