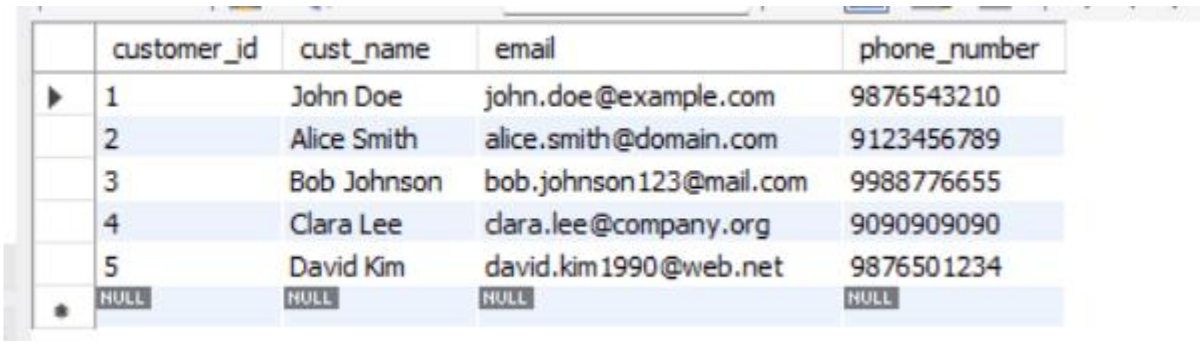
MYSQL_PRACTICE_SCENARIOS

1) You are working on an e-commerce database. The "customers" table needs a column that uniquely identifies each customer. The table should also store customer name, email, and phone number. Ensure that no two customers have the same identifier and the primary key should not allow null values. How would you design this table and enforce the uniqueness of the customer ID?

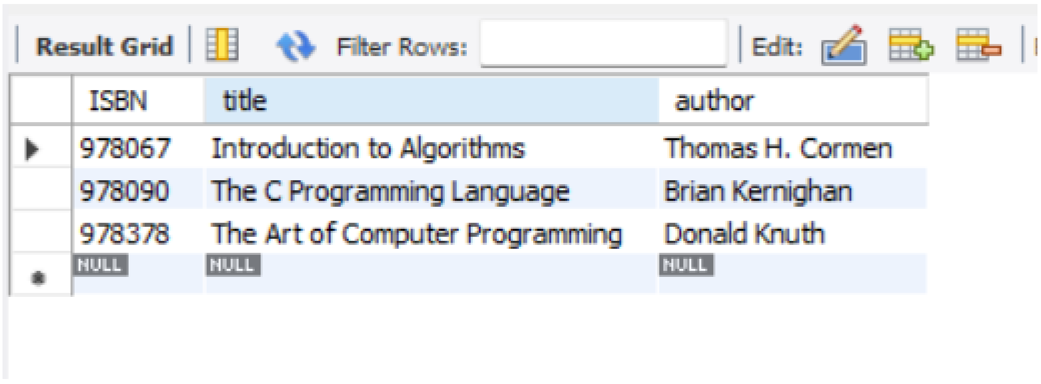
```
CREATE DATABASE e_commerce;

USE e_commerce;

CREATE table customers(

 customer_id int PRIMARY KEY AUTO_INCREMENT,

 cust_name VARCHAR(50) NOT NULL,

 email VARCHAR(50) NOT NULL,

 phone_number VARCHAR(15) NOT NULL UNIQUE CHECK(length(phone_number)>=10)

)

INSERT INTO customers (cust_name, email, phone_number) VALUES

('John Doe', 'john.doe@example.com', '9876543210'),

('Alice Smith', 'alice.smith@domain.com', '9123456789'),

('Bob Johnson', 'bob.johnson123@mail.com', '9988776655'),

('Clara Lee', 'clara.lee@company.org', '9090909090'),

('David Kim', 'david.kim1990@web.net', '9876501234');


select * from customers;
```



customer_id	cust_name	email	phone_number
1	John Doe	john.doe@example.com	9876543210
2	Alice Smith	alice.smith@domain.com	9123456789
3	Bob Johnson	bob.johnson123@mail.com	9988776655
4	Clara Lee	clara.lee@company.org	9090909090
5	David Kim	david.kim1990@web.net	9876501234
NULL	NULL	NULL	NULL

2) In a library management system, you have a "books" table that stores book titles, ISBN numbers, and authors. ISBN numbers must be unique because each book has a distinct ISBN. You are tasked with ensuring that no two books in the system share the same ISBN number, while allowing multiple books with the same title or same author. How would you implement this constraint?

```
CREATE DATABASE library_management_db;

USE library_management_db;

CREATE table books(
ISBN BIGINT PRIMARY KEY,
title VARCHAR(80) NOT NULL,
author VARCHAR(50) NOT NULL
)

INSERT INTO books (ISBN, title, author) VALUES
(978378, 'The Art of Computer Programming', 'Donald Knuth'),
(978090, 'The C Programming Language', 'Brian Kernighan'),
(978067, 'Introduction to Algorithms', 'Thomas H. Cormen');

select * from books;
```



ISBN	title	author
978067	Introduction to Algorithms	Thomas H. Cormen
978090	The C Programming Language	Brian Kernighan
978378	The Art of Computer Programming	Donald Knuth
NULL	NULL	NULL

3) You are designing a payroll system. The "employees" table has columns for employee ID, name, date of birth, salary, and date of hire. The employee ID should be a string that supports both numbers and letters (e.g., "EMP123"), the salary should be a decimal to store precise values with two decimal places, and the date of birth and date of hire should be stored in a date format. What would be the most appropriate data types for these columns and why?

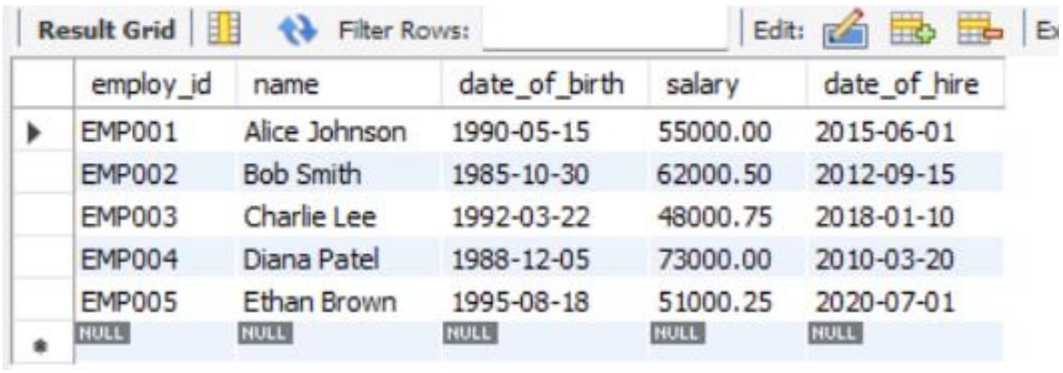
```
CREATE DATABASE payroll_db;

USE payroll_db;

CREATE table employees(
employ_id VARCHAR(10) PRIMARY KEY,
name VARCHAR(50) NOT NULL,
date_of_birth date NOT NULL,
salary decimal(10,2) NOT NULL,
date_of_hire date NOT NULL
)

INSERT INTO employees (employ_id, name, date_of_birth, salary, date_of_hire) VALUES
('EMP001', 'Alice Johnson', '1990-05-15', 55000.00, '2015-06-01'),
('EMP002', 'Bob Smith', '1985-10-30', 62000.50, '2012-09-15'),
```

('EMP003', 'Charlie Lee', '1992-03-22', 48000.75, '2018-01-10'),

('EMP004', 'Diana Patel', '1988-12-05', 73000.00, '2010-03-20'),

('EMP005', 'Ethan Brown', '1995-08-18', 51000.25, '2020-07-01');

select * from employees;



employ_id	name	date_of_birth	salary	date_of_hire
EMP001	Alice Johnson	1990-05-15	55000.00	2015-06-01
EMP002	Bob Smith	1985-10-30	62000.50	2012-09-15
EMP003	Charlie Lee	1992-03-22	48000.75	2018-01-10
EMP004	Diana Patel	1988-12-05	73000.00	2010-03-20
EMP005	Ethan Brown	1995-08-18	51000.25	2020-07-01
NULL	NULL	NULL	NULL	NULL

4) You are building a database for an online course platform. You have two tables: one for "students" (containing student ID, name, and email) and another for "courses" (containing course ID, name, and instructor). You need to generate a report that lists all the students enrolled in each course, with the student's name, course name, and instructor's name. How would you retrieve this information using a SQL JOIN?

CREATE DATABASE online_course_db;

create table student(

id int PRIMARY KEY AUTO_INCREMENT,

name varchar(50) NOT NULL,

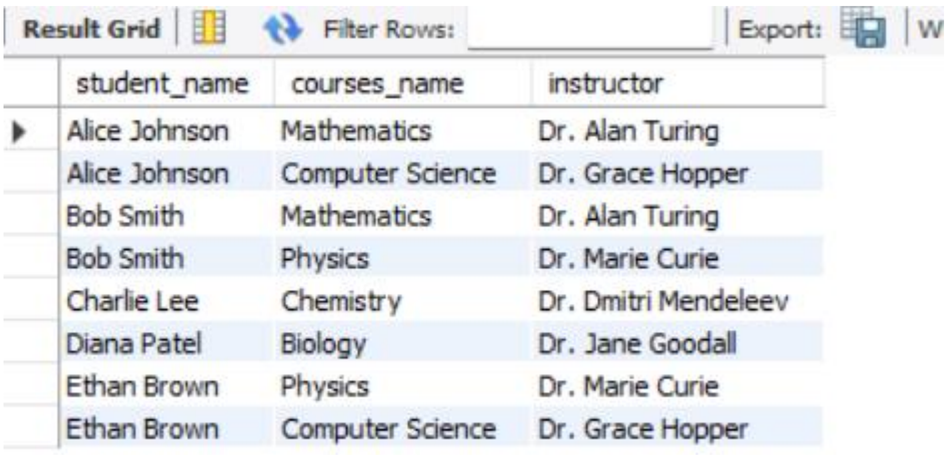
emil varchar(60) NOT NULL

)

ALTER TABLE student

RENAME COLUMN name TO student_name;

```
INSERT INTO student (student_name, emil) VALUES
('Alice Johnson', 'alice.johnson@example.com'),
('Bob Smith', 'bob.smith@example.com'),
('Charlie Lee', 'charlie.lee@example.com'),
('Diana Patel', 'diana.patel@example.com'),
('Ethan Brown', 'ethan.brown@example.com');

create table courses(
id int PRIMARY KEY AUTO_INCREMENT,
name varchar(100) NOT NULL,
instructor varchar(50) NOT NULL
)

ALTER TABLE courses
RENAME COLUMN name TO courses_name;

INSERT INTO courses (courses_name, instructor) VALUES
('Mathematics', 'Dr. Alan Turing'),
('Physics', 'Dr. Marie Curie'),
('Chemistry', 'Dr. Dmitri Mendeleev'),
('Biology', 'Dr. Jane Goodall'),
('Computer Science', 'Dr. Grace Hopper');


create table students_courses_enrolled(
student_id int,
course_id int,
PRIMARY KEY (student_id,course_id),

FOREIGN KEY (student_id) REFERENCES student(id),

FOREIGN KEY (course_id) REFERENCES courses(id)

)


INSERT INTO students_courses_enrolled (student_id, course_id) VALUES

(1, 1),

(1, 5),

(2, 1),

(2, 2),

(3, 3),

(4, 4),

(5, 2),

(5, 5);


select student_name, courses_name,instructor

 from students_courses_enrolled

 JOIN student ON students_courses_enrolled.student_id=student.id

 JOIN courses ON students_courses_enrolled.course_id=courses.id
```

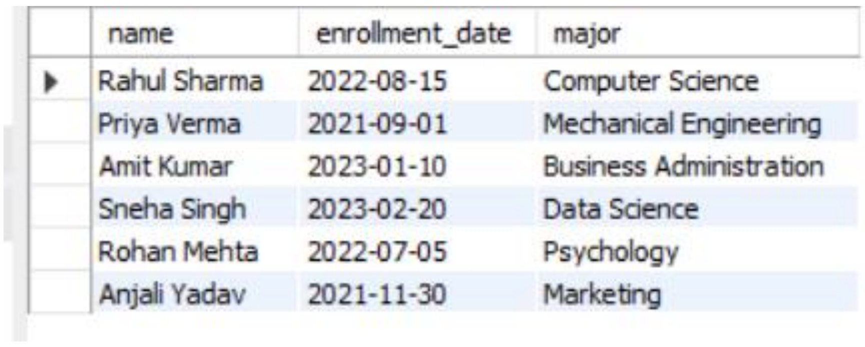


student_name	courses_name	instructor
Alice Johnson	Mathematics	Dr. Alan Turing
Alice Johnson	Computer Science	Dr. Grace Hopper
Bob Smith	Mathematics	Dr. Alan Turing
Bob Smith	Physics	Dr. Marie Curie
Charlie Lee	Chemistry	Dr. Dmitri Mendeleev
Diana Patel	Biology	Dr. Jane Goodall
Ethan Brown	Physics	Dr. Marie Curie
Ethan Brown	Computer Science	Dr. Grace Hopper

5) A university system has two separate tables: one for full-time students and one for part-time students. Both tables have the same structure (student ID, name, enrollment date, and major). The university wants a unified list of all students enrolled in the system, regardless of their status. How would you combine these two tables into a single result set that lists all the students?

```
CREATE DATABASE college_db;

use college_db;

CREATE table full_time_students (

 student_id INT PRIMARY KEY AUTO_INCREMENT,

  name VARCHAR(50) NOT NULL,

  enrollment_date DATE NOT NULL,

  major VARCHAR(50) NOT NULL

)

INSERT INTO full_time_students (name, enrollment_date, major) VALUES

('Rahul Sharma', '2022-08-15', 'Computer Science'),

('Priya Verma', '2021-09-01', 'Mechanical Engineering'),

('Amit Kumar', '2023-01-10', 'Business Administration');


CREATE TABLE part_time_students (

  student_id INT PRIMARY KEY AUTO_INCREMENT,

  name VARCHAR(50) NOT NULL,

  enrollment_date DATE NOT NULL,

  major VARCHAR(50) NOT NULL

);


INSERT INTO part_time_students (name, enrollment_date, major) VALUES

('Sneha Singh', '2023-02-20', 'Data Science'),

('Rohan Mehta', '2022-07-05', 'Psychology'),

('Anjali Yadav', '2021-11-30', 'Marketing');
```

```
SELECT name,enrollment_date,major from full_time_students
union all
SELECT name,enrollment_date,major from part_time_students
```

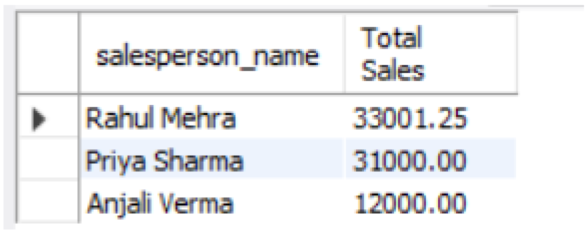


name	enrollment_date	major
Rahul Sharma	2022-08-15	Computer Science
Priya Verma	2021-09-01	Mechanical Engineering
Amit Kumar	2023-01-10	Business Administration
Sneha Singh	2023-02-20	Data Science
Rohan Mehta	2022-07-05	Psychology
Anjali Yadav	2021-11-30	Marketing

6) You are creating a report for a sales department. The report needs to show the total sales for each salesperson, but you want the column names to be more user-friendly. Instead of showing "SUM(sales_amount)", you want it to appear as "Total Sales". How would you use aliases to modify the column headings in your result set?

```
CREATE TABLE sales (
 sale_id INT PRIMARY KEY AUTO_INCREMENT,
 salesperson_name VARCHAR(50) NOT NULL,
 sales_amount DECIMAL(10, 2) NOT NULL,
 sale_date DATE
);
INSERT INTO sales (salesperson_name, sales_amount, sale_date) VALUES
('Rahul Mehra', 15000.50, '2024-01-15'),
('Priya Sharma', 22000.00, '2024-01-20'),
('Rahul Mehra', 18000.75, '2024-02-10'),
('Anjali Verma', 12000.00, '2024-03-01'),
('Priya Sharma', 9000.00, '2024-03-10');
```

```sql
SELECT salesperson_name, SUM(sales_amount) AS "Total Sales"

FROM sales

GROUP BY salesperson_name;
```

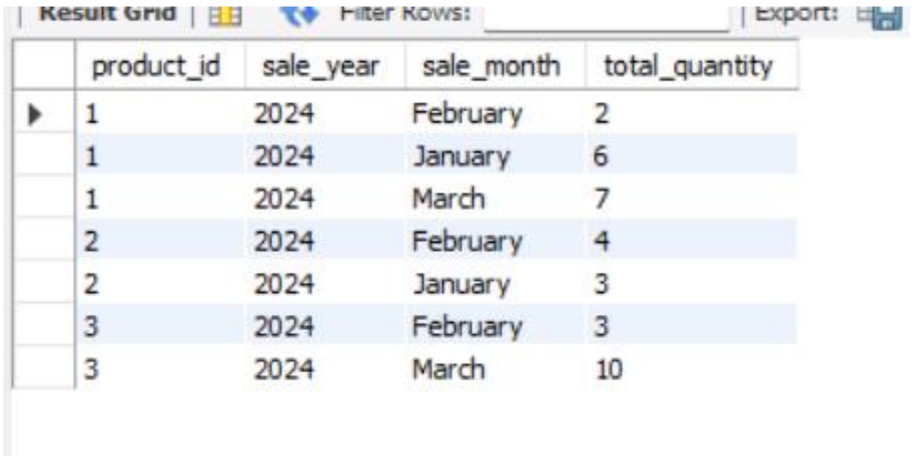


salesperson_name	Total Sales
Rahul Mehra	33001.25
Priya Sharma	31000.00
Anjali Verma	12000.00

7) In a database tracking the sales transactions of a retail store, you have a "sales" table containing transaction IDs, employee IDs, product IDs, quantities sold, and sale dates. You need to generate a report that shows the total quantity of each product sold per month, organized by product and month. How would you use the GROUP BY clause to aggregate this data?

```sql
CREATE TABLE sales2 (

 transaction_id INT PRIMARY KEY AUTO_INCREMENT,

 employee_id INT NOT NULL,

 product_id INT NOT NULL,

 quantity_sold INT NOT NULL,

 sale_date DATE NOT NULL

);


INSERT INTO sales2 (employee_id, product_id, quantity_sold, sale_date) VALUES

(101, 1, 5, '2024-01-10'),

(102, 2, 3, '2024-01-15'),

(101, 1, 2, '2024-02-12'),

(103, 2, 4, '2024-02-20'),

(104, 1, 7, '2024-03-05'),

(101, 3, 6, '2024-03-15'),

(102, 1, 1, '2024-01-28'),

(104, 3, 3, '2024-02-10'),

(103, 3, 4, '2024-03-20');
```

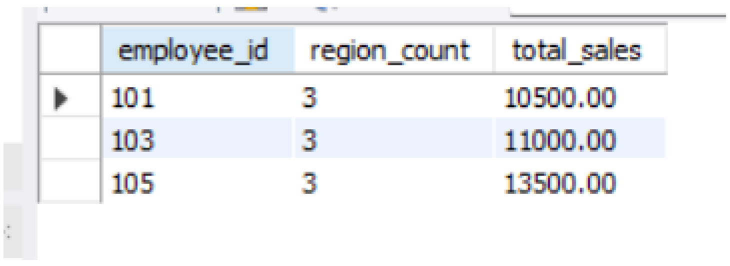
```
SELECT
  product_id,
  YEAR(sale_date) AS sale_year,
  monthname(sale_date) AS sale_month,
  SUM(quantity_sold) AS total_quantity
FROM sales2
GROUP BY product_id, monthname(sale_date),YEAR(sale_date)
ORDER BY product_id,sale_year,sale_month;
```



product_id	sale_year	sale_month	total_quantity
1	2024	February	2
1	2024	January	6
1	2024	March	7
2	2024	February	4
2	2024	January	3
3	2024	February	3
3	2024	March	10

8) You are analyzing employee performance in a company. You have a table called "employee_sales" which records sales figures for each employee. You want to generate a list of employees who have made more than $10,000 in sales, but only include those employees who have made sales in at least 3 different regions. How would you use the HAVING clause to filter the data after applying GROUP BY?

```
CREATE TABLE employee_sales (
  employee_id INT,
  region VARCHAR(50),
  sales_amount DECIMAL(10, 2)
);

INSERT INTO employee_sales (employee_id, region, sales_amount) VALUES
(101, 'North', 4000.00),
(101, 'East', 3500.00),
(101, 'South', 3000.00),

(102, 'North', 6000.00),
(102, 'East', 2000.00),

(103, 'West', 4500.00),
(103, 'South', 3500.00),
(103, 'East', 3000.00),

(104, 'North', 3000.00),
(104, 'West', 2500.00),
(104, 'East', 2000.00),
(104, 'South', 1500.00),

(105, 'North', 5000.00),
(105, 'East', 6000.00),
(105, 'South', 2500.00);
SELECT
  employee_id,
  COUNT(DISTINCT region) AS region_count,
  SUM(sales_amount) AS total_sales
FROM employee_sales
GROUP BY employee_id
HAVING total_sales > 10000 AND region_count >= 3;
```



employee_id	region_count	total_sales
101	3	10500.00
103	3	11000.00
105	3	13500.00