1)

CREATE TABLE students (

student\_id INTEGER PRIMARY KEY,

student\_name VARCHAR(255),

age INTEGER,

email VARCHAR(255),

frontend\_mark INTEGER,

backend\_mark INTEGER,

status VARCHAR(255)

);

INSERT INTO students

(student\_id, student\_name, age, email, frontend\_mark, backend\_mark, status)

VALUES

(1, 'Alice', 22, 'alice@example.com', 55, 57, NULL),

(2, 'Bob', 21, 'bob@example.com', 34, 45, NULL),

(3, 'Charlie', 23, 'charlie@example.com', 60, 59, NULL),

(4, 'David', 20, 'david@example.com', 40, 49, NULL),

(5, 'Eve', 24, 'newemail@example.com', 45, 34, NULL) ,

(6, 'Rahim', 23, 'rahim@gmail.com', 46, 42, NULL) ,

(7, 'Dhanish', 22, 'dhanishkumar@gmail.com', 40, 40, NULL);



CREATE TABLE courses (

course\_id INTEGER PRIMARY KEY,

course\_name VARCHAR(255),

credits INTEGER

);

INSERT INTO courses

(course\_id, course\_name, credits)

VALUES

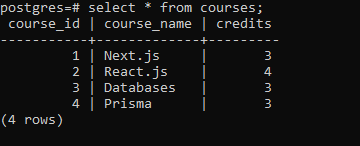
(1, 'Next.js', 3),

(2, 'React.js', 4),

(3, 'Databases', 3),

(4, 'Prisma', 3);

SELECT \* FROM COURSES;



CREATE TABLE enrollments (

enrollment\_id INTEGER PRIMARY KEY,

student\_id INTEGER,

course\_id INTEGER,

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

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

);

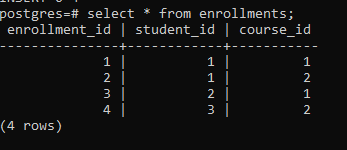
INSERT INTO enrollments (enrollment\_id, student\_id, course\_id) VALUES

(1, 1, 1),

(2, 1, 2),

(3, 2, 1),

(4, 3, 2);



2)

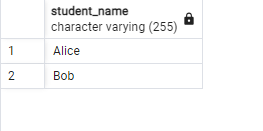
SELECT student\_name FROM students WHERE student\_id IN (

SELECT student\_id FROM enrollments WHERE course\_id = (

SELECT course\_id FROM courses WHERE course\_name = 'Next.js'

)

);

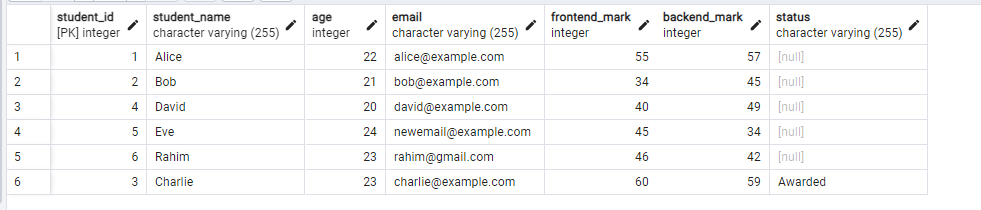


3)

UPDATE students SET status = 'Awarded' WHERE student\_id = (

SELECT student\_id FROM students ORDER BY (frontend\_mark + backend\_mark) DESC LIMIT 1

);

SELECT \* FROM students;  


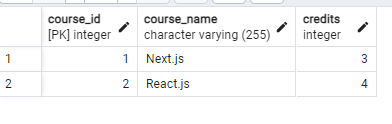
4)

DELETE FROM courses WHERE course\_id NOT IN (

SELECT DISTINCT course\_id FROM enrollments

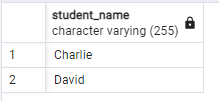
);

SELECT \* FROM courses;



5)

SELECT student\_name FROM students ORDER BY student\_id LIMIT 2 OFFSET 2;



6)

SELECT c.course\_name, COUNT(e.student\_id) AS student\_enrolled

FROM courses c

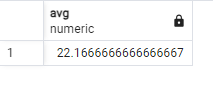
LEFT JOIN enrollments e ON c.course\_id = e.course\_id

GROUP BY c.course\_name;



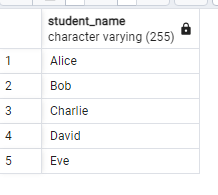
7)

SELECT AVG(age) FROM STUDENTS;



8)

SELECT student\_name FROM STUDENTS WHERE student\_email LIKE '%example.com';



1. Explain the primary key and foreign key concepts in PostgreSQL.

* Primary Key is used to uniquely identify the record in a table. It is immutable once set its value can’t be changed and its value can’t be null .
* Foreign key is used to establish relationship between 2 tables it is the primary key of one key is used in another table.

1. What is the difference between the VARCHAR and CHAR data types?

Char data type has a fixed length where as VARCHAR data type has a variable length we can specify the length in bracket

Column\_name VARCHAR(255);

1. Explain the purpose of the WHERE clause in a SELECT statement.

The where clause is used to filter the data that is displayed when using the select query.

1. What are the LIMIT and OFFSET clauses used for?

* Limit is used to limit the number of row when using select query
* Offset is used to change the starting point of the select statement

1. How can you perform data modification using UPDATE statements?

* Update table\_name set column\_name =value ;
* We can use it along with a where class too

1. What is the significance of the JOIN operation, and how does it work in PostgreSQL?

Combines the rows of a two table or more tables by using the related columns as key when joining them.

1. Explain the GROUP BY clause and its role in aggregation operations.

Groups rows have a common value like role as employee, manager etc.

Used with aggregate functions like COUNT, SUM, AVG, etc., to perform operations on groups of data.

1. How can you calculate aggregate functions like COUNT, SUM, and AVG in PostgreSQL?

SELECT COUNT(\*) FROM students;

Return the count of number of rows in student table

SELECT SUM(mark) FROM students;

Return the sum of mark in student table

SELECT AVG(mark) FROM students;

Return the average of mark in the students table

1. What is the purpose of an index in PostgreSQL, and how does it optimize query performance?

Improves query performance by reducing the number of data pages PostgreSQL needs to read. Speeds up data retrieval operations.

1. Explain the concept of a PostgreSQL view and how it differs from a table.

The view is used a virtual table based on the result of a select query. Stores query definition but not data itself.