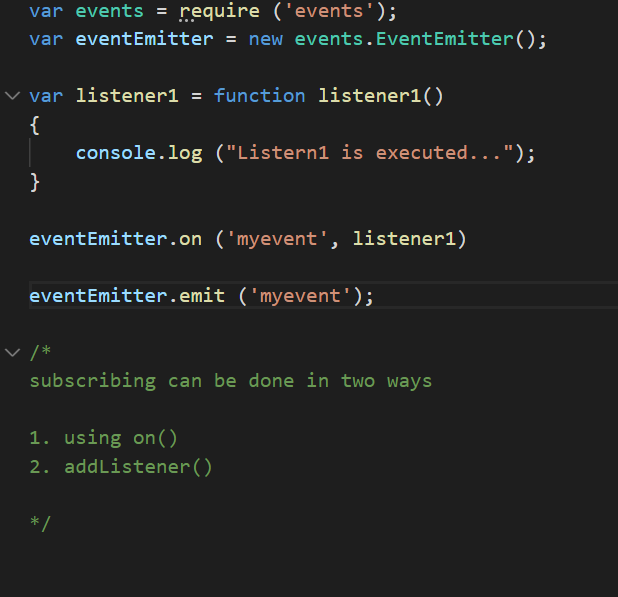
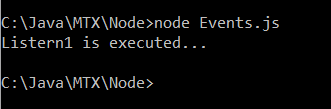
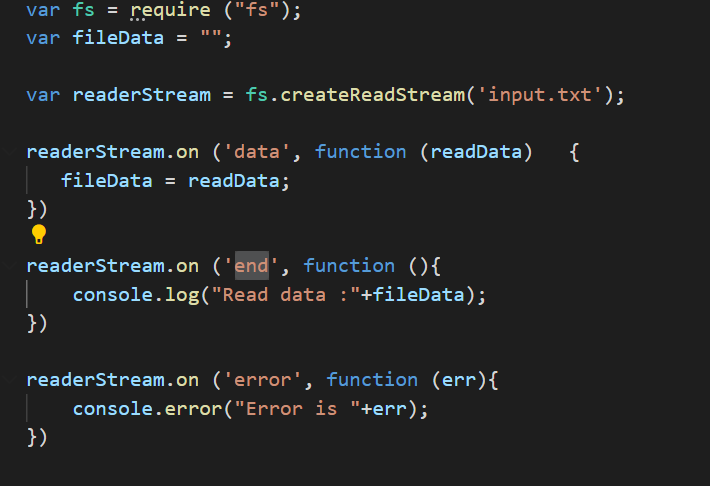
Node

Event Emitter

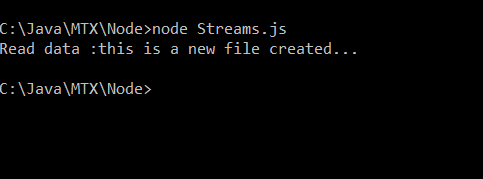




Reading the file through Streams.



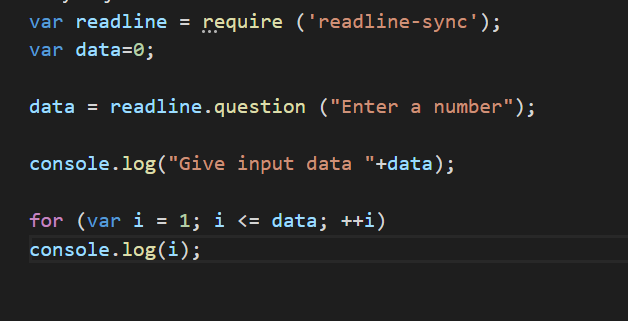
Output :



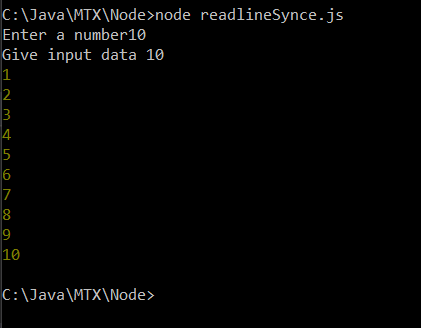
Reading input from readline-sync

First install readline-sync module

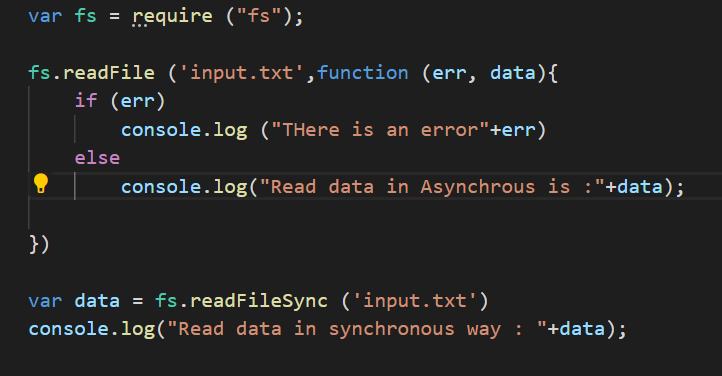
Npm install readline-sync



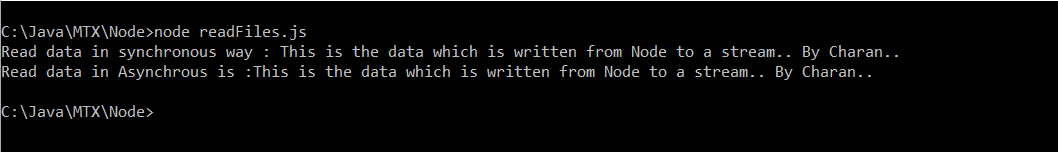
Output :



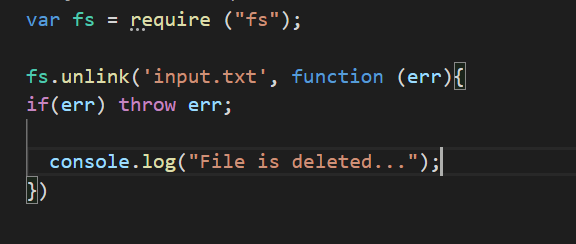
Reading the file both in synchronous and asynchronous way

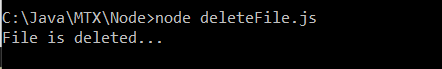


Output :

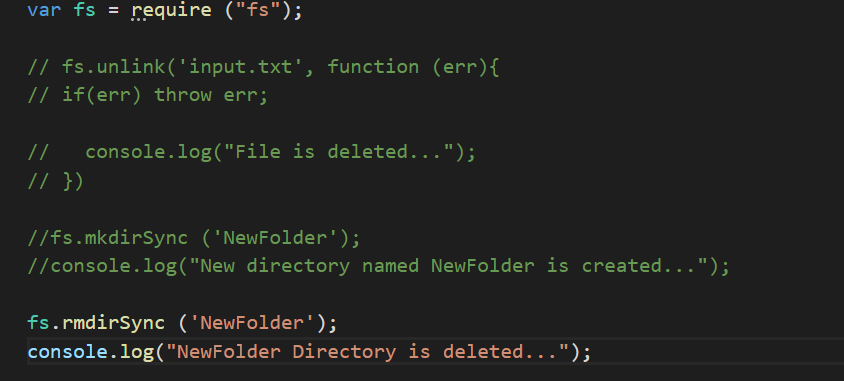


Deleting a file

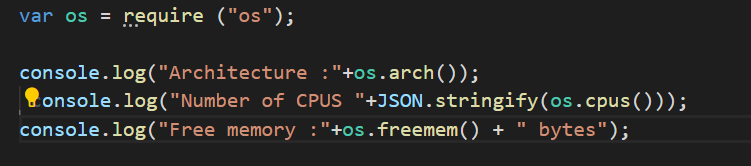




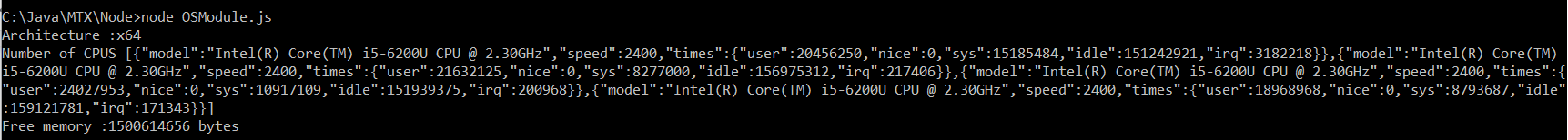
Creating and deleting a directory



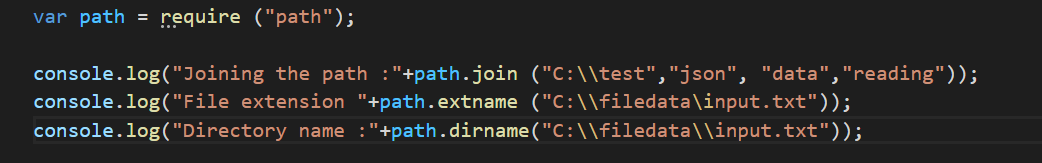
Node - Information on System and OS and CPU and Memory.



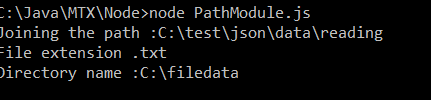
Output :



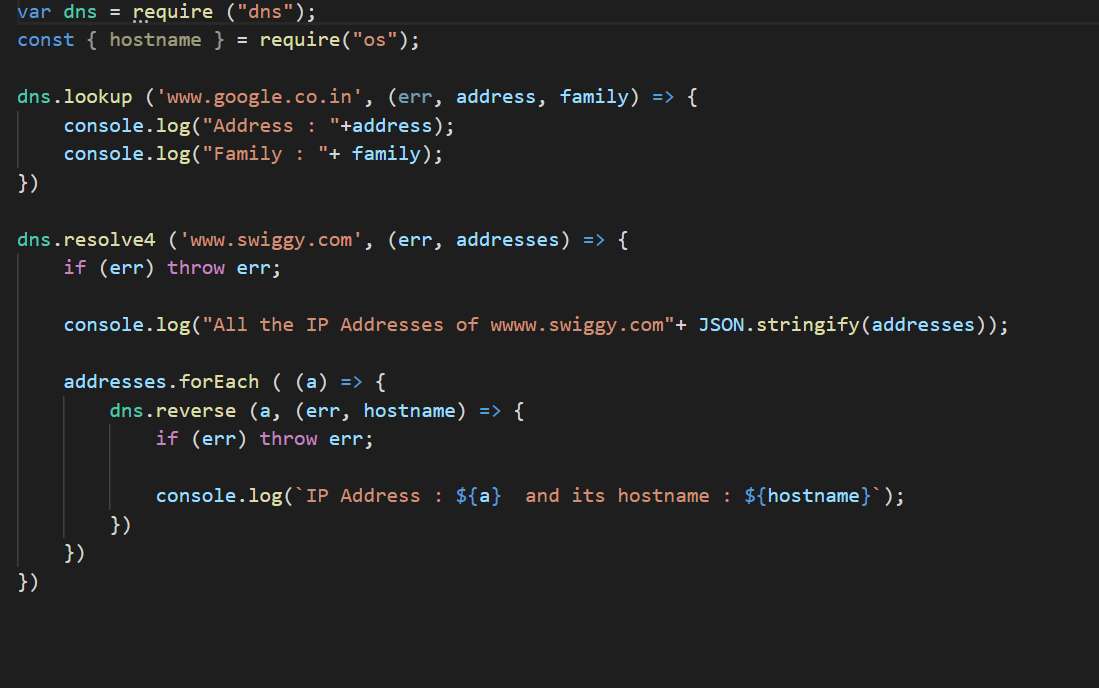
Path Module:



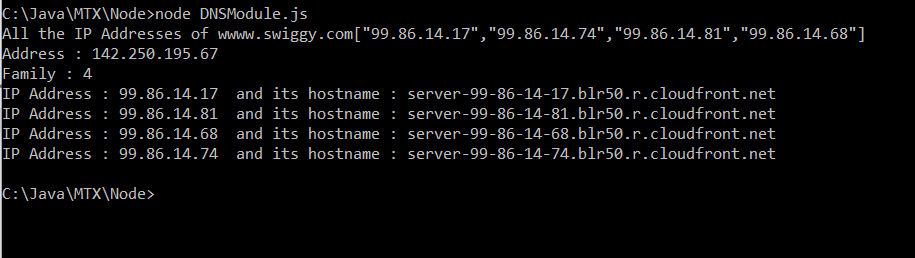
Output :



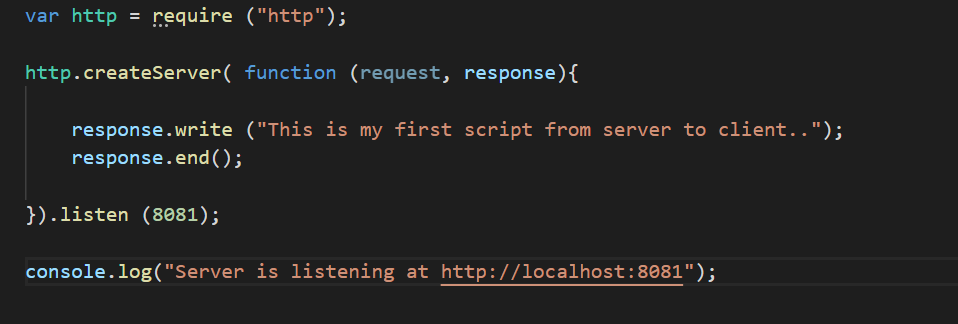
DNS Module



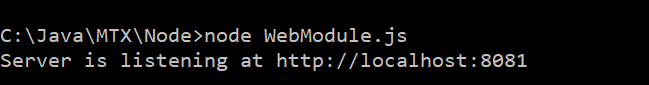
Output :



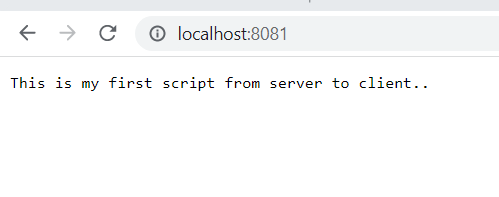
Web Module



Run the script:

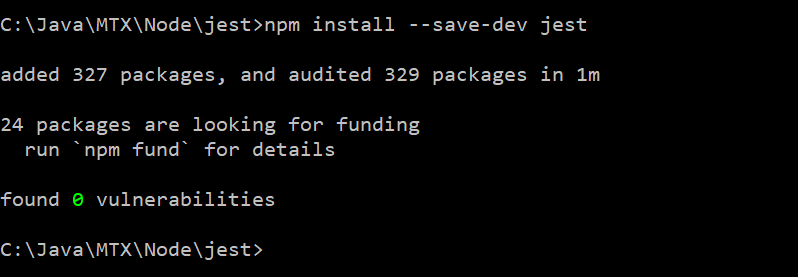


Output on client browser.

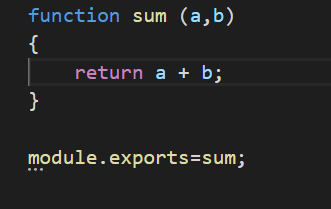


JEST

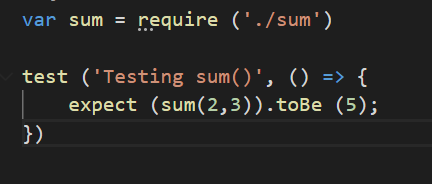
Install JEST



In Sum.js



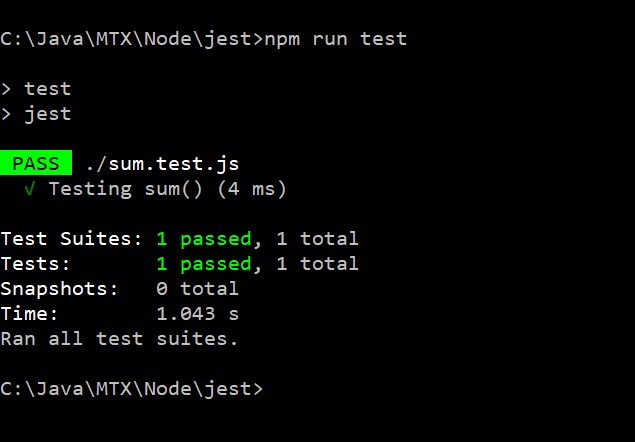
In sum.test.js



In package.json



Run the test

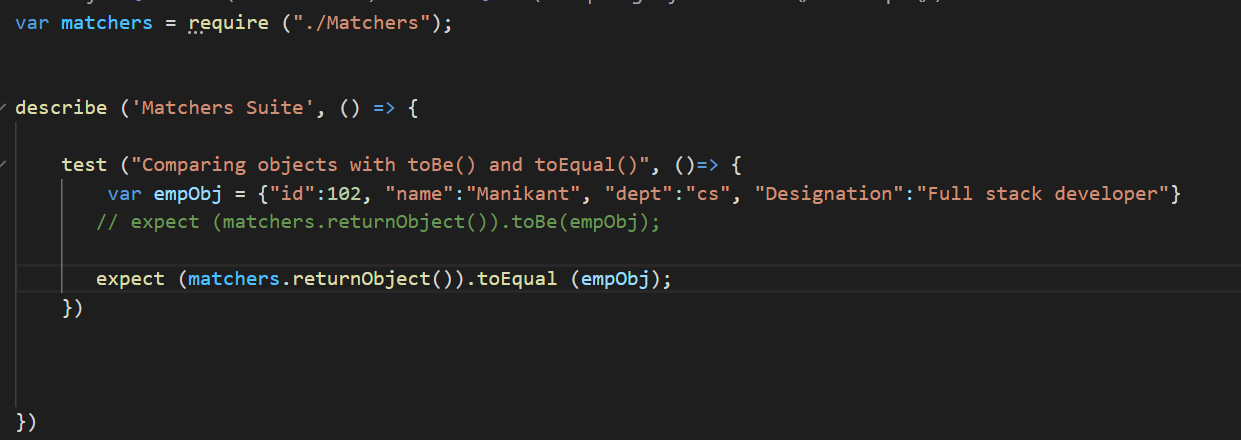


**Using matcher toEqual()**

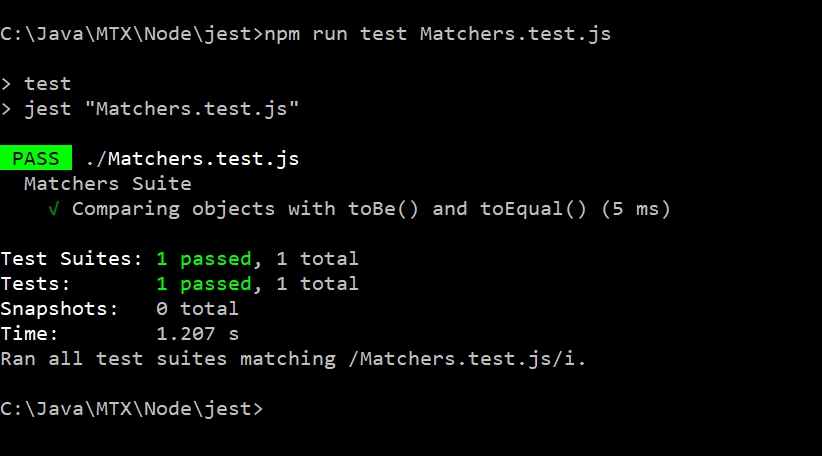
In Matchers.js



In Matchers.test.js



Run the test using npm run test <test file name>

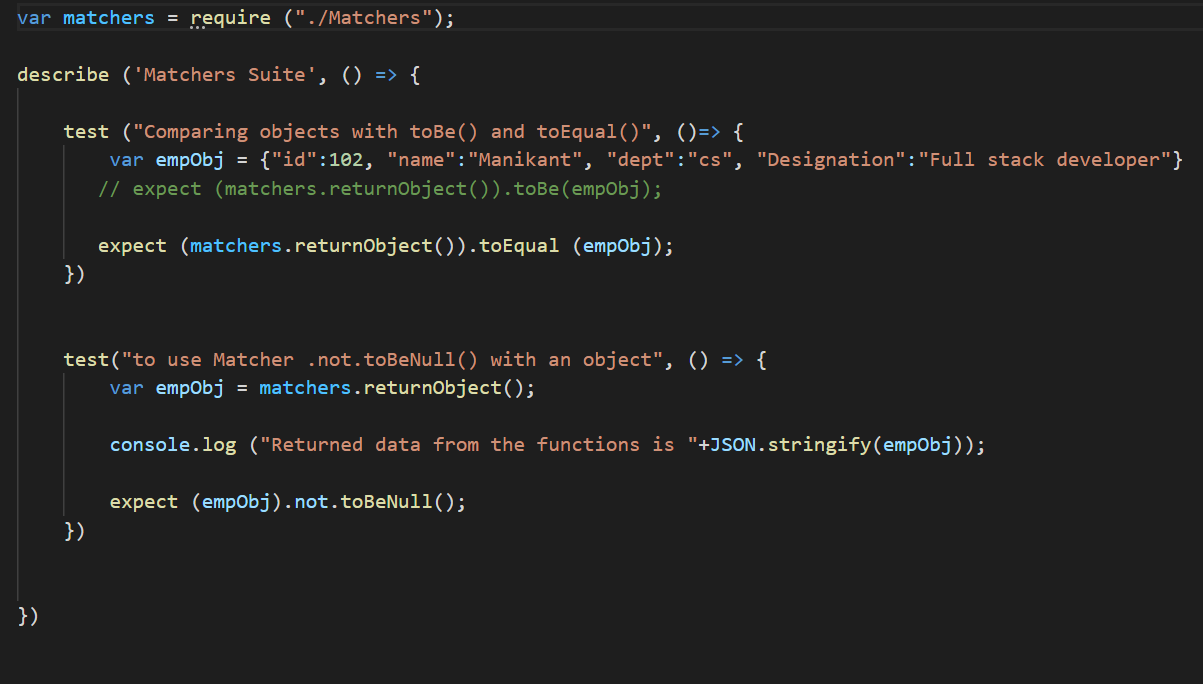


Using Matcher .not.toBeNull()

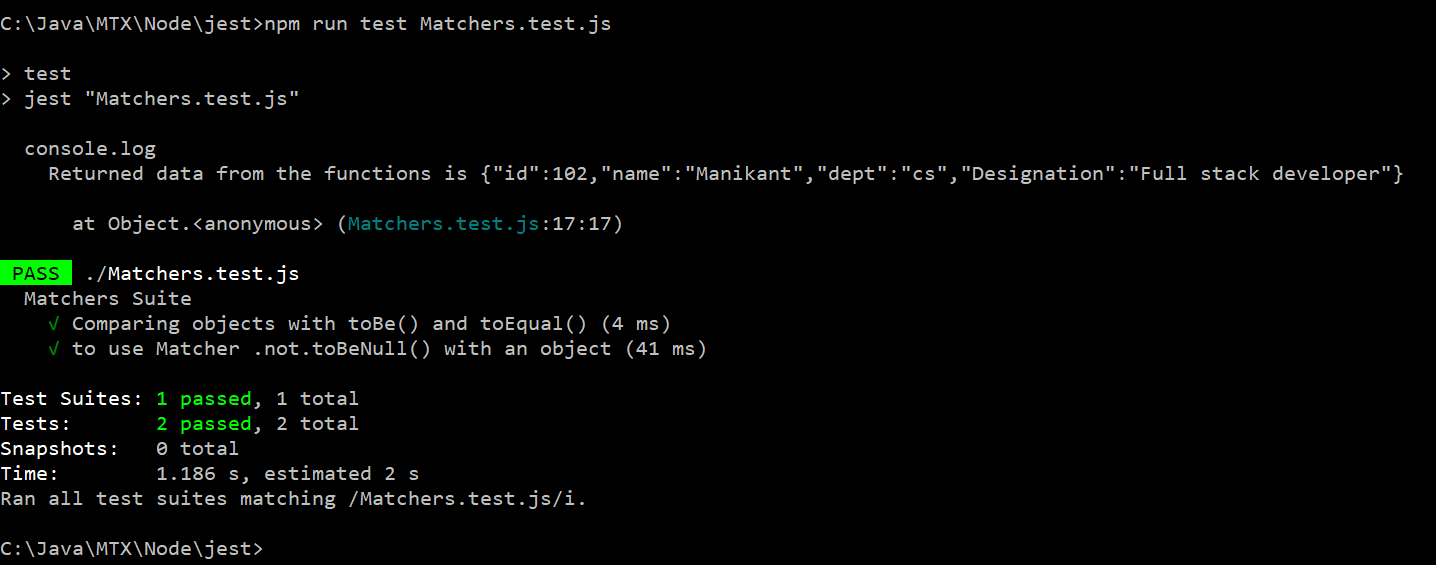
In Matchers.js



In Matchers.test.js

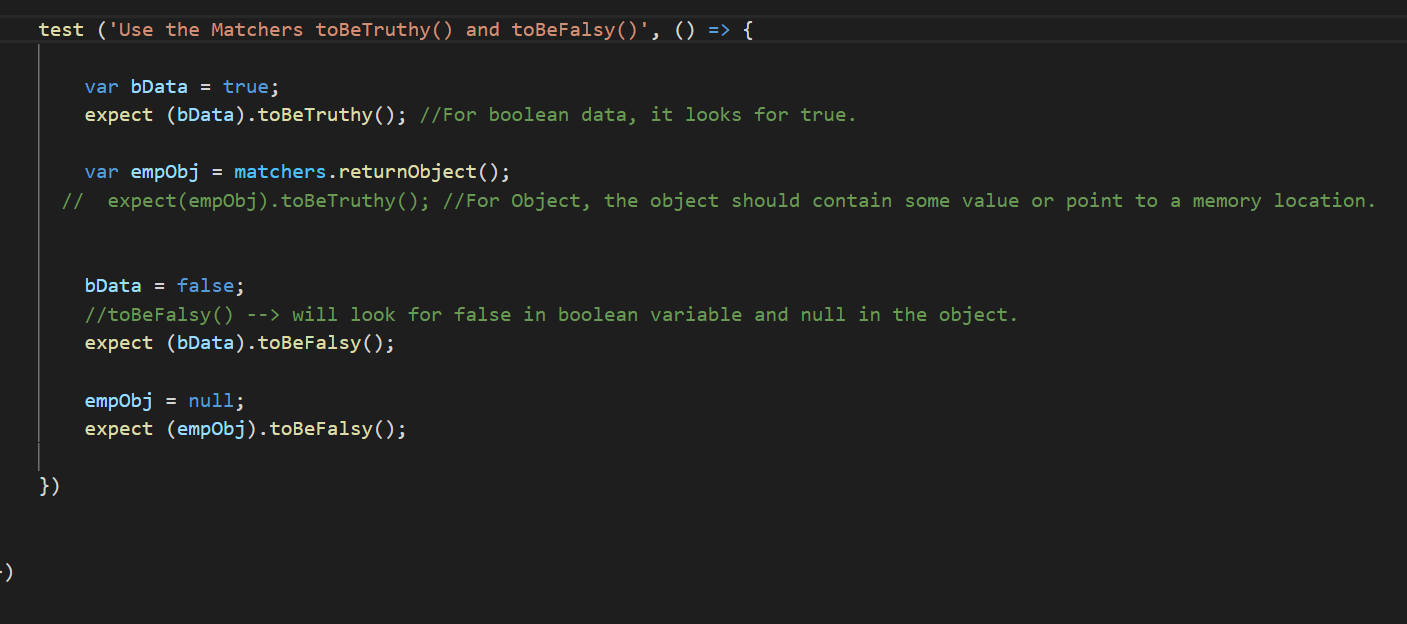


Run the test case:

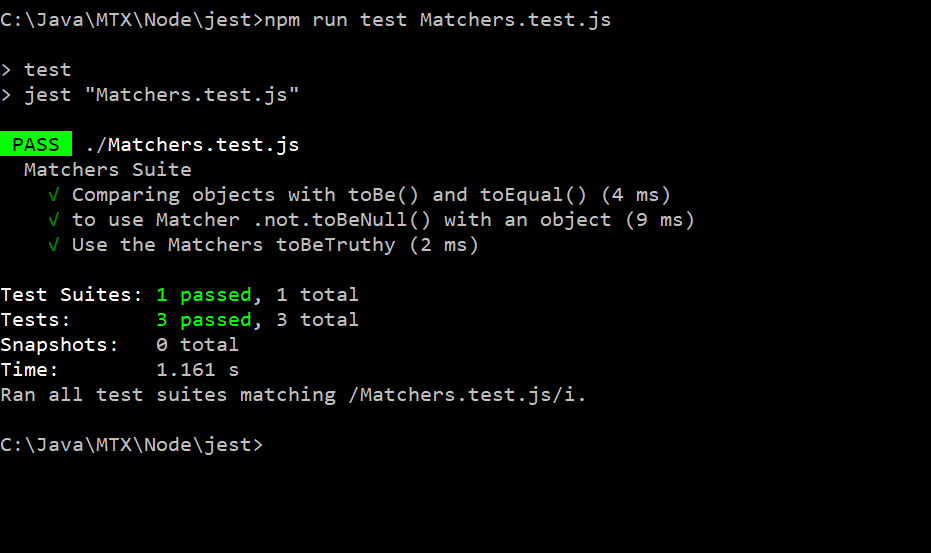


Using matcher toBeTruthy() and toBeFalsy()

In Matchers.js



Run the test

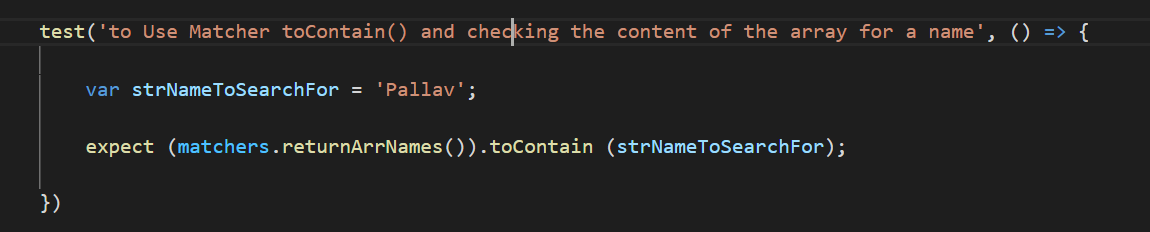


**Using Matcher toContain()**

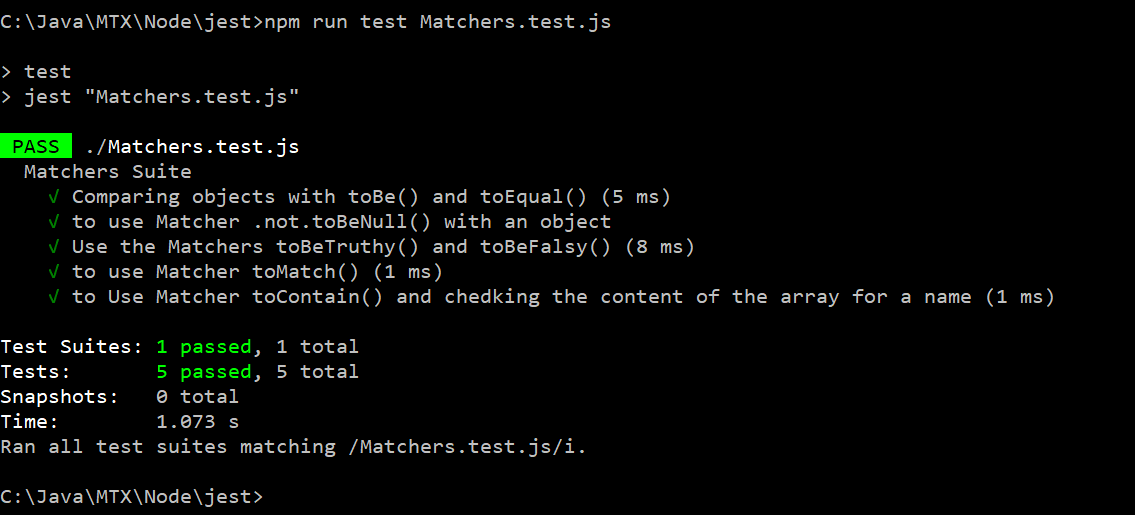
In Matchers.js



In Matchers.test.js



Run the tests:

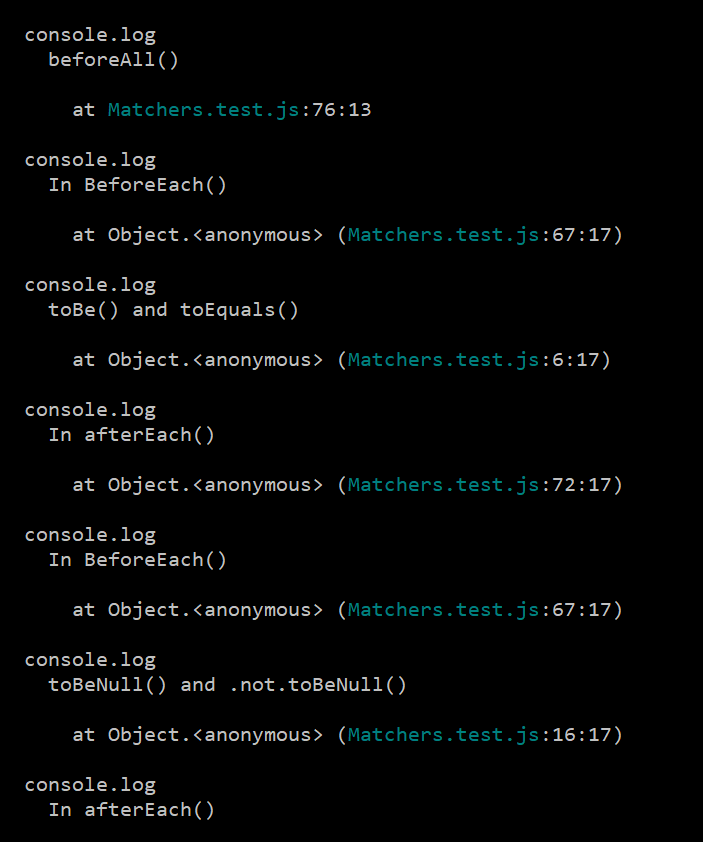


**Setup and Teardown functions…**

In Matchers.test.js



Output after running test using npm run test Matchers.test.js



**Node Assignments**

Assignment :

(1) Have 2 listeners and connect it with the same event using .on() and emit the event.

(2) Display the event names associated with the event emitter object using eventNames().

(3) Create 2 listners for an event 'myevent' using add listnerers. emit the 'myevent' and Display

the count of event listners and the event listerns. Remove one event listern and Display

the count of event listerns for 'myevent' and also which event is listening to 'myevent' event.

(4) Type the program on Reading a file and display the output. Do not have the file in the directory

and check the error it generates.

(5) From the given the file, along with the data, also display number of characters, words and lines

in the file.

(6) Write the script to search for a string in the file. Take the string from the user to serach for

(7) Print the number of occurance of the search string in the given file. Take the search string from

the user from command line.

(8) Type the code to see the writing into a stream as shown in the class.

(9) Copy one stream file to another. (Hint : Read from one file and write into antoher file)

(10) Create a folder in node, creat text file in it using commanline. Delete this file using unlink()

of node and delete the directory using node.

(11) Accept user input to create a folder and create the given folder and accept the folder name to

delete the folder and delete that folder.

(12) Along with the given parameters of OS module, try the following parameters and see the output.

os.homedir() - Home Directory.

JSON.stringify(os.networkInterfaces()) - Network Interfaces.

os.platform() - OS Platform.

os.release() - Release of the OS.

+os.tmpdir() - Temporary directory.

os.totalmem() -Total Memory in bytes.

os.type() - This method returns the operating system name.

os.uptime() - Total uptme of CPU in seconds.

(13) For the DNS module, the code is given with fixed domain names. Perform the similar operation

given by taking domain names from the user.

(14a) Practice web module with http://localhost:8081/index.html.

(14b) With web module --> When the url is http://localhost:8081/getData --> Let the node

return employee json object with the fields empid, name, dept and designation.

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

JEST assignments.

1. Create functions for subtaction, multiplication division and test them using jest.

2. Write functions for

a. Find Largest of 2 numbers.

b. Largest of 3 numbers.

c. Factorial of a numbers

d. Check given number is even or odd. For Even it returns true else it returns false. use toBeTruthy() or toBeFalsy() in jest

e. Largest element in the array.

f. Find whether the given element is there in the array or not.

and write the 6 testcases using jest to tese these functions are working fine or not.

3. Create array of 5 objects of Library domain which consists of the fields bookid, bookname, author, cost.

Create a function named returnBookDataById(num). This function will return book record based on the book id.

if the correct book id is sent then that record is returned else null data will be returned.

Create a jest test case to check this function returnBookDataById(). Pass the correct data and check with matchers not.toBeNull()

and send wrong the bookid and check the result with jest matcher toBeNull().

4. Create a function in javascript to store array of Objects of Library books. Have a map to filter only the bookids and store them in

an array. Return this array from the function. In Jest, write a test script to check whether a bookid is there or in the list of

BookIds returned by the javascript function.

5. Create a Login funciont in javascript, This function takes two parameters. They are UID and PWD. Login() will use global array of

Objects and check whether the given user and pwd is present as a record or not. If yes, this function will return as "Valid user"

else it will return as invalid user. Write a Jest test script by sending valid uid and pwd and verify that Login() is working Fine.

Also send the wrong uid and pwd and check that Login() is returning as "Invalid User"

6. Write a javascript function Named Registration(). This function takes 3 parameters they are (a) UID (b)PWD (c) Role (Admin/User).

Write a Jest test script to register a record in User Array of Objects. Afer registraton, Registration() should return as

"Registratio is successful" or failed to Register with a reason. Test for both the conditions in Jest testscript.

7. Write Jest test script to register a user first and after that with that user write a script to Login() and Login() should return as

"Valid user".

**SQL Assignments**

1. Create Products table with the fields

(a) ProductID, (b) ProductName, (c)Description, (d) cost.

and add 5 records into this table and commit the transaction and display all the records of this table.

2. Create the Customer table with the fields

(a) Customer Id (b) Customer Name (c) Address (d) City (e)Postal Code (f) Country

insert 5 records into this table display all of them using select statement.

3. Create another table by name purchase and it will have two fields

(a) Customer ID (b) ProductID.

and 5 records.

3. Display distinct cities from customer table.

4. Display the count of distinct cities in customer table. This Count should be displayed as the column name (Use AS in select).

5. Display all the rows from Product table where cost >= 15000.

7. Use customer table and Purchase table and display all the customers who have purchased. Display in the following format

Customer ID, Customer Name, City. (Use IN in select statement).

8. Display all the products information which has been purcahsed. Merge two tables in here Product Table and Customer table.

(Use subqueries for this).

9. Display the fields customer names and the country from customer table who lives in India.

10. Display all the customer names from customer table sorted on customer name.

11. Display all the customer names from cutomser table sored on customer name in ascending order and by country in descending order.

12. Insert into customer table for the fields (a) Customer Id (b) Customer Name (c) Address (d) City (e) Country. Specify the columns in insert into command.

13. Display all the fields of customer table whose pincode does not have any data.

14. Display all the fields of customer table for all the records where pincode is there.

15. Replace all product names to Vivo mobile when the product name is "mobile"

16. Update the product description based on Product Id and Product Name. Take any example and replace the data.

17. Delete customer record based on a id.

18. Delete all the customer records.

19. Display the product id, product name, cost whose cost the largest in the table. Use inner query for this.

20. Perform the above query using Limit in select query.

21. Show the product Id, name which has the lowest cost.

22. Display sum of the purcahses done by all the clients. i.e., show the sum of money spent by all the clients in purchases.

23. Rename the column postal code to pin code in customer table.

24. Add a new column Rating in products table.

25. create a temporary table named student with fields named id, name. Add few records in it. Drop all the records and delete this table.

26. Demonstrate rollback operation on customer table.

27. Display the how many purchases are done by each customer id. (group by of customer\_id in purchase table).

28. Display all the Customer IDs, Customer names and the products they purchased and their price.

29. Create employee Table with fields emp\_id, name, address, city, state, country and department table with fields dept\_id, dept\_name, emp\_id.

Display employee name and in which dept the employee is working on. (Use INNER JOIN).

Try Primary key and foreign keys and then only join the tables.

create table student\_SQL (id int, --> Primary key --> Student --> Student info.. 102 -- Rahsmi, 103 -- Prakash

name varchar(25),

stream varchar(25),

primary key (id))

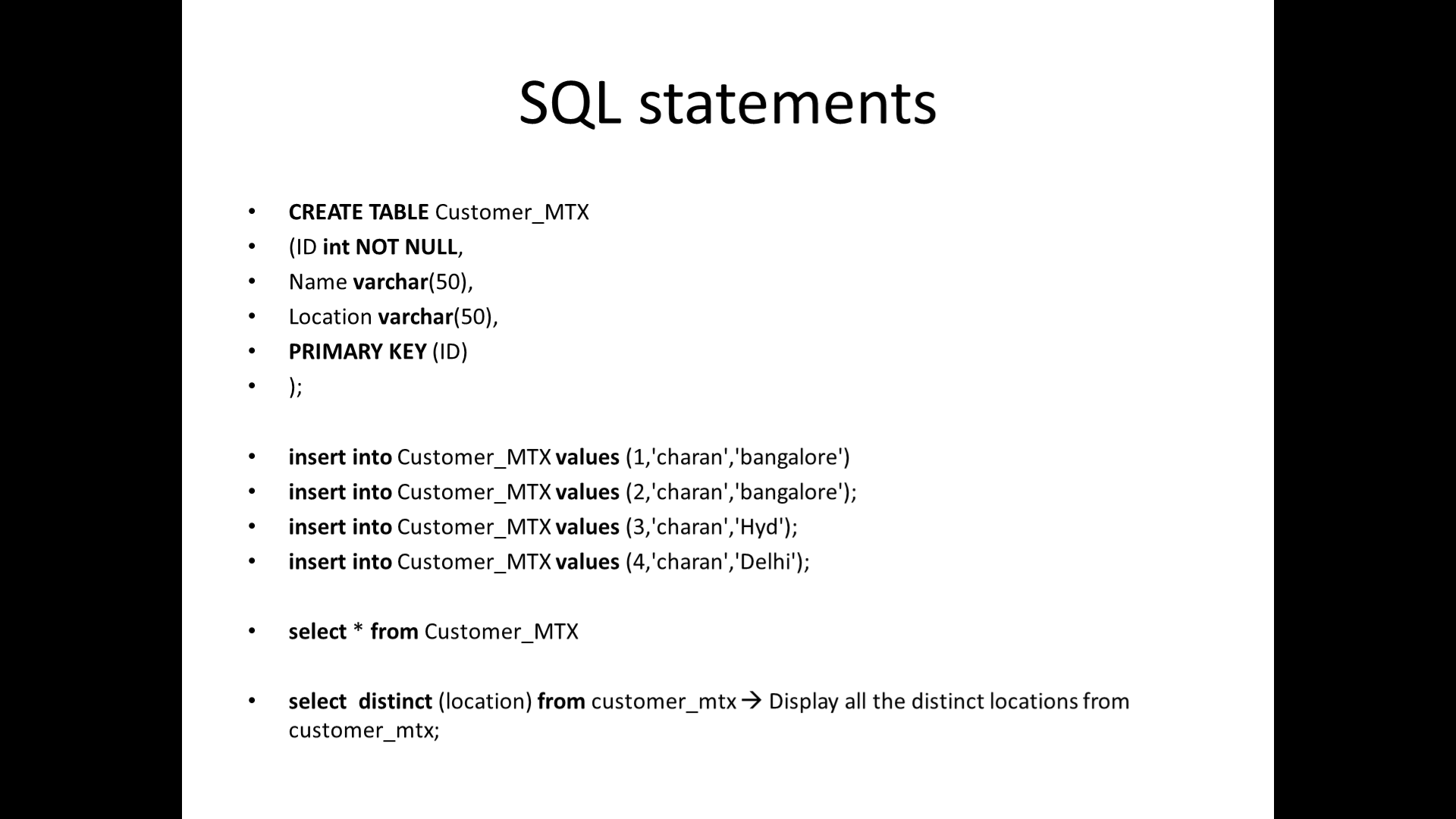
create table student\_marks (

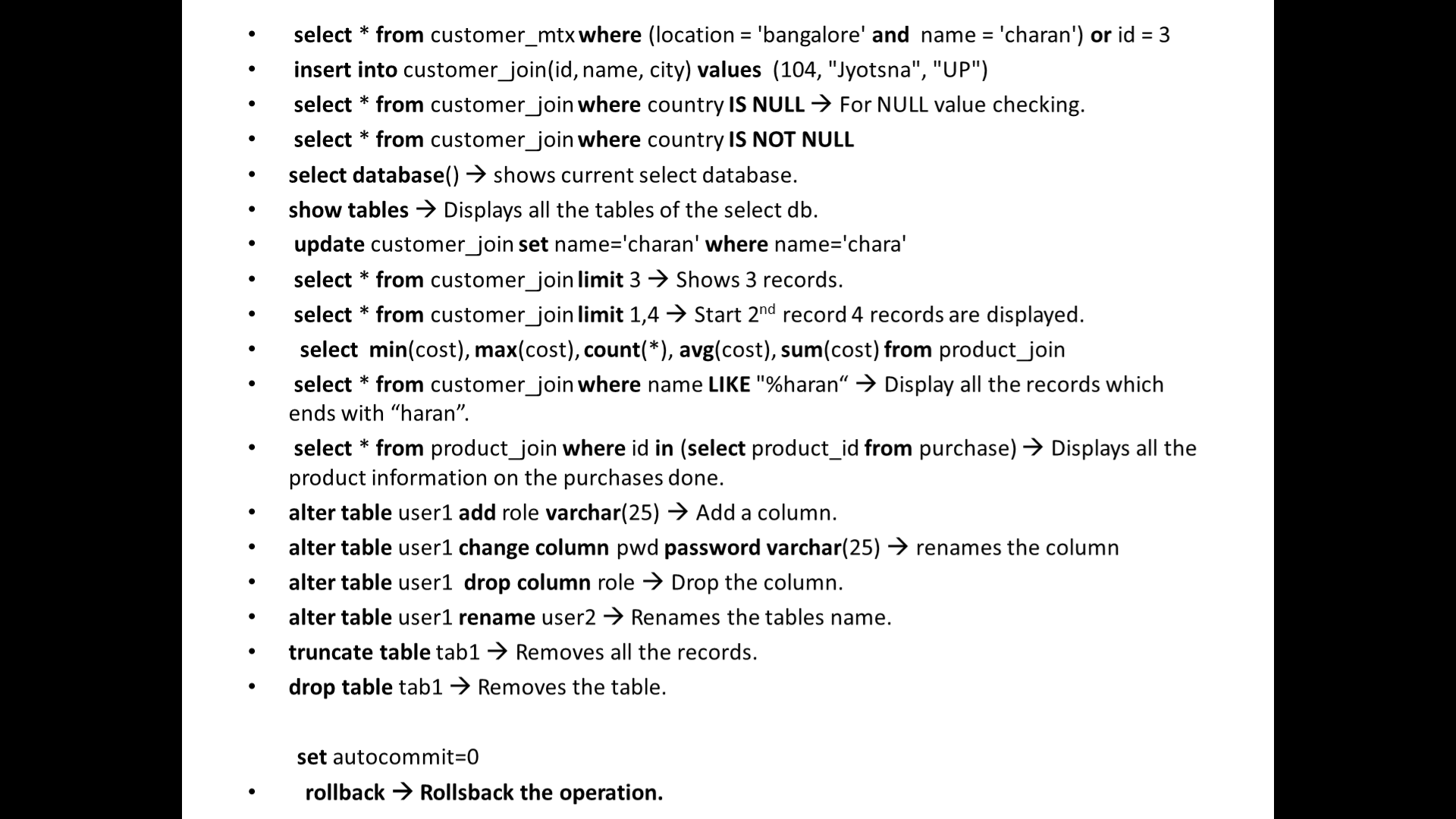
id int, --> Foreign key -- 102 -- 70, 103 - 72

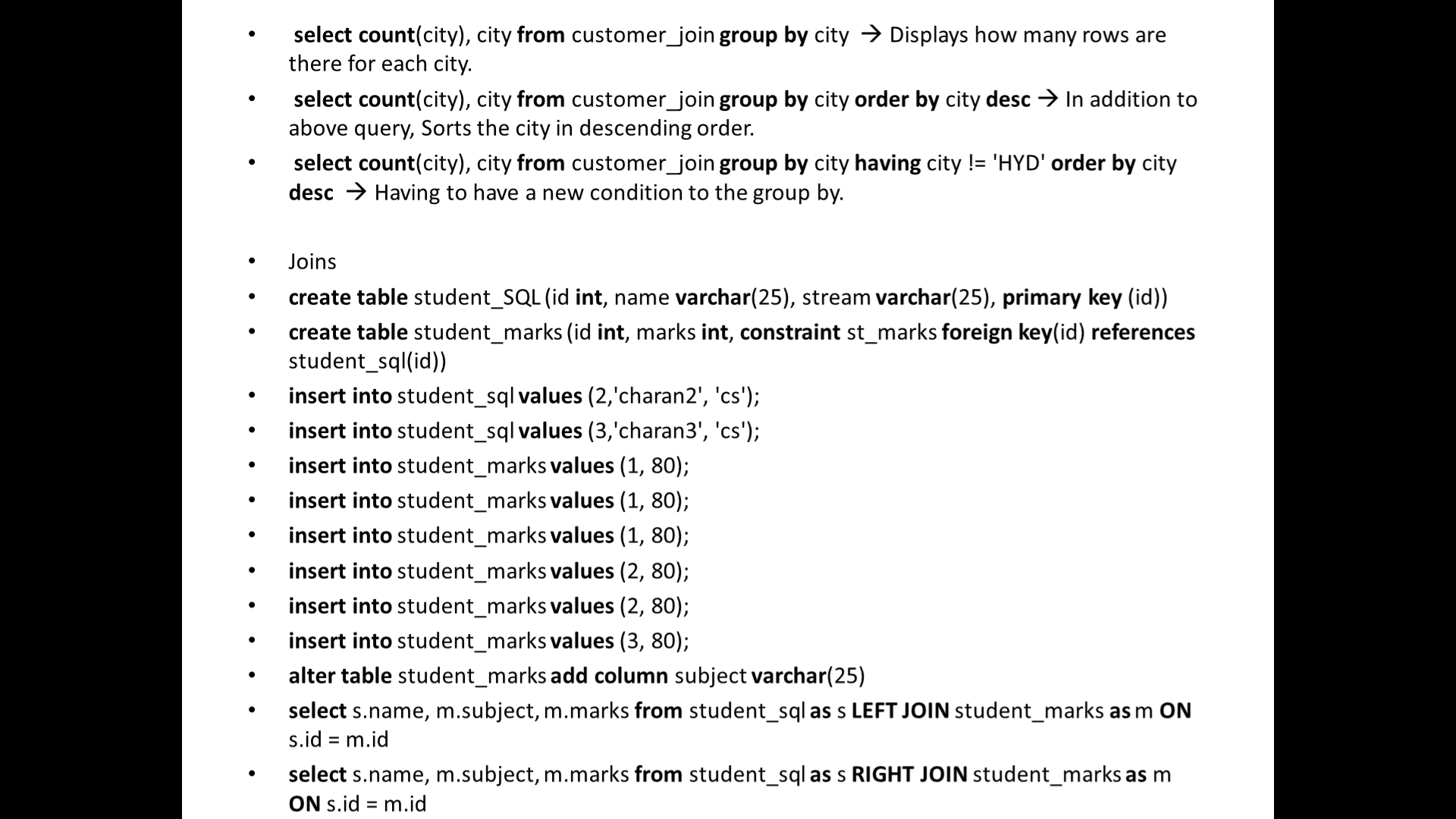
marks int,

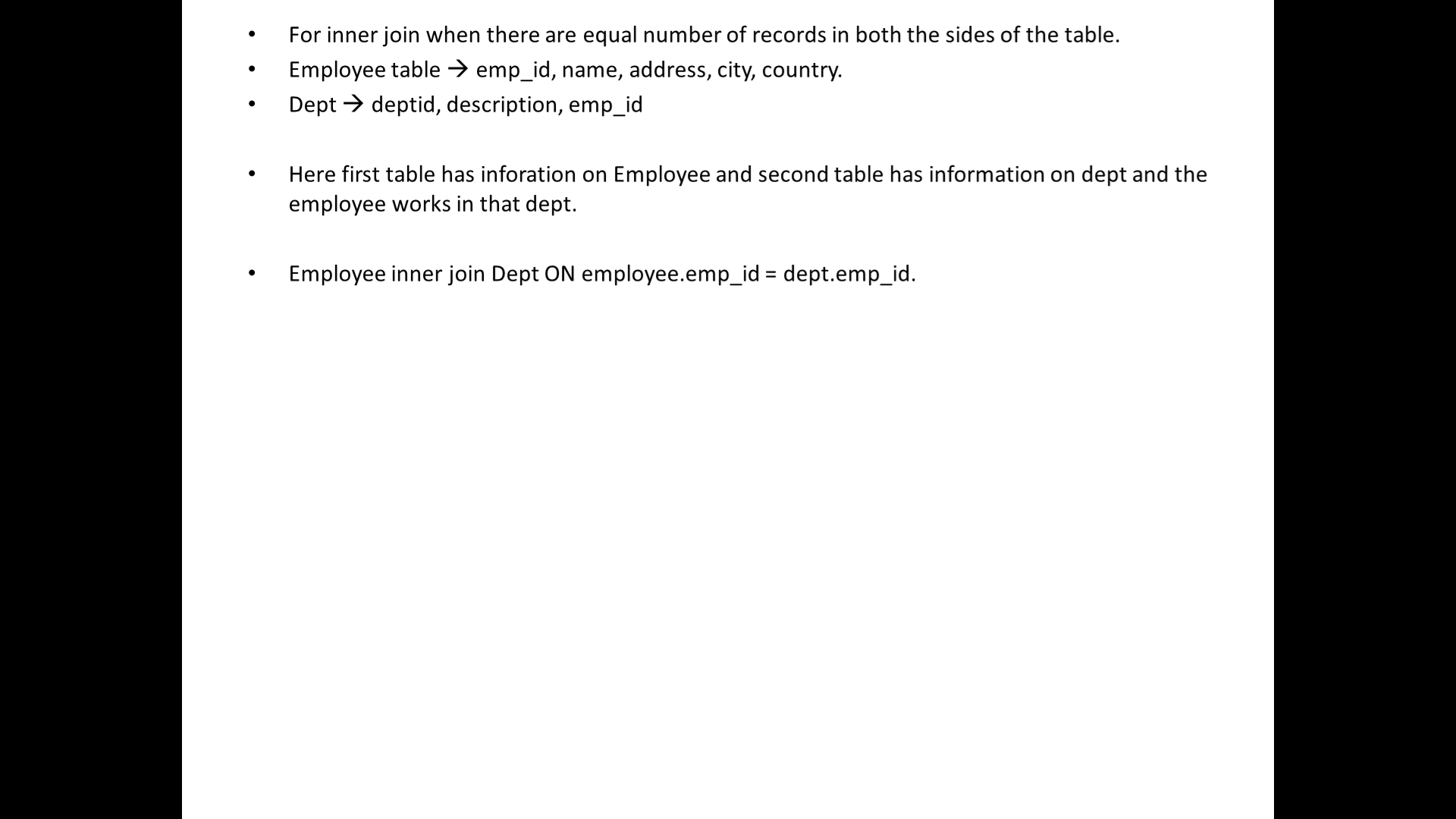
constraint st\_marks

foreign key(id) references student\_sql(id))

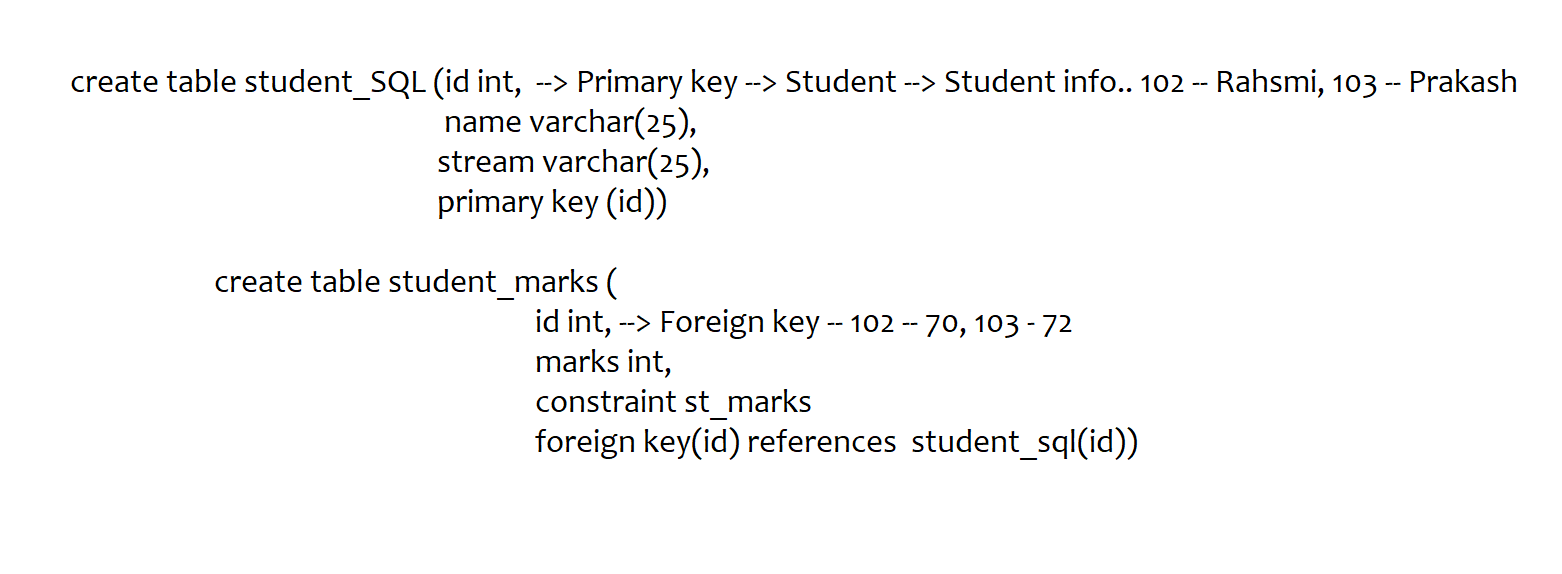








Use tables with primary key and foreign key



Queries executed in class :

