
Correzione LAIB 1

Effettuare una query ad un server MySql

```
const csv = require('csv-parser');
const fs = require('fs');
var TOTAL = 0;
var obj = {};
fs.createReadStream('studenti.csv')
  .pipe(csv())
  .on('data', (row) => {
    if (!(row['STA_NASCITA'] in obj)) obj[row['STA_NASCITA']] = 0;
    obj[row['STA_NASCITA']] ++;
    TOTAL ++;
  })
  .on('end', () => {
    for (var key in obj) obj[key]=(obj[key]/TOTAL)*100;
    console.log(obj);
  });
```

Filestream

- `fs.createReadStream()` reads sequentially from the current file position.
- The function `fs.createReadStream()` allows you to open up a readable stream in a very simple manner. All you have to do is pass the path of the file to start streaming in. It turns out that the response (as well as the request) objects are streams.
- link utili:
 - <https://nodejs.org/api/fs.html>
 - <https://nodejs.org/en/knowledge/advanced/streams/how-to-use-fs-create-read-stream/>

Pipe

`pipe()` => This just pipes the read stream to the response object (which goes to the client)

```
getName = (person) => person.name;
uppercase = (string) => string.toUpperCase();
get6Characters = (string) => string.substring(0, 6);
result = get6Characters(uppercase(getName({ name: 'Buckethead' })));
console.log(result);
```



```
getName = (person) => person.name;
uppercase = (string) => string.toUpperCase();
get6Characters = (string) => string.substring(0, 6);
result = pipe(
  getName,
  uppercase,
  get6Characters
)({ name: 'Buckethead' });
console.log(result);
```

freecodecamp.org/news/pipe-and-compose-in-javascript-5b04004ac937/

Events emitted by csv()

The following events are emitted during parsing:

- **data** => Emitted for each row of data parsed with the notable exception of the header row. Please see **Usage** for an example.
- **headers** => Emitted after the header row is parsed. The first parameter of the event callback is an `Array[String]` containing the header names.

```
fs.createReadStream('data.csv')  
  .pipe(csv())  
  .on('headers', (headers) => {  
    console.log(`First header: ${headers[0]}`)  
  })
```

- **Readable Stream Events** => Events available on Node built-in **Readable Streams** are also emitted. The `end` event should be used to detect the end of parsing.