Part1 :

<!DOCTYPE html>

<meta charset="utf-8">

<!-- Load d3.js you can use ither version if you want -->

<script src="https://d3js.org/d3.v4.js"></script>

<div id="my\_dataviz"> </div>

<div id="my\_data"> </div>

<!-- Create a div where the graph will take place -->

<div id="dataviz"></div>

<p style="background-color:powderblue;"> The distribution is mainly around 1 -3, and the highest is around 450.</p>

<script>

// set the dimensions and margins of the graph

// You can change these values these are just sample values given

var margin = {top: 20, right: 50, bottom: 50, left: 60},

width = 460 - margin.left - margin.right,

height = 400 - margin.top - margin.bottom;

// append the svg object to the body of the page

var svg = d3.select("#dataviz")

.append("svg")

.attr("width", width + margin.left + margin.right)

.attr("height", height + margin.top + margin.bottom)

.append("g")

.attr("transform",

"translate(" + margin.left + "," + margin.top + ")");

// uncomment the function and complete this function to plot required graphs

// d3.csv("https://github.com/vega/vega/blob/main/docs/data/seattle-weather.csv", function(data) {

d3.csv("seattle-weather.csv", function(data) {

console.log(data[0])

// X axis: scale and draw:

var x = d3.scaleLinear()

// d3.max(data, function(d) { return + d.wind }

// .domain([0, 1000]) // can use this instead of 1000 to have the max of data: d3.max(data, function(d) { return +d.price })

// var xaxis= d3.max(data, function(d) { return + d.wind })

.domain([0, 10])

.range([0, width]);

svg.append("g")

.attr("transform", "translate(0," + height + ")")

.call(d3.axisBottom(x));

// set the parameters for the histogram

var histogram = d3.histogram()

.value(function(d) { return d.wind; }) // I need to give the vector of value

.domain(x.domain()) // then the domain of the graphic

.thresholds(x.ticks(10)); // then the numbers of bins

// And apply this function to data to get the bins

var bins = histogram(data);

// Y axis: scale and draw:

var y = d3.scaleLinear()

.range([height, 0]);

y.domain([0, d3.max(bins, function(d) { return d.length; })]); // d3.hist has to be called before the Y axis obviously

svg.append("g")

.call(d3.axisLeft(y));

// append the bar rectangles to the svg element

svg.append("text")

.attr("class", "x label")

.attr("text-anchor", "end")

.attr("x", width)

.attr("y", height - 6)

.text("Group");

svg.append("text")

.attr("class", "y label")

.attr("text-anchor", "end")

.attr("x", 35)

.attr("y", 6)

.attr("dy", ".75em")

// .attr("transform", "rotate(-90)")

.text("wind");

svg.selectAll("rect")

.data(bins)

.enter()

.append("rect")

.attr("x", 1)

.attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })

.attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })

.attr("height", function(d) { return height - y(d.length); })

.style("fill", "#69b3a2")

});

</script>

<!DOCTYPE html>

<meta charset="utf-8">

<!-- Load d3.js you can use ither version if you want -->

<script src="https://d3js.org/d3.v4.js"></script>

<div id="my\_dataviz"> </div>

<div id="my\_data"> </div>

<!-- Create a div where the graph will take place -->

<div id="dataviz"></div>

<p> The distribution is mainly around 1 -3, and the highest is around 260.</p>

<script>

// set the dimensions and margins of the graph

// You can change these values these are just sample values given

var margin = {top: 20, right: 50, bottom: 50, left: 60},

width = 460 - margin.left - margin.right,

height = 400 - margin.top - margin.bottom;

// append the svg object to the body of the page

var svg = d3.select("#dataviz")

.append("svg")

.attr("width", width + margin.left + margin.right)

.attr("height", height + margin.top + margin.bottom)

.append("g")

.attr("transform",

"translate(" + margin.left + "," + margin.top + ")");

// uncomment the function and complete this function to plot required graphs

// d3.csv("https://github.com/vega/vega/blob/main/docs/data/seattle-weather.csv", function(data) {

d3.csv("seattle-weather.csv", function(data) {

console.log(data[0])

// X axis: scale and draw:

var x = d3.scaleLinear()

// d3.max(data, function(d) { return + d.wind }

// .domain([0, 1000]) // can use this instead of 1000 to have the max of data: d3.max(data, function(d) { return +d.price })

// var xaxis= d3.max(data, function(d) { return + d.wind })

.domain([0, 10])

.range([0, width]);

svg.append("g")

.attr("transform", "translate(0," + height + ")")

.call(d3.axisBottom(x));

// set the parameters for the histogram

var histogram = d3.histogram()

.value(function(d) { return d.wind; }) // I need to give the vector of value

.domain(x.domain()) // then the domain of the graphic

.thresholds(x.ticks(25)); // then the numbers of bins

// And apply this function to data to get the bins

var bins = histogram(data);

// Y axis: scale and draw:

var y = d3.scaleLinear()

.range([height, 0]);

y.domain([0, d3.max(bins, function(d) { return d.length; })]); // d3.hist has to be called before the Y axis obviously

svg.append("g")

.call(d3.axisLeft(y));

// append the bar rectangles to the svg element

svg.append("text")

.attr("class", "x label")

.attr("text-anchor", "end")

.attr("x", width)

.attr("y", height - 6)

.text("Group");

svg.append("text")

.attr("class", "y label")

.attr("text-anchor", "end")

.attr("x", 35)

.attr("y", 6)

.attr("dy", ".75em")

// .attr("transform", "rotate(-90)")

.text("wind");

svg.selectAll("rect")

.data(bins)

.enter()

.append("rect")

.attr("x", 1)

.attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })

.attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })

.attr("height", function(d) { return height - y(d.length); })

.style("fill", "#69b3a2")

});

</script>

<!DOCTYPE html>

<meta charset="utf-8">

<!-- Load d3.js you can use ither version if you want -->

<script src="https://d3js.org/d3.v4.js"></script>

<div id="my\_dataviz"> </div>

<div id="my\_data"> </div>

<!-- Create a div where the graph will take place -->

<div id="dataviz"></div>

<p> The distribution is mainly occupied by rain and sun. The highest percent is 44%</p>

<script>

// set the dimensions and margins of the graph

var width = 450

height = 450

margin = 40

// The radius of the pieplot is half the width or half the height (smallest one). I subtract a bit of margin.

var radius = Math.min(width, height) / 2 - margin

var arcGenerator = d3.arc()

.innerRadius(0)

.outerRadius(radius+200)

// append the svg object to the div called 'my\_dataviz'

var svg1 = d3.select("#my\_dataviz")

.append("svg")

.attr("width", width)

.attr("height", height)

.append("g")

.attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");

var data = { sun: 640, snow : 26, rain :641, fog:101, drizzle:53}

// Compute the position of each group on the pie:

// set the color scale

var color = d3.scaleOrdinal()

.domain(data)

.range(["#98abc5", "#8a89a6", "#7b6888", "#6b486b", "#a05d56"])

var pie = d3.pie()

.value(function(d) {return d.value; })

var data\_ready = pie(d3.entries(data))

// Build the pie chart: Basically, each part of the pie is a path that we build using the arc function.

svg1

.selectAll('whatever')

.data(data\_ready)

.enter()

.append('path')

.attr('d', d3.arc()

.innerRadius(0)

.outerRadius(radius)

)

.attr('fill', function(d){ return(color(d.data.key)) })

.attr("stroke", "black")

.style("stroke-width", "2px")

.style("opacity", 0.7)

svg1

.selectAll('mySlices')

.data(data\_ready)

.enter()

.append('text')

.text(function(d){ return d.data.key ; })

.attr("transform", function(d) { return "translate(" + arcGenerator.centroid(d) + ")"; })

.style("text-anchor", "middle")

.style("font-size", 10)

var arc = d3.arc()

.innerRadius(0)

.outerRadius(radius-5)

svg1

.selectAll('mySlices')

.data(data\_ready)

.enter()

.append('text')

.text(function(d){ return (Math.round(d.data.value/1461 \* 100)).toFixed(0) + '%' ; })

.attr("transform", function(d) { return "translate(" + arc.centroid(d) + ")"; })

.style("text-anchor", "middle")

.style("font-size", 10)

</script>

<!DOCTYPE html>

<meta charset="utf-8">

<!-- Load d3.js you can use ither version if you want -->

<script src="https://d3js.org/d3.v4.js"></script>

<div id= "my\_data"> </div>

<p> 2015 April and October have the high value around 55. </p>

<script>

var margin = {top: 20, right: 50, bottom: 50, left: 60},

width = 660 - margin.left - margin.right,

height = 400 - margin.top - margin.bottom;

var svg2 = d3.select("#my\_data")

.append("svg")

.attr("width", width + margin.left + margin.right)

.attr("height", height + margin.top + margin.bottom)

.append("g")

.attr("transform",

"translate(" + margin.left + "," + margin.top + ")");

d3.csv("seattle-weather.csv", function(d) {

return {date: d3.timeParse("%Y-%m-%d")(d.date), value: d.precipitation}

},

function(data){

var x = d3.scaleTime()

.domain(d3.extent(data,function(d){ return d.date; }))

.range([0, width]);

svg2.append("g")

.attr("transform", "translate(0,"+ height +")")

.call(d3.axisBottom(x));

var y = d3.scaleLinear()

.domain([0, d3.max(data, function(d){return +d.value; })])

.range([height, 0]);

svg2.append("g")

.call(d3.axisLeft(y));

svg2.append("path")

.datum(data)

.attr("fill","none")

.attr("stroke","steelblue")

.attr("stroke-width", 1.5)

.attr("d", d3.line()

.x(function(d){return x(d.date) })

.y(function(d){return y(d.value) })

)

})

</script>

1.3.1

<script src="https://d3js.org/d3.v4.js"></script>

<!-- Create a div where the graph will take place -->

<div id="my\_dataviz"></div>

<p>

<label># bins</label>

<input type="number" min="1" max="100" step="30" value="20" id="nBin">

</p>

<script>

// set the dimensions and margins of the graph

var margin = {top: 10, right: 30, bottom: 30, left: 40},

width = 460 - margin.left - margin.right,

height = 400 - margin.top - margin.bottom;

// append the svg object to the body of the page

var svg = d3.select("#my\_dataviz")

.append("svg")

.attr("width", width + margin.left + margin.right)

.attr("height", height + margin.top + margin.bottom)

.append("g")

.attr("transform",

"translate(" + margin.left + "," + margin.top + ")");

// get the data

d3.csv("seattle-weather.csv", function(data) {

// X axis: scale and draw:

var x = d3.scaleLinear()

.domain([0, 10]) // can use this instead of 1000 to have the max of data: d3.max(data, function(d) { return +d.price })

.range([0, width]);

svg.append("g")

.attr("transform", "translate(0," + height + ")")

.call(d3.axisBottom(x));

// Y axis: initialization

var y = d3.scaleLinear()

.range([height, 0]);

var yAxis = svg.append("g")

// A function that builds the graph for a specific value of bin

function update(nBin) {

// set the parameters for the histogram

var histogram = d3.histogram()

.value(function(d) { return d.wind; }) // I need to give the vector of value

.domain(x.domain()) // then the domain of the graphic

.thresholds(x.ticks(nBin)); // then the numbers of bins

// And apply this function to data to get the bins

var bins = histogram(data);

console.log(bins)

// Y axis: update now that we know the domain

y.domain([0, d3.max(bins, function(d) { return d.length; })]); // d3.hist has to be called before the Y axis obviously

yAxis

.transition()

.duration(100)

.call(d3.axisLeft(y));

// Join the rect with the bins data

var u = svg.selectAll("rect")

.data(bins)

// Manage the existing bars and eventually the new ones:

u

.enter()

.append("rect") // Add a new rect for each new elements

.merge(u) // get the already existing elements as well

.transition() // and apply changes to all of them

.duration(1000)

.attr("x", 1)

.attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })

.attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })

.attr("height", function(d) { return height - y(d.length); })

.style("fill", "#69b3a2")

// If less bar in the new histogram, I delete the ones not in use anymore

u

.exit()

.remove()

}

// Initialize with 20 bins

update(20)

// Listen to the button -> update if user change it

d3.select("#nBin").on("input", function() {

update(+this.value);

});

});

</script>

</script>

<!Doctype Html>

<Html>

<Head>

<Title>

Make a Dropdown Menu using Internal CSS

</Title>

<style>

.dropbtn {

background-color: yellow;

color: black;

padding: 10px;

font-size: 12px;

}

.dropdown {

display: inline-block;

position: relative

}

.dropdown-content {

position: absolute;

background-color: lightgrey;

min-width: 200px;

display: none;

z-index: 1;

}

.dropdown-content a {

color: black;

padding: 12px 16px;

text-decoration: none;

display: block;

}

.dropdown-content a:hover {

background-color: orange;

}

.dropdown:hover .dropdown-content {

display: block;

}

.dropdown:hover .dropbtn {

background-color: grey;

}

</style>

</Head>

<Body>

Dropdown Menu <br>

<div class="dropdown">

<button class="dropbtn"> Choose attributes </button>

<div class="dropdown-content">

<a href="preci.html"> precipitation </a>

<a href="temp\_max.html"> temp\_max </a>

<a href="temp\_min.html"> temp\_min </a>

<a href="1.2.1to2.html"> wind </a>

</div>

</div>

</Body>

</Html>

Part2:

2.1

<!DOCTYPE html>

<html>

<head>

<title>CSV File to HTML Table Using AJAX jQuery</title>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" />

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<div class="table-responsive">

<h1 align="center">CSV File to HTML Table Using AJAX jQuery</h1>

<br />

<div align="center">

<button type="button" name="load\_data" id="load\_data" class="btn btn-info">Load Data</button>

</div>

<br />

<div id="table">

</div>

</div>

</div>

</body>

</html>

<script>

$(document).ready(function(){

$('#load\_data').click(function(){

$.ajax({

url:"auto-mpg.csv",

dataType:"text",

success:function(data)

{

var data = data.split(/\r?\n|\r/);

var table\_data = '<table class="table table-bordered table-striped">';

for(var count = 0; count< 6; count++)

{

var cell\_data = data[count].split(",");

table\_data += '<tr>';

for(var cell\_count=0; cell\_count<cell\_data.length; cell\_count++)

{

if(count === 0)

{

table\_data += '<th>'+cell\_data[cell\_count]+'</th>';

}

else

{

table\_data += '<td>'+cell\_data[cell\_count]+'</td>';

}

}

table\_data += '</tr>';

}

table\_data += '</table>';

// console.log(table\_data)

$('#table').html(table\_data);

}

});

});

});

</script>

<!DOCTYPE html>

<html>

<head>

<title>Read Data from HTML Table uisng JavaScript</title>

<style>

table, th, td

{

border: solid 1px #DDD;

border-collapse: collapse;

padding: 2px 3px;

text-align: center;

}

th {

font-weight:bold;

}

</style>

</head>

<body>

<table id="empTable">

<tr>

<th>each model year </th>

<th> number of cars </th>

</tr>

<tr>

<td>70</td>

<td> 29</td>

</tr>

<tr>

<td>71</td>

<td>28</td>

</tr>

<tr>

<td>72</td>

<td>28</td>

</tr>

<tr>

<td>73</td>

<td>40</td>

</tr>

<tr>

<td>74</td>

<td>27</td>

</tr>

<tr>

<td>75</td>

<td>30</td>

</tr>

<tr>

<td>76</td>

<td>34</td>

</tr>

<tr>

<td>77</td>

<td>28</td>

</tr>

<tr>

<td>78</td>

<td>29</td>

</tr>

<tr>

<td>79</td>

<td>29</td>

</tr>

<tr>

<td>80</td>

<td>29</td>

</tr>

<tr>

<td>81</td>

<td>31</td>

</tr>

</table>

<p><input type="button" id="bt" value="Show Table Data" onclick="showTableData()" /></p>

<p id="info"></p>

</body>

<script>

data = {70:29, 71:28, 72:28,73:40, 74:27, 75:30, 76:34 , 77:28 , 78:29 , 79:29, 80: 29, 81 : 31 }

function showTableData() {

document.getElementById('info').innerHTML = "";

var myTab = document.getElementById('empTable');

// LOOP THROUGH EACH ROW OF THE TABLE AFTER HEADER.

for (i = 1; i < myTab.rows.length; i++) {

// GET THE CELLS COLLECTION OF THE CURRENT ROW.

var objCells = myTab.rows.item(i).cells;

// LOOP THROUGH EACH CELL OF THE CURENT ROW TO READ CELL VALUES.

for (var j = 0; j < objCells.length; j++) {

info.innerHTML = info.innerHTML + ' ' + objCells.item(j).innerHTML;

}

info.innerHTML = info.innerHTML + '<br />'; // ADD A BREAK (TAG).

}

}

</script>

</html>

2.1.3

<!DOCTYPE html>

<html>

<head>

<title>CSV File to HTML Table Using AJAX jQuery</title>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" />

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<div class="table-responsive">

<h1 align="center">CSV File to HTML Table Using AJAX jQuery</h1>

<br />

<div align="center">

<button type="button" name="load\_data" id="load\_data" class="btn btn-info">Load Data</button>

</div>

<br />

<div id="employee\_table">

</div>

</div>

</div>

</body>

</html>

<script>

$(document).ready(function(){

$('#load\_data').click(function(){

$.ajax({

url:"2.1.3\_cur.csv",

dataType:"text",

success:function(data)

{

var data = data.split(/\r?\n|\r/);

var table\_data = '<table class="table table-bordered table-striped">';

for(var count = 0; count<data.length; count++)

{

var cell\_data = data[count].split(",");

table\_data += '<tr>';

for(var cell\_count=0; cell\_count<cell\_data.length; cell\_count++)

{

if(count === 0)

{

table\_data += '<th>'+cell\_data[cell\_count]+'</th>';

}

else

{

table\_data += '<td>'+cell\_data[cell\_count]+'</td>';

}

}

table\_data += '</tr>';

}

table\_data += '</table>';

// console.log(table\_data)

$('#employee\_table').html(table\_data);

}

});

});

});

</script>

2.1.4

<!DOCTYPE html>

<html>

<head>

<title>CSV File to HTML Table Using AJAX jQuery</title>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" />

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<div class="table-responsive">

<h1 align="center">CSV File to HTML Table Using AJAX jQuery</h1>

<br />

<div align="center">

<button type="button" name="load\_data" id="load\_data" class="btn btn-info">Load Data</button>

</div>

<br />

<div id="employee\_table">

</div>

</div>

</div>

</body>

</html>

<script>

$(document).ready(function(){

$('#load\_data').click(function(){

$.ajax({

url:"2.1.4.csv",

dataType:"text",

success:function(data)

{

var data = data.split(/\r?\n|\r/);

var table\_data = '<table class="table table-bordered table-striped">';

for(var count = 0; count<data.length; count++)

{

var cell\_data = data[count].split(" ");

table\_data += '<tr>';

for(var cell\_count=0; cell\_count<cell\_data.length; cell\_count++)

{

if(count === 0)

{

table\_data += '<th>'+cell\_data[cell\_count]+'</th>';

}

else

{

table\_data += '<td>'+cell\_data[cell\_count]+'</td>';

}

}

table\_data += '</tr>';

}

table\_data += '</table>';

// console.log(table\_data)

$('#employee\_table').html(table\_data);

}

});

});

});

</script>

2.2

<!DOCTYPE html>

<meta charset="utf-8">

<!-- Load d3.js you can use ither version if you want -->

<script src="https://d3js.org/d3.v4.js"></script>

<div id= "my\_data"> </div>

<script>

var margin = {top: 20, right: 50, bottom: 50, left: 60},

width = 660 - margin.left - margin.right,

height = 400 - margin.top - margin.bottom;

var svg2 = d3.select("#my\_data")

.append("svg")

.attr("width", width + margin.left + margin.right)

.attr("height", height + margin.top + margin.bottom)

.append("g")

.attr("transform",

"translate(" + margin.left + "," + margin.top + ")");

d3.csv("2.1.3\_cur.csv", function(d) {

console.log(d)

return {date: d.modelyear, value: d.cylinder}

},

function(data){

var x = d3.scaleLinear()

.domain(d3.extent(data,function(d){ return d.date; }))

.range([0, width]);

svg2.append("g")

.attr("transform", "translate(0,"+ height +")")

.call(d3.axisBottom(x));

var y = d3.scaleLinear()

.domain([0, d3.max(data, function(d){return +d.value; })])

.range([height, 0]);

svg2.append("g")

.call(d3.axisLeft(y));

svg2.append("path")

.datum(data)

.attr("fill","none")

.attr("stroke","steelblue")

.attr("stroke-width", 1.5)

.attr("d", d3.line()

.x(function(d){return x(d.date) })

.y(function(d){return y(d.value) })

)

})

</script>

2.2.2

<!DOCTYPE html>

<meta charset="utf-8">

<!-- Load d3.js you can use ither version if you want -->

<script src="https://d3js.org/d3.v4.js"></script>

<div id= "my\_data"> </div>

<script>

var margin = {top: 20, right: 50, bottom: 50, left: 60},

width = 660 - margin.left - margin.right,

height = 400 - margin.top - margin.bottom;

var svg2 = d3.select("#my\_data")

.append("svg")

.attr("width", width + margin.left + margin.right)

.attr("height", height + margin.top + margin.bottom)

.append("g")

.attr("transform",

"translate(" + margin.left + "," + margin.top + ")");

d3.csv("2.1.3\_cur.csv", function(d) {

console.log(d)

return {date: d.modelyear, value: d.cylinder}

},

function(data){

var x = d3.scaleLinear()

.domain(d3.extent(data,function(d){ return d.date; }))

.range([0, width]);

svg2.append("g")

.attr("transform", "translate(0,"+ height +")")

.call(d3.axisBottom(x));

var y = d3.scaleLinear()

.domain([0, d3.max(data, function(d){return +d.value; })])

.range([height, 0]);

svg2.append("g")

.call(d3.axisLeft(y));

svg2.append("path")

.datum(data)

.attr("fill","none")

.attr("stroke","steelblue")

.attr("stroke-width", 1.5)

.attr("d", d3.line()

.x(function(d){return x(d.date) })

.y(function(d){return y(d.value) })

)

})

</script>

2.2.3

<!DOCTYPE html>

<meta charset="utf-8">

<!-- Load d3.js you can use ither version if you want -->

<script src="https://d3js.org/d3.v4.js"></script>

<div id="my\_dataviz"> </div>

<div id="my\_data"> </div>

<!-- Create a div where the graph will take place -->

<div id="dataviz"></div>

<script>

// set the dimensions and margins of the graph

// You can change these values these are just sample values given

var margin = {top: 20, right: 50, bottom: 50, left: 60},

width = 460 - margin.left - margin.right,

height = 400 - margin.top - margin.bottom;

// append the svg object to the body of the page

var svg = d3.select("#dataviz")

.append("svg")

.attr("width", width + margin.left + margin.right)

.attr("height", height + margin.top + margin.bottom)

.append("g")

.attr("transform",

"translate(" + margin.left + "," + margin.top + ")");

// uncomment the function and complete this function to plot required graphs

// d3.csv("https://github.com/vega/vega/blob/main/docs/data/seattle-weather.csv", function(data) {

d3.csv("auto-mpg.csv", function(data) {

console.log(data[0])

// X axis: scale and draw:

var x = d3.scaleLinear()

// d3.max(data, function(d) { return + d.wind }

// .domain([0, 1000]) // can use this instead of 1000 to have the max of data: d3.max(data, function(d) { return +d.price })

// var xaxis= d3.max(data, function(d) { return + d.wind })

.domain([8, 10])

.range([0, width]);

svg.append("g")

.attr("transform", "translate(0," + height + ")")

.call(d3.axisBottom(x));

// set the parameters for the histogram

// acceleration numberofcar

var histogram = d3.histogram()

.value(function(d) { return d.accel; }) // I need to give the vector of value

.domain(x.domain()) // then the domain of the graphic

.thresholds(x.ticks(10)); // then the numbers of bins

// And apply this function to data to get the bins

var bins = histogram(data);

// Y axis: scale and draw:

var y = d3.scaleLinear()

.range([height, 0]);

y.domain([0, d3.max(bins, function(d) { return d.length; })]); // d3.hist has to be called before the Y axis obviously

svg.append("g")

.call(d3.axisLeft(y));

// append the bar rectangles to the svg element

svg.append("text")

.attr("class", "x label")

.attr("text-anchor", "end")

.attr("x", width)

.attr("y", height - 6)

.text("Group");

svg.append("text")

.attr("class", "y label")

.attr("text-anchor", "end")

.attr("x", 35)

.attr("y", 6)

.attr("dy", ".75em")

// .attr("transform", "rotate(-90)")

.text("acc");

svg.selectAll("rect")

.data(bins)

.enter()

.append("rect")

.attr("x", 1)

.attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })

.attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })

.attr("height", function(d) { return height - y(d.length); })

.style("fill", "#69b3a2")

});

</script>

Part3:

<html>

<head>

<style>

svg{

outline: 3px solid green;

}

</style>

<script src="https://d3js.org/d3.v7.min.js"></script>

<script src="./js/fisheye.js"></script>

</head>

<body>

<div id="fig-3"></div>

<script>

const visConfig = {

figMargin: 50,

width: 1200,

height: 900,

};

function exportNodes(data){

const sourceNodes = data.map(e => e["source"]);

const targetNodes = data.map(e => e["target"]);

let result = new Set([

...sourceNodes,

...targetNodes

]);

result = [...result].map(e=>({

name: e

}))

return result;

}

function drawNetwork(sel, data){

const nodes = exportNodes(data);

const links = data;

const fisheye = d3.fisheye.circular()

.radius(200)

.distortion(10);

const svg = sel.append("svg")

.attr("width", visConfig.width)

.attr("height", visConfig.height);

const lines = svg.selectAll("NONE")

.data(links)

.enter()

.append("line")

.style("stroke", "rgba(255,0,0,0.3)")

const circleGroup = svg.selectAll("NONE")

.data(nodes)

.enter()

.append("g");

const circles = circleGroup.append("circle")

.attr("r", 10)

.style("fill", "lightgreen");

circleGroup.append("text")

.style("user-select", "none")

.text(node => node.name);

const forceSimul = d3.forceSimulation(nodes)

.force("center", d3.forceCenter(visConfig.width/2, visConfig.height/2+100))

.force("collide", d3.forceCollide().radius(12))

.force("charge", d3.forceManyBody().strength(-130))

.force("links", d3.forceLink(links).id(node=>node.name).distance(60))

.on("tick", tick);

// svg.on("mousemove", function(e) {

// fisheye.focus([e.offsetX, e.offsetY]);

//

// circleGroup.each(function(d) { d.fisheye = fisheye(d); })

// .attr("transform", function(d) {

// const t = [d.fisheye.x, d.fisheye.y];

// return `translate(${t})`

// })

//

// circles.attr("r", function(d) { return d.fisheye.z \* 4.5; });

//

// lines.attr("x1", function(d) { return d.source.fisheye.x; })

// .attr("y1", function(d) { return d.source.fisheye.y; })

// .attr("x2", function(d) { return d.target.fisheye.x; })

// .attr("y2", function(d) { return d.target.fisheye.y; });

// });

function tick(){

circleGroup

.attr("transform", node=>{

return `translate(${[node.x, node.y]})`

});

lines

.attr("x1", link => link.source.x)

.attr("y1", link => link.source.y)

.attr("x2", link => link.target.x)

.attr("y2", link => link.target.y)

}

}

d3.csv("https://gist.githubusercontent.com/timelyportfolio/5049980/raw/66a239b4aa325c05c7a19bd96bf093632591e809/les\_mis.csv")

// d3.csv("les\_mis.csv")

.then(data => {

drawNetwork(d3.select("#fig-3"),data);

})

</script>

</body>

</html>