

대구광역시 코로나19 확진자수 현황

이종운



목차

1. 소개
2. 목적
3. 사용 데이터
4. 소스 설명
5. 구현 화면

소개

대구광역시 코로나19 확진자수 현황

본 웹페이지는 대구에서 처음 코로나19 확진자가 발생한 2020년 2월부터 2023년 1월까지 약 3년에 걸친 대구광역시 소재의 신규 코로나19 확진자수를 일별로 조회할 수 있는 사이트입니다.

웹페이지에서 조회 희망 월을 선택하면, 해당 월의 신규 확진자수를 조회할 수 있습니다. 조회된 자료는 일반적인 표와 함께 표현됩니다.

또한 전반적인 추세의 변화를 손쉽게 파악할 수 있는 꺾은선, 막대 그래프와 자료의 직관적인 파악을 도와주는 트리맵, 원형 그래프도 함께 표현됩니다.

“

지혜의 말

역사는 반복되지
않지만, 패턴은
유사하다

클라우드 와이즈

”

목적

왜 알아야 하는가

이 페이지를 통해 우리는 전 세계적으로 많은 영향을 주었던 코로나19가 지역사회 내에서 확산되는 추세와 패턴을 손쉽게 파악할 수 있습니다.

이를 통해 개인으로 하여금 지속적으로 감염성 질병에 대한 인식을 환기할 수 있습니다.

또한 추후 발생 가능한 새로운 대규모 감염성 질병에 대응하기 위한 참고 자료로 활용할 수 있습니다.

이러한 자료를 계속적으로 분석하고 이를 교훈으로 삼아 대응책을 강화함으로써, 비슷한 상황에서 빠르고 효과적으로 대응하는 데 도움이 될 것입니다.



사용 데이터

: 공공데이터포털 활용



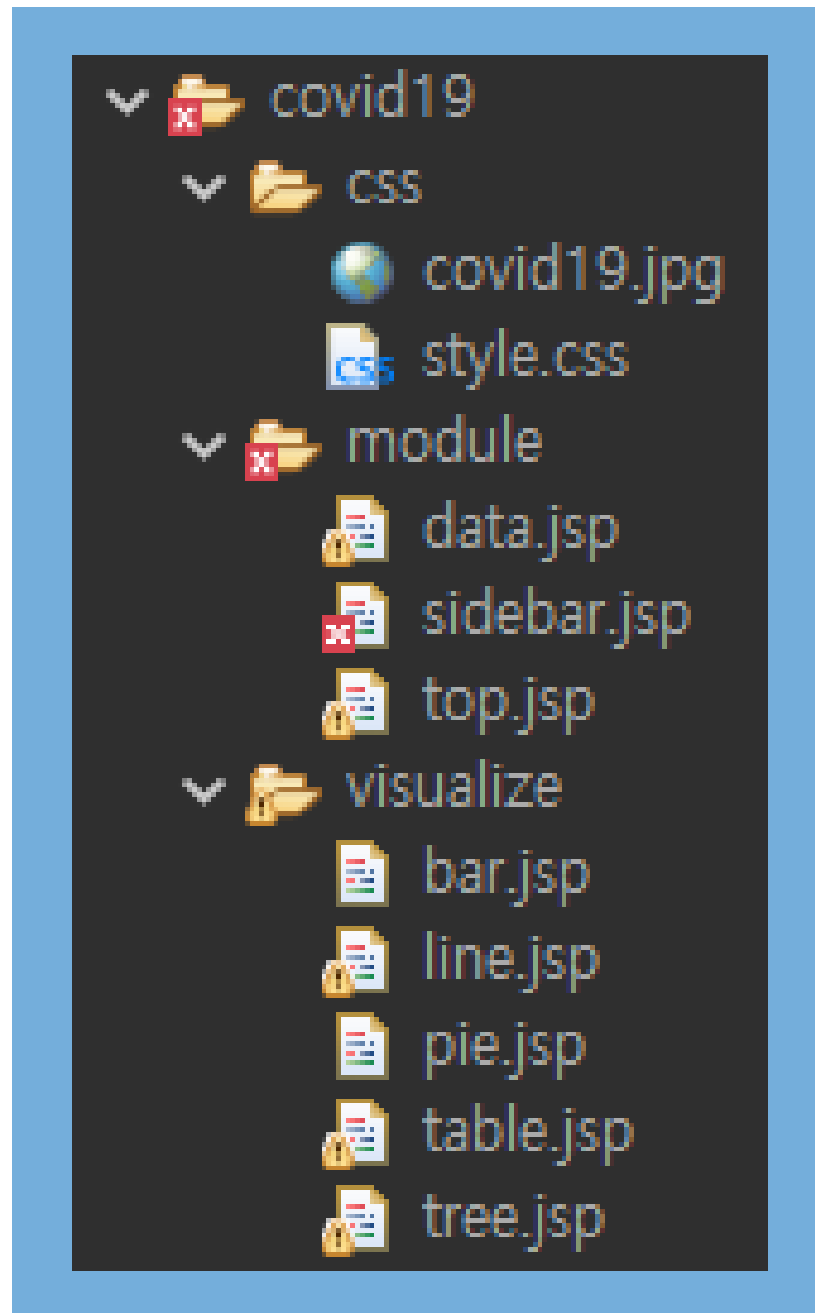
본 웹페이지에는

공공데이터포털(<https://www.data.go.kr/index.do>)에서 제공하는 대구광역시_일일
코로나19 확진자수.csv 자료가 사용되었습니다.

본 데이터에는 대구광역시에서 첫 확진자가 발생한 2020년 2월(2월 18일)부터 2023년
1월까지의 대구광역시 소재의 코로나19 신규 확진자수 현황에 대해 다루고 있습니다.

대구광역시 보건의료정책과 등록(2023-02-17)

소스 설명



1. covid19.jpg

: 웹페이지 상단에 노출되는 배너의 이미지 파일입니다.

2. table, line, bar, tree, pie.jsp

: 실제 작동되는 jsp 페이지입니다. style.css, top, data, sidebar.jsp를 import하여 사용합니다.

3. style.css

: covid19.jsp에서 사용되는 간단한 style 정보를 저장하는 파일입니다.

4. top.jsp

: covid19.jsp에서 사용되는 조회 희망 월 선택에 필요한 form 정보를 저장하는 파일입니다.

5. data.jsp

: 데이터베이스에서 데이터를 불러오는데 필요한 코드를 저장하는 파일입니다.

6. sidebar.jsp

: 조회 희망 시각화 자료형 선택에 필요한 form 정보를 저장하는 파일입니다.

소스 설명

```
14 <link rel="stylesheet" type="text/css" href="style.css">
15 <script src="https://d3js.org/d3.v7.min.js"></script>
```

```
24 <%@ include file="top.jsp" %>
```

```
15 <form method="post" action="#">
16 <h1>조회 희망 월을 지정하세요</h1>
17 <select class="button" name="month">
```

```
38 if(request.getParameter("month") != null) {
39     month = request.getParameter("month"); // 조회 희망 월 top 에서 받아옴
40 } else {
41     month = "20-02"; // 기본값은 20년 2월 조회
42 }
```

```
73 pstmt = conn.prepareStatement(
74     "select * from covid19 where date between '20" + month + "-01' and '20" + month + "-31' order by date");
75 rs = pstmt.executeQuery();
```

style.css를 link 태그를 이용하여 사용합니다.
또한 기본적으로 d3를 이용하여 데이터를 시각화합니다.

top.jsp를 include 액션태그를 이용하여 사용합니다.

top.jsp에서 form 태그를 이용하여 희망 월을 입력받으면, 입력 값에 따라 data.jsp에서 해당 월의 데이터를 받아오고, 그 값을 이용하여 각각의 jsp 파일에서 타입에 맞게 시각화 합니다.

소스 설명

```
122 <!-- 꺾은선 그래프 -->
123•<script>
124 <%
125   out.println("var dataset = ["); // 브라우저 쪽으로 <script>안에 들어갈 코드를 출력해줌
126   for (int i = 0; i < date.length - 1; i++) {
127     if (date[i] != null) {
128       out.println("{ \"name\": \"" + date[i] + "\", \"value\": " + num[i] + " },");
129     }
130   }
131
132   // 마지막 요소인 경우에는 실패를 출력하지 않음 + null인 경우에는 마지막 요소 날려버림
133   if (date[date.length - 1] != null) {
134     out.println("{ \"name\": \"" + date[date.length - 1] + "\", \"value\": " + num[num.length - 1] + " }");
135   }
136   out.println("]");
137 %>
138
139 var width = 2500; // 그래프 넓이
140 var height = 800; // 그래프 높이
141 var padding = 50; // 스케일 표시용 여백
142
143 // 2. SVG 영역 설정
144 var svg = d3.select("body").append("svg").attr("width", width).attr(
145   "height", height);
146
147 // 3. 축 스케일(눈금) 설정
148 var xScale = d3.scaleBand()
149   .rangeRound([padding, width - padding])
150   .padding(0.5)
151   .domain(dataset.map(function (d) { return d.name; }));
152
153 var yScale = d3.scaleLinear()
154   .domain([0, d3.max(dataset, function (d) { return d.value; })])
155   .range([height - padding, padding]);
156
157 // 4. 축 표시
158 svg.append("g").attr("transform",
159   "translate(0," + yScale(0) + ")").call(
160   d3.axisBottom(xScale));
161
162 svg.append("g").attr("transform",
163   "translate(" + padding + "," + 0 + ")").call(
164   d3.axisLeft(yScale));
165
```

▲ 꺾은선 그래프 코드

```
166 // 5. 선 그래프 표시
167 svg.append("path")
168   .datum(dataset)
169   .attr("fill", "none")
170   .attr("stroke", "#add8e6")
171   .attr("stroke-width", 2)
172   .attr("d", d3.line()
173     .x(function(d) { return xScale(d.name) + xScale.bandwidth() / 2; })
174     .y(function(d) { return yScale(d.value); })
175   );
176
177 // 6. 각 점 표시
178 svg.selectAll("circle").data(dataset).enter().append("circle")
179   .attr("cx", function(d) {
180     return xScale(d.name) + xScale.bandwidth() / 2;
181   })
182   .attr("cy", function(d) {
183     return yScale(d.value);
184   })
185   .attr("r", 5) // 원의 반지름 설정
186   .attr("fill", "#add8e6");
187
188 // 7. 각 점 위에 확진자수 값 텍스트 표시
189 svg.selectAll("text.value").data(dataset).enter().append("text")
190   .attr("class", "value")
191   .attr("x", function(d) {
192     return xScale(d.name) + xScale.bandwidth() / 2;
193   })
194   .attr("y", function(d) {
195     return d.value >= 0 ? yScale(d.value) - 10 : yScale(d.value) + 20;
196   })
197   .attr("text-anchor", "middle")
198   .attr("alignment-baseline", function(d) {
199     return d.value >= 0 ? "baseline" : "hanging";
200   })
201   .text(function(d) {
202     return d.value + "명"; // 확진자수 값 표시
203   });
204 </script>
205 <h2>▲ 대구 지역 코로나 19 확진자 현황 - 꺾은선</h2>
```

▲ 꺾은선 그래프 코드

소스 설명

```
208 <!-- 막대 그래프 -->
209<script>
210 <%
211     out.println("var dataset = []"); // 브라우저 쪽으로 <script>안에 들어갈 코드를 출력해줌
212     for (int i = 0; i < date.length - 1; i++) {
213         if (date[i] != null) {
214             out.println("{ \"name\": \"" + date[i] + "\", \"value\": " + num[i] + " },");
215         }
216     }
217
218     // 마지막 요소인 경우에는 샘플을 출력하지 않음 + null인 경우에는 마지막 요소 날려버림
219     if (date[date.length - 1] != null) {
220         out.println("{ \"name\": \"" + date[date.length - 1] + "\", \"value\": " + num[num.length - 1] + " }");
221     }
222     out.println("]");
223 %>
224
225     var width = 2500; // 그래프 넓이
226     var height = 800; // 그래프 높이
227     var padding = 50; // 스케일 표시용 여백
228
229     // 2. SVG 영역 설정
230     var svg = d3.select("body").append("svg").attr("width", width).attr(
231         "height", height);
232
233     // 3. 축 스케일(눈금) 설정
234     var xScale = d3.scaleBand()
235         .rangeRound([padding, width - padding])
236         .padding(0.5)
237         .domain(dataset.map(function (d) { return d.name; }));
238
239     var yScale = d3.scaleLinear()
240         .domain([0, d3.max(dataset, function (d) { return d.value; })])
241         .range([height - padding, padding]);
242
243     // 4. 축 표시
244     svg.append("g").attr("transform",
245         "translate(0," + yScale(0) + ")").call(
246         d3.axisBottom(xScale));
247
248     svg.append("g").attr("transform",
249         "translate(" + padding + "," + 0 + ")").call(
250         d3.axisLeft(yScale));
251
```

▲ 막대 그래프 코드

```
253     // 5. 막대 표시
254     svg.append("g").selectAll("rect").data(dataset).enter().append("rect")
255         .attr("x", function(d) {
256             return xScale(d.name);
257         }).attr("y", function(d) {
258             return yScale(d.value);
259         }).attr("width", xScale.bandwidth()).attr("height",
260             function(d) {
261                 return height - padding - yScale(d.value);
262             }).attr("fill", "#add8e6");
263
264     // 6. 각 막대 위에 확진자수 값 텍스트 표시
265     svg.selectAll("text.value").data(dataset).enter().append("text")
266         .attr("class", "value")
267         .attr("x", function(d) {
268             return xScale(d.name) + xScale.bandwidth() / 2;
269         })
270         .attr("y", function(d) {
271             return d.value >= 0 ? yScale(d.value) - 10 : yScale(d.value) + 20;
272         })
273         .attr("text-anchor", "middle")
274         .attr("alignment-baseline", function(d) {
275             return d.value >= 0 ? "baseline" : "hanging";
276         })
277         .text(function(d) {
278             return d.value + "명"; // 확진자수 값 표시
279         });
280 </script>
281 <h2>▲ 대구 지역 코로나 19 확진자 현황 - 막대</h2>
```

▲ 막대 그래프 코드

소스 설명

```
284 <!-- 트리맵 -->
285<script>
286 // 트리맵에 사용할 데이터 예시
287 var data = {
288   name: "Root",
289 <%
290 out.println("children: ["); // 브라우저 쪽으로 <script>안에 들어갈 코드를 출력해줌
291 for (int i = 0; i < date.length - 1; i++) {
292   if (date[i] != null && num[i] != 0) {
293     out.println("{ name: \"\" + date[i] + "\", value: " + num[i] + " },");
294   }
295 }
296
297 // 마지막 요소인 경우에는 싼표를 출력하지 않음
298 if (date[date.length - 1] != null && num[num.length - 1] != 0) {
299   out.println("{ name: \"\" + date[date.length - 1] + "\", value: " + num[num.length - 1] + " }");
300 }
301 out.println("]");
302 %>
303 };
304
305 // 트리맵의 크기 설정
306 var width = 2500;
307 var height = 1800;
308
309 // D3.js의 트리맵 레이아웃 생성
310 var treemap = d3.treemap().size([width, height]);
311
312 // 데이터를 트리맵에 맞게 변형
313 var root = d3.hierarchy(data).sum(function(d) {
314   return d.value;
315 });
316
317 // 트리맵 레이아웃에 데이터 적용
318 treemap(root);
319
320 // SVG 요소 생성
321 var svg = d3.select("body").append("svg")
322   .attr("width", width)
323   .attr("height", height);
324
325 // 파스텔톤으로 연한 블루색상을 만드는 함수
326 function getPastelBlue(t) {
327   return d3.interpolate("lightblue", "steelblue")(t);
328 }
```

▲ 트리맵 코드

```
330 // 트리맵의 각 사각형을 그리기
331 svg.selectAll("rect")
332   .data(root.leaves()) // 말단 노드만 선택
333   .enter().append("rect")
334   .attr("x", function(d) { return d.x0; })
335   .attr("y", function(d) { return d.y0; })
336   .attr("width", function(d) { return d.x1 - d.x0; })
337   .attr("height", function(d) { return d.y1 - d.y0; })
338   .style("fill", function(d) {
339     // 크기에 따라 파스텔 블루색상 적용
340     return getPastelBlue(d.value / d3.max(root.leaves(), function(d) { return d.value; }));
341   })
342   .style("stroke", "white");
343
344 // 각 사각형에 텍스트 추가
345 svg.selectAll("text")
346   .data(root.leaves())
347   .enter().append("text")
348   .attr("x", function(d) { return d.x0 + 5; })
349   .attr("y", function(d) { return d.y0 + 15; })
350   .text(function(d) { return d.data.name + ": " + d.data.value + "명"; })
351   .attr("font-size", "15px")
352   .attr("fill", "white");
353 </script>
354 <h2>▲ 대구 지역 코로나 19 확진자 현황 - 트리맵</h2>
```

▲ 트리맵 코드

소스 설명

```
357 <!-- 원형 그래프 -->
358<script>
359 <%
360   out.println("var dataset = ["); // 브라우저 쪽으로 <script>안에 들어갈 코드를 출력해줌
361   for (int i = 0; i < date.length - 1; i++) {
362       if (date[i] != null && num[i] != 0) {
363           out.println("{ \"name\": \"" + date[i] + "\", \"value\": " + num[i] + " },");
364       }
365   }
366
367   // 마지막 요소인 경우 + 0인 경우에는 실패를 출력하지 않음 + null인 경우에는 마지막 요소 날려버림
368   if (date[date.length - 1] != null && num[num.length - 1] != 0) {
369       out.println("{ \"name\": \"" + date[date.length - 1] + "\", \"value\": " + num[num.length - 1] + " }");
370   }
371   out.println("]");
372 %>
373
374 var width = 2500; // 그래프 넓이
375 var height = 2500; // 그래프 높이
376 var radius = Math.min(width, height) / 2 - 10;
377
378 // 2. SVG 영역 설정
379 var svg = d3.select("body").append("svg").attr("width", width).attr("height", height);
380
381 var g = svg.append("g").attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");
382
383 //3. 컬러 설정
384 var color = d3.scaleOrdinal()
385     .range(d3.schemePastel1); // d3.schemePastel1은 푸른 계열의 파스텔톤 컬러 스킴
386
387 // 4. pie 차트 dataset에 대한 함수 설정
388 var pie = d3.pie()
389     .value(function(d) { return d.value; })
390     .sort(null);
```

▲ 원형 그래프 코드

```
392 // 5. pie 차트 SVG 요소 설정
393 var pieGroup = g.selectAll(".pie")
394     .data(pie(dataset))
395     .enter()
396     .append("g")
397     .attr("class", "pie");
398
399 arc = d3.arc()
400     .outerRadius(radius)
401     .innerRadius(0);
402
403 pieGroup.append("path")
404     .attr("d", arc)
405     .attr("fill", function(d) { return color(d.index) })
406     .attr("opacity", 0.75)
407     .attr("stroke", "white");
408
409 // 6. pie 차트 텍스트 SVG 요소 설정
410 var text = d3.arc()
411     .outerRadius(radius - 30)
412     .innerRadius(radius - 30);
413
414 pieGroup.append("text")
415     .attr("fill", "black")
416     .attr("transform", function(d) {
417         var pos = text.centroid(d);
418         // 글자 회전 설정 추가
419         return "translate(" + pos + ") rotate(" + angle(d) + ")";
420     })
421     .attr("dy", "3") // dy 값을 조절하여 텍스트 위치 조절
422     .attr("font-size", "20px") // 글자 크기 조절
423     .attr("text-anchor", "start")
424     .text(function(d) { return d.data.name + ": " + d.data.value + "명";});
425
426 // 7. 회전 각도 계산 함수 추가
427 function angle(d) {
428     var a = (d.startAngle + d.endAngle) * 90 / Math.PI - 90;
429     return a - 180;
430 }
431 </script>
432 <h2>▲ 대구 지역 코로나 19 확진자 현황 - 원</h2>
433
```

▲ 원형 그래프 코드

구현화면

웹페이지의 전체적인 화면입니다.

상단 배너 아래의 입력창에서 조회 희망 월을 선택하면 해당 월의 자료를 기반으로 한 데이터가 조회됩니다.

시각화된 자료는 표, 꺾은선 그래프, 막대 그래프, 트리맵, 원형 그래프를 선택하여 조회할 수 있습니다.

월을 선택하지 않으면, 기본적으로 가장 첫번째 월인 2020년 2월의 자료가 조회, 출력됩니다.



대구 코로나19 확진자 현황

조회 희망 월을 지정하세요

선택하세요

조회

2020-02월 자료

월별 전체 확진자 : 2055명

표

꺾은선

막대

트리

원

날짜	확진자수(명)
2020-02-01	0
2020-02-02	0
2020-02-03	0
2020-02-04	0
2020-02-05	0
2020-02-06	0
2020-02-07	0
2020-02-08	0
2020-02-09	0

조회 희망 월을 지정하세요

선택하세요

조회

자료

: 2055명

표

꺾은선

막대

트리

원

날짜	확진자수(명)
2020-02-01	0
2020-02-02	0
2020-02-03	0
2020-02-04	0
2020-02-05	0
2020-02-06	0
2020-02-07	0
2020-02-08	0
2020-02-09	0

입력 창을 통하여 조회 희망 월을 선택하고,
조회 버튼을 클릭하면 조회가 됩니다.

조회 희망 월을 지정하세요

선택하세요 ▼

2020년

2월

3월

4월

5월

6월

7월

8월

9월

10월

11월

12월

2021년

1월

2월

3월

4월

5월

6월

7월

조회

표

트리

원

날짜	확진자수(명)
2020-02-01	0
2020-02-02	0
2020-02-03	0
2020-02-04	0
2020-02-05	0
2020-02-06	0
2020-02-07	0
2020-02-08	0
2020-02-09	0
2020-02-10	0

자료

: 2055명

2020년 6월 선택 후 조회 클릭 시
2020-06월 자료로 조회가 진행됩니다.

조회 희망 월을 지정하세요

선택하세요 ▼

조회

2020-06월 자료

월별 전체 확진자 : 23명

표

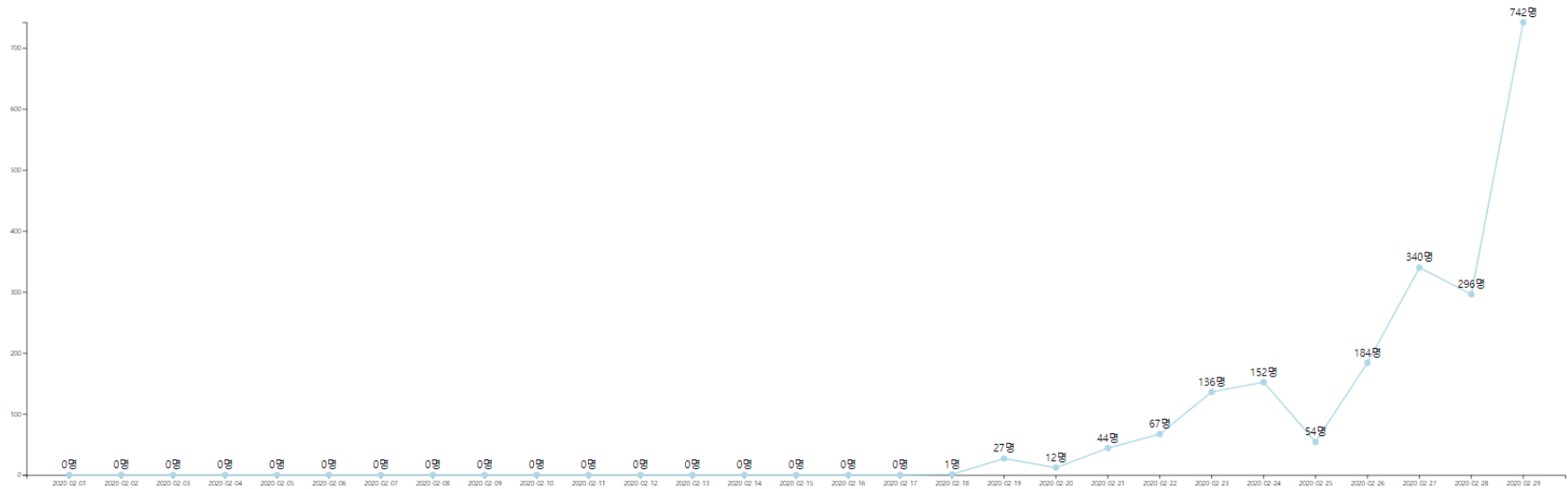
꺾은선

막대

트리

원

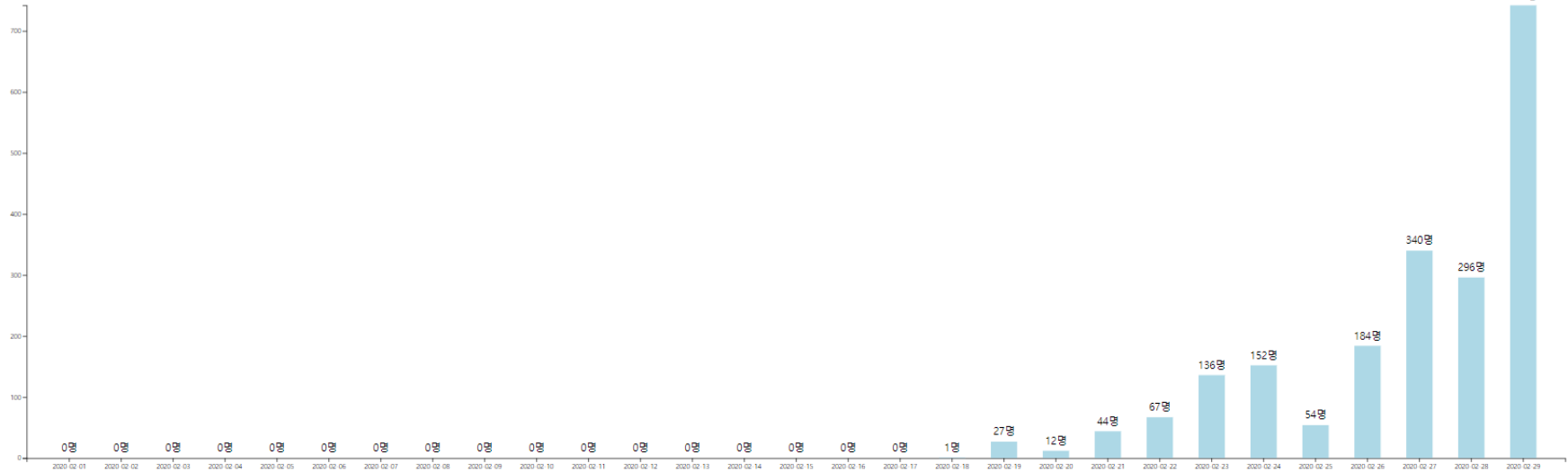
날짜	확진자수(명)
2020-06-01	1
2020-06-02	0
2020-06-03	1
2020-06-04	0
2020-06-05	1
2020-06-06	0
2020-06-07	1
2020-06-08	1
2020-06-09	0
2020-06-10	0
2020-06-11	0
2020-06-12	1
2020-06-13	3
2020-06-14	2



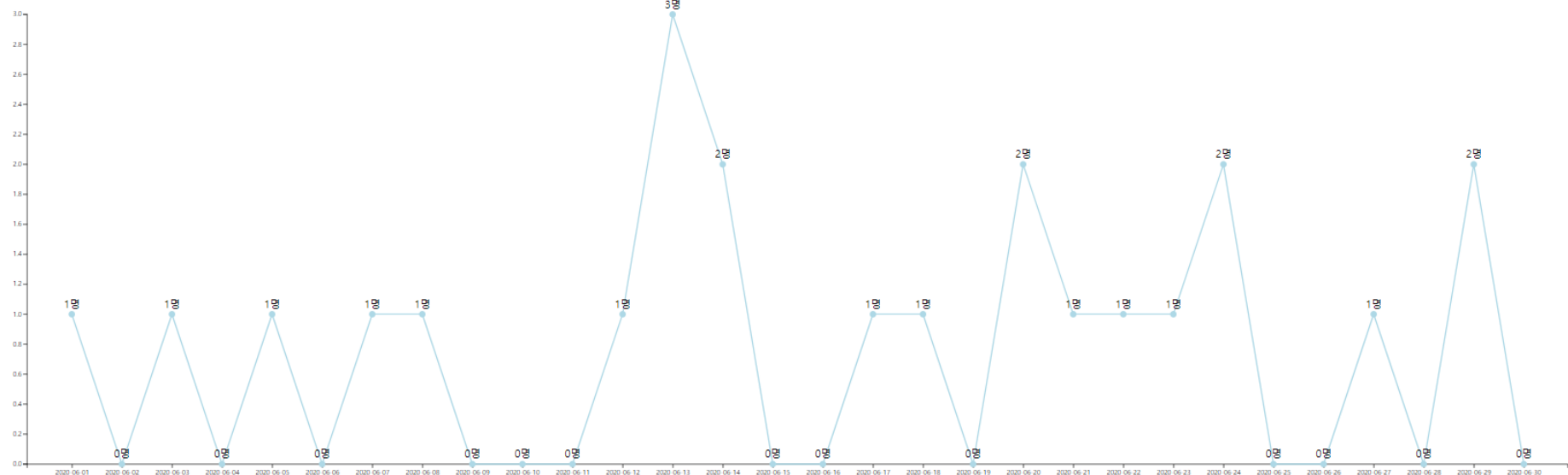
▲ 대구 지역 코로나 19 확진자 현황 - 꺾은선



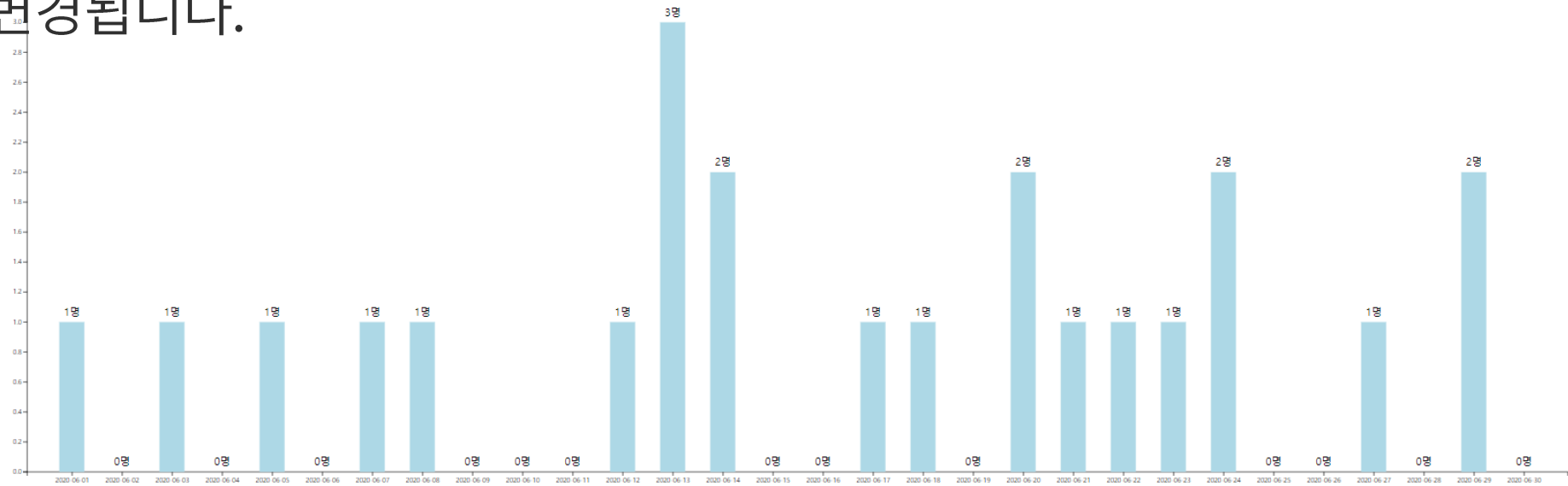
시각화 자료도 변경됩니다.



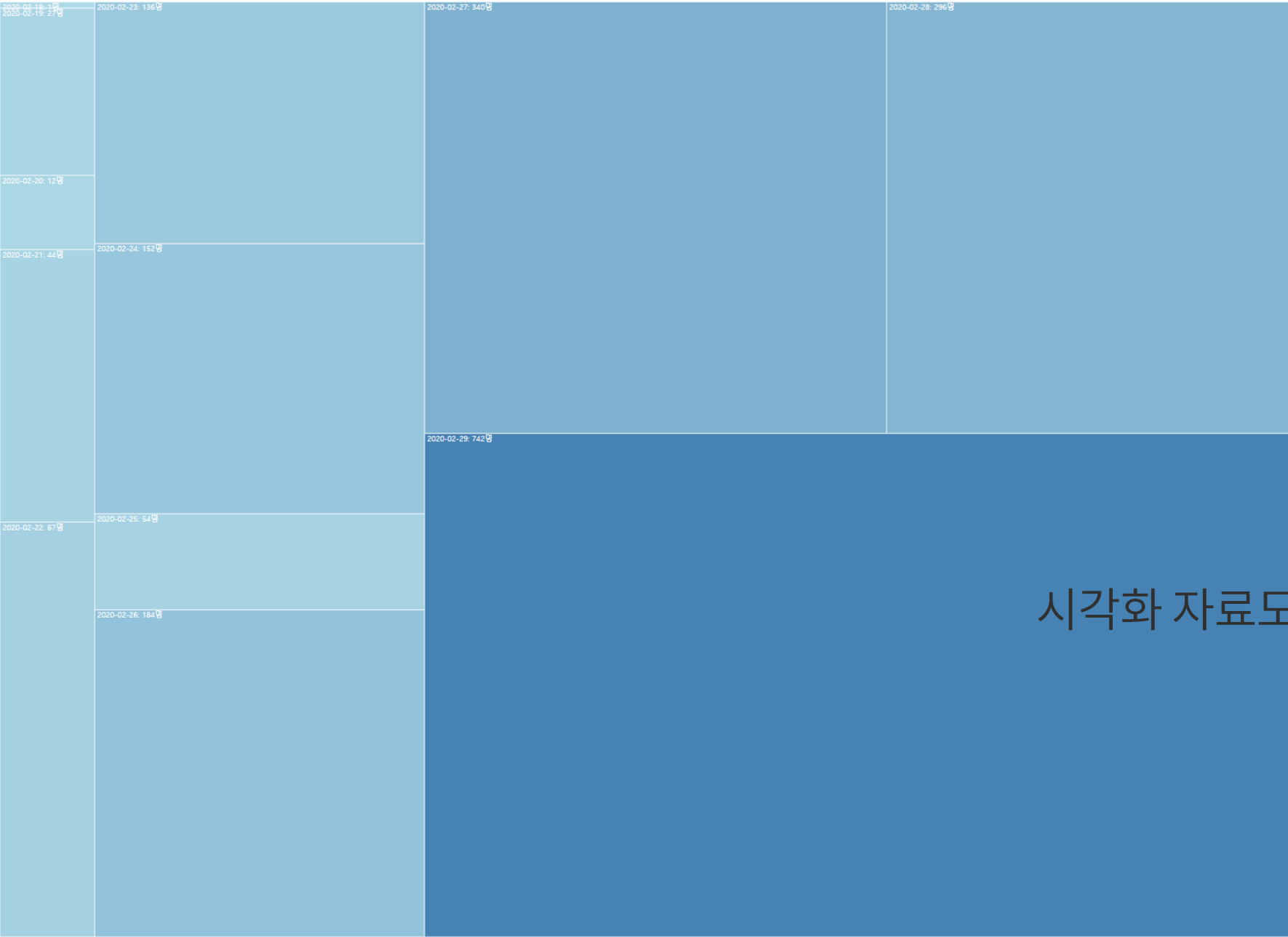
▲ 대구 지역 코로나 19 확진자 현황 - 막대



▲ 대구 지역 코로나 19 확진자 현황 - 꺾은선



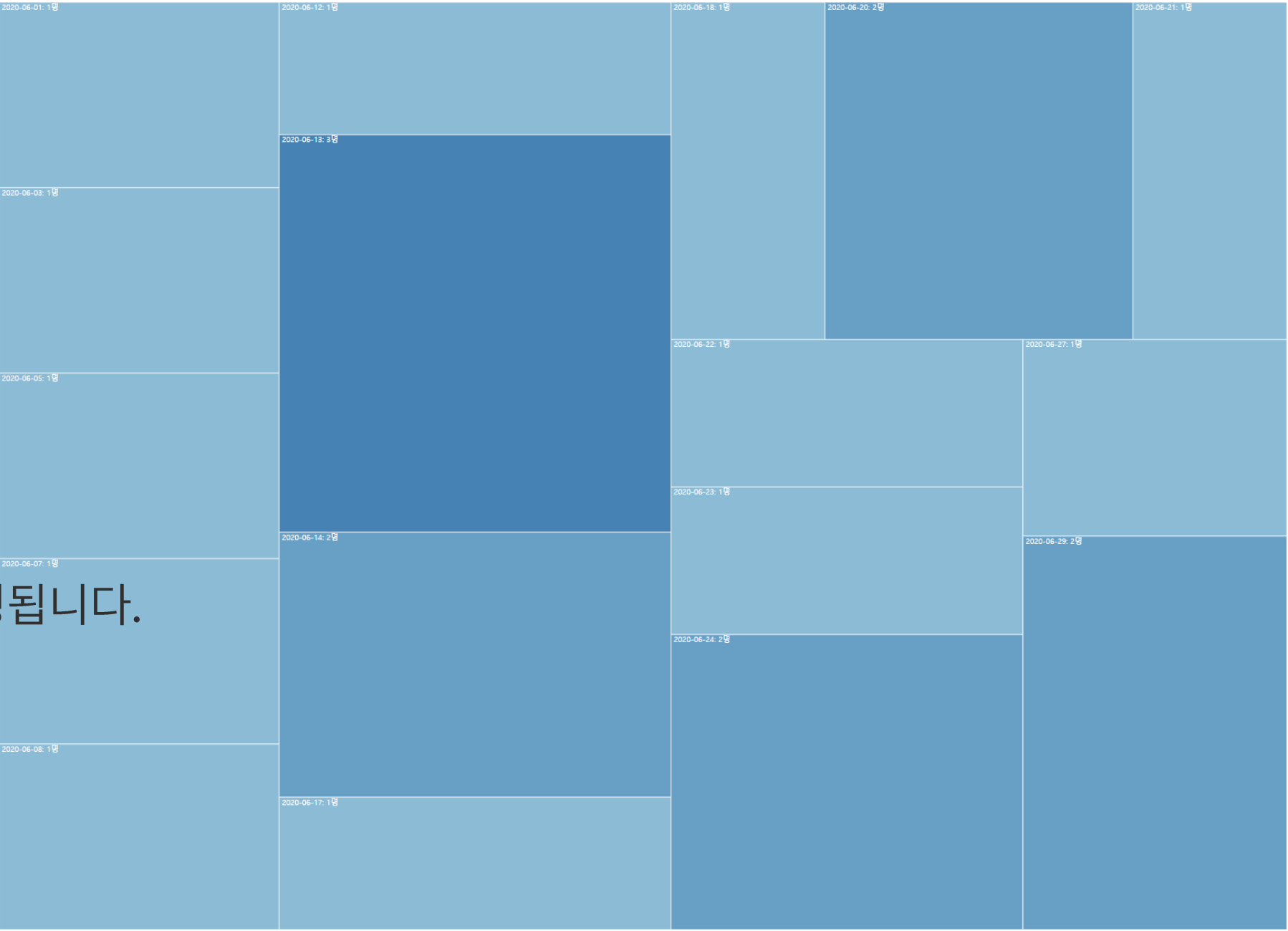
▲ 대구 지역 코로나 19 확진자 현황 - 막대



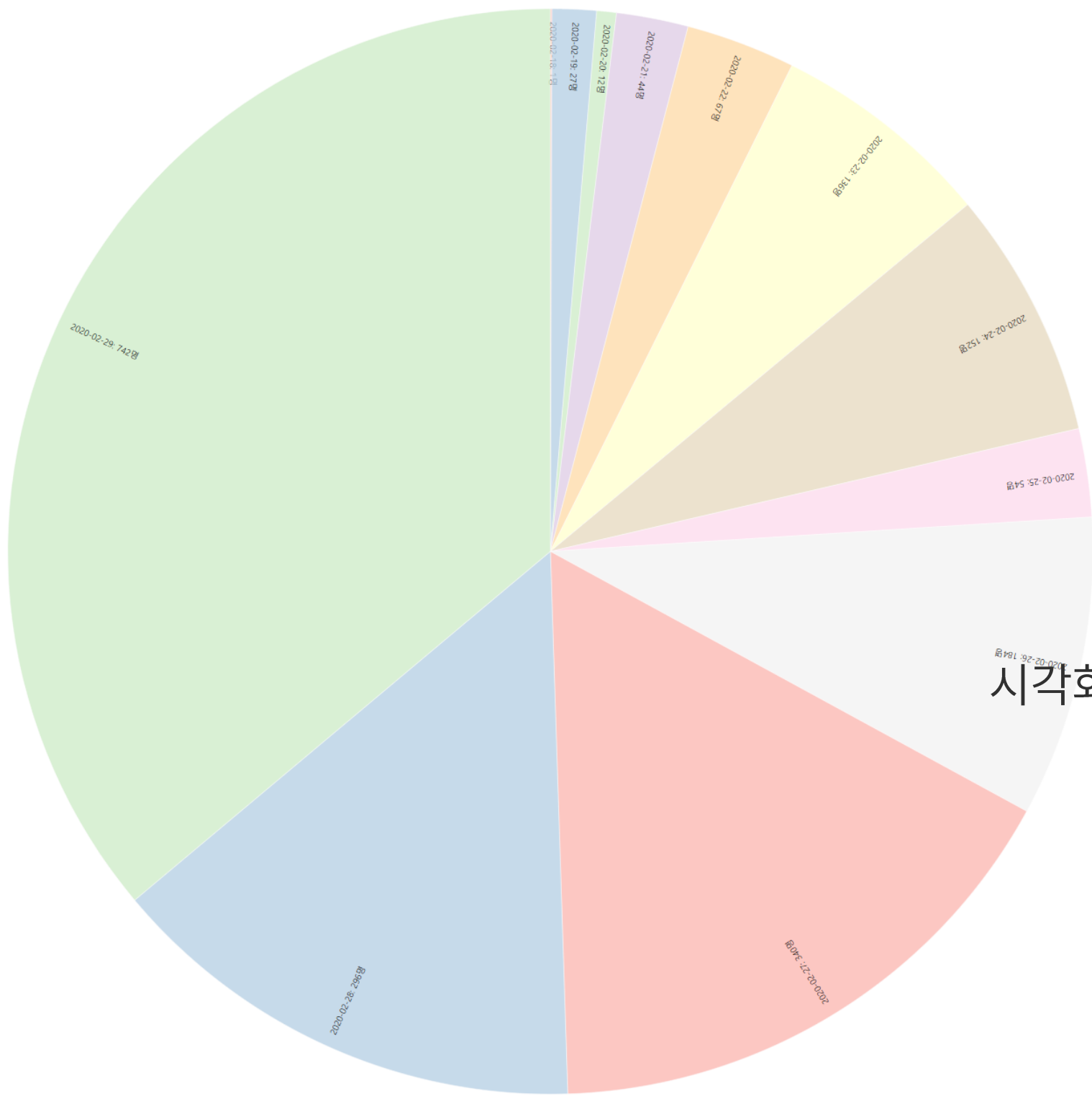
▲ 대구 지역 코로나 19 확진자 현황 - 트리맵



시각화 자료도 변경됩니다.



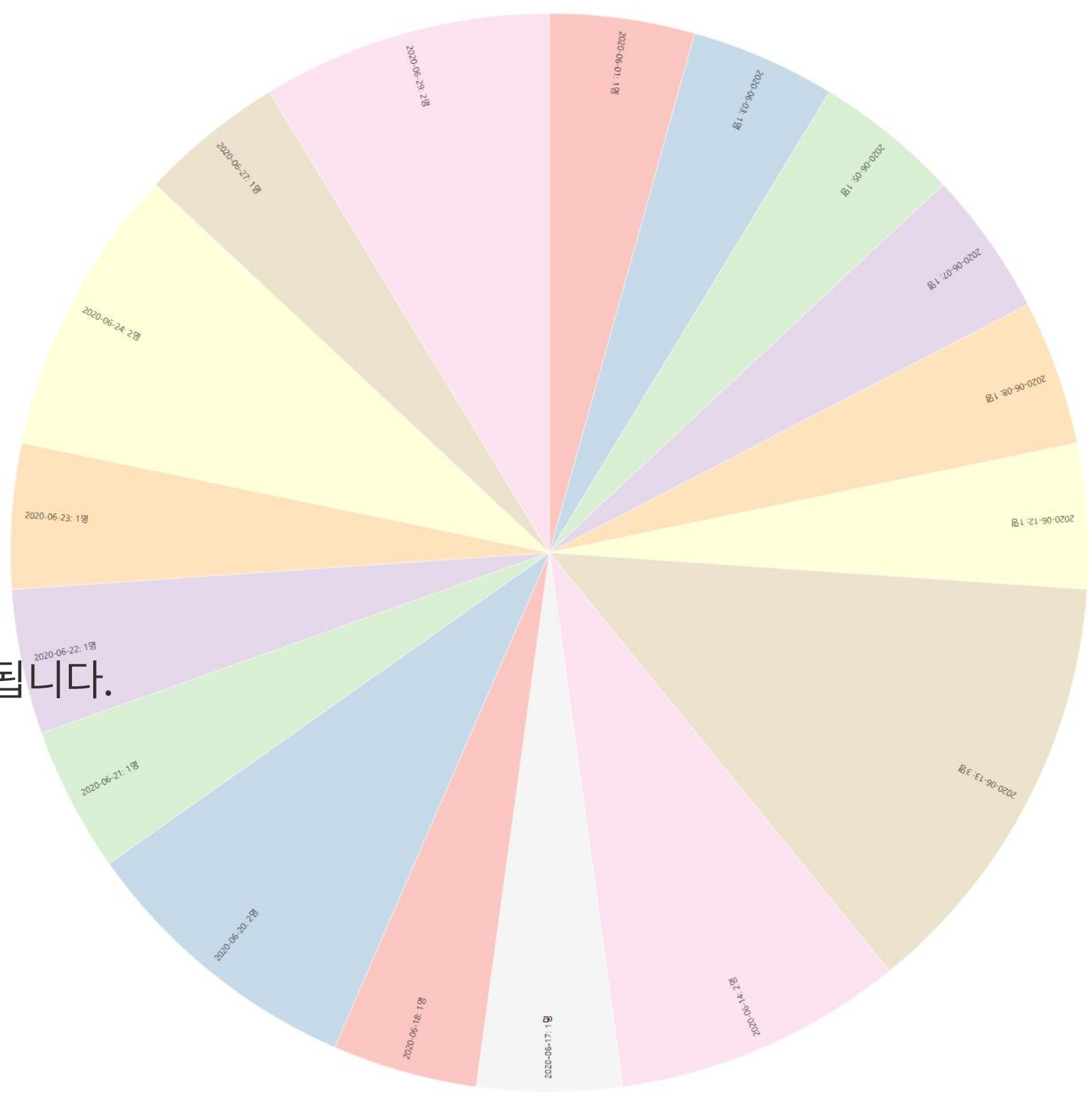
▲ 대구 지역 코로나 19 확진자 현황 - 트리맵



▲ 대구 지역 코로나 19 확진자 현황 - 원



시각화 자료도 변경됩니다.



▲ 대구 지역 코로나 19 확진자 현황 - 원



감사합니다.