데이터베이스 최종 보고서

제목: **COVID-19 알림이**

제출일:   2021년 12월 08일

1반 9조

팀 원: 201820802 강선규

201823780 정석화

201720738 주정호

201621886 김영희

목차

[1. Introduction 3](#_Toc89950343)

[1.1 개발 목표 3](#_Toc89950344)

[1.2 주제선정 배경 및 기대 효과 3](#_Toc89950345)

[2. Requirements 4](#_Toc89950346)

[2.1 주요 기능 5](#_Toc89950347)

[2.2 System Structure 5](#_Toc89950348)

[2.3 Description of Mini world 6](#_Toc89950349)

[3. Data Schema 6](#_Toc89950350)

[3.1 수정 사항 6](#_Toc89950351)

[3.2 ER Diagram 6](#_Toc89950352)

[3.3 Data Table 8](#_Toc89950353)

[3.4 Constraints 11](#_Toc89950354)

[4. DB Result : Relational Model 12](#_Toc89950355)

[5. Front Result : WEB Application 12](#_Toc89950356)

[6. Crawling Data & DevOps 13](#_Toc89950357)

[6.1 Crawling Data 13](#_Toc89950358)

[6.2 DevOps 13](#_Toc89950359)

[7. 참고자료 14](#_Toc89950360)

## **1. Introduction**

# **1.1 개발 목표**

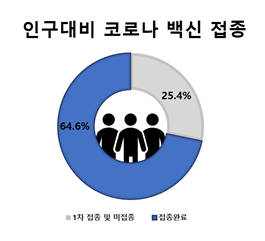
코로나19 백신 예방 접종이 전연령층을 대상으로 진행되고 있다. 본 프로젝트는 코로나 확진 정보에 백신 접종 정보를 추가해 통합 데이터베이스를 구축하는 것을 목표로 한다. 따라서 지역별, 날짜별로 사용자가 빠르게 정보를 확인할 수 있도록 돕는다.

# **1.2 주제선정 배경 및 기대 효과**

* **주제선정 배경**
* **백신접종률 증가에 따른 백신 접종에 관한 관심증대**

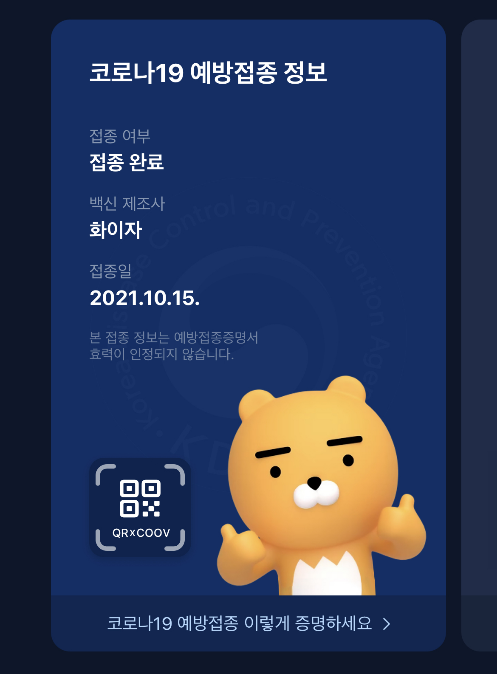
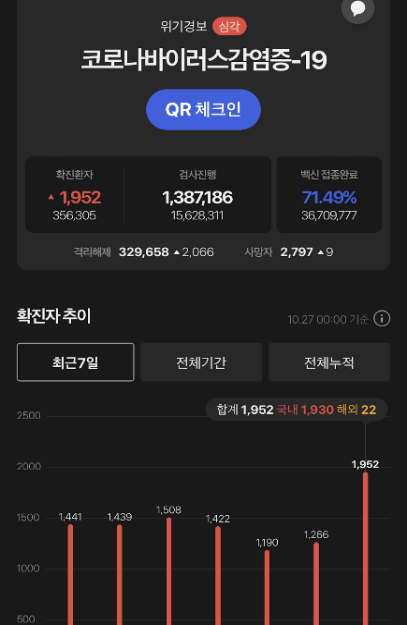
코로나 예방 백신 접종률이 10월 14일 기준 인구대비 60%를 넘어서면서 2019년부터 시작된 코로나19 사태가 새로운 ‘with코로나’시대를 맞이하고 있다.

백신 접종에 대한 관심과 참여로 접종완료율은 먼저 접종을 시작한 프랑스 67.4%, 영국 66.7%, 이스라엘 65.0% 등의 국가들과 유사하다. 또한 백신관련 정보를 전달하고 있는 국민 비서 ‘구삐’의 가입자는 9월이후 1,300만명을 넘어섰다. 백신 접종률 증가와 “with코로나”로 인한 방역 정책변화로 코로나 및 백신에 대한 관심은 높아지고 있다.



* **코로나19 유사 앱**

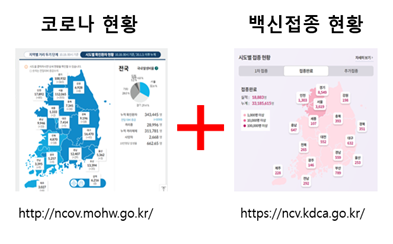
기존의 코로나19 관련 알림 앱들은 백신 종류별 접종자 수 까지 세세하게  제공하고 있지는 않았다.  본 프로젝트에서는 상세한 백신 접종 정보를 날짜별로 시각화할 예정이다.

* **기대효과**

**분리된 정보로 인한 불편함 해결**

기존 코로나 사이트는 백신별 현황, 지역별 등 여러가지 데이터가 분리된 상태로 한 눈에 데이터를 파악하기 어렵다. 본 프로젝트를 통해 분리된 데이터들을 크롤링하고 취합하여 데이터베이스에 저장하고 관리할 것이다. 이로 인해 종합된 데이터를 사용자에게 시각적으로 제공할 것이다. 나아가서 사용자가 백신 접종 상태 및 신상정보를 등록함으로써 백신 접종 관리의 편리함을 높인다.



2. Requirements

# **2.1 주요 기능**

* **사용자**

1) 지역별 확진 및 접종 현황 조회

지역별 코로나 확진자 수, 백신 접종자 수, 거리두기 단계를 조회할 수 있다.

2) 날짜별 접종 정보 조회

확진 현황 및 날짜별 백신 종류별 접종자 수를 조회할 수 있다.

3) 사용자 백신 정보 등록

사용자의 신상 정보 및 백신 접종 상태를 등록할 수 있다.

* **개발자**

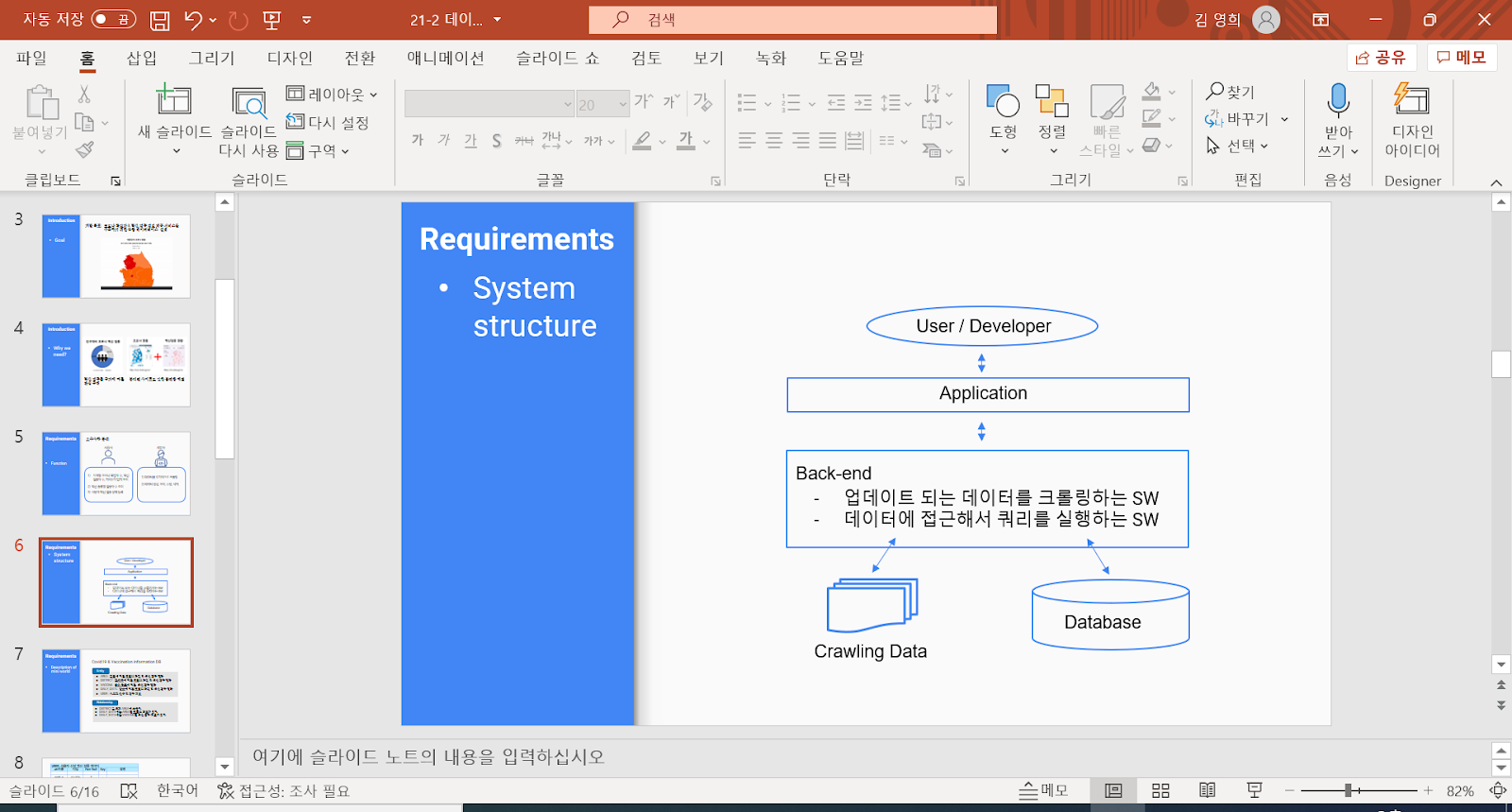
1) 데이터 크롤링

데이터를 주기적으로 크롤링할 수 있다.

2) 데이터 관리

데이터 생성, 조회, 수정, 삭제할 수 있다.

# **2.2 System Structure**



COVID-19알리미의 통합 데이터베이스 액터는 사용자와 개발자 총 2명이 있다. 사용자는 어플리케이션을 통해 3가지 주요 기능을 사용할 수 있다. 또한 개발자는 애플리케이션 서버를 통하여 2가지 주요 기능을 사용할 수 있다.

 사용자는 어플리케이션을 통해 [지역별 확진 및 접종 현황 조회, 날짜별 접종 정보 조회, 사용자 백신 정보 등록] 기능을 수행할 수 있다. DBMS는 데이터 접근 쿼리를 수행하고 응답을 전달할 수 있다.

개발자는 애플리케이션 서버를 통해 [주기적 데이터 크롤링, 데이터 관리] 기능을 수행할 수 있다.

어플리케이션 서버에서 데이터 크롤링 작업을 수행하고 모은 데이터를 데이터베이스에 업데이트 할 수 있다.  데이터베이스는 MySQL을 사용하고, 참조 사이트에서 데이터를 수집한다.

# **2.3 Description of Mini world**

* Entity

AREA : 지역에 따른 코로나 확진 및 백신 접종 현황

DISTRICT : 자치구에 따른 코로나 확진 및 백신 접종 현황

VACCINE : 백신 종류에 따른  백신 접종 현황

DAILY\_DATA : 날짜에 따른 코로나 확진 및 백신 접종 현황

SOCIAL\_DIST : 거리두기 단계에 따른 내용

USER : 사용자 신상 및 접종 정보

TODAY\_CONFIRMED : 당일 날 해당 지역의 확진자 수

HOSPTAL : 지역별 백신 접종하는 병원

* Relationship

DISTRICT와 SOCIAL\_DIST는 특정 AREA에 속한다.

DAILY\_DATA에는 AREA별 코로나 정보가 있다.

DAILY\_DATA에는 VACCINE별 백신 접종 정보가 있다.

TODAY\_AREA에는 USER의 지역의 실시간 코로나 감염 현황 정보가 있다.

# **3. Data Schema**

# **3.1 수정 사항**

본 프로젝트는 USER entity와 다른 entity사이의 관계성이 부족하다는 점, 기존 서비스와의 차이점을 보여 달라는 피드백을 받았다.

문제점을 개선하기 위해 기존 E-R Model에 relation과 entity를 추가하는 방향으로 DB 스키마 디자인을 수정했다.

또한 사용자의 지역구를 중심으로 다양한 정보를 제공한다. hospital 데이터를 크롤링 추가로 크롤링 했고, sql query를 추가해 사용자 개개인을 대상으로 한 정보를 늘렸다.

주요 개선 사항 :

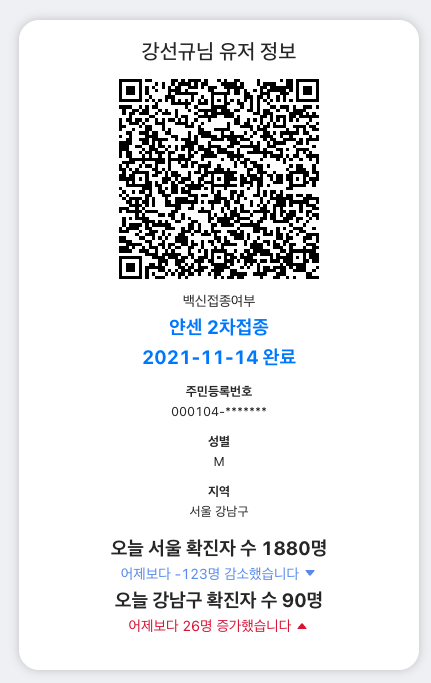
1. TODAY\_AREA 테이블을 생성해서 district, area entity들과 user entity사이의 연결관계를 생성했다. [수정 후 E-R Diagram] 참조.

e-r 관계 설명 추가

1. SQL 쿼리를 추가 :

웹 어플리케이션을 통해 사용자는 단순히 정보를 등록하는 것을 넘어서,

사용자가 속한 지역 및 자치구를 중심으로 당일 확진자 정보, 누적 확진자 정보 제공받을 수 있다. [5. Result : WEB Application 참조]

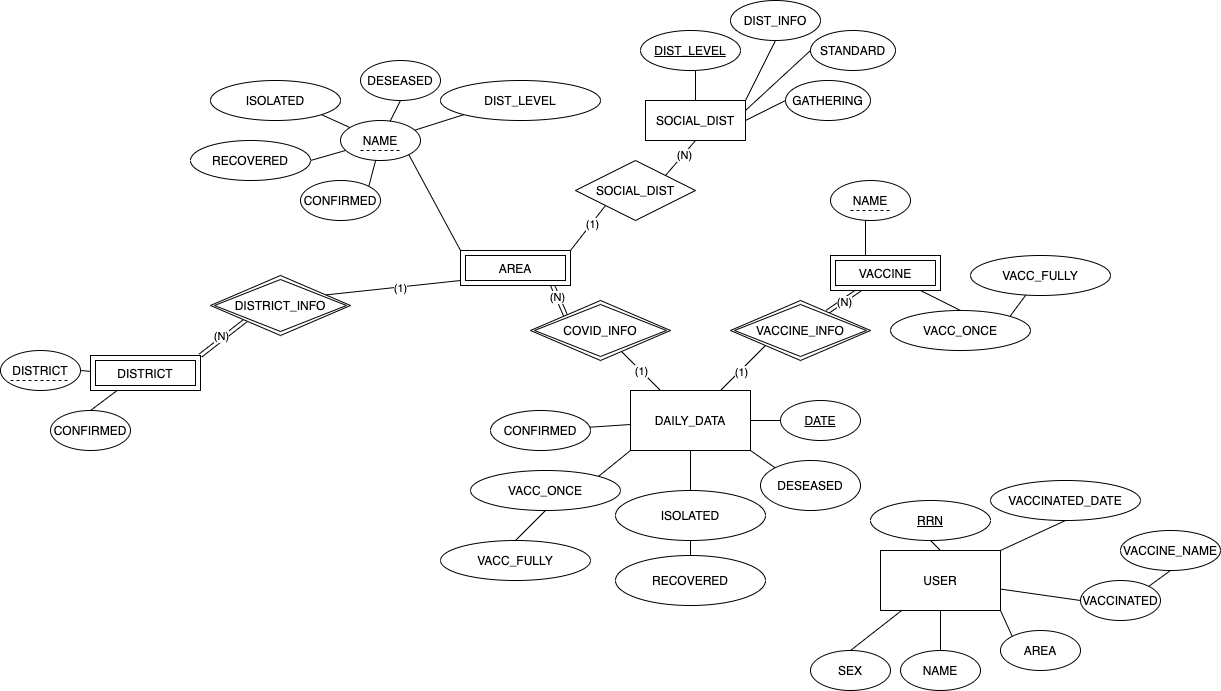


1. HOSPITAL 테이블을 생성하고 데이터를 추가적으로 크롤링 :

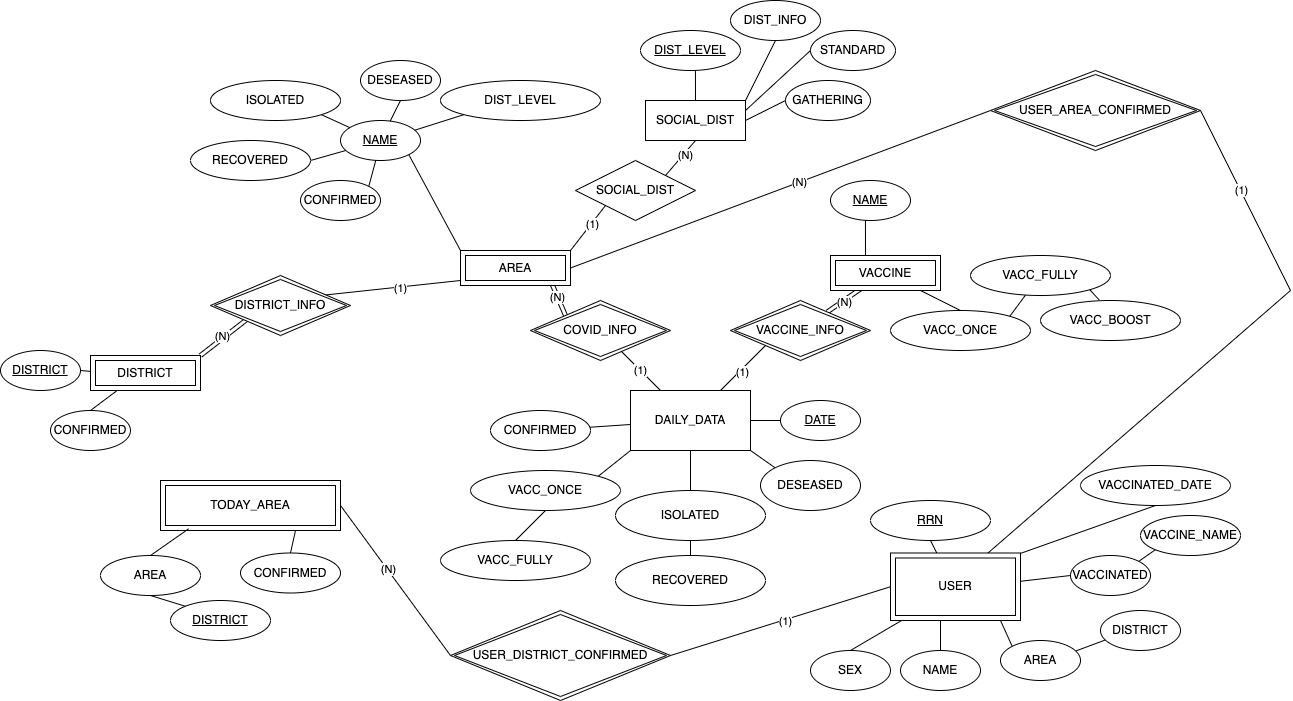
[4. 참조]

이를 통해 기존의 entity간의 약한 관계성 보완하고 어플리케이션 만의 차별성을 보여주기를 기대한다.

# **3.2 ER Diagram**

[수정 전]

[수정 후]

****

# **3.3 Data Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DAILY\_DATA Table** | | | | |
| Attribute | Data Type | Not NULL | Key | 설명 |
| DATE | DATE | Y | PK | 데이터 업데이트 날짜 |
| CONFIRMED | Int | Y | - | 확진자 인원 수 |
| ISOLATED | Int | Y | - | 격리자 인원 수 |
| DESEASED | Int | Y | - | 사망자 인원 수 |
| RECOVERED | Int | Y | - | 격리 해제 인원 수 |
| VACC\_ONCE | Int | Y | - | 1차 백신 접종 인원 수 |
| VACC\_FULLY | Int | Y | - | 2차 백신 접종 인원 수 |
| VACC\_BOOST | Int | Y | - | 추가 백신 접종 인원 수 |

* 종합적인 데이터 테이블이다. primary key는 데이터 업데이트 날짜이다.
* 모든 attriute에 not null 제약조건을 적용하기 때문에 데이터 값을 입력해야 한다.
* 추가접종이 늘어나고 있어서 ‘VACC\_BOOST’를 추가했다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **AREA Table** | | | | |
| Attribute | Data Type | Not NULL | Key | 설명 |
| DATE | DATE | Y | PK | 데이터 업데이트 날짜 |
| NAME | Varchar | Y | PK | 지역 이름 |
| CONFIRMED | Int | Y | - | 확진자 인원 수 |
| ISOLATED | Int | Y | - | 격리자 인원 수 |
| DESEASED | Int | Y | - | 사망자 인원 수 |
| RECOVERED | Int | Y | - | 격리 해제 인원 수 |
| DIST\_LEVEL | Int | N | - | 거리두기 단계 |

* 지역별 데이터 테이블이다. primary key는 데이터 업데이트 날짜와 지역 이름이다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DISTRICT Table** | | | | |
| Attribute | Data Type | Not NULL | Key | 설명 |
| DATE | DATE | Y | PK | 데이터 업데이트 날짜 |
| AREA\_NAME | Varchar | Y | PK, FK | 지역 이름 |
| DISTRICT | Varchar | Y | PK | 자치구 이름 |
| DISTRICT\_CONFIMED | Int | N | - | 자치구 확진 인원 수 |

* 자치구(구)별 데이터 테이블이다.
* primary key는 지역(시) 이름, 자치구(구) 이름, 데이터 업데이트 날짜이다. 지역 이름은 AREA 테이블의 지역 이름을 참조한 FK이면서 PK이다.
* 크롤링 데이터가 없는 경우, 자치구 확진 인원 수는 null값을 가질 수 있다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOCIAL\_DIST Table** | | | | |
| Attribute | Data Type | Not NULL | Key | 설명 |
| DIST\_LEVEL | Int | Y | PK | 거리두기 단계 |
| DIST\_INFO | Varchar | Y | - | 거리두기 단계 정보 |
| STANDARD | Varchar | Y | - | 거리두기 단계 기준 |
| GATHERING | Int | N | - | 거리두기 단계에 따른 모임 가능 인원 수 |

* 사회적 거리두기 데이터 테이블이다. primary key는 거리두기 단계이다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **VACCINE Table** | | | | |
| Attribute | Data Type | Not NULL | Key | 설명 |
| DATE | DATE | Y | PK | 데이터 업데이트 날짜 |
| NAME | Varchar | Y | PK | 백신 이름 |
| VACC\_ONCE | Int | Y | - | 백신 1차 접종 인원 수 |
| VACC\_FULLY | Int | N | - | 백신 2차 접종 인원 수 |
| VACC\_BOOST | Int | N | - | 백신 추가 접종 인원 수 |

* 백신 데이터 테이블이다. primary key는 백신 이름과 데이터 업데이트 날짜이다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USER Table** | | | | |
| Attribute | Data Type | Not NULL | Key | 설명 |
| NAME | Varchar | Y | - | 사용자의 이름 |
| RRN | Char(14) | Y | PK | 사용자의 주민등록번호 |
| SEX | Char(1) | Y | - | 사용자의 성별 |
| AREA | Varchar | Y | - | 사용자의 지역(시) |
| DISTRICT | Varchar | Y | - | 사용자의 지역(구) |
| VACC\_NAME | Varchar | N | - | 접종 백신 종류 |
| VACCINATED | Varchar | N | - | 사용자의 백신 접종 여부 |
| VACCINATED\_DATE | Date | N | - | 사용자의 백신 접종 날짜 |

* 사용자 데이터 테이블이다. primary key는 사용자의 주민등록 번호(RRN) 이다.
* 사용자의 성별 데이터에 따른 주민등록 식별 번호(8)가 1 또는 2로 구분되는 constraint를 갖는다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TODAY\_AREA Table** | | | | |
| Attribute | Data Type | Not NULL | Key | 설명 |
| AREA | Varchar | Y | PK | 오늘 확진자 발견된 지역(시) |
| DISTRICT | Varchar | Y | PK | 오늘 확진자 발견된 지역(구) |
| CONFIRMED | Int | Y | - | 실시간 지역별 확진자 수 |

- 오늘 확진자 데이터 테이블로 새롭게 추가되었다. 당일 생성되고 삭제되는 데이터이다. 유저와 연동해, 유저의 지역구 기준 확진 정보를 알려주기 위해 활용된다.

- primary key는 오늘 확진자가 발견된 지역(시), 자치구(구)이다.

# **4. Implementation : with Result of DBMS Relational Model**

4.1 DB Schema Create

4.2 Constraints

* Physical DB 제약조건 : Covid19 & Vaccination information DB Data Schema
* Business semantic 제약조건 : Covid19 & Vaccination information DB Data Schema

|  |  |
| --- | --- |
| **제약 조건** | **내용 및 주요 코드** |
| 인원 | 인원과 관련한 데이터는 0과 같거나 그보다 커야 한다. |
| 격리자의 경우 인원이 줄어들 수 있어 음의 정수의 값도 허용한다. |
| CONSTRAINT chk\_daily\_data CHECK (daily\_confirmed >= 0 AND daily\_deased >= 0 AND daily\_recovered >= 0 AND daily\_vacc\_once >= 0)  CONSTRAINT chk\_area CHECK (area\_confirmed >= 0 AND area\_deseased >= 0 AND area\_recovered >= 0 AND area\_dist\_level > 0)  CONSTRAINT chk\_district CHECK (district\_confirmed >= 0)  CONSTRAINT chk\_today\_confirmed CHECK (today\_confirmed >= 0)  CONSTRAINT chk\_vaccine CHECK (vacc\_once >= 0) | |
| 거리두기 | 거리두기 단계는 반드시 양의 정수의 단계를 가져야 한다. |
| CONSTRAINT chk\_social\_dist CHECK (dist\_level > 0) | |
| 사용자 | 사용자의 성별과 주민등록번호의 성별 확인 데이터가 일치해야한다 |
| CONSTRAINT chk\_user CHECK ((user\_sex = 'M' AND substr(user\_rrn, 8,1) = 1) OR (user\_sex = 'M' AND substr(user\_rrn, 8,1) = 3) OR (user\_sex = 'W' AND substr(user\_rnn, 8, 1) = 2) OR (user\_sex = 'W' AND substr(user\_rnn, 8, 1) = 4)) | |

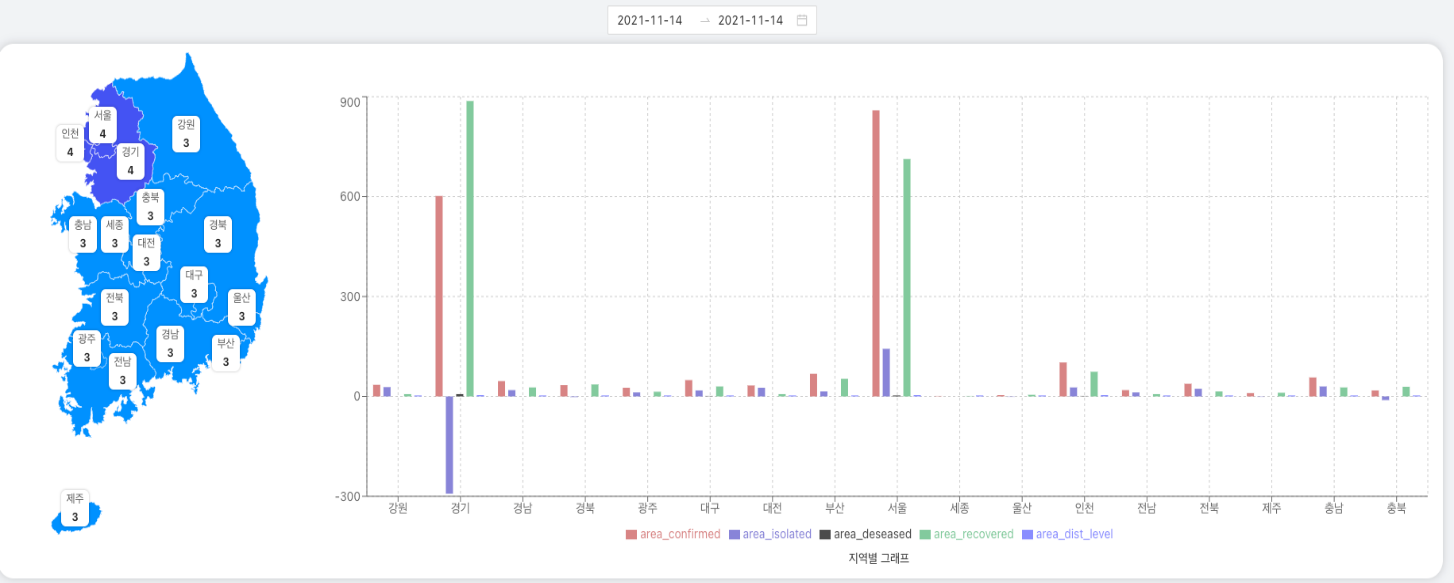
4.3 Case scenarios with relational model

Sample:

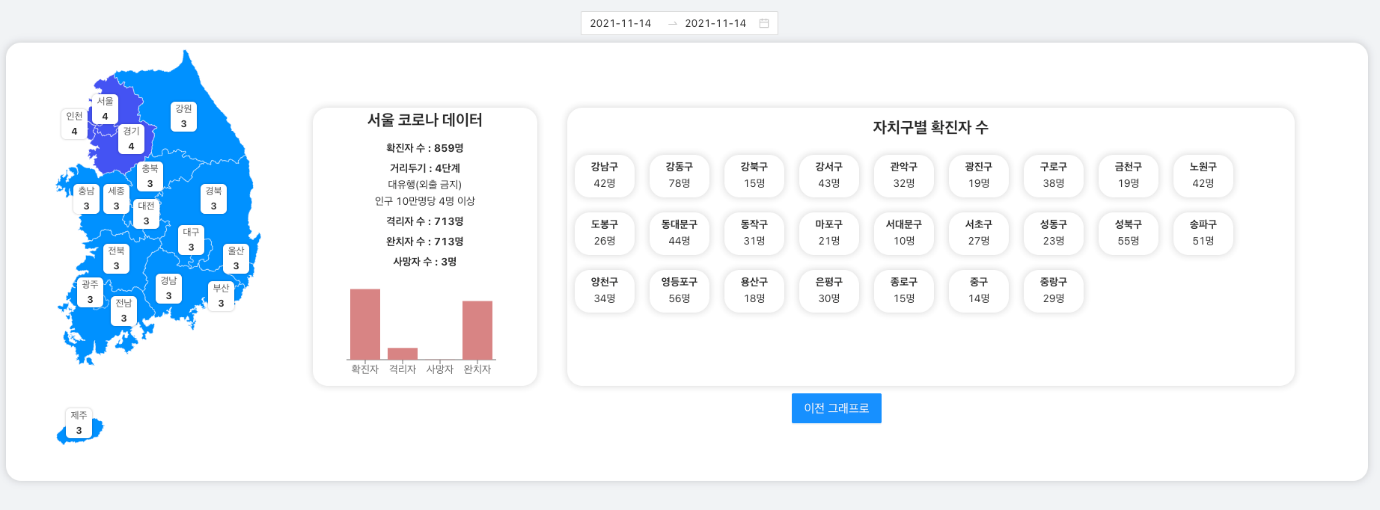
|  |
| --- |
| **SQL QUERY** |
| 1. 해당 날짜의 모든 지역의 코로나 데이터를 가져온다.   SELECT \* FROM area WHERE update\_date = **날짜** |
| 결과: |

# **5. Result : WEB Application**

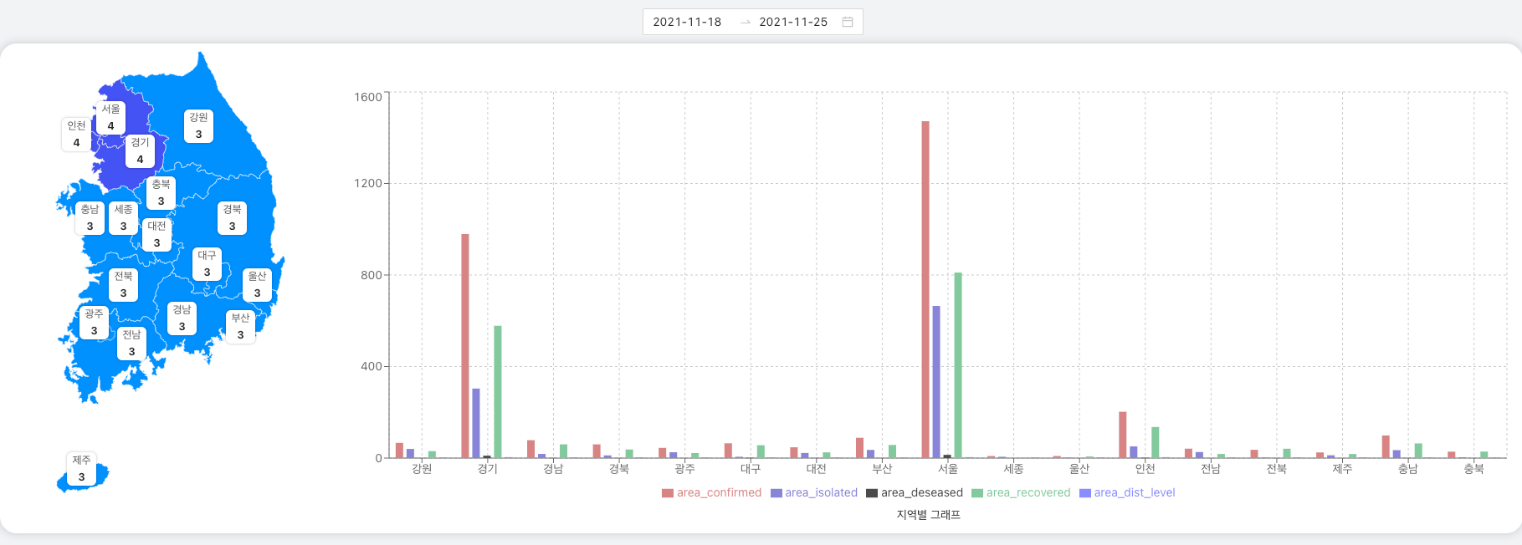
|  |  |
| --- | --- |
| action | 1. 지역 별 데이터 🡪 날짜 하나 선택 |
| 해당 날짜의 모든 지역의 코로나 데이터를 가져온다.  SELECT \* FROM area WHERE update\_date = **날짜** | |



|  |  |
| --- | --- |
| action | 1. 지역 별 데이터 🡪 날짜 하나 선택 후 지역 선택 |
| 1. 해당 지역 코로나 데이터를 가져온다.   SELECT \* FROM area WHERE update\_date = **날짜** | |
| 1. 해당 지역의 자치구 확진자 데이터를 가져온다.   SELECT \* FROM district WHERE update\_date = **날짜** AND area\_name = **지역** | |



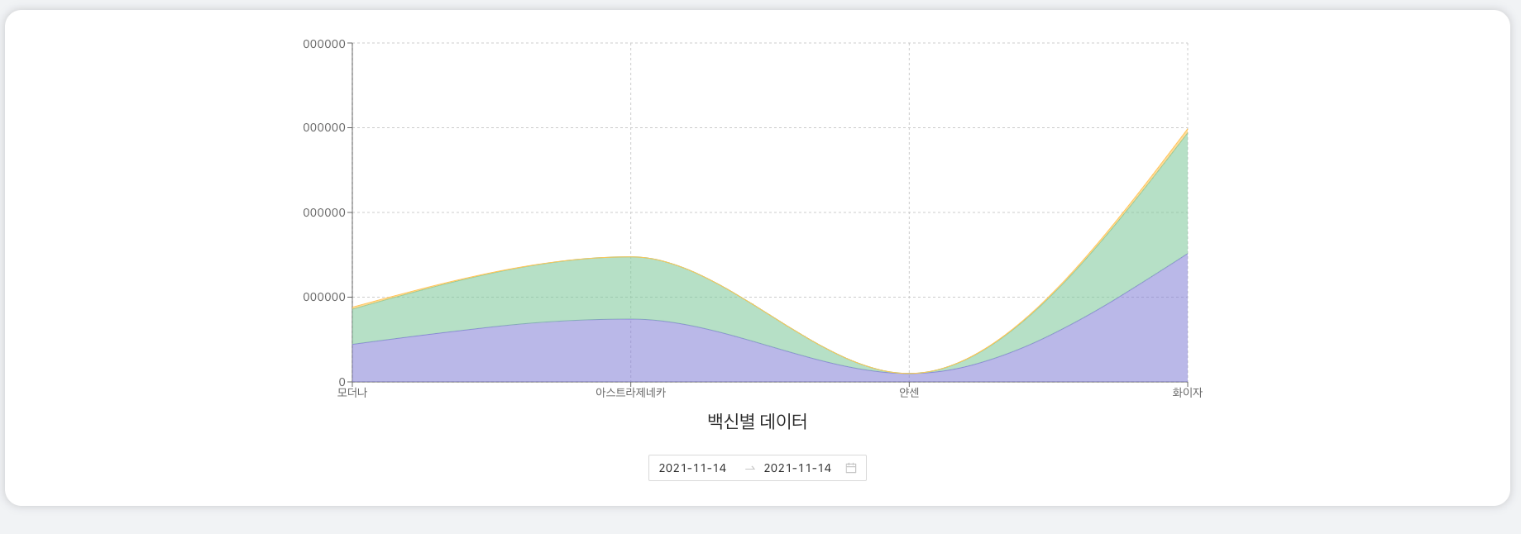
|  |  |
| --- | --- |
| action | 지역 별 데이터 🡪 날짜 두개 선택 |
| 두개 날짜 사이의 전국 코로나 데이터의 평균 데이터를 가져온다. (GROUP BY 사용)  SELECT area\_name, avg(area\_confirmed), avg(area\_isolated), avg(area\_deseased), avg(area\_recovered), ROUND(avg(area\_dist\_level))  FROM area  WHERE update\_date BETWEEN **날짜1** AND **날짜2** GROUP BY area\_name | |



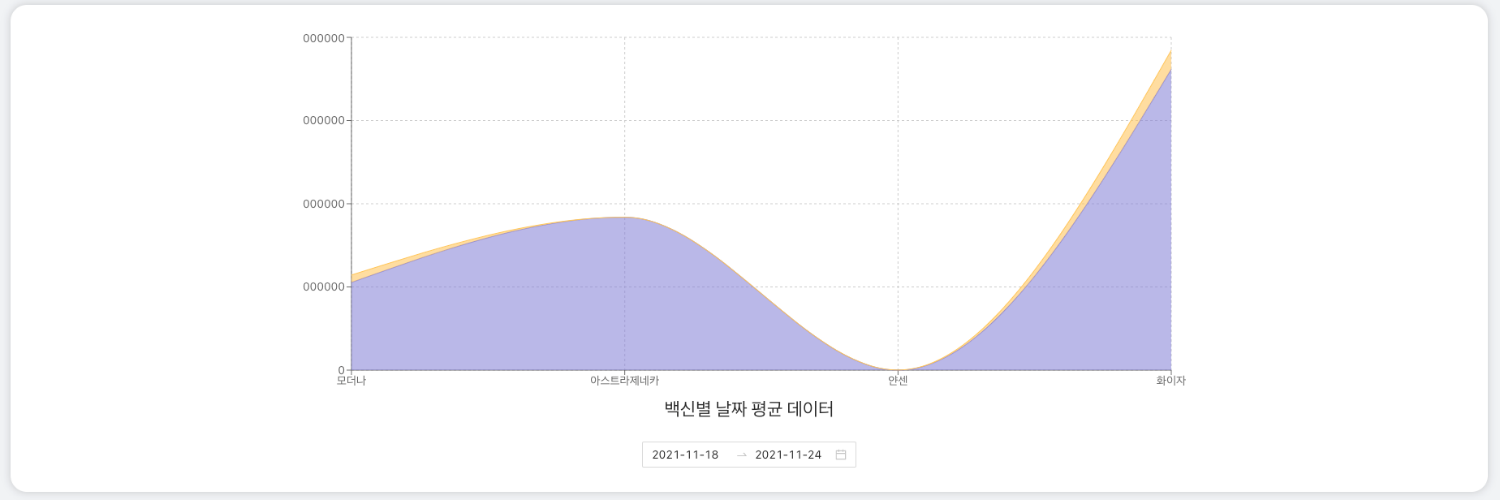
|  |  |
| --- | --- |
| action | 지역 별 데이터 🡪 날짜 두개 선택 후 지역 선택 |
| 1. 두개 날짜 사이 지역의 코로나 데이터의 평균 값을 가져온다. (GROUP BY 사용)   SELECT area\_name, avg(SELECT area\_name, avg(area\_confirmed), avg(area\_isolated), avg(area\_deseased), avg(area\_recovered), ROUND(avg(area\_dist\_level))  FROM area  WHERE update\_date BETWEEN **날짜1** AND **날짜2**  GROUP BY area\_name HAVING area\_name = **지역** | |
| 1. 두개 날짜 사이 자치구의 코로나 데이터의 평균 값을 가져옴 (GROUP BY 사용)   SELECT \* FROM district WHERE update\_date = **날짜** AND area\_name = **지역** | |

이미지

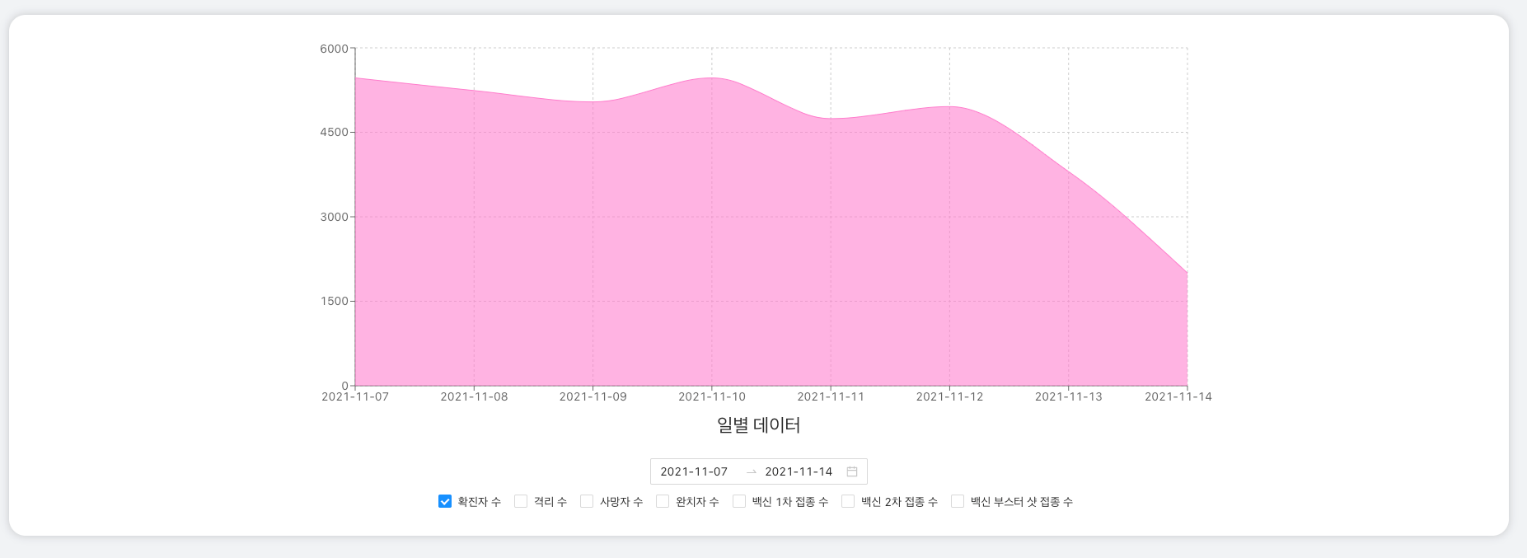
|  |  |
| --- | --- |
| Action | 백신 별 데이터 🡪 날짜 하나 선택 |
| 해당 날짜의 백신 데이터를 가져온다.  SELECT \* FROM vaccine WHERE update\_date = **날짜** | |



|  |  |
| --- | --- |
| action | 백신별 데이터 🡪 날짜 두개 선택 |
| 날짜 사이 데이터의 평균 데이터를 가져온다. (GROUP BY 사용)  SELECT vacc\_name, avg(vacc\_once), avg(vacc\_fully), avg(vacc\_boost)  FROM vaccine  WHERE update\_date BETWEEN **날짜1** AND **날짜2** GROUP BY vacc\_name | |

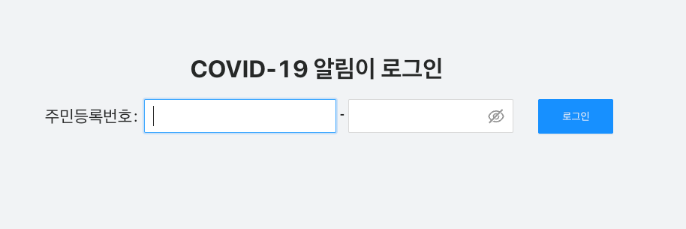
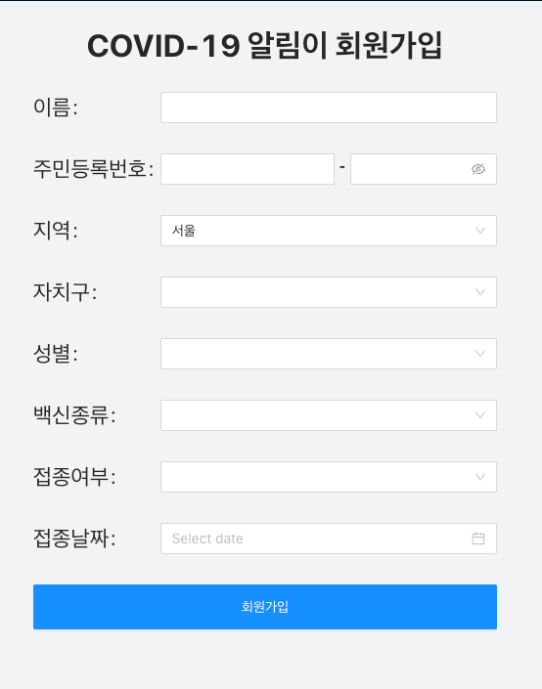


|  |  |
| --- | --- |
| action | 일별 데이터 🡪 날짜 선택 |
| 1. 날짜 하나 선택해서 일별 총 데이터를 가져온다.   SELECT **(체크박스에서 선택한 수)** FROM daily\_data WHERE update\_date = **날짜** | |
| 1. 날짜 두개 선택해서 일별 총 데이터를 가져온다.   SELECT **(체크박스에서 선택한 열)** FROM daily\_data WHERE update\_date  BETWEEN **날짜1** AND **날짜2** | |



|  |  |
| --- | --- |
| action | 로그인 |
| 해당 데이터 존재한다면 로그인 처리  SELECT \* FROM user WHERE user\_rrn = **유저주민번호** | |

|  |  |
| --- | --- |
| action | 회원가입 |
| INSERT INTO user SET **(이름,주민번호,지역,자치구,성별,백신종류,접종여부,접종날짜)** | |



# **6. Crawling Data & DevOps**

# **6.1 Crawling Data**

당일 업데이트로 제공되는 데이터들은 이전 데이터를 얻을 수 없다. 그래서 해당 프로젝트는 백엔드 측에서 cheerio 라이브러리를 활용해 크롤러 기능을 도입 후 서버가 구동되는 N일 동안 데이터를 수집할 예정이다. 아래는 데이터별 크롤러 타겟 사이트이다.

|  |  |
| --- | --- |
| Table | 크롤링한 사이트 URL |
| DAILY\_DATA | https://corona-live.com/ |
| AREA | <https://kosis.kr/covid/covid_index.do>  http://ncov.mohw.go.kr/regSocdisBoardView.do |
| DISTRICT | https://www.seoul.go.kr/coronaV/coronaStatus.do |
| VACCINE | https://ncv.kdca.go.kr/vaccineStatus.es?mid=a11710000000 |
| DIST\_LEVEL | http://ncov.mohw.go.kr/socdisBoardList.do?brdId=6&brdGubun=64&dataGubun=641 |
| TODAY\_CONFIRMED | https://corona-live.com/ |

# **6.2 DevOps**

|  |  |
| --- | --- |
| **사용 언어** | JavaScript |
| **Frontend개발 도구** | React |
| **Backend 개발도구** | Node JS, Express |
| **데이터베이스** | MySQL |
| **버전 관리** | Git, GitHub |

텍스트이(가) 표시된 사진

자동 생성된 설명

해당 프로젝트는 위 스택을 기반으로 웹 어플리케이션으로 개발되었다.

결과물은 **4. 5.** 내용에서 확인할 수 있다.

Node.js와 MySQL을 연결하여 물리적 설계를 진행하고 연결된 MySQL에 Express 프레임워크 REST API와 연결해 DML을 요청 후 데이터를 조회/수정할 수 있다.

클라이언트는 전 단계에서 구현된 REST API를 통해 데이터 조회를 할 수 있다.

# 7. 참고자료