

ver.2021 노베이스 모던 웹개발

- 5. GraphQL을 위한 PostgreSQL

작성자: 경상국립대학교 컴퓨터과학과 조교 이자룡

목차

p.2	-	0. Orientation
p.5	-	1. 윈도우 세팅
p.5	-	1-1. VSCode
p.5	-	1-2. 윈도우 업데이트
p.7	-	1-3. Windows Terminal
p.8	-	1-4. WSL2 Ubuntu
p.15	-	1-5. Remote WSL
p.17	-	1-6. oh-my-zsh
p.22	-	1-7. node.js
p.25	-	2. 기본적인 리눅스 명령어
p.30	-	3. PostgreSQL, TablePlus 설치
p.30	-	3-1. PostgreSQL 설치
p.36	-	3-2. TablePlus 설치
p.38	-	4. CRUD
p.38	-	4-1. C: CREATE TABLE
p.42	-	4-2. C: INSERT INTO
p.43	-	4-3. R: SELECT
p.46	-	4-4. U: UPDATE
p.47	-	4-5. D: DELETE, DROP
p.48	-	5. 관계형 데이터베이스 모델링
p.51	-	5-1. UI 작성
p.54	-	5-2. ERD
p.56	-	5-3. LucidChart - 테이블 만들기
p.63	-	5-4. LucidChart - 관계
p.66	-	5-4-1. one to many
p.69	-	5-4-2. one to one
p.70	-	5-4-3. many to many
p.73	-	6. 마치며

0. Orientation

* 이 수업을 시작하기 전에 반드시, JavaScript, TypeScript, React Hook 기초를 알고 시작하길 바란다. 《 ver.2021 노베이스 모던 웹개발 》 첫번째 강좌인 『 React 시작 전에 알아야 할 JavaScript, TypeScript 』, 두번째 강좌인 『 React Hook 』이 준비되어 있다.

데이터베이스까지 온다고 고생 많았다! 여기서부터 옵션이다. 무슨말이냐, 해도 되고 안해도 된다는 뜻이다. DB없이도 친구들에게, 교수님에게, 엄마에게 자랑할만한 앱 만들 수 있다. 그래서, 이제까지 배운것으로 만족하는 사람은 중단해도 좋다. 웹앱에 꼭 DB가 있어야하는것이 아니다! 우리가 단순히 누구한테 자랑할만한 웹앱을 만들고싶다면, 지금까지 배운것을 활용해보는것으로도 충분하다.

그러나, 컴퓨터공학 전공생 기준으로 데이터베이스까지 만들어야 프로젝트로 인정해준다. 프로젝트를 준비하는 학부생은 DB연동을 피할 수 없다. 앞으로 고된 길을 걷게 될 것이다. 백엔드, 즉 서버의 세상이다. 더이상 화면에 보이는 세상이 아니므로 난이도가 급격히 올라갈 것이다. 리눅스 터미널 명령어를 쓰는것부터가 이미 마우스로 클릭하면서 확인했던 눈에 보이는 세계와는 다르다. 특히 이 과목은 시작부터 윈도우 유저에게 리눅스를 설치하라고 “강요”할 것이다. 그러므로, 난이도때문에 중도 하차하고 싶은 사람이 있다면 그것도 환영한다. 아직 여러분들에게 때가 아닌 것이다. 그런 사람들은 지금까지 배운 것으로 무언가를 만들어보는 시도를 해봐야한다. 만들다보면 DB와 서버에 대한 갈증이 생길 것이다. 그때 돌아와서 이 강의를 보거나, 다른 사람들이 찍은 튜토리얼, 다른 사람들이 써놓은 저작물들을 참고하면 된다. 필요성을 느끼지 못했을 때 하는 공부는 끔찍할 뿐이다.

데이터베이스란, 컴퓨터 시스템에서 전자적으로 저장되고 **엑세스되는** 정리된 데이터 컬렉션이다. 여기서 핵심은 엑세스, 즉 접근이다. 몇 명이 접근할 수 있는가? 구글을 생각해보자. 하루 평균 몇 명이 구글링을 할까? 35억건이다. 즉, 웹앱은 기본적으로 혼자 쓰지 않는다. 여러명이 데이터베이스에 접근하고, 허용되는 범위 내에서 데이터를 변경할 수 있다.

데이터베이스를 관리할 수 있는 프로그램을 데이터베이스 관리 시스템(DBMS)이라고 한다. DBMS는 크게 두 종류, SQL 과 NoSQL 로 나뉜다. MongoDB, Cassandra 같은, SQL을 사용하지 않는 NoSQL 기반 DBMS도 인기있지만, 전통적이고 안정적인 업계 표준 데이터베이스는 SQL을 사용하는 관계형 데이터베이스(Relational Database.

RDB)이며, 이러한 RDB를 관리하는 프로그램을 RDBMS 라고 한다. SQL(Structured Query Language)은 다른 걸 의미하는것이 아니라, RDB에서 사용하는 프로그래밍 언어다. 이것은, RDBMS를 배우는 것이 SQL을 배우는 것과 같다는것을 의미한다. 시장에 출시된 RDBMS의 종류는 많지만, SQL 만 제대로 알아도 다른 RDBMS 의 기본적인 기능을 사용하는데엔 무리가 없다. 제품별로 약간의 문법차이만 있을 뿐이다.

RDB란 무엇인가? 테이블(표)간의 “관계” 로 이루어진 데이터베이스다. 우리는 여러 테이블을 만들게 될 것이고, 이 테이블의 관계를 정의하게 될 것이다.

시장에 정착한 RDBMS 는 다음 여섯가지이다.

1. Oracle
2. PostgreSQL
3. SQLite
4. MariaDB
5. MySQL
6. MSSQL

우리는 이 중에서 PostgreSQL (psql, postgres 라고도 하며, 포스트그래스큐엘이라고 발음함) 을 배워볼 것이다. PostgreSQL은 업계 표준 Oracle과는 다르게 무료이며, 한국에선 인기가 없지만 북미 기준 2위의 점유율을 차지하고 있다. 가볍지는 않지만, 가벼움을 추구하는 SQLite 에 비해 환경설정에서 신경써야할것들이 적다. 사실 PostgreSQL 을 선택한 가장 큰 이유는 따로 있다. 이 수업 다음에 배워볼 PostGraphile 을 통해 GraphQL 과의 연동을 하게 될텐데, 해당 라이브러리가 굉장히 편리하다. 따라서 이 수업에선 PostgreSQL을 통해 관계형 데이터베이스를, SQL을 학습해보도록 할 것이다.

다른 데이터베이스 수업과 다른점은 다음과 같다.

1. JOIN을 가르치지 않음

GraphQL 을 사용할 경우 JOIN을 사용하지 않기 때문이다. 다른 라이브러리를 통해 억지로 사용할 수는 있지만, 우리 수준에선 불필요하다고 판단했다.

2. 리눅스로 진행

정확히는 윈도우에서 WSL 을 세팅해 리눅스를 사용할 것이다.

3. TablePlus 사용

터미널만 가지고 진행할수도 있지만, 초보자에겐 눈에 확실히 보이는 쉬운 툴들이 중요하다. 이 수업에선 TablePlus 를 사용해볼 것이다.

4. ERD 제작

DB의 진정한 꽃이다. 우리는 ERD라는, RDB의 설계도를 LucidChart라는 툴을 이용해 만들어볼 것이다.

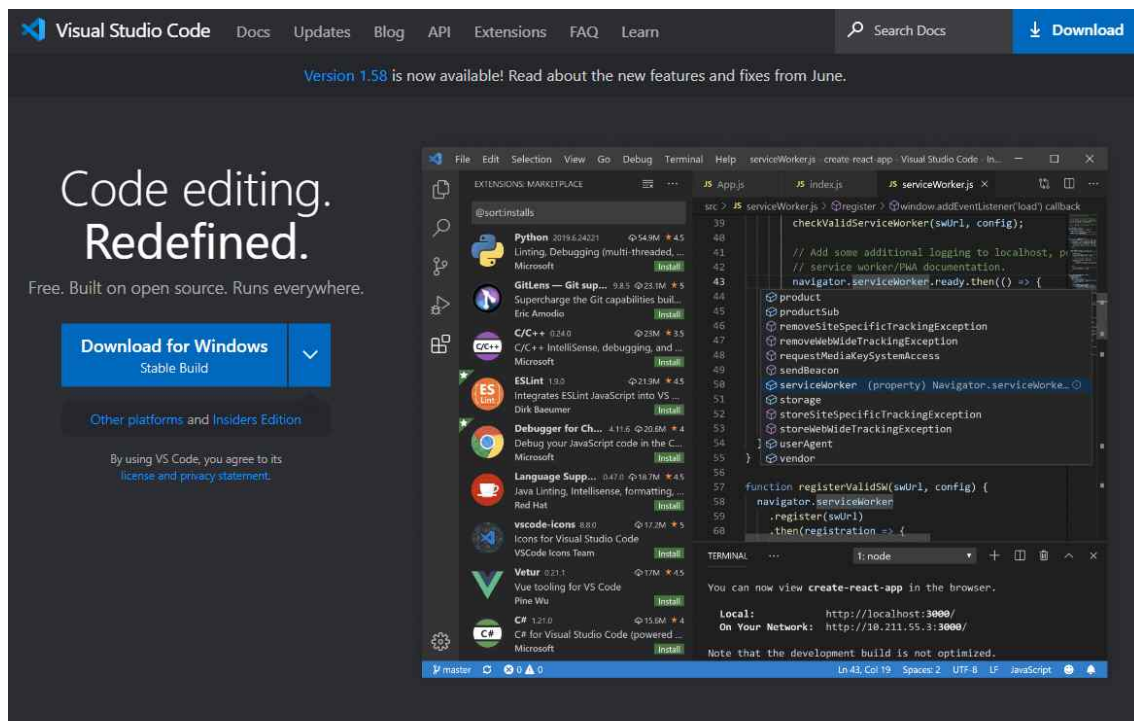
이 수업은 컴퓨터공학 전공자들에게 있어서 매우 얇은 지식만 가르친다. 그렇다고 일반인에게 쉽지는 않다. 다시 묻는다. **DB가 정말 필요한가? 필요하지 않다면 이 수업을 들을 때가 아니다.** 이 수업은 어렵다. 그럼에도 DB 수업을 듣고 싶은가? 여러분의 윈도우 세팅부터 시작해보도록 하자.

1. 윈도우 세팅

* 이 교안에선 Mac 에서의 개발환경세팅은 다루지 않는다. Mac 유저는 homebrew 와 구글링을 이용해 개발환경세팅을 해주도록 하자.

1-1. VSCode

맨 먼저, VSCode 를 설치할 것이다. 여러분의 코드를 웹 환경이 아니라, 여러분의 컴퓨터에서 작성해볼 것인데, 이를 위해 가장 인기있는 텍스트 에디터인 VSCode를 사용할 것이다.



IntelliSense



Run and Debug



Built-in Git

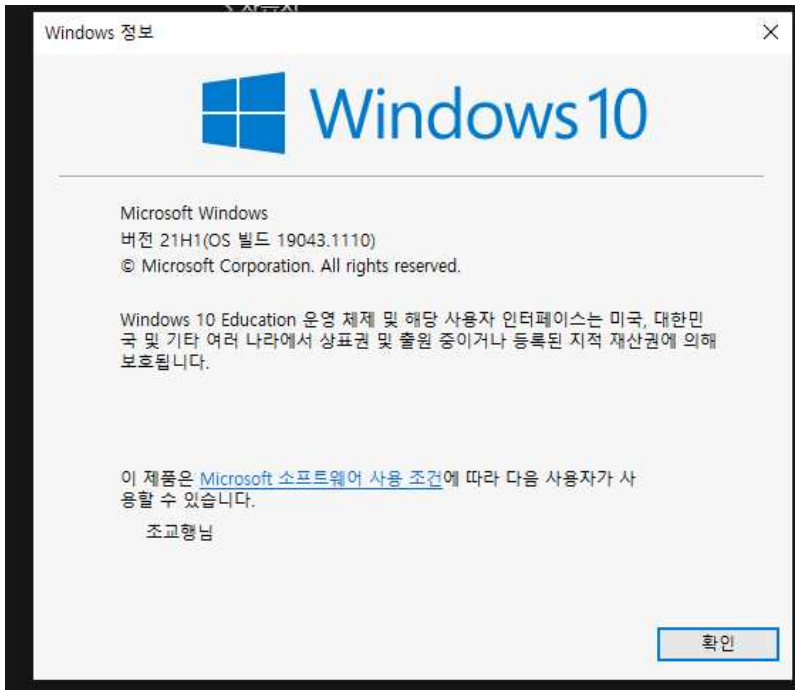


Extensions

파란 버튼 눌러서 다운로드해준다.

1-2. 윈도우 업데이트

먼저 윈도우 업데이트부터 해주자. WSL을 설치하기 위해선, 64비트 기준 윈도우 버전 1903 이상, 빌드 18362 이상이 필요하다. 이것을 확인하기 위해서, 윈도우버튼(왼쪽 ctrl 옆) + R 을 누르면 실행창이 뜨고, winver 를 입력해서 실행시킨다.



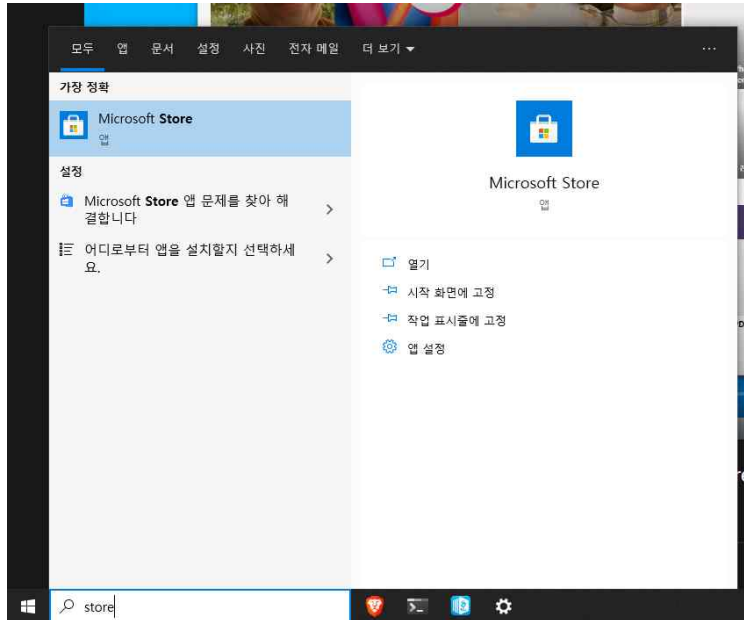
현재 나의 PC는 버전 21H1, 빌드 19043 이므로 이 기준을 충족한다. 만약 요구조건을 만족하지 못한다면, 윈도우즈 업데이트를 실행시켜서 이 조건을 만족시켜야한다.



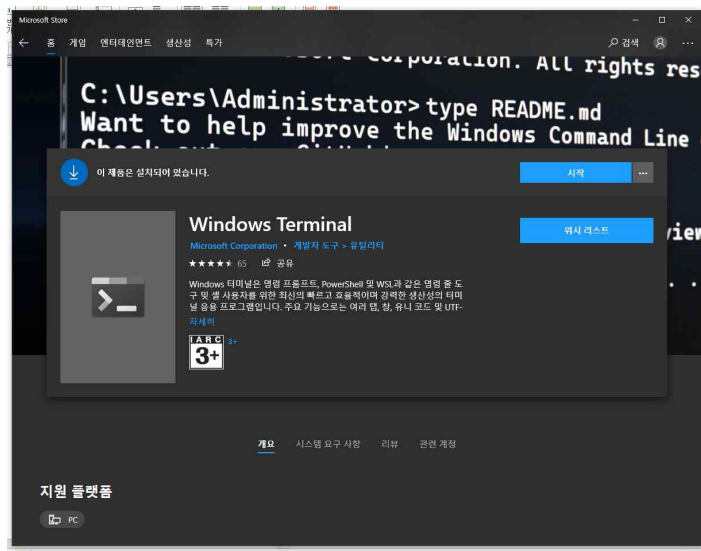
업데이트 확인 버튼을 클릭했을 때 현재 최신 상태입니다 라는 문구가 떠야한다.
선택적 업데이트도 웬만하면 모두 설치해주자.

1-3. Windows Terminal

우리 컴퓨터에 제대로 된 터미널을 설치할 것이다. 터미널은 콘솔이라고도 하는데,
까만 배경화면에 명령어를 치는 프로그램이다. Microsoft Store에서 깔아주기로
한다.

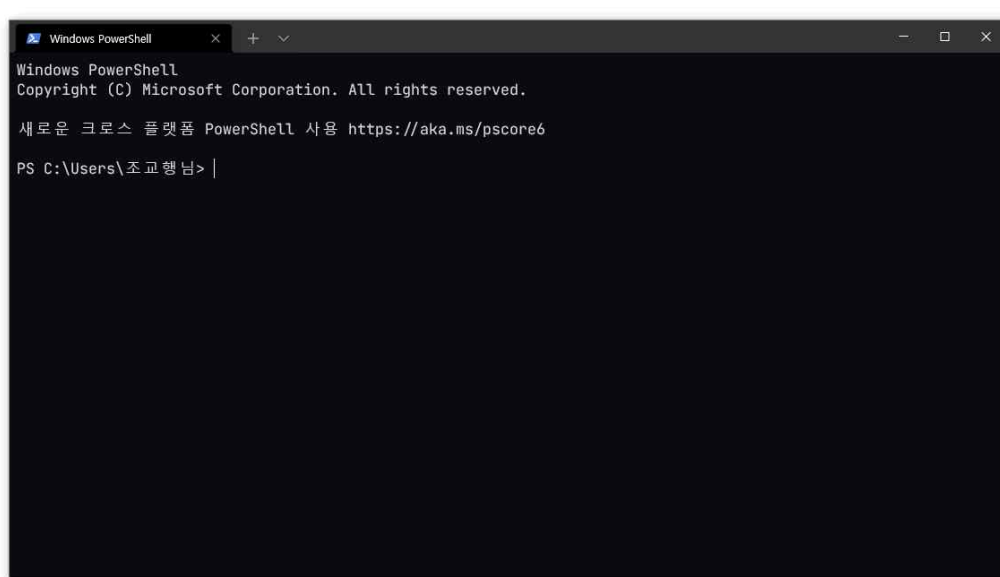


Windows Terminal 을 검색해 설치해주면 된다.





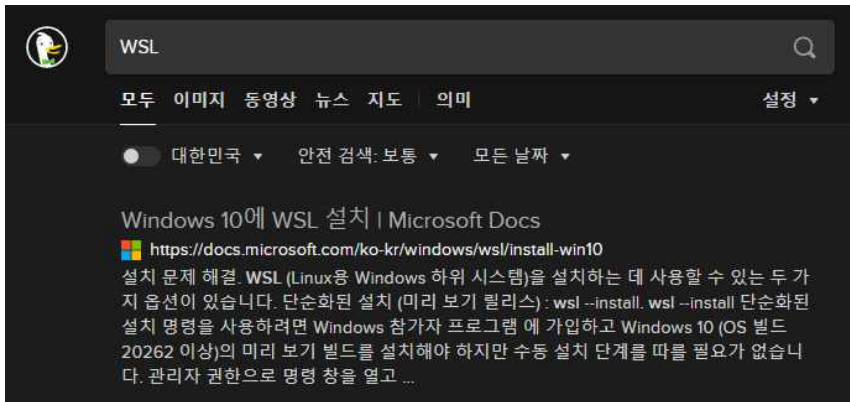
Microsoft 로그인 요청할 경우, 관심 없음 클릭하면 된다.



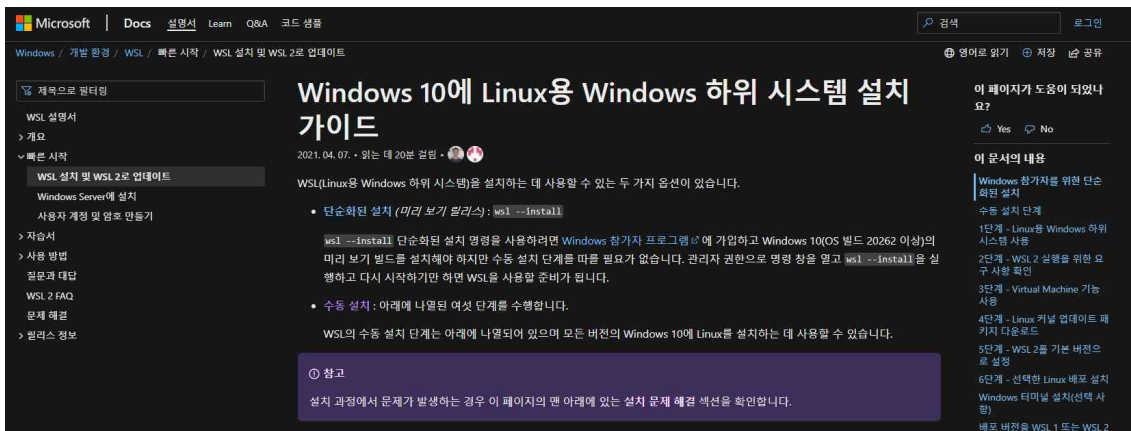
실행시키면 다음과 같다. 여러분의 컴퓨터에 깔려있는 각종 콘솔프로그램들, 명령 프롬프트(cmd), PowerShell, Ubuntu 를 이 터미널 프로그램으로 실행시킬 것이다. 화면에 보이는것은 PowerShell 이며, 우리 여기에 Ubuntu를 추가할 것이다.

1-4. WSL2 Ubuntu

리눅스는 윈도우, Mac 과 같은 하나의 운영체제이다. 개발할 때의 편의성은 윈도우보다 리눅스가 훨씬 좋기 때문에, 우리 터미널에 리눅스의 가장 인기있는 배포판인 Ubuntu 를 추가해 줄 것이다.



WSL 이라고 검색한다. 상위의 광고들은 무시하고 Microsoft 공식 홈페이지의 WSL 설치 페이지로 가자.



단순화된 설치와 수동 설치가 있다. 단순화된 설치는 여러분의 컴퓨터를 매우 불안정하게 만들 것이기 때문에 추천하지 않는다. 수동 설치를 클릭하자

수동 설치 단계

Windows 참가자 빌드를 사용하지 않는 경우 WSL에 필요한 기능을 아래 단계에 따라 수동으로 활성화해야 합니다.

1단계 - Linux용 Windows 하위 시스템 사용

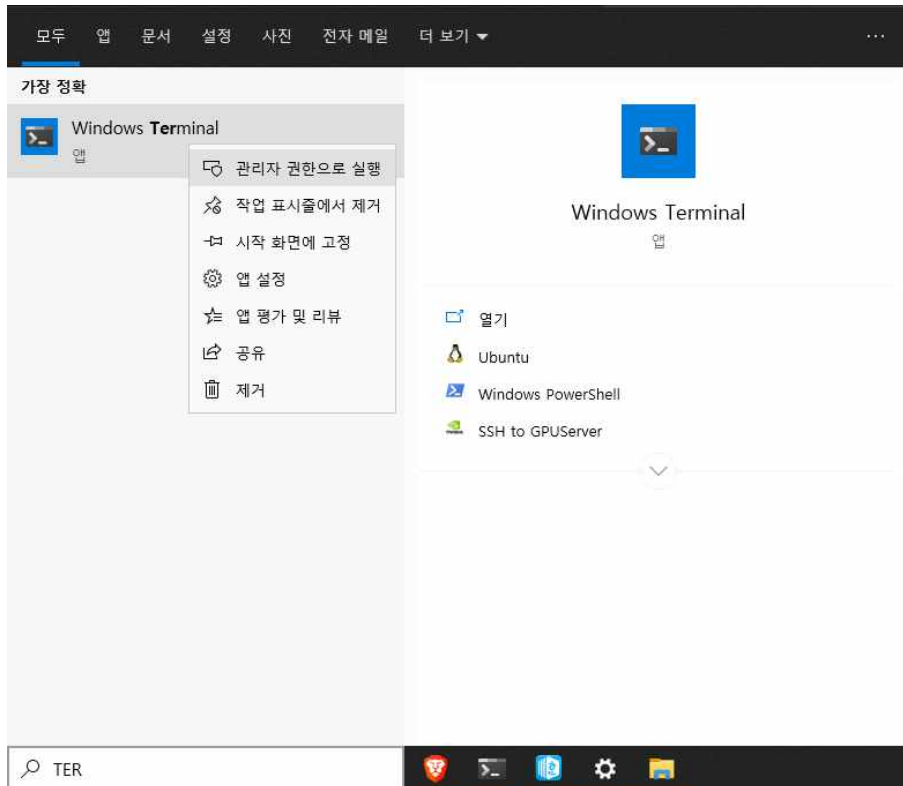
Windows에서 Linux 배포를 설치하려면 먼저 "Linux용 Windows 하위 시스템" 옵션 기능을 사용하도록 설정합니다.

PowerShell을 관리자 권한으로 열어 실행합니다.

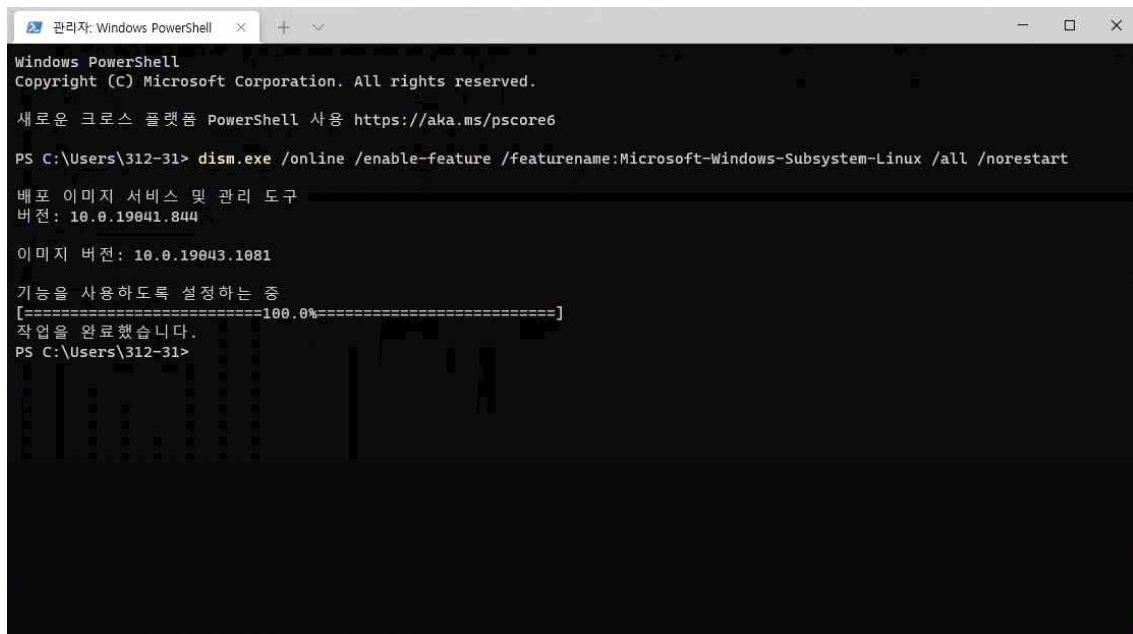
```
PowerShell
dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
```

이제 2단계로 이동하여 WSL 2로 업데이트하는 것이 좋습니다. 그러나 WSL 1만 설치하려면 이제 머신을 다시 시작 하여 6단계 - 선택한 Linux 배포 설치로 이동할 수 있습니다. WSL 2로 업데이트하려면 머신이 다시 시작될 때까지 기다린 후 다음 단계로 이동합니다.

우리 윈도우엔 본래 WSL을 사용할 수 있는 옵션이 있는데, 그것부터 켜줘야 한다.
코드 오른쪽의 “복사”버튼을 클릭한 후, 터미널을 관리자 권한으로 실행하자.



PowerShell을 이용해 컴퓨터에 무엇인가를 설치할 경우, 또는 컴퓨터 전체에 영향을 끼치는 큰 작업을 할 경우엔 관리자 권한으로 실행해야 한다. 이것은 나중에 배우게 될 Ubuntu 의 `sudo` 명령과 같다.



코드를 붙여넣으려면 터미널에서 마우스 오른쪽 버튼을 클릭하면 된다.

2단계 - WSL 2 실행을 위한 요구 사항 확인

WSL 2로 업데이트하려면 Windows 10을 실행해야 합니다.

- x64 시스템의 경우: 버전 1903 이상, 빌드 18362 이상
- ARM64 시스템의 경우: 버전 2004 이상, 빌드 19041 이상
- 18362보다 낮은 빌드는 WSL 2를 지원하지 않습니다. [Windows Update Assistant](#)를 사용하여 Windows 버전을 업데이트합니다.

버전 및 빌드 번호를 확인하려면 Windows 로고 키 + R 을 선택하고, winver 를 입력하고, 확인 을 선택합니다. [설정] 메뉴에서 최신 Windows 버전으로 업데이트합니다.

① 참고

Windows 10 버전 1903 또는 1909를 실행하고 있는 경우 Windows 메뉴에서 "설정"을 열고, "업데이트 및 보안"으로 이동한 다음, "업데이트 확인"을 선택합니다. 빌드 번호는 18362.1049 이상 또는 18363.1049 이상이고, 부 빌드 번호는 .1049 이상이어야 합니다. 자세한 정보: [WSL 2 지원이 Windows 10 버전 1903 및 1909에 제공됨](#). 문제 해결 지침을 참조하세요.

버전 및 빌드 확인단계이다. 이 단계는 이미 진행했으니 건너뛰겠다.

3단계 - Virtual Machine 기능 사용

WSL 2를 설치하려면 먼저 Virtual Machine 플랫폼 옵션 기능을 사용하도록 설정해야 합니다. 이 기능을 사용하려면 머신에 가상화 기능이 필요합니다.

PowerShell을 관리자 권한으로 열어 실행합니다.

```
PowerShell 복사  
  
dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
```

머신을 다시 시작하여 WSL 설치를 완료하고 WSL 2로 업데이트합니다.

아까와 같은 방식으로 복사하고 붙여넣어 실행한다. 이후 컴퓨터를 **재부팅**해줘야한다.

4단계 - Linux 커널 업데이트 패키지 다운로드

1. 최신 패키지를 다운로드합니다.

- x64 머신을 최신 WSL2 Linux 커널 업데이트 패키지

① 참고

ARM64 머신을 사용하는 경우 ARM64 패키지 를 대신 다운로드하세요. 사용하고 있는 머신의 종류를 잘 모르는 경우 명령 프롬프트 또는 PowerShell을 열고 `systeminfo | find "System Type"` 을 입력합니다. 주의: 비 영어 Windows 버전에서는 검색 텍스트를 수정해야 할 수 있습니다(예: 독일어의 경우 `systeminfo | find "Systemtyp"`).

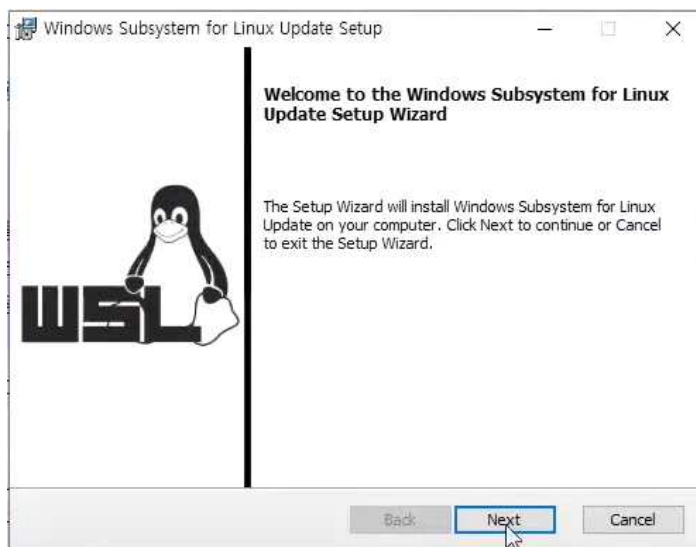
2. 이전 단계에서 다운로드한 업데이트 패키지를 실행합니다. (실행하려면 두 번 클릭 - 관리자 권한을 요구하는 메시지가 표시되면 '예'를 선택하여 이 설치를 승인합니다.)

설치가 완료되면 새 Linux 배포를 설치할 때 WSL 2를 기본 버전으로 설정하는 다음 단계로 이동합니다. (새 Linux 설치를 WSL 1로 설정하려면 이 단계를 건너뛸니다.)

① 참고

자세한 내용은 Windows 명령줄 블로그 에서 사용할 수 있는 WSL2 Linux 커널업데이트 변경 을 문서를 참조하세요.

재부팅이 완료되면 다시 WSL 설치 페이지로 접속해, 4단계의 파란 글씨를 클릭해서 커널 업데이트 패키지를 다운로드받아 설치하자.



5단계 - WSL 2를 기본 버전으로 설정

PowerShell을 열고 이 명령을 실행하여 새 Linux 배포를 설치할 때 WSL 2를 기본 버전으로 설정합니다.

PowerShell

복사

```
wsl --set-default-version 2
```

Ubuntu 설치 시 WSL2 로 설정해주는 단계이다. 복사해서 실행하자.

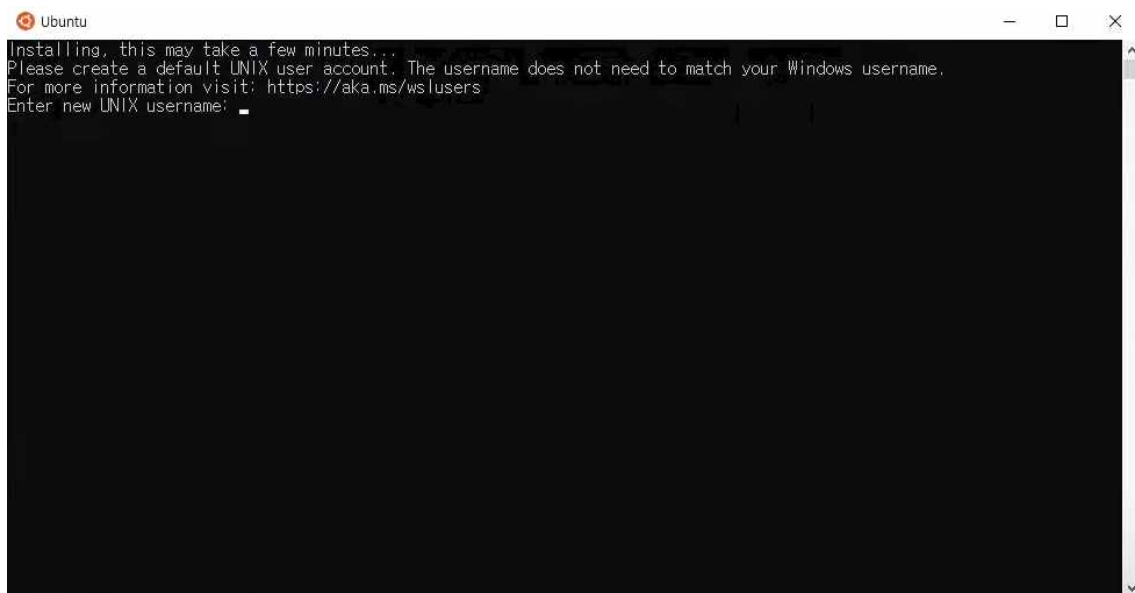
6단계 - 선택한 Linux 배포 설치

1. [Microsoft Store](#) 를 열고 즐겨 찾는 Linux 배포를 선택합니다.

Microsoft Store 에서 ubuntu 를 검색해 설치해주자. 세 가지가 있다.

ubuntu	20.04
ubuntu20.04	20.04
ubuntu18.04	18.04

우린 버전 없이 그냥 ubuntu 라고 적혀있는걸 받으면 된다.



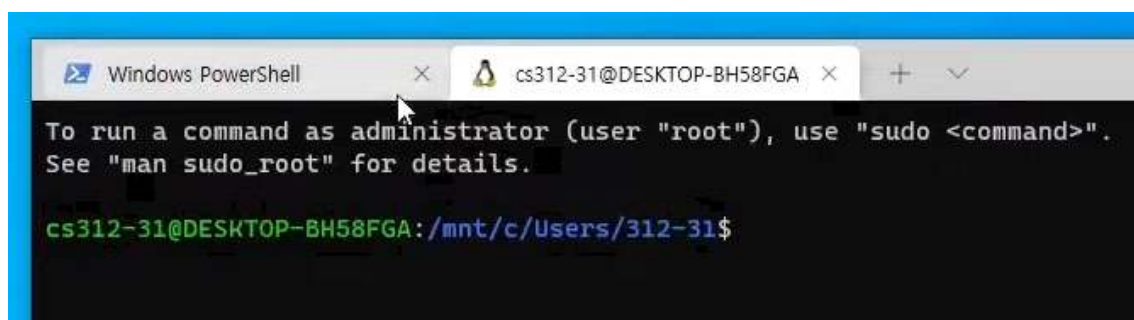
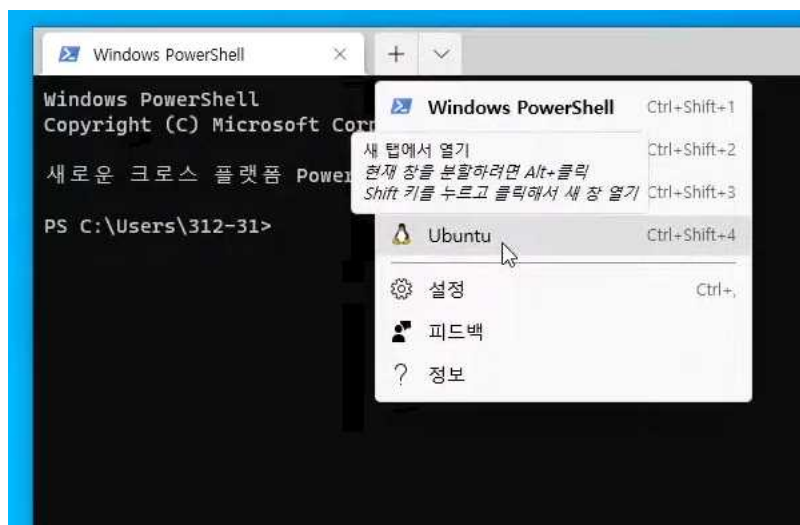
Installing, this may take a few minutes...

이후 조금 시간이 지난 후에 UNIX ID와 비밀번호를 정해주는 칸이 나온다.
설정하면 다음과 같은 창이 뜬다.

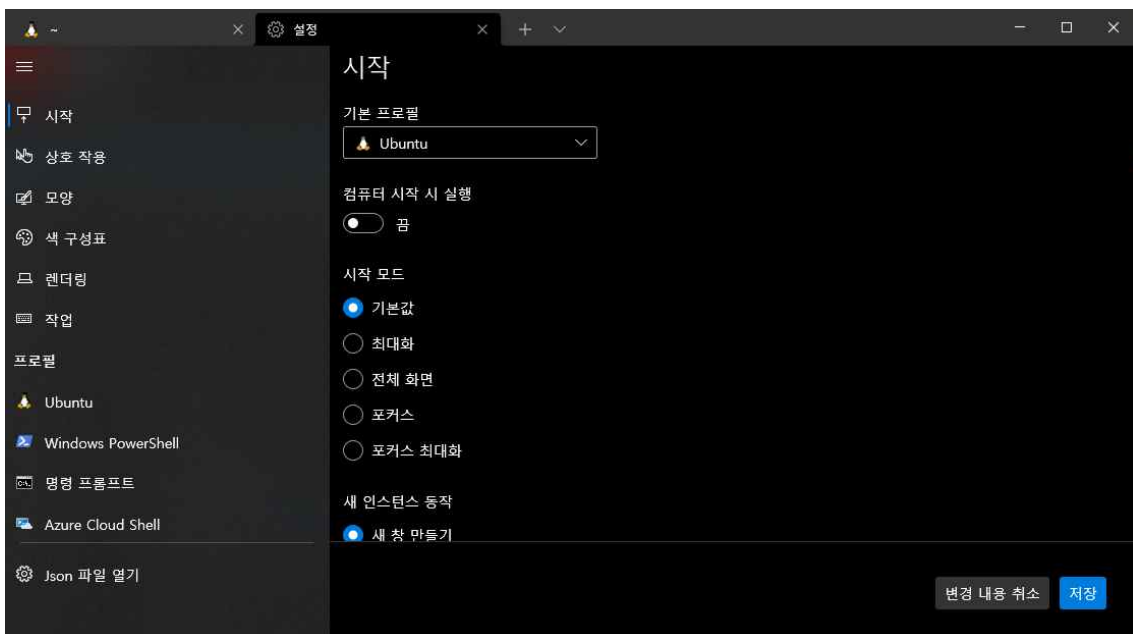
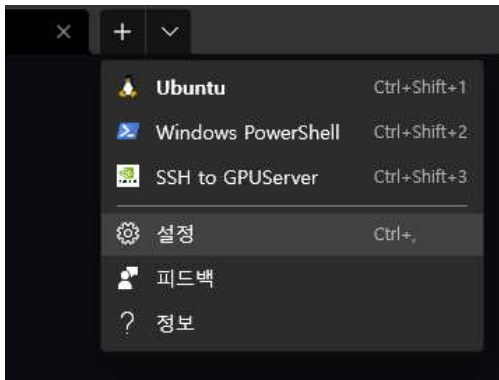
```
cs312-31@DESKTOP-BH58FGA: ~  
passwd: password updated successfully  
Installation successful!  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.72-microsoft-standard-WSL2 x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
System information as of Mon Jul  5 14:55:25 KST 2021  
  
System load:  0.05          Processes:            8  
Usage of /:   0.4% of 250.98GB   Users logged in:    0  
Memory usage: 1%           IPv4 address for eth0: 172.31.88.191  
Swap usage:   0%  
  
1 update can be installed immediately.  
0 of these updates are security updates.  
To see these additional updates run: apt list --upgradable  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
This message is shown once a day. To disable it please create the  
/home/cs312-31/.hushlogin file.  
cs312-31@DESKTOP-BH58FGA:~$
```

다음과 같은 화면이 나오면 성공이다.

이제 Windows Terminal 을 켜보면, 펭귄이 한마리 추가되어있을 것이다.

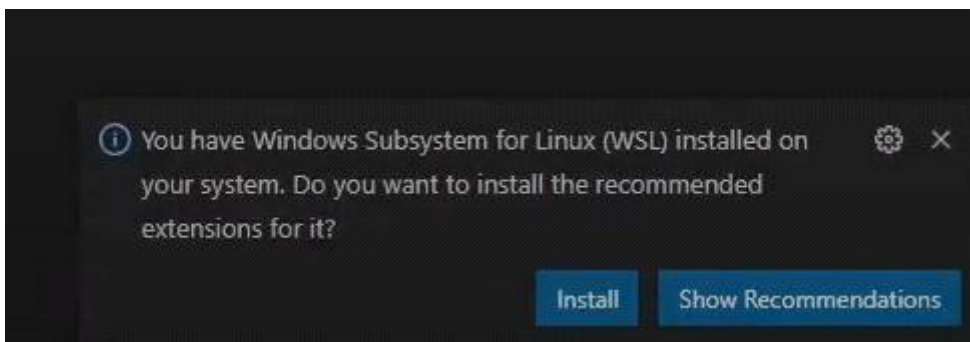


여러분의 컴퓨터에 ubuntu 를 설치했다. 하나의 운영체제이기 때문에, 일반적으로 윈도우가 깔려있는 c드라이브 아래층(Windows Subsystem)에 설치되어있다. 맨 처음에 PowerShell 이 떠서 불편할 수 있으므로, Ubuntu 로 바꿔주도록 하겠다.

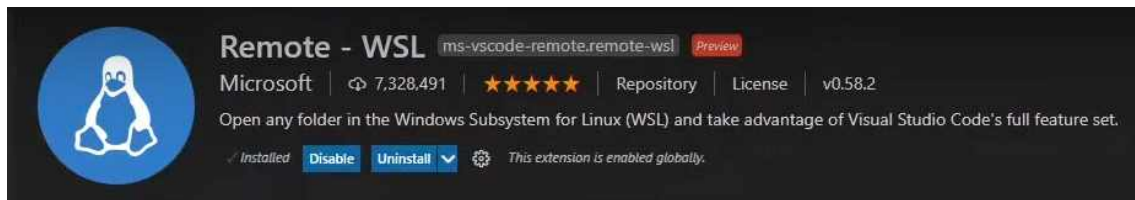


기본 프로필을 Ubuntu로 바꿔주고 저장하자.

1-5. Remote WSL



VSCode를 통해 WSL 에 접근할 수 있다. VSCode를 키면 다음과 같이, WSL 이 설치되어 있기 때문에 관련 익스텐션을 설치할것인지 묻는다. install 눌러주자.

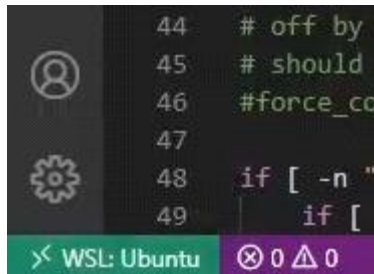


install 이 완료된 모습.

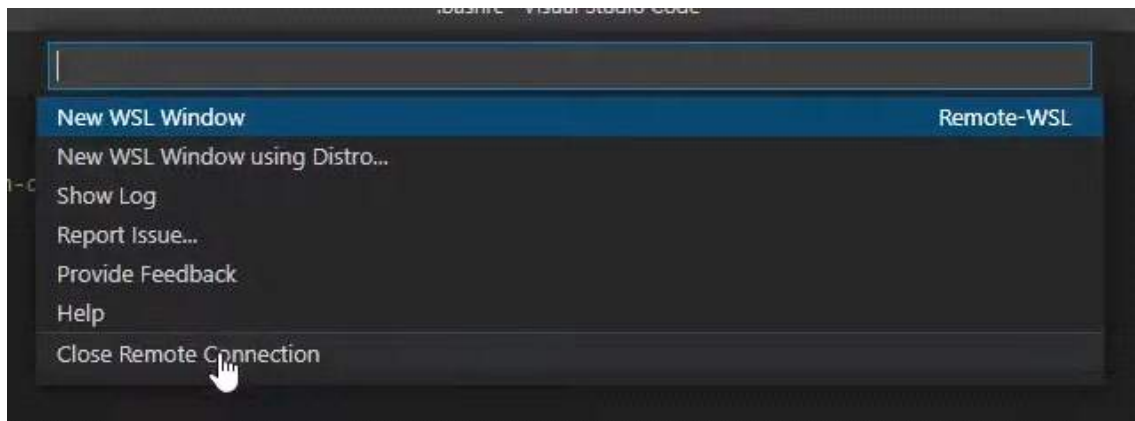
Windows Terminal 에서 Ubuntu 에 들어가서 code 라고 입력하면, 다시 필요한 프로그램을 다운로드 받을 것이다.

```
cs312-31@DESKTOP-BH58FGA:~$ code
Installing VS Code Server for x64 (507ce72a4466fbb27b715c3722558bb15afa9f48)
Downloading: 100%
Unpacking: 100%
Unpacked 1951 files and folders to /home/cs312-31/.vscode-server/bin/507ce72a4466fbb27b715c3722558bb15afa9f48.
```

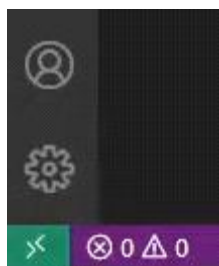
그러면 vscode 가 WSL 환경에서 실행된다.



VSCode 왼쪽 아래에 보면 WSL: Ubuntu 라고 뜨는데, 지금의 VSCode는 WSL 을 위한, 즉, 윈도우가 아닌 Ubuntu 를 위한 VSCode이다. 저 초록색을 클릭하면



다음과 같은 창이 뜬다. Close Remote Connection을 클릭하면



왼쪽 아래를 봤을 때 이전 다음과 같은 모습으로 바뀌었다. 이것은 윈도우를 위한 VSCode이다. 그렇기에, 일부 익스텐션(확장)의 경우, 설치 시 윈도우에서의 VSCode와 Ubuntu에서의 VSCode 둘 다 설치해줘야 동작한다.

1-6. oh-my-zsh

리눅스를 조작할 때 쓰는 프로그램인 shell은 여러가지 종류가 있다. 기본적으로 bash라는 걸 사용하는데, 조금 불편하다. 그래서 사용자 편의성을 높인 oh-my-zsh을 설치해 주겠다.

ubuntu 터미널 키고,

```
$ sudo apt update
```

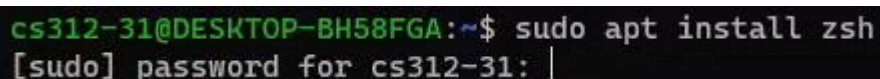
* 앞으로 터미널에 들어갈 코드는 다음과 같이, 맨 앞에 달러 표시를 붙여줄 것이다. 달러 표시는 제외하고 명령어를 입력하면 된다.

ubuntu apt 패키지 리스트를 최신화하는 명령이다. apt는 Microsoft Store 같은, 패키지 매니저이다. 몇 년 전까진 apt-get이었는데 지금은 apt를 표준으로 사용한다. apt-get을 쓰더라도 잘 작동한다. 내가 가지고 있는 패키지 리스트와 서버의 패키지 리스트를 비교해서, update하는 것이다. 그러나 이것은 리스트 update일 뿐, 실제 프로그램 업데이트를 하고 싶으면 upgrade 명령을 써야한다.

맨 앞에 쓰인 sudo는 무슨 뜻일까? 이것은 관리자, 즉 super user로 실행시킨다는 의미이다. 리눅스에서 사용자 전체에 영향을 끼치는 중대한 작업이 필요할 때 맨 앞에 sudo 명령어를 쓴다. 리눅스는 여러 사용자가 원격으로 접속해서 쓰는 컴퓨터이다. apt 리스트를 업데이트할 경우, 다른 사용자가 영향을 받을 수 있기 때문에 sudo를 사용한다. sudo가 맨 앞에 쓰이면 항상 내가 하고자 하는 작업을 해도 되는지 신중하게 생각하는 습관이 필요하다.

업데이트가 끝나면 다음 명령을 실행시킨다.

```
$ sudo apt install zsh
```



```
cs312-31@DESKTOP-BH58FGA:~$ sudo apt install zsh
[sudo] password for cs312-31: |
```

입력하면 비밀번호를 요청한다. 여러분이 UNIX 사용자 생성 시 입력한 비밀번호를 쓰면 된다.

명령어를 해석해보면,

sudo	super user 의 권한으로 ~~~ 실행.
apt	apt 패키지 매니저 실행
install	설치하라
zsh	zsh을.

그러면 zsh 설치가 진행될 것이다.

```
cs312-31@DESKTOP-BH58FGA:~$ sudo apt install zsh
[sudo] password for cs312-31:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  zsh-common
Suggested packages:
  zsh-doc
The following NEW packages will be installed:
  zsh zsh-common
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 4450 kB of archives.
After this operation, 18.0 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

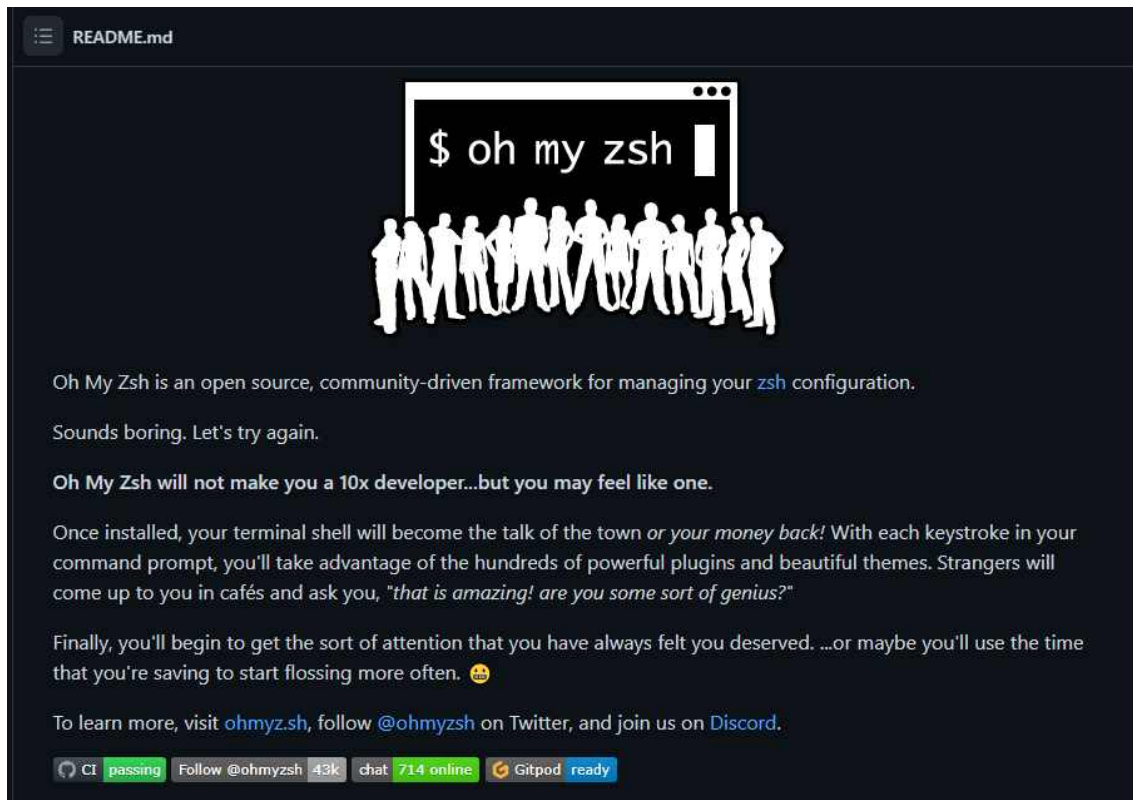
계속 진행할것인지를 묻기에, y를 입력해주자.

이제 zsh을 업그레이드한, oh-my-zsh을 설치할 것이다.

구글검색 oh-my-zsh github

<https://github.com/ohmyzsh/ohmyzsh>

이 사이트로 들어가야한다.



스크롤 내려보면 Getting Started 부분에 이것을 찾을 수 있다.

Basic Installation

Oh My Zsh is installed by running one of the following commands in your terminal. You can install this via the command-line with either `curl`, `wget` or another similar tool.

Method	Command
<code>curl</code>	<code>sh -c "\$(curl -fsSL https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"</code>
<code>wget</code>	<code>sh -c "\$(wget -O- https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"</code>
<code>fetch</code>	<code>sh -c "\$(fetch -o - https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"</code>

위의 `curl` 부분을 복사해서 터미널에 붙여넣어주자.

```
cs312-31@DESKTOP-BH58FGA:~$ sh -c "$(curl -fsSL https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"
Cloning Oh My Zsh...
Cloning into '/home/cs312-31/.oh-my-zsh'...
remote: Enumerating objects: 1239, done.
remote: Counting objects: 100% (1239/1239), done.
remote: Compressing objects: 100% (1204/1204), done.
remote: Total 1239 (delta 20), reused 1103 (delta 15), pack-reused 0
Receiving objects: 100% (1239/1239), 863.71 KiB | 2.22 MiB/s, done.
Resolving deltas: 100% (20/20), done.
Looking for an existing zsh config...
Using the Oh My Zsh template file and adding it to ~/.zshrc.
Time to change your default shell to zsh:
Do you want to change your default shell to zsh? [Y/n]
```

여기서, default shell 을 zsh 로 바꿀것인지 묻는다. y를 입력하면 비밀번호를 입력하고 zsh 이 실행된다.

```
Do you want to change your default shell to zsh? [Y/n] y
Changing the shell...
Password:
chsh: PAM: Authentication failure
Error: chsh command unsuccessful. Change your default shell manually.

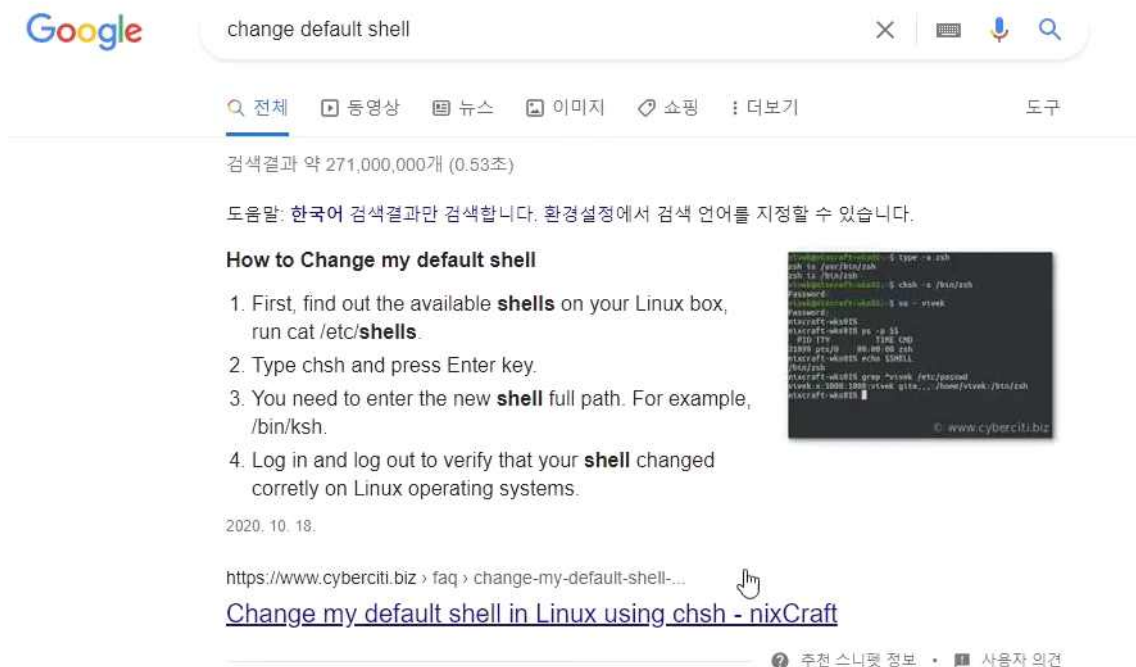
...is now installed!

Before you scream Oh My Zsh! please look over the ~/.zshrc file to select plugins, themes, and options.

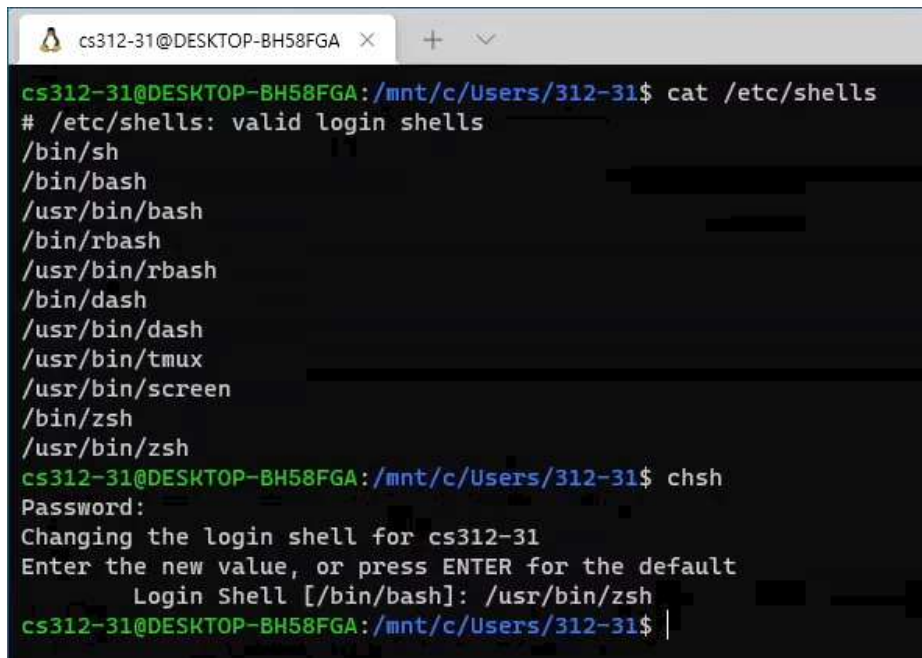
• Follow us on Twitter: https://twitter.com/ohmyzsh
• Join our Discord server: https://discord.gg/ohmyzsh
• Get stickers, shirts, coffee mugs and other swag: https://shop.planetargon.com/collections/oh-my-zsh
```

zsh 은 실행되었지만 비밀번호를 틀리게 쓴 경우, 수동으로 default shell을 바꿔줘야 한다.

구글 검색 change default shell



구글검색을 통해 shell 바꾸는 방법을 알아본 결과, 다음과 같이 진행하면 된다.



```
cs312-31@DESKTOP-BH58FGA: /mnt/c/Users/312-31$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/usr/bin/bash
/bin/rbash
/usr/bin/rbash
/bin/dash
/usr/bin/dash
/usr/bin/tmux
/usr/bin/screen
/bin/zsh
/usr/bin/zsh
cs312-31@DESKTOP-BH58FGA: /mnt/c/Users/312-31$ chsh
Password:
Changing the login shell for cs312-31
Enter the new value, or press ENTER for the default
  Login Shell [/bin/bash]: /usr/bin/zsh
cs312-31@DESKTOP-BH58FGA: /mnt/c/Users/312-31$
```

\$ cat /etc/shells

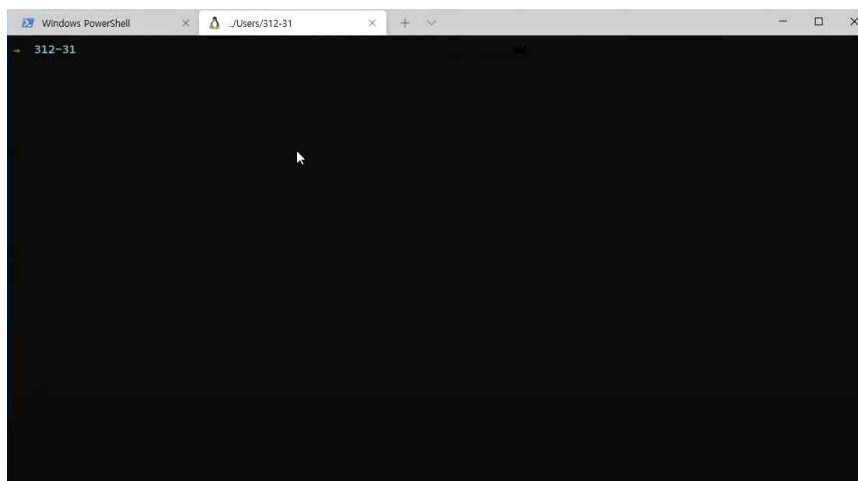
내가 무슨 shell을 가지고 있는지 확인

\$ chsh

chsh 이라는 “프로그램” 실행. 비밀번호를 입력 후 내가 쓰고싶은 shell의 경로를 입력해주면 되는데, /usr/bin/zsh 을 입력해주면 된다.

* 리눅스에서 맨 앞에 쓰이는 명령은 모두 프로그램이다. 심지어 sudo 역시도 프로그램이며, 늘상 쓰게 되는 pwd, ls, mkdir, rm 등의 기본 명령어들도 사실은 프로그램이다.

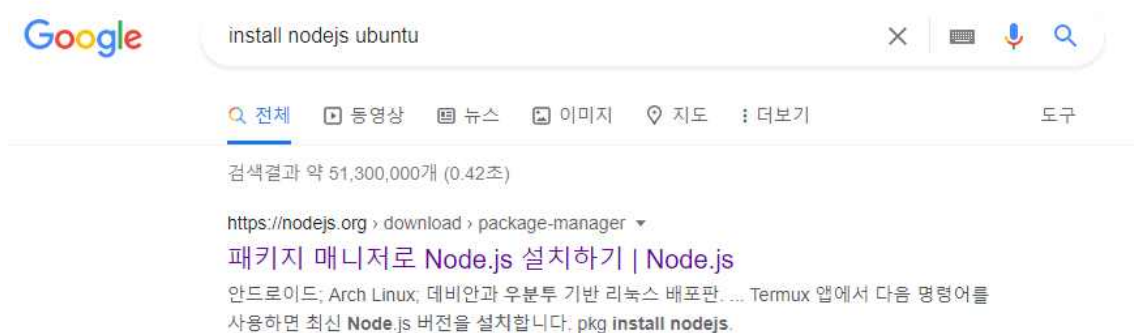
터미널을 꺾다가 키면 다음과 같이 뜰 것이다.



1-7. node.js

여러분의 컴퓨터에 node.js를 설치할 것인데, 윈도우가 아닌 ubuntu 에 설치하도록 하겠다. node.js는 서버에서 실행되는 JavaScript 이다. 무작정 `sudo apt install nodejs` 라고 치지 말자. 리눅스에서 뭔가를 설치할 땐 항상, 어떻게 설치하는지 구글링해보고 설치해야한다.

구글 검색 `install nodejs ubuntu`



패키지 매니저로 Node.js 설치하기

Note: 이 페이지에 나오는 패키지는 각 패키지 관리자가 관리하고 Node.js 코어 팀이 **관리하지 않습니다**. 이슈가 있다면 패키지 관리자에게 보고해 주세요. 해당 이슈가 Node.js 자체의 버그라면 관리자가 이슈를 Node.js에 보고할 것입니다.

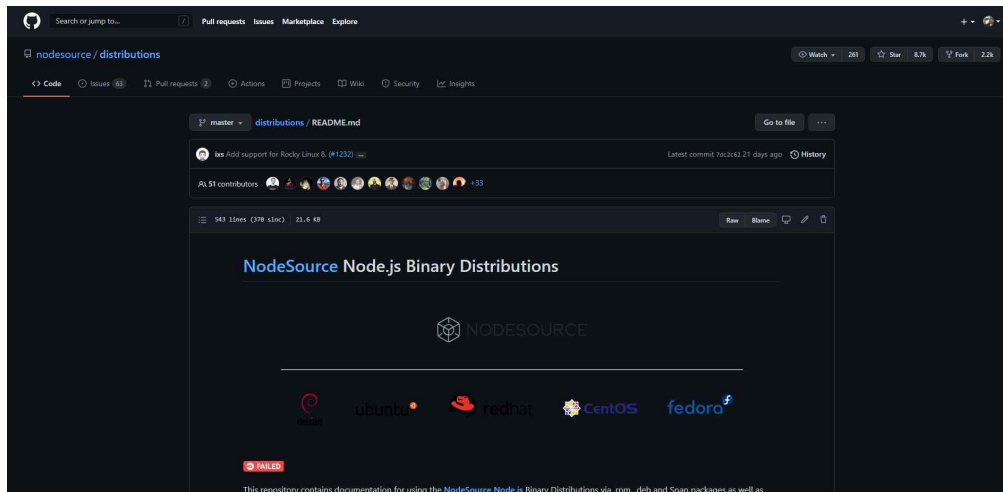
- 안드로이드
- Arch Linux
- 데비안과 우분투 기반 리눅스 배포판, 엔터프라이즈 리눅스/페도라와 Snap 패키지
- FreeBSD
- Gentoo
- IBM i
- NetBSD
- nvm
- nvs
- OpenBSD
- openSUSE 와 SLE
- macOS
- SmartOS 와 illumos
- Solus
- Void Linux
- Windows

리눅스는 Ubuntu 이외에 여러 배포판이 있기 때문에, 그에 맞게 설치해야 한다. 데비안과 우분투 기반 리눅스 배포판 ~~~ 을 클릭하자.

패키지

공식 Node.js 바이너리 배포판은 NodeSource가 제공합니다.

초록색 글씨를 클릭하자.



node.js 의 여러 버전의 binary source를 제공하는 github 페이지다. 스크롤 내려서 다음 부분을 찾자.

Node.js LTS (v14.x):

```
# Using Ubuntu
curl -fsSL https://deb.nodesource.com/setup_lts.x | sudo -E bash -
sudo apt-get install -y nodejs

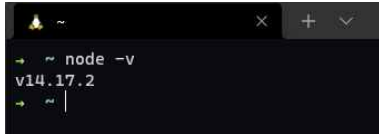
# Using Debian, as root
curl -fsSL https://deb.nodesource.com/setup_lts.x | bash -
apt-get install -y nodejs
```

현재 기준으로 LTS 버전은 14 이다. LTS는 Long-Term support 의 약자로, 최신버전은 아니지만 장기적으로 지원되는 안정된 버전이다.

Using Ubuntu 아래 두 줄을 복사해서 터미널에 붙여넣어 node.js를 설치하자.

설치 이후에 다음 명령어를 입력해보자.

```
$ node -v
```


A terminal window with a dark background. The prompt is a green arrow. The command entered is 'node -v' and the output is 'v14.17.2'.

```
➔ ~ node -v
v14.17.2
➔ ~ |
```

node.js의 현재 버전이 출력되었다. 설치 성공이다. 보통 패키지 설치하고나면 이렇게 버전체크를 해서 설치가 잘 되었는지 확인한다.
node.js 설치하면 npm도 같이 설치된다.

```
$ npm -v
```

입력하여 npm 버전을 확인해보자.

2. 기본적인 리눅스 명령어

리눅스로 개발을 진행하기 전, 기본적으로 알아야 할 명령어들에 대해 연습해보자. 명령어를 축약해놓아서 암기하기 꽤 어려운데, 무엇이 축약인지 이해하는것이 필요하다.

`pwd`

print working directory. 터미널에서 현재 위치를 알려준다.

~

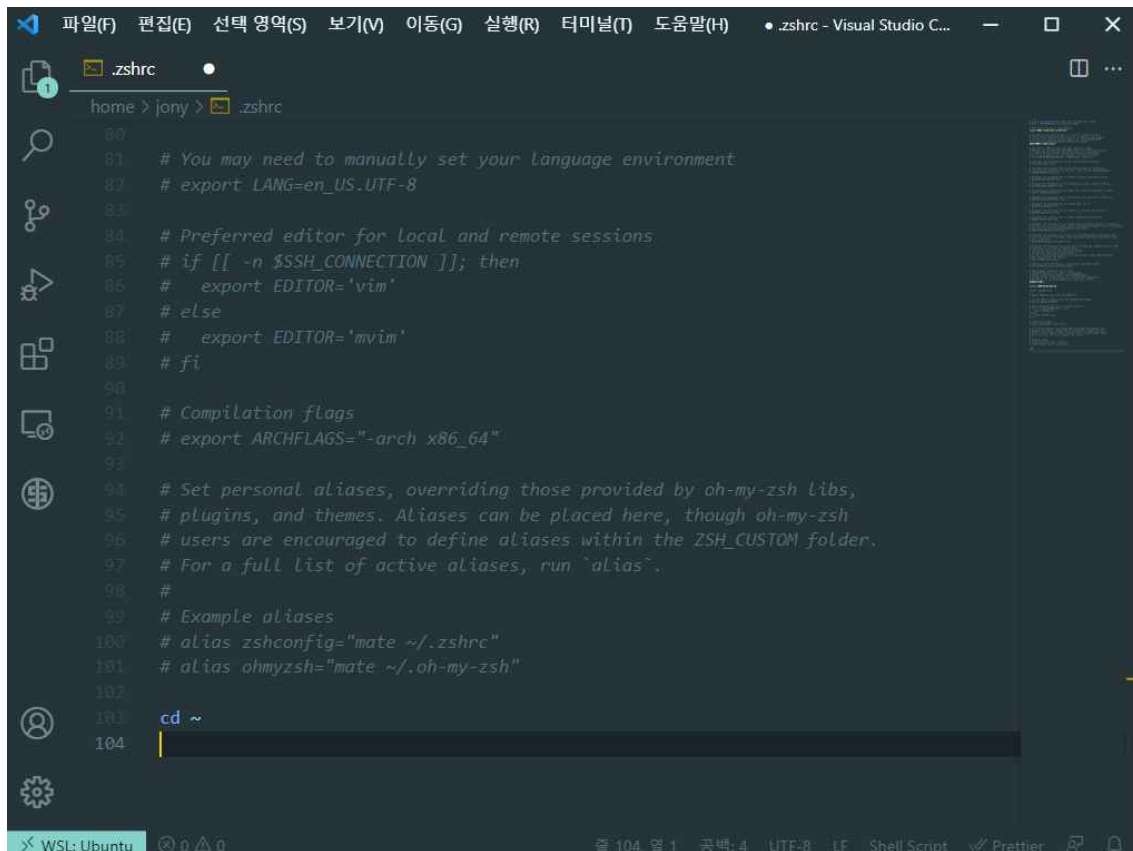
ubuntu 의 `home/username` 디렉터리, 즉 리눅스 홈 디렉터리로 이동한다.

디렉터리는 폴더와 같은 뜻이다. 홈 디렉터리 안에서 일어나는 작업은 `sudo` 명령어가 필요없다. 다른 사용자에게 영향을 끼치지 않기 때문이다. 여러분이 설치한 각종 프로그램의 설정파일이 위치한 디렉터리이다. 리눅스에서 여러명의 user를 생성하면 각각의 user 마다 홈 디렉터리가 부여되어, 각자의 환경설정을 할 수 있다.

* 현재 시작 디렉터리는 기본적으로 `C:/Users/username` 으로, 즉 윈도우 홈 디렉터리로 잡혀있는데, 이것을 리눅스 홈 디렉터리로 바꾸고 싶다면 `zsh` 설정파일인 `.zshrc` 를 바꾸면 된다.

\$ ~

\$ `code .zshrc`



```
home > jony > .zshrc
80
81 # You may need to manually set your language environment
82 # export LANG=en_US.UTF-8
83
84 # Preferred editor for local and remote sessions
85 # if [[ -n $SSH_CONNECTION ]]; then
86 #   export EDITOR='vim'
87 # else
88 #   export EDITOR='mvim'
89 # fi
90
91 # Compilation flags
92 # export ARCHFLAGS="-arch x86_64"
93
94 # Set personal aliases, overriding those provided by oh-my-zsh libs,
95 # plugins, and themes. Aliases can be placed here, though oh-my-zsh
96 # users are encouraged to define aliases within the ZSH_CUSTOM folder.
97 # For a full list of active aliases, run `alias`.
98 #
99 # Example aliases
100 # alias zshconfig="mate ~/.zshrc"
101 # alias ohmyzsh="mate ~/.oh-my-zsh"
102
103 cd ~
104
```

맨 아래에

`cd ~`

를 추가하면 된다. 이후 터미널을 키면 리눅스 홈 디렉터리인 `/home/username` 에서 시작할 것이다.

`/`

루트(root) 디렉터리로 이동한다. 여러분의 컴퓨터의 가장 밑바닥으로 이동하는 것이다. 리눅스 사용자 전체에 영향을 미치는 작업을 할 때 접근하게 된다.

`ls`

list segments. 이건 축약의 뜻을 외우는게 더 어렵다. 안경 낀 사람이 제대로 보고싶으면? 라식(ls)을 하면 된다! 현재 디렉터리에 있는 파일, 하위 디렉터리를 보여준다. 숨김파일까지 확인하고 싶다면 `-a` 를, 권한 등을 포함한 상세 정보를 알고싶으면 `-l` 옵션을 사용한다. 둘을 결합해서 `ls -al` 이라고 입력해보자.

`$ ls -al`

```
→ ~ ls -al
total 204
drwxr-xr-x 10 jony jony 4096 Jul 19 17:22 .
drwxr-xr-x  3 root root 4096 Jun 23 13:51 ..
lrwxrwxrwx  1 jony jony   30 Jun 23 15:39 .aws -> /mnt/c/Users/조교행님/.aws
lrwxrwxrwx  1 jony jony   32 Jun 23 15:39 .azure -> /mnt/c/Users/조교행님/.azure
-rw-----  1 jony jony  156 Jul 19 15:50 .bash_history
-rw-r--r--  1 jony jony  220 Jun 23 13:51 .bash_logout
-rw-r--r--  1 jony jony 3771 Jun 23 13:51 .bashrc
drwxr-xr-x  5 jony jony 4096 Jun 25 13:41 .cache
drwxr-xr-x  4 jony jony 4096 Jun 23 15:39 .docker
drwxr-xr-x  2 jony jony 4096 Jun 23 13:51 .landscape
-rw-r--r--  1 jony jony    0 Jun 23 13:51 .motd_shown
-rw-----  1 jony jony   79 Jun 27 21:39 .node_repl_history
drwxr-xr-x  5 jony jony 4096 Jun 24 17:35 .npm
drwxr-xr-x 12 jony jony 4096 Jun 23 13:52 .oh-my-zsh
-rw-r--r--  1 jony jony  807 Jun 23 13:51 .profile
-rw-r--r--  1 jony jony   10 Jun 23 13:52 .shell.pre-oh-my-zsh
-rw-r--r--  1 jony jony    0 Jun 23 13:51 .sudo_as_admin_successful
drwxr-xr-x  3 jony jony 4096 Jun 25 13:27 .vim
-rw-----  1 jony jony 8317 Jul 14 16:17 .viminfo
-rw-r--r--  1 jony jony 3415 Jun 25 13:36 .vimrc
drwxr-xr-x  5 jony jony 4096 Jun 23 14:01 .vscode-server
-rw-r--r--  1 jony jony  226 Jul 19 17:17 .wget-hsts
-rw-r--r--  1 jony jony 49053 Jun 23 15:30 .zcompdump
-rw-r--r--  1 jony jony 50370 Jul  8 09:39 .zcompdump-DESKTOP-4S6IR37-5.8
-rw-----  1 jony jony 16246 Jul 19 17:22 .zsh_history
-rw-r--r--  1 jony jony  3704 Jun 24 17:19 .zshrc
drwxr-xr-x  3 jony jony 4096 Jul 14 16:29 jonyDev
→ ~ |
```

* 리눅스에서 .으로 시작하는 파일은 숨김파일이다. 윈도우에선 확장자를 쓸때나 사용하지만, 리눅스에서 파일 확장자는 큰 의미가 없다. 특히 리눅스에선 모든 것이 파일로 이루어져있다. 심지어 디렉터리도 파일이다. 그래서 정확한 파일 형태를 보려면 확장자가 아니라 file 명령어를 사용한다.

cd

change directory. 위치 이동할 때 사용한다. 탭(tab)키와 결합해 자주 사용한다. 리눅스에선 디렉터리 이동에 익숙해져야한다.

먼저 절대경로와 상대경로를 복습해보자.

절대경로: 루트가 기준. / 로 시작

상대경로: 내 현재 위치가 기준. / 없이 시작

```
$ cd /ho
```

라고 쓴 다음 탭키를 누르면,

```
$ cd /home/
```

으로 바뀔 것이다. 즉, 루트(/) 에서 ho 로 시작하는 디렉터를 찾아 하나밖에 없으면 그것에 맞게 자동완성해준다. 만약 두 개 이상이라면 그 중에서 탭으로 골라 엔터치면 된다.

그리고 이 상태에서 탭을 또 치면, 이 안에서 들어갈 수 있는 디렉터리로 자동완성해준다. 탭을 자주 사용하는 습관은 매우 좋은데, 오타를 방지할 수 있고, 코딩 속도를 늘릴 수 있기 때문이다.

현재 위치에서 부모 디렉터리로 가고 싶은 경우는 `..` 을 사용한다.

```
$ cd ..
```

내 위치 기준에서, `/` 으로 시작하지 않았으니 상대경로이며, 부모디렉터리로 이동한다.

참고로 `.` 하나는 현재 위치를 의미한다.

`mkdir`

make directory. 현재 위치에 디렉터를 만든다.

```
$ ~
```

```
$ mkdir test_directory
```

```
$ cd test_directory
```

```
$ pwd
```

`rm`

remove. 파일을 삭제할 때 사용한다. `rm` 뒤에 현재 디렉터리에 위치한 파일 이름이나, 파일 경로를 입력한다.

`rm -rf`

remove -recursive -force. 디렉터를 삭제할 때 사용한다. `-r`, recursive 는 재귀라는 뜻인데, 삭제하고자 하는 디렉터리의 밑으로 밑으로 밑으로 밑으로 들어가서 안에 있는걸 다 삭제하고 다시 위로 올라와서 다 삭제하고 위로 위로 올라오는 것이다. `-f`, force 는 삭제 과정 중 삭제할건지 계속 묻지 않는, 강제 삭제 옵션이다.

`cat`

concatenate. 영어사전에선 사슬같이 있다, 연쇄시키다라는 뜻인데 왜 이딴식으로 이름지었는지 모르겠다. 고양이가 코드를 본다고 외우자. 이것은 파일의 코드를 간단하게 확인할 때 쓰는 명령이다.

```
$ ~
```

```
$ cat .zshrc
```

code

VSCode 로 파일을 열 때 사용한다.

\$ ~

\$ code .zshrc

*나의 리눅스 홈 디렉터리, 또는 윈도우 홈 디렉터리에 있는 파일을 수정할 때는 수정이 자유롭지만, 모든 사용자에게 영향을 주는 중요 파일을 수정할 때는 code가 아닌 vi 명령어를 사용해야한다.

clear

명령어를 입력하다보면 화면이 더러워진다. clear 라고 입력하면 터미널이 깔끔해진다.

↑↓

화살표를 사용하면 내가 이전에 썼던 명령어를 불러올 수 있다.

일단은 이 정도만 알아두자. 깊게 들어가면 엄청나게 어려워지기 때문에, 당장 쓰는 명령들만 정리해보았다.

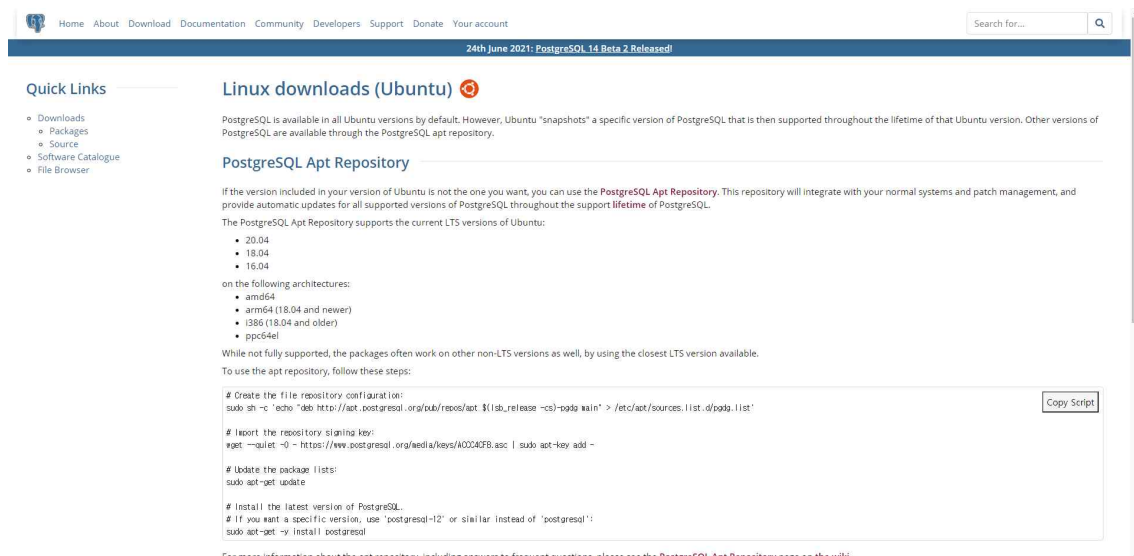
3. PostgreSQL, TablePlus 설치

3-1. PostgreSQL 설치

이제 PostgreSQL을 설치할 것이다.

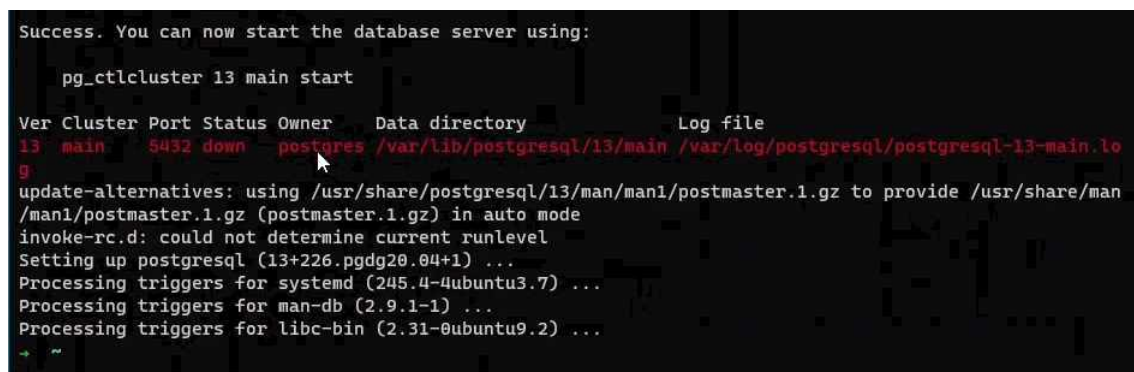
구글 검색 install postgresql ubuntu

한국어 검색결과를 거르고, postgres 공식 사이트에 접속하자.



버튼 눌러서 복사한 후, 터미널에 붙여넣어 Ubuntu 에 설치해주도록 하자.

* 구글 번역을 사용할 경우, 코드 복사를 할 땐 번역을 해제한 후에 복사해야
에러가 뜨지 않는다.



성공이다. 빨간색은 에러가 아니고, PostgreSQL 에 대한 중요한 정보들을 한번
읽으라는 것이다.

지금 중요한 건 세가지다.

버전은 13, 포트는 5432, 오너는 postgres

포트(port)는 항구, 또는 공항을 의미한다. 외부에서 주소를 입력한다고 가정하면,

http://조교행님.org

이 경우, 조교행님.org 라는 “컴퓨터에” 접속하는 것이다. 그 컴퓨터에서 제공하는 index.html 을 보게 될 것이고, 사이트가 보이는 것이다. 그런데 그 컴퓨터에 접속해서 “특정 프로그램을 사용”하고 싶을 경우가 있을 것인데, 이 프로그램이 위치하는 곳이 바로 포트다. 각 데이터베이스 제품별로 포트번호가 다른데, PostgreSQL은 기본적으로 5432 포트를 사용한다.

만약, 조교행님 컴퓨터에 접속해서 PostgreSQL 을 사용하고 싶다면?

http://조교행님.org:5432

이렇게 접속한다.

만약 “내” 컴퓨터의 PostgreSQL에 접속하려면 어떻게 해야할까? localhost를 사용한다.

http://localhost:5432

포트는 알겠고, 오너는 무엇일까? 이것은 해당 프로그램, PostgreSQL 의 Ubuntu 사용자가 postgres 임을 의미한다. 설치와 동시에 여러분의 Ubuntu에는 postgres 라는 이름을 가진 사용자가 생성되었다.

postgres라는 사용자가 생성되었으니, 비밀번호를 바꿔주도록 하겠다.

```
$ sudo passwd postgres
```

사용자 비밀번호를 바꾸는 작업은 당연히 중요한 작업이니, sudo 를 맨 앞에 써야한다. passwd 라는 프로그램을 실행시킬 것이고, postgres라는 이름을 가진 유저의 비밀번호를 바꿀 것이다.

```
→ ~ sudo passwd postgres
[sudo] password for jony:
New password:
Retype new password:
passwd: password updated successfully
```

두 번 입력해서 바꿔주었다.

이제 postgres에 접속해보자.

```
$ sudo service postgresql start
```

service 는 PostgreSQL 데이터베이스 서버를 실행시키기 위해 사용한다.

```
→ ~ sudo service postgresql start
* Starting PostgreSQL 13 database server
```

이 명령어는 컴퓨터를 재부팅할때마다 매번 실행해줘야한다. 매번 입력해주는 게 귀찮다면, .zshrc 맨 아래에 이 명령을 추가해주도록 하자. 그러면 터미널이 시작되면서 자동으로 실행될 것이다.

데이터베이스 서버를 켜으니 PostgreSQL에 접속해보자.

```
$ sudo -u postgres psql
```

-u 옵션 다음엔 유저 이름을 입력하는데, postgres 를 쓰고, psql 서버에 접속한다.

```
→ ~ sudo -u postgres psql
psql (13.3 (Ubuntu 13.3-1.pgdg20.04+1))
Type "help" for help.

postgres=# |
```

다음과 같이 나오면 성공이다. 현재 DB 리스트를 출력해보자.

```
\l
```

```
postgres=# \l
          List of databases
  Name      | Owner   | Encoding | Collate | Ctype   | Access privileges
-----+-----+-----+-----+-----+-----
 postgres   | postgres | UTF8     | C.UTF-8 | C.UTF-8 |
 template0  | postgres | UTF8     | C.UTF-8 | C.UTF-8 | =c/postgres +
            |          |          |          |          | postgres=CTc/postgres
 template1  | postgres | UTF8     | C.UTF-8 | C.UTF-8 | =c/postgres +
            |          |          |          |          | postgres=CTc/postgres
(3 rows)

postgres=#
```

현재 총 3개의 데이터베이스가 있다. Ubuntu 유저 이름과 동일한 postgres 데이터베이스가 있고, template0, template1 이 있다.

데이터베이스를 하나 생성해보자. PostgreSQL 에서 실행시키는 명령은 \$ 가 아닌 # 으로 시작하도록 하겠다. 실습 시엔 # 을 떼고 입력하자.

```
# CREATE DATABASE test;
```

test 라는 이름을 가진 DATABASE 를 생성한다. 키워드는 대문자로 쓴다.

PostgreSQL 안에서는 맨 마지막에 세미콜론을 꼭 붙여줘야 실행된다. 엔터를 치거나 탭을 쓰거나 스페이스 두 칸 이상을 치는 것은 개발자의 편의를 위한 것이고, PostgreSQL 안에서 명령의 종료는 항상 세미콜론이다.

```
postgres=# CREATE DATABASE test;
postgres=# |
```

세미콜론을 입력하지 않았을 경우 다음과 같이, = 에서 - 으로 바뀌었는데, 이것은 해당 명령이 끝나지 않았다는 뜻이다. 세미콜론을 입력하면 명령이 실행된다.

```
postgres=# CREATE DATABASE test;
CREATE DATABASE
postgres=# \l

          List of databases
  Name      | Owner   | Encoding | Collate | Ctype   | Access privileges
-----+-----+-----+-----+-----+-----
 postgres   | postgres | UTF8     | C.UTF-8 | C.UTF-8 |
 template0  | postgres | UTF8     | C.UTF-8 | C.UTF-8 | =c/postgres +
            |          |          |          |          | postgres=Ctc/postgres
 template1  | postgres | UTF8     | C.UTF-8 | C.UTF-8 | =c/postgres +
            |          |          |          |          | postgres=Ctc/postgres
 test       | postgres | UTF8     | C.UTF-8 | C.UTF-8 |
(4 rows)

postgres=# |
```

test 라는 이름을 가진 데이터베이스를 생성하고 DB 리스트를 출력해보았다. test 데이터베이스가 정상적으로 만들어진 것을 확인할 수 있다.

이제 이 데이터베이스에 접속해볼 것이다.

```
# \c test
```

이 경우엔 세미콜론을 붙이던 안 붙이던 작동은 된다. \ 로 시작하는 명령의 경우는 굳이 안 붙여도 되는데, 헛갈리면 붙여주자.

```
postgres=# \c test;
You are now connected to database "test" as user "postgres".
test=#
```

test 데이터베이스에 연결한 경우, 맨 앞부분이 postgres 가 아닌 test로 바뀌었다. 즉, 맨 앞은 현재 접속한 DB의 이름이다.

이제 PostgreSQL 에서의 유저 비밀번호를 바꿔줘야한다. 아까 바꾸지 않았는가?
그건 Ubuntu 에서 postgres 라는 유저의 비밀번호를 바꿔준 것이고, 지금 할 것은
PostgreSQL 에서의 유저 비밀번호 변경이다. 어떻게 바꿀까? 직접 찾아보자.

구글 검색 postgres user password change



Stack Overflow 라는 사이트가 보인다. 이것은 코딩계의 네이버 지식인이다.
여러분이 코딩하면서 겪게 되는 수많은 문제들은 “왜만하면 여기 다 있다.”

PostgreSQL: How to change PostgreSQL user password?

Asked 8 years, 9 months ago · Active 3 months ago · Viewed 1.5m times

How do I change the password for PostgreSQL user?

1252

postgresql change-password



Share Improve this question Follow

267



Add a comment

edited May 31 '20 at 15:47
Promise Preston
9,790 ● 6 ● 49 ● 74

asked Oct 4 '12 at 5:45
Saad
20.2k ● 13 ● 41 ● 68

20 Answers

Active Oldest Votes



To login without a password:

1790

`sudo -u user_name psql db_name`



To reset the password if you have forgotten:



`ALTER USER user_name WITH PASSWORD 'new_password';`

Share Improve this answer Follow

edited Dec 29 '20 at 22:31
Rob Bednark
20.4k ● 18 ● 71 ● 102

answered Oct 4 '12 at 5:55
solaimuruganv
20.9k ● 1 ● 16 ● 23

159 This let the clear password in the user's postgresql command history. – greg Oct 4 '12 at 7:42

152 @greg: so delete it: `rm ~/.psql_history` – RickyA Oct 30 '13 at 13:03

58 off topic but if anyone looking for "how to change name of user" than do `ALTER USER myuser RENAME TO newname;` ...for some reason google was pointing me here when I was googling that :) – equivalent8 Apr 14 '14 at 15:58

해당 질문에 대한 추천이 1252, 답변에 대한 추천이 1790 인것을 볼 때, 다른 사람에게도 매우 훌륭한 정보였다는것을 알 수 있다.

일단 첫 줄은 PostgreSQL 에 접속하는 내용이었고, 두번째 라인이 핵심으로 보인다.

ALTER USER 유저이름 WITH PASSWORD '변경하고자하는비밀번호';

```
test=# ALTER USER postgres WITH PASSWORD '111111';
ALTER ROLE
test=#
```

ALTER ROLE 이 뜨면서 성공한것을 확인할 수 있다. 물론, 비밀번호를 이렇게 쉬운 것으로 하면 보안에 심각한 문제가 생긴다. 우리 DB를 연습해보는 학생이므로, 일단은 단순한것으로 지정했다.

이것으로 PostgreSQL에 대한 세팅을 마쳤다.

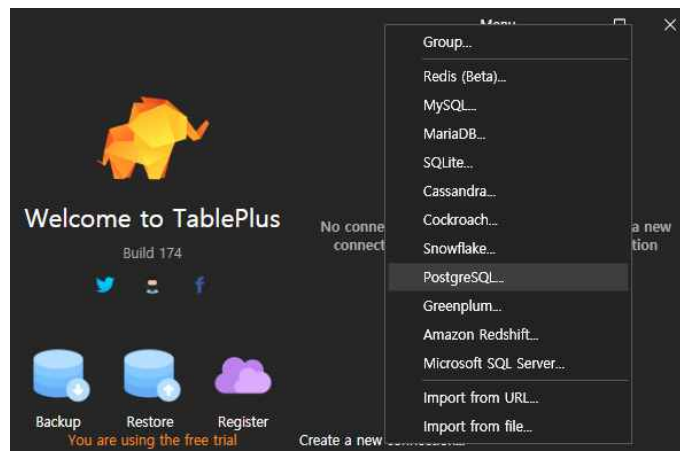
3-2. TablePlus 설치

터미널에서 웬만한 작업은 다 할 수 있지만, 초보자에게는 불편할 수 있으므로, 우리의 삶을 조금 편하게 만들어 줄 툴 하나를 추가적으로 설치해보겠다.
TablePlus 라는 프로그램이다.

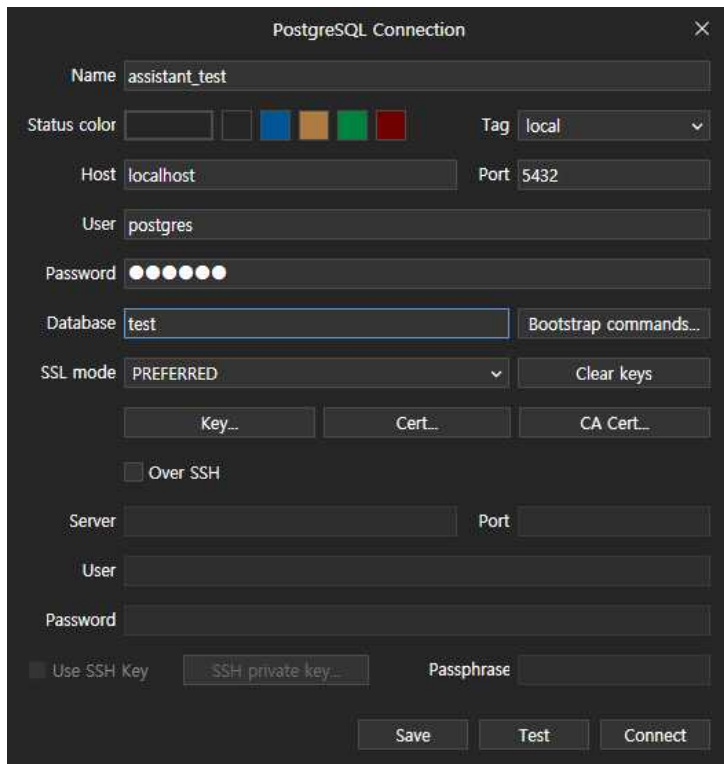
구글 검색 tableplus install



TablePlus 공식 사이트에 접속해서, Windows installer 를 다운로드 받아 설치하자.



Create a new connection 눌러서 PostgreSQL 을 선택한다. TablePlus 는 어느 하나의 DB를 위한 프로그램이 아니다. Cassandra, MariaDB, MySQL, MSSQL, PostgreSQL 을 지원한다. PostgreSQL 을 눌러보자.



The image shows a 'PostgreSQL Connection' dialog box with the following fields and options:

- Name:** assistant_test
- Status color:** A row of five colored squares (white, blue, orange, green, red).
- Tag:** local (dropdown menu)
- Host:** localhost
- Port:** 5432
- User:** postgres
- Password:** A field with six dots indicating a masked password.
- Database:** test
- Bootstrap commands...** (button)
- SSL mode:** PREFERRED (dropdown menu)
- Clear keys** (button)
- Key...** (button)
- Cert...** (button)
- CA Cert...** (button)
- Over SSH** (checkbox, currently unchecked)
- Server:** (empty text field)
- Port:** (empty text field)
- User:** (empty text field)
- Password:** (empty text field)
- Use SSH Key** (checkbox, currently unchecked)
- SSH private key...** (button)
- Passphrase:** (empty text field)
- Save** (button)
- Test** (button)
- Connect** (button)

Name: 내가 쓰고자 하는 DB 이름

Host: 접속하고자하는 DB 주소. 내 컴퓨터의 PostgreSQL 에 접속할 것이기에 localhost 라고 쓴다,

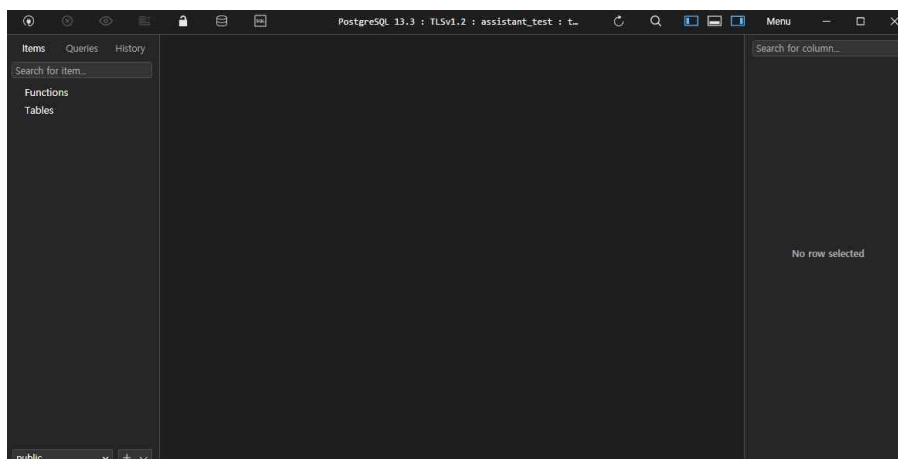
Port: PostgreSQL 기본 포트번호는 5432이다.

User: 사용자 이름이다. postgres 라고 써주자

Password: 설정한 비밀번호

Database: 접속하고자 하는 PostgreSQL 의 DB 이름

그리고 Connect 버튼을 눌러주고, 아무 에러 없이 다음 화면이 뜨면 성공이다.



4. CRUD

지금까지의 과정을 잘 따라왔으면 느꼈겠지만, DB 실습을 위해 우리 컴퓨터를 세팅하는데에만 한참 걸렸다. TypeScript Playground, CodeSandBox 처럼 웹으로 당장 시작할 수 있는 툴들만 써본 여러분들에겐 아주 힘든 시간이었을 것이다. DB 환경 구성만해도 이런데, 나만의 서버를 운용하기 위해선 얼마나 많은 과정이 필요하겠는가!

이제 본격적으로, 우리의 DB 에 데이터를 입력하고, 활용해보는 시간을 가질 것이다. 그 전에 먼저, CRUD (씨알유디, 또는 크루드 라고 발음함) 에 대해 알아야 한다. 이것은 데이터를 다루는 정보과학의 본질이라고 할 수 있는 네가지를 말한다.

C reate	생성
R ead	읽기
U pdate	수정
D elete	삭제

우리의 데이터베이스에 접근해서, 데이터를 생성하고, 읽고, 수정하고, 삭제하는 것이다. 제 아무리 몇천줄, 몇만줄의 코드라도 본질은 CRUD 이다. 회사가 보유한 데이터를 사용자에게 제공하는 것, 그것이 수많은 스타트업의 존재 이유다.

4-1. C: CREATE TABLE

RDB는 여러 개 테이블간의 관계로 이루어져있다. 가장 먼저, 테이블을 생성해야한다.

테이블은 행과 열로 이루어져있다. 손가락으로 가로 세로를 그리면서 다음과 같이 외우자.

가로→ 세로↓ 로우→ 컬럼↓ 행→ 열↓

person					
id	name	gender	date_of_birth	email	create_at
1	조니	남성	1993.03.21	jony123@gmail.com	한국시간 2021년 07월 20일 오후 7시 30분 28초
2	수지니	여성	2000.11.21	sujini33@naver.com	한국시간 2021년 05월 12일 오전 8시 13분 06초
3	무야호할배	남성	1948.02.13		한국시간 2021년 03월 20일 오후 3시 28분 06초

이러한 형태의 테이블을 만든다고 가정해보자. 테이블 생성 시 문법은 다음과 같다.

```
CREATE TABLE person (
  id SERIAL NOT NULL PRIMARY KEY,
  name TEXT NOT NULL,
  gender TEXT NOT NULL,
  date_of_birth DATE NOT NULL,
  email TEXT,
  create_at TIMESTAMPTZ DEFAULT now()
);
```

CREATE TABLE 테이블이름 ();

원하는 테이블이름을 붙여주고 소괄호로 시작해서 소괄호로 끝난 다음에 세미콜론을 붙여준다. 여기서 person 이라는 이름을 가진 테이블을 만들 것이다.

소괄호 안을 살펴보면, 맨 처음 나오는 건 우리가 설정할 컬럼 이름이다. 컬럼은 총 여섯개, id, name, gender, date_of_birth, email, create_at 이다.

각각의 컬럼명 바로 옆엔 데이터 타입이 나온다. 그 옆엔, 만약 null 값 (한국어로 널값이라고 함)을 허용하고싶지 않을 경우에 NOT NULL 이라고 적는다.

* null 은 의도적으로 비워진 값을 뜻한다. 의도치않게 비워진 undefined 와는 다르다. 예를 들면, 모든 사람의 이메일이 필요하지 않을 수도 있기에 email은 null을 허용하므로 NOT NULL을 쓰지 않았다.

id 의 경우, PRIMARY KEY 라고 썼다. 프라이머리키, PK, 주키 라고도 부르는데, 이것은 해당 테이블에서 중복되지 않는, 고유한 데이터를 식별할 수 있는 값이다.

create_at 처럼, 입력하지 않았을 때 기본값으로 입력될 데이터를 지정하고 싶으면 DEFAULT 라고 쓰고, 해당 값을 옆에 쓴다. now() 라고 쓴 것은 현재 시간을 의미한다. person 데이터 행을 생성할 때, 아무것도 입력하지 않으면 생성된 시간이 입력된다.

각각의 컬럼은 콤마로 구분한다.

PostgreSQL 에서는 굉장히 많은 데이터타입을 가지고 있고, 모든 것을 사전식으로 가르쳐주는것은 무의미하다. 지금 당장 알아야 할 데이터타입은 다음과 같다.

INT	정수 JavaScript에서 number 와 비슷한데, 소수가 아니라 정수만 표현
SERIAL	시퀀스 정수 자동 증가하는 정수.
TEXT	문자열
BOOL	true 또는 false
DATE	날짜
TIMESTAMPZ	타임존이 포함된 날짜 및 시간

SERIAL 은 왜 쓰는 걸까? id의 경우, SERIAL 과 PRIMARY KEY 가 같이 쓰였다. SERIAL 타입의 컬럼은 데이터를 수동으로 입력하지 않을 것이며, 데이터가 채워질수록 자동으로 증가하므로 “중복” 을 피할것이다. PRIMARY KEY 에서 제일 중요한 건, 해당 테이블에 중복된 PRIMARY KEY 가 있으면 안된다는 것이다.

만약 SERIAL 이 아닌, INT 로 설정했기때문에 중복이 생겼다고 가정해보자.

person					
id	name	gender	date_of_birth	email	create_at
1	조니	남성	1993.03.21	jony123@gmail.com	한국시간 2021년 07월 20일 오후 7시 30분 28초
2	수지니	여성	2000.11.21	sujini33@naver.com	한국시간 2021년 05월 12일 오전 8시 13분 06초
3	무야호할배	남성	1948.02.13		한국시간 2021년 03월 20일 오후 3시 28분 06초
3	펑수	남성	2020.12.25		한국시간 2021년 07월 21일 오후 6시 00분 00초

내가 일일이 입력하다보니 이미 3번이 있는걸 잠시 까먹고, 다시 3번을 넣었다. 이 경우 PRIMARY KEY 가 중복되었다. 데이터에 접근할 땐 보통 PRIMARY KEY 로 접근하는데, 클라이언트가 3번 데이터를 요청했다 하면 무야호할배와 펑수 중 어떤 데이터를 가져와야될지 모르게 된다! 만약 알아서 증가하는 SERIAL 타입으로

설정해두었을 경우엔 이런 문제를 생각할 필요가 없는 것이다.

이제 터미널 키고 입력해보자.

```
test=# CREATE TABLE person (  
test(# id SERIAL NOT NULL PRIMARY KEY,  
test(# name TEXT NOT NULL,  
test(# gender TEXT NOT NULL,  
test(# date_of_birth DATE NOT NULL,  
test(# email TEXT,  
test(# create_at TIMESTAMPTZ DEFAULT now()  
test(# );  
CREATE TABLE  
test=# \d  
  
              List of relations  
Schema | Name          | Type      | Owner  
-----+-----+-----+-----  
public | person        | table     | postgres  
public | person_id_seq | sequence  | postgres  
(2 rows)  
  
test=#
```

\d 를 입력해 현재 데이터베이스인 test 의 relations 을 확인할 수 있다. 당장은 테이블과 시퀀스 확인할때만 쓸건데, person 테이블이 생성되었고, person_id_seq 라는 시퀀스도 생성되었다. person의 id 컬럼을 SERIAL PRIMARY KEY 로 지정해주었기 때문에, 자동 증가되는 시퀀스가 생성된 것이다.

\d person 을 입력해 person 테이블의 형태를 확인할 수 있다.

```
test=# \d person  
  
              Table "public.person"  
Column      | Type          | Collation | Nullable | Default  
-----+-----+-----+-----+-----  
id           | integer       |           | not null | nextval('person_id_seq'::regclass)  
name         | text          |           | not null |  
gender       | text          |           | not null |  
date_of_birth | date          |           | not null |  
email        | text          |           |          |  
create_at    | timestamp with time zone |           |          | now()  
Indexes:  
    "person_pkey" PRIMARY KEY, btree (id)  
  
test=# |
```

4-2. C: INSERT INTO

person					
id	name	gender	date_of_birth	email	create_at
1	조니	남성	1993.03.21	jony123@gmail.com	한국시간 2021년 07월 20일 오후 7시 30분 28초
2	수지니	여성	2000.11.21	sujini33@naver.com	한국시간 2021년 05월 12일 오전 8시 13분 06초
3	무야호할배	남성	1948.02.13		한국시간 2021년 03월 20일 오후 3시 28분 06초

이제 각각의 데이터를 입력할 시간이다. 이런 형태의 테이블에 데이터를 추가한다고 가정하면,

```
INSERT INTO person (
    name,
    gender,
    date_of_birth)
VALUES ('조교행님', '남자', DATE '1993-03-21');
```

문법은 다음과 같다. CRUD 의 C 에 해당하지만, CREATE 가 아닌 INSERT INTO 로 시작한다.

데이터를 입력하고 싶은 테이블이름을 쓰고, 역시 소괄호로 시작하며, 입력하고자 하는 컬럼이름을 쓰고 콤마로 구분한다.

다음, VALUES 를 입력하고 소괄호로 시작한다. 입력한 컬럼이름 순서에 맞춰서 작성한다. id 는 시퀀스이기때문에 자동증가할것이므로 입력하지 않았고, 이메일은 입력하지 않았다.

날짜를 입력할 땐, 앞에 DATE 라고 써주면 된다.

소괄호 닫고 세미콜론 찍어주자.

```
test=# INSERT INTO person (
test=# name, gender, date_of_birth)
test=# VALUES ('조교행님', '남자', DATE '1993-03-21');
INSERT 0 1
test=# SELECT * FROM person;
 id |  name  | gender | date_of_birth | email |          create_at
-----+-----+-----+-----+-----+-----
  1 | 조교행님 | 남자   | 1993-03-21   |      | 2021-07-20 15:03:50.424756+09
(1 row)

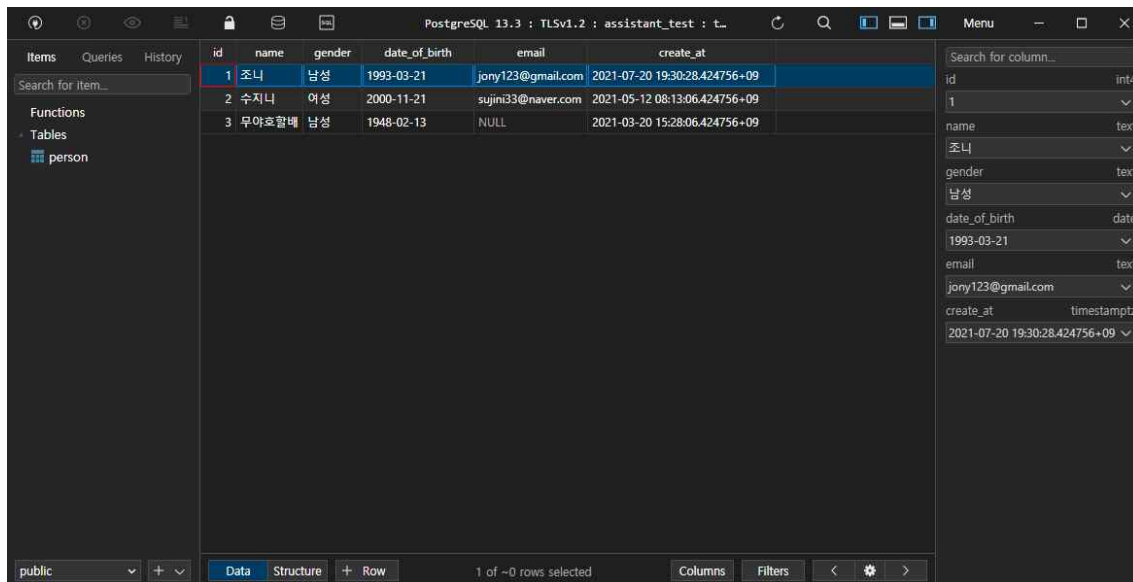
test=# |
```

확인해보자

```
# SELECT * FROM person;
```

데이터가 정상적으로 들어간것을 확인할 수 있다.

이제 어떻게 만드는지 알았으므로, 3개의 행을 더 추가해주도록 하고, TablePlus 에서 F5 를 눌러 새로고침 해보면 다음과 같다.



CRUD 는 TablePlus 에서도 가능하다. 직관적이므로, 따로 가르치진 않겠다. 다만 적용을 할 땐 F5를 눌러서 commit 해야 한다는것만 알아두자.

4-3. R: SELECT

TablePlus 를 통해 데이터베이스를 쉽게 확인할 수 있지만, 터미널에서 쓸 수 있는 기본적인 SQL 명령들은 알아두도록 하자. SQL CRUD 에서 가장 어려운게 바로 R, SELECT 이다. 고수가 되려면 SELECT 를 잘 다룰 줄 알아야 한다.

테이블의 모든 내용 보기

SELECT * FROM 테이블이름;

```
test=# SELECT * FROM person;
 id | name | gender | date_of_birth | email | create_at
-----+-----+-----+-----+-----+-----
  1 | 조니 | 남성 | 1993-03-21 | jony123@gmail.com | 2021-07-20 19:30:28.424756+09
  2 | 수지니 | 여성 | 2000-11-21 | sujini33@naver.com | 2021-05-12 08:13:06.424756+09
  3 | 무야호할배 | 남성 | 1948-02-13 | | 2021-03-20 15:28:06.424756+09
(3 rows)

test=#
```

참고로, * 은 프로그래밍에서 ‘모든 것’을 의미한다. * 은 실무에선 신중하게 사용해야한다. 데이터가 너무 많아서 서버가 멈출수도 있기 때문이다.

열 제한

SELECT 컬럼명, 컬럼명, 컬럼명 FROM person;

```
test=# SELECT name, date_of_birth, email FROM person;
 name | date_of_birth | email
-----+-----+-----
 조니  | 1993-03-21    | jony123@gmail.com
수지니 | 2000-11-21    | sujini33@naver.com
무야호할배 | 1948-02-13    |
(3 rows)

test=# |
```

이름과 생일, 이메일만 출력되도록 했다.

행 제한

SELECT * FROM 테이블이름 WHERE 조건;

여기서 WHERE 을 썼는데, 뒤에 가져오고자 하는 데이터의 조건이 나온다. 실무에선 반드시 써줘야한다. * 을 설명한것과 마찬가지로 이유이다. 1억건의 유저 데이터를 저장하고있는데 WHERE 로 조건을 지정해주지 않으면 서버가 멈춰버릴수도 있다.

아이디가 1 초과인것만.

```
test=# SELECT * FROM person WHERE id > 1;
 id | name | gender | date_of_birth | email | create_at
-----+-----+-----+-----+-----+-----
  2 | 수지니 | 여성 | 2000-11-21 | sujini33@naver.com | 2021-05-12 08:13:06.424756+09
  3 | 무야호할배 | 남성 | 1948-02-13 | | 2021-03-20 15:28:06.424756+09
(2 rows)

test=#
```

아이디가 1인것만. ===(세개)나 ==(두개) 가 아니라 =(한개) 사용

```
test=# SELECT * FROM person WHERE id = 1;
 id | name | gender | date_of_birth | email | create_at
-----+-----+-----+-----+-----+-----
  1 | 조니 | 남성 | 1993-03-21 | jony123@gmail.com | 2021-07-20 19:30:28.424756+09
(1 row)

test=# |
```

아이디가 1 이상인것만.

```
test=# SELECT * FROM person WHERE id >= 1;
 id | name | gender | date_of_birth | email | create_at
-----+-----+-----+-----+-----+-----
  1 | 조니 | 남성 | 1993-03-21 | jony123@gmail.com | 2021-07-20 19:30:28.424756+09
  2 | 수지니 | 여성 | 2000-11-21 | sujini33@naver.com | 2021-05-12 08:13:06.424756+09
  3 | 무야호할배 | 남성 | 1948-02-13 | | 2021-03-20 15:28:06.424756+09
(3 rows)

test=# |
```

이름이 '조니'인것만

```
test=# SELECT * FROM person WHERE name = '조니';
 id | name | gender | date_of_birth | email | create_at
-----+-----+-----+-----+-----+-----
  1 | 조니 | 남성   | 1993-03-21    | jony123@gmail.com | 2021-07-20 19:30:28.424756+09
(1 row)

test=#
```

이제 정렬을 배워보자.

```
# SELECT * FROM 테이블이름 ORDER BY create_at;
```

```
test=# SELECT * FROM person ORDER BY create_at;
 id | name | gender | date_of_birth | email | create_at
-----+-----+-----+-----+-----+-----
  3 | 무야호할배 | 남성 | 1948-02-13 | | 2021-03-20 15:28:06.424756+09
  2 | 수지니 | 여성 | 2000-11-21 | sujini33@naver.com | 2021-05-12 08:13:06.424756+09
  1 | 조니 | 남성 | 1993-03-21 | jony123@gmail.com | 2021-07-20 19:30:28.424756+09
(3 rows)

test=#
```

유저 생성 순으로 정렬했다. 기본값은 ASC, 즉 오름차순인데, 내림차순으로 보고싶다면 맨 뒤에 DESC 을 붙여주면 된다.

```
test=# SELECT * FROM person ORDER BY create_at DESC;
 id | name | gender | date_of_birth | email | create_at
-----+-----+-----+-----+-----+-----
  1 | 조니 | 남성 | 1993-03-21 | jony123@gmail.com | 2021-07-20 19:30:28.424756+09
  2 | 수지니 | 여성 | 2000-11-21 | sujini33@naver.com | 2021-05-12 08:13:06.424756+09
  3 | 무야호할배 | 남성 | 1948-02-13 | | 2021-03-20 15:28:06.424756+09
(3 rows)

test=#
```

혼합해서 사용할수도 있을 것이다.

무야호할배의 이름과 생일 컬럼만 보여주려면?

```
# SELECT name, date_of_birth FROM person WHERE name = '무야호할배';
```

```
test=# SELECT name, date_of_birth FROM person WHERE name = '무야호할배';
 name | date_of_birth
-----+-----
 무야호할배 | 1948-02-13
(1 row)

test=#
```

4-4. U: UPDATE

UPDATE를 사용할 땐, 반드시 WHERE 를 같이 써줘야 한다. 무슨말이냐, 여러분들이 수정할 “대상” 이 무엇인지 지정해주지 않으면 전체 행에서 해당 항목이 전부 바뀐다! 다른 행도 영향을 받을 수 있으므로 꼭 WHERE 를 써주도록 하자.

```
UPDATE person
SET
    name = '조교동생',
    gender = '여자'
WHERE
    id = 1;
```

문법은 다음과 같다. UPDATE 로 시작하고, 테이블이름을 적고 SET 이라고 입력한 후, 바꾸고자 하는 컬럼을 써준다. 여러개면 콤마로 구분하자. 그리고 반드시 WHERE 를 써주고, 무엇에서 바꿀것인지를 정한다.

실습해보자.

```
test=# SELECT * FROM person;
 id |  name  | gender | date_of_birth |      email      |      create_at
-----+-----+-----+-----+-----+-----
  1 | 조니   | 남성   | 1993-03-21    | jony123@gmail.com | 2021-07-20 19:30:28.424756+09
  2 | 수지니 | 여성   | 2000-11-21    | sujini33@naver.com | 2021-05-12 08:13:06.424756+09
  3 | 무야호할배 | 남성 | 1948-02-13    |                  | 2021-03-20 15:28:06.424756+09
(3 rows)

test=#
```

현재의 person 테이블은 다음과 같다. 우린 조니를 리사 로 바꾸고, 남성에서 여성으로 바꿀것이다. WHERE id = 1 을 써주면 될 것이다.

```
# UPDATE person SET name = '리사', gender = '여성' WHERE id = 1;
```

```
test=# UPDATE person SET name = '리사', gender = '여성' WHERE id = 1;
UPDATE 1
test=# SELECT * FROM person;
 id |  name  | gender | date_of_birth |      email      |      create_at
-----+-----+-----+-----+-----+-----
  2 | 수지니 | 여성   | 2000-11-21    | sujini33@naver.com | 2021-05-12 08:13:06.424756+09
  3 | 무야호할배 | 남성 | 1948-02-13    |                  | 2021-03-20 15:28:06.424756+09
  1 | 리사   | 여성   | 1993-03-21    | jony123@gmail.com | 2021-07-20 19:30:28.424756+09
(3 rows)

test=#
```

정상적으로 변경된 것을 확인할 수 있다.

4-5. D: DELETE, DROP

삭제는 항상 파괴적인 결과를 가져오기 때문에, 조심히 다뤄야한다. **삭제하기전에 항상 세번 생각하자! 반드시 WHERE 를 같이 써줘야한다.**

```
test=# SELECT * FROM person;
 id |  name  | gender | date_of_birth | email          | create_at
-----+-----+-----+-----+-----+-----
  2 | 수지니 | 여성   | 2000-11-21    | sujini33@naver.com | 2021-05-12 08:13:06.424756+09
  3 | 무야호 할배 | 남성   | 1948-02-13    |                  | 2021-03-20 15:28:06.424756+09
  1 | 리사   | 여성   | 1993-03-21    | jony123@gmail.com  | 2021-07-20 19:30:28.424756+09
(3 rows)

test=#
```

여기서, 수지니를 삭제해보자.

```
# DELETE FROM person WHERE id = 2;
```

```
test=# DELETE FROM person WHERE id = 2;
DELETE 1
test=# SELECT * FROM person;
 id |  name  | gender | date_of_birth | email          | create_at
-----+-----+-----+-----+-----+-----
  3 | 무야호 할배 | 남성   | 1948-02-13    |                  | 2021-03-20 15:28:06.424756+09
  1 | 리사   | 여성   | 1993-03-21    | jony123@gmail.com  | 2021-07-20 19:30:28.424756+09
(2 rows)

test=#
```

수지니가 삭제된것을 알 수 있다.

테이블 전체를 지우되, 테이블은 남겨두고 싶다면 WHERE 을 사용하지 않는다.

```
# DELETE FROM person;
```

만약 테이블 자체도 없애고 싶다면 DROP 을 사용한다.

```
# DROP TABLE person;
```

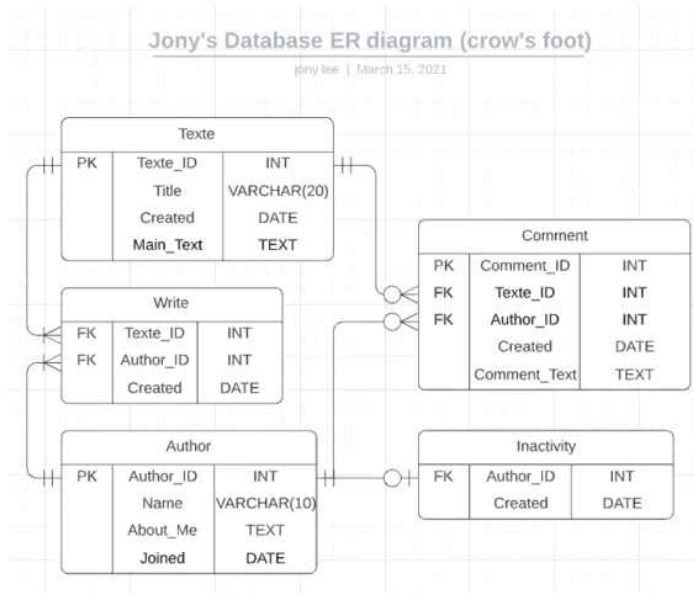
데이터베이스를 없앨 때도 DROP을 사용한다.

```
# DROP DATABASE test;
```

물론 이 명령을 실행시키고 싶다면 \c 으로 접속된 데이터베이스가 test 가 아니어야한다.

5. 관계형 데이터베이스 모델링

관계형 데이터베이스, RDB 는 여러 테이블들의 ‘관계’ 로 이루어져있다.



이런 그림과 같이, 여러 테이블이 서로 연결된 관계로 이루어져 있다는 것이다.
근데 왜 이렇게 나눠놨을까? 하나의 테이블에서 모든 것을 처리하면 안되는걸까?
이것은 두 가지 이유 때문이다.

1. 중복
2. NULL



이러한 UI 를 가진 웹앱을 만든다고 하자. 하나의 테이블만 사용한다면 다음과 같은 형태일 것이다.

사용자ID	사용자닉네임	글ID	글제목	글생성일	글내용	댓글에 해당하는ID	댓글ID	댓글내용	댓글생성일
assistant0603	조교행님	1	조교행님의 첫 글 ㅎㅎ	2012.03.02	우리 같이 열심히 해보자 ㅎㅎ				
sujin02	수지니					1	1	잘 부탁드립니다~	2015.06.01
jjoayo	조아요맨					1	2	좋아요~~	2012.03.02
hope123	코린이	2	리액트 커리큘럼좀	2016.11.11	리액트...어떻게 시작하나요??	2	3	리액트 정말 막막하네요...ㅠㅠ	2016.11.11
hope123	코린이					2	4	혹시 잘 아시는 분 있나요? ㅠㅠ	2016.11.11
sujin02	수지니					2	5	조교행님 유튜브는 어때요?	2016.11.12
assistant0603	조교행님					2	6	제가 바로 조교행님입니다.	2016.11.12

빨간색은 중복을 의미한다. 코린이는 자기 댓글에 또 댓글을 달았기 때문에 중복이 발생했고, 수지니는 옛날에 단 댓글이 있는데 다른 글에 댓글을 달아서 중복이 발생했다. 조교행님은 어떨까? 옛날에 자기가 쓴 글이 있었는데 다른 글에 댓글을 달았기때문에 중복이 발생했다. 현재 사용자 관련 데이터는 사용자ID와 닉네임뿐인데, 비밀번호, 생일, 자기소개, 가입일자 등의 정보가 들어간다면? 쓸데없이 중복되는 정보들은 더 늘어날 것이고, 사용자가 10명만 되어도 중복으로 가득 찰 것이다.

노란색은 NULL 을 의미한다. 코린이는 댓글을 썼기 때문에 데이터행이 추가되었는데, 아무 상관없는 글 ID와 글제목, 글생성일, 글내용까지 NULL 로

추가되었다. 코린이뿐만아니라 다른 데이터도 마찬가지로. 이처럼, 테이블을 하나만 사용할 경우엔 누가봐도 비효율적이다.

글 테이블

글 ID	사용자ID	글제목	글생성일	글내용
1	assistant0603	조교행님의 첫 글 ㅎㅎ	2012.03.02	우리 같이 열심히 해보자~
2	hope123	리액트 커리큘럼좀	2016.11.11	리액트... 어떻게 시작하나요??

사용자 테이블

사용자ID	사용자닉네임
assistant0603	조교행님
sujin02	수지니
jjoayo	조아요맨
hope123	코린이

댓글 테이블

댓글ID	댓글에대응하는글ID	사용자ID	댓글내용	댓글생성일
1	1	sujin02	잘부탁드려요~	2015.06.01
2	1	jjoayo	쫌아요~	2012.03.02
3	2	hope123	리액트 정말 막막하네요	2016.11.11
4	2	hope123	혹시 잘 아시는 분 있나요? ㅠㅠ	2016.11.11
5	2	sujin02	조교행님 유튜브는 어때요?	2016.11.12
6	2	assistant0603	제가 바로 조교행님입니다.	2016.11.12

세 개의 테이블로 나뉘었을 경우다. 글은 글만, 사용자는 사용자만, 댓글은 댓글만 담당한다. 중복도 없고, 널값도 없다. 훨씬 보기 깔끔하지 않은가?
 그런데, 댓글 테이블의 경우, 누가 댓글을 썼는지 알아야하기에 사용자 아이디가 필요하고, 어떤 글에 댓글을 썼는지 알아야하기에 글 ID가 필요하다. 즉, 다른 테이블의 정보에 접근할 일이 생기는 것이다. 이 때, ‘관계’가 필요한 것이다. 관계는 어떻게 정의할까? 두 개의 키를 알아야 한다.

PK: Primary key (주키)

FK: Foreign key (외래키)

원래 테이블에 존재하는, 중복없는 키가 Primary key 이다. 어떤 것이 될 수 있나? 사용자 ID가 될 수 있을 것이다. 사용자가 회원가입할 때 쓴 ID는 다른 사용자와 중복되지 않는다. 그렇다면 글과 댓글의 경우는 어떠한가? 우리가 일일이 아이디를 입력하는건 비효율적이기에, SQL CRUD 시간에 했던 것처럼 시퀀스, 즉 SERIAL 타입으로 설정하면, 따로 입력하지 않아도 알아서 ID가 부여되며, 중복될 일이 없다.

그럼 FK는 무엇인가? 이것은 ‘다른 테이블에 있는 PK’ 라고 생각하자. 댓글 테이블을 보면, 초록색 부분 두 컬럼이 있는데, 댓글 테이블은 FK 를 두 종류 가지고 있는 것이다. 하나는 글 테이블에 접근하기 위한 글 ID, 다른 하나는 유저 정보에 접근하기 위한 유저 ID 이다. FK 가 있으면 해당 테이블에 접근해서 원하는 정보를 가져올 수 있다.

즉, 어떤 테이블에 FK가 있다는 뜻은, 다른 테이블과의 ‘관계’가 성립한다는 것이다.

이제 본격적으로 설계를 해보자. 우리 현실의 수많은 문제를 여러개의 테이블로 바꾸고, 그 관계를 정의할 것이다. 물론, 이것은 매우 어렵다! 개인적으로, 개발 과정에서 가장 중요한 단계라고 생각한다. 코딩을 가르치는 수업이기 때문에 착각할 수 있는데, 코딩은 전체 개발과정의 20% 만 차지한다고 생각한다. 중요한건 설계다!

다음과 같은 단계로 진행된다.

1. UI 작성
2. ERD 설계
3. ERD 를 TABLE 로 변환

가장 먼저 해야 할 일은 UI 를 짜는 것이다. UI 는 User Interface 의 약자다. 여러분의 앱을 그려보는 것이다. 키보드에 손 올리기 전에 먼저 펜과 종이를 들어야한다. 내 앱이 어떻게 생겼는지 아는것에서부터 설계는 시작된다. 어떻게 생겼고, 무엇을 클릭하면 어떤 동작을 하고, 입력하는 부분은 어디에 있고, 사용자의 이벤트에 따라서 뭘 보여줄지를 상세하게 짜야한다.

두번째로, 해당 UI 를 바탕으로 DB 설계도인 ERD 를 만들어보는 것이다. ERD는 Entity Relational Diagram 의 약자다. 엔티티의 관계를 정의해놓은 다이어그램, 그림을 말한다.

세번째로, 이것을 테이블로 바꾸는 것이다. 지금까지 SQL CRUD를 배웠기 때문에 여러분들이 직접 할 수 있다.

5-1. UI 작성

먼저 첫번째, UI 작성부터 해보도록 한다. 다른 툴 생각하지 말고, 일단 펜과 종이만으로 작성하라. 시장에 나온 여러가지 툴들이 있는데, 생각 그려보기엔 펜과 종이가 최고다. 컴퓨터로 꼭 해야만 한다면, 파워포인트, 그림판도 좋다. 흔히 보는 툴이라 무시할 수 있는데, 핵심은 머릿속에 있는 생각을 대략적으로 작성하는데 있다.

눈에 보이는것만큼이나 중요한 건 사용자가 어떤 행동을 했을 때, 웹앱이 어떤 동작을 하느냐이다. 즉, 이벤트를 생각해야한다. 클릭, 스크롤, 키보드입력, 마우스 올리기(hover), 화면 크기 변경 등, 사용자의 모든 행동은 이벤트이다.

또한 기능도 고려해야한다. 이 웹앱의 목적은 무엇인가를 생각해봐야한다.

검색창이 있다면 그 기능은 무엇인가? 어떤 세부 기능으로 나눌 수 있는가?

쇼핑몰이라면 로그인, 주문하기, 주문조회, 제품검색, 배송조회 등의 기능이 있을 수 있다.

이런 식으로, 간단한 UI를 작성해보고, 토론해보고, 주변사람에게 물어보면 수정할 부분이 보일 것이다. 어느 정도 수정한 후에 좀 더 전문적인 툴을 사용해 UI를 제대로 작성하면 훨씬 편하다. 일이 한참 진행되고나서 무엇인가를 바꾸기엔 굉장히 힘든 법이다.

펜과 종이, 그림판, 파워포인트보다 괜찮은 툴들은 다음이 있다.

- | | |
|-----------|--|
| Balsamiq | 간단하게 아이디어를 그리기에 좋다. 써보면 스케치북에 그림그리는 느낌이다. |
| Photoshop | 전통적인 툴이다. 예전엔 이걸로 다 했다. 그러나, UI를 위한 것이 아니라 사진 편집용 프로그램이기때문에 UI 제작을 위한 기능이 부족하거나, 필요없는 기능이 지나치게 많아 무겁다. |

여기까지는 그래픽 툴이지만, 아래는 프로토타이핑 툴이다. 프로토타이핑이란, 코드한줄없이 작동하는 프로그램 샘플이다.

- | | |
|--------------|---|
| Adobe XD | 어도비에서 출시한, UI 프로토타이핑 툴이다. |
| Sketch | Mac 용 프로그램이다. 아직까진 업계 표준 UI 프로토타이핑 툴이다. |
| Figma | 현 시점에서 업계 표준이 될 가능성이 가장 높다. 개인 사용자에게겐 무료이며, 설치가 필요없는 웹앱이다. 웹 기반이기 때문에 다른 사람과 같이 작업하기도 쉽다. |

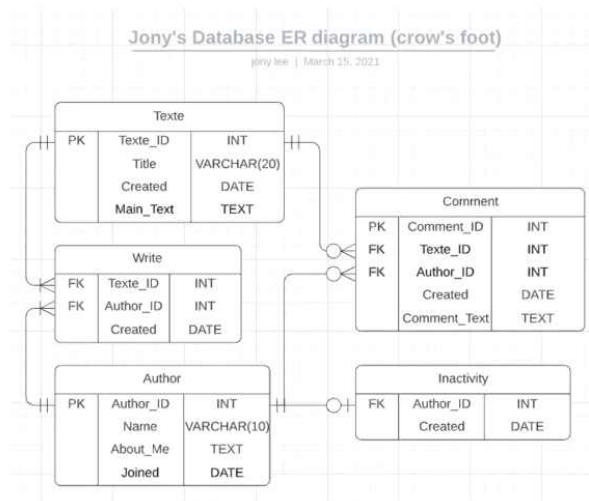
이런 툴들을 꼭 사용할 필요는 없지만, 웹디자인을 제대로 하고싶다면 배워보는것을 추천한다. 특히 Figma 는 배워보는 수준에선 무료버전으로 충분한데다, 크게 어렵지도 않고, UI 제작에 딱 필요한 기능들만 있으며, 웹앱이기 때문에 운영체제에 관계없이 사용할 수 있으므로 꼭 배워보길 바란다.

ERD 수업을 위해 대충 만들어본 UI 는 다음과 같다.



멋진 무엇인가를 기대했을지 모르겠지만, ERD 를 익히는 데 충분하다. 웹 디자인 수업이 아니라 데이터베이스 수업이다!

ERD(Entity-relationship Diagram, ER Diagram) 은 Entity와 Entity 사이의 “관계”를 그림으로 나타낸 것이다. 이 장 첫 페이지에서 봤던 다음 형태의 그림을 말한다.



Entity 하나의 주제, 그룹. 나중에 테이블이 될 대상이다.

엔티티, 즉 테이블을 구분하려면 연습이 필요하다. UI 가 복잡하면 복잡할수록 테이블을 나누는 것은 힘든 일이다. 핵심은 기능별로 구분하는 것이다. 실습용 UI를 보면,



UI를 봤을 때 총 몇 개의 기능이 있는가? 유저를 담당하는 유저 페이지가 따로 있고, 해당 페이지엔 각 유저에 대한 정보들이 있다. 이것은 하나의 테이블이 될 수 있을 것이다. 글은 어떠한가? 글의 제목, 글쓴이, 글의 내용, 글 생성 날짜가 있다. 댓글도 보인다. 댓글 내용, 댓글 작성자, 작성일을 확인할 수 있다.

즉, 이 화면은 세 개의 테이블로 나눌 수 있다.

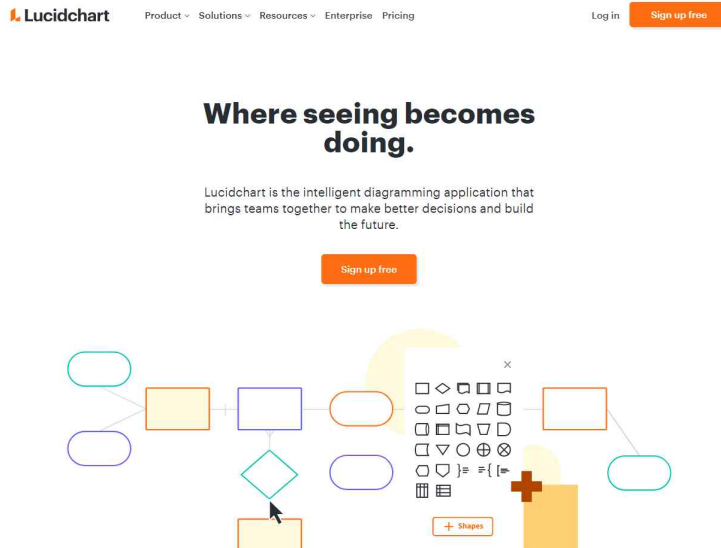
1. 글
2. 댓글
3. 유저

그러나, 테이블 이름만 가지고선 테이블이라고 할 수 없다. 컬럼이 없지 않은가? 글 테이블이라면 글 제목, 글 내용, 글 작성 시간 등이 들어가야 할 것이다. ERD에선, Entity 의 하위 항목을 Attribute, 다른 말로 Field 라고 한다. Attribute는 컬럼 이름이 될 것이다.

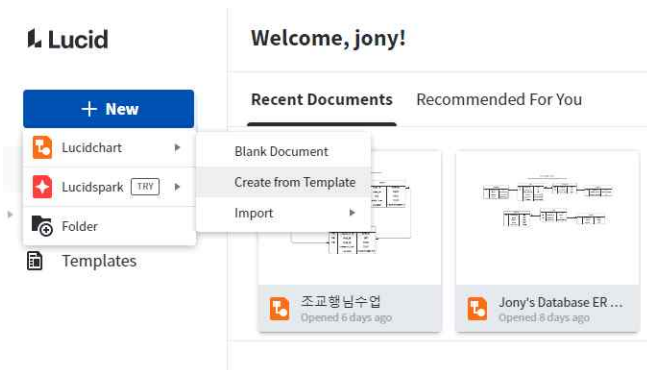
5-3. LucidChart – 테이블 만들기

ERD 실습을 펜과 종이로 진행할 수 있지만, 이 수업에선 LucidChart 라는 웹앱을 이용해볼 것이다.

구글 검색 lucidchart

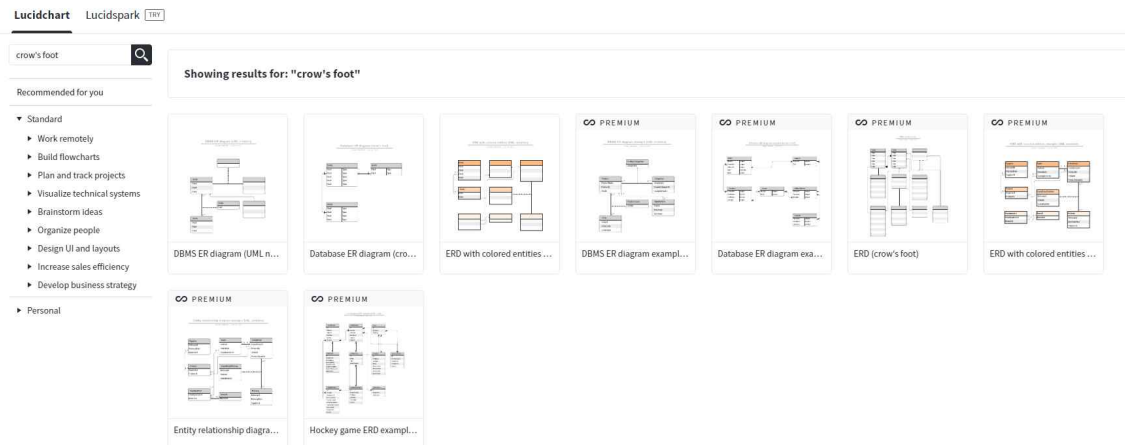


회원가입을 진행하도록 하자.

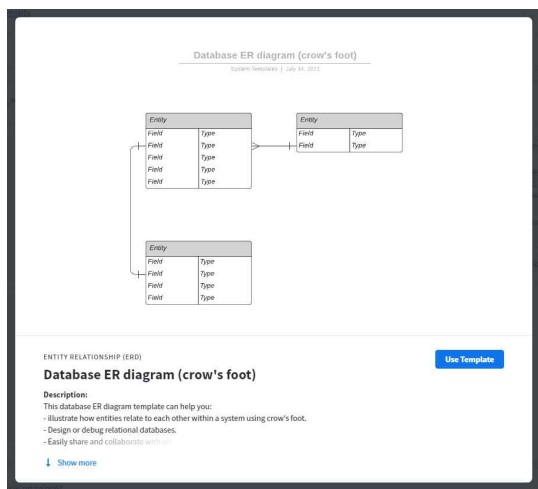


New 버튼 – LucidChart – Create from Template 클릭

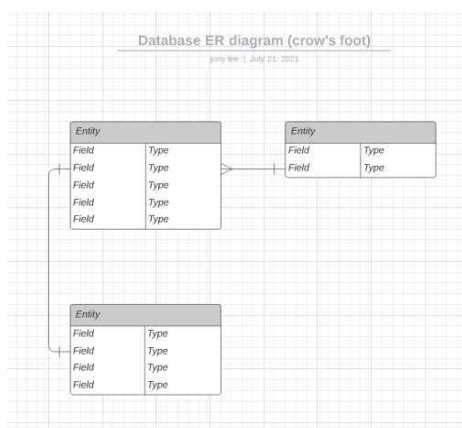
ERD는 크게 두 가지 형태가 있는데, UML 과 Crow's foot이다. 우린 Crow's foot 으로 진행할 것이다.



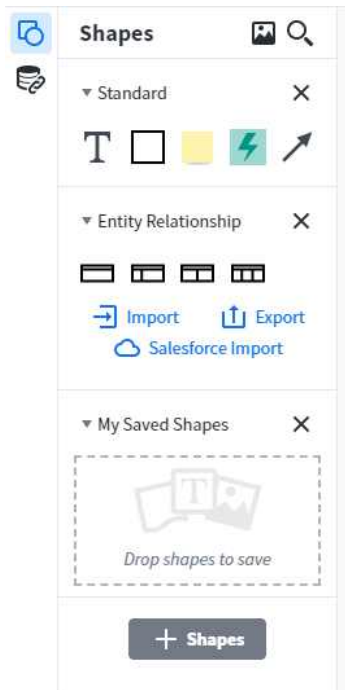
왼쪽 검색창에 crow's foot 을 입력했을 경우. 프리미엄은 돈을 내고 써야되고, 무료로 제공되는 템플릿 세 개중 UML 이 아닌 Crow's foot 은 딱 하나다. 클릭해주자.



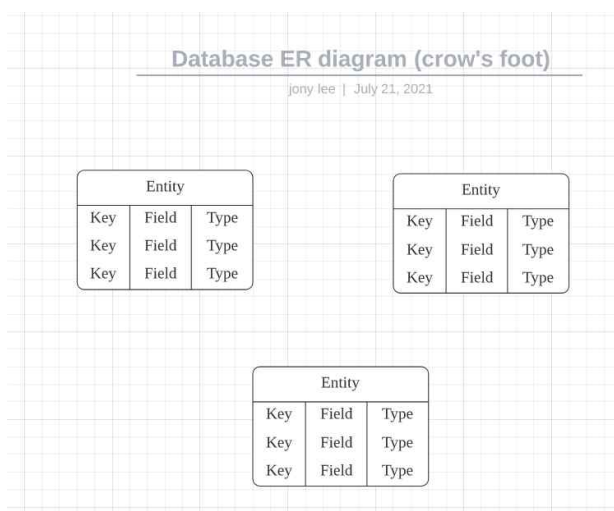
파란색 버튼 클릭



다음과 같은 다이어그램이 생성되었다. 테이블간 연결되어 있는 모습을 잘 보면, 까마귀 발처럼 생겨서 crow's foot 이다. 현재 필드는 총 두 부분으로 나뉘어져 있는데, 우린 이거 말고 세 개로 나누어진 Entity 를 사용할 것이다. 드래그해서 다 지워주자.



왼쪽에서 Entity Relationship 아래에 세 부분으로 나뉘어진 테이블이 있는데, 이것을 드래그해서 화면에 옮기면 Entity 가 만들어진다. 우린 세 개의 테이블을 만들 것이기 때문에, 세 번 옮겨주자.



각각의 Entity 는 테이블이 될 것이다. 여기에 들어가는 정보는 모두 영어로 지어야 한다.

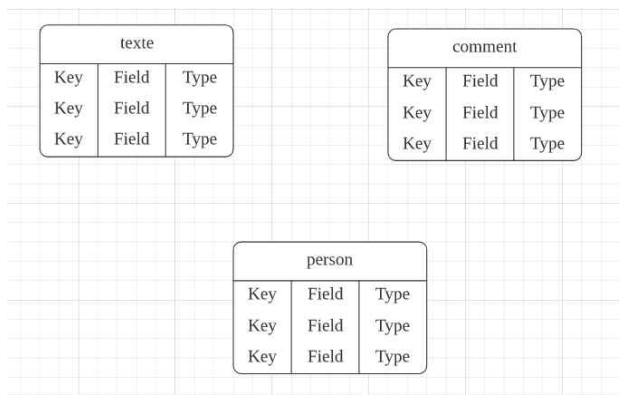
이름을 정해보면,

글 - text

댓글 - comment

유저 - user

여기서, text 와 user 는 수정해 줄 필요가 있는 이름이다. PostgreSQL 에서 TEXT 라는 자료형이 있고, USER 라는 키워드가 존재하기 때문이다. text는 프랑스로 texte 로 쓰인다. 맨 끝에 e를 붙여줘서 TEXT 타입과의 중복을 피하고, user 는 person 으로 바꿔주자.



Attribute 를 자세히 보면, 세 부분으로 나뉘어진다.

첫번째는 Key. PK 또는 FK가 들어갈 곳이다.

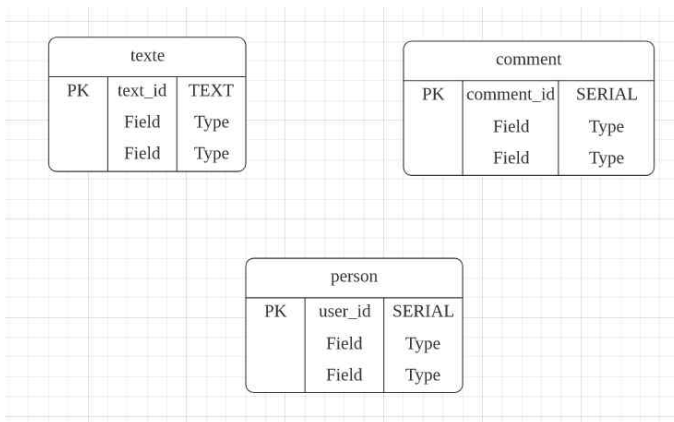
두번째는 Field. 컬럼 이름이 될 부분이다.

세번째는 Type. 해당 컬럼의 데이터타입을 써준다.

먼저, 각각의 테이블에 PK를 하나씩 달아주자.

* 모든 테이블이 PK를 가지고 있는것은 상황에 따라 효율적이지 않을 수 있다.

지금 이 수업에선 간단한 예제를 보여주는 것이기에, 모든 테이블에 PK를 달도록 하겠다.



각각의 테이블이름에 언더바(_) 쓰고 id 라고 이름지었는데, text_id 라고 써도 더이상 키워드가 아니기 때문에 e 를 떼버렸고, person 에서 id는 user_id 라고 써주었다. 그리고 나머지 키는 FK 가 들어가기 전까진 필요가 없기 때문에 삭제했다.

text_id 는 왜 SERIAL 이 아니라 TEXT 타입인가? 우리가 회원가입할 때 입력하는 아이디는 중복되지 않는다. 만약 중복된 아이디로 가입하려고 하면 가입 자체가 안된다. 즉, 중복을 걱정할 이유가 없기 때문에 SERIAL 이 아니다.

이제 각 테이블에 어떤 컬럼이 들어갈지 생각해보자.

조교행님

ID: assistant0603
pw: 123456
한줄소개: 내가 바로 조교행님이다
가입일자: 2012.03.02
휴면여부: 휴면아님

수지니

ID: sujin02
pw: 121212
한줄소개: 안녕하세요 ㅎㅎ
가입일자: 2012.03.02
휴면여부: 휴면

조아요맨

ID: joayo
pw: mypassword
한줄소개: 조아요~
가입일자: 2012.03.02
휴면여부: 휴면

조교행님의 첫 글 ㅎㅎ 조교행님

어서와 웹개발은 처음이지?
이 수업은 React와 GraphQL을 이용해서
Nobase 라도 시작할 수 있는 수업이야
우리 같이 할 해보자 ㅎㅎ
2012.03.02

잘 부탁드립니다~ 수지니 2015.06.01
좋아요~~~ 조아요맨 2015.06.01

리액트 커리큘럼종 코린이

리액트... 어떻게 시작하나요??
정말 막막하네요
혹시 제대로 된 강좌가 있을까요?
2016.11.11

리액트... 정말 막막하네요 코린이 2016.11.11
혹시 잘 아시는 분 있으신가요? 코린이 2016.11.11
조교행님 유용한 정보는 아세요? 수지니 2016.11.12
제가 새로 조교행님입니다. 조교행님 2016.11.12

글 - 글 제목, 글 작성자, 글 내용, 글 작성 시간

댓글 - 댓글 내용, 댓글 작성자, 댓글 작성 시간

유저 - 아이디, 닉네임, 패스워드, 한줄소개, 가입일자, 휴면여부

빨간색으로 되어 있는 부분은 글 테이블과 댓글 테이블에 있어야하지만, 실제 어디에 있는가? 유저 테이블에서 가져올 부분이다. 즉, 관계를 이어주면 해결될 문제이기에 글 테이블과 댓글 테이블엔 일단은 입력하지 않도록 하겠다.
유저 테이블의 아이디는 PK로 이미 입력되어 있기 때문에 제외하겠다.

각 테이블에 들어갈 컬럼을 다시 정리해보면,

글 - title, main_text, created

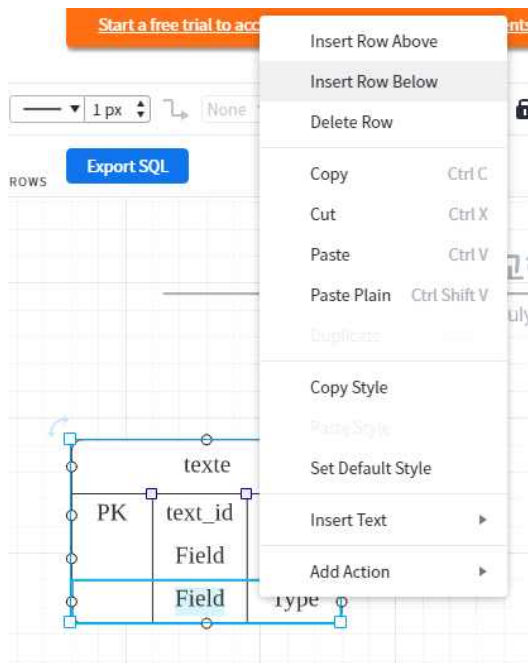
댓글 - comment_text, created

유저 - name, password, about_me, joined, is_activate

* 휴면여부를 is_active 라고 썼다. 프로그래밍에서 true, false 타입을 가지는 변수는 보통 is + 동사 로 이름짓는다.

* 글에도 created 가 있고, 댓글에도 created 가 있어서 중복되지 않나 걱정스러울수도 있는데, 다른 테이블에 속하기 때문에 이름이 같아도 상관없다.

* 낙타체(camelCase) 대신 언더바(_)를 사용했다. PostgreSQL 에션 대문자로 써도 어차피 소문자로 바뀔 것이다.



데이터를 추가하고 싶을 땐 행을 선택한 상태에서 오른쪽 클릭해 Insert Row Below 를 클릭한다.

texte		
PK	text_id	TEXT
	title	TEXT
	main_text	TEXT
	created	TIMESTAMPZ

comment		
PK	comment_id	SERIAL
	comment_text	TEXT
	created	TIMESTAMPZ

person		
PK	user_id	SERIAL
	name	TEXT
	password	TEXT
	about_me	TEXT
	joined	DATE
	is_activate	BOOL

새로 추가된 행이 안 예쁠 수 있다. 이 경우, 전체 테이블을 마우스 드래그해서 선택해준 다음, 상단에 있는 탭에서 글자크기와 글씨체, 볼드체, 정렬을 맞춰주면 된다.

texte		
PK	text_id	TEXT
	title	TEXT
	main_text	TEXT
	created	TIMESTAMPZ

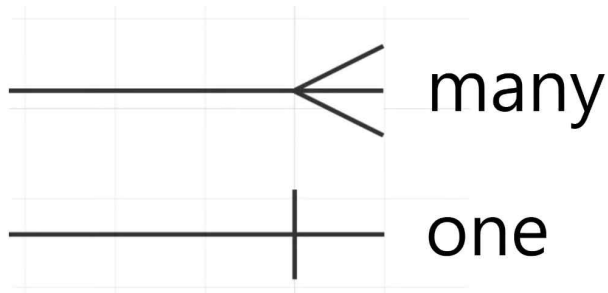
comment		
PK	comment_id	SERIAL
	comment_text	TEXT
	created	TIMESTAMPZ

person		
PK	user_id	SERIAL
	name	TEXT
	password	TEXT
	about_me	TEXT
	joined	DATE
	is_activate	BOOL

예쁘게 수정된 모습.

테이블간의 관계를 설정해보자. 핵심은, 두 관계 사이의 최대(maximun)를 아는 것이다.

어렵게 가르치면 한없이 어렵지만, 당장 알아야할건 이 두개밖에 없다.



약관화면

유저 관리자 페이지

조교행님의 첫 글 ㅎㅎ

조교행님

어서와 땀겨발은 처음이지?
이 수업은 React와 GraphQL을 이용해서
Nobase 라도 시작할 수 있는 수업이야
우리 같이 잘 해보자 ㅎㅎ

2012.03.02

방 부탁드려요~

수지나

2015.06.01

좋아요~~

조아요맨

2015.06.01

리액트 커리큘럼쭈

코린이

리액트.... 어떻게 시작하나요?
정말 막막하네요
혹시 제대로 된 강좌가 있을까요?

2016.11.11

리액트... 정말 막막하네요

코린이

2016.11.11

혹시 잘 아시는 분 있으신가요??

코린이

2016.11.11

조교행님 유용해보는 아재요?

수지나

2016.11.12

제가 바로 조교행님입니다.

조교행님

2016.11.12

조교행님

ID: assistant0603

pw: 123456

한줄소개: 내가 바로 조교행님이다

가입일자: 2012.03.02

휴면여부: 휴면아님

수지나

ID: sujin02

pw: 121212

한줄소개: 인생아세요 ㅎㅎ

가입일자: 2012.03.02

휴면여부: 휴면

조아요맨

ID: jioayo

pw: mypasswd

한줄소개: 조아요~

가입일자: 2012.03.02

휴면여부: 휴면

texte		
PK	text_id	TEXT
	title	TEXT
	main_text	TEXT
	created	TIMESTAMPZ

comment		
PK	comment_id comment_text created	SERIAL TEXT TIMESTAMPZ

두 가지 질문이 필요하다.

1. 하나의 글은 최대 몇 개의 댓글을 가지는가?

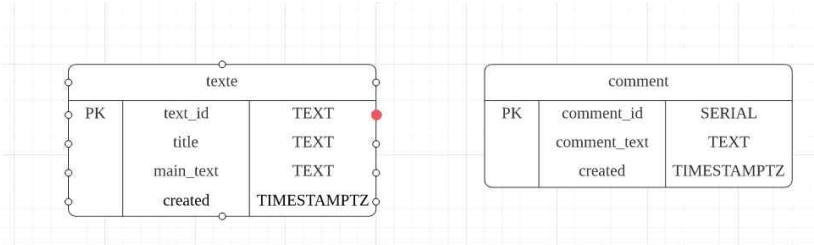
여러개다. 하나의 글에 여러 개 댓글이 달릴 수 있다. many

2. 하나의 댓글은 최대 몇 개의 글을 가지는가?

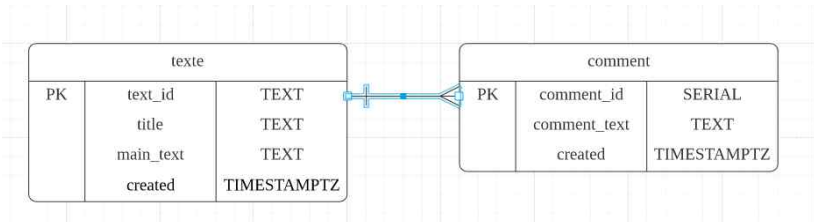
하나다. 댓글은 하나의 정해진 글에 달린다. one

어떻게 그리면 될까?

마우스 커서를 올리면 선을 그릴 수 있도록 활성화된다.



texte 와 comment 를 이어준 경우.



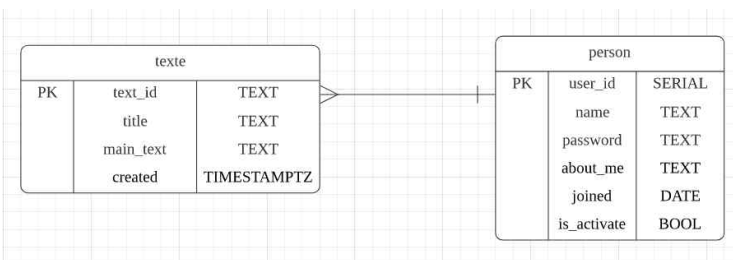
comment 는 최대 하나의 texte 를 가지고, texte는 여러개의 댓글을 가질 수 있기 때문에, 알맞게 그려졌다.

만약, 이 형태를 수정하고 싶다면?



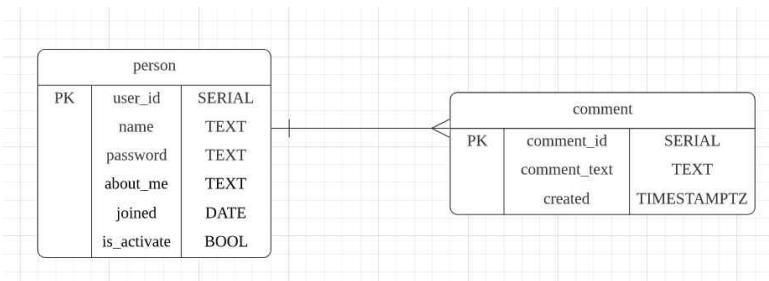
위의 도구상자에서 이 부분을 바꿔주면 된다.

다른 테이블들간의 관계를 살펴보면,



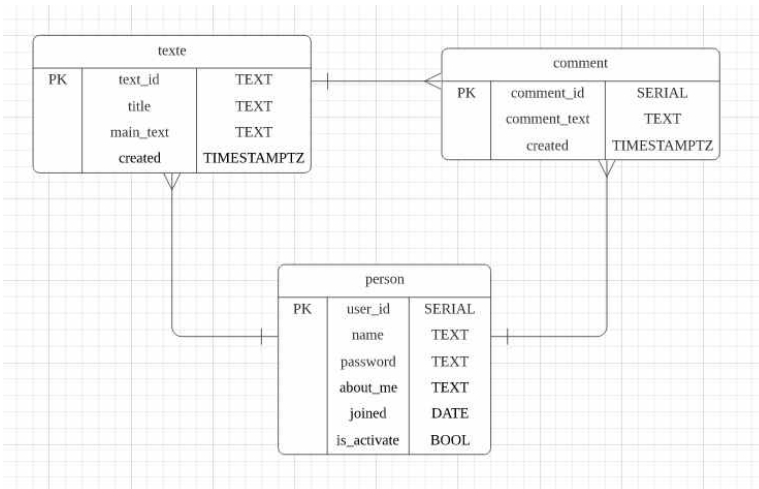
한 명의 유저는 최대 여러개의 글을 쓸 수 있고,
하나의 글은 딱 한 사람한테만 쓰여진다.
따라서, 위와 같이 그려야한다.

유저와 댓글의 관계는 어떠한가?



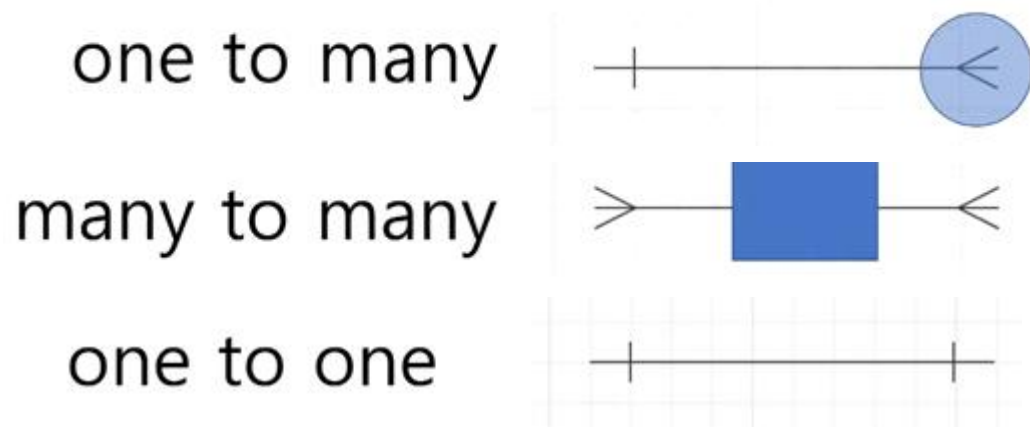
한 명의 유저는 여러개 댓글을 쓸 수 있고,
하나의 댓글은 한 사람에게서 쓰여진다.
즉, 위와 같이 그려야한다.

종합해보면 다음과 같다.



그래서 이걸 왜 그렸는가? 누군가에게 이런 걸 만들었다고 자랑하려고 그렸는가?
우리의 ERD는 아직 완성되지 않았다. FK 가 있어야 두 테이블간의 관계가 성립하기
때문이다.

중요한 건, 어디에 FK를 둘 것인가이다. 다음 세 가지 상황으로 나뉜다.



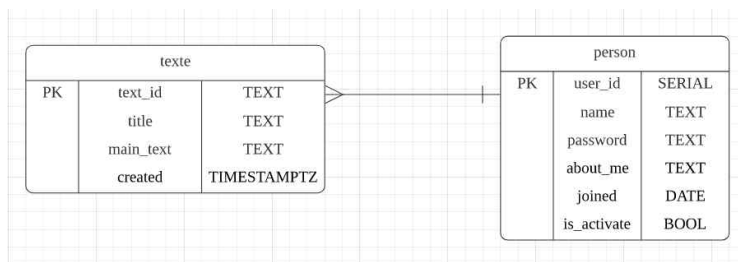
5-4-1. one to many

우리가 만든 ERD 는 모두 one to many 관계로 이루어져있다. 사실, one to many 가 가장 쉽다! 결론부터 말하자면, many 쪽에 FK를 두면 된다.

one to many



유저와 게시글간의 관계를 생각해보자. 다음과 같다.



만약, one 쪽인 person 에 FK 를 둔다면 어떤 일이 생기는가?

person						
user_id(PK)	text_id(FK)	password	name	about_me	joined	is_activate
assistant0603	1	123456	조교행님	내가 바로 조교행님이다	2012.03.02	TRUE
sujin02		121212	수지니	안녕하세요 ㅎㅎ	2012.03.02	FALSE
corini	2	corini1453	코린이	코딩 처음합니다	2016.11.11	TRUE
assistant0603	3	123456	조교행님	내가 바로 조교행님이다	2012.03.02	TRUE
jjoayo		mypassword	조아요맨	조아요~	2012.03.02	FALSE

다음과 같이 중복이 발생하는것을 확인할 수 있다. 왜? 유저가 글을 쓰면 쓸수록 행은 늘어날 것이고, 글 하나 더 썼을 뿐인데 아이디, 패스워드, 닉네임 등의 쓸데없는 정보가 늘어나는것이다.

만약 many 쪽인 person에 FK를 둔다면 어떠한가?

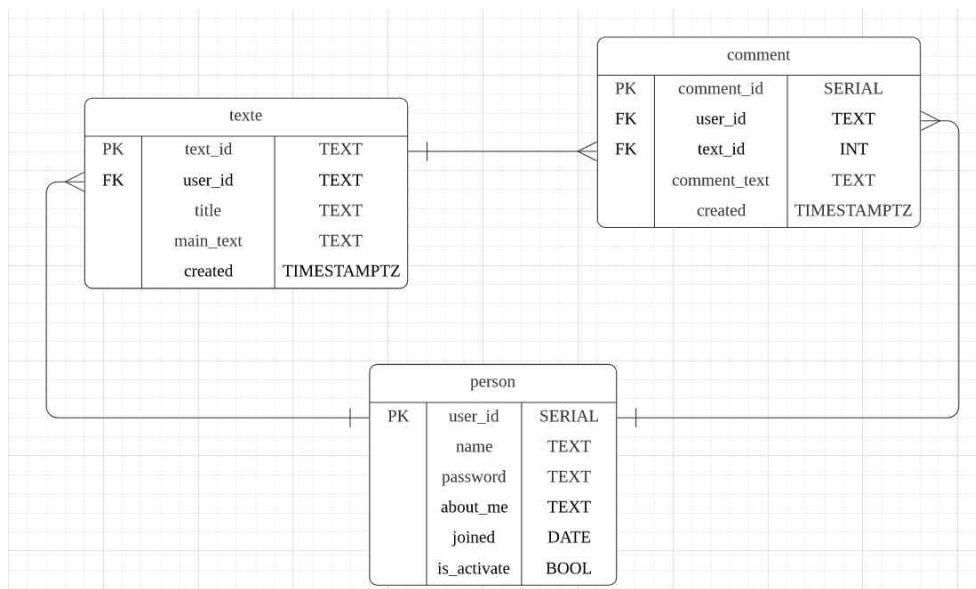
person					
user_id(PK)	password	name	about_me	joined	is_activate
assistant0603	123456	조교형님	내가 바로 조교형님이다	2012.03.02	TRUE
sujin02	121212	수지니	안녕하세요 ㅎㅎ	2012.03.02	FALSE
corini	corini1453	코린이	코딩 처음합니다	2016.11.11	TRUE
jjoyo	mypassword	조아요맨	조아요~	2012.03.02	FALSE

texte				
text_id(PK)	user_id(FK)	title	main_text	created
1	assistant0603	조교형님의 첫 글 ㅎㅎ	어서와 웰개발은 처음이지? 이 수업은 ~~~	2012.03.02 오후 7시 37분
2	corini	리엑트 커리큘럼좀	리엑트... 어떻게 시작하나요?? ~~~	2016.11.11 오전 4시 08분

게시글에 해당 아이디만 넣어주면 되기 때문에 중복이 발생하지 않는다.

따라서, one to many 관계일 경우엔 many 에 FK 를 두어야한다.

이것을 우리 차트에서 정리해보면 다음과 같다.

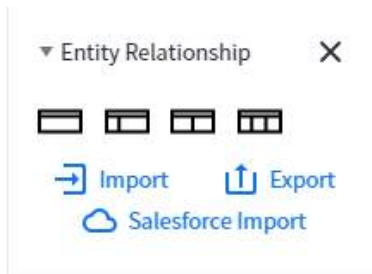


FK를 만든 후, PK와 FK를 정확히 맞춰서 선으로 이어주면 된다.

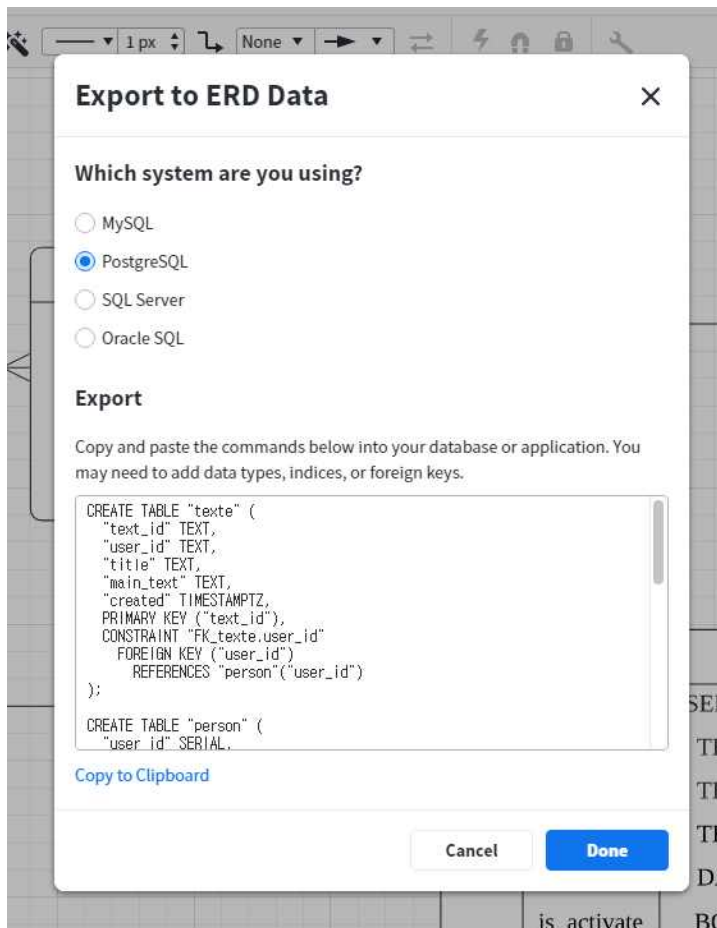
* comment의 text_id는 왜 SERIAL이 아니라 INT일까? comment 테이블에선 자동증가할 일이 없기 때문이다.

일단 우리 ERD는 이것으로 완성되었다. 우리가 가진 관계는 one to many 밖에 없기 때문이다. ERD를 이용해 PostgreSQL 샘플 데이터를 만들어보는것은 여러분에게 맡기겠다.

LucidChart엔 재미있는 기능이 있다. 이렇게 만든 ERD 를 SQL 문법으로 바꿔주는것이다.



왼쪽 도구모음에서 Export 를 클릭해보자.



PostgreSQL 을 선택할 경우, 다음과 같이 ERD에 맞는 SQL을 제공해준다. 그러나, DEFAULT 값 설정, NOT NULL 지정은 직접 입력해야되며, 배우지도 않은 외래키 연결도 포함되어있다. 개인적으로 외래키 연동은 지금 단계에선 불필요하다고 생각한다. 여러분이 배우는 과정이기 때문에 직접 SQL을 써보는것을 권장하므로, 이걸 그대로 복사 붙여넣기하는건 추천하진 않는다. 지금까지 배운 SQL 문법을 활용해서 우리의 ERD를 실제 관계형 데이터베이스로 바꿔보길 바란다.

5-4-2. one to one

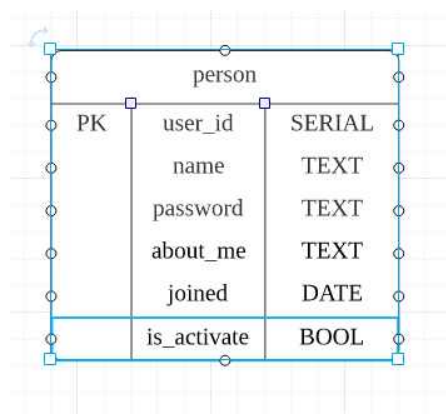
아직 수업은 끝나지 않았다. 여러분이 어떤 형태의 관계를 마주하게 될지 모르기때문에, 이론적으로만 알아두자.

one to one

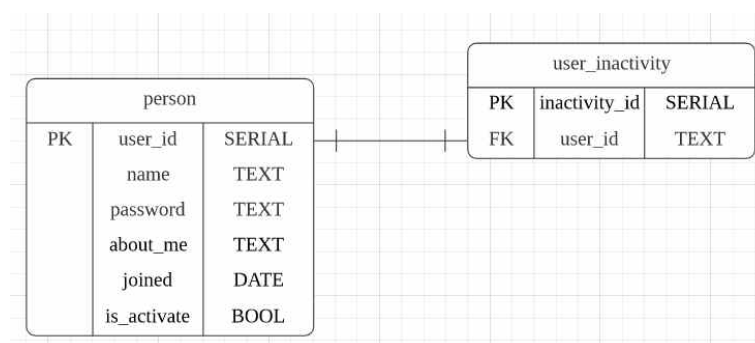


세 가지 관계 중, 가장 어렵다. 어디에 FK를 뒤편할지 “판단” 해야한다. 판단기준은 어느 테이블이 데이터를 추가하기 더 자유로운가이다.

어떤 경우가 있을 수 있을까?



휴면여부를 판단하는 is_activate 를 특정한 이유때문에 하나의 테이블로 분리할 계획이라고 하자. 그럼 다음과 같은 형태가 된다.



둘은 one to one 관계다. 한 명의 유저는 최대 한 개의 휴면여부를 가진다. 하나의 휴면여부는 최대 한 명의 유저를 가진다.

이 경우, 데이터를 자유롭게 추가할 수 있는 테이블은 당연히 person 이다. person 이 존재해야 user_inactivity 도 존재 이유가 있다. 자유로운 person에 FK를 하나 더 두는건 부담스러운 일이다. 왜 그럴까?

person

user_id(PK)	inactivity_id(FK)	password	name	about_me	joined
assistant0603		123456	조교행님	내가 바로 조교행님이다	2012.03.02
sujin02	1	121212	수지니	안녕하세요 ㅎㅎ	2012.03.02
corini		corini1453	코린이	코딩 처음합니다	2016.11.11
jjoayo	2	mypassword	조아요맨	조아요~	2012.03.02

user_inactivity

inactivity_id(PK)
1
2

중복은 발생하지 않는데, NULL 값이 발생하기 때문이다. 중복보다 심각한 문제는 아니지만, 아래와 같이 설계한다면 훨씬 깔끔하다.

person

user_id(PK)	password	name	about_me	joined
assistant0603	123456	조교행님	내가 바로 조교행님이다	2012.03.02
sujin02	121212	수지니	안녕하세요 ㅎㅎ	2012.03.02
corini	corini1453	코린이	코딩 처음합니다	2016.11.11
jjoayo	mypassword	조아요맨	조아요~	2012.03.02

user_inactivity

inactivity_id(PK)	user_id(FK)
1	sujin
2	jjoayo

자유롭지 않은 user_inactivity 에 FK를 맡길 경우, 다음과 같이 NULL 값이 발생하지 않는다.

즉, one to one 관계에서 판단을 하려면, 어떤 테이블이 더 자유롭게 데이터를 추가할 수 있는가를 따진 다음에, 덜 자유로운 테이블에 FK를 맡겨야한다. 이게 어렵다면, 실제 데이터를 넣어보고 더 효율적인 결과대로 FK를 추가한다.

5-4-3. many to many

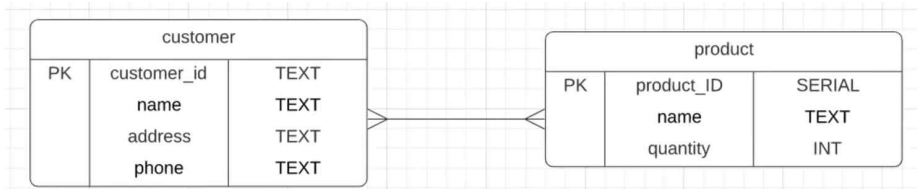
이 경우엔 가만히 놔두면 안된다. 중간에 ‘브릿지 테이블’이라고 부르는 테이블을 하나 뒀야한다.

* 브릿지(bridge) 는 ‘다리’ 라는 뜻이다.

many to many



우리 ERD에는 없는 관계이기 때문에, 다음과 같은 예제를 만들어보았다.



주문배송 시스템이다. 한명의 고객은 여러개의 상품을 살 수 있고, 하나의 상품은 여러명의 고객에게 주문될 수 있으니 many to many 관계이다.

무슨 문제가 발생하는가?

customer

customer_id(PK)	product_id(FK)	name	address	phone
jony1021	1	일건	부산시 사하구	010-5555-4444
youyou24		유유유	서울시 동대문구	010-7878-3333
jony1021	1	일건	서울시 동대문구	010-5555-4444
david7878		윌슨	울산광역시	010-9999-9999
sujeong2		소소한일상	춘천시	010-1234-5678

product

product_id(PK)	name	quantity
1	다이슨청소기	28
2	메뚜기쌀	144

customer 쪽에 FK를 둘 경우이다. 일건이라는 닉네임을 가진 고객이 다이슨 청소기를 한 대 샀고, 또 한 대 더 주문했다. 주문이 늘어날수록 중복이 발생한다. 다른 고객들은 아직 상품을 사지도 않았기 때문에 널값으로 남아있다.

또한, 언제 주문했는지 주문 시간도 알 수 없다. ordered_time 이라는 컬럼을 새로 만든다치고 customer 테이블에 넣으면 되지 않는가? 그러나 상식적으로, 사용자 데이터를 담는 테이블에 주문 시간이 들어간다는 건 생똥맞은 일이다.

다음 경우는 어떤가?

customer

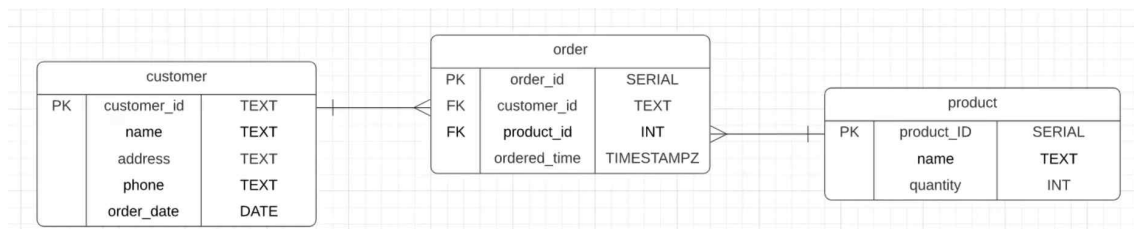
customer_id(PK)	name	address	phone
jony1021	일건	부산시 사하구	010-5555-4444
youyou24	유유유	서울시 동대문구	010-7878-3333
david7878	윌슨	울산광역시	010-9999-9999
sujeong2	소소한일상	춘천시	010-1234-5678

product

product_id(PK)	customer_id(FK)	name	quantity
1	jony1021	다이슨청소기	28
2	jony1021	메뚜기쌀	144
3	jony1021	메뚜기쌀	143
4	jony1021	메뚜기쌀	142
5	jony1021	메뚜기쌀	141

product 테이블에 FK를 둔 경우이다. 널값은 발생하지 않았지만, 주문이 늘어날수록 중복도 늘어난다. 주문 시간을 알기 위해 product 테이블에 ordered_time 컬럼을 추가할수는 없는데, product 테이블은 현재 보유하고 있는 제품만 관리해야하기 때문이다.

그렇다면 many to many 관계에선 어떻게 해야하는가? 브릿지 테이블을 뒤탈다.



order 라는 브릿지 테이블을 만들고, 둘을 이어주어 many to many 관계를 끊어버려야 한다. 브릿지 테이블에 두 개의 FK를 두며, 관습적으로 ‘시간’ 컬럼을 추가한다.

이 경우, 모든 문제가 해결될 수 있다. 중복도 발생하지 않고, 널값도 없을 것이며, 주문 시간도 알 수 있을 것이다.

6. 마치며

정말 어려운 수업이었다. 단순히 DB 사용법 뿐만 아니라, 전공생도 어려워하는 ERD 설계까지 포함된 수업이라서 많이 힘들었을 것이다. 정말 수고 많았다!

이제 DB를 만들었으니, React 에서 PostgreSQL 에 접근해 데이터 CRUD 를 할 수 있어야한다. 클라이언트에서 접근할 수 없는 DB는 아무 쓸모가 없다!

이를 위해 먼저 GraphQL 을 배워볼 것이다. 어떻게 데이터를 가져오고(query), 수정할 수 있는지(mutation) 알아보도록 하자. 그리고, PostgreSQL 과 GraphQL 을 결합해주는 라이브러리인 PostGraphile 을 사용해 볼 것이다.

하나 확실한 건, 전체 과정 중에서 ERD 가 가장 어려웠다는 것이다. 《ver.2021 노베이스 모던 웹개발》코스의 가장 어려운 부분은 끝났다.