

2

Chapter 2. 실전 SQL - 어떤 Query를 작성해야 할까? (1)

 학습 목표

 잠깐 ! 스터디 인증샷은 찍으셨나요? 

 2주차 주제

 2주차 본문

 SQL 마스터 도전

 여러 요구 사항에 대응하기

 페이지네이션 배워보기

 2주차를 마치며

 쿼리 실행 흐름

 핵심 키워드

 학습 후기

 스터디 진행 방법

 실습 체크리스트

 실습 인증

 위클리 스크럼 과제

 미션

 미션 기록 (여기에 해도 되고 위의 미션에서 각 페이지 밑에 간단하게 블록 만들어서 하셔도 됩니다!)



 트러블 슈팅

 참고 자료

학습 목표

1. 1주차 때 예시를 기반으로 여러가지 요구 사항에 대한 SQL 쿼리를 고민한다.
2. paging을 고려하여 쿼리를 작성한다.

잠깐 ! 스터디 인증샷은 찍으셨나요?

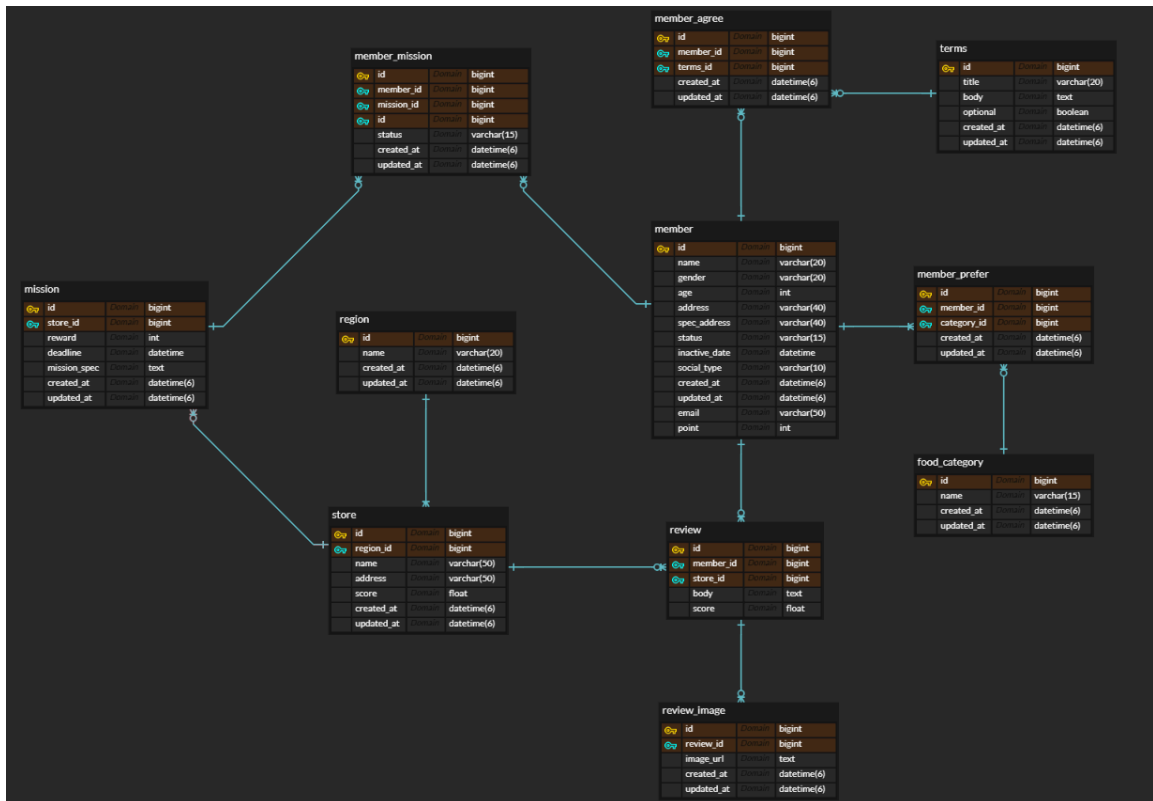
* 스터디리더께서 대표로 매 주차마다 한 장 남겨주시면 좋겠습니다!  
(사진을 저장해서 이미지 임베드를 하셔도 좋고, 복사+붙여넣기해서 넣어주셔도 좋습니다!)

2주차 주제

다들 데이터베이스 설계(1주차 미션)는 잘 하셨나요?

▼ 저 같은 경우는 아래의 형태로 설계를 했습니다. (참고해주세요!)

! 1주차 미션에 포함되지 않는 부분은 설계를 하지 않았습니다!



1주차 미션 ERD

글씨가 잘 보이지 않는다면 링크를 참고하시면 됩니다!

umc9th2week

Draw ERD with your team members. All states are shared in real time. And it's FREE. Database modeling tool.

ERD <https://www.ercloud.com/d/k7phBagbMpAeiTZBj>


이번 주차는 몇 가지 요구 사항에 대해 어떻게 쿼리를 만들어야 하는지 같이 고민해보는 주차입니다.

join, subquery는 알고 있다는 전제로 시작합니다!

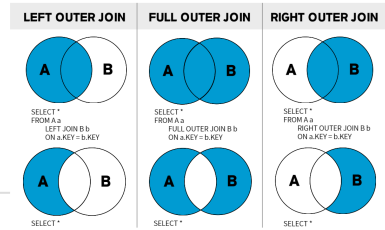
▼ 참고 자료

SQL 기본 문법: JOIN(INNER, OUTER, CROSS, SELF JOIN)

조인은 두 개의 테이블을 서로 묶어서 하나의 결과를 만들어 내는 것을 말한다. INNER JOIN(내부 조인)은 두 테이블을 조인할 때, 두 테이블에 모두 지정된 열의 데이터가 있어야 한다. OUTER JOIN(외부 조인)은 두 테

 <https://hongong.hanbit.co.kr/sql-기본-문법-joininner-outer-cross-self-join/>

OUTER JOIN



[SQL] 테이블 JOIN의 개념과 예제

테이블 JOIN의 개념과 예제


 <https://velog.io/@wijoonwu/JOIN>

[SQL] 테이블 JOIN

INNER JOIN, OUTER JOIN

[MYSQL] 📖 테이블 조인(JOIN) - 그림으로 알기 쉽게 정리

SQL JOIN JOIN은 데이터베이스 내의 여러 테이블에서 가져온 레코드를 조합하여 하나의 테이블이나 결과 집합으로 표현해 주는, Relation Database 에서 가장 많이 쓰이는 녀석이다. (INNER)

 <https://inpa.tistory.com/entry/MYSQL-📖-JOIN-조인-그림으로-알기쉽게-정리>


SQL / MySQL 서브쿼리(SubQuery)

서브쿼리(Subquery) 서브쿼리(Subquery)란 하나의 SQL 문 안에 포함되어 있는 또 다른 SQL문을 말한다. 서브쿼리는 메인쿼리가 서브쿼리를 포함하는 종속적인 관계이다. #메인쿼리 SELECT

 <https://snowple.tistory.com/360>

[MYSQL] 📖 서브쿼리 개념 & 문법 100 정리

서브쿼리(Subquery) 서브쿼리(subquery)란 다른 쿼리 내부에 포함되어 있는 SELETE 문을 의미한다. 서브쿼리를 포함하고 있는 쿼리를 외부쿼리(outer query)라고 부르며, 서브쿼리는 내부

 <https://inpa.tistory.com/entry/MYSQL-📖-서브쿼리-정리>

😊 **mysql 기준으로 진행합니다.**

📌 2주차 본문

기존 1주차 워크북 본문서 요구하던 요구 사항,
그리고 전 미션(1주차 미션)에서 요구했던 요구 사항 일부에 대해
어떻게 쿼리를 만드는 것이 좋을지 고민을 해봅시다.

query 역시 Database 설계처럼 정해진 답이 없고,
join을 사용할 지 혹은 subquery를 사용할 지 선택하시면 됩니다!

그리고 너무 무리해서 **한번에 거대한 query를 보내기** 보다는 **힘들 경우, 2개로 쪼개서 보내는 것도** 좋습니다.

특히 복잡한 쿼리를 작성할 때는 같은 쿼리라도 어떤 쿼리가 더 효율적인지 여러 가지 테스트 방법을 통해서 확인하는 것도 좋습니다.

워낙 ORM과 같은 도구들이 우리를 편하게 해주긴 하지만, 그럼에도 불구하고 직접 쿼리 짜는 연습을 해두면 그 노력이 언젠가 빛을 발할 날이 있을 거예요.

💡 SQL 마스터 도전



본격적으로 어떤 쿼리가 좋은 쿼리인지 고민해보기 전에, 예제로 워밍업을 해봅시다!

💡 Chapter 2. 실전 SQL- 워밍업

💖 여러 요구 사항에 대응하기



워밍업 했던 내용들을 토대로 여러 요구 사항에 대응하는 쿼리를 짜봅시다.

💖 Chapter 2. 실전 SQL- 어떤 Query를 작성해야 할까?

💖 페이지네이션 배워보기



Offset 기반 페이징과 Cursor 기반 페이징에 대해 배워봅시다.

💖 Chapter 2. 실전 SQL- SQL과 페이지네이션

✍️ 2주차를 마치며

수고하셨습니다 :) 이 내용은 가볍게 읽고 넘어가 볼게요. 다음주도 화이팅!

🔧 쿼리 실행 흐름



쿼리는 서버에 어떻게 반영되는 걸까요?

🔧 Chapter 2 BOUNS. 내가 짠 쿼리는 어떻게 반영되는걸까?



핵심 키워드



주요 내용들에 대해 조사해보고, 자신만의 생각을 통해 정리해보세요!

레퍼런스를 참고하여 정의, 속성, 장단점 등을 적어주셔도 됩니다.

조사는 공식 홈페이지 **Best**, 블로그(최신 날짜) **Not Bad**

▼ 데이터베이스 정규화

중복된 데이터를 허용하지 않음으로써 이상현상이 있는 릴레이션을 분해하여 그걸 없애는 과정, 무결성 유지

[장점]

- 데이터베이스 변경 시 이상현상을 제거할 수 있다
- 정규화된 데이터베이스 구조에서는 새로운 데이터 형의 추가로 인한 확장 시 구조를 변경하지 않거나, 일부만 변경 가능하다

[단점]

- 릴레이션을 분해로 인해 조인 연산이 많아져 질의에 대한 응답 시간이 느려질 수가 있다

→ 정규화에는 제 1 정규화, 제 2 정규화, 제 3 정규화, BCNF(Boyce-codd Normal Form), 제 4 정규화, 제 5 정규화

▼ 인덱스(Index)

- 추가적인 쓰기작업과 저장공간을 활용하여 데이터베이스 테이블의 검색 속도를 향상시키기 위한
데이터와 데이터의 위치를 포함한 자료구조
- DBMS는 인덱스를 항상 최신 정렬 상태로 유지해야지만 빠른 검색이 가능하기 때문에 인덱스가 적용된 컬럼에 Insert, Update, Delete 가 수행된다면 오버헤드가 발생한다.
- **인덱스를 사용하면 좋은 경우의 테이블**
 - 규모가 작지 않은 테이블
 - INSERT, UPDATE, DELETE가 자주 발생하지 않는 컬럼
 - 데이터의 중복도가 낮은 컬럼
- **인덱스의 자료구조**
 - 해시 테이블 :
Key, Value 로 이루어져 있으며 빠른 데이터 검색에 사용,
(=) 연산에 특화 되어있기 때문에 부등호 연산이 사용되는 데이터 베이스 검색에서는 해시 테이블이 적합하지 않음
 - B+Tree :
인덱스를 위해 자식 노드가 2개 이상인 B-Tree 를 개선시킨 자료구조,
각 리프노드들을 LinkedList로 연결하여 순차검색을 용이하게 최적화 하였기
때문에 해시 테이블보다 인덱싱에 더 적합한 자료구조!

▼ ORM VS Raw SQL

ORM 이란?

- Object Relational Mapping
- 객체 지향적 사고방식과 관계형 데이터베이스 사이를 연결할 계층의 역할
- 백인드 개발자는 SQL 문법을 알고 있어야하고, 개발 코드와 DB 쿼리코드가 서로 종속 되는 문제 해결을 위해 ORM 이 등장함
- 따라서 SQL 쿼리 문법에서 자유로워졌고, 개발 코드와 DB를 아예 분리 하여 객체 지향에 가깝게 됨

[ORM 사용 시]

- 프로그래밍의 생산성 향상

선언문, 콜백 함수, 종료 등과 같은 쓸데없는 코드가 사라져 코드의 가독성 향상 및 생산성 향상

- 눈에 잘 띄는 쿼리 (ORM 함수 사용)

코드가 간결해지기 때문에 보기가 쉬워진다.

- DB 쿼리 의존성 감소

대부분 ORM 솔루션이 특정 DB에 종속적이지 않다.

개발자는 DB에 상관없이 Object에만 집중이 가능하다.

[SQL 사용 시]

- 수정이 더 용이함

SQL Query이 ORM보다 자세하기 때문에 수정이 상대적으로 용이하다

- ORM이 지정해 놓은 명령은 제한이 있지만, SQL 에는 더 다양한 문법이 있기 때문에 자유로움

- ORM 보다 빠르다



학습 후기

- 이번 주차 워크북을 해결해보면서 어땠는지 회고해봅시다.
- 핵심 키워드에 대해 완벽하게 이해했는지? 혹시 이해가 안 되는 부분은 무엇였는지?



스터디 진행 방법

1. 스터디를 진행하기 전, 워크북 내용들을 모두 채우고 스터디에서는 서로 모르는 내용들을 공유해주세요.
2. 미션은 워크북 내용들을 모두 완료하고 나서 스터디 전/후로 진행해보세요.
3. 다음주 스터디를 진행하기 전, 지난주 미션을 서로 공유해서 상호 피드백을 진행하시면 됩니다.

✓ 실습 체크리스트

- ✓ SQL 마스터 도전
- ✓ 여러 요구 사항에 대응하기
- ✓ 페이지네이션 배워보기
- ✓ 쿼리 실행 흐름 알아보기

✓ 실습 인증

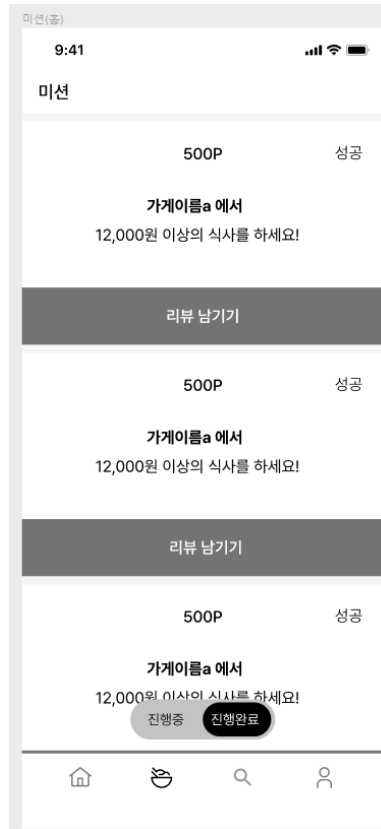
✨ 위클리 스크럼 과제

위클리 스크럼 과제는 OO주차 위클리 스크럼 아이스 브레이킹과 공통 질문을 모두 작성한 후, 아래에 질문을 복사 붙여넣기 한 다음 다같이 작성해주시면 됩니다.

1. ERD를 설계할 때 N:M 관계를 설계해야 할 일이 많은데 어떤 기준으로 매핑 테이블을 설계했나요?
 - 관계형 데이터베이스는 N:M 을 직접적으로 저장할 수 없기 때문에 1:N-1:N 관계로 으로 설계했습니다
 - 사용자 리뷰 같은 매핑 테이블은 추가 속성이 필요 없다고 생각되어 기본키로만 속성을 구성하였고,
사용자 미션/약관 같은 매핑 테이블에서는 각 사용자의 진행여부/성공여부/동의여부가 필요하다고 생각되어 추가적으로 속성을 추가하였습니다.
2. DB를 설계하면서 처음에 그린 ERD에서 바뀐 부분이 있었나요? 어떤 부분이었고 왜 바꿨나요?
 - a. 1주차 미션 때 설계를 안 했던 부분이었기에 리뷰 ERD 설계를 추가하였습니다.
 - b. 사용자 테이블에서 주소 테이블을 따로 분리하여 설계하였습니다.
회원이입에서 주소 입력 시 전체 주소를 입력할텐데 이걸 값을 넘겨줄 때 따로 저장을 하지 않는다면 따로 동 부분을 뽑아내는 부분에서 더욱 번거로워질 거라고 생각했기 때문입니닷

🔥 미션

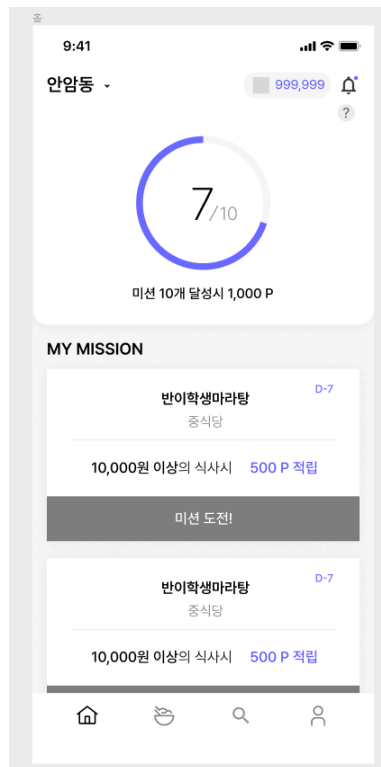
1. 1주차 때 설계한 데이터베이스를 토대로 아래의 화면에 대한 쿼리를 작성



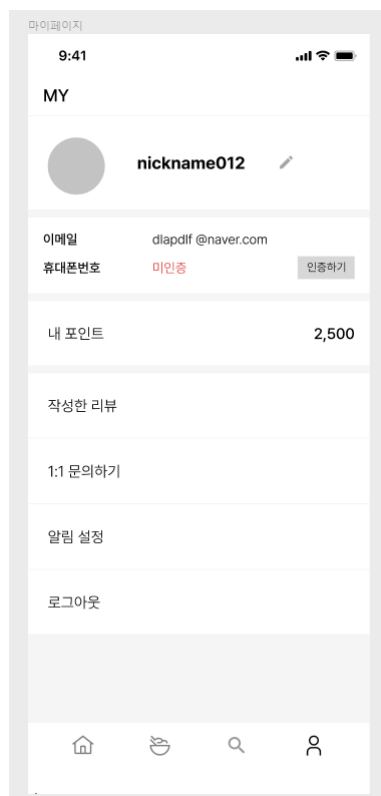
내가 진행중, 진행 완료한 미션 모아서 보는 쿼리(페이징 포함)



리뷰 작성하는 쿼리,
* 사진의 경우는 일단 배제



홈 화면 쿼리
(현재 선택 된 지역에서 도전이 가능한 미션 목록, 페이징 포함)



마이 페이지 화면 쿼리

< 시니어 미션 >

😊 시니어 미션

💪 **미션 기록 (여기에 해도 되고 위의 미션에서 각 페이지 밑에 간단하게 블록 만들어서 하셔도 됩니다!)**



미션 기록의 경우, 아래 미션 기록 토글 속에 작성하시거나, 페이지를 새로 생성하여 해당 페이지에 기록하여도 좋습니다!

하지만, 결과물만 올리는 것이 아닌, **중간 과정 모두 기록하셔야 한다는 점!** 잊지 말아주세요.

▼ 미션 기록

2주차 미션 기록

⚡ 트러블 슈팅



실습하면서 생긴 문제들에 대해서, **이슈 - 문제 - 해결** 순서로 작성해주세요.



스스로 해결하기 어렵다면? 스터디원들에게 도움을 요청하거나 **너디너리의 지식 IN 채널에 질문**해보세요!

▼ ⚡ **이슈 작성 예시** (이슈가 생기면 아래를 복사해서 No.1, No.2, No.3 ... 으로 작성해서 트러블 슈팅을 꼭 해보세요!)

이슈

👉 앱 실행 중에 노래 다음 버튼을 누르니까 앱이 종료되었다.

문제

👉 노래클래스의 데이터리스트의 Size를 넘어서 NullPointerException이 발생하여 앱이 종료된 것이었다.

해결

👉 노래 다음 버튼을 눌렀을 때 데이터리스트의 Size를 검사해 Size보다 넘어가려고 하면 다음으로 넘어가는 메서드를 실행시키지 않고, 첫 노래로 돌아가게끔 해결

참고레퍼런스

- 링크

▼ ⚡ 이슈 No.1

이슈

👉 [트러블이 생긴 상태 작성]

문제

👉 [어떤 이유로 해당 이슈가 일어났는지 작성]

해결

👉 [해결 방법 작성]

참고레퍼런스

- [문제 해결 시 참고한 링크]

🤔 참고 자료

2주차

Aa 이름	🕒 생성일	☰ 태그
커서 기반 페이지 네이션 참고 자료	@2025년 9월 18일 오후 5:56	

Copyright © 2023 최용욱(똥이) All rights reserved.

Copyright © 2024, 2025 제이미(김준환) All rights reserved.

Copyright © 2025 전하경(재서) All rights reserved.

