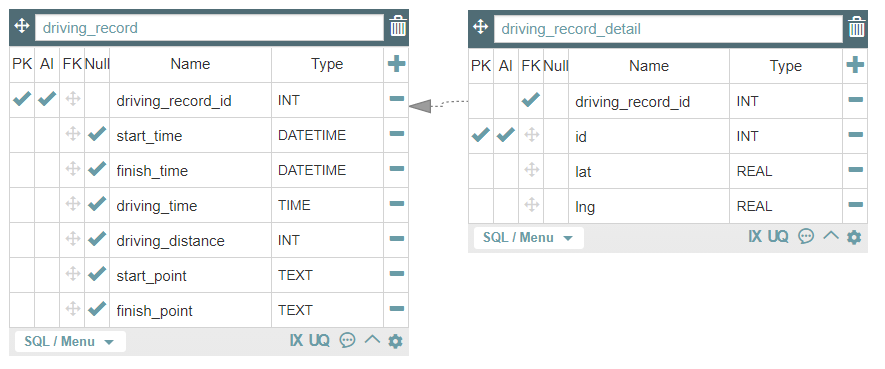
AutoDistanceDriven 설계

1. DB
   1. 필요 DB
      1. 주행 기록
      2. 실시간 주행 기록 - 3초마다 위치 기록(주행 기록id 포린키)
      3. ERD



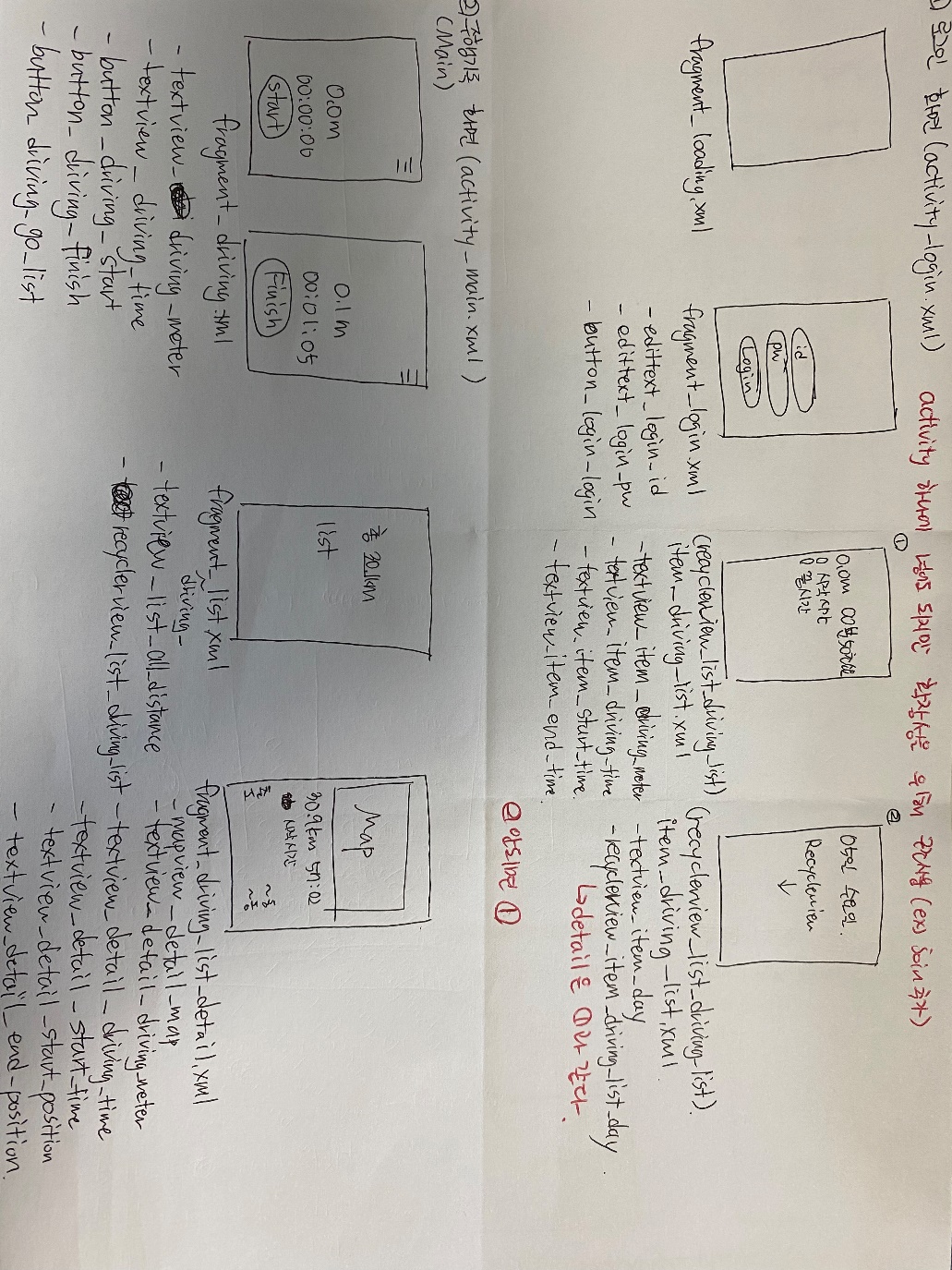
<https://araikuma.tistory.com/681> - SQLite 가능 데이터 타입

실시간 주행 기록을 기록하기 위해서는 주행 기록이 먼저 생성되야한다. => 포린키 실제 생성 X

필요 쿼리

* + 1. driving\_record CUR
       1. Update – 주행기록 업데이트
       2. Read 1 – 주행기록 가져오기
       3. Read 2 – driving\_record\_id에 따라 주행기록 가져오기
       4. Read 3 – 주행기록 중 start\_time의 Day만 가져오기
       5. Read 4 – 최신 주행기록 id 가져오기
    2. driving\_record CR
       1. Read 1 - driving\_record\_id에 따라 주행 기록 상세 가져오기

1. 화면 설계 – 후에 디자인 적용
   1. 로그인 엑티비티
   2. 주행(main) 엑티비티



1. MVVM 설계 - 도메인 중심 패키지(기본패키지: co.kr.daou.autodistancedriven)
   1. feature
      1. login
         1. LoginActivity.kt
         2. LoginViewModel.kt
         3. fragment
            1. LoginFragment.kt
            2. LoadingFragment.kt(넣을지 말지 생각 중)
      2. driving
         1. MainActivity.kt
         2. DrivingViewModel.kt
         3. fragment
            1. DrivingFragment.kt
            2. DrivingListFragment.kt
            3. DrivingListDetailFragment.kt
         4. adapter
            1. DrivingRecyclerviewAdapter.kt
            2. DrivingRecyclerviewDayAdapter.kt(고민)
   2. global
      1. Application.kt
   3. model
      1. db
         1. DrivingDatabase.kt
         2. repository
            1. DrivingRepository.kt
         3. dto
            1. DrivingRecordDto.kt
            2. DrivingRecordDetailDto.kt
            3. UserDto.kt
      2. network
         1. api
   4. util
2. 기능 설계

|  |  |
| --- | --- |
| View | ViewModel |
| LoginFragment | LoginViewModel |
| 1. 로그인 세션 확인  LoginViewModel.islogin() true(메인페이지 이동), false(로그인 페이지 유지) 2. Login 버튼 클릭 - LoginViewModel.login(id,pw) true(메인페이지 이동), false(alert) | * isLogin() : boolean 저장된 cookie로 세션 만료 확인 * login(id, pw) : boolean 입력된 아이디 패스워드로 post요청 응답 성공시 cookie 저장 |
| DrivingFragment | DrivingViewModel |
| 1. 초기에 필요 ViewModel 없음 2. List 버튼 클릭 시 DrivingListFragment로 이동 3. Start 버튼 클릭 시 - DrivingViewModel.start() - ViewModel의 data 관찰 => UI 변화 - Start 버튼 없애고 Finish 버튼 보이기 4. Finish 버튼 클릭 시 - DrivingViewModel.finish() - DrivingListDetailFragment로 이동 | * start() 주행 기록 생성(startTime, startPoint)   생성한 주행 기록 id 값 가져오기 백그라운드{   3초마다 gps 위치 db 저장(id, lat, lng)  => DB 저장시 insert 몇번? 실제 시간 측정해보기  => 일정 시간마다 백그라운드 처리  서비스 or AlramManager(elapsed real time) 사용  주행거리 3초마다 데이터 변경 (livedata)  주행시간 1초씩 데이터 변경 (livedata)  }   * finish()  백그라운드 처리 종료   주행 기록 업데이트   * getRecord() – 주행기록 전부가져오기 * getRecordByID() – 주행기록 ID로 찾아서 가져오기 * getRecordDetailByID() – 주행기록 디테일 아이디로 가져오기 |
| DrivingListFragment |
| 1. 초기 필요 데이터 - 리스트리사이클러뷰에 들어갈 날짜 - 데이리스트리사이클러뷰에 들어갈 전체 주행기록 => 리사이클러뷰 어뎁터에 넣기 2. 리사이클러뷰 item 클릭 시 detail 페이지로 이동 |
| DrivingListDetailFragment |
| 1. 리사이클러뷰에서 넘어온 id 값에 따른 초기 데이터 가져오기 => 리사이클러뷰 어뎁터에서 처리 2. 뒤로가기 시 원래 있던 자리로 가기 |