



실전 프로젝트1





10

TodoList SQLite Version (1)

Java - SQLite 연동 Review

DB를 다루는 전체적인 흐름
DB들끼리 거의 다 비슷함

1. 사전 준비 - DB설계, DB 파일 생성, Java Project 내에 DB 파일 복사

2. DB Connection 생성

```
String dbFile = "<db filename>";
conn = DriverManager.getConnection("jdbc:sqlite:" + dbFile);
```

3. SQL 실행

Ⓡ

```
Statement stmt = conn.createStatement();
String sql = "SELECT * FROM <table name> WHERE <conditions>";
ResultSet rs = stmt.executeQuery(sql);
```

ⓈⓊⓈ

```
Statement stmt = conn.createStatement();
String sql = "INSERT INTO <table name> (fields...) VALUES (values...)";
int count = stmt.executeUpdate(sql);
```

4. SQL 결과 사용

Ⓡ

```
while (rs.next()) {
    int id = rs.getInt("<numeric field>");
    String str = rs.getString("<string field>");
    System.out.println(id + " " + str);
}
```

```
if(count > 0)
    ;// Query 성공!
else
    ;// Query 실패!
```

5. Statement closing

```
stmt.close();
```

ⓈⓊⓈ

Java - SQLite 연동 Review

```
Statement stmt = conn.createStatement();
String sql = "insert into g_artists (name, a_type, a_year, debut, regdate)"
    + " values ('방탄소년단', '남성', '2010년대', '2013년', datetime('now', 'localtime'))";
int count = stmt.executeUpdate(sql);

if(count > 0)
    System.out.println("새로운 데이터가 추가되었습니다!");
else
    System.out.println("[Error] 데이터 추가 오류!");
stmt.close();
```

```
String sql = "insert into g_artists (name, a_type, a_year, debut, regdate)"
    + " values (?, ?, ?, ?, datetime('now', 'localtime'))";
PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setString(1, "방탄소년단");
pstmt.setString(2, "남성");
pstmt.setString(3, "2010년대");
pstmt.setString(4, "2013년");
int count = pstmt.executeUpdate();

if(count > 0)
    System.out.println("새로운 데이터가 추가되었습니다!");
else
    System.out.println("[Error] 데이터 추가 오류!");
pstmt.close();
```

ToDoListSQLApp Project

- ToDoListApp Project 복제하여 생성
- todolist.db 생성 / DbConnect 클래스 제작
- 초기 데이터 이전
- ToDoList method 수정 - ArrayList를 다루는 방식에서 DB를 다루는 방식으로 변경
 - C : addItem()
 - R : getItem(), getList()
 - U : updateItem() *→ 새로 만들어야 하는 것*
 - D : deleteItem()
 - 기타 : getCount()
- ToDoUtil, ToDoMain 과의 연동 부분 check!
 - *당라리게 변경 없음*

데이터베이스(todolist.db) 생성

DB Browser4

SQL CMD 에게 만들어 줄

Field name	설명	타입	비고
id	일련번호	Integer	PK Primary key
title	제목	text	
memo (= desc)	내용	text	
category	카테고리	text	
current_date	등록일	text	
due_date	마감일	text	

```
public class TodoItem {  
    private String title;  
    private String desc;  
    private String current_date;  
    private String category;  
    private String due_date;  
}
```

```
CREATE TABLE list (  
    id INTEGER NOT NULL,  
    title TEXT NOT NULL,  
    memo TEXT,  
    category TEXT NOT NULL,  
    current_date TEXT NOT NULL,  
    due_date TEXT,  
    PRIMARY KEY(id AUTOINCREMENT)  
);
```

DB Connection 관리

- DbConnect.java
 - static members
 - Connection 객체 생성
 - getConnection()
 - 연결 만들어 리턴
 - closeConnection()
 - 해제
- 다른 곳에서 사용하려면

```
public class TodoList {  
    Connection conn;  
  
    public TodoList() {  
        this.conn = DbConnect.getConnection();  
    }  
}
```

```
package com.todo.service;  
  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
  
public class DbConnect {  
    private static Connection conn = null;  
  
    public static void closeConnection() {  
        if (conn != null) {  
            try {  
                conn.close();  
            } catch (SQLException e) {  
                e.printStackTrace();  
            }  
        }  
    }  
  
    public static Connection getConnection() {  
        if (conn == null) {  
            try {  
                Class.forName("org.sqlite.JDBC");  
                conn = DriverManager.getConnection("jdbc:sqlite:" + "todolist.db");  
            } catch (Exception e) {  
                e.printStackTrace();  
            }  
        }  
        return conn;  
    }  
}
```

초기 데이터 이전

```
public void importData(String filename) {
    try {
        BufferedReader br = new BufferedReader(new FileReader(filename));
        String line;
        String sql = "insert into list (title, memo, category, current_date, due_date)"
            + " values (?, ?, ?, ?, ?)";
        int records = 0;
        while((line = br.readLine()) != null) {
            StringTokenizer st = new StringTokenizer(line, "##");
            String category = st.nextToken();
            String title = st.nextToken();
            String description = st.nextToken();
            String due_date = st.nextToken();
            String current_date = st.nextToken();
```

```
        PreparedStatement pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, title);
        pstmt.setString(2, description);
        pstmt.setString(3, category);
        pstmt.setString(4, current_date);
        pstmt.setString(5, due_date);
        int count = pstmt.executeUpdate();
        if(count > 0) records++;
        pstmt.close();
    }
    System.out.println(records + " records read!!");
    br.close();
}
```

```
} catch (Exception e) {
    e.printStackTrace();
}
```

ToDoList

```
public class TodoMain {
    public static void start() {
        Scanner sc = new Scanner(System.in);
        ToDoList l = new ToDoList();
        l.importData("todolist.txt");
    }
}
```

이런거! : 한 번만 실행하고 지워야 함

```
1 과제##2주차과제하기##운영체제, 알고리즘 과제하기##2021/09/15##2021/09/01 10:00:01
2 기타##동아리모임##동아리 동기들과 밥먹기##2021/09/22##2021/09/02 12:10:21
3 취미##산책하기##평봉필드 2바퀴 돌기##2021/09/05##2021/09/03 20:30:31
4 과제##세미나자료찾기##랩실 세미나 자료 준비하기##2021/09/17##2021/09/07 09:20:45
5 과제##3주차과제하기##운영체제, 알고리즘 과제하기##2021/09/21##2021/09/10 10:00:01
6 기타##동아리회식##동아리 동기들과 밥먹기##2021/09/29##2021/09/12 12:10:21
7 취미##운동##1킬로미터 빠르게 걷기##2021/09/15##2021/09/13 20:30:31
8 과제##세미나발표##랩실 세미나 준비하기##2021/09/27##2021/09/14 09:20:45
9 |
```

	id	title	memo	category	current_date	due_date
	...	필터	필터	필터	필터	필터
1	1	2주차과제하기	운영체제, 알고리즘 과제하기	과제	2021/09/01 10:00:01	2021/09/15
2	2	동아리모임	동아리 동기들과 밥먹기	기타	2021/09/02 12:10:21	2021/09/22
3	3	산책하기	평봉필드 2바퀴 돌기	취미	2021/09/03 20:30:31	2021/09/05
4	4	세미나자료찾기	랩실 세미나 자료 준비하기	과제	2021/09/07 09:20:45	2021/09/17
5	5	3주차과제하기	운영체제, 알고리즘 과제하기	과제	2021/09/10 10:00:01	2021/09/21
6	6	동아리회식	동아리 동기들과 밥먹기	기타	2021/09/12 12:10:21	2021/09/29
7	7	운동	1킬로미터 빠르게 걷기	취미	2021/09/13 20:30:31	2021/09/15
8	8	세미나발표	랩실 세미나 준비하기	과제	2021/09/14 09:20:45	2021/09/27

TodoList method 수정 - addItem()

```
public int addItem(TodoItem t) {  
    String sql = "insert into list (title, memo, category, current_date, due_date)"  
        + " values (?, ?, ?, ?, ?)";  
    PreparedStatement pstmt;  
    int count=0;  
    try {  
        pstmt = conn.prepareStatement(sql);  
        pstmt.setString(1,t.getTitle());  
        pstmt.setString(2,t.getDesc());  
        pstmt.setString(3,t.getCategory());  
        pstmt.setString(4,t.getCurrent_date());  
        pstmt.setString(5,t.getDue_date());  
        count = pstmt.executeUpdate();  
        pstmt.close();  
    } catch (SQLException e) {  
        e.printStackTrace();  
    }  
    return count;  
}
```

```
public class TodoUtil {  
  
    public static void createItem(TodoList l) {  
  
        String title, desc, category, due_date;  
        Scanner sc = new Scanner(System.in);  
  
        System.out.print("[항목 추가]\n"  
            + "제목 > ");  
        title = sc.next();  
        if (l.isDuplicate(title)) {  
            System.out.println("제목이 중복됩니다!");  
            return;  
        }  
        System.out.print("카테고리 > ");  
        category = sc.next();  
        sc.nextLine();  
        System.out.print("내용 > ");  
        desc = sc.nextLine().trim();  
        System.out.print("마감일자 > ");  
        due_date = sc.nextLine().trim();  
  
        TodoItem t = new TodoItem(title, desc, category, due_date);  
        if (l.addItem(t)>0)  
            System.out.println("추가되었습니다.");  
    }  
}
```

이단 건너뛰고
다음번에 고빈

TodoList method 수정 - getList(), getCount()

```
public ArrayList<TodoItem> getList() {
    ArrayList<TodoItem> list = new ArrayList<TodoItem>();
    Statement stmt;
    try {
        stmt = conn.createStatement();
        String sql = "SELECT * FROM list";
        ResultSet rs = stmt.executeQuery(sql);
        while(rs.next()) {
            int id = rs.getInt("id");
            String category = rs.getString("category");
            String title = rs.getString("title");
            String description = rs.getString("memo");
            String due_date = rs.getString("due_date");
            String current_date = rs.getString("current_date");
            TodoItem t = new TodoItem(title, description, category, due_date);
            t.setId(id);
            t.setCurrent_date(current_date);
            list.add(t);
        }
        stmt.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return list;
}
```

```
public int getCount() {
    Statement stmt;
    int count=0;
    try {
        stmt = conn.createStatement();
        String sql = "SELECT count(id) FROM list;";
        ResultSet rs = stmt.executeQuery(sql);
        rs.next();
        count = rs.getInt("count(id)");
        stmt.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return count;
}
```

```
public static void listAll(TodoList l) {
    System.out.printf("[전체 목록, 총 %d개]\n", l.getCount());
    for (TodoItem item : l.getList()) {
        System.out.println(item.toString());
    }
}
```

TodoUtil

ls, add

Command > **ls**

[전체 목록, 총 8개]

- 1 [과제] 2주차과제하기 - 운영체제, 알고리즘 과제하기 - 2021/09/15 - 2021/09/01 10:00:01
- 2 [기타] 동아리모임 - 동아리 동기들과 밥먹기 - 2021/09/22 - 2021/09/02 12:10:21
- 3 [취미] 산책하기 - 평봉필드 2바퀴 돌기 - 2021/09/05 - 2021/09/03 20:30:31
- 4 [과제] 세미나자료찾기 - 랩실 세미나 자료 준비하기 - 2021/09/17 - 2021/09/07 09:20:45
- 5 [과제] 3주차과제하기 - 운영체제, 알고리즘 과제하기 - 2021/09/21 - 2021/09/10 10:00:01
- 6 [기타] 동아리회식 - 동아리 동기들과 밥먹기 - 2021/09/29 - 2021/09/12 12:10:21
- 7 [취미] 운동 - 1킬로미터 빠르게 걷기 - 2021/09/15 - 2021/09/13 20:30:31
- 8 [과제] 세미나발표 - 랩실 세미나 준비하기 - 2021/09/27 - 2021/09/14 09:20:45

Command > **add**

[항목 추가]

제목 > **SQL공부**

카테고리 > **과제**

내용 > **SQLite 공부하기**

마감일자 > **2021/10/05**

추가되었습니다.

Command > **ls**

[전체 목록, 총 9개]

- 1 [과제] 2주차과제하기 - 운영체제, 알고리즘 과제하기 - 2021/09/15 - 2021/09/01 10:00:01
- 2 [기타] 동아리모임 - 동아리 동기들과 밥먹기 - 2021/09/22 - 2021/09/02 12:10:21
- 3 [취미] 산책하기 - 평봉필드 2바퀴 돌기 - 2021/09/05 - 2021/09/03 20:30:31
- 4 [과제] 세미나자료찾기 - 랩실 세미나 자료 준비하기 - 2021/09/17 - 2021/09/07 09:20:45
- 5 [과제] 3주차과제하기 - 운영체제, 알고리즘 과제하기 - 2021/09/21 - 2021/09/10 10:00:01
- 6 [기타] 동아리회식 - 동아리 동기들과 밥먹기 - 2021/09/29 - 2021/09/12 12:10:21
- 7 [취미] 운동 - 1킬로미터 빠르게 걷기 - 2021/09/15 - 2021/09/13 20:30:31
- 8 [과제] 세미나발표 - 랩실 세미나 준비하기 - 2021/09/27 - 2021/09/14 09:20:45
- 11 [과제] SQL공부 - SQLite 공부하기 - 2021/10/05 - 2021/10/05 24:27:07

Command >

TodoList method 수정 - updateItem()

```
public int updateItem(TodoItem t) {
    String sql = "update list set title=?, memo=?, category=?, current_date=?, due_date=?"
        + " where id = ?;";
    PreparedStatement pstmt;
    int count=0;
    try {
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1,t.getTitle());
        pstmt.setString(2,t.getDesc());
        pstmt.setString(3,t.getCategory());
        pstmt.setString(4,t.getCurrent_date());
        pstmt.setString(5,t.getDue_date());
        pstmt.setInt(6,t.getId());
        count = pstmt.executeUpdate();
        pstmt.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return count;
}
```

```
public static void updateItem(TodoList l) {
```

```
    String new_title, new_desc, new_category, new_due_date;
    Scanner sc = new Scanner(System.in);
```

```
    System.out.print("[항목 수정]\n"
        + "수정할 항목의 번호를 입력하십시오 > ");
    int index = sc.nextInt();
```

```
    System.out.print("새 제목 > ");
    new_title = sc.next().trim();
    System.out.print("새 카테고리 > ");
    new_category = sc.next();
    sc.nextLine();
    System.out.print("새 내용 > ");
    new_desc = sc.nextLine().trim();
    System.out.print("새 마감일자 > ");
    new_due_date = sc.nextLine().trim();
```

```
    TodoItem t = new TodoItem(new_title, new_desc, new_category, new_due_date);
    t.setId(index);
    if(l.updateItem(t) > 0)
        System.out.println("수정되었습니다.");
}
```

TodoUtil

edit

Command > edit

[항목 수정]

수정할 항목의 번호를 입력하시오 > 11

새 제목 > SQL연습

새 카테고리 > 과제

새 내용 > SQLite 연습하기

새 마감일자 > 2021/10/10

수정되었습니다.

Command > ls

[전체 목록, 총 9개]

- 1 [과제] 2주차과제하기 - 운영체제, 알고리즘 과제하기 - 2021/09/15 - 2021/09/01 10:00:01
- 2 [기타] 동아리모임 - 동아리 동기들과 밥먹기 - 2021/09/22 - 2021/09/02 12:10:21
- 3 [취미] 산책하기 - 평봉필드 2바퀴 돌기 - 2021/09/05 - 2021/09/03 20:30:31
- 4 [과제] 세미나자료찾기 - 랩실 세미나 자료 준비하기 - 2021/09/17 - 2021/09/07 09:20:45
- 5 [과제] 3주차과제하기 - 운영체제, 알고리즘 과제하기 - 2021/09/21 - 2021/09/10 10:00:01
- 6 [기타] 동아리회식 - 동아리 동기들과 밥먹기 - 2021/09/29 - 2021/09/12 12:10:21
- 7 [취미] 운동 - 1킬로미터 빠르게 걷기 - 2021/09/15 - 2021/09/13 20:30:31
- 8 [과제] 세미나발표 - 랩실 세미나 준비하기 - 2021/09/27 - 2021/09/14 09:20:45
- 11 [과제] SQL연습 - SQLite 연습하기 - 2021/10/10 - 2021/10/05 24:29:53

Command >

TodoList method 수정 - deleteItem()

```
public int deleteItem(int index) {
    String sql = "delete from list where id=?";
    PreparedStatement pstmt;
    int count=0;
    try {
        pstmt = conn.prepareStatement(sql);
        pstmt.setInt(1,index);
        count = pstmt.executeUpdate();
        pstmt.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return count;
}
```

TodoUtil

```
public static void deleteItem(TodoList l) {
    Scanner sc = new Scanner(System.in);

    System.out.print("[항목 삭제]\n"
        + "삭제할 항목의 번호를 입력하시오 > ");
    int index = sc.nextInt();
    if (l.deleteItem(index)>0)
        System.out.println("삭제되었습니다.");
}
```

del

Command > del

[항목 삭제]

삭제할 항목의 번호를 입력하시오 > 11

삭제되었습니다.

Command > ls

[전체 목록, 총 8개]

- 1 [과제] 2주차과제하기 - 운영체제, 알고리즘 과제하기 - 2021/09/15 - 2021/09/01 10:00:01
- 2 [기타] 동아리모임 - 동아리 동기들과 밥먹기 - 2021/09/22 - 2021/09/02 12:10:21
- 3 [취미] 산책하기 - 평봉필드 2바퀴 돌기 - 2021/09/05 - 2021/09/03 20:30:31
- 4 [과제] 세미나자료찾기 - 랩실 세미나 자료 준비하기 - 2021/09/17 - 2021/09/07 09:20:45
- 5 [과제] 3주차과제하기 - 운영체제, 알고리즘 과제하기 - 2021/09/21 - 2021/09/10 10:00:01
- 6 [기타] 동아리회식 - 동아리 동기들과 밥먹기 - 2021/09/29 - 2021/09/12 12:10:21
- 7 [취미] 운동 - 1킬로미터 빠르게 걷기 - 2021/09/15 - 2021/09/13 20:30:31
- 8 [과제] 세미나발표 - 랩실 세미나 준비하기 - 2021/09/27 - 2021/09/14 09:20:45

Command >