

```
1 Sqlite3 Commands
2
3 1. Shell Mode
4 - Windows 의 명령 프롬프트나 리눅스의 셸과 같이 SQLite3 엔진과 대화식으로 명령어를 전달하
  고 그에 대한 결과를 받을 수 있다.
5 - 이 모드는 개발 과정에서 SQL 구문을 작성하거나 디버깅 과정에서 데이터베이스의 내용을 확인할
  때 가장 많이 사용된다.
6 1) 데이터베이스 생성
7   - SQLite3는 기본적으로 데이터베이스 파일 하나에 테이블, 레코드, 인덱스, 설정 정보 등 모
  든 데이터를 저장한다.
8   - 아래와 같이 데이터베이스를 커맨드 라인 인자로 전달하여 생성한다.
9   sqlite3 [데이터베이스 파일명]
10
11 D:\temp>sqlite3 test.db
12 SQLite version 3.8.6 2014-08-15 11:46:33
13 Enter ".help" for usage hints.
14 sqlite>
15   - 셸을 종료하려면 .exit를 입력한다.
16     sqlite> .exit
17
18 ※ SQLite3 의 셸 모드에서 사용되는 명령어는 모두 마침표(.)로 시작하고 명령어 뒤에 세미콜
  론(;)을 붙이지 않는다.
19
20 - SQLite3 셸을 종료하고 데이터베이스 파일인 'test.db' 파일을 찾아보면 파일이 생성되어
  있지 않는 것을 발견한다. SQLite3에서는 기본적으로 실제 데이터베이스에 테이블 생성과 같이
  어떤 데이터가 입력되기 전까지는 인자로 전달된 데이터베이스 파일을 생성하지 않는다. 이는 페
  이지 크기(Page Size)나 인코딩(Encoding)과 같은 값을 실제 데이터베이스 파일이 생성되기
  전에 설정해야 하기 때문이다.
21 - 인자 없이 sqlite3 명령어만 실행하면 데이터베이스의 내용이 파일에 저장되지 않고 메모리
  내에서만 존재한다. 이 경우 sqlite3 셸이 실행되는 동안에만 정보가 유지되고, 셸을 종료하면
  모든 내용이 사라진다.
22 - 아래와 같이 .databases 명령어를 통해 test.db를 생성한다.
23 sqlite> .databases
24 seq  name                file
25 ---  -
26 0    main                D:\temp\test.db
27 sqlite>
28
29 2) SQL 문 수행
30 - sqlite3 셸에서 직접 SQL 문을 입력해서 수행할 수 있다.
31 - 만약 질의가 복잡하다면 줄 바꿈 문자를 입력해서 SQL 문을 여러 줄에 걸쳐 작성할 수 있다.
32 sqlite> CREATE TABLE EMP
33   ...> (
34   ...> EMPNO    INTEGER PRIMARY KEY,
35   ...> ENAME    TEXT NOT NULL,
36   ...> JOB      TEXT,
37   ...> SAL      INTEGER
```

```

38     ...> );
39  sqlite> CREATE INDEX IDX_EMPNO ON EMP(EMPNO);
40  sqlite> INSERT INTO EMP
41     ...> VALUES (7369, 'SMITH', 'CLERK', 800);
42  sqlite> INSERT INTO EMP
43     ...> VALUES (7499, 'ALLEN', 'SALESMAN', 1600);
44  sqlite> INSERT INTO EMP
45     ...> VALUES (7521, 'WARD', 'SALESMAN', 1250);
46  sqlite> INSERT INTO EMP
47     ...> VALUES (7566, 'JONES', 'MANAGER', 2975);
48
49  - SQL 문을 작성할 때 추후 유지보수를 위해 구문 중간에 주석을 입력해야 할 때가 있다. 한
    줄 주석인 경우에는 '--' 이후에 주석 내용을 입력하고, 여러 줄에 걸친 주석의 경우 C언어와
    동일하게 '/*' 과 '*/' 사이에 주석 내용을 작성한다.
50
51  3) 테이블 조회
52  - 레코드를 조회하는 SELECT 문도 다른 SQL 문과 같이 셸에서 수행한다.
53  - 아래와 같이 각 레코드마다 한 줄씩 표시되며, '|'를 구분자로 써서 각 필드를 구분하고 있
    다.
54  - 만일 NULL 값일 경우에는 출력할 때 아무것도 표시하지 않으므로 '||' 와 같이 구분자가
    연이어 화면에 나타난다.
55
56  sqlite> SELECT * FROM EMP;
57  7369|SMITH|CLERK|800
58  7499|ALLEN|SALESMAN|1600
59  7521|WARD|SALESMAN|1250
60  7566|JONES|MANAGER|2975
61  7654|MARTIN|SALESMAN|1250
62  7698|BLAKE||2850
63  7782|CLARK|MANAGER|
64  sqlite>
65
66  - 이번에는 출력되는 결과를 테이블과 같이 보기 쉽게끔 필드의 이름을 표시(.header ON) 하
    고, 각 필드는 공백으로 구분(.mode column)하도록 설정을 변경하겠다. NULL 값에 대해서
    도 공백으로 출력하는 대신 NULL 이라고 명시적으로 출력(.nullvalue NULL)하자.
67
68  sqlite> .header ON
69  sqlite> .mode column
70  sqlite> .nullvalue NULL
71  sqlite> SELECT * FROM EMP;
72  EMPNO      ENAME      JOB      SAL
73  -----
74  7369      SMITH      CLERK      800
75  7499      ALLEN      SALESMAN    1600
76  7521      WARD      SALESMAN    1250
77  7566      JONES      MANAGER     2975
78  7654      MARTIN     SALESMAN    1250

```

```

79      7698      BLAKE      NULL      2850
80      7782      CLARK      MANAGER    NULL
81      sqlite>
82
83      - 입력한 데이터가 길어서 잘리는 경우에는 '.width' 명령어로 출력될 필드의 길이를 조정할
      수 있다.
84      - 이때 '.width' 명령어 다음에 출력할 필드의 갯수만큼 길이를 나열하면 된다.
85
86      sqlite> .width 6 10 15 6
87      sqlite> SELECT * FROM EMP;
88      EMPNO      ENAME      JOB      SAL
89      -----
90      7369      SMITH      CLERK      800
91      7499      ALLEN      SALESMAN    1600
92      7521      WARD      SALESMAN    1250
93      7566      JONES      MANAGER     2975
94      7654      MARTIN    SALESMAN    1250
95      7698      BLAKE      NULL      2850
96      7782      CLARK      MANAGER     NULL
97      sqlite>
98
99      4) 데이터베이스 및 테이블 스키마 정보
100     - .tables 명령어를 사용하면 한 데이터베이스가 포함하고 있는 테이블의 이름을 조회할 수 있
      다.
101
102     sqlite> .tables
103     EMP
104     sqlite>
105     - .indices 명령어는 특정 테이블에 포함된 인덱스의 이름을 모두 출력한다. 이 때 테이블의
      이름을 지정하면 그 테이블의 인덱스를 출력한다.
106     sqlite> .indices
107     IDX_EMPNO
108     sqlite>
109
110     - 다른 데이터베이스의 DESC 명령어 처럼 테이블의 스키마, 인덱스와 같은 테이블의 정보를
      SQL 형태로 조회하려면 아래와 같이 .schema table_name 으로 할 수 있다.
111     sqlite> .schema EMP
112     CREATE TABLE EMP
113     (
114     EMPNO      INTEGER PRIMARY KEY,
115     ENAME      TEXT NOT NULL,
116     JOB        TEXT,
117     SAL        INTEGER
118     );
119     CREATE INDEX IDX_EMPNO ON EMP(EMPNO);
120     sqlite>
121

```

122 - DDL 구문으로 생성되는 모든 정보는 `sqlite_master` 라는 시스템 테이블에 저장된다. 이
테이블에는 타입(`type`), 이름(`name`), 생성한 테이블 이름(`tbl_name`), 루트 페이지 번호
(`rootpage`), SQL 구문(`sql`)이 저장되며, 다음과 같이 개발자가 생성한 테이블과 마찬가지로
SELECT 문으로 조회할 수 있다.

```
123
124 sqlite> SELECT * FROM sqlite_master;
125 type      name      tbl_name      rootpa  sql
126 -----
127 table      EMP      EMP      2      CREATE TABLE EMP
128           (
129           EMPNO    INTEGER PRIMARY KEY,
130           ENAME    TEXT NOT NULL,
131           JOB      TEXT,
132           SAL      INTEGER
133           )
134 index      IDX_EMPNO  EMP      3      CREATE INDEX IDX_EMPNO ON
135           EMP(EMPNO)
136
137 sqlite>
```

137 5) 데이터 추출

138 - `dump` 명령어를 이용하면 특정 테이블의 내용을 추출할 수 있다. 이 때 특정 테이블과 관련
된 스키마, 트리거, 인덱스 정보를 비롯해 테이블에 저장된 레코드까지 추출할 수 있다.
139 - 또한 바이너리 포맷이 아닌 SQL 구분 형태로 추출하므로 SQLite3뿐만 아니라 다른 DBMS
에서 추출한 데이터를 가지고 동일한 데이터베이스를 구축할 수 있다.
140 - 이 때 추출되는 SQL 구문은 명시적으로 트랙잭션을 사용해서 해당 데이터베이스에 온전히 작
용하게 된다
141 - `.dump` 명령어에 인자가 없으면 데이터베이스의 모든 내용을 추출한다.

```
142
143 sqlite> .dump EMP
144 PRAGMA foreign_keys=OFF;
145 BEGIN TRANSACTION;
146 CREATE TABLE EMP
147 (
148 EMPNO    INTEGER PRIMARY KEY,
149 ENAME    TEXT NOT NULL,
150 JOB      TEXT,
151 SAL      INTEGER
152 );
153 INSERT INTO "EMP" VALUES(7369,'SMITH','CLERK',800);
154 INSERT INTO "EMP" VALUES(7499,'ALLEN','SALESMAN',1600);
155 INSERT INTO "EMP" VALUES(7521,'WARD','SALESMAN',1250);
156 INSERT INTO "EMP" VALUES(7566,'JONES','MANAGER',2975);
157 INSERT INTO "EMP" VALUES(7654,'MARTIN','SALESMAN',1250);
158 INSERT INTO "EMP" VALUES(7698,'BLAKE',NULL,2850);
159 INSERT INTO "EMP" VALUES(7782,'CLARK','MANAGER',NULL);
160 CREATE INDEX IDX_EMPNO ON EMP(EMPNO);
```

```
161 COMMIT;
162 sqlite>
163
164     - 기본적으로 화면에 추출되는 내용을 특정 파일에 저장하려면 .output 명령어를 사용하면 된다.
165     - 저장할 파일의 이름을 인자로 전달하면 화면에 내용이 해당 파일에 저장된다.
166
167 sqlite> .output EMP.sql
168 sqlite> .dump EMP
169 sqlite>
170
171     - .output 명령어를 사용해서 출력 파일을 지정하고 난 이후, sqlite3 셸에서 수행해서 나온 모든 결과는 화면이 아니라 파일에 저장된다.
172     - 원래대로 화면에 출력하려면 다음과 같은 명령어를 사용한다.
173
174 sqlite>
175 sqlite> .output stdout
176 sqlite>
177
178 6) 파일에 저장된 SQL 구문의 실행
179     - .dump 명령어로 파일에 저장된 SQL 구문을 읽어서 수행해 보자.
180     - .read 명령어를 이용하면 파일에 저장된 SQL 구문을 수행할 수 있다.
181     - 수행할 파일 이름을 인자로 전달하면 된다.
182
183 sqlite> .exit
184
185 D:\temp>sqlite3 test2.db
186 SQLite version 3.8.6 2014-08-15 11:46:33
187 Enter ".help" for usage hints.
188 sqlite> .read EMP.sql
189
190 sqlite> .schema
191 CREATE TABLE EMP
192 (
193 EMPNO    INTEGER PRIMARY KEY,
194 ENAME    TEXT NOT NULL,
195 JOB      TEXT,
196 SAL      INTEGER
197 );
198 CREATE INDEX IDX_EMPNO ON EMP(EMPNO);
199 sqlite>
200
201     - .read 명령어를 수행한 후, .schema 명령어로 생성된 스키마를 확인하고, SELECT 문으로 데이터를 조회하면 정상적으로 데이터가 복원된 것을 확인할 수 있다.
202
203 sqlite> SELECT * FROM EMP;
204 7369|SMITH|CLERK|800
```

```
205 7499|ALLEN|SALESMAN|1600
206 7521|WARD|SALESMAN|1250
207 7566|JONES|MANAGER|2975
208 7654|MARTIN|SALESMAN|1250
209 7698|BLAKE| |2850
210 7782|CLARK|MANAGER|
211 sqlite>
```

7) 출력 형식 변경

- .mode 명령어를 이용하면 HTML 형식으로 결과를 출력할 수 있다.

```
215
216 sqlite> .mode html
217 sqlite> SELECT * FROM EMP;
218 <TR><TD>7369</TD>
219 <TD>SMITH</TD>
220 <TD>CLERK</TD>
221 <TD>800</TD>
222 </TR>
223 <TR><TD>7499</TD>
224 <TD>ALLEN</TD>
225 <TD>SALESMAN</TD>
226 <TD>1600</TD>
227 </TR>
228 <TR><TD>7521</TD>
229 <TD>WARD</TD>
230 <TD>SALESMAN</TD>
231 <TD>1250</TD>
232 </TR>
233 <TR><TD>7566</TD>
234 <TD>JONES</TD>
235 <TD>MANAGER</TD>
236 <TD>2975</TD>
237 </TR>
```

- 엑셀에서 자주 사용되는 CSV 형식으로도 출력가능하다.

```
238
239
240 sqlite> .mode csv
241 sqlite> SELECT * FROM EMP;
242 7369,SMITH,CLERK,800
243 7499,ALLEN,SALESMAN,1600
244 7521,WARD,SALESMAN,1250
245 7566,JONES,MANAGER,2975
246 7654,MARTIN,SALESMAN,1250
247 7698,BLAKE,,2850
248 7782,CLARK,MANAGER,
249 sqlite>
```

- 또한 다음과 같이 탭(tab)으로도 각 필드를 구분할 수 있다.

```
252  sqlite> .mode tabs
253  sqlite> SELECT * FROM EMP;
254  7369      SMITH      CLERK      800
255  7499      ALLEN      SALESMAN      1600
256  7521      WARD      SALESMAN      1250
257  7566      JONES      MANAGER 2975
258  7654      MARTIN     SALESMAN      1250
259  7698      BLAKE      2850
260  7782      CLARK      MANAGER
261  sqlite>
262
263  8) 기타 설정
264      - .echo 명령어를 사용하면 SQL 문을 수행할 때 명령어를 다시 화면에 출력하는 기능을 활성화
        하거나 (ON), 비활성화(OFF)할 수 있다.
265      - 이 기능이 활성화되면 SQL 구문을 수행하기 전에 해당 구문을 다시 한번 출력한다.
266      - 따라서 SELECT 문의 결과를 파일에 따로 남기는 경우 SELECT 문과 함께 저장되기 때문에
        나중에 참고하는데 도움이 된다.
267
268  sqlite> .echo ON
269  sqlite> SELECT * FROM EMP;
270  SELECT * FROM EMP;
271  7369      SMITH      CLERK      800
272  7499      ALLEN      SALESMAN      1600
273  7521      WARD      SALESMAN      1250
274  7566      JONES      MANAGER 2975
275  7654      MARTIN     SALESMAN      1250
276  7698      BLAKE      2850
277  7782      CLARK      MANAGER
278  sqlite>
279
280      - .separator 명령어는 SELECT 문에서 구분자를 변경할 때 사용한다.
281      - 명령어의 인자로 새로 사용할 구분자를 전달하면 된다.
282
283  sqlite> .separator %
284  sqlite> SELECT * FROM EMP;
285  7369%SMITH%CLERK%800
286  7499%ALLEN%SALESMAN%1600
287  7521%WARD%SALESMAN%1250
288  7566%JONES%MANAGER%2975
289  7654%MARTIN%SALESMAN%1250
290  7698%BLAKE%%2850
291  7782%CLARK%MANAGER%
292  sqlite>
293
294      - .prompt 명령어는 현재 출력되는 sqlite3 셸 모드의 프롬프트를 변경할 때 사용한다.
295      - 명령어의 인자로 변경할 프롬프트를 입력하면 현재 연결에 한해 변경할 프롬프트가 유지된다.
296  sqlite> .prompt SQLITE3>
```

```
297  SQLITE3>
298  SQLITE3>
299
300  - .timer 명령어를 이용해 SQL 문의 수행 시간 측정을 활성화(ON)하거나 비활성화(OFF)할
    수 있다.
301  sqlite> .timer ON
302  sqlite> SELECT * FROM EMP;
303  7369|SMITH|CLERK|800
304  7499|ALLEN|SALESMAN|1600
305  7521|WARD|SALESMAN|1250
306  7566|JONES|MANAGER|2975
307  7654|MARTIN|SALESMAN|1250
308  7698|BLAKE||2850
309  7782|CLARK|MANAGER|
310  Run Time: real 0.016 user 0.000000 sys 0.000000
311  sqlite>
312
313  9) 현재 설정된 상태 확인
314  - .show 명령어는 현재 설정된 상태값을 확인하는 데 사용된다.
315  sqlite> .header ON
316  sqlite> .mode column
317  sqlite> .width 8 15 20 8
318  sqlite> .nullvalue NULL
319  sqlite> .show
320      echo: off
321      eqp: off
322      explain: off
323      headers: on
324      mode: column
325      nullvalue: "NULL"
326      output: stdout
327      separator: "|" "\r\n"
328      stats: off
329      width: 8 15 20 8
330  sqlite>
331
332  10) 도움말 보기
333  - .help 명령어를 이용하면 sqlite3에서 수행할 수 있는 명령어와 해당 명령어의 옵션을 확인
    할 수 있다.
334  sqlite> .help
335
336
337
338
339
```