In [1]:
```
import sqlite3

con = sqlite3.connect('test1.db')
cur = con.cursor()
```

In [2]:
```
cur.execute('''
    SELECT city.name, supplier.name FROM city
    LEFT JOIN supplier
    ON city.pk = supplier.fk
    ''')
cur.fetchall()
```

Out[2]:
```
[('성북구', '안암1호점'),
 ('성북구', '안암2호점'),
 ('성북구', '종암1호점'),
 ('중구', None),
 ('강북구', None),
 ('어쩌구', None),
 ('저쩌구', None)]
```

In [3]:
```
cur.execute('SELECT * FROM city')
cur.fetchall()
```

Out[3]:
```
[(1, '성북구'), (2, '중구'), (3, '강북구'), (4, '어쩌구'), (5, '저쩌구')]
```

In [4]:
```
cur.execute('''
    SELECT city.name, supplier.name FROM city
    RIGHT JOIN supplier
    ON city.pk = supplier.fk
    ''')
cur.fetchall()
```

Out[4]:
```
[('성북구', '안암1호점'), ('성북구', '안암2호점'), ('성북구', '종암1호점')]
```

In [5]:
```
cur.execute('''
    SELECT city.name, supplier.name FROM supplier
    LEFT JOIN city
    ON city.pk = supplier.fk
    ''')
cur.fetchall()
```

Out[5]:
```
[('성북구', '안암1호점'), ('성북구', '안암2호점'), ('성북구', '종암1호점')]
```

In [6]:
```
cur.execute('''
    SELECT city.name, supplier.name FROM city
    OUTER JOIN supplier
    ON city.pk = supplier.fk
    ''')
cur.fetchall()
```

```
---------------------------------------------------------------------------
OperationalError                          Traceback (most recent call last)
Cell In [6], line 1
----> 1 cur.execute('''
      2     SELECT city.name, supplier.name FROM city
      3     OUTER JOIN supplier
      4     ON city.pk = supplier.fk
      5     ''')
      6 cur.fetchall()

OperationalError: unknown join type: OUTER
```

In [7]:
```
con.close()
```

In [8]:
```
con = sqlite3.connect('playlist.db')
cur = con.cursor()
```

In [9]:
```
cur.executescript('''
    CREATE TABLE artist (
        pk INTEGER PRIMARY KEY,
        name TEXT DEFAULT '무명'
    );

    CREATE TABLE album (
        pk INTEGER PRIMARY KEY,
        name TEXT DEFAULT '무제',
        fk INTEGER NOT NULL
    );

    CREATE TABLE genre (
        pk INTEGER PRIMARY KEY,
        name TEXT DEFAULT '장르없음'
    );

    CREATE TABLE track (
        pk INTEGER PRIMARY KEY,
        name TEXT DEFAULT 'Track',
        length INTEGER DEFAULT 0,
        rating INTEGER DEFAULT 0,
        count INTEGER DEFAULT 0,
        fk1 INTEGER NOT NULL,
        fk2 INTEGER NOT NULL
    );
''')
```

Out[9]:
```
<sqlite3.Cursor at 0x11042e490>
```

In [10]:
```
cur.execute('INSERT INTO artist VALUES(?,?)', [None, '가수1'])
```

Out[10]:
```
<sqlite3.Cursor at 0x11042e490>
```

In [11]:
```
cur.execute('SELECT * FROM artist')
cur.fetchall()
```

Out[11]:
```
[(1, '가수1')]
```

In [12]:
```
cur.execute('INSERT INTO artist VALUES(:pk,:name)',
            {'pk':None,'name':'가수2'})
```

```
Out[12]: <sqlite3.Cursor at 0x11042e490>
```

```
In [13]: cur.execute('SELECT * FROM artist')
         cur.fetchall()
```

```
Out[13]: [(1, '가수1'), (2, '가수2')]
```

```
In [14]: data = [['가수3'], ['가수4']]
         cur.executemany('INSERT INTO artist(name) VALUES(?)', data)
```

```
Out[14]: <sqlite3.Cursor at 0x11042e490>
```

```
In [15]: cur.execute('SELECT * FROM artist')
         cur.fetchall()
```

```
Out[15]: [(1, '가수1'), (2, '가수2'), (3, '가수3'), (4, '가수4')]
```

```
In [16]: cur.lastrowid
```

```
Out[16]: 4
```

```
In [17]: cur.executescript('''
             INSERT INTO genre(pk, name) VALUES(NULL, '장르1');
             INSERT INTO genre(name) VALUES('장르2');
             INSERT INTO genre VALUES(NULL, '장르3');
             INSERT INTO genre VALUES(NULL, '장르4');
         ''')
```

```
Out[17]: <sqlite3.Cursor at 0x11042e490>
```

```
In [18]: cur.execute('SELECT * FROM genre')
         cur.fetchall()
```

```
Out[18]: [(1, '장르1'), (2, '장르2'), (3, '장르3'), (4, '장르4')]
```

```
In [19]: cur.execute('SELECT * FROM artist')
         artist = cur.fetchall()
```

```
In [21]: cur.execute('SELECT * FROM artist WHERE name=?', ['가수1'])
         cur.fetchall()
```

```
Out[21]: [(1, '가수1')]
```

```
In [22]: cur.execute('SELECT * FROM artist WHERE name LIKE ?', ['%1'])
         cur.fetchall()
```

```
Out[22]: [(1, '가수1')]
```

```
In [25]: data = ['1', '2', '3', '4']

         for val in data:
             cur.execute('SELECT pk FROM artist WHERE name LIKE ?', ['%'+val])
             pk = cur.fetchall()
             if len(pk) > 0:
         #           cur.execute('INSERT INTO album(pk, name, fk) VALUES(?,?,?)')
         #           cur.execute('INSERT INTO album VALUES(NULL,?,?)')
         #           cur.execute('INSERT INTO album(name,fk) VALUES(?,?)')
```

```
        cur.execute('INSERT INTO album VALUES(NULL,?,?)',
                    ['앨범'+val, pk[0][0]])
```

In [26]:
```
cur.execute('SELECT * FROM album')
cur.fetchall()
```

Out[26]:
```
[(1, '앨범1', 1), (2, '앨범2', 2), (3, '앨범3', 3), (4, '앨범4', 4)]
```

In [27]:
```
data = [('싱글1', '%1'), ('싱글2', '%2'), ('싱글3', '%3'), ('싱글4', '%4')]

cur.executemany('''
    INSERT INTO album(name, fk) VALUES(?, (
        SELECT pk FROM artist WHERE name LIKE ?
    ))
''', data)
```

Out[27]:
```
<sqlite3.Cursor at 0x11042e490>
```

In [28]:
```
cur.execute('SELECT * FROM album')
cur.fetchall()
```

Out[28]:
```
[(1, '앨범1', 1),
 (2, '앨범2', 2),
 (3, '앨범3', 3),
 (4, '앨범4', 4),
 (5, '싱글1', 1),
 (6, '싱글2', 2),
 (7, '싱글3', 3),
 (8, '싱글4', 4)]
```

In [29]:
```
cur.execute('SELECT * FROM album')
FK1 = cur.fetchall()

cur.execute('SELECT * FROM genre')
FK2 = cur.fetchall()
```

In [31]:
```
for row in FK1:
    if row[1] == '싱글1':
        fk1 = row[0]
        break
```

In [33]:
```
for row in FK2:
    if row[1] == '장르3':
        fk2 = row[0]
        break
```

In [35]:
```
cur.execute('''
    INSERT INTO track(pk, name, length, rating, count, fk1, fk2)
    VALUES(NULL, ?, ?, ?, ?, ?, ?)
''', ['노래1', 270, 5, 100, fk1, fk2])
# map, filter
```

Out[35]:
```
<sqlite3.Cursor at 0x11042e490>
```

In [37]:
```
cur.execute('SELECT * FROM track')
cur.fetchall()
```

Out[37]:
```
[(1, '노래1', 270, 5, 100, 5, 3)]
```

In [38]:
```
cur.execute('''
    INSERT INTO track(name, fk1, fk2)
    VALUES(?, ?, ?)
''', ['노래2', fk1, fk2])
# map, filter
```

Out[38]:
```
<sqlite3.Cursor at 0x11042e490>
```

In [39]:
```
cur.execute('SELECT * FROM track')
cur.fetchall()
```

Out[39]:
```
[(1, '노래1', 270, 5, 100, 5, 3), (2, '노래2', 0, 0, 0, 5, 3)]
```

In [40]:
```
cur.execute('DELETE FROM track WHERE pk=2')
```

Out[40]:
```
<sqlite3.Cursor at 0x11042e490>
```

In [41]:
```
cur.execute('SELECT * FROM track')
cur.fetchall()
```

Out[41]:
```
[(1, '노래1', 270, 5, 100, 5, 3)]
```

In [42]:
```
cur.execute('INSERT INTO track(name, fk1, fk2) VALUES(?,?,?)',
            ['노래2', 1, 2])
```

Out[42]:
```
<sqlite3.Cursor at 0x11042e490>
```

In [43]:
```
cur.execute('SELECT * FROM track')
cur.fetchall()
```

Out[43]:
```
[(1, '노래1', 270, 5, 100, 5, 3), (2, '노래2', 0, 0, 0, 1, 2)]
```

In [44]:
```
cur.execute('''
    INSERT INTO track(name, fk1, fk2) VALUES(?,?,(
        SELECT pk FROM genre WHERE name LIKE ?
    ))
''', ['노래3', 2, '%4'])
```

Out[44]:
```
<sqlite3.Cursor at 0x11042e490>
```

In [45]:
```
cur.execute('SELECT * FROM track')
cur.fetchall()
```

Out[45]:
```
[(1, '노래1', 270, 5, 100, 5, 3),
 (2, '노래2', 0, 0, 0, 1, 2),
 (3, '노래3', 0, 0, 0, 2, 4)]
```

In [46]:
```
cur.execute('''
    INSERT INTO track(name, fk1, fk2) VALUES(?,(
        SELECT pk FROM album WHERE name LIKE ?
    ),(
        SELECT pk FROM genre WHERE name LIKE ?
    ))
''', ['노래4', '%싱글%1%', '%장르%3%'])
```

Out[46]:
```
<sqlite3.Cursor at 0x11042e490>
```

```
In [ ]: cur.execute('''
            INSERT INTO track(name, fk1, fk2)
            VALUES(?, ?, ?)
        ''', ['노래2', fk1, fk2])
        # map, filter
```

```
In [48]: data = [
             ['노래5', '%싱글%3%', '%장르%4%'],
             ['노래6', '%앨범%1%', '%장르%1%'],
             ['노래7', '%앨범%2%', '%장르%2%'],
             ['노래8', '%앨범%3%', '%장르%3%'],
             ['노래9', '%앨범%4%', '%장르%4%'],
             ['노래10', '%싱글%4%', '%장르%3%'],
             ['노래11', '%싱글%3%', '%장르%2%'],
             ['노래12', '%싱글%2%', '%장르%1%'],
         ]
         cur.executemany('''
             INSERT INTO track(name, fk1, fk2) VALUES(?,(
                 SELECT pk FROM album WHERE name LIKE ?
             ),(
                 SELECT pk FROM genre WHERE name LIKE ?
             ))
         ''', data)

Out[48]: <sqlite3.Cursor at 0x11042e490>
```

```
In [50]: cur.execute('SELECT COUNT(*) FROM track')
         cur.fetchall()

Out[50]: [(12,)]
```

```
In [52]: cur.execute('SELECT fk2, COUNT(*) FROM track GROUP BY fk2')
         cur.fetchall()

Out[52]: [(1, 2), (2, 3), (3, 4), (4, 3)]
```

```
In [53]: cur.execute('''
             SELECT T_B.pk, T_B.name, T_A.B FROM genre AS T_B
             INNER JOIN
             (SELECT fk2 AS A, COUNT(*) AS B FROM track GROUP BY fk2) AS T_A
             ON T_A.A = T_B.pk
         ''')
         cur.fetchall()

Out[53]: [(1, '장르1', 2), (2, '장르2', 3), (3, '장르3', 4), (4, '장르4', 3)]
```

```
In [57]: cur.execute('''
             SELECT T_B.pk, T_B.name, T_A.B FROM genre AS T_B
             INNER JOIN
             (SELECT fk2 AS A, COUNT(*) AS B FROM track GROUP BY fk2) AS T_A
             ON T_A.A = T_B.pk
             WHERE T_A.B > 2
             ORDER BY T_B.pk DESC
             LIMIT 0,2
         ''')
         cur.fetchall()

Out[57]: [(4, '장르4', 3), (3, '장르3', 4)]
```

In [58]:
```
cur.execute('''
    SELECT T_A.pk, T_A.name, T_B.CNT FROM artist AS T_A
    INNER JOIN
    (SELECT fk, COUNT(*) AS CNT FROM album GROUP BY fk) AS T_B
    ON T_B.fk = T_A.pk
''')
cur.fetchall()
```

Out[58]:
```
[(1, '가수1', 2), (2, '가수2', 2), (3, '가수3', 2), (4, '가수4', 2)]
```

In [60]:
```
cur.execute('''
    SELECT track.PK, track.name, genre.name FROM track
    INNER JOIN genre ON genre.pk = track.fk2
    ORDER BY genre.name ASC
''')
cur.fetchall()
```

Out[60]:
```
[(6, '노래6', '장르1'),
 (12, '노래12', '장르1'),
 (2, '노래2', '장르2'),
 (7, '노래7', '장르2'),
 (11, '노래11', '장르2'),
 (1, '노래1', '장르3'),
 (4, '노래4', '장르3'),
 (8, '노래8', '장르3'),
 (10, '노래10', '장르3'),
 (3, '노래3', '장르4'),
 (5, '노래5', '장르4'),
 (9, '노래9', '장르4')]
```

In [62]:
```
cur.execute('''
    SELECT track.PK, track.name, album.name, genre.name FROM track
    INNER JOIN genre ON genre.pk = track.fk2
    INNER JOIN album ON album.pk = track.fk1
    ORDER BY album.name, genre.name
''')
cur.fetchall()
```

Out[62]:
```
[(1, '노래1', '싱글1', '장르3'),
 (4, '노래4', '싱글1', '장르3'),
 (12, '노래12', '싱글2', '장르1'),
 (11, '노래11', '싱글3', '장르2'),
 (5, '노래5', '싱글3', '장르4'),
 (10, '노래10', '싱글4', '장르3'),
 (6, '노래6', '앨범1', '장르1'),
 (2, '노래2', '앨범1', '장르2'),
 (7, '노래7', '앨범2', '장르2'),
 (3, '노래3', '앨범2', '장르4'),
 (8, '노래8', '앨범3', '장르3'),
 (9, '노래9', '앨범4', '장르4')]
```

In [69]:
```
cur.execute('''
    SELECT track.PK, artist.name, track.name, album.name, genre.name
    FROM track
    INNER JOIN genre ON genre.pk = track.fk2
    INNER JOIN album ON album.pk = track.fk1
    INNER JOIN artist ON artist.pk = album.fk
    WHERE track.count < 10
    ORDER BY artist.name, album.name, genre.name
''')
```

```
#    WHERE artist.name LIKE '%1'
cur.fetchall()
```

Out[69]:
```
[(4, '가수1', '노래4', '싱글1', '장르3'),
 (6, '가수1', '노래6', '앨범1', '장르1'),
 (2, '가수1', '노래2', '앨범1', '장르2'),
 (12, '가수2', '노래12', '싱글2', '장르1'),
 (7, '가수2', '노래7', '앨범2', '장르2'),
 (3, '가수2', '노래3', '앨범2', '장르4'),
 (11, '가수3', '노래11', '싱글3', '장르2'),
 (5, '가수3', '노래5', '싱글3', '장르4'),
 (8, '가수3', '노래8', '앨범3', '장르3'),
 (10, '가수4', '노래10', '싱글4', '장르3'),
 (9, '가수4', '노래9', '앨범4', '장르4')]
```

In [ ]:
```
1. 새 게시물 생성
2. 생성 시, 사용자태그(0~N)
   3. 사용자태그가 해시태그 풀에 있는지 확인
   4. 새 게시물 - 해시태그 관계 만들어
   5. 해시태그 풀에 있는 빈도 정보 +1
```

In [ ]:
```
posting - pk, title, content, date

hashtag - pk, name, count

posting-hashtag : posting.pk, hashtag,pk
```

In [71]:
```
con = sqlite3.connect('sns.db')
cur = con.cursor()
```

In [73]:
```
cur.execute('SELECT CURRENT_TIMESTAMP')
cur.fetchall()
```

Out[73]:
```
[('2023-03-06 02:50:32',)]
```

In [75]:
```
cur.executescript('''
    DROP TABLE IF EXISTS posting;
    CREATE TABLE posting (
        pk INTEGER PRIMARY KEY,
        title TEXT,
        content TEXT,
        regdate DATE DEFAULT CURRENT_TIMESTAMP
    );
    DROP TABLE IF EXISTS hashtag;
    CREATE TABLE hashtag (
        pk INTEGER PRIMARY KEY,
        name TEXT,
        count INTEGER DEFAULT 0
    );
    DROP TABLE IF EXISTS poshas;
    CREATE TABLE poshas (
        fk1 INTEGER NOT NULL,
        fk2 INTEGER NOT NULL
    );
''')
```

Out[75]:
```
<sqlite3.Cursor at 0x110023810>
```

In [76]:
```
# def addPosting(title, content, *hashtag)
# 1. 새 게시물 생성
```

```
cur.execute('INSERT INTO posting(title, content) VALUES(?,?)',
            ['제목1', '내용1'])
pid = cur.lastrowid
```

In [84]:
```
hashtag = ['태그1', '태그2']
tagids = list()

# 2. 생성 시, 사용자태그(0~N)
#    3. 사용자태그가 해시태그 풀에 있는지 확인
for tag in hashtag:
    cur.execute('SELECT pk FROM hashtag WHERE name=?', [tag])
#     print(cur.fetchone()) # 없으면 None
    tid = cur.fetchone()
    if tid is not None:
        tagids.append(tid[0])
tagids
```

Out[84]:
```
[1, 2]
```

In [85]:
```
#    4. 새 게시물 - 해시태그 관계 만들어
#    5. 해시태그 풀에 있는 빈도 정보 +1
for tid in tagids:
    cur.execute('''
        INSERT INTO poshas VALUES(?,?)
    ''', [pid, tid])

    cur.execute('''
        UPDATE hashtag
        SET count = count + 1
        WHERE pk=?
    ''', [tid])
```

In [82]:
```
# 태그풀 생성
tags = [['태그1'], ['태그2'], ['태그3']]
cur.executemany('INSERT INTO hashtag(name) VALUES(?)', tags)
```

Out[82]:
```
<sqlite3.Cursor at 0x110023810>
```

In [86]:
```
con.commit()
```

In [90]:
```
cur.execute('''
    SELECT posting.title, posting.content, posting.regdate, hashtag.name
    FROM poshas
    INNER JOIN posting ON posting.pk = fk1
    INNER JOIN hashtag ON hashtag.pk = fk2
''')
cur.fetchall()
```

Out[90]:
```
[('제목1', '내용1', '2023-03-06 02:57:21', '태그1'),
 ('제목1', '내용1', '2023-03-06 02:57:21', '태그2')]
```

In [102…
```
cur.execute('''
    SELECT hashtag.name
    FROM poshas
    INNER JOIN posting ON posting.pk = fk1
    INNER JOIN hashtag ON hashtag.pk = fk2
    WHERE fk1 = ?
''', [pid])
cur.fetchall()
```

```
Out[102]: [('태그1',), ('태그2',)]
```

```
In [103…  # 수정
          # 제목1, 내용1, 태그1, 태그2
          # -> 제목1-1, 내용1, 태그1, 태그3
          cur.execute('UPDATE posting SET title=?, content=?',
                      ['제목1-1', '내용1'])

Out[103]: <sqlite3.Cursor at 0x110023810>
```

```
In [105…  cur.rowcount

Out[105]: 1
```

```
In [106…  hashtag = ['태그1', '태그3']
          oldtag = list()
          tagids = list()

          cur.execute('SELECT fk2 FROM poshas WHERE fk1=?', [pid])
          for row in cur.fetchall():
              oldtag.append(row[0])

          for tag in hashtag:
              cur.execute('SELECT pk FROM hashtag WHERE name=?', [tag])
          #     print(cur.fetchone()) # 없으면 None
              tid = cur.fetchone()
              if tid is not None:
                  tagids.append(tid[0])
          oldtag, tagids

Out[106]: ([1, 2], [1, 3])
```

```
In [108…  for tid in oldtag:
              if tid not in tagids: # 없으면
                  cur.execute('DELETE FROM poshas WHERE fk1=? AND fk2=?',
                              [pid, tid])

                  cur.execute('UPDATE hashtag SET count = count - 1 WHERE pk=?',
                              [tid])
```

```
In [109…  for tid in tagids:
              if tid not in oldtag: # 없으면
                  cur.execute('INSERT INTO poshas VALUES(?,?)',
                              [pid, tid])

                  cur.execute('UPDATE hashtag SET count = count + 1 WHERE pk=?',
                              [tid])
```

```
In [110…  cur.execute('SELECT * FROM poshas')
          cur.fetchall()

Out[110]: [(1, 1), (1, 3)]
```

```
In [111…  cur.execute('SELECT * FROM hashtag')
          cur.fetchall()

Out[111]: [(1, '태그1', 1), (2, '태그2', 0), (3, '태그3', 1)]
```

In [112…

```
con.close()
```