In [ ]:
```
Client(Python)       RDBMS(Sqlite)      DB
             (sqlite)
- Connection
- Cursor(작업) 1. execute*
              2. fetch*
              ------> (SQL)     <-------->
                         ?
             1. DDL(Create, Drop, Alter)
             2. DML(Insert, Select)(Delete/Update)
                Join(inner, left, right, cross)

- Insert (반드시 column 수와 value의 수가 일치해야 함) -> Transaction/Commit
  -> 1. 모든 column, 2. 특정 column, 3. column 생략
  -> execute*(params -> ?, {k(named):v(SQL 변수의 값)})
  ->                  iterable => column의 values
  -> executemany      iterable(iterable) => rows
  -> 중첩 (SQL - SELECT 괄호; group, sort, limit, ...)
- Select - Join
  -> 교집합, Left/Right, Cross(모든쌍 -> cost 큼)
- Begin transaction - SQLs - end
  try except(rollback)
```

In [1]:
```python
import sqlite3
```

In [2]:
```python
con = sqlite3.connect('playlist.db')
cur = con.cursor()
```

In [10]:
```python
# 4개의 테이블(가수, 앨범, 장르, 곡) <= ER(Entities)
# 가수:PK(integer),이름(text=>char/varchar),    앨범:PK,이름,가수FK
# 장르:PK,이름
# 곡:PK,이름,길이(time,int,real),rating(int/real),count(int),앨범FK,장르FK
cur.executescript('''
DROP TABLE IF EXISTS artist;
CREATE TABLE artist(
    pk   INTEGER PRIMARY KEY,
    name TEXT NOT NULL
);

DROP TABLE IF EXISTS album;
CREATE TABLE album(
    pk   INTEGER PRIMARY KEY,
    name TEXT NOT NULL,
    fk   INTEGER NOT NULL
);

DROP TABLE IF EXISTS genre;
CREATE TABLE genre(
    pk   INTEGER PRIMARY KEY,
    name TEXT NOT NULL
);

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

Out[10]:
```
<sqlite3.Cursor at 0x10a420260>
```

In [11]:
```
cur.execute('INSERT INTO track(fk1,fk2) VALUES(1,1)')
```

Out[11]:
```
<sqlite3.Cursor at 0x10a420260>
```

In [12]:
```
cur.execute('SELECT * FROM track')
cur.fetchall()
```

Out[12]:
```
[(1, '무제', 0, 0, 0, 1, 1)]
```

In [13]:
```
data = [(1, '가수A'), (None, '가수B')]
cur.executemany('INSERT INTO artist VALUES(?,?)', data)
```

Out[13]:
```
<sqlite3.Cursor at 0x10a420260>
```

In [17]:
```
cur.lastrowid
```

Out[17]:
```
2
```

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

Out[15]:
```
[(1, '가수A'), (2, '가수B')]
```

In [22]:
```
# 가수 입력 -> 있으면 PK, 없으면 insert 후 PK
def addArtist(name, flag=0):
    # con 확인
    v = None
    if flag == 0:
        v = name
    elif flag == 1:
        v = '%'+name
    elif flag == 2:
        v = name+'%'
    else:
        v = '%'+name+'%'

    cur.execute('SELECT PK FROM artist WHERE name LIKE ?', (v,))
    rst = cur.fetchone() # tuple(column...)

    if rst is None:
        cur.execute('INSERT INTO artist(name) VALUES(?)', (name,))
        rst = cur.lastrowid
    else:
        rst = rst[0] # column-pk

    return rst
```

In [29]:
```
# 0 - 정확히 일치, 1 - ~로 끝나는, 2 - ~로 시작하는, 3 - 중간에
addArtist('가수', 3)
```

```
Out[29]: 1
```

```
In [30]: cur.execute('SELECT * FROM artist')
         cur.fetchall()
```

```
Out[30]: [(1, '가수A'), (2, '가수B'), (3, '가수'), (4, 'A')]
```

```
In [31]: def modArtist(name, rename):
             pk = addArtist(name)

             cur.execute('UPDATE artist SET name=? WHERE pk=?', (rename, pk))

             return cur.rowcount
```

```
In [32]: modArtist('A', '가수D')
```

```
Out[32]: 1
```

```
In [33]: cur.execute('SELECT * FROM artist')
         cur.fetchall()
```

```
Out[33]: [(1, '가수A'), (2, '가수B'), (3, '가수'), (4, '가수D')]
```

```
In [34]: def delArtist(name):
             pk = addArtist(name)

             cur.execute('DELETE FROM artist WHERE pk=?', (pk,))

             return cur.rowcount
```

```
In [35]: delArtist('가수D')
```

```
Out[35]: 1
```

```
In [36]: cur.execute('SELECT * FROM artist')
         cur.fetchall()
```

```
Out[36]: [(1, '가수A'), (2, '가수B'), (3, '가수')]
```

```
In [37]: # 장르
         def addGenre(name):
             cur.execute('SELECT PK FROM genre WHERE name=?', (name,))
             rst = cur.fetchone()

             if rst is None:
                 cur.execute('INSERT INTO genre(name) VALUES(?)', (name,))
                 rst = cur.lastrowid
             else:
                 rst = rst[0]

             return rst

         def modGenre(name, rename):
             pk = addGenre(name)

             cur.execute('UPDATE genre SET name=? WHERE pk=?', (rename, pk))
```

```
    return cur.rowcount

def delGenre(name):
    pk = addGenre(name)

    cur.execute('DELETE FROM genre WHERE pk=?', (pk,))

    return cur.rowcount
```

In [39]:
```
for g in ['락', '발라드', '어쩌고']:
    addGenre(g)

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

Out[39]:
```
[(1, '락'), (2, '발라드'), (3, '어쩌고')]
```

In [57]:
```
def addRow(tablename, columns, values):
    if len(columns) != len(values):
        return False

    c = ', '.join(columns)
    q = ', '.join(['?' for v in values])

    where = list()
    for i in range(len(columns)):
        where.append('='.join([columns[i], '?'])) # [name=?, name=?]
    where = ' and '.join(where)

    cur.execute('SELECT PK FROM '+ tablename +' WHERE '+ where,
                values)
    rst = cur.fetchone()

    if rst is None:
        cur.execute('INSERT INTO '+ tablename + '('+ c +')'+\
                    'VALUES('+ q +')', values)
        rst = cur.lastrowid
    else:
        rst = rst[0]

    return rst
```

In [56]:
```
addRow('artist', ['name'], ['가수E'])
cur.execute('SELECT * FROM artist')
cur.fetchall()
```

Out[56]:
```
[(1, '가수A'), (2, '가수B'), (3, '가수'), (4, '가수D'), (5, '가수E')]
```

In [58]:
```
p1 = '가수A'
p2 = '앨범1'

pk = addRow('artist', ['name'], [p1])
addRow('album', ['name', 'fk'], [p2, pk])
```

Out[58]:
```
1
```

In [59]:
```
cur.execute('SELECT * FROM album')
cur.fetchall()
```

```
Out[59]: [(1, '앨범1', 1)]
```

```
In [60]: pk1 = addRow('artist', ['name'], [p1])
         pk2 = addRow('genre', ['name'], ['발라드'])
         addRow('track', ['name', 'fk1', 'fk2'], [p1, pk1, pk2])
```

```
Out[60]: 2
```

```
In [61]: cur.execute('SELECT * FROM track')
         cur.fetchall()
```

```
Out[61]: [(1, '무제', 0, 0, 0, 1, 1), (2, '가수A', 0, 0, 0, 1, 2)]
```

```
In [71]: # 수정 SET=> column=value, 삭제
         def modRow(tablename, columns, values, newvalues):
             if len(columns) != len(values):
                 return False
             pk = addRow(tablename, columns, values)

             where = list()
             for i in range(len(columns)):
                 where.append('='.join([columns[i], '?'])) # [name=?, name=?]
             where = ' and '.join(where)

             s = list()
             for i in range(len(columns)):
                 if newvalues[i]: # not None
                     s.append(columns[i]+'="'+newvalues[i]+'"')
             s = ', '.join(s)
             print('UPDATE '+tablename+' SET '+s+' WHERE '+where)
             cur.execute('UPDATE '+tablename+' SET '+s+' WHERE '+where,
                         (values))

             return cur.rowcount
```

```
In [69]: modRow('track', ['name', 'fk1', 'fk2'],
                ['가수A', 1, 2], ['노래1', None, None])
```

```
         UPDATE track SET name="노래1" WHERE name=? and fk1=? and fk2=?
Out[69]: 1
```

```
In [70]: cur.execute('SELECT * FROM track')
         cur.fetchall()
```

```
Out[70]: [(1, '무제', 0, 0, 0, 1, 1), (2, '노래1', 0, 0, 0, 1, 2)]
```

```
In [77]: # 가수 기준 앨범
         # 가수 1-* 앨범
         cur.execute('''
             SELECT name1, COUNT(name2)
             FROM
             (SELECT artist.name AS name1, album.name AS name2
             FROM artist
             LEFT JOIN album
             ON album.fk = artist.pk) AS A
             GROUP BY name1
         ''')
```

```
cur.fetchall()
```

```
Out[77]: [('가수', 0), ('가수A', 1), ('가수B', 0), ('가수D', 0), ('가수E', 0)]
```

```
In [80]: cur.execute('''
             SELECT DISTINCT(artist.name)
             FROM artist
             LEFT JOIN album
             ON album.fk = artist.pk
         ''')
         cur.fetchall()
```

```
Out[80]: [('가수A',), ('가수B',), ('가수',), ('가수D',), ('가수E',)]
```

```
In [81]: cur.execute('''
             SELECT D.name, B.name, C.name, A.name
             FROM track as A
             INNER JOIN album AS B
             ON A.fk1 = B.pk
             INNER JOIN genre AS C
             ON A.fk2 = C.pk
             INNER JOIN artist AS D
             ON D.pk = B.fk
         ''')
         cur.fetchall()
```

```
Out[81]: [('가수A', '앨범1', '락', '무제'), ('가수A', '앨범1', '발라드', '노래1')]
```

1. 장르별 노래 목록 SELECT genre.pk, genre.name, track.pk, track.name FROM track INNER JOIN genre ON track.fk2 = genre.pk ORDER BY genre.pk, track.pk ASC

2. 장르별 노래 갯수 SELECT genre.pk, genre.name, COUNT(track.pk) FROM track INNER JOIN genre ON track.fk2 = genre.pk GROUP BY genre.pk ORDER BY genre.name, track.pk ASC

3. 앨범별 노래 목록 SELECT album.pk, album.name, track.pk, track.name FROM track INNER JOIN album ON track.fk1 = album.pk ORDER BY album.fk, album.pk, track.pk ASC

4. 앨범별 노래 갯수

5. 전체 플레이리스트에서 count가 > 5 초과이면서, 내림차순

-> 자주 듣는 목록 SELECT artist.name, album.name, genre.name, A.name, A.length, A.rating, A.count FROM track AS A INNER JOIN genre ON genre.pk = A.fk2 INNER JOIN album ON album.pk = A.fk1 INNER JOIN artist ON artist.pk = album.fk WHERE A.count > 5 ORDER BY A.count DESC

6. 장르가 댄스 일 때, 노래가 짧은것 순서로 SELECT B.name, A.name FROM track AS A INNER JOIN genre AS B ON B.pk = A.fk2 AND B.name='댄스' ORDER BY A.length ASC

   SELECT B.name, A.name FROM track AS A INNER JOIN genre AS B ON B.pk = A.fk2 WHERE B.name = '댄스' ORDER BY A.length ASC

   SELECT B.name, A.name FROM track AS A INNER JOIN genre AS B ON B.pk = A.fk2 WHERE B.pk = (SELECT pk FROM genre WHERE name='댄스') ORDER BY A.length ASC