資料庫技術解析與應用: Jam

林君華，蔡桂毓，陳品妤，連英茹，張力文，林于婷，莊又齊 (2022)

國立政治大學資訊科學系

Author Note

資料庫系統 (Database System) 期末專案，授課老師: 沈錳坤

僅作為學術用途

目錄

目的與背景

圖 1. Jam

一、題目以及說明: 音樂查詢

現今的歌曲搜尋通常都是藉由演唱者來做查詢，很少針對用製作人或作詞作曲的人來搜尋歌曲，但是他們也是歌曲成功的因素之一，他們的其他歌曲可能也同樣地令人驚艷或是符合使用者的風格，我們的資料庫將整合原曲的 Meta Data，讓使用者也能憑藉幕後製作團隊去搜尋更多音樂，此外，還針對使用者(會員)和管理者提供不同的服務。

二、資料需求

[原歌曲資料] 歌曲名、作曲、歌手、唱片製作人、歌曲風格、發行時間、國家

[歌手資料] 姓名、性別、出生日期(年齡)、國籍

[唱片製作人資料] 姓名、性別、出生日期(年齡)、國籍

[作曲者資料] 姓名、性別、出生日期(年齡)、國籍

[會員資料] 姓名、性別、出生日期(年齡)、國籍

三、系統功能分析

1. 可以以製作人員來搜尋作品及進行作品間的連結

2. 會員: 註冊會員、查詢歌曲、訂閱

3. 後台管理者: 修改會員資料、刪除會員帳號、新增作品

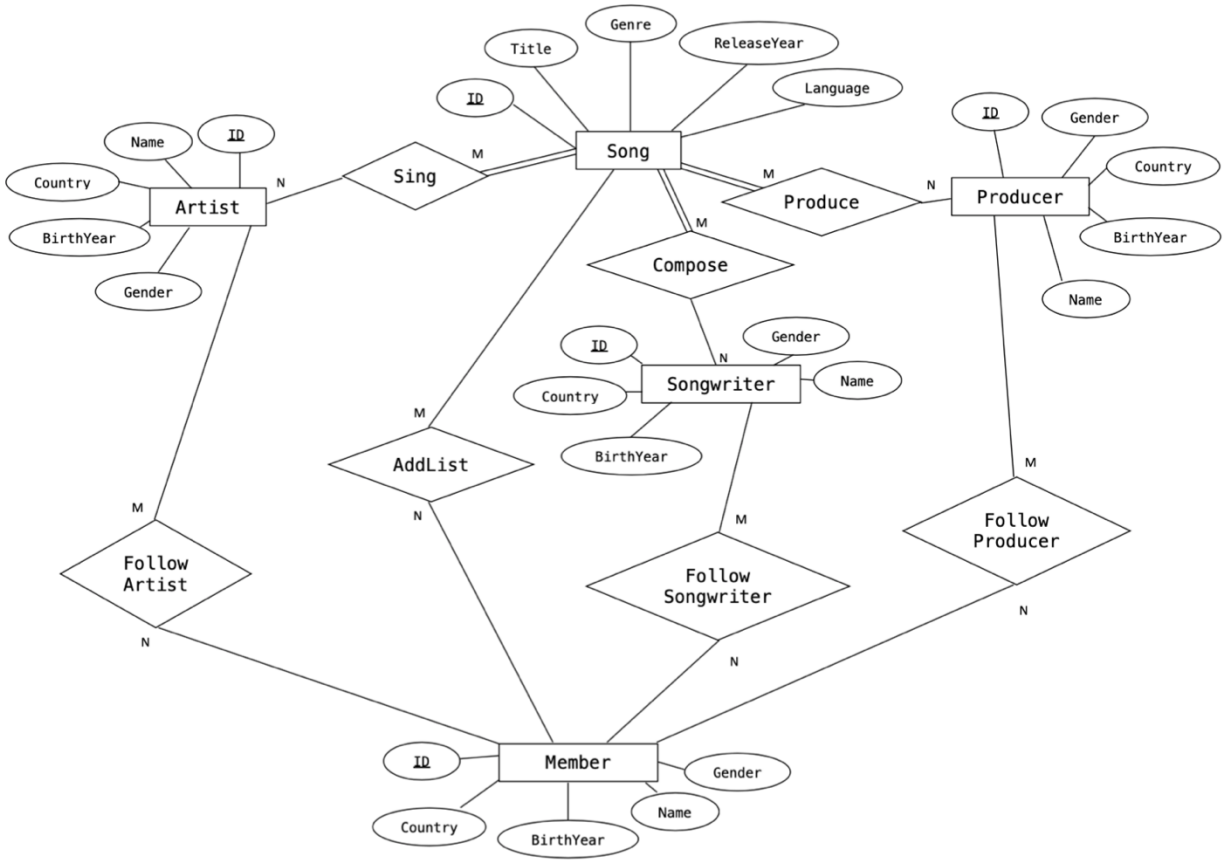
四、使用對象

1. 會員

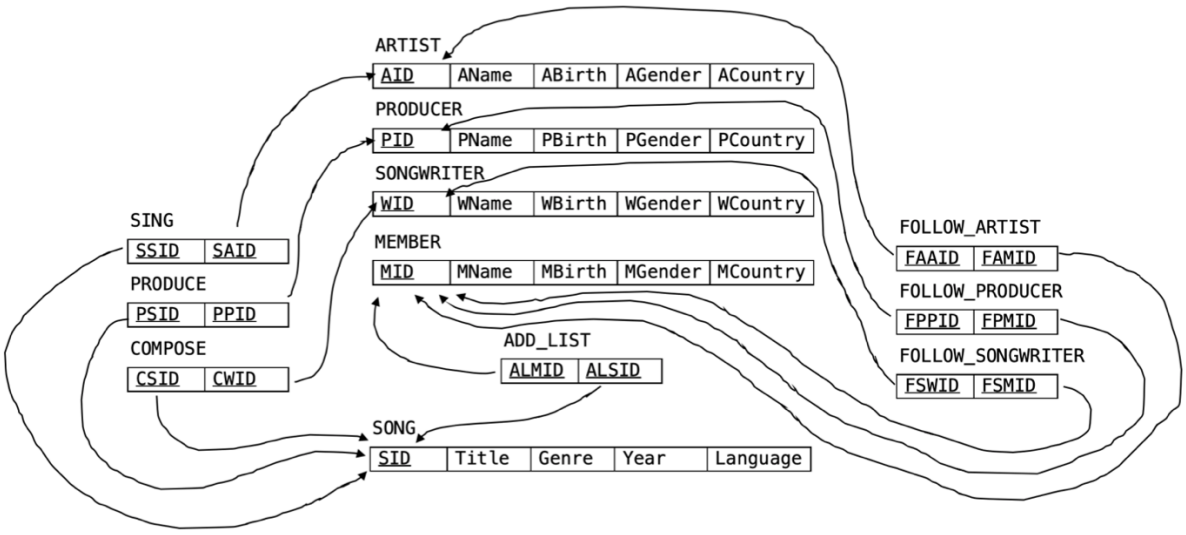
2. 後台管理者

架構解析

1. ER Model



2. Relational Model

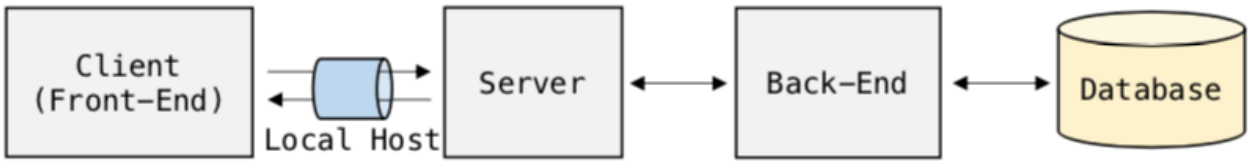


3. System Structure

Programming Language: Python3, SQL, JavaScript

DBMS: sqlite3

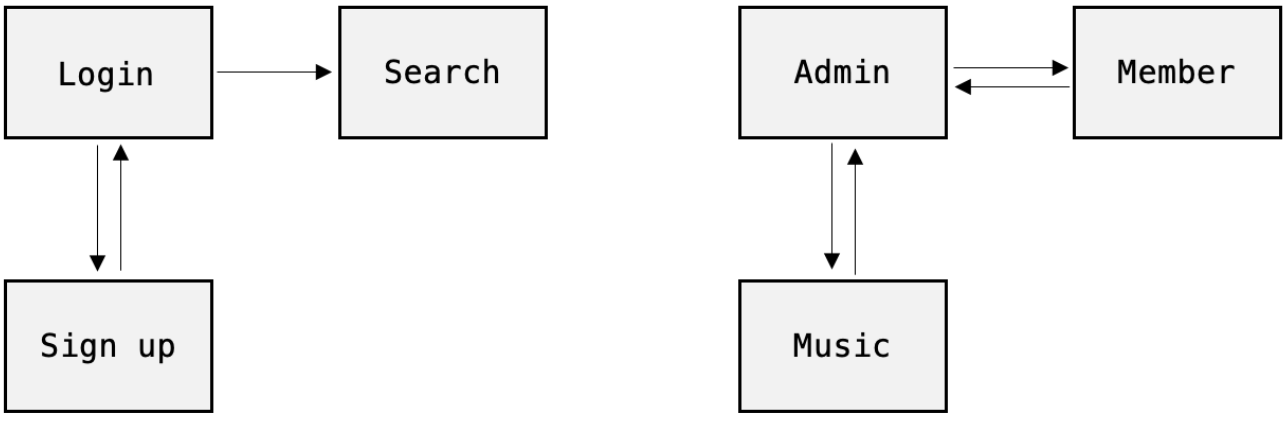
Framework: Flask



- Sign in
- Sign up
- Edit profile
- Follow
- Search
- Browse

- Create
- Insert
  - Sign up
  - Follow
  - Songs
- Read
  - Sign in
  - Search
  - Browse
- Update
  - Edit profile
- Delete
  - Account

4. UI Flow



技術實作 — CRUD

Create:

1. schema.sql

```sql
CREATE TABLE song (
  sid VARCHAR PRIMARY KEY,
  title VARCHAR,
  genre VARCHAR,
  year INTEGER,
  language VARCHAR
);

CREATE TABLE artist (
  aid VARCHAR PRIMARY KEY,
  aname VARCHAR,
  abirth INTEGER,
  agender INTEGER,
  acountry VARCHAR
);

CREATE TABLE producer (
  pid VARCHAR PRIMARY KEY,
  pname VARCHAR,
  pbirth INTEGER,
  pgender INTEGER,
  pcountry VARCHAR
);

CREATE TABLE songwriter (
  wid VARCHAR PRIMARY KEY,
  wname VARCHAR,
  wbirth INTEGER,
  wgender INTEGER,
  wcountry VARCHAR
);

CREATE TABLE member (
  mid VARCHAR PRIMARY KEY,
  mname VARCHAR,
  mbirth INTEGER,
  mgender INTEGER,
  mcountry VARCHAR
);
```

```sql
CREATE TABLE follow_artist (
  faaid VARCHAR REFERENCES artist (aid),
  famid VARCHAR REFERENCES member (mid) ON DELETE CASCADE
);

CREATE TABLE follow_producer (
  fppid VARCHAR REFERENCES producer (pid),
  fpmid VARCHAR REFERENCES member (mid) ON DELETE CASCADE
);

CREATE TABLE follow_songwriter (
  fswid VARCHAR REFERENCES songwriter (wid),
  fsmid VARCHAR REFERENCES member (mid) ON DELETE CASCADE
);

CREATE TABLE sing (
  ssid VARCHAR REFERENCES song (sid),
  said VARCHAR REFERENCES artist (aid)
);

CREATE TABLE produce (
  psid VARCHAR REFERENCES song (sid),
  ppid VARCHAR REFERENCES producer (pid)
);

CREATE TABLE compose (
  csid VARCHAR REFERENCES song (sid),
  cwid VARCHAR REFERENCES songwriter (wid)
);

CREATE TABLE add_list (
  almid VARCHAR REFERENCES member (mid) ON DELETE CASCADE,
  alsid VARCHAR REFERENCES song (sid)
);
```

2. init_db.py: 根據 schema.sql 創建 Database

```python
import sqlite3

connection = sqlite3.connect('jam.db')

with open('schema.sql') as f:
    connection.executescript(f.read())

cur = connection.cursor()

connection.commit()
connection.close()
```

Read:

app.py

```
@app.route('/searchSong/<string:title>', methods=['GET'])
def searchSong(title):
    conn = get_db_connection()
    db = conn.cursor()
    songs = db.execute(
        'Select song.title,song.genre,song.year,song.language from song where song.title= ?', [title]).fetchall()
    conn.commit()
    conn.close()
    res = app.response_class(response=json.dumps(
        [dict(s) for s in songs]), status=200, mimetype='application/json')
    return res
```

Update:

app.py

```
@app.route('/updateUser/<string:mid>/<string:mname>', methods=['GET'])
def UpdateMember(mid, mname):
    conn = get_db_connection()
    db = conn.cursor()
    db.execute(
        'UPDATE member SET mname=? WHERE mid = ?', (mname, mid))
    conn.commit()
    conn.close()
    return "success update"
```

Delete:

app.py

```
@app.route('/delete/<string:mid>', methods=['GET'])
def delete_account(mid):
    conn = get_db_connection()
    db = conn.cursor()
    result = db.execute(
        'Select member.mid from member where member.mid = ?', [mid]).fetchall()
    conn.commit()
    print(result)
    print(type(result))
    if not result:
        conn.close()
        print("the account is not exist.")
        return "the account is not exist."
    else:
        db.execute('Delete from member where mid = ?', [mid])
        conn.commit()
        conn.close()
        print("successful delete.")
        return "successful delete."
```

Insert:

app.py

```
@app.route('/register/<string:mid>/<string:mname>/<string:mbirth>/<string:mgender>/<string:mcountry>', methods=['GET'])
def register(mid, mname, mbirth, mgender, mcountry):
    conn = get_db_connection()
    db = conn.cursor()
    uni = db.execute('SELECT * FROM member WHERE mid = ?', [mid]).fetchall()
    conn.commit()
    if not uni:
        db.execute('INSERT INTO member (mid,mname,mbirth,mgender,mcountry) VALUES (?,?,?,?,?)',
                   (mid, mname, mbirth, mgender, mcountry))
        conn.commit()
        conn.close()
        print("register success")
        return "register success"
    else:
        conn.close()
        print("Already registered")
        return "ID taken!"
```

For full code please visit our github.

組員分工

林君華: ER Model、系統架構、schema.sql、init_db.py、註冊、後台加歌、專題報告 (18%)

林于婷: Relational Model、後端搜尋功能、後端清單查詢功能、後端統整、後端技術支援、前後端整合 (18%)

陳品妤: 前端登入、前端註冊、前端整合、前端技術支援、ER Model 討論 (18%)

莊又齊: 訂閱、更新資料 (11.5%)

蔡桂毓: 網頁後台前端設計、歌曲及會員資料修改 (11.5%)

連英茹: 會員登入、刪除帳號 、Demo(11.5%)

張力文: 網頁前端查詢歌曲、作曲人、製作人及訂閱四種清單 (11.5%)