

# DBProject\_F74046284

資訊 108 黃柏瑄

## Environment

- Ubuntu 18.04
- Gtk 3.20
- SQLite
- Rust 1.35.0

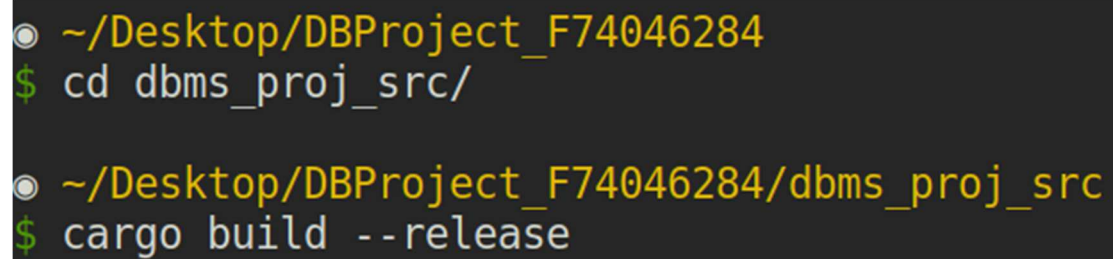
## Architecture design

```
DBProject_F74046284/dbms_proj_src/
├─ Cargo.toml <----- Program configuration.
├─ README.md
└─ src/
    ├─ db/ <----- The code related with database.
    │   └─ sqlite/
    │       ├─ init.rs <-- Initialize the data to database.
    │       └─ mod.rs <-- Execute the command and return result.
    ├─ gui/ <----- The code related with GUI
    │   └─ gtk3/
    │       ├─ app.ui <--- The GUI generated by using Glade.
    │       └─ mod.rs <--- Connect components with the behaviors.
    └─ main.rs <----- The entry point of the program.
```

# Usage

## Build the executable and run

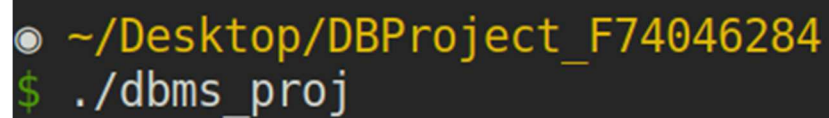
```
$ cd dbms_proj_src/  
$ cargo build --release  
$ ./target/release/dbms_proj
```

A terminal window with a dark background and yellow text. The prompt is ~/Desktop/DBProject\_F74046284. The user enters 'cd dbms\_proj\_src/' and the prompt changes to ~/Desktop/DBProject\_F74046284/dbms\_proj\_src. The user then enters 'cargo build --release'.

```
● ~/Desktop/DBProject_F74046284  
$ cd dbms_proj_src/  
● ~/Desktop/DBProject_F74046284/dbms_proj_src  
$ cargo build --release
```

## Run the program from the pre-build executable

```
$ ./dbms_proj
```

A terminal window with a dark background and yellow text. The prompt is ~/Desktop/DBProject\_F74046284. The user enters './dbms\_proj'.


```
● ~/Desktop/DBProject_F74046284  
$ ./dbms_proj
```

Note.

When you start the program, it will create “resources” for placing the database named “mydb.sqlite”.

# The execution flows and screenshots

## Initial frame:

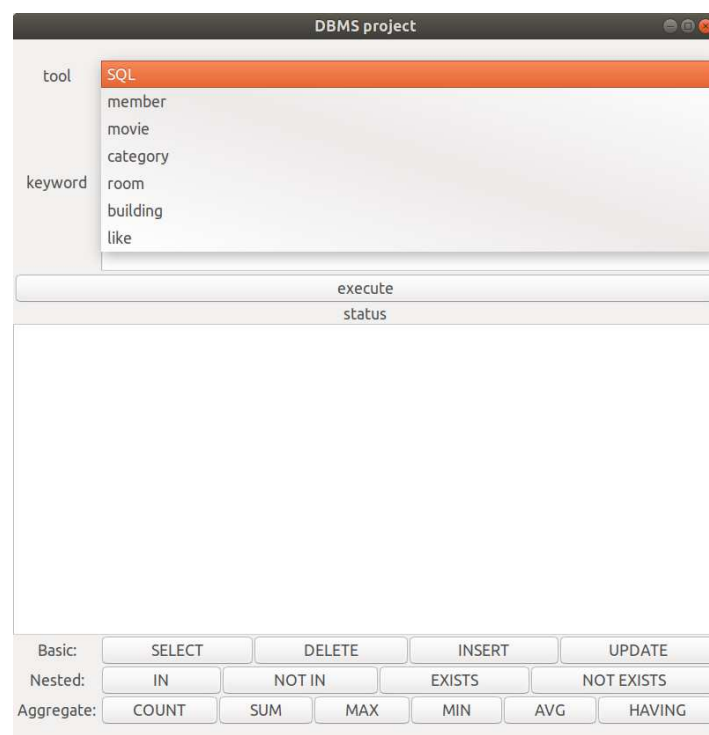


The screenshot shows the initial frame of the DBMS project interface. It includes a 'tool' dropdown menu, a 'keyword' text area containing the SQL query 'SELECT MemberId, Name, Gender FROM member', an 'execute' button, a 'status' label, a large empty 'result view' area, and a bottom panel with buttons for SQL commands categorized by Basic, Nested, and Aggregate.

Annotations on the right side of the screenshot:

- tool options combo box
- keyword or commands
- button for executing the action
- result view
- buttons for changing commands

## Tool options:



The screenshot shows the 'tool' dropdown menu open, displaying a list of options: SQL, member, movie, category, room, building, and like. The 'keyword' text area is empty. The 'execute' button and 'status' label are visible below the dropdown. The bottom panel with SQL command buttons remains the same.

SQL:

DBMS project

tool

SQL

SELECT MemberId, Name, Gender FROM member

keyword

execute

success

MemberId	Name	Gender
1	Michel	M
2	Sara	F
3	Liam	M
4	Zelda	F
5	Neo	M
6	Octopus	M
7	Ben	M
8	OuO	?
9	XiongJJ	M
10	Hello World	F

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

SQL SELECT:

DBMS project

tool

SQL

SELECT  
MemberId AS ID, member.Name, Gender,  
Title AS Movie, room.Name AS Room,  
building.Name AS Building  
FROM member  
JOIN movie USING (MovieId)  
JOIN room USING (RoomId)  
JOIN building USING (BuildingId)  
ORDER BY MemberId

keyword

execute

success

ID	Name	Gender	Movie	Room	Building
1	Michel	M	Avengers: Infinity War	D1	Ping An Finance Center
2	Sara	F	Aquamanara	B1	Burj Khalifa
3	Liam	M	Solo: A Star Wars Story	C1	Makkah Royal Clock To
4	Zelda	F	Aquamanara	B1	Burj Khalifa
5	Neo	M	Avengers: Infinity War	D1	Ping An Finance Center
6	Octopus	M	Aquamanara	B0	Shanghai Tower
7	Ben	M	Fantastic Beasts: The Crimes of Grindelwald	A2	Burj Khalifa
8	OuO	?	Black Panther	D2	Burj Khalifa
9	XiongJJ	M	Ant-Man and the Wasp	A1	Shanghai Tower
10	Hello World	F	Fantastic Beasts: The Crimes of Grindelwald	A0	Burj Khalifa

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

SQL DELETE:

DBMS project

toolSQL

keywordDELETE FROM member WHERE MemberId = 1

execute

success

Basic:SELECTDELETEINSERTUPDATE

Nested:INNOT INEXISTSNOT EXISTS

Aggregate:COUNTSUMMAXMINAVGHAVING

DBMS project

toolSQL

keywordSELECT  
MemberId AS ID, member.Name, Gender,  
Title AS Movie, room.Name AS Room,  
building.Name AS Building  
FROM member  
JOIN movie USING (MovieId)  
JOIN room USING (RoomId)  
JOIN building USING (BuildingId)  
ORDER BY MemberId

execute

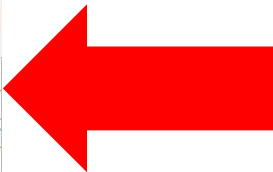
success

ID	Name	Gender	Movie	Room	Building
2	Sara	F	Aquamanara	B1	Burj Khalifa
3	Liam	M	Solo: A Star Wars Story	C1	Makkah Royal Clock To
4	Zelda	F	Aquamanara	B1	Burj Khalifa
5	Neo	M	Avengers: Infinity War	D1	Ping An Finance Center
6	Octopus	M	Aquamanara	B0	Shanghai Tower
7	Ben	M	Fantastic Beasts: The Crimes of Grindelwald	A2	Burj Khalifa
8	OuO	?	Black Panther	D2	Burj Khalifa
9	XiongJJ	M	Ant-Man and the Wasp	A1	Shanghai Tower
10	Hello World	F	Fantastic Beasts: The Crimes of Grindelwald	A0	Burj Khalifa

Basic:SELECTDELETEINSERTUPDATE

Nested:INNOT INEXISTSNOT EXISTS

Aggregate:COUNTSUMMAXMINAVGHAVING



SQL INSERT:

DBMS project

tool

SQL

keyword

INSERT INTO member  
(name, gender, phone, movieid, roomid) VALUES  
("哈哈", "F", "8767654637", "4", "5")

execute

success

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

DBMS project

tool

SQL

keyword

SELECT  
MemberId AS ID, member.Name, Gender,  
Title AS Movie, room.Name AS Room,  
building.Name AS Building  
FROM member  
JOIN movie USING (MovieId)  
JOIN room USING (RoomId)  
JOIN building USING (BuildingId)  
ORDER BY MemberId

execute

success

ID	Name	Gender	Movie	Room	Building
2	Sara	F	Aquamanara	B1	Burj Khalifa
3	Liam	M	Solo: A Star Wars Story	C1	Makkah Royal Clock To
4	Zelda	F	Aquamanara	B1	Burj Khalifa
5	Neo	M	Avengers: Infinity War	D1	Ping An Finance Center
6	Octopus	M	Aquamanara	B0	Shanghai Tower
7	Ben	M	Fantastic Beasts: The Crimes of Grindelwald	A2	Burj Khalifa
8	OuO	?	Black Panther	D2	Burj Khalifa
9	XiongJJ	M	Ant-Man and the Wasp	A1	Shanghai Tower
10	Hello World	F	Fantastic Beasts: The Crimes of Grindelwald	A0	Burj Khalifa
11	哈哈	F	Fantastic Beasts: The Crimes of Grindelwald	B1	Burj Khalifa

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

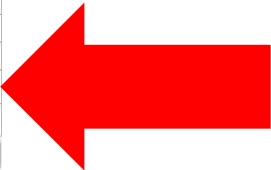
SUM

MAX

MIN

AVG

HAVING



SQL UPDATE:

DBMS project

tool

SQL

keyword

UPDATE member SET Gender = "X" WHERE Name = "OuO"

execute

success

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

DBMS project

tool

SQL

keyword

SELECT  
MemberId AS ID, member.Name, Gender,  
Title AS Movie, room.Name AS Room,  
building.Name AS Building  
FROM member  
JOIN movie USING (MovieId)  
JOIN room USING (RoomId)  
JOIN building USING (BuildingId)  
ORDER BY MemberId

execute

success

ID	Name	Gender	Movie	Room	Building
2	Sara	F	Aquamanara	B1	Burj Khalifa
3	Liam	M	Solo: A Star Wars Story	C1	Makkah Royal Clock To
4	Zelda	F	Aquamanara	B1	Burj Khalifa
5	Neo	M	Avengers: Infinity War	D1	Ping An Finance Center
6	Octopus	M	Aquamanara	B0	Shanghai Tower
7	Ben	M	Fantastic Beasts: The Crimes of Grindelwald	A2	Burj Khalifa
8	OuO	X	Black Panther	D2	Burj Khalifa
9	XiongJJ	M	Ant-Man and the Wasp	A1	Shanghai Tower
10	Hello World	F	Fantastic Beasts: The Crimes of Grindelwald	A0	Burj Khalifa
11	哈哈	F	Fantastic Beasts: The Crimes of Grindelwald	B1	Burj Khalifa

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

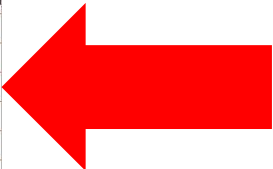
SUM

MAX

MIN

AVG

HAVING



SQL IN:

DBMS project

tool

SQL

keyword

SELECT MemberId, Name FROM member WHERE MovieId IN  
(SELECT MovieId FROM movie WHERE Title = "Aquamanara")

execute

success

MemberId	Name
2	Sara
4	Zelda
6	Octopus

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

SQL NOT IN:

DBMS project

tool

SQL

keyword

SELECT MemberId, Name FROM member WHERE MovieId NOT IN  
(SELECT MovieId FROM movie WHERE Title = "Aquamanara")

execute

success

MemberId	Name
3	Liam
5	Neo
7	Ben
8	OuO
9	XiongJJ
10	Hello World
11	哈哈哈

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING



## SQL EXISTS:

DBMS project

tool

SQL

SELECT CategoryId, Name FROM category  
WHERE EXISTS  
(SELECT \* FROM movie WHERE movie.CategoryId = category.CategoryId)

keyword

execute

success

MemberId	Name
3	Liam
5	Neo
7	Ben
8	OuO
9	XiongJJ
10	Hello World
11	哈哈哈

Basic: SELECT DELETE INSERT UPDATE

Nested: IN NOT IN EXISTS NOT EXISTS

Aggregate: COUNT SUM MAX MIN AVG HAVING

## SQL NOT EXISTS:

DBMS project

tool

SQL

SELECT CategoryId, Name FROM category  
WHERE NOT EXISTS  
(SELECT \* FROM movie WHERE movie.CategoryId = category.CategoryId)

keyword

execute

success

CategoryId	Name
6	Crime
7	Thriller
8	Mystery
9	Horror

Basic: SELECT DELETE INSERT UPDATE

Nested: IN NOT IN EXISTS NOT EXISTS

Aggregate: COUNT SUM MAX MIN AVG HAVING

## SQL COUNT:

DBMS project

tool

SQL

SELECT COUNT(MemberId) FROM member WHERE RoomId = "5"

keyword

execute

success

COUNT(MemberId)

3

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

## SQL SUM:

DBMS project

tool

SQL

SELECT SUM(Seats) FROM room WHERE buildingId = "1"

keyword

execute

success

SUM(Seats)

1100

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

# SQL MAX:

DBMS project

tool

SQL

SELECT MAX(Seats) FROM room

keyword

execute

success

MAX(Seats)

400

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

# SQL MIN:

DBMS project

tool

SQL

SELECT MIN(Seats) FROM room

keyword

execute

success

MIN(Seats)

10

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

# SQL AVG:

DBMS project

tool

SQL

SELECT AVG(Seats) FROM room

keyword

execute

success

AVG(Seats)

248.000000

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

# SQL HAVING:

DBMS project

tool

SQL

SELECT Name, Seats FROM room GROUP BY Name HAVING Seats > 300

keyword

execute

success

Name	Seats
A1	400
A2	400
B0	350
D1	400

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

MIN

AVG

HAVING

# Table SELECT:

DBMS project

tool

member

MemberId > 4 AND Gender = "M"

keyword

execute

success

MemberId	Name	Gender	Phone	MovieId	RoomId
5	Neo	M	8208092830	1	7
6	Octopus	M	1297180287	2	4
7	Ben	M	2628761980	4	3
9	XiongJJ	M	1652765256	9	2

Basic:

SELECT

DELETE

INSERT

UPDATE

Nested:

IN

NOT IN

EXISTS

NOT EXISTS

Aggregate:

COUNT

SUM

MAX

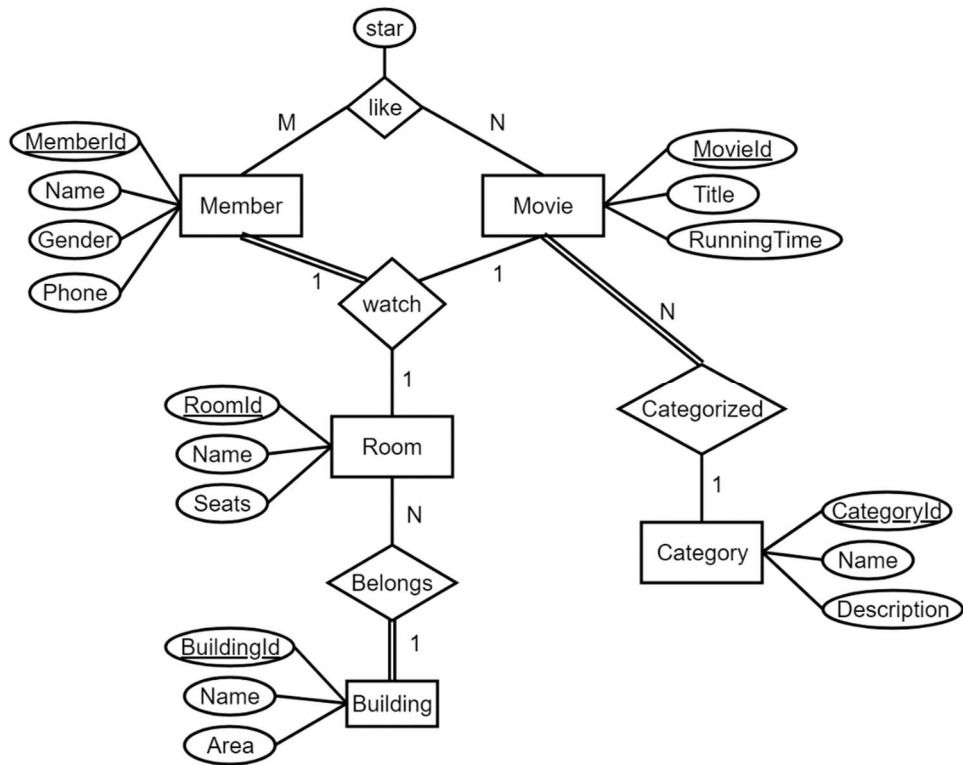
MIN

AVG

HAVING

# Design of database

ER diagram:



Relation Schema:

Member

<u>MemberId</u>	Name	Gender	Phone
-----------------	------	--------	-------

Movie

<u>MovieId</u>	Title	RunningTime
----------------	-------	-------------

Category

<u>CategoryId</u>	Name	Description
-------------------	------	-------------

Room

<u>RoomId</u>	Name	Seats
---------------	------	-------

Building

<u>BuildingId</u>	Name	Area
-------------------	------	------

Like

<u>LikeId</u>	MemberId	MovieId	Star
---------------	----------	---------	------

MMR

<u>MemberId</u>	MovieId	RoomId
-----------------	---------	--------

MC

<u>MovieId</u>	CategoryId
----------------	------------

RB

<u>RoomId</u>	BuildingId
---------------	------------

## Descriptions:

### Member:

- Record the data of members of a movie theater.
- A member has Name, Gender, Phone data.
- All members in the database have a movie to watch in a specific room today.

### Movie:

- Record the data of movies' information like Title, Running time.
- Each movie has a category.

### Category:

- Record the Names and Descriptions of all categories.

### Room:

- A room has a Name and the number of the seats in it.
- Each room is in a building.

### Building:

- A building has a Name and the Area that it is located.

### Like:

- A member can like zero or more movies which has a score Star is record in this table.