Introduction

The aim of this project is to provide a database system to store information about tourists and cruise ships. Different information is stored for different tours and different passengers, in order to facilitate the management and service of the company.

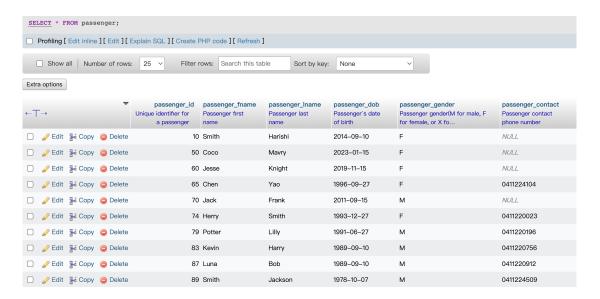
The background of this project is that the cruise line has more and more lines, cruise ships and passengers to manage more efficiently. Based on this DMBS, we can add, delete and modify cruise information at any time, including cruise facilities, itineraries, departure dates and sailing times. Passengers can choose different rooms, different itineraries and cancel their travel plans at any time.

Query Demonstration

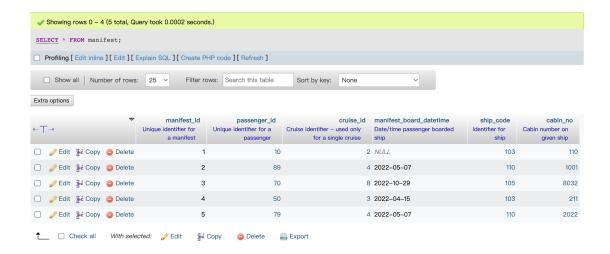
1. Join query:

In this project I have passenger and manifest two tables, when crew what to know passenger's cabin number they can join these two tables to check.

Passenger table:



Manifest table:



After join:



By joining the two tables, we can clearly see the corresponding cabin number and ship number of each passenger.

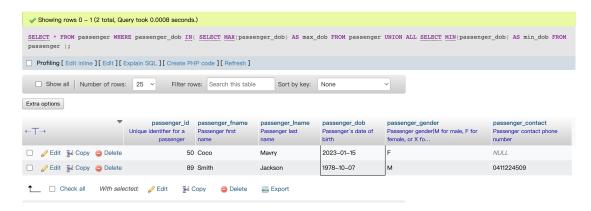
SQL code:

*
FROM
passenger

JOIN manifest ON passenger.passenger_id = manifest.passenger_id;

2. Aggregation query (functions such as min, max, average or count)

The crew wanted to find the oldest and youngest passengers to determine if any special items were needed for the voyage.



SQL code:

SELECT

*

```
FROM

passenger

WHERE

passenger_dob IN(

SELECT

MAX(passenger_dob) AS max_dob

FROM

passenger

UNION ALL

SELECT

MIN(passenger_dob) AS min_dob

FROM

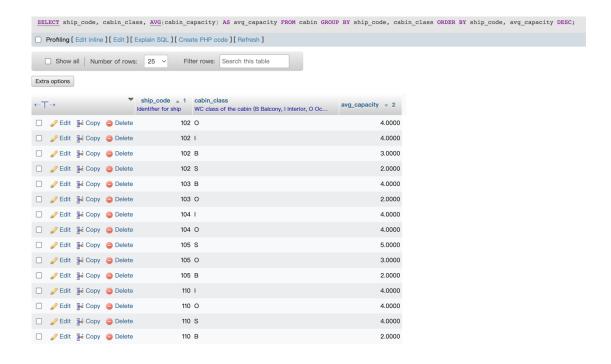
passenger

passenger

);
```

3. Aggregation with group-by (aggregated value for each group)

This query can be used to check when crews wants to see the number of occupants in the rooms of different ships, sorted by the number of occupants from most to least.



SQL code:

```
SELECT
ship_code,
cabin_class,
AVG(cabin_capacity) AS avg_capacity
FROM
cabin
GROUP BY
ship_code,
cabin_class
ORDER BY
ship_code,
avg_capacity DESC;
```

4. Delete operation with Cascade

This system can delete a passenger's reservation when a passenger does not want to continue the trip.



SQL code:

DELETE FROM

```
manifest
WHERE
manifest_id = 1;
```

5. Update operation

This statement is required when a passenger's cell phone number is changed

This is original data.



SQL code:

```
UPDATE

passenger

SET

passenger_contact = '0469987230'

WHERE

passenger_id = 65;
```

Conclusion

For this project, it took more time to think about the whole design at first, maybe one or two weeks, and then the realistic structure and code of each part took about four or five weeks, and after finishing these, it took about two days to write the report. Overall, it was a challenging project for beginners.

When I design the database system, I think this was the most interesting part of this project because I can actually do something on it. The part of generating an idea was relatively boring, because it was not possible to make a reasonable model very well by a beginner.

The lecture materials and the real cases studies I googled were very helpful when I was implementing this project and allowed me to get some ideas from them to complete my project.

When a beginner wants to design a database, they must first understand the relationship of each entity, and the domain of each entity's attributes, and must be careful when determining the relationship of each entity, to make sure that the FK and PK must be correct. Only then can you achieve the effect you want when building the database system.