CS307 Principles of Database Systems

PROJECT 2 REPORT

Database for Management of SUSTC

Student name	Student ID
Hu Qiang	12111214
Peng Zishen	12110323

Semester: 2022 Autumn
Lab Session: Monday 5-6 (Class 1)
Teacher: Chen Ran
Student Assistants: Wang Lishuang, Li Boyan

Contents:

PROJECT 2 REPORT

1 **Plan and Contributions**

2 **Task 1: Database Design**

2.1 E-R Diagram

2.2 Database Diagram and Clarification

2.3 database user creation and permission granting descriptions

3 **Task 2: Basic API Specification**

3.1 Table Manipulation

3.2 SUSTC Department Manager User

3.3 Courier User

3.4 Company Manager User

3.5 Seaport Officer User

4 **Advanced APIs and Other Requirements**

4.1 Other API

4.2 FrontEnd

4.2.1 Log In

4.2.2 basic functions with permission control

4.2.3 Log Out

1 Plan and Contributions

Table 1: Plan

Time	Task	
Week14	1.Design E-R diagram	2.Write SQL æle
	3.Script to import data	4.All the APIs
Week15	1.Optimization	2.FrontEnd
	3.Report completed	4.Typesetting

Table 2: Contributions

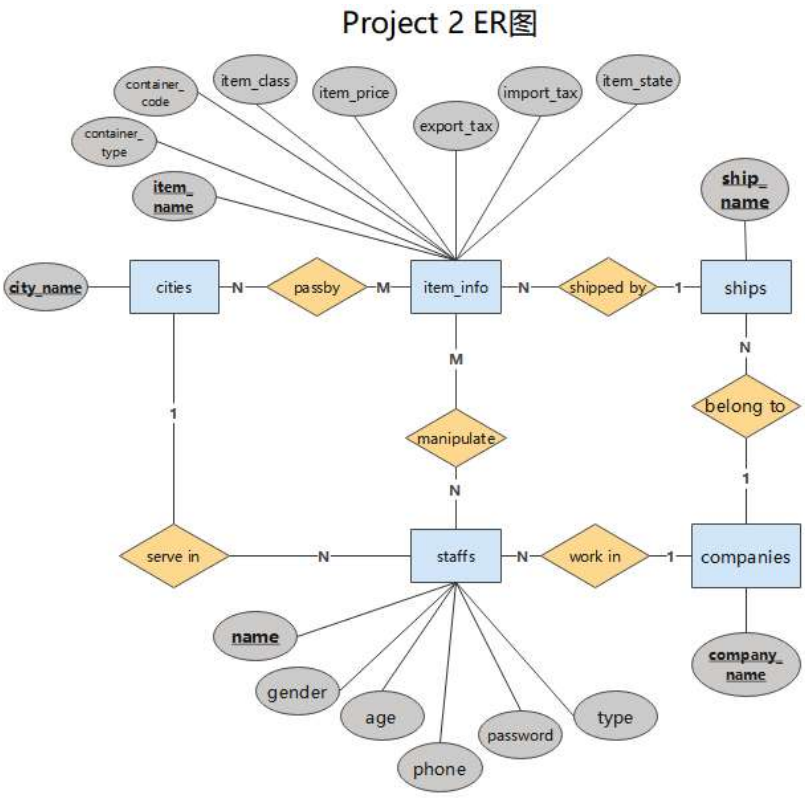
Hu Qiang	1. Design E-R Diagram 2. Write the DDL and create database users 3. The APIs of SUSTC manager and courier 4. Report typesetting
Peng Zishen	1. Design and draw E-R Diagram 2. The APIs of company manager and seaport officer 3. Write FrontEnd

2 Task 1: Database Design

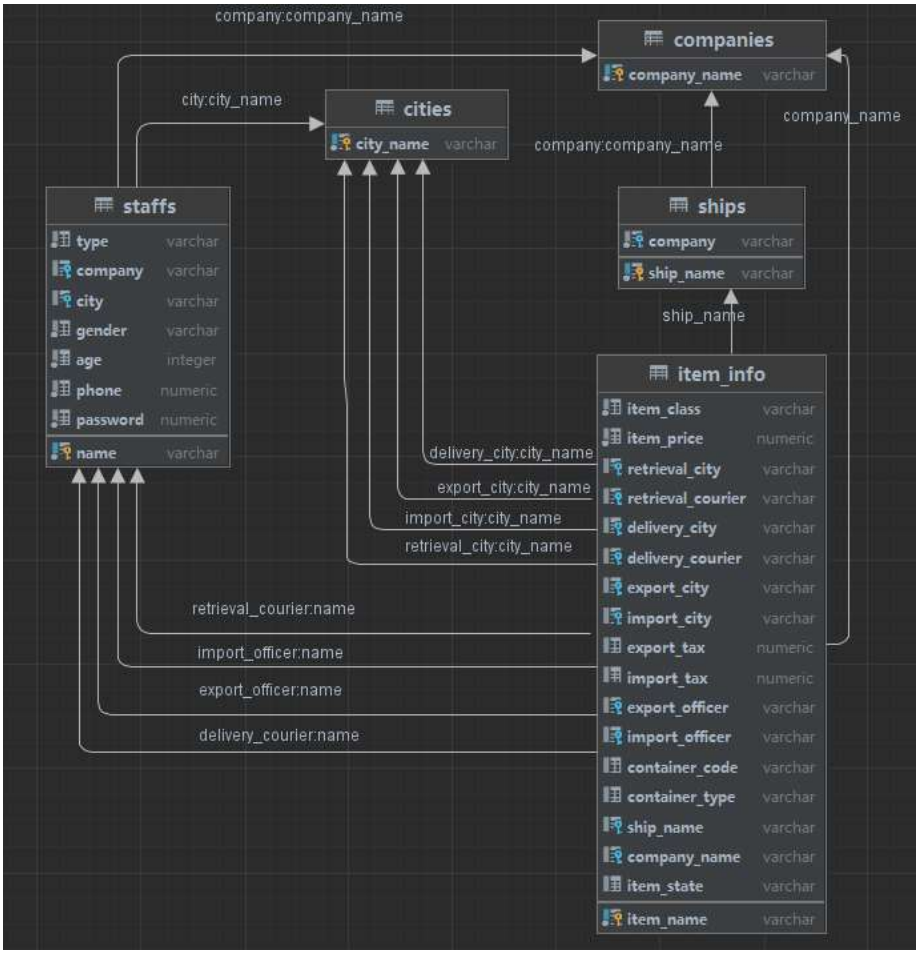
2.1 E-R Diagram

After consideration about the efficiency, we simplify the structure of the database compared with the project 1.

To decrease the time consumed when import and join query, we cancel the four tables named import, export, retrieval and delivery, which related table item to other tables. And substitute the four relationship table with foreign keys.



2.2 Database Diagram and Clarification



1. ***cities***: all cities appeared in the records so far(including retrieval city, delivery city, export city and import city).
    - city_name: the name of each city.
2. ***companies***: all companies appeared in the records so far.
    - company_name: the name of each company.
3. ***ships***: all ships appeared in the records so far.
    - ship_name: the name of each ship.
    - company: the id of the company who possessing the ship.
4. ***staffs***: all registered staffs so far.
    - name: the unique name of each staff.
    - type: type of the staff(one in SUSTCManager, courier, company manager or seaport officer).
    - company: company that the staff belong to, is null when type is seaport officer or SUSTCManager
    - city: the city where the staff served, is empty when type is SUSTCManager or company manager.
    - gender: the gender of the staff.
    - age: the age of the staff.
    - phone: the phone number of the staff.
    - password: the login password of the staff.
5. ***item_info***: information of each item that has been transported or is transporting.
    - item_name: the name of the item

- item_class: the class of the item.
- item_price: the price of the item.
- retrieval_city: the city where the item was retrieved.
- retrieval_courier: the courier who retrieval the item.
- delivery_city: the city where the item was delivered.
- delivery_courier: the courier who delivered the item, is empty when the item isn't delivered.
- export_city: the city where the item was exported.
- import_city: the city where the item was imported.
- export_tax: the tax cost to export the item.
- import_tax: the tax cost to import the item.
- export_officer: the seaport officer who check the item when exporting.
- import_officer: the seaport officer who check the item when importing.
- container_code: the code of the container which load the item.
- container_type: the type of the container which load the item.
- ship_name: the name of the ship which ship the item.
- company_name: the name of the company who arranges the transporting of this item.
- item_state: the current state of the item.

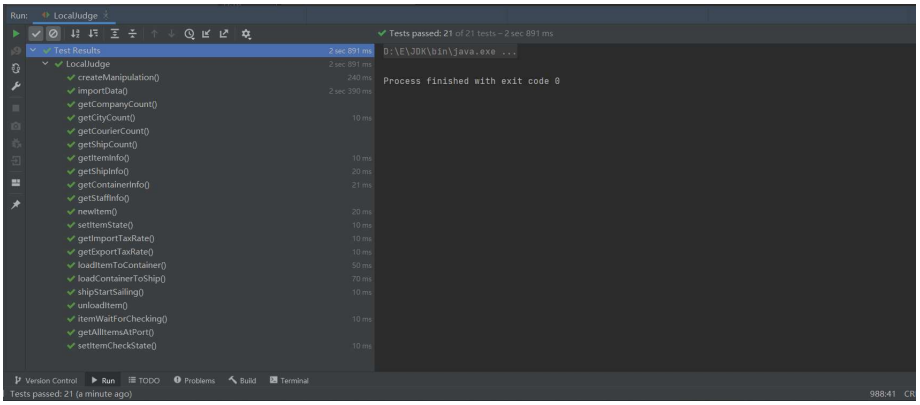
2.3 database user creation and permission granting descriptions

After importing the data, we choose out the first staff of each type and create a database user for them.

```
create user courier WITH LOGIN PASSWORD '582433470701600771';
grant update(item_state,delivery_courier),insert on public.item_info to courier;
grant select,insert on public.cities to courier;
grant connect on database project2 to courier;

create user SUSTCManager WITH LOGIN PASSWORD '248442857068449968';
grant select on public.item_info to SUSTCManager;
grant select on public.cities to SUSTCManager;
grant select on public.staffs to SUSTCManager;
grant select on public.companies to SUSTCManager;
grant select on public.ships to SUSTCManager;
grant connect on database project2 to SUSTCManager;

create user companyManager WITH LOGIN PASSWORD '2113353997898296736';
grant update(item_state),select on public.item_info to companyManager;
grant connect on database project2 to companyManager;

create user seaportOfficer WITH LOGIN PASSWORD '6542608543306460801';
grant update(item_state),select(import_city,export_city,item_state) on public.item_info to seaportOfficer;
grant connect on database project2 to seaportOfficer;
```

When call the method of specific type, get connection to the database with the corresponding database user and execute SQL statements.

3 Task 2: Basic API Specification



3.1 Table Manipulation

1. **Constructor(String database, String root, String pass);**

   Get connection with the given ***database*** path,***root*** user, and ***pass***. Execute DDL using prestatements to create the tables needed in the assigned database.

2. **void import(String recordsCSV, String staffsCSV);**

   Split the Strings ***recordsCSV*** and ***staffsCSV*** to acquire attributes of each row, then use batch loader which based on the good loader in project 1 to insert each row into the established tables in the method ***Constructor*** mentioned above.

   To optimize the import and save time, we create the necessary foreign keys and triggers after the data was imported, since the imported data is always valid. Compared with creating foreign keys and triggers before the import which consume 8 s, creating after import only consume 2.5 s.

3.2 SUSTC Department Manager User

Before execute SQL statements in the each following methods, verify the log information by query the table ***staffs***. If the staff doesn't exists or the type and password don't fit, return -1 or null, else get connection to database with the role of **SUSTCManager** and execute following process with SQL statement.

1. **int getCompanyCount(LogInfo log);**
   - **Select** the total counts from the table ***companies*** and return the counts.
2. **int getCityCount(LogInfo log);**
   - **Select** the total counts from the table ***cities*** and return the counts.
3. **int getCourierCount(LogInfo log);**
   - **Select** the total counts from the table ***couriers*** and return the counts.
4. **int getShipCount(LogInfo log);**

- **Select** the total counts from the table ***ships*** and return the counts.

5. **ItemInfo getItemInfo(LogInfo log, String name);**
    - **Select** out the item whose ***item_name*** fit the input parameter ***name*** and return its information. If no item found, return null.
6. **ShipInfo getShipInfo(LogInfo log, String name);**
    - **Select** out the item whose ***ship_name*** fit the input parameter ***name*** and get its ***ship_name*** and *company* as ***owner***. If no item found, return null.
    - Then query the count of item whose ***ship_name*** equals ***name*** and ***item_state*** equals ***Shipping***. If the count >0, then the ship is sailing, set ***sailing*** as true.
7. **ContainerInfo getContainerInfo(LogInfo log, String code);**
    - **Select** out the item whose ***container_code*** fit the input parameter ***code*** and return its ***container_code*** and ***container_type***. If no item found, return null.
    - Then query the count of item whose ***container_code*** equals ***code*** and ***item_state*** equals ***Shipping***, ***Waiting for Shipping*** or ***Packing to Container***. If the count >0, then the container is using, set ***using*** as true.
8. **StaffInfo getStaffInfo(LogInfo log, String name);**
    - **Select** out the item whose ***name*** fit the input parameter ***name*** and return its full information. If no item found, return null.

3.3 Courier User

Before execute SQL statements in the each following methods, verify the log information by query the table ***staffs***. If the staff doesn't exists or the type and password don't fit, return -1 or null or false, else get connection to database with the database role **Courier** and execute following process with SQL statement.

1. **bool newItem(LogInfo log, ItemInfo item);**
    - Check whether the necessary attributes is null.
    - Check whether ***import_city*** equals ***export_city*** or ***retrieval_city*** equals ***delivery_city***.
    - Check if item with given name already exist.
    - Check if the courier's ***city*** equals the item's ***retrieval_city***.
    - **Insert** a new row into table ***item_info*** with given ItemInfo ***item***(set the ***company_name*** as the company of whom logged in).
2. **bool setItemState(LogInfo log, String name, ItemState s);**
    - Check if the item whose ***item_name*** equals name is in proper item_state.
    - If the ***item_state*** is in the scope of retrieval, check if the courier logged in is the corresponding ***retrieval_courier***

and if the state to set is valid(in the correct scope of retrieval and don't skip any other states). If true, update ***item_state*** of the item with the given ***name***, else return false.
- If the ***item_state*** is in the scope of delivery, check if the ***delivery_courier*** is null.
- If the ***delivery_courier*** is null, check if the courier logged in is of the corresponding city and company. If true, update ***item_state*** of the item with the given name, else return false.
- If the ***delivery_courier*** already exists, check if the courier logged in is the corresponding ***delivery_courier*** and if the state to set is valid(in the correct scope of delivery and don't skip any other states). If true, update ***item_state*** of the item with the given name, else return false.

3.4 Company Manager User

Before execute SQL statements in the each following methods, verify the log information by query the table ***staffs***. If the staff doesn't exists or the type and password don't fit, return -1 or null or false, else get connection to database with the database role **CompanyManager** and execute following process with SQL statement.

1. **double getImportTaxRate(LogInfo log, String city, String itemClass);**
    - Select the item with the given import_city ***city***. If no item found, return -1.
    - calculate the import tax rate by *import_tax/item_price* and return it.
2. **double getExportTaxRate(LogInfo log, String city, String itemClass);**
    - Select the item with the given export_city ***city***. If no item found, return -1.
    - calculate the export tax rate by *export_tax/item_price* and return it.
3. **bool loadItemToContainer(LogInfo log, String itemName, String containerCode);**
    - Check if there's item with the given ***container_code*** and item state of ***Shipping***, ***Waiting for Shipping*** or ***Packing to Container*** using select statement in table ***item_info***. If true, return false.
    - Check if the item with the given ***itemName*** is in state of ***Packing to Container*** and has no ***container_code*** yet using select statement in table ***item_info***. If false, return false.
    - Update the ***container_code*** and ***container_type*** of item whose ***item_name*** equals ***itemName***.
4. **bool loadContainerToShip(LogInfo log, String shipName, String containerCode);**

- Check if there's item with the given ***container_code*** and item state of ***Shipping*** or ***Waiting for Shipping*** using select statement in table ***item_info***. If true, return false.
- Check if the item with the given ***ship_name*** is in state of ***Shipping*** using select statement in table ***item_info***. If true, return false.
- Check if the ship with ***shipName*** and the item are from the same company. If false, return false.
- Update ***ship_name*** of the item with the given ***container_code*** from null to ***shipName***.

5. **bool shipStartSailing(LogInfo log, String shipName);**
    - Check if the ship is sailing by selecting the item whose ***ship_name*** equals ***shipName*** and check its state in table ***item_info***.
    - If there's no item with state of shipping, then update the state of items whose ***ship_name*** equals ***shipName*** and ***item_state*** equals ***Waiting for Shipping*** to ***Shipping***.
6. **bool unloadItem(LogInfo log, String item);**
    - Select item with item_name of ***item*** and item_state of ***Shipping*** from table ***item_info*** to check if the specific item exists.
    - If such item exists, update its item_state to ***Unpacking from Container***.
7. **bool itemWaitForChecking(LogInfo log, String item);**
    - Select item with item_name of item and item_state of ***UnpackingFromContainer*** from table ***item_info*** to check if the specific item exists.
    - If such item exists, update its item_state to ***Importing Checking***.

3.5 Seaport Officer User

Before execute SQL statements in the each following methods, verify the log information by query the table ***staffs***. If the staff doesn't exists or the type and password don't fit, return -1 or null or false, else get connection to database with the database role **SeaportOfficer** and execute following process with SQL statement.

1. **String[] getAllItemsAtPort(LogInfo log);**
    - Select all the items whose import_city/export_city equals the logged seaport's city and item_state equals ***Importing Checking/Export Checking***.
2. **bool setItemCheckState(LogInfo log, String itemName, bool success);**
    - Check if the item_state of given item equals ***Export Checking or Import Checking***.
    - If the given item exists and the item_state valid, update the item's item_state to ***Export Check Fail/Import Check***

Fail or ***Packing to Container/From-Import Transporting*** according to the boolean ***success***.

4 Advanced APIs and Other Requirements

4.1 Other API

1. **boolean checkLogInfo(String name,int type,String password)**

   Check the log information when login.Created to facilitate the implementation of front end.

4.2 FrontEnd

4.2.1 Log In

Input name and password to log in. The program would search in the table staffs through the name to check if the password is correct. If correct, succeed to log in, else failed.

```
D:\E\JDK\bin\java.exe ...
Connection succeeded. Welcome to the SUSTC operating system.

Input your name and password to log in, or input -1 to exit the system.
Name:Yang Lizhen
Password:4364098817269137411
Login succeeded.

Hello Yang Lizhen, you are a courier, you can do the following methods.
Set a new item -- 1
Set a new item state -- 2
Log out -- -1
```

4.2.2 basic functions with permission control

When a staff log in, according to his/her type, some specific operations are allowed to conduct. And each operation are corresponding to one API in task 2. The following 2 pictures are the status when the operation succeed/failed.

```
Hello Yu Bao, you are a company manager, you can do the following methods.
Get import tax rate -- 1
Get export tax rate -- 2
Load an item to container -- 3
Load a container to ship -- 4
Set ship start sailing -- 5
Unload an item -- 6
Set item wait for checking -- 7
Log out -- -1
5
Please input the ship name.
Ship name:2f29b41b
Operation succeeded.
```

```
Hello Yu Bao, you are a company manager, you can do the following methods.
Get import tax rate -- 1
Get export tax rate -- 2
Load an item to container -- 3
Load a container to ship -- 4
Set ship start sailing -- 5
Unload an item -- 6
Set item wait for checking -- 7
Log out -- -1
4
Please input the container code and the ship name.
Container code:ac639e5a
Ship name:08b0579c
Operation failed, please try again.
```

4.2.3 Log Out

Input -1 to log out.

```
Hello Yu Bao, you are a company manager, you can do the following methods.
Get import tax rate -- 1
Get export tax rate -- 2
Load an item to container -- 3
Load a container to ship -- 4
Set ship start sailing -- 5
Unload an item -- 6
Set item wait for checking -- 7
Log out -- -1
-1
Logout succeeded.
```