CS307 - Project Ⅱ Report

Class: Lab class 3

Name: 陈茜 SID: 12011136 Contribution ratio: 50%

Name: 加泽瑄 SID: 12011126 Contribution ratio: 50%

API Description

0) Using MyBatis，creating classes for evey table and their DAO file to implement basic sql sentence.

```
public class center {}                        public interface centerDAO {}
public class contract {}                      public interface contractDAO {}
public class enterprise {}                    public interface enterpriseDAO {}
public class model {}                         public interface modelDAO {}
public class orders {}                        public interface ordersDAO {}
public class product {}                       public interface productDAO {}
public class staff {}                         public interface staffDAO {}
```

DAO file implements basic add, delete, query and update for every table, which is used for service label.

1) APIs for manipulating the original data:

This module is used to operate some simple operator such as select, delete, update and insert. （table center as example，insert as example)

```
public interface BasicCenterService {}
public class BasicCenterServiceImpl{
    public boolean add(center center);
    public boolean delete(int id);
    public center query(int id);
    public boolean change(center center);
}
    <insert id="insertCenter">
        insert into center(id,name)
        values (#{id}, #{name})
    </insert>
```

Input : what operator we want to do, 1 is select, 2 is delete, 3 is update, 4 is insert.

output : select、delete、update、insert one table

2) stock In

This module is used to stock product item in table product.

```
public interface stockInService2 {
    public void stockIn(int id, String supply_center, String product_model, int
supply_staff,
                               String date, int purchase_price, int quantity)
}
public class stockInService2impl {
    public static void stockIn(int id, String supply_center, String
product_model, int supply_staff,
                               String date, int purchase_price, int quantity) {
         SqlSession sqlSession = MyBatisUtil.getSqlSession();
         mixDAO mixDAO = sqlSession.getMapper(mixDAO.class);
         //....
}
```

Input : information: int id, String supply_center, String product_model, int supply_staff,String date, int purchase_price, int quantity

output : rows filtered which will be inserted into or updated the table product

3)placeOrder

This modle is used to place an order in table orders.

```
public interface placeOrderService3 {
    public void placeOrder(String contract_num, String enterprise, String
product_model,
                              int quantity, int contract_manager, String d, String
d1, String d2,
                              int salesman_num, String contract_type);
}
and Implementation in placeOrdersImpl3.java
```

Input : information in ()

output : rows filtered which will be inserted into the table orders

4)updateOrder

This modle is used to update an order in table orders.

```
public interface updateOrdersService4 {
    public void updateOrder(String contract_num, String product_model, int
salesman_num,
                                    int quantity, String a, String b);
}
and Implementation in updateOrdersImpl4.java
```

Input : information in ()

output : rows filtered which will be updated into the table orders according to the given information

5)deleteOrder

This modle is used to delete an order in table orders.

```
public interface deleteOrdersService5 {
    public void deleteOrder(String contract, int salesman, int sequence);
}
and Implementation in deleteOrdersImpl5.java
```

Input : information in ()

output : rows filtered which will be deleting from the table orders according to the given information

6)getAllStaffCount();

This modle is used to return number of people of all types of staffs.

```
public static void getAllStaffCount() throws SQLException {
    PreparedStatement staff_count = con.prepareStatement("select type
staff_type, count(*) from staff where type = 'Director' or type = 'Contracts
Manager' or type = 'Salesman' or type = 'Supply Staff' group by type");
}
```

7)getContractCount();

This modle is used to count the total number of the contract.

```
public static void getContractCount() throws SQLException {
    PreparedStatement contract = con.prepareStatement("select count(*) from
contract");
}
```

8)getOrderCount();

This modle is used to count the total number of the order.

```
public static void getOrderCount() throws SQLException {
    PreparedStatement order = con.prepareStatement("select count(*) from
orders");
}
```

9)getNeverSoldProductCount();

This modle is used to count the total number of the orders in the stock that were never sold out.

```
public static void getNeverSoldProductCount() throws SQLException {
}
```

10)getFavoriteProductModel();

Find the models with the highest sold quantity, and the number of sales.

```
public static void getFavoriteProductModel() throws SQLException {}
```

11)getAvgStockByCenter();

For each supply center, calculate the average quantity of the remaining product models.

```
public static void getAvgStockByCenter() throws SQLException {}
```

12)getProductByNumber(String product_number);

Find a product according to the product number and return the current inventory capacity of each product model in each supply center.

```
public static void getProductByNumber(String product_number) throws
SQLException{}
```

13)getContractInfo(String contract_number)

This modle is used to return the contract information by using a contract num.

```
public static void getContractInfo(String contract_number) throws SQLException{}
```

Database Design

Four Basic Tables: staff, enterprise, center, model.

Three self-made tables: product, orders, contract.

Create seven tables in the database, directly import the data of four basic tables, and then operate three self_made tables based on the basic data in the database.

The three tables are empty at the beginning. StockIn filters and imports data into product, and may update product according to different information.

The placeOrder method filters and imports data to Orders to add orders, and determines whether a Contract exists and imports data to a Contract.

The updateOrder method filters and updates the Orders table;

DeleteOrder prevents a judgment and deletes the Orders table.

The remaining methods are select-related methods, so I won't go into details here.

```
create table orders
(
    contract_num            varchar,
    enterprise              varchar,
    product_model           varchar,
    quantity                integer,
    contract_manager        integer not null,
```

```sql
    contract_date           date,
    estimated_delivery_date date,
    lodgement_date          date,
    salesman_num            integer not null,
    contract_type           varchar(15),
    id                      serial
);

create table product
(
    id             integer
        unique,
    supply_center  varchar,
    product_model  varchar,
    supply_staff   integer,
    date           date,
    purchase_price integer,
    quantity       integer
);

create table contract
(
    contract_num            varchar,
    enterprise              varchar,
    contract_manager        integer,
    contract_date           date,
    estimated_delivery_date date,
    lodgement_date          date,
    contract_type           varchar
);

create table center
(
    id   int,
    name varchar
);

create table enterprise(
    id int,
    name varchar,
    country varchar,
    city varchar,
    supply_center varchar,
    industry varchar
);

create table model(
    id int,
    number varchar,
    model varchar,
    name varchar,
    unit_price int
);

create table staff(
    id int,
    name varchar,
    age int,
    gender varchar,
    number int,
    supply_center varchar,
    mobile_number bigint,
    type varchar
);
```

Advance

1 Use MyBatis semi-automatic ORM framework， build the encapsulation class Util, build model->DAO->Service Interface(and Implementation) -> Controller structure, contact the servlet layer.

2 Database based data to achieve user rights management, different staff number and password can manage different data content.

3 Automatically filters out illegitimate data.

4 Better authority management. Staffs are divided to three kinds, Salesman, Supply Staff and Manager. Salesman can only do simple operator and do API2,3 5 for his own orders. Supply Staff can only stock in his own product. Manager can do all operators without limitation.

5 More vivid GUI frame and initial view. Each API has one unique sub frame to input information. After each kind staff login, system will exhibit his own details using a pie chart, a staff-info table and a item-info table.

6 Account system. In acquiescence, the password is equal to the SID.

(*More details please access https://github.com/Cici1217/databaseProj2*)