CS307 Project2: Database Application

> Main Contributors:
>
> Leader and Overall Design: ZHU Yueming
>
> Data Preparation and Documentation: WANG Lishuang, WANG Zhiyuan, ZHANG Chaozu, and WU Shangxuan
>
> Review: MA Yuxin

Introduction

Design a database based the provided data files (including the original data and the test data), and address all the requirements mentioned below.

1. Original Data

1. center.csv
2. enterprise.csv
3. model.csv
4. staff.csv

2. Basic Requirements

2.1 API Specification

To provide basic functionality of accessing a database system, you are required to build a backend library which exposes a set of application programming interfaces (APIs). The detailed specification for each API is described below.

Note that the order and naming of the parameters in the function calls can be different from what is specified in the following sections once the proper outputs can be generated (either returned or printed) in the **required format**.

> 在下面的接口设计中，我们不关注接口输入参数的顺序与返回值的顺序、名称和数量，只要大家可以打印（或展示）出我们想要的格式就可以

1) APIs for manipulating the original data:

First, it is required to implement APIs that can access the original data, including `add`, `select`, `delete` and `update` for the original data files (`enterprise.csv`, `staff.csv`, `model.csv`, `center.csv`).

- When deleting or updating a single row, the data stored in corresponding tables should be deleted or updated accordingly.
- Single-column and multi-column selections should be supported when selecting data from

the tables.

2) stockIn:

Design an API called `stockIn` to support the action of increasing the product inventory.

Some test cases for possible inputs are listed in `task1_in_stoke_test_data_publish.csv`.

Description of the columns:

- supply_center: The name of supply center
- product_model: The model of the product which is selled to the client enterprise
- supply_staff: The number of staff
- date: The time when the products are put into storage
- purchase_price: Purchase price
- quantity: Purchase quantity

Situations for invalid actions or inputs:

- The supply center and the supply center to which the supply staff belongs do not match
- The type of the supply staff is not "supply_staff"
- Supply center does not exist
- Product does not exist
- Staff does not exist

```
非法数据：
1. 供应中心与人员所在的供应中心对应不上
2. 人员的类型不是“supply_staff"
3. 供应中心不存在
4. 产品不存在
5. 人员不存在
```

3) placeOrder:

Design an API called `placeOrder` to place an order which sells specific products to an enterprise.

Some test cases for possible inputs are listed in `task2_test_data_publish.csv`.

Description of the columns:

- contract_num: The ID of the contract
- enterprise: The Name of the client enterprise
- product_model: The model of the product which is selled to the client enterprise
- quantity: The number of the product which is selled to the client enterprise
- contract_manager: The ID of the contract manager
- contract_date: The date that the contract signed
- estimated_delivery_date: Estimated delivery date of the product
- lodgement_date: Actual delivery date of the product
- salesman_num: The id of the salesman
- contract_type: Current type of the contract type

Situations for invalid actions or inputs:

- Quantity in an order larger than the stock, (The stocks are different in supply centers for each product model)
- The position of the employee corresponding to the `salesman_num` is not "Salesman".

非法数据：
1. 订单中的商品数量大于库存数量。（每个商品型号的库存在不同供应中心是不同的）
2. 人员类型不是“Salesman"

4) updateOrder

Design an API called `updateOrder` to update orders according to the contract number, product model and salesman number.

Some test cases for possible inputs are listed in `task34_update_test_data_publish.tsv`.

Description of the columns:

- contract: The contract number
- product_model: The model of the product which is selled to the client enterprise
- salesman: The salesman number
- quantity: The number of the product which is selled to the client enterprise, this vcalue may be update
- estimate_delivery_date: Estimated delivery date of the product, this vcalue may be update
- lodgement_date: Actual delivery date of the product, this vcalue may be update

Hints and possible invalid actions or inputs:

- The salesman can only update the order correspond to himself, error should be taken when illegal update appears
- The "quantity" in stock should update according to the update of quantity in order
- If the final value of quantity after update is 0, this order should be deleted from the contract
- After updating, if there is no order in contract, do not delete contract.

非法或要注意之处：
1. 销售员只能更新自己的订单。
2. 更新订单数量的同时，库存数量也要随之改变
3. 如果一个订单更新后的数量是0， 那么这个订单要在合同中移除
4. 更新后，如果一个合同中没有订单，不要删除合同

5) deleteOrder:

Design an API called `deleteOrder` to delete orders. If there is no order in a contract, please make sure you do NOT delete the contract.

Some test cases for possible inputs are listed in `task34_delete_test_data_publish.tsv`. -->

Description of the columns:

- contract: The contract number
- salesman: The salesman number
- seq: the sequence number of order, belonging to the salesman in the given contract, that should be deleted. Specifically, the sequence numeber is calculated by sort the record in the "order" table **with the same contract and salesman** by the value of **estimate_delivery_date**. If several records exist with the same estimated delivery date, sort these records by the **lexicographical order of product_model** (a-z, A-Z).

  ```
  seq：同一个合同，同一个销售员对于销售的订单有一个顺序，排序规律为首先根据
  estimate_delivery_date， 如果estimate_delivery_date相同，再根据lexicographical
  order of product_model 排序
  ```

Situations for invalid actions or inputs:

- The salesman can only delete his own order.
- The quantity in stock should also update correspond to the change of quantity in order
- After deletion, if there is no order in contract, do not delete contract.

```
非法或要注意的地方：
1. 销售员只能删除自己的订单
2. 删除订单后，库存数量要随之改变
3. 删除后，如果合同里没有订单，不要删除合同
```

6) getAllStaffCount:

Return the numbers of people for all types of staffs.

Return two columns: staff_type, count

```
-- 返回每一类staff有多少人
```

7) getContractCount:

Return the total number of existing contracts.

Return a number: count of the existing contracts

```
-- 合同的总数量
```

8) getOrderCount:

Return the total number of existing orders.

Return a number: count of the existing orders

```
-- 订单的总数量
```

9) getNeverSoldProductCount:

The number of `product_models` that are in stock but have never been ordered (sold).

Return a number: count of the never-sold products

```
-- 库存有余量但还没有卖出过的product_model的总量是
```

10) getFavoriteProductModel:

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

Return two columns: model_name, quantity

```
-- 销售量最高的商品型号，以及销量
```

11) getAvgStockByCenter:

For each supply center, calculate the average quantity of the remaining product models. The results should be ordered by the name of the supply centers and rounded to one decimal place.

Return two columns: supply_center, average

```
-- 按供应中心划分，计算商品型号库存数量的平均数。结果按供应中心名称排序，每行的平均数保留1位小数
```

12) getProductByNumber:

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

Input parameters: product_number

Return five columns: supply_center, product_number, product_model, purchase_prise, quantity

```
--根据产品number查找，当前各个供应中心该产品各个型号的库存容量
```

13) getContractInfo

Find a contract with a contract number and return the content of the contract.

Input parameters: contract_number

Return values:

contract_number, contract_manager_name, enterprise_name, supply_center

All orders in contract including:

Product_model, salesman_name, quantity, unit_price, estimate_delivery_date, lodgement_date.

```
--根据合同编号，返回合同信息。包含：
-- 合同编号，合同负责人姓名，企业名称，供应中心。
-- 所有的产品型号、销售员、数量、单价、预计到货日期，实际到货日期
```

2.2 Functional Requirements (75%):

- It is required to use a general-purpose programming language which can interact with the database to fulfill all the requirements mentioned in Section 2.1.
- **Independent testing**: Your program should provide a command-line-based, GUI-based or Webpage-based testing interface to test all the APIs with given input data.
- **One-step input:** You program should be able to run the following tasks with a single command:
  - Read all original data into the database
  - Stoke-in all data in `task1_in_stoke_test_data_publish.csv`
  - Place all order in `task2_test_data_publish.csv`
  - Update order and delete order according to `task34_update_test_data_publish.tsv`, `task34_delete_test_data_publish.tsv`
- **One-step export**: In addition, your program should support exporting data into the testing format described in Section 4.
- Please use database systems to store all the data but not files or in-memory storage with variables.

1. 以上要求要用编程语言访问数据库。
2. 独立测试：对于上述每一个接口，可以在命令端、图形页面或网页进行独立的输入输出测试。
3. 一步读入数据设计：使用程序可以一键读入原始数据，再依次读入`task1_in_stoke_test_data_publish.csv`，`task2_test_data_publish.csv`，`task34_update_test_data_publish.tsv`，`task34_delete_test_data_publish.tsv`等测试文件。
4. 一步输出数据设计：使用程序可以一键按照Section 4格式导出输出数据。
5. 要用到数据库，而非文件或内存存储

3. Advanced Requirements: (25%)

> If you would like to get the full mark for any advanced requirement, please try to demonstrate yourself by providing a high-quality solution.

- Based on the basic requirements in Section 2, further enhance the usability of the APIs to accept more flexible types of requests. You may think about more requirements than those proposed in Section 2 and implement the new requirements. (up to 15%)
- Encapsulate the features and implement a real back-end server instead of several independent scripts or programs. The server should provide socket-based communication or HTTP/RESTful web services. (up to 10%)
  - For example: Query the order list based on multiple parameters, and the parameters can be null or not.
  - For example: Design the Bill Module

    - For example: Design a mechanism to change order status according to time and date.
- Apply database connection pools; (up to 4%)
- A usable and beautiful GUI design for data presentation; (up to 4%)
- Proper use of user privileges, procedures, indexing, and views in a reasonable manner; (up to 10%)
- Support high-concurrent or distributed access with proper pressure tests. (up to 15%)

1. 在保证Section2需求的基础上，进一步完善API设计，可以让其接受更为灵活参数的请求。源于Section2的同时，设计出更多的系统功能性需求。
    - 例如：根据多个参数查询订单列表，传入的参数可以为空也可以不为空。
    - 例如：设计账单模块
    - 例如：设计根据时间日期更改订单状态的机制
2. 封装并实现一个真正的后端服务器，而不是几个独立的脚本程序。服务器应提供基于套接字的通信或HTTP/RESTful Web 服务。
3. 使用数据库连接池。
4. 构造一个用于呈现数据的实用又美化的界面
5. 合理使用用户权限、过程、索引、视图等功能
6. 支持高并发，分布式设计，并要通过适当的压力测试。

4. How to Test Basic Requirement?

Testing Procedure

1. Import all the original data files into database. `center.csv`, `enterprise.csv`, `model.csv`, `staff.csv`
2. **Data Import Test**: Import all testing files in the order of `in_stock.csv`, `place_order.tsv`, `.tsv` by invoking the `stockIn`, `placeOrder`, `updateOrder`, `deleteOrder` methods.
3. **Output Test**: Test the outputs by calling the interfaces in Section 2 from 6) to 13) with parameters: `product_number` (for `12`), and `contract_number` (for `13`).

   Input:

   ```
   A50L172   // product number
   CSE0000106 // contract number
   CSE0000209 // contract number
   CSE0000306 // contract number
   ```

   Output:

   ```
   find in output.txt
   ```

4. Test the deleting and updating of the tables which store the original data (i.e., `center.csv`, `enterprise.csv`, `model.csv`, `staff.csv`).

5. What to Submit

Please submit your report and attachments before **23:55** on **June 2nd 2022**, including:

- All scripts you have written.
- A report no longer than 8 pages. The following content should be include:
    - Basic information of your group: Please follow the same requirement as described in Project 1.
    - API specification of your code: Please describe the purpose and use of interfaces (you may only use 1-2 sentences for each API in case the report becomes too long). Also, you need to illustrate the types and meanings of the parameters and return values. You can take any API documents of mature open source projects as references of how to organize the specification of your interfaces.
    - If you have finished any advanced requirements, describe what you have done and how you did it.

Disclaimer

The names, characters, businesses, and events in the background of this project are purely fictional. The items in the files are randomly-generated fake data. Any resemblance to actual events, entities or persons is entirely coincidental and should not be interpreted as views or implications of the teaching group of CS307.