SEJONG UNIVERSITY

**Department of Computer Science and Engineering**

# PROJECT REPORT

**18011557**

**SooHwan Son(손수환)**

**Spring 2022**

# Table of Contents

# I. Introduction

## I - 1. Abstract about Project

I utilized MS-SQL as well as flask which is one of python back-end framework.

With ‘pymssql’ module, We can connect Database with local server, add user’s input data, and remove DB data on web environment. Also we can view DB data on web environment

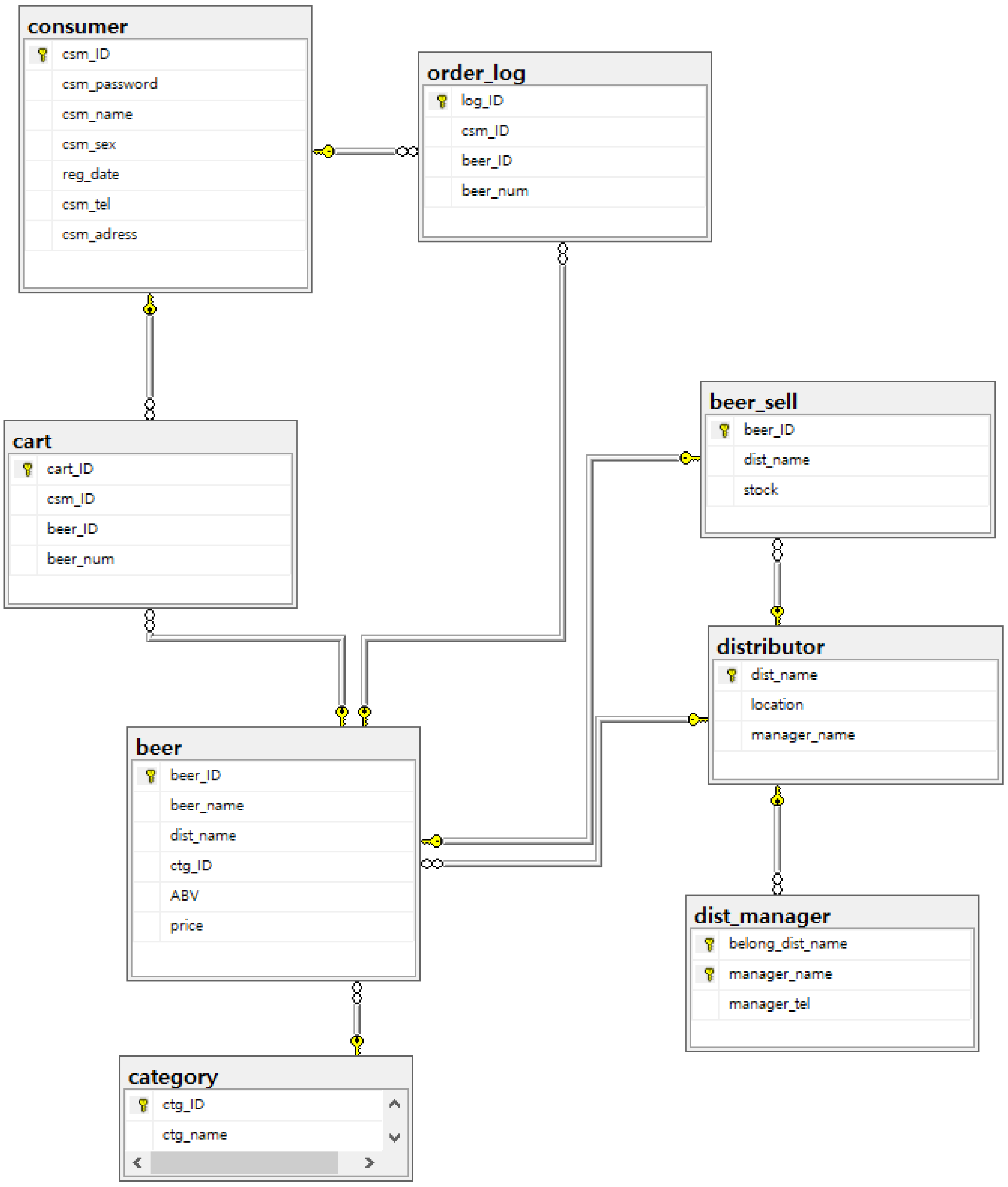
## I - 2. Background

Because I want to get a job about Back-end field, I interested in database and link DB with web.
Not only that, I love beer and my second dream is beer-shop manager(or owner)
So, I have hoped to design, create, and manage my own beer Database

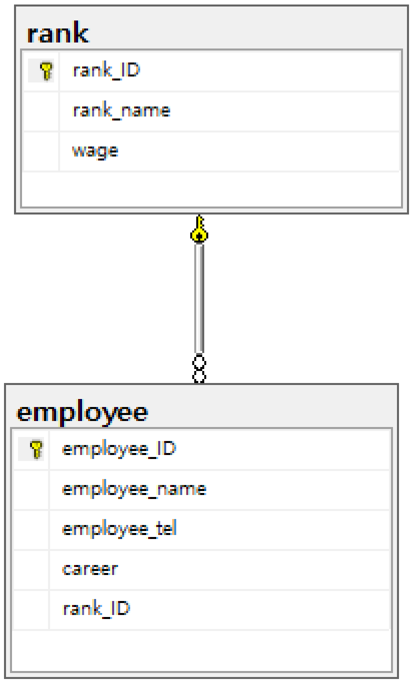
## II. DB Structure

### II - 1. Schema Diagram

**<Relations about beers and customers>**



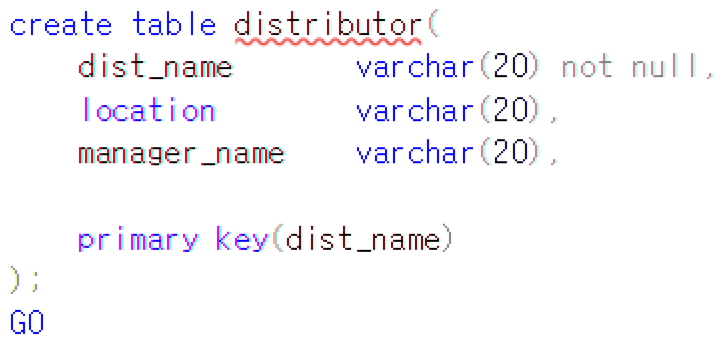
## <Relations about employee>



employee and beer is not related. So, They are should be seperated

## II - 2. Tables

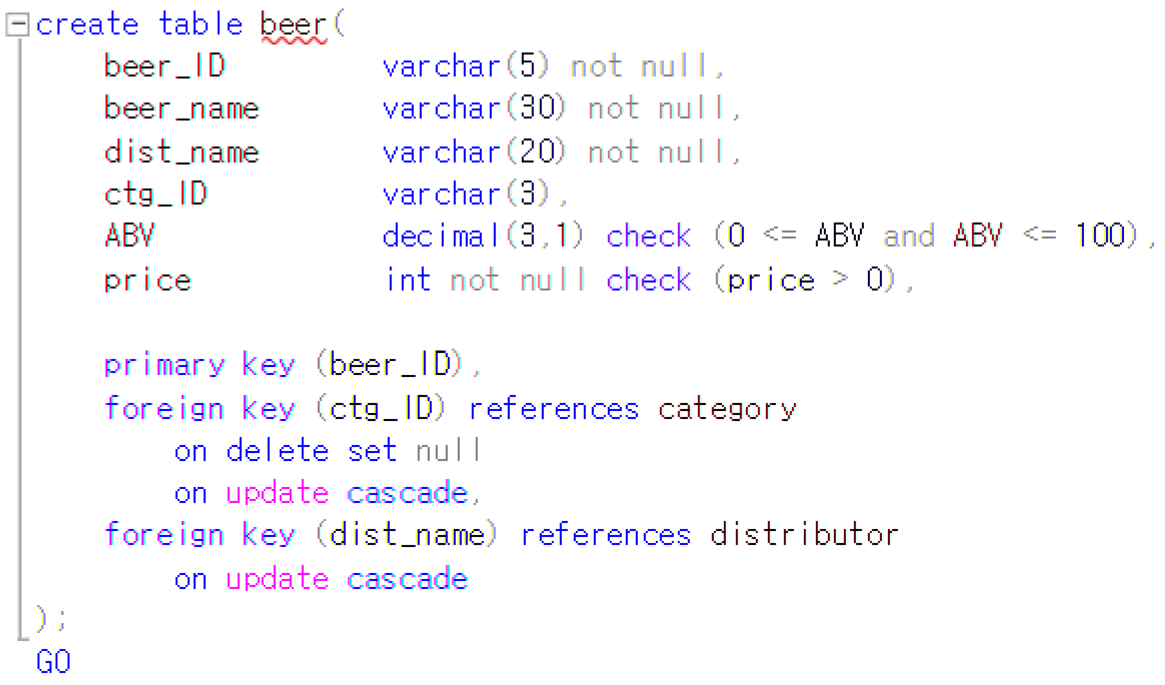
**<distributor>**



```
create table distributor(
    dist_name       varchar(20) not null,
    location        varchar(20),
    manager_name    varchar(20),

    primary key(dist_name)
);
GO
```

Table about distributor that distribute beer to shop

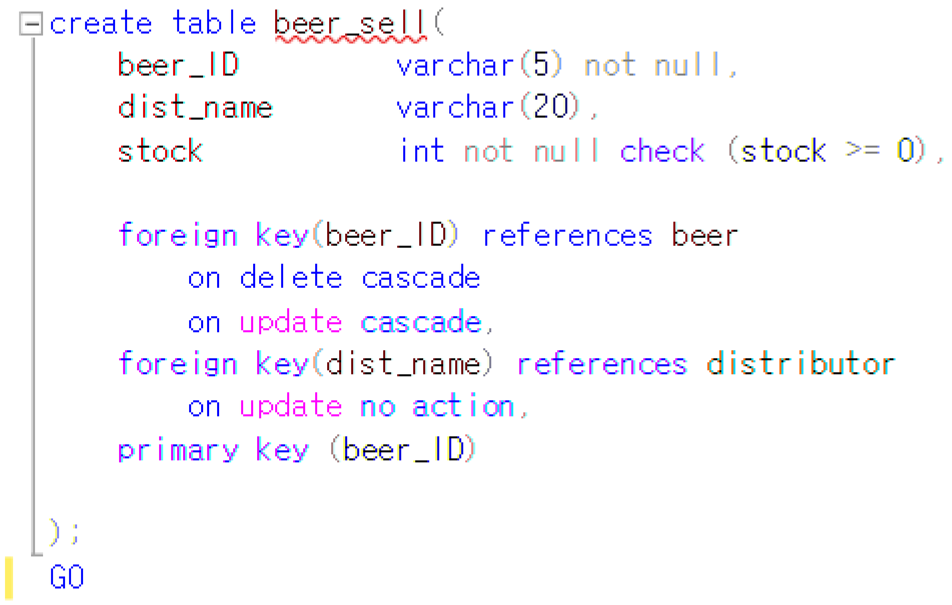
**<beer>**



```
create table beer(
    beer_ID         varchar(5) not null,
    beer_name       varchar(30) not null,
    dist_name       varchar(20) not null,
    ctg_ID          varchar(3),
    ABV             decimal(3,1) check (0 <= ABV and ABV <= 100),
    price           int not null check (price > 0),

    primary key (beer_ID),
    foreign key (ctg_ID) references category
        on delete set null
        on update cascade,
    foreign key (dist_name) references distributor
        on update cascade
);
GO
```

Table about information of beer
price should be over 0

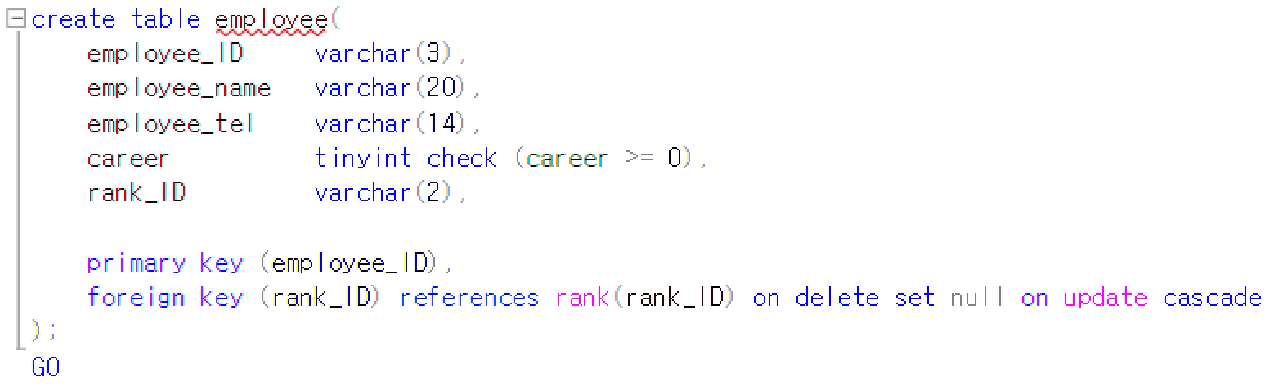
## <beer_sell>



```
create table beer_sell(
    beer_ID         varchar(5) not null,
    dist_name       varchar(20),
    stock           int not null check (stock >= 0),

    foreign key(beer_ID) references beer
        on delete cascade
        on update cascade,
    foreign key(dist_name) references distributor
        on update no action,
    primary key (beer_ID)

);
GO
```

Table about information of selling beer
stock must be a positive integer(include zero)

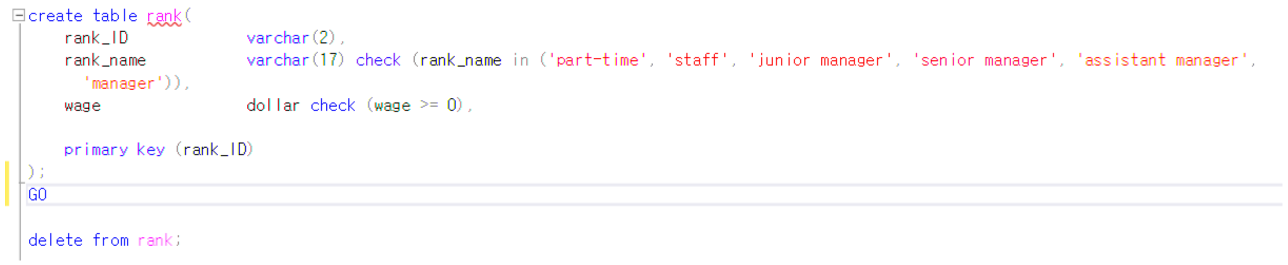
## <employee>



```
create table employee(
    employee_ID      varchar(3),
    employee_name    varchar(20),
    employee_tel     varchar(14),
    career           tinyint check (career >= 0),
    rank_ID          varchar(2),

    primary key (employee_ID),
    foreign key (rank_ID) references rank(rank_ID) on delete set null on update cascade
);
GO
```

Table about information of employee

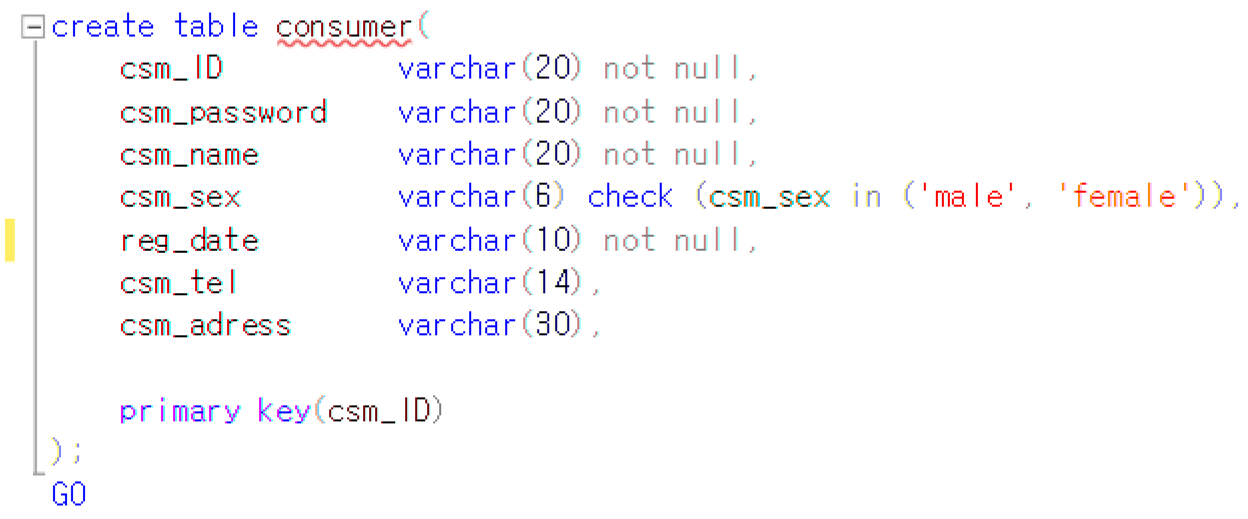
## <rank>



```
create table rank(
    rank_ID             varchar(2),
    rank_name           varchar(17) check (rank_name in ('part-time', 'staff', 'junior manager', 'senior manager', 'assistant manager',
      'manager')),
    wage                dollar check (wage >= 0),

    primary key (rank_ID)
);
GO

delete from rank;
```

Table about information of employee’s rank
rank_name must one of the exist positions

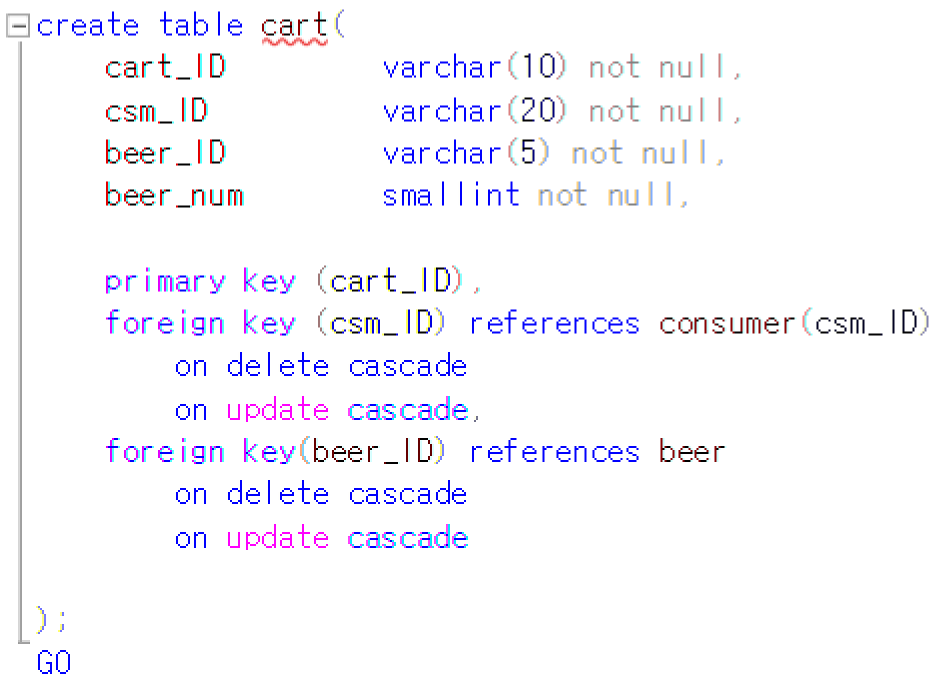
**<consumer>**



```sql
create table consumer(
	csm_ID		varchar(20) not null,
	csm_password	varchar(20) not null,
	csm_name	varchar(20) not null,
	csm_sex		varchar(6) check (csm_sex in ('male', 'female')),
	reg_date	varchar(10) not null,
	csm_tel		varchar(14),
	csm_adress	varchar(30),

	primary key(csm_ID)
);
GO
```

Table about information of consumer
csm_sex must be male or female

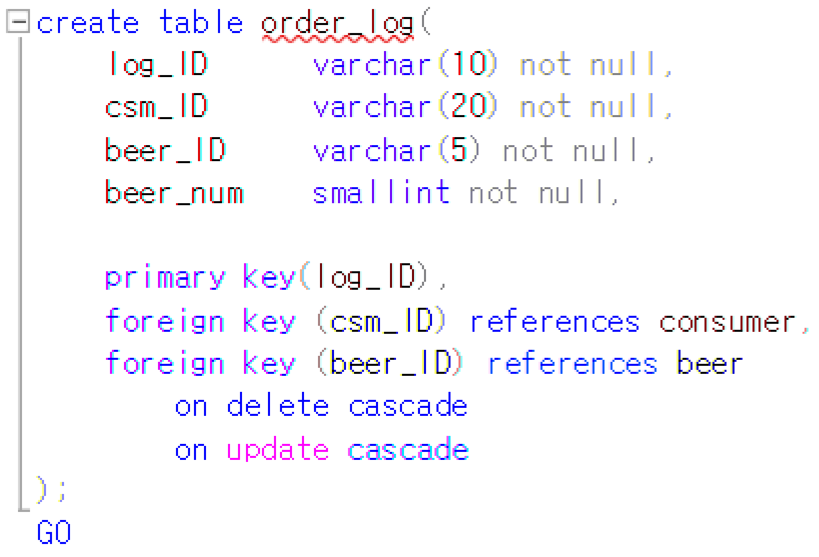
**<cart>**



```sql
create table cart(
	cart_ID		varchar(10) not null,
	csm_ID		varchar(20) not null,
	beer_ID		varchar(5) not null,
	beer_num	smallint not null,

	primary key (cart_ID),
	foreign key (csm_ID) references consumer(csm_ID)
		on delete cascade
		on update cascade,
	foreign key(beer_ID) references beer
		on delete cascade
		on update cascade

);
GO
```

Table about information of consumer’s cart

**<order_log>**



```
create table order_log(
    log_ID      varchar(10) not null,
    csm_ID      varchar(20) not null,
    beer_ID     varchar(5) not null,
    beer_num    smallint not null,

    primary key(log_ID),
    foreign key (csm_ID) references consumer,
    foreign key (beer_ID) references beer
        on delete cascade
        on update cascade
);
GO
```

Table about information of order log that consumer ordered

# III. Tools

## III - 1. MS-SQL

Microsoft SQL Server(hereafter MS-SQL) is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications

## III - 2. Python

Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.
Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming.
It can be used on a variety of platforms and is rich in libraries (modules). So, It has high scalability.

### i) pymssql

pymssql is a module that can link python and MS-SQL. We can manipulate MS-SQL like below
First, Connect with SQL server

```
conn = pymssql.connect(host='localhost',
                       database='beerShop_DB')
```

Second, Create object about MS-SQL server and execute query

```
cursor = conn.cursor()
cursor.execute('str that execute')
```

These are all thing that manipulate MS-SQL with python

### ii) flask

Flask is a micro web framework written in Python. It helps developer can make web quickly, easily and concisely. Also, It has extensible because it is 'micro' web framework

We can make web page like below

```
@app.route('/', methods=['GET'])
def home():
    return render_template('index.html')
```

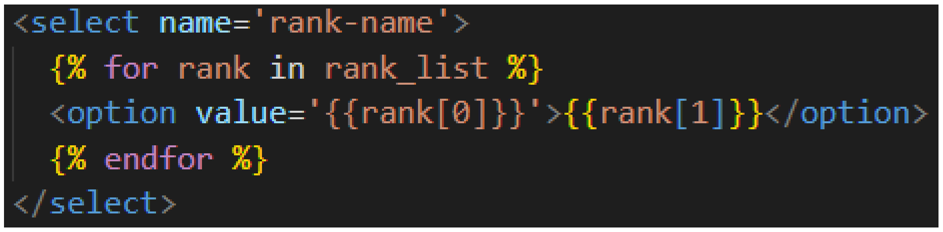
We can decide route and methods.
Also, Giving the name of the html you created as an argument in render_template function, a web page is created.

Not only that, We can write python's data to html page like below

<in python>

```
return render_template('search_employee.html',
                       rank_list = get_rank_info(cursor),
                       employee_list = get_employee_list(cursor, args_dict))
```

<in HTML>



```
<select name='rank-name'>
  {% for rank in rank_list %}
  <option value='{{rank[0]}}'>{{rank[1]}}</option>
  {% endfor %}
</select>
```

We can use for-loop in HTML with flask

If you want to get query in URL, We can use this code(in POST method)

```
args_dict = request.form.to_dict()
```

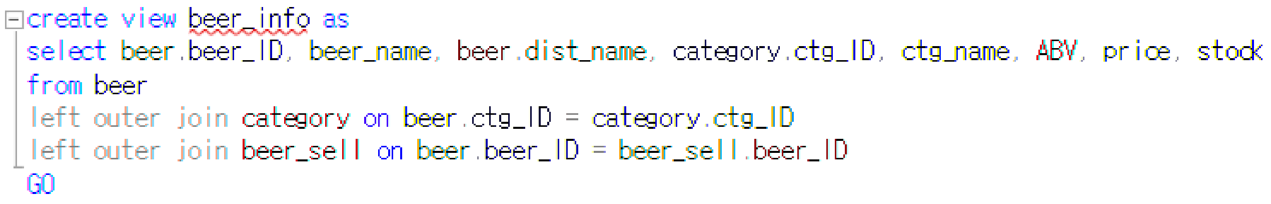
then, you can get query string with dictionary

# IV. Method

## IV - 1. View

I created views for convenience, speed up, and protection

For convenience, I organized the information of different tables into one view like below



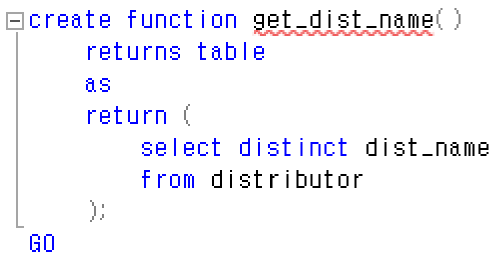
```
create view beer_info as
select beer.beer_ID, beer_name, beer.dist_name, category.ctg_ID, ctg_name, ABV, price, stock
from beer
left outer join category on beer.ctg_ID = category.ctg_ID
left outer join beer_sell on beer.beer_ID = beer_sell.beer_ID
GO
```

beer_info is composed of informations of beer, category, and beer_sell
we can get all of beer's information which is not in beer table

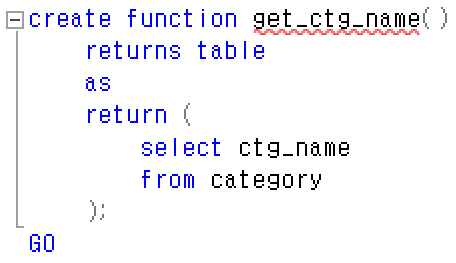
For speed up, Viewing using a pre-made view is faster than continuing to join a table

For protection, Even if the view is changed by a malicious person(such as hacker), there is no effect on the original table
Therefore, the original data is unlikely to be tampered with

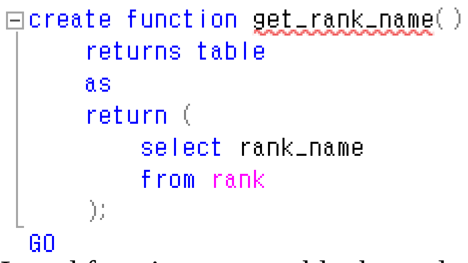
## III - 2. Function



```
create function get_dist_name()
    returns table
    as
    return (
        select distinct dist_name
        from distributor
    );
GO
```



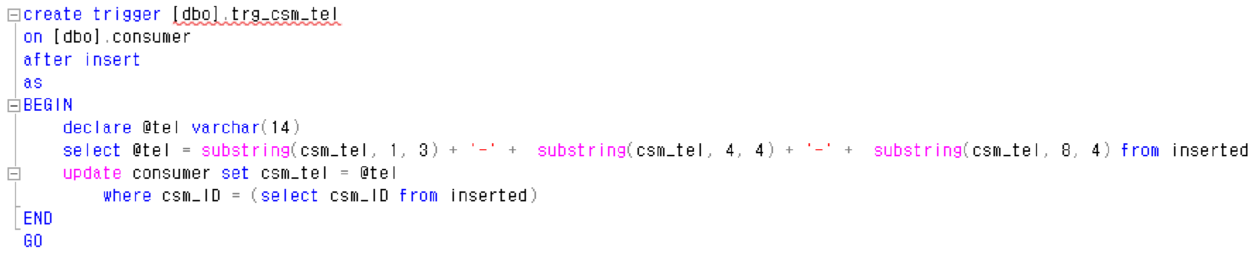
```
create function get_ctg_name()
    returns table
    as
    return (
        select ctg_name
        from category
    );
GO
```



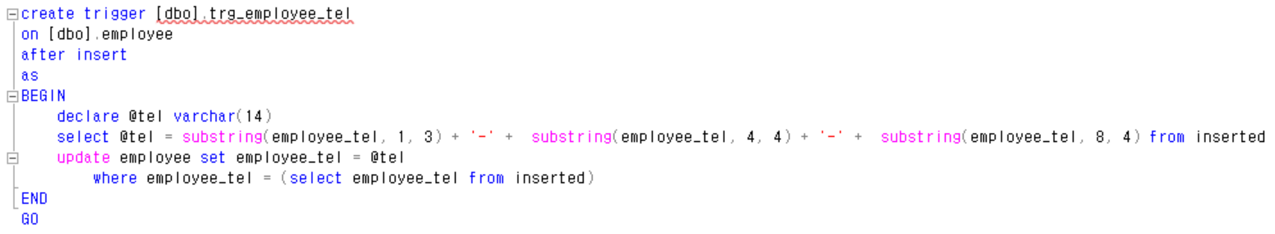
```
create function get_rank_name()
    returns table
    as
    return (
        select rank_name
        from rank
    );
GO
```

I used function to get table that only has 'name' for convenient
Select distinct name and create table with them, and return

## IV - 3. Trigger



```
create trigger [dbo].trg_csm_tel
on [dbo].consumer
after insert
as
BEGIN
    declare @tel varchar(14)
    select @tel = substring(csm_tel, 1, 3) + '-' +  substring(csm_tel, 4, 4) + '-' +  substring(csm_tel, 8, 4) from inserted
    update consumer set csm_tel = @tel
        where csm_ID = (select csm_ID from inserted)
END
GO
```

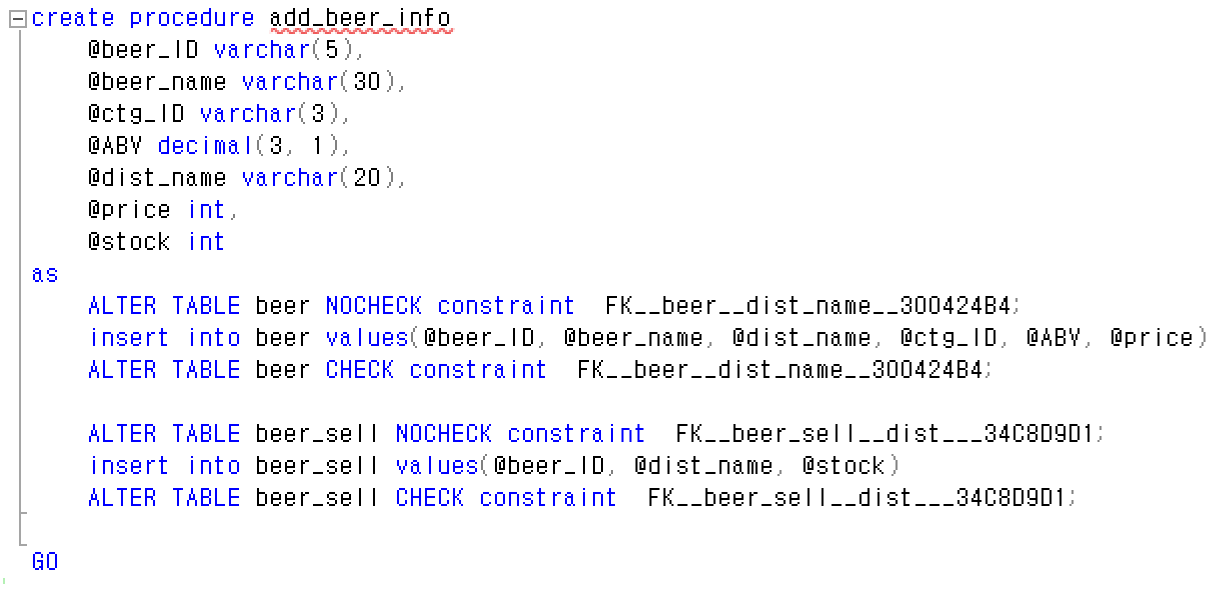


```
create trigger [dbo].trg_employee_tel
on [dbo].employee
after insert
as
BEGIN
    declare @tel varchar(14)
    select @tel = substring(employee_tel, 1, 3) + '-' +  substring(employee_tel, 4, 4) + '-' +  substring(employee_tel, 8, 4) from inserted
    update employee set employee_tel = @tel
        where employee_tel = (select employee_tel from inserted)
END
GO
```

Trigger to change the mobile phone number form like
“01012345678” → “010-1234-5678”

## IV - 4. Procedure

### <add_beer_info>

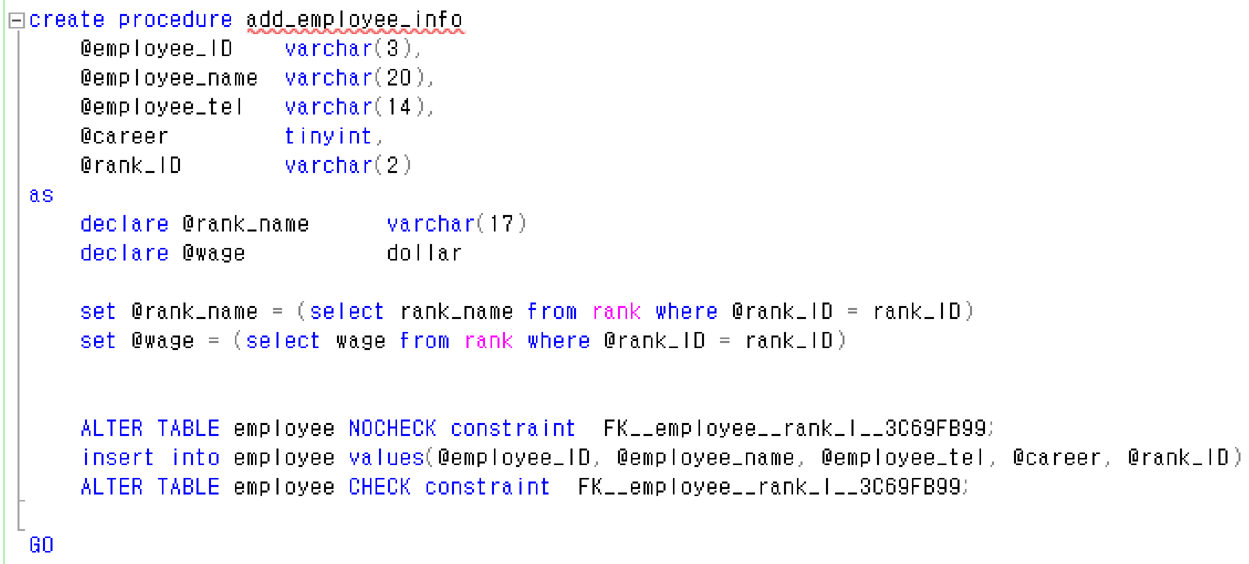


```
create procedure add_beer_info
    @beer_ID varchar(5),
    @beer_name varchar(30),
    @ctg_ID varchar(3),
    @ABV decimal(3, 1),
    @dist_name varchar(20),
    @price int,
    @stock int
as
    ALTER TABLE beer NOCHECK constraint  FK__beer__dist_name__300424B4;
    insert into beer values(@beer_ID, @beer_name, @dist_name, @ctg_ID, @ABV, @price)
    ALTER TABLE beer CHECK constraint  FK__beer__dist_name__300424B4;

    ALTER TABLE beer_sell NOCHECK constraint  FK__beer_sell__dist___34C8D9D1;
    insert into beer_sell values(@beer_ID, @dist_name, @stock)
    ALTER TABLE beer_sell CHECK constraint  FK__beer_sell__dist___34C8D9D1;

GO
```

When user input data to add beer data from server, this procedure will execute to add the information to database

dist_name in beer and beer_sell is foreign key.
So, Before insert the values, constraint must be temporary removed

### <add_employee _info>



```
create procedure add_employee_info
    @employee_ID    varchar(3),
    @employee_name  varchar(20),
    @employee_tel   varchar(14),
    @career         tinyint,
    @rank_ID        varchar(2)
as
    declare @rank_name      varchar(17)
    declare @wage           dollar

    set @rank_name = (select rank_name from rank where @rank_ID = rank_ID)
    set @wage = (select wage from rank where @rank_ID = rank_ID)


    ALTER TABLE employee NOCHECK constraint  FK__employee__rank_I__3C69FB99;
    insert into employee values(@employee_ID, @employee_name, @employee_tel, @career, @rank_ID)
    ALTER TABLE employee CHECK constraint  FK__employee__rank_I__3C69FB99;

GO
```

This is similar procedure to the above, but rank_name and wage is retrieved from employee table

## IV - 5. Extract Data from MS-SQL

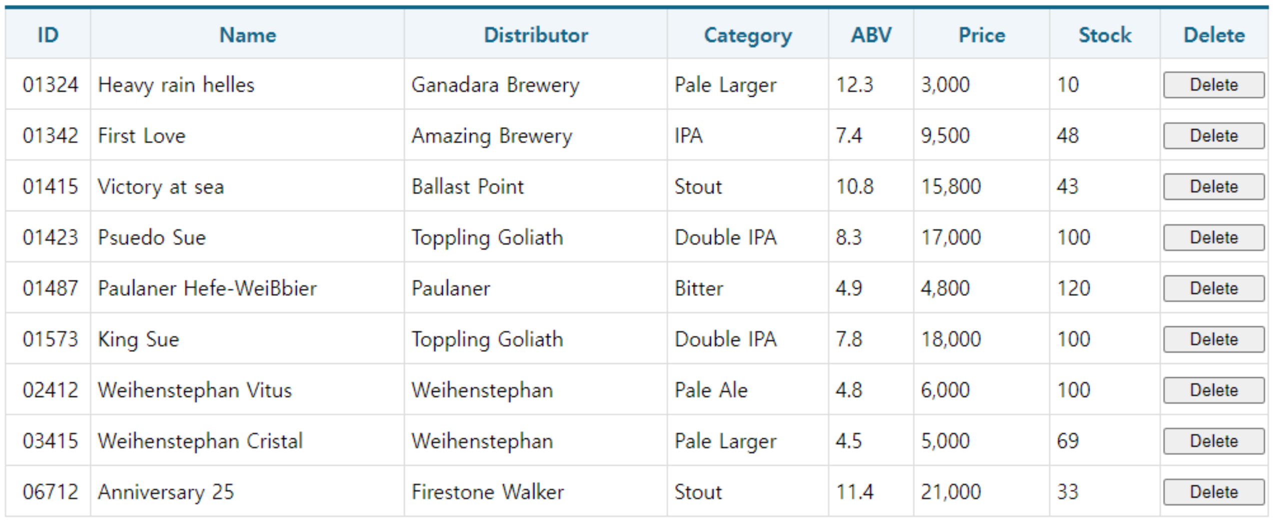
```python
def get_beer_list(cursor, args_dict):
    # Join other tables to make beer info table
    query_str = "select * from beer_info"

    # if URL query string exist
    if args_dict:
        print(args_dict)
        query_str += f" where ({args_dict['ABV-min']} <= ABV and ABV <= {args_dict['ABV-max']})"

        if args_dict['beer-name'] !='':
            query_str += f" AND beer_name like '%{args_dict['beer-name']}%'"
        if args_dict['category'] != 'all':
            query_str += f" AND ctg_ID = '{args_dict['category']}'"
        if args_dict['distributor'] != 'all':
            query_str += f" AND dist_name = '{args_dict['distributor']}'"

        if args_dict['order-by'] == 'name-asce':
            query_str += " order by beer_name"
        elif args_dict['order-by'] == 'name-dsce':
            query_str += " order by beer_name DESC"
        elif args_dict['order-by'] == 'ABV-asce':
            query_str += " order by ABV"
        elif args_dict['order-by'] == 'ABV-dsce':
            query_str += " order by ABV DESC"

    #print(query_str)

    # execute query
    cursor.execute(query_str)

    # list to save every beer info
    beer_list = []

    row = cursor.fetchone()
    while row:

        # add comma to price
        row = list(row)

        row[6] = format(row[6], ',d')

        # add beer information
        beer_list.append(row)

        # fetch next beer's info
        row = cursor.fetchone()

    return beer_list
```

As Using query string, We can extract data what we want
Then, data we extract transform to list and define

## IV - 6. Real-time Delete

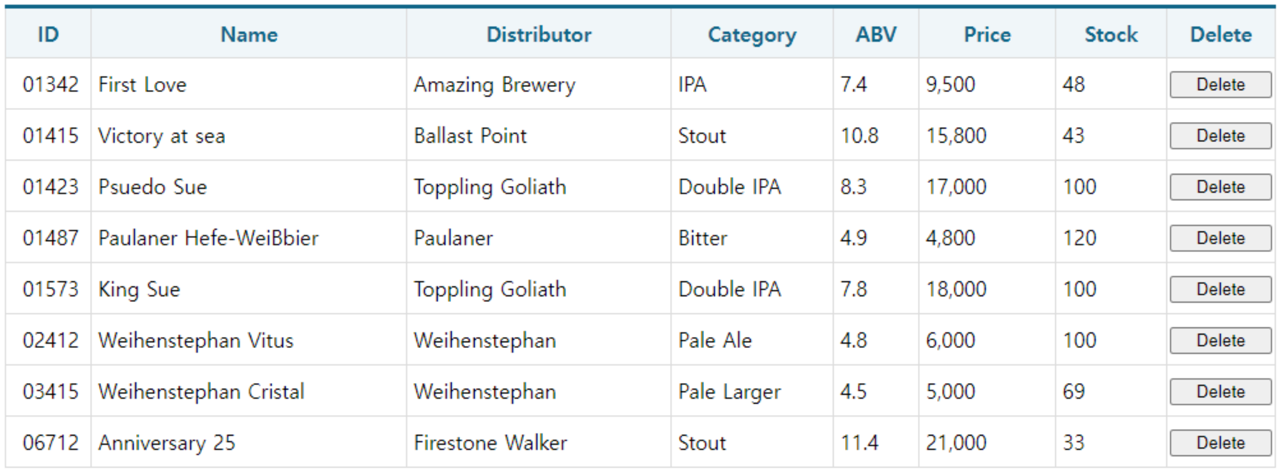
If  you click delete button in this page, the data is deleted real-time

<before>



| ID | Name | Distributor | Category | ABV | Price | Stock | Delete |
|---|---|---|---|---|---|---|---|
| 01324 | Heavy rain helles | Ganadara Brewery | Pale Larger | 12.3 | 3,000 | 10 | Delete |
| 01342 | First Love | Amazing Brewery | IPA | 7.4 | 9,500 | 48 | Delete |
| 01415 | Victory at sea | Ballast Point | Stout | 10.8 | 15,800 | 43 | Delete |
| 01423 | Psuedo Sue | Toppling Goliath | Double IPA | 8.3 | 17,000 | 100 | Delete |
| 01487 | Paulaner Hefe-WeiBbier | Paulaner | Bitter | 4.9 | 4,800 | 120 | Delete |
| 01573 | King Sue | Toppling Goliath | Double IPA | 7.8 | 18,000 | 100 | Delete |
| 02412 | Weihenstephan Vitus | Weihenstephan | Pale Ale | 4.8 | 6,000 | 100 | Delete |
| 03415 | Weihenstephan Cristal | Weihenstephan | Pale Larger | 4.5 | 5,000 | 69 | Delete |
| 06712 | Anniversary 25 | Firestone Walker | Stout | 11.4 | 21,000 | 33 | Delete |

<after>



| ID | Name | Distributor | Category | ABV | Price | Stock | Delete |
|---|---|---|---|---|---|---|---|
| 01342 | First Love | Amazing Brewery | IPA | 7.4 | 9,500 | 48 | Delete |
| 01415 | Victory at sea | Ballast Point | Stout | 10.8 | 15,800 | 43 | Delete |
| 01423 | Psuedo Sue | Toppling Goliath | Double IPA | 8.3 | 17,000 | 100 | Delete |
| 01487 | Paulaner Hefe-WeiBbier | Paulaner | Bitter | 4.9 | 4,800 | 120 | Delete |
| 01573 | King Sue | Toppling Goliath | Double IPA | 7.8 | 18,000 | 100 | Delete |
| 02412 | Weihenstephan Vitus | Weihenstephan | Pale Ale | 4.8 | 6,000 | 100 | Delete |
| 03415 | Weihenstephan Cristal | Weihenstephan | Pale Larger | 4.5 | 5,000 | 69 | Delete |
| 06712 | Anniversary 25 | Firestone Walker | Stout | 11.4 | 21,000 | 33 | Delete |

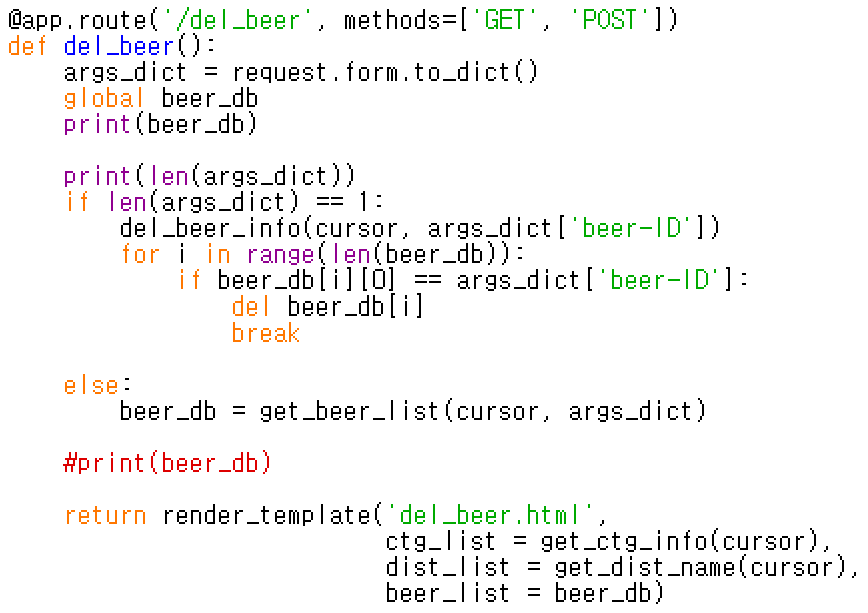
ID 01324 beer is deleted!

To make it like this, it was implemented as follows.

First of all, I declared temporary database

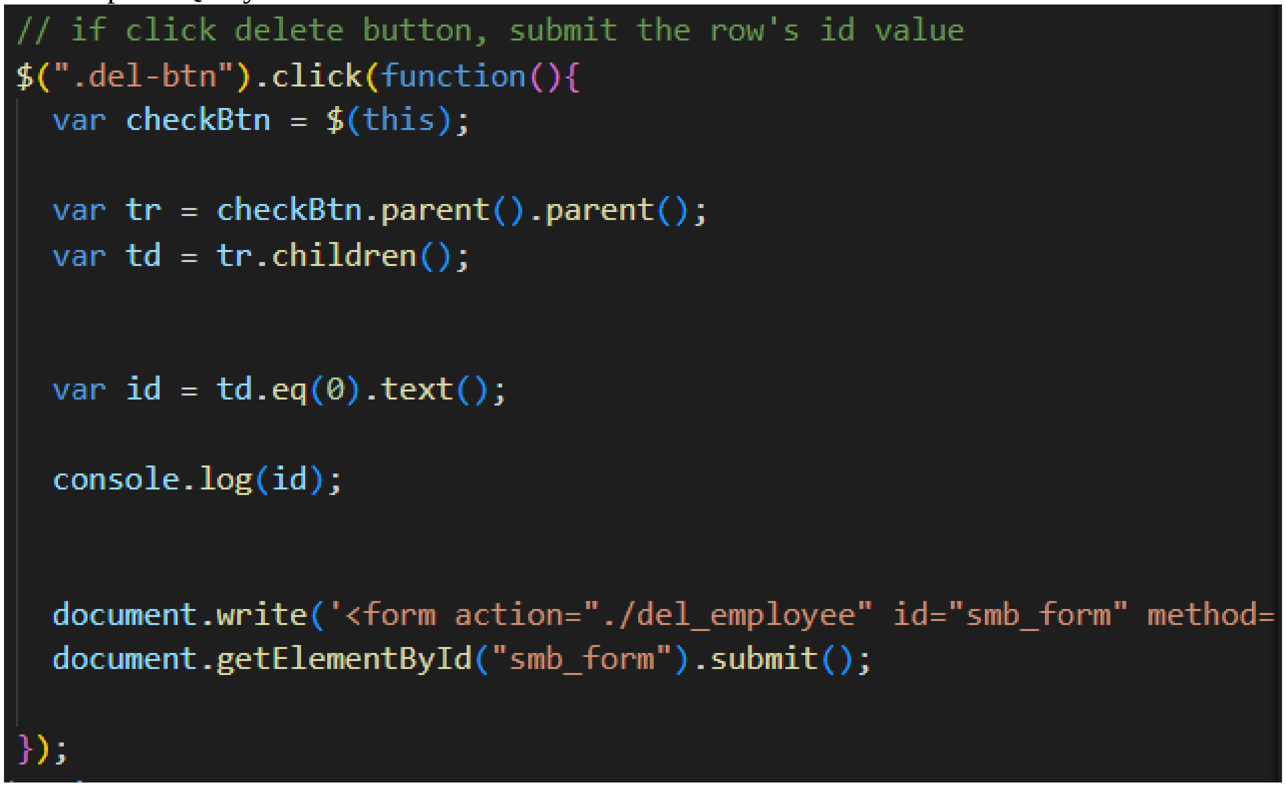
```
# temporary database
beer_db = []
employee_db = []
```

Second, I implement function like below



```python
@app.route('/del_beer', methods=['GET', 'POST'])
def del_beer():
    args_dict = request.form.to_dict()
    global beer_db
    print(beer_db)

    print(len(args_dict))
    if len(args_dict) == 1:
        del_beer_info(cursor, args_dict['beer-ID'])
        for i in range(len(beer_db)):
            if beer_db[i][0] == args_dict['beer-ID']:
                del beer_db[i]
                break

    else:
        beer_db = get_beer_list(cursor, args_dict)

    #print(beer_db)

    return render_template('del_beer.html',
                           ctg_list = get_ctg_info(cursor),
                           dist_list = get_dist_name(cursor),
                           beer_list = beer_db)
```

if query string has only one, it means user click the delete button. Because, I implement this Javascript + JQuery function



```javascript
// if click delete button, submit the row's id value
$(".del-btn").click(function(){
  var checkBtn = $(this);

  var tr = checkBtn.parent().parent();
  var td = tr.children();


  var id = td.eq(0).text();

  console.log(id);


  document.write('<form action="./del_employee" id="smb_form" method=
  document.getElementById("smb_form").submit();

});
```

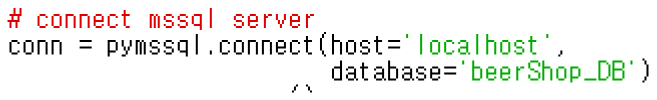
then, delete DB data and temporary DB. if there is not temporary DB, the web page can’t show any data

if query string is not only one(zero or two more), It means user enter this page for the first time or search data.

So, temporary db must be updated

# V. Outcome

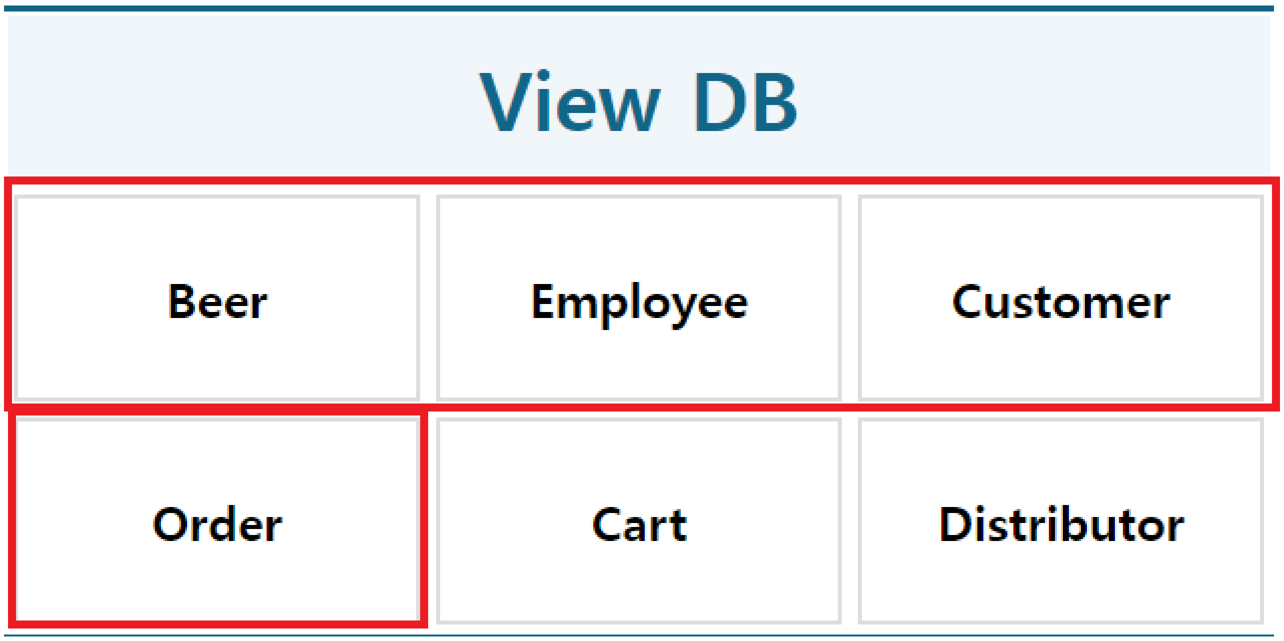
To run the server, host name should be changed if name is not ‘localhost’



```
# connect mssql server
conn = pymssql.connect(host='localhost',
                       database='beerShop_DB')
```

Then, execute ‘web page.py’. The web site URL is ‘127.0.0.1:5000’

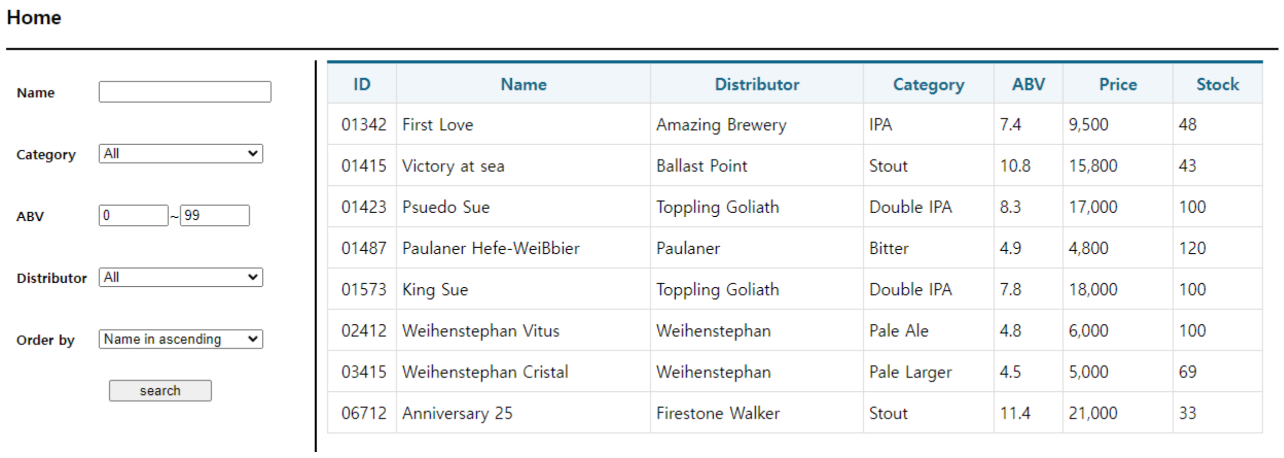
## V - 1. View DB



View DB

I couldn’t implement cart and distributor because of a lack of time

<Beer View>



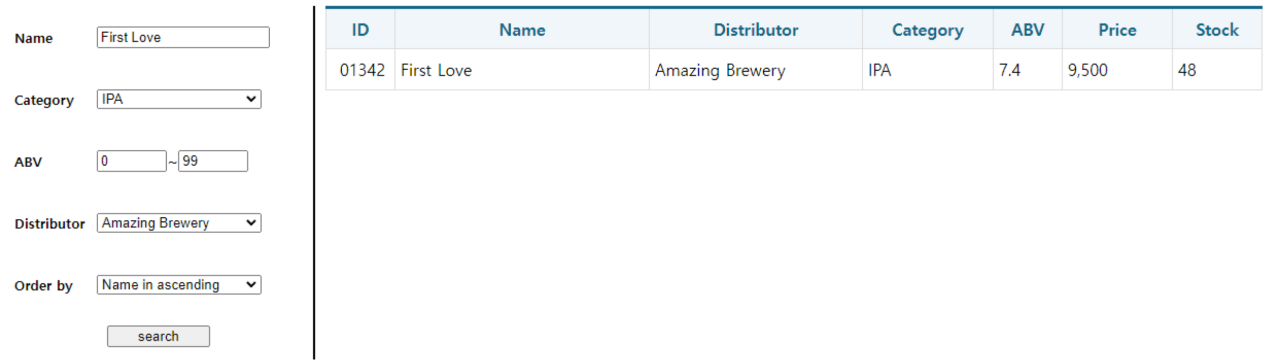
Home

| | |
|---|---|
| Name | |
| Category | All |
| ABV | 0 ~ 99 |
| Distributor | All |
| Order by | Name in ascending |

| ID | Name | Distributor | Category | ABV | Price | Stock |
|---|---|---|---|---|---|---|
| 01342 | First Love | Amazing Brewery | IPA | 7.4 | 9,500 | 48 |
| 01415 | Victory at sea | Ballast Point | Stout | 10.8 | 15,800 | 43 |
| 01423 | Psuedo Sue | Toppling Goliath | Double IPA | 8.3 | 17,000 | 100 |
| 01487 | Paulaner Hefe-WeiBbier | Paulaner | Bitter | 4.9 | 4,800 | 120 |
| 01573 | King Sue | Toppling Goliath | Double IPA | 7.8 | 18,000 | 100 |
| 02412 | Weihenstephan Vitus | Weihenstephan | Pale Ale | 4.8 | 6,000 | 100 |
| 03415 | Weihenstephan Cristal | Weihenstephan | Pale Larger | 4.5 | 5,000 | 69 |
| 06712 | Anniversary 25 | Firestone Walker | Stout | 11.4 | 21,000 | 33 |

We can search data like below

<Beer Veiw with Searching ‘First Love’>



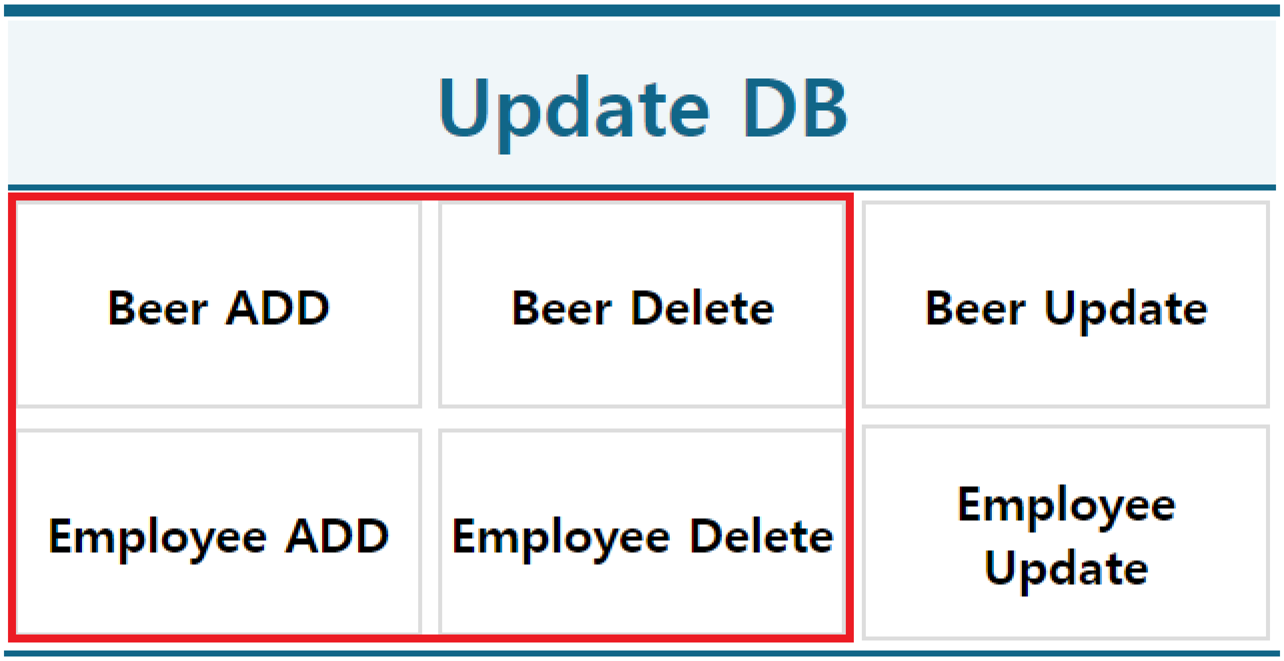
| | |
|---|---|
| Name | First Love |
| Category | IPA |
| ABV | 0 ~ 99 |
| Distributor | Amazing Brewery |
| Order by | Name in ascending |

| ID | Name | Distributor | Category | ABV | Price | Stock |
|---|---|---|---|---|---|---|
| 01342 | First Love | Amazing Brewery | IPA | 7.4 | 9,500 | 48 |

category and distributor is consist of only existing in the DB.

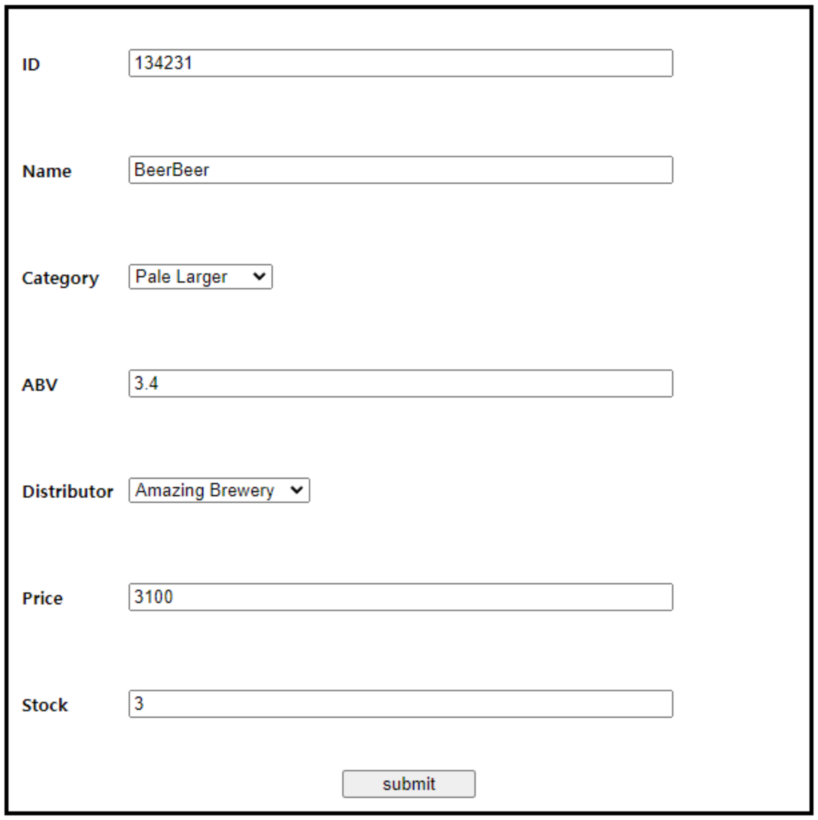
Employee, Customer, and Order View is similar with Beer View

## V - 2. Edit DB



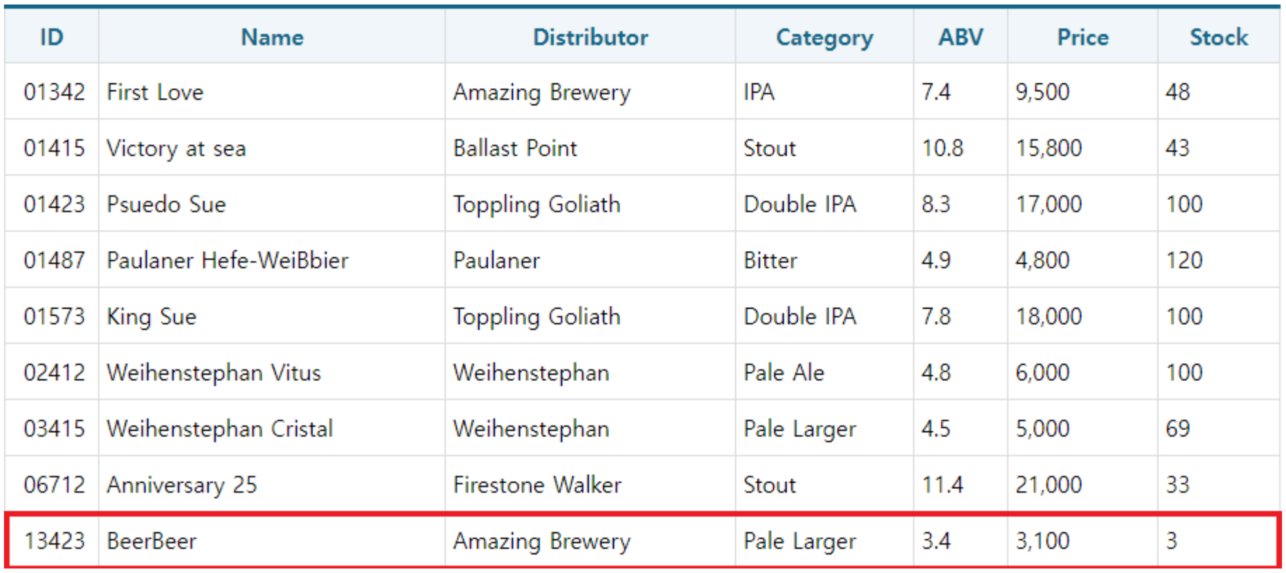
Implementation update was too difficult, So I couldn't did it

<add beer>



| | |
|---|---|
| ID | 134231 |
| Name | BeerBeer |
| Category | Pale Larger |
| ABV | 3.4 |
| Distributor | Amazing Brewery |
| Price | 3100 |
| Stock | 3 |

After editing DB, View is also updated!



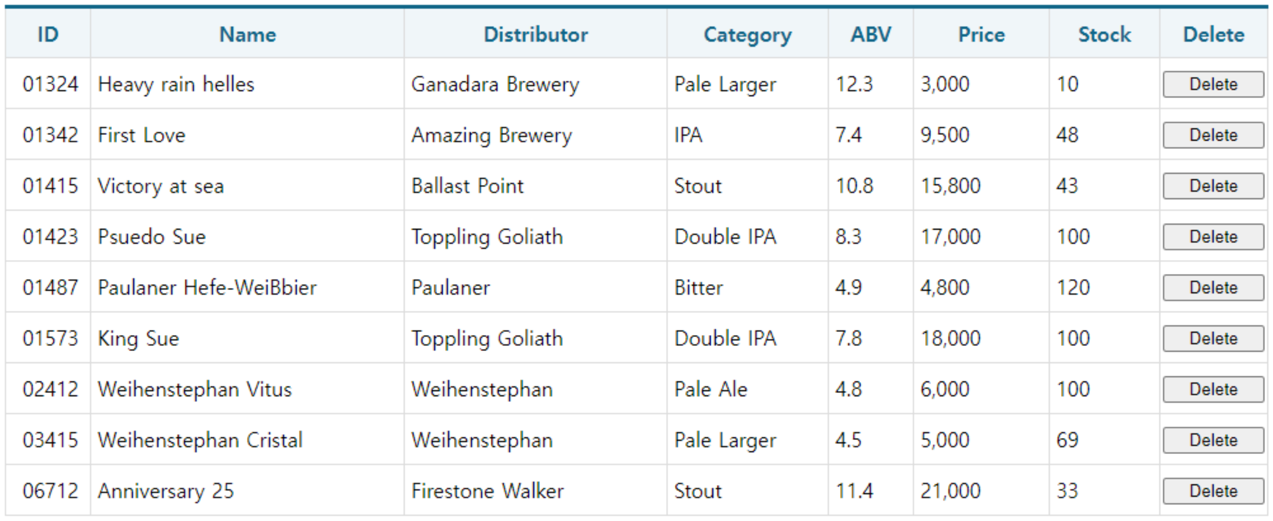
| ID | Name | Distributor | Category | ABV | Price | Stock |
|---|---|---|---|---|---|---|
| 01342 | First Love | Amazing Brewery | IPA | 7.4 | 9,500 | 48 |
| 01415 | Victory at sea | Ballast Point | Stout | 10.8 | 15,800 | 43 |
| 01423 | Psuedo Sue | Toppling Goliath | Double IPA | 8.3 | 17,000 | 100 |
| 01487 | Paulaner Hefe-WeiBbier | Paulaner | Bitter | 4.9 | 4,800 | 120 |
| 01573 | King Sue | Toppling Goliath | Double IPA | 7.8 | 18,000 | 100 |
| 02412 | Weihenstephan Vitus | Weihenstephan | Pale Ale | 4.8 | 6,000 | 100 |
| 03415 | Weihenstephan Cristal | Weihenstephan | Pale Larger | 4.5 | 5,000 | 69 |
| 06712 | Anniversary 25 | Firestone Walker | Stout | 11.4 | 21,000 | 33 |
| 13423 | BeerBeer | Amazing Brewery | Pale Larger | 3.4 | 3,100 | 3 |

add emeployee is similar too

<delete beer>

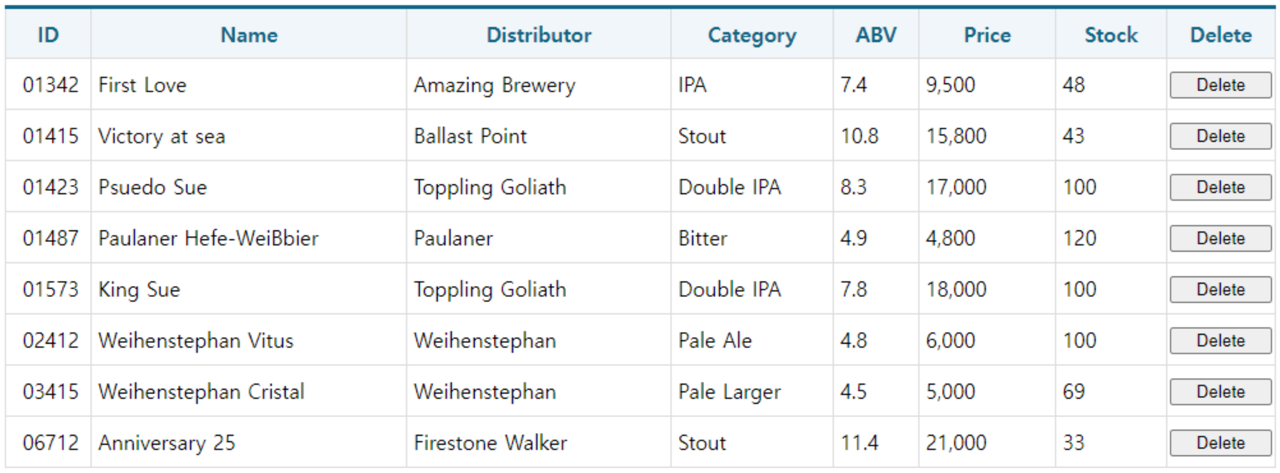
As I wrote above, Deletion work real-time when we click delete button

<before>



| ID | Name | Distributor | Category | ABV | Price | Stock | Delete |
|---|---|---|---|---|---|---|---|
| 01324 | Heavy rain helles | Ganadara Brewery | Pale Larger | 12.3 | 3,000 | 10 | Delete |
| 01342 | First Love | Amazing Brewery | IPA | 7.4 | 9,500 | 48 | Delete |
| 01415 | Victory at sea | Ballast Point | Stout | 10.8 | 15,800 | 43 | Delete |
| 01423 | Psuedo Sue | Toppling Goliath | Double IPA | 8.3 | 17,000 | 100 | Delete |
| 01487 | Paulaner Hefe-WeiBbier | Paulaner | Bitter | 4.9 | 4,800 | 120 | Delete |
| 01573 | King Sue | Toppling Goliath | Double IPA | 7.8 | 18,000 | 100 | Delete |
| 02412 | Weihenstephan Vitus | Weihenstephan | Pale Ale | 4.8 | 6,000 | 100 | Delete |
| 03415 | Weihenstephan Cristal | Weihenstephan | Pale Larger | 4.5 | 5,000 | 69 | Delete |
| 06712 | Anniversary 25 | Firestone Walker | Stout | 11.4 | 21,000 | 33 | Delete |

<after>



| ID | Name | Distributor | Category | ABV | Price | Stock | Delete |
|---|---|---|---|---|---|---|---|
| 01342 | First Love | Amazing Brewery | IPA | 7.4 | 9,500 | 48 | Delete |
| 01415 | Victory at sea | Ballast Point | Stout | 10.8 | 15,800 | 43 | Delete |
| 01423 | Psuedo Sue | Toppling Goliath | Double IPA | 8.3 | 17,000 | 100 | Delete |
| 01487 | Paulaner Hefe-WeiBbier | Paulaner | Bitter | 4.9 | 4,800 | 120 | Delete |
| 01573 | King Sue | Toppling Goliath | Double IPA | 7.8 | 18,000 | 100 | Delete |
| 02412 | Weihenstephan Vitus | Weihenstephan | Pale Ale | 4.8 | 6,000 | 100 | Delete |
| 03415 | Weihenstephan Cristal | Weihenstephan | Pale Larger | 4.5 | 5,000 | 69 | Delete |
| 06712 | Anniversary 25 | Firestone Walker | Stout | 11.4 | 21,000 | 33 | Delete |

ID 01324 beer is deleted

delete emeployee is similar too