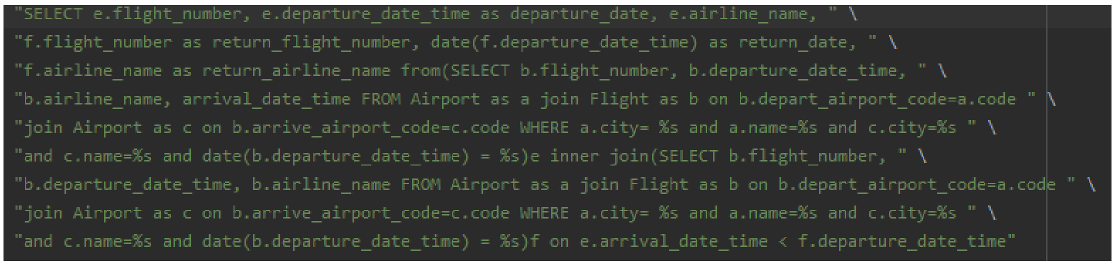
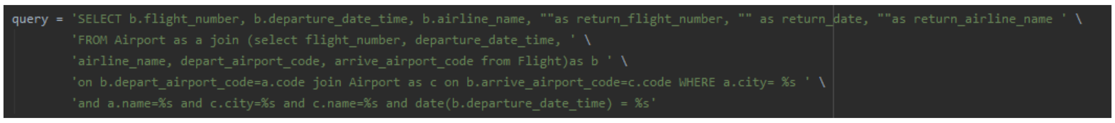
We use the following 4 queries to find round trip or single trip in both home page and customer home page.
Round trip



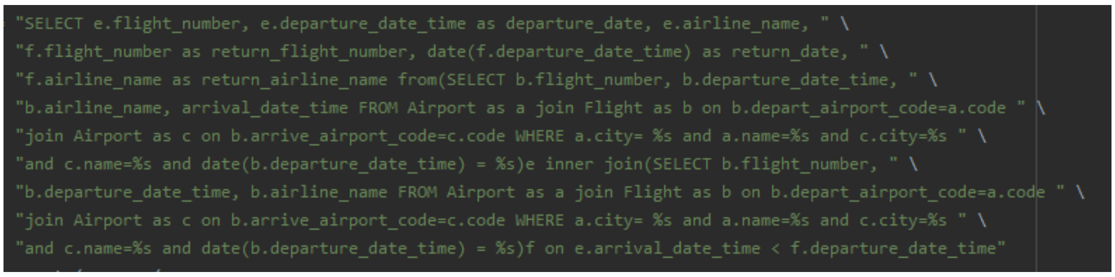
```
"SELECT e.flight_number, e.departure_date_time as departure_date, e.airline_name, " \
"f.flight_number as return_flight_number, date(f.departure_date_time) as return_date, " \
"f.airline_name as return_airline_name from(SELECT b.flight_number, b.departure_date_time, " \
"b.airline_name, arrival_date_time FROM Airport as a join Flight as b on b.depart_airport_code=a.code " \
"join Airport as c on b.arrive_airport_code=c.code WHERE a.city= %s and a.name=%s and c.city=%s " \
"and c.name=%s and date(b.departure_date_time) = %s)e inner join(SELECT b.flight_number, " \
"b.departure_date_time, b.airline_name FROM Airport as a join Flight as b on b.depart_airport_code=a.code " \
"join Airport as c on b.arrive_airport_code=c.code WHERE a.city= %s and a.name=%s and c.city=%s " \
"and c.name=%s and date(b.departure_date_time) = %s)f on e.arrival_date_time < f.departure_date_time"
```

Single trip



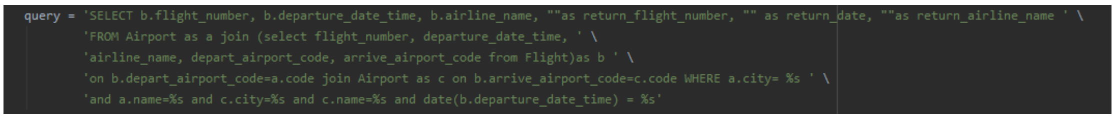
```
query = 'SELECT b.flight_number, b.departure_date_time, b.airline_name, ""as return_flight_number, "" as return_date, ""as return_airline_name ' \
        'FROM Airport as a join (select flight_number, departure_date_time, ' \
        'airline_name, depart_airport_code, arrive_airport_code from Flight)as b ' \
        'on b.depart_airport_code=a.code join Airport as c on b.arrive_airport_code=c.code WHERE a.city= %s ' \
        'and a.name=%s and c.city=%s and c.name=%s and date(b.departure_date_time) = %s'
```

Customer Flight search round trip



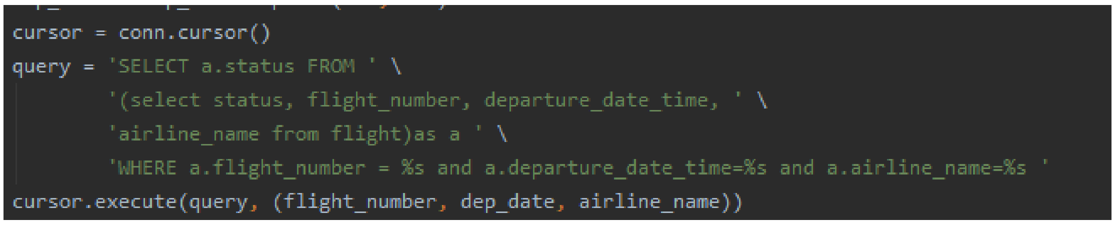
```
"SELECT e.flight_number, e.departure_date_time as departure_date, e.airline_name, " \
"f.flight_number as return_flight_number, date(f.departure_date_time) as return_date, " \
"f.airline_name as return_airline_name from(SELECT b.flight_number, b.departure_date_time, " \
"b.airline_name, arrival_date_time FROM Airport as a join Flight as b on b.depart_airport_code=a.code " \
"join Airport as c on b.arrive_airport_code=c.code WHERE a.city= %s and a.name=%s and c.city=%s " \
"and c.name=%s and date(b.departure_date_time) = %s)e inner join(SELECT b.flight_number, " \
"b.departure_date_time, b.airline_name FROM Airport as a join Flight as b on b.depart_airport_code=a.code " \
"join Airport as c on b.arrive_airport_code=c.code WHERE a.city= %s and a.name=%s and c.city=%s " \
"and c.name=%s and date(b.departure_date_time) = %s)f on e.arrival_date_time < f.departure_date_time"
```

Customer Flight search single trip



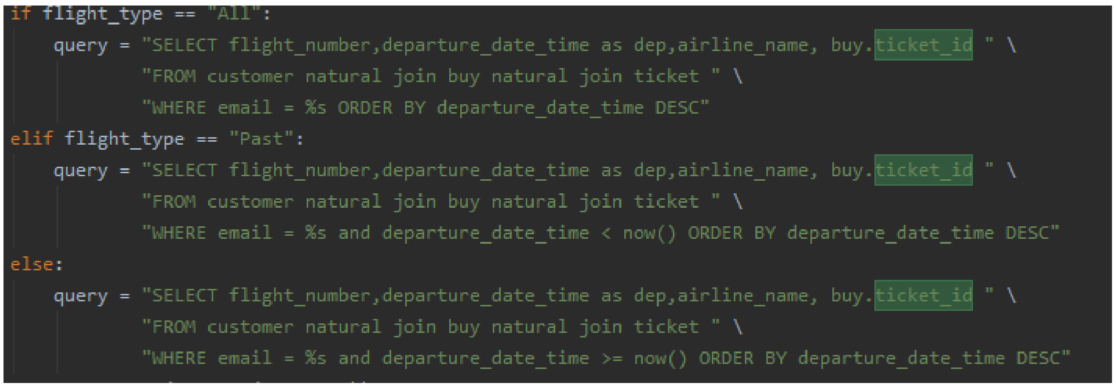
```
query = 'SELECT b.flight_number, b.departure_date_time, b.airline_name, ""as return_flight_number, "" as return_date, ""as return_airline_name ' \
        'FROM Airport as a join (select flight_number, departure_date_time, ' \
        'airline_name, depart_airport_code, arrive_airport_code from Flight)as b ' \
        'on b.depart_airport_code=a.code join Airport as c on b.arrive_airport_code=c.code WHERE a.city= %s ' \
        'and a.name=%s and c.city=%s and c.name=%s and date(b.departure_date_time) = %s'
```

A simple query is used to check the flight status with all information give by users.
Check status



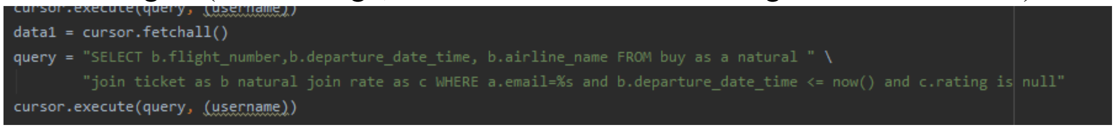
```
cursor = conn.cursor()
query = 'SELECT a.status FROM ' \
        '(select status, flight_number, departure_date_time, ' \
        'airline_name from flight)as a ' \
        'WHERE a.flight_number = %s and a.departure_date_time=%s and a.airline_name=%s '
cursor.execute(query, (flight_number, dep_date, airline_name))
```

During this part we allow users to choose from Future, pass, or All tickets they have bought. We will show the ticket ID according to current time to distinguish pass and future tickets.
View my flights (All/Future/Past)



```
if flight_type == "All":
    query = "SELECT flight_number,departure_date_time as dep,airline_name, buy.ticket_id " \
            "FROM customer natural join buy natural join ticket " \
            "WHERE email = %s ORDER BY departure_date_time DESC"
elif flight_type == "Past":
    query = "SELECT flight_number,departure_date_time as dep,airline_name, buy.ticket_id " \
            "FROM customer natural join buy natural join ticket " \
            "WHERE email = %s and departure_date_time < now() ORDER BY departure_date_time DESC"
else:
    query = "SELECT flight_number,departure_date_time as dep,airline_name, buy.ticket_id " \
            "FROM customer natural join buy natural join ticket " \
            "WHERE email = %s and departure_date_time >= now() ORDER BY departure_date_time DESC"
```

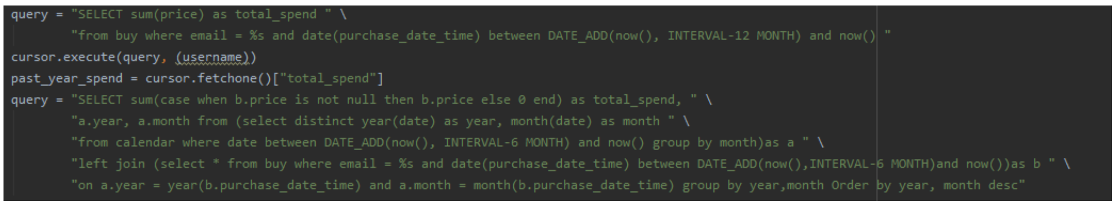
Unrated flights (in our design, we will show uncomment flights to the customer)



```
cursor.execute(query, (username))
data1 = cursor.fetchall()
query = "SELECT b.flight_number,b.departure_date_time, b.airline_name FROM buy as a natural " \
        "join ticket as b natural join rate as c WHERE a.email=%s and b.departure_date_time <= now() and c.rating is null"
cursor.execute(query, (username))
```

For past 6-month spending in customer home page we create a calendar to help find out the month wise spending, since we want to show the month even if our customer don't do any purchase.

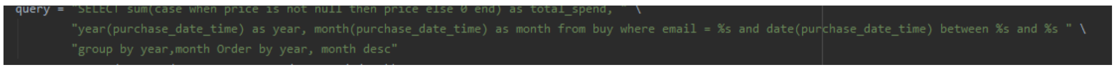
Past year spending & recent 6-month spending (Default)



```
query = "SELECT sum(price) as total_spend " \
        "from buy where email = %s and date(purchase_date_time) between DATE_ADD(now(), INTERVAL-12 MONTH) and now() "
cursor.execute(query, (username))
past_year_spend = cursor.fetchone()["total_spend"]
query = "SELECT sum(case when b.price is not null then b.price else 0 end) as total_spend, " \
        "a.year, a.month from (select distinct year(date) as year, month(date) as month " \
        "from calendar where date between DATE_ADD(now(), INTERVAL-6 MONTH) and now() group by month)as a " \
        "left join (select * from buy where email = %s and date(purchase_date_time) between DATE_ADD(now(),INTERVAL-6 MONTH)and now())as b " \
        "on a.year = year(b.purchase_date_time) and a.month = month(b.purchase_date_time) group by year,month Order by year, month desc"
```

For the month wise spending part we only output the amount moneys spend for month with purchase records. Using another function to fill out those months without purchase as 0.

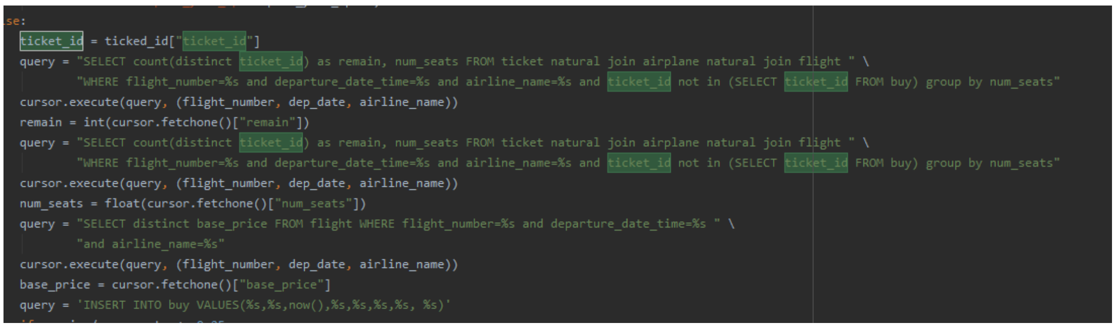
Track customer month wise spending



```
query = "SELECT sum(case when price is not null then price else 0 end) as total_spend, " \
        "year(purchase_date_time) as year, month(purchase_date_time) as month from buy where email = %s and date(purchase_date_time) between %s and %s " \
        "group by year,month Order by year, month desc"
```

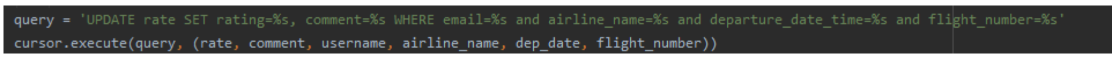
Simple insertion is used in buy ticket and add rate & comment. We also add a attribute of purchase price, in sake of easy computation of customers month wise spending.

Buy tickets



```
se:
  ticket_id = ticked_id["ticket_id"]
  query = "SELECT count(distinct ticket_id) as remain, num_seats FROM ticket natural join airplane natural join flight " \
          "WHERE flight_number=%s and departure_date_time=%s and airline_name=%s and ticket_id not in (SELECT ticket_id FROM buy) group by num_seats"
  cursor.execute(query, (flight_number, dep_date, airline_name))
  remain = int(cursor.fetchone()["remain"])
  query = "SELECT count(distinct ticket_id) as remain, num_seats FROM ticket natural join airplane natural join flight " \
          "WHERE flight_number=%s and departure_date_time=%s and airline_name=%s and ticket_id not in (SELECT ticket_id FROM buy) group by num_seats"
  cursor.execute(query, (flight_number, dep_date, airline_name))
  num_seats = float(cursor.fetchone()["num_seats"])
  query = "SELECT distinct base_price FROM flight WHERE flight_number=%s and departure_date_time=%s " \
          "and airline_name=%s"
  cursor.execute(query, (flight_number, dep_date, airline_name))
  base_price = cursor.fetchone()["base_price"]
  query = 'INSERT INTO buy VALUES(%s,%s,now(),%s,%s,%s,%s, %s)'
```

Add rate and comment

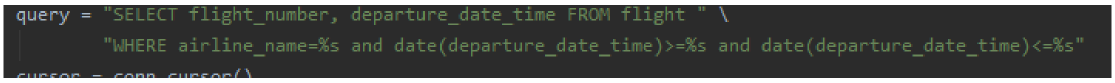


```
query = 'UPDATE rate SET rating=%s, comment=%s WHERE email=%s and airline_name=%s and departure_date_time=%s and flight_number=%s'
cursor.execute(query, (rate, comment, username, airline_name, dep_date, flight_number))
```

Staff home page:

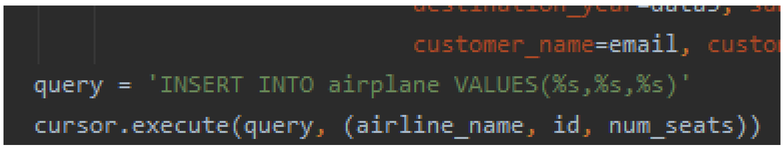
We use several updates, and insert for staffs to modify or add in flights, airports, and airplanes.

Staff home page search



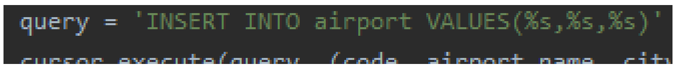
```
query = "SELECT flight_number, departure_date_time FROM flight " \
        "WHERE airline_name=%s and date(departure_date_time)>=%s and date(departure_date_time)<=%s"
```

Insert airplane



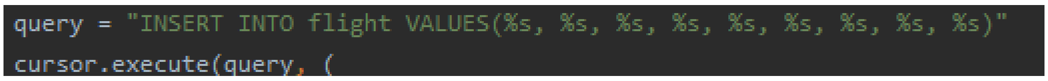
```
                              customer_name=email, custo
query = 'INSERT INTO airplane VALUES(%s,%s,%s)'
cursor.execute(query, (airline_name, id, num_seats))
```

Insert airport



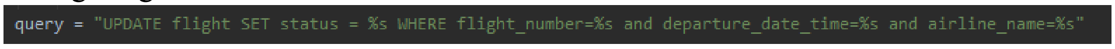
```
query = 'INSERT INTO airport VALUES(%s,%s,%s)'
```

Create new flight



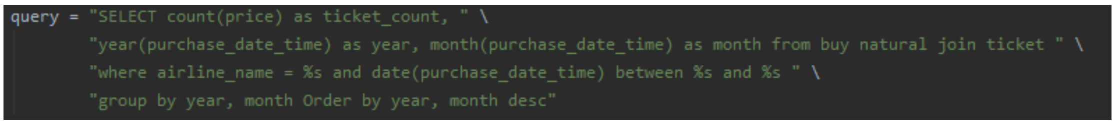
```
query = "INSERT INTO flight VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s)"
cursor.execute(query, (
```

Change flight



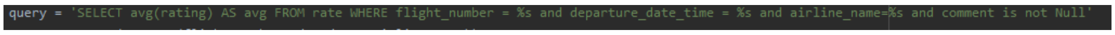
```
query = "UPDATE flight SET status = %s WHERE flight_number=%s and departure_date_time=%s and airline_name=%s"
```

Amount ticket sell



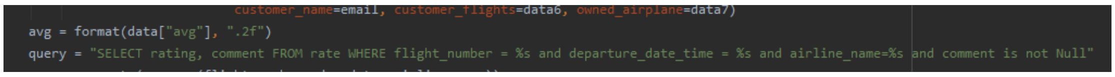
```
query = "SELECT count(price) as ticket_count, " \
        "year(purchase_date_time) as year, month(purchase_date_time) as month from buy natural join ticket " \
        "where airline_name = %s and date(purchase_date_time) between %s and %s " \
        "group by year, month Order by year, month desc"
```

Average rating



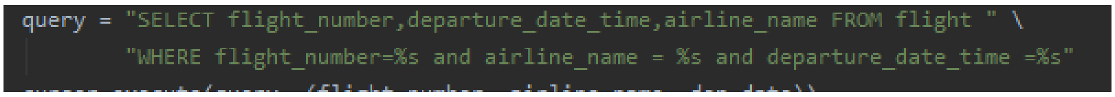
```
query = 'SELECT avg(rating) AS avg FROM rate WHERE flight_number = %s and departure_date_time = %s and airline_name=%s and comment is not Null'
```

View comment

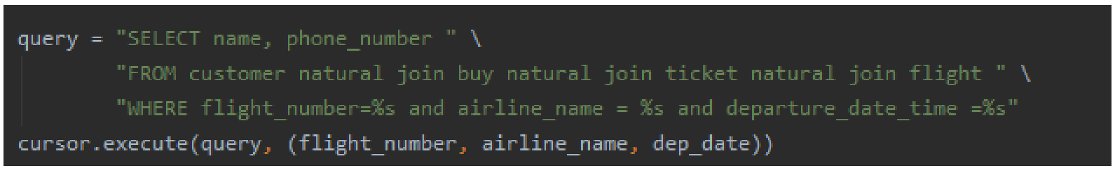


```
                         customer_name=email, customer_flights=data6, owned_airplane=data7)
avg = format(data["avg"], ".2f")
query = "SELECT rating, comment FROM rate WHERE flight_number = %s and departure_date_time = %s and airline_name=%s and comment is not Null"
```

Find customer

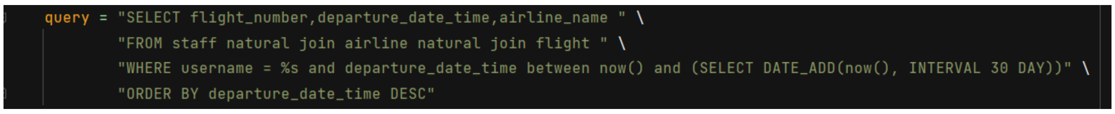


```
query = "SELECT flight_number,departure_date_time,airline_name FROM flight " \
        "WHERE flight_number=%s and airline_name = %s and departure_date_time =%s"
```



```
query = "SELECT name, phone_number " \
        "FROM customer natural join buy natural join ticket natural join flight " \
        "WHERE flight_number=%s and airline_name = %s and departure_date_time =%s"
cursor.execute(query, (flight_number, airline_name, dep_date))
```

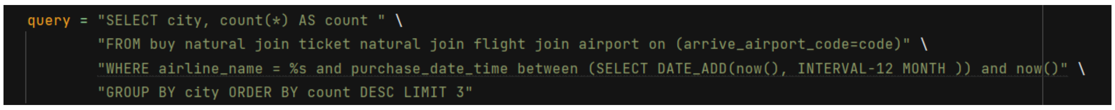
Default airline flight view (30 days)



```
query = "SELECT flight_number,departure_date_time,airline_name " \
        "FROM staff natural join airline natural join flight " \
        "WHERE username = %s and departure_date_time between now() and (SELECT DATE_ADD(now(), INTERVAL 30 DAY))" \
        "ORDER BY departure_date_time DESC"
```

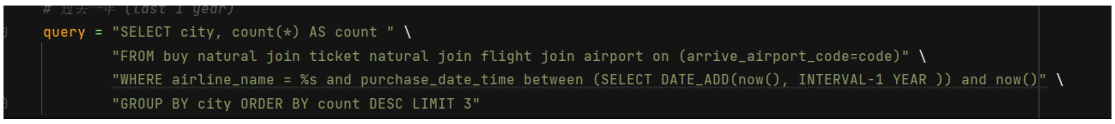
We just find the airport with most customers bought tickets.

Last 3 months top destination



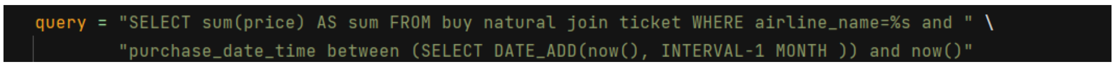
```
query = "SELECT city, count(*) AS count " \
        "FROM buy natural join ticket natural join flight join airport on (arrive_airport_code=code)" \
        "WHERE airline_name = %s and purchase_date_time between (SELECT DATE_ADD(now(), INTERVAL-12 MONTH )) and now()" \
        "GROUP BY city ORDER BY count DESC LIMIT 3"
```

Last 6 months top destination



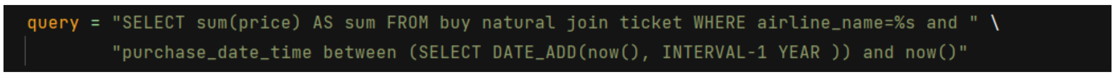
```
query = "SELECT city, count(*) AS count " \
        "FROM buy natural join ticket natural join flight join airport on (arrive_airport_code=code)" \
        "WHERE airline_name = %s and purchase_date_time between (SELECT DATE_ADD(now(), INTERVAL-1 YEAR )) and now()" \
        "GROUP BY city ORDER BY count DESC LIMIT 3"
```

For revenues we add all ticket price together to calculate the price.
Last month revenue



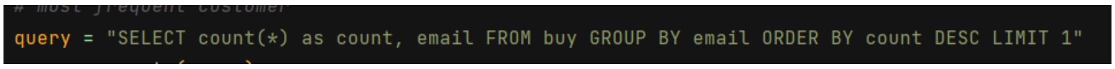
```
query = "SELECT sum(price) AS sum FROM buy natural join ticket WHERE airline_name=%s and " \
        "purchase_date_time between (SELECT DATE_ADD(now(), INTERVAL-1 MONTH )) and now()"
```

Last year revenue



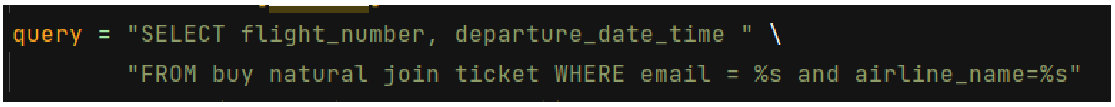
```
query = "SELECT sum(price) AS sum FROM buy natural join ticket WHERE airline_name=%s and " \
        "purchase_date_time between (SELECT DATE_ADD(now(), INTERVAL-1 YEAR )) and now()"
```

Most frequent customer (email)



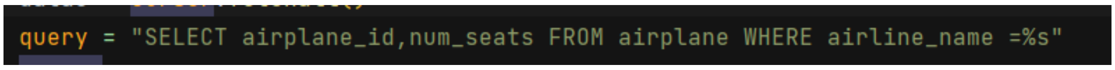
```
query = "SELECT count(*) as count, email FROM buy GROUP BY email ORDER BY count DESC LIMIT 1"
```

Flight information by the customer



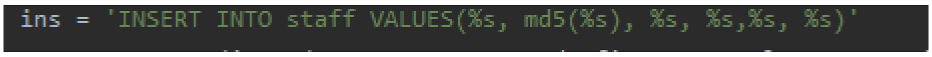
```
query = "SELECT flight_number, departure_date_time " \
        "FROM buy natural join ticket WHERE email = %s and airline_name=%s"
```

Airplane owned by the airline



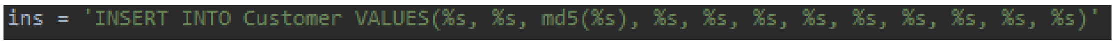
```
query = "SELECT airplane_id,num_seats FROM airplane WHERE airline_name =%s"
```

Registration:
We user md5 format to add in passwords.
Staff registration



```
ins = 'INSERT INTO staff VALUES(%s, md5(%s), %s, %s,%s, %s)'
```

Customer registration



```
ins = 'INSERT INTO Customer VALUES(%s, %s, md5(%s), %s, %s, %s, %s, %s, %s, %s, %s, %s)'
```