

# NoSQL Customer Database

Create Database:

```
> use customer
< switched to db customer
```

```
db.createCollection("Customer");
```

Insert Values Into Collection:

```
mydb> db.Customers.insertMany([{"custId":100,"accBal":2000,"accType":"a"}, {"custId":101,"accBal":5000,"accType":"b"}, {"custId":102,"accBal":7000,"accType":"c"}]);
{
  acknowledged: true,
  insertedIds: [
    '0': ObjectId('6942a8f3dc619776c91e2621'),
    '1': ObjectId('6942a8f3dc619776c91e2622'),
    '2': ObjectId('6942a8f3dc619776c91e2623')
  ]
}
mydb> db.Customers.find();
[
  {
    _id: ObjectId('6942a8f3dc619776c91e2621'),
    custId: 100,
    accBal: 2000,
    accType: 'a'
  },
  {
    _id: ObjectId('6942a8f3dc619776c91e2622'),
    custId: 101,
    accBal: 5000,
    accType: 'b'
  },
  {
    _id: ObjectId('6942a8f3dc619776c91e2623'),
    custId: 102,
    accBal: 7000,
    accType: 'c'
  }
]
```

## Queries

Write a query to display those records whose total account balance is greater than 1200 of account type ‘a’ for each Cust\_id.

```
mydb> db.Customers.aggregate([
  {$match: {accType: "a"}},
  {$group: {_id: "$custId", totalAccBal: {$sum: "$accBal"}}},
  {$match: {totalAccBal: {$gt: 1200}}}
]);
[ { _id: 100, totalAccBal: 2000 } ]
```

Determine Minimum and Maximum account balance for each customer\_id.

```
mydb> db.Customers.aggregate([{$group:{_id:"$customerId",minAccBal:{$min:"$accBal"},maxAccBal:{$max:"$accBal"}}}]);  
[  
  { _id: 100, minAccBal: 2000, maxAccBal: 2000 },  
  { _id: 101, minAccBal: 5000, maxAccBal: 5000 },  
  { _id: 102, minAccBal: 7000, maxAccBal: 7000 }  
]
```

Drop the table

```
mydb> db.Customers.drop();  
true  
"
```