MONGODB - Tuần 5

```
// Date Expression Operators
db.trips.aggregate([
  {$project:
    {_id:0, tripduration: 1,
    journeytime: {$divide: [{$subtract: ['$stop time', '$start time']}, 1000]}}}
]);

// $dateDiff
db.trips.aggregate([
  {$project:
    {_id:0, tripduration: 1,
    journeytime: {$dateDiff: {
                  startDate: '$start time',
                  endDate: '$stop time',
                  unit: 'second'}
                }}}
]);

// $month, $day, $year
db.trips.aggregate([
  {$project:
    {"start time": 1,
    "month_no": {$month: '$start time'}}}
])

// calculate age of user
db.trips.aggregate([
```

```
  {$match: {'birth year': {$ne: ""}}},
  {$project: {'birth year': 1,
     'age': {$subtract: [{$year: new Date()}, '$birth year']}
  }}
])

// $addFields
db.trips.aggregate([
  {$addFields: {'tripduration_hrs': {$divide: ["$tripduration", 60]}}}
])



db.trips.aggregate([
  {$project: {'tripduration_hrs': {$divide: ["$tripduration", 60]}}}
])

db.trips.aggregate([
  {$project: {
    _id:0, tripduration: 1, bikeid: 1, usertype: 1
  }},
  {$addFields: {
     tripduration_hrs: {$divide: ["$tripduration", 60]}
  }}
])

// $count
db.trips.aggregate([
  {$count: "total docs"}
])
```

```
db.trips.aggregate([
  {$match: { $and: [{'tripduration': {$gt: 50}}, {'tripduration': {$lte: 100}}]}},
  {$count: "number of trips between 50 and 100"}
])

// $switch condition
db.trips.aggregate([
  {$match: {'birth year': {$ne: ""}}},
  {$addFields: {'age': {$subtract: [{$year: new Date()}, '$birth year']}}},
  {$project:
     {   _id:0, 'birth year': 1, age: 1,
        'age class': {$switch: {branches: [
           {case: {$lt: ['$age', 30]}, then: 'young'},
           {case: {$gt: ['$age', 60]}, then: 'Old'}
        ], default: 'Mid'}}
     }}
])

// $sortByCount: returns the count of each group = {$group: }, {$sort:}
db.trips.aggregate([
  {$sortByCount: '$usertype'}
])

db.trips.aggregate([
  {$group: {_id: '$usertype',
            'count': {$sum: 1}}
  },
  {$sort: {count: -1}}
```

```
])

// In the trips collection what is the most common "start station name"?
db.trips.aggregate([
  {$sortByCount: '$start station name'},
  {$limit: 1}
])

// $bucket: {
//    groupBy: expression, // is like _id field in $group
//    boundaries: [lowerbound1, lowerbound2,...], //e.g [0, 100, 200]
//    default: string literal, // if documents do not fit any boundary
//    output: {
//       output1: {accumulator expression}, ...,
//       outputN: {accumulator expression}
//    }}
db.trips.aggregate([
  {$bucket: {
          groupBy: '$tripduration',
          boundaries: [ 0, 100, 1000, 10000, 1000000 ]
  }}])

db.trips.aggregate([
  {$bucket: {
    groupBy: '$tripduration',
    boundaries: [ 0, 100, 1000, 10000 ],
    default: "other",
    output: {
      "avg_duration": {$avg: '$tripduration'},
```

```
      "count": {$sum: 1}
    }}}
])

db.grades.findOne()
// $unwind: //split the array into separate documents
db.grades.aggregate([
  {$unwind: {
    path: '$scores'
  }}
])

// $out: save the result in a collection
db.grades.aggregate([
          {$match: {class_id: 339}},
           {$project: {
    "class_id": 1, "student_id": 1
  }},
  {$out: "class_id_339"}
])
db.class_id_339.find()

//
stage1 = {$match: {class_id: 339}};
stage2 = {$project: {"class_id": 1, "student_id": 1}};
stage3 = {$limit: 5};
db.grades.aggregate([stage1, stage2, stage3]);

//
```

```
stage1 = {$unwind: '$scores'};
stage2 = {$match: {'scores.type': 'homework'}};
stage3 = {$group: {_id: '$scores.type',
                   avg_score: {$avg: '$scores.score'},
                   max_score: {$max: '$scores.score'}}};
db.grades.aggregate([stage1, stage2, stage3]);
```

Bài tập: (sử dụng aggregate framework)

Database: sales.json

1. Hiển thị ngày bán hàng đầu tiên và ngày bán hàng cuối cùng
2. Hiển thị ngày bán hàng gần nhất có số lượng items bán được nhiều nhất
3. Hiển thị tên sản phẩm và số lượng đã bán của sản phẩm có số lượng bán được nhiều nhất
4. Hiển thị 'storeLocation', số lượng khách hàng ('no_of_customers') theo từng storeLocation và từng 'purchaseMethod',sắp xếp theo storeLocation và purchaseMethod theo bảng chữ cái từ A - Z
5. Thống kê số lượng khách hàng theo độ tuổi như sau: 15-29, 30-44, 45-59, 60-74, 75+
6. Hiển thị số lượng khách hàng (no_of_customers), độ tuổi trung bình (avg_age), mức độ hài lòng trung bình (avg_satisfaction) của khách hàng theo từng khu vực. Làm tròn kết quả: avg_age: làm tròn lên, avg_satisfaction: làm tròn sau dấu phẩy 1 chữ số. Sắp xếp kết quả theo no_of_customers từ cao đến thấp
7. Hiển thị số lượng khách hàng, độ tuổi trung bình, mức độ hài lòng trung bình của khách hàng đã mua hàng ở cửa hàng 'New York' theo từng nhóm giới tính, làm tròn kết quả như câu trên
8. Hiển thị tất cả các distinct tags có trong sales collection
9. Hiển thị 'saleDate', 'items.name', 'items.price', 'items.quantity', và thêm 1 field 'items.revenue' với 'items.revenue' = 'items.price' * 'items.quantity', sort kết quả theo 'saleDate' từ cao đến thấp và chỉ hiển thị 2 kết quả đầu tiên

10. Tính tổng doanh thu (totalSalesAmount) theo từng 'items.name'. Ví dụ: binder có totalSalesAmount: 511644.57
11. Tính tổng doanh thu theo từng năm
12. Tổng số lượng đã bán được và tổng doanh thu của sản phẩm 'laptop' tại cửa hàng New York?