몽고DB실습 _AI(Y-A,kim)

1 . 내용입력

```
db.Product.insertMany([
  { Name: "notebook", Price: 200, Category: "material" },
  { Name: "pencil", Price: 80, Category: "material" },
  { Name: "salad", Price: 220, Category: "food" },
  { Name: "others", Price: 20, Category: "material" },
  { Name: "bread", Price: 100, Category: "food" }
]);
```

1. 전체 출력

```
db.Product.find()
```

2. id 와 가격을 출력해 보자.

```
db.Product.aggregate([
 { $project: { _id: 1, Price: 1 } }
  ]);

  db.Product.aggregate([
    {$project :{Price :1}}
    ]);

  가격만 출력 해보자.
  db.Product.aggregate([
    {$project :{_id:0, Price :1}}
    ]);
```

==========

3. Name 과 가격을 출력 해보자.

```
db.Product.aggregate([
 {$project :{Name :1, Price :1 }}
 ]);
```

4. 가격과 카테고리만 출력 해 보자. 단 가격별 오름차순으로 정렬을 한다.

```
db.Product.aggregate([
{$project :{_id :0, Price :1, Category:1 }},
{ $sort: { Price: 1 }    }
]);
```

5. 목록과 가격만을 출력하되 목록을 내림차순으로 정렬을 해보자.

```
db.Product.aggregate([
{$project :{_id :0, Name :1, Price:1 }},
{ $sort: {Name: -1 }    }
]);
```

==========

6. 이름만 출력하되 별칭을 주자

```
db.Product.aggregate([
 { $project: { _id: 0, alias_name: "$Name" } }
  ]);
```

7.Name 을 목록, Price 가격, Category 는 타입으로 별칭을 주고 출력하자

```
db.Product.aggregate([
 { $project: { _id: 0, 목록: "$Name", 가격 :"$Price",타입:"$Category" } }
  ]);
```

==========

8.상품의 목록(Name)과 inc_price 라는 별칭을 주고 Price 에 100 을 더하자

```
db.Product.aggregate([
      { $project: { _id: 0, Name: 1,
    inc_price: { $add: ["$Price", 100] } } }
  ]);
```

9. 카테고리로 그룹화를 한 다음 최대 가격을 출력 해보자.

```
db.Product.aggregate([
```

```
{ $group: { _id:   "$Category", max_price: { $max: "$Price" } }   }
 ]);
```

10 . 카테고리로 그룹화를 한 다음 최소 가격을 출력 해보자.

```
 db.Product.aggregate([
{ $group: { _id:   "$Category", min_price: { $min: "$Price" } }   }
 ]);
```

11. 상품 목록을 출력하고 가격의 합과 가격의 평균과 목록의 개수를 구하자.

```
db.Product.aggregate([
   {$group:   {_id:"$Category", sum :{$sum :"$Price"   },
                                avg :{$avg :"$Price"   },
                                count: { $sum: 1 }}}

 ]);
```

12.상품목록을 출력하고 그룹화 한 다음 개수를 구해보자

```
db.Product.aggregate([
   {$project: { "Category": 1,count:{ $literal: 1 }}},
   {$group:   {_id:"$Category", count:{$sum:"$count"}}}
 ]);

db.Product.aggregate([
   {$project: { "Category": 1,count:{ $literal: 1 }}}
 ]);
```

==========

==========$match 활용해보자.

13. Name 에서 bread 를 찾아 출력 하자.

```
db.Product.aggregate([
{$project: { _id: 0, Name:1 }},
{$match : {Name :"bread"}}
]);
```

14. Category 에서 food 만 출력 해보자.

```
db.Product.aggregate([
  {$match : {Category :"food"}}
]);
```

15. Category 의 food 의 가격의 최대값, 최소값, 총합, 평균, 개수를 출력 하자.

```
db.Product.aggregate([
  {$match : {Category :"food"}},
  {$group:  {_id:"$Category",   max :{$max :"$Price"   },
                                 min :{$min :"$Price"   },
                                sum :{$sum :"$Price"   },
                                avg :{$avg :"$Price"   },
                                count: { $sum: 1 }}}
   ]);
```

- Q1) 모든 제품 중에서 가장 높은 가격을 찾아서 리턴하자.
- Q2) 가격이 100보다 큰 상품의 개수를 출력 하자.
- Q3) name이 문자 S로 시작하는 제품의 총 가격을 계산하자.
- Q4) Category의 meterial의 평균 가격을 출력하자.
- Q5) material 있는 모든 제품의 총 가격을 계산하되 가격이 50보다 큰 제품만 포함한다. 또한 제품 가격이 150보다 큰 경우 총 가격에 10% 할인을 포함한다.

  ➔ $match, $group $sum $cond($gt $multiply) 사용