1. Get the first name, last name and job title of all employees.

**db.Cust\_Emp\_Pay\_Offices.find({}, {firstName: 1, lastName: 1, jobTitle: 1, \_id: 0});**

2. Sort the products by the productCode column in ascending order. Display the product names in uppercase.

**db.Prod\_ProdLine\_OrderDet\_Orders\_Cust.find({},{\_id: 0, productName: {$toUpper: "$productName"}}).sort({productCode: 1});**

3. Find the distinct territory of employees whose last name is “Patterson” and “Bondur”.

**db.Cust\_Emp\_Pay\_Offices.distinct('territory', {lastName: {$in: ['Patterson', 'Bondur']}});**

4. Find employees who have their extensions beginning with 2.

**db.Cust\_Emp\_Pay\_Offices.find({extension: {$regex: '^x2.'}});**

5. Retrieve results where the reportsTo column is NULL

**db.Cust\_Emp\_Pay\_Offices.find({reportsTo : null});**

6. Find all employees whose job titles are Sales Rep. Concat the first and last names. Display “JobTitle” as “Designation”.

**db.Cust\_Emp\_Pay\_Offices.find({jobTitle : "Sales Rep"}, {Designation: {$concat: ["$firstName"," ", "$lastName"]}, JobTitle : "$jobTitle", \_id : 0});**

7. Find products that have the string ‘Ford‘ or ‘Mercedes’.

**db.Prod\_ProdLine\_OrderDet\_Orders\_Cust.find({$or: [{productName : /Ford/}, {productName: /Mercedes/}]});**

8. Find customers who live in Singapore or France.

**db.Cust\_Emp\_Pay\_Offices.find({country : {$in: ["Singapore","France"]}});**

9. Return employees with office code less than or equal to 4

**db.** **Cust\_Emp\_Pay\_Offices.find({officeCode : {$lte : 4}});**

10. Use the DISTINCT clause to select unique product lines.

**db.Prod\_ProdLine\_OrderDet\_Orders\_Cust.distinct('productLine');**

11. Write SQL query that returns the customers who are located in California, USA, and have the credit limit greater than 100K.

**db.Prod\_ProdLine\_OrderDet\_Orders\_Cust.find({$and: [{state: 'CA'}, {creditLimit: {$gt: 100000}}]});**

12. Report the total payments by Date.

**db.Cust\_Emp\_Pay\_Offices.aggregate([{$group: {\_id: "$paymentDate", sum\_val:{$sum:"$amount"}}}]);**

BONUS:

1. Find sales price of the product whose code is S10\_1678 that is less than the manufacturer’s suggested retail price (MSRP) for that product.

**db.Prod\_ProdLine\_OrderDet\_Orders\_Cust.find({$and: [{productCode : "S10\_1678"}, {$expr: {$lt:["$priceEach", "$MSRP"]}}]}, {productCode: 1, priceEach : 1, MSRP: 1, \_id: 0})**

2. Identify the product with highest sales in terms of quantity and revenue.

**db.Prod\_ProdLine\_OrderDet\_Orders\_Cust.aggregate([{$project: {product: '$productLine', quantity: '$quantityOrdered', revenue:{$multiply:["$priceEach", "$quantityOrdered"]}}}, {$sort: {revenue: -1}}, {$limit: 1]);**

**db.Prod\_ProdLine\_OrderDet\_Orders\_Cust.aggregate( [**   
 **{**   
 **$addFields: {**   
 **'totalGain': {$multiply: ['$priceEach','$quantityOrdered']}**   
 **}**   
 **},**   
 **{**   
 **$group : {**   
 **\_id : '$productCode',**   
 **totalOrdered: {$sum: '$quantityOrdered'},**   
 **revenue: { $sum: '$totalGain' }**   
 **}**   
 **},**   
 **{**   
 **$sort: {revenue: -1}**   
 **},**   
 **{**   
 **$limit: 1**   
 **},**   
 **{**   
 **$project: {quantityOrdered: '$totalOrdered' , revenue: '$revenue'}**   
 **}**   
**] )**

3. List the products ordered on a Friday.

**db.Prod\_ProdLine\_OrderDet\_Orders\_Cust.aggregate([{$project: {\_id: 1, productLine: '$productLine', day\_of\_week\_of\_order: {$dayOfWeek: '$orderDate'}}}, {$match: {day\_of\_week\_of\_order: 6}}]);**