

**Licenciatura em Engenharia Informática**

**Processamento Estruturado de informação**

**BikeOnTrack (Parte II)**

Docentes: Bruno Oliveira

Marco Gomes

Alunos: João Marques 8170200

Pedro Pinto 8170262

Ricardo Ferreira 8170279

**Índice**

[**Referências** 2](#_Toc535010549)

[**Introdução** 3](#_Toc535010550)

[**Alterações** 3](#_Toc535010551)

[**Vocabulário** 4](#_Toc535010552)

[**XSD** 4](#_Toc535010553)

[**Loja** 4](#_Toc535010554)

[**Exercise** 5](#_Toc535010555)

[**Sales** 6](#_Toc535010556)

[**Cliente** 6](#_Toc535010557)

[**SaleLine** 7](#_Toc535010558)

[**Produto** 7](#_Toc535010559)

[**CurrencyDetails** 8](#_Toc535010560)

[**ReceiptInfo** 8](#_Toc535010561)

[**CommonTypes** 9](#_Toc535010562)

[**XML(Exemplo)** 10](#_Toc535010563)

[**MongoDB** 11](#_Toc535010564)

[**Querys** 12](#_Toc535010565)

[**Por venda** 12](#_Toc535010566)

[**Por exercicio** 12](#_Toc535010567)

[**Por loja** 14](#_Toc535010568)

[**Exportação para XML** 14](#_Toc535010569)

[**Conclusão** 14](#_Toc535010570)

# **Referências**

<https://moodle.estg.ipp.pt>

<https://docs.mongodb.com/manual/>

# **Introdução**

Nesta parte do trabalho, pretendemos demonstrar os nossos conhecimentos acerca dos conceitos de armazenamento de documentos e as tecnologias atuais para a estruturação de informação, o reconhecimento e a compreensão da semântica e da sintaxe da notação JSON (Javascript Object Notation) bem como mecanismos para a sua preservação e exploração utilizando coleções de documentos e, por último, a implementação de web APIs para suportar o processo de preservação, processamento e exploração de documentos.

Com estes conceitos, pretendemos agora, nesta parte do trabalho, que os dados extraídos diretamente dos sistemas de faturação de cada loja possam ser alvo de um processo de processamento e preservação utilizando um repositório específico para que possa ser utilizada a geração automática de documentos XML suportados pelo vocabulário desenvolvido.

Logo, resumidamente, pretendemos armazenar os dados todos no MongoDB, desenvolver um programa em linguagem Java para suportar a geração dos documentos XML (que representam os dados de auditoria de cada loja) de forma automatizada, e usar o serviço REST capaz de expor a funcionalidade de exportação dos dados para XML.

# **Alterações**

* Decidimos remover os atributos “ref”, porque averiguamos que era uma má prática e que deviamos evitar o seu uso. *“Definir elementos root apenas para elementos que podem representar o root do documento (evitar referências – atributo ref). Caso contrário criar tipos globais com ComplexType e SimpleType;”*
* Alteração da estrutura de maneira em que a loja seja o ponto mais alto, em vez do exercício como tínhamos desenvolvido na primeira entrega. Depois de refletirmos a primeira entrega, chegamos á conclusão que devíamos mudar o nosso tipo de pensamento á cerca do assunto. Basicamente o nosso raciocínio não estava mal, mas não era a melhor opção. Com a estrutura deste enunciado, ainda veio reforçar mais essa decisão, passando assim a ser “Loja” o ponto mais alto do vocabulário e não “Exercicio”.
  + - * Loja
        + Exercício

Vendas

Cliente

Linhas de Venda

Produto

Moeda

Data do exercício

* Enriquecemos o vocabulário com os dados pedidos no enunciado, como mostraremos a seguir.

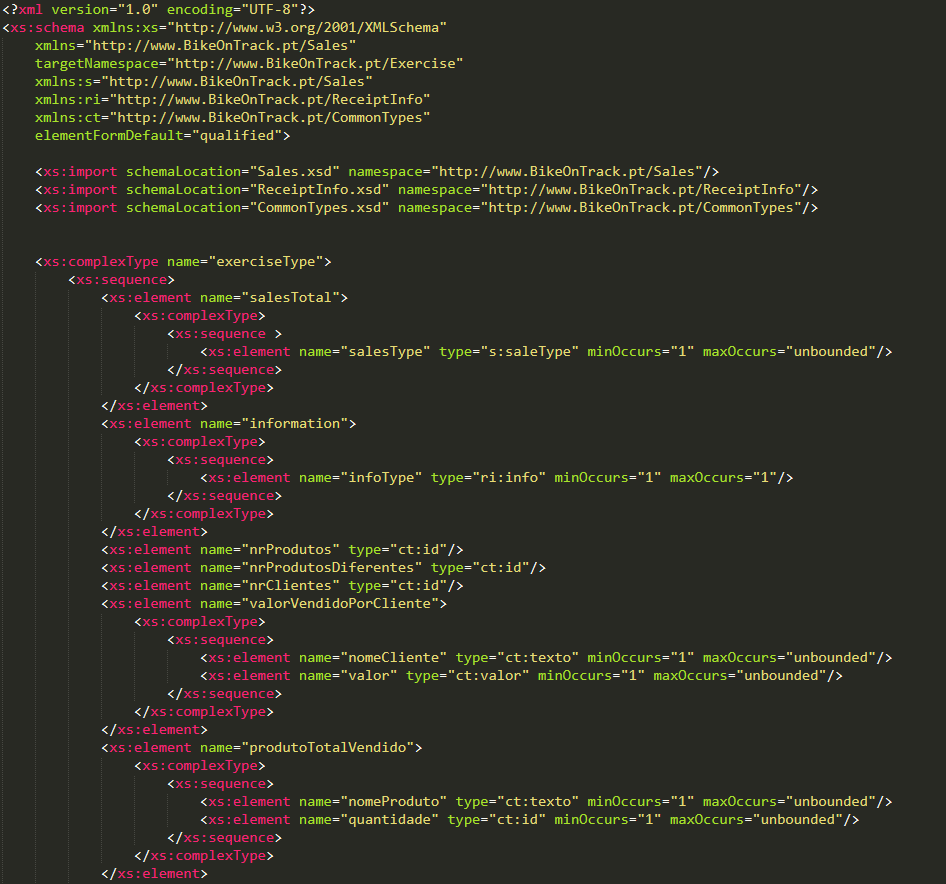
# **Vocabulário**

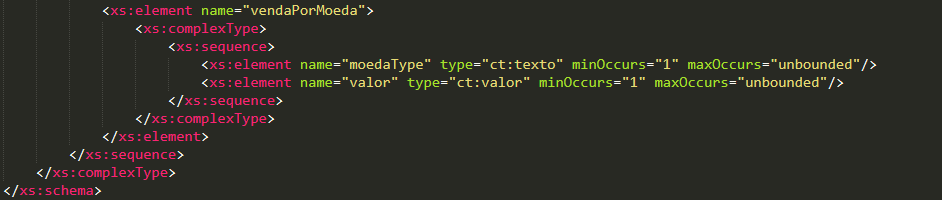
## **XSD**

### **Loja**

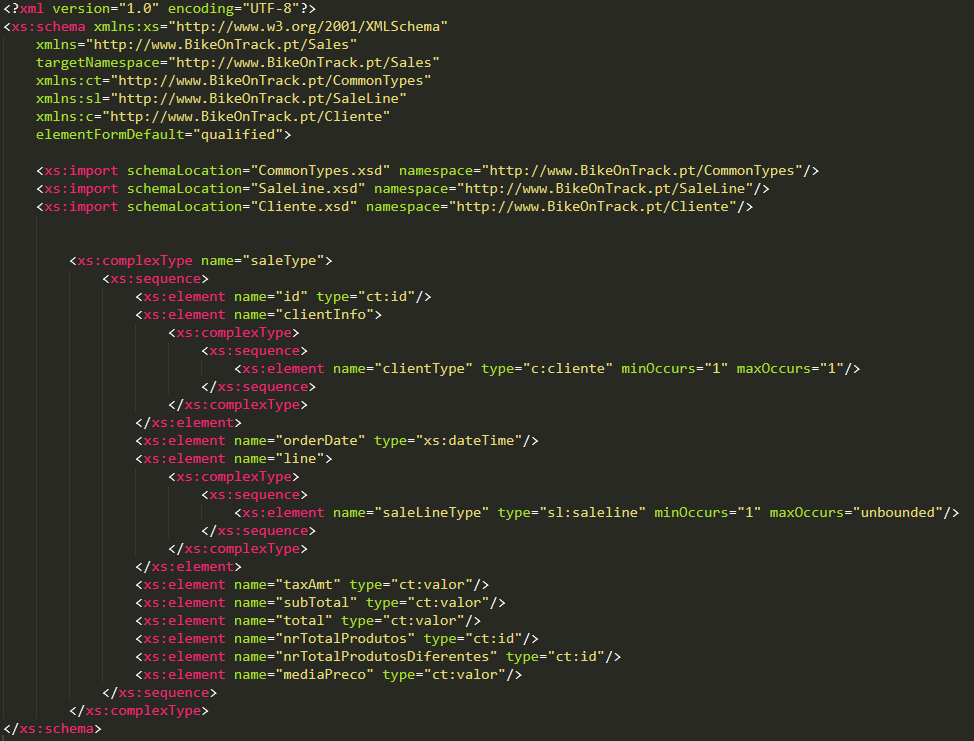


### **Exercise**





### **Sales**



### **Cliente**



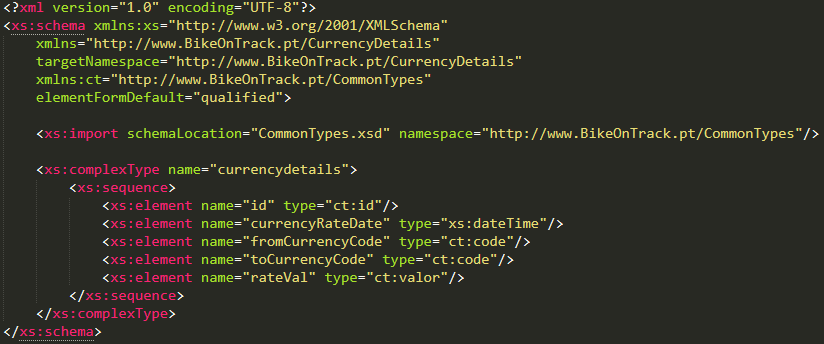
### **SaleLine**



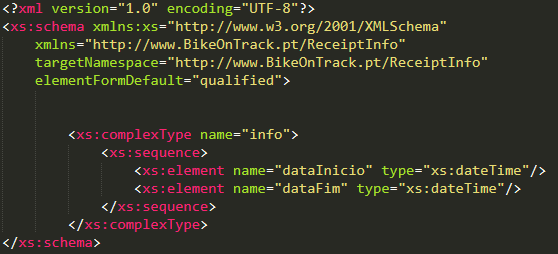
### **Produto**



### **CurrencyDetails**

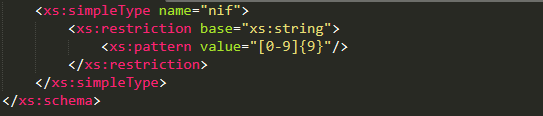


### **ReceiptInfo**

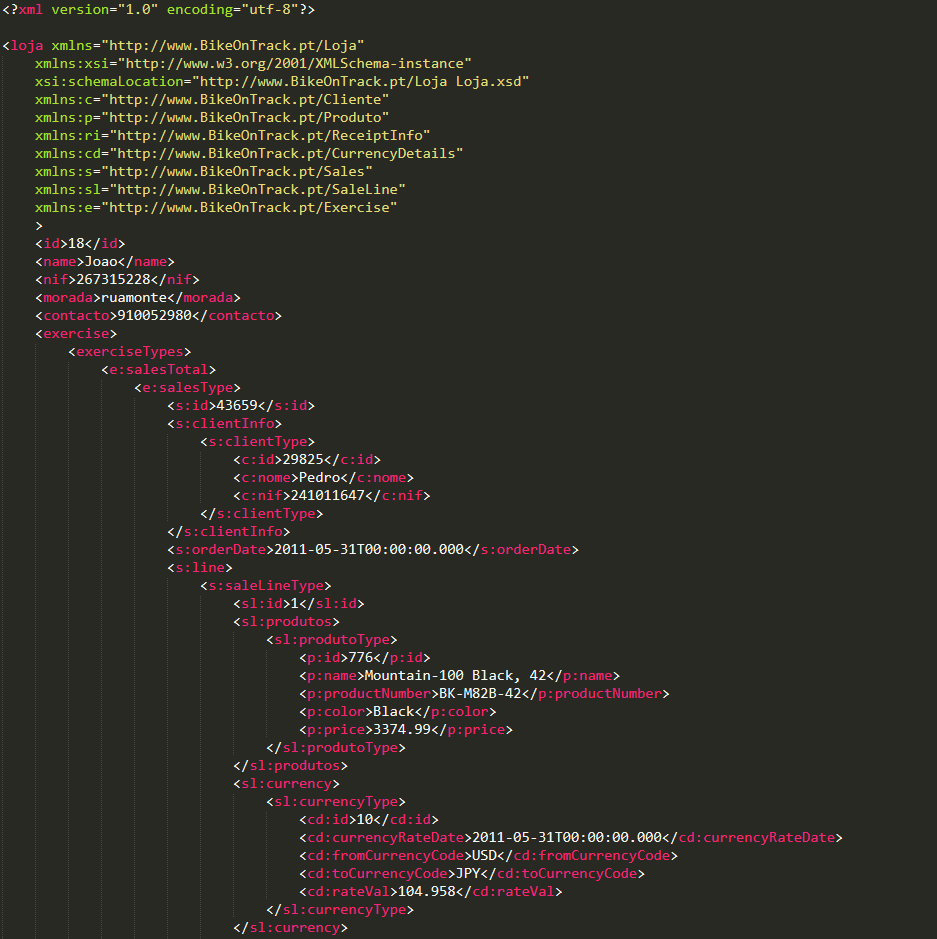


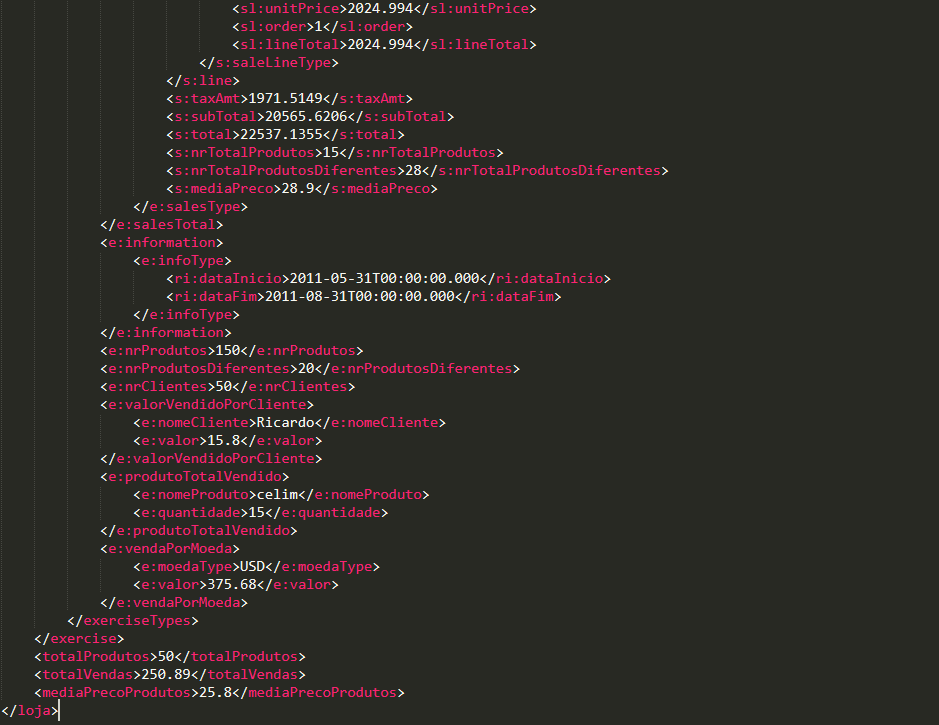
### **CommonTypes**





## **XML(Exemplo)**





# **MongoDB**

**1º]** Diretório do mongodb

cd C:\Program Files\MongoDB\Server\4.0\bin

**2º]** Importar todos documentos para a base de dados “PEIproject”

mongoimport --db PEIproject --collection SalesDetails --file ~/… / SalesDetails.csv --headerline

mongoimport --db PEIproject --collection CurrencyDetails --file ~/… / CurrencyDetails.csv --headerline

mongoimport --db PEIproject --collection ProductDetailsv2 --file ~/… / ProductDetailsv2.csv --headerline

mongoimport --db PEIproject --collection ProductListPriceHistoryV2 --file ~/… / ProductListPriceHistoryV2.csv --headerline

**3º]** show dbs 🡪 mostra todas as base de dados no mongo

Use PEIproject 🡪 para dizermos que é esta base de dados que queremos utilizar

**4º]** show collections 🡪 para vermos as collections criadas anteriormente

**5º]** Está pronto para começar a fazer as consultas.

# **Querys**

### **Por venda**

Total de produtos vendidos

String query1="[{$match:{StoreName: {$eq: '"+store+"'}}},{$group: {\_id:\"$ReceiptID\",Total:{$sum:\"$Quantity\"}}},{$sort:{Total:1}}]";

Esta query agrupa ($group) todos os receiptID iguais e soma ($sum) as quantidades ({$sum:\"$Quantity\"}).

Total de produtos diferentes

String query2="[{$match:{StoreName: {$eq: '"+store+"'}}},{$group:{\_id:\"$ReceiptID\",Total:{$sum:1}}},{$sort:{Total:1}}]";

Esta query agrupa ($group) todos ReceiptID iguais e soma ($sum) 1 por cada ReceiptID ($sum:1).

Média de preço dos produtos vendidos

String query3="[{$match:{StoreName: {$eq: '"+store+"'}}},{$group: {\_id:\"$ReceiptID\",MediaPrecoProduto:{$avg:\"$UnitPrice\"}}},{$sort:{MediaPrecoProduto:1}}]";

Esta query agrupa ($group) todos os ReceiptID e calcula a média ($avg) do preço do produto por unidade ({$avg:\"$UnitPrice\"}).

### **Por exercicio**

Total de produtos vendidos

String query4="[{$match:{StoreName: {$eq: '"+store+"'},OrderDate:{$gte:'"+data+"-01 00:00:00.000',$lte:'"+data+"-31 00:00:00.000'}}},{$group: {\_id:\"TotalProdutos\",Total:{$sum:\"$Quantity\"}}},{$sort:{Total:1}}]";

Total de produtos diferentes

String query5="[{$match:{StoreName: {$eq: '"+store+"'},OrderDate:{$gte:'"+data+"-01 00:00:00.000',$lte:'"+data+"-31 00:00:00.000'}}},{$group:{\_id:\"$ProductID\"}},{$group:{\_id:\"TotalProdutosDiferentes\",Total:{$sum:1}}},{$sort:{Total:1}}]";

Total de Clientes diferentes

String query6="[{$match:{StoreName: {$eq: '"+store+"'},OrderDate:{$gte:'"+data+"-01 00:00:00.000',$lte:'"+data+"-31 00:00:00.000'}}}, {$group:{\_id:\"$Customer\"}},{$group:{\_id:\"NumerosClientesDiferentes\",Total:{$sum:1}}},{$sort:{Total:1}}]";

Valor vendido por Cliente

String query7="[{$match:{StoreName: {$eq: '"+store+"'},OrderDate:{$gte:'"+data+"-01 00:00:00.000',$lte:'"+data+"-31 00:00:00.000'}}},{$group: {\_id:\"$Customer\",TotalVendidoCliente:{$sum:\"$LineTotal\"}}},{$sort:{TotalVendidoCliente:-1}}]";

Total de unidades vendidas por produto

String query8="[{$match:{StoreName: {$eq: '"+store+"'},OrderDate:{$gte:'"+data+"-01 00:00:00.000',$lte:'"+data+"-31 00:00:00.000'}}},{$group: {\_id:\"$ProductID\",TotalProdutosVendidos:{$sum:\"$Quantity\"}}},{$sort:{TotalProdutosVendidos:-1}}]";

Valor vendido por moeda

String query9="[{$match:{StoreName: {$eq: '"+store+"'},OrderDate:{$gte:'"+data+"-01 00:00:00.000',$lte:'"+data+"-31 00:00:00.000'}}},{$group: {\_id:\"$CurrencyRateID\",TotalVendidoMoeda:{$sum:\"$LineTotal\"}}}]";

### **Por loja**

Valor total de produtos por Loja

String query10="[{$match:{StoreName: {$eq: '"+store+"'}}},{$group: {\_id:\"$ReceiptID\",TotalProdutos:{$sum:\"$Quantity\"}}},{$sort:{TotalProdutos:1}}]";

Total de vendas por Loja

String query11="[{$match:{StoreName: {$eq: '"+store+"'}}},{$group: {\_id:\"$ReceiptID\",Total:{$sum:\"$LineTotal\"}}},{$group:{\_id:VendasTotais, Resultado:{$sum:\"$Total\"}}},{$sort:{Resultado:1}}]";

Valor médio do preço dos produtos por Loja

String query12="[{$match:{StoreName: {$eq: '"+store+"'}}},{$group: {\_id:\"$StoreName\",ValorMedioProduto:{$avg:\"$UnitPrice\"}}}]";

# **Exportação para XML**

Muito teoricamente, derivado do facto de que ainda não tivemos maneira de testar, o que pretendemos fazer é agregar todos os dados do mongoDB, de modo a que com a nossa REST API e com a classe XSLTransformer disponibilizada pelos docentes, criar um documento xsl de modo a transformar esses dados do mongoDB para os nossos XML’s.

# **Conclusão**

Cumprimos com todos objetivos, apesar de ainda não termos implementado a parte de exportação para XML, mostramos em cima como pretendemos fazê-lo. Ainda fizemos alterações ao vocabulário que entregamos na primeira entrega.

Para terminar, concluímos que nesta parte do trabalho aprendemos a trabalhar com o mongoDB, a armazenar dados na base de dados e posteriormente manipulá-los através de querys. Tivemos também um primeiro contacto com o serviço REST, apesar de nesta entrega ainda não trabalharmos muito com ele.