**Hands On 2: Write queries on stock table using Query Methods**   
  
**Stock.java**

package com.cognizant.ormlearn.model;  
  
import jakarta.persistence.\*;  
import java.math.BigDecimal;  
import java.util.Date;  
  
@Entity  
@Table(name = "stock")  
public class Stock {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private int stId;  
  
 @Column(name = "st\_code")  
 private String stCode;  
  
 @Column(name = "st\_date")  
 @Temporal(TemporalType.*DATE*)  
 private Date stDate;  
  
 @Column(name = "st\_open")  
 private BigDecimal stOpen;  
  
 @Column(name = "st\_close")  
 private BigDecimal stClose;  
  
 @Column(name = "st\_volume")  
 private BigDecimal stVolume;  
  
 *// Getters and Setters* public int getStId() {  
 return stId;  
 }  
  
 public void setStId(int stId) {  
 this.stId = stId;  
 }  
  
 public String getStCode() {  
 return stCode;  
 }  
  
 public void setStCode(String stCode) {  
 this.stCode = stCode;  
 }  
  
 public Date getStDate() {  
 return stDate;  
 }  
  
 public void setStDate(Date stDate) {  
 this.stDate = stDate;  
 }  
  
 public BigDecimal getStOpen() {  
 return stOpen;  
 }  
  
 public void setStOpen(BigDecimal stOpen) {  
 this.stOpen = stOpen;  
 }  
  
 public BigDecimal getStClose() {  
 return stClose;  
 }  
  
 public void setStClose(BigDecimal stClose) {  
 this.stClose = stClose;  
 }  
  
 public BigDecimal getStVolume() {  
 return stVolume;  
 }  
  
 public void setStVolume(BigDecimal stVolume) {  
 this.stVolume = stVolume;  
 }  
  
 @Override  
 public String toString() {  
 return "Stock{" +  
 "stId=" + stId +  
 ", stCode='" + stCode + '\'' +  
 ", stDate=" + stDate +  
 ", stOpen=" + stOpen +  
 ", stClose=" + stClose +  
 ", stVolume=" + stVolume +  
 '}';  
 }  
}

**StockRepository.java**

package com.cognizant.ormlearn.repository;  
  
import com.cognizant.ormlearn.model.Stock;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
import java.math.BigDecimal;  
import java.time.LocalDate;  
import java.util.List;  
  
@Repository  
public interface StockRepository extends JpaRepository<Stock, Integer> {  
List<Stock> findByStCodeAndStDateBetween(String code, LocalDate start, LocalDate end);

List<Stock> findByStCodeAndStCloseGreaterThan(String code, BigDecimal price);  
List<Stock> findTop3ByOrderByStVolumeDesc();  
List<Stock> findTop3ByStCodeOrderByStCloseAsc(String code);  
}

**OrmlearnApplication.java**

package com.cognizant.ormlearn;  
  
import com.cognizant.ormlearn.model.Stock;  
import com.cognizant.ormlearn.repository.StockRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.boot.CommandLineRunner;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
import java.math.BigDecimal;  
import java.time.LocalDate;  
import java.util.List;  
  
@SpringBootApplication  
public class OrmlearnApplication implements CommandLineRunner {  
  
 @Autowired  
 private StockRepository stockRepository;  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(OrmlearnApplication.class, args);  
 }  
  
 @Override  
 public void run(String... args) {  
 getFacebookStocksInSep2019();  
 getGoogleStocksGreaterThan1250();  
 getTop3HighestVolume();  
 getNetflixLowest3Close();  
 }  
  
 private void getFacebookStocksInSep2019() {  
 System.*out*.println("\nFB Stocks in Sep 2019:");  
 List<Stock> stocks = stockRepository.findByStCodeAndStDateBetween(  
 "FB", LocalDate.*of*(2019, 9, 1), LocalDate.*of*(2019, 9, 30));  
 stocks.forEach(this::printStock);  
 }  
  
 private void getGoogleStocksGreaterThan1250() {  
 System.*out*.println("\nGOOGL Stocks with close > 1250:");  
 List<Stock> stocks = stockRepository.findByStCodeAndStCloseGreaterThan(  
 "GOOGL", new BigDecimal("1250"));  
 stocks.forEach(this::printStock);  
 }  
  
 private void getTop3HighestVolume() {  
 System.*out*.println("\nTop 3 stocks by volume:");  
 List<Stock> stocks = stockRepository.findTop3ByOrderByStVolumeDesc();  
 stocks.forEach(this::printStock);  
 }  
  
 private void getNetflixLowest3Close() {  
 System.*out*.println("\nNetflix - 3 lowest closing prices:");  
 List<Stock> stocks = stockRepository.findTop3ByStCodeOrderByStCloseAsc("NFLX");  
 stocks.forEach(this::printStock);  
 }  
  
 private void printStock(Stock s) {  
 System.*out*.println(s.getStCode() + " " + s.getStDate() + " " + s.getStOpen() + " " + s.getStClose() + " " + s.getStVolume());  
 }  
}

**Output**



