

# Esper – podstawy

Nasze warsztaty dotyczące narzędzia Esper będą oparte na strumieniu danych dotyczących rynków finansowych – kursów akcji – NASDAQ oraz NYSE (*Nowojorska Giełda Papierów Wartościowych*).

## Zapoznanie się z projektem „startowym”

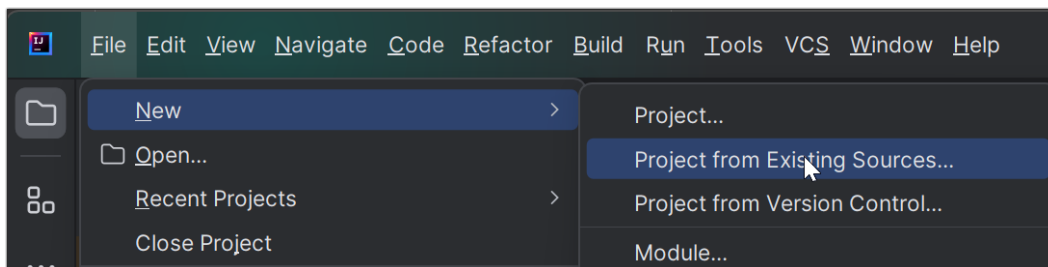
1. Pobierz wersję początkową naszego projektu

```
git clone https://github.com/BigDataStreamProcessing/esper-podstawy.git
```

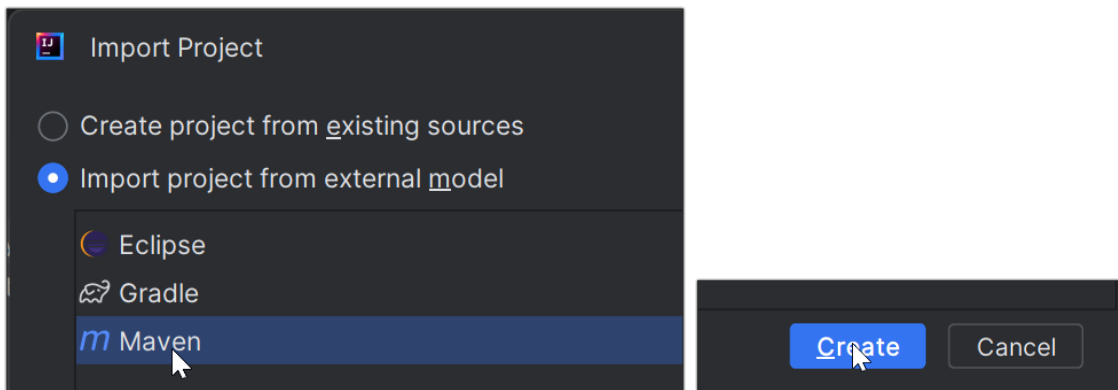
2. Przejdź do katalogu projektu, a następnie usuń powiązanie z projektem zewnętrznym

```
cd esper-podstawy  
git remote rm origin
```

3. Otwórz środowisko *IntelliJ IDEA Community Edition*, a następnie zaimportuj pobrany projekt
  - a. *File -> New -> Project from Existing Sources...*

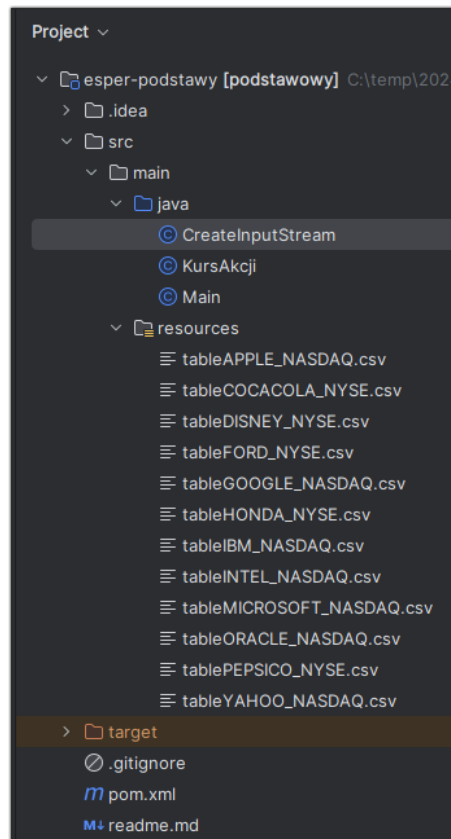


- b. Wybierz katalog *esper-podstawy*
- c. Zaznacz *Import project from external model: Maven*, a następnie wybierz przycisk *Create* (jeśli nie masz dostępnej pozycji *Maven* – doinstaluj stosowną wtyczkę w *Settings -> Plugins*)



- d. Resztę ustawień pozostaw domyślną i dokończ importowanie projektu

## 4. Obejrzyj zawartość projektu.



- a. W katalogu resources znajdziesz dane dotyczące historycznych notowań poszczególnych spółek giełdowych
- b. W katalogu src znajdziesz zdefiniowane trzy klasy
  - i. Klasa KursAkcji będzie wykorzystana przez nas jako typ dla zdarzeń strumienia wejściowego

```

4      public class KursAkcji {
5          private String spolka;
6          private String market;
7          private Date data;
8          private Float kursOtwarcia;
9          private Float wartoscMax;
10         private Float wartoscMin;
11         private Float kursZamknienia;
12         private Float obrot;

```

- ii. Klasa CreateInputStream będzie odpowiadała za tworzenie strumienia wejściowego. Zawiera ona kilka istotnych fragmentów:
  1. Stałe określające zakres dat dla wejściowego strumienia akcji

```

20     private static final String DATA_ROZPOCZECIA = "2001-09-05";
    1 usage
21     private static final String DATA_ZAKONCZENIA = "2001-09-20";

```

2. Metodę `generuj()`, która zawiera między innymi:
  - a. inicjalizację tablicy z informacjami o wykorzystywanych plikach z kursami akcji, oraz

```

28  public void generuj() throws IOException {
29      tablicaInformacjiOPlikach[0] = new InformacjeOPliku(
30          nazwaPliku: "tableAPPLE_NASDAQ.csv", nazwaSpolki: "Apple", nazwaMarketu: "NASDAQ");
31      tablicaInformacjiOPlikach[1] = new InformacjeOPliku(
32          nazwaPliku: "tableCOCACOLA_NYSE.csv", nazwaSpolki: "CocaCola", nazwaMarketu: "NYSE");
33      tablicaInformacjiOPlikach[2] = new InformacjeOPliku(
34          nazwaPliku: "tableDISNEY_NYSE.csv", nazwaSpolki: "Disney", nazwaMarketu: "NYSE");
35      tablicaInformacjiOPlikach[3] = new InformacjeOPliku(
36          nazwaPliku: "tableFORD_NYSE.csv", nazwaSpolki: "Ford", nazwaMarketu: "NYSE");

```

- b. tworzenie kolejnych obiektów klasy `KursAkcji`.

```

130      if ((dataNotowania != null) && (dataNotowania.equals(iteratorDaty))) {
131          // Tworzenie obiektu notowania
132          if (linie[i] != null) {
133              splitResult = linie[i].split(regex: ",");
134              KursAkcji kurs = new KursAkcji(
135                  tablicaInformacjiOPlikach[i].getNazwaSpolki(),
136                  tablicaInformacjiOPlikach[i].getNazwaMarketu(),
137                  dataNotowania,
138                  Float.valueOf(splitResult[1].trim()),
139                  Float.valueOf(splitResult[2].trim()),
140                  Float.valueOf(splitResult[3].trim()),
141                  Float.valueOf(splitResult[4].trim()),
142                  Float.valueOf(splitResult[5].trim()));
143              System.out.println(kurs);
144          }
145      }

```

- c. Obiekty te są tworzone we wnętrzu zagnieżdżonych pętli (pierwsza iteruje po datach, druga iteruje po plikach z kursami akcji)

```

103      while ((iteratorDaty.compareTo(dataZakonczenia) <= 0)
104          && (liczbaBledow < MAX_LICZBA_BLEDOW)) {
105
106          for (int i = 0; i < LICZBA_PLIKOW; i++) {
107              try {
108                  Date dataNotowania = null;
109
110                  if (linie[i] != null) {
111                      dataNotowania = wyodrebnijDate(linie[i]);

```

5. Uruchom obecną postać programu. W tym celu:

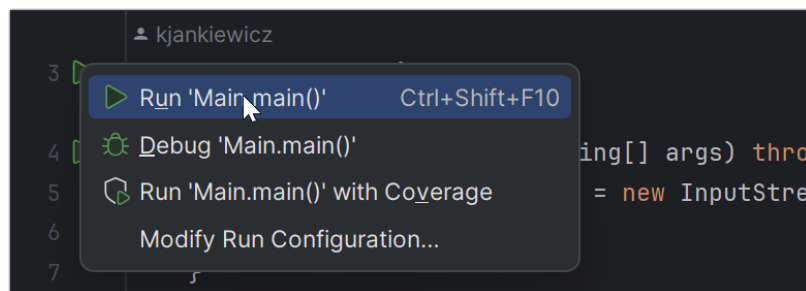
a. Przejdź do klasy Main

```

3  public class Main {
    public static void main(String[] args) throws IOException {
        InputStream inputStream = new InputStream();
        inputStream.generuj();
    }
}

```

b. Zawiera ona w tej chwili, w metodzie main, jedynie utworzenie obiektu klasy InputStream i uruchomienie metody generuj. Uruchom projekt.



c. Zobacz wynik działania metody generuj – w aktualnej formie wypisuje ona zawartość tworzonych obiektów (reprezentacji zdarzeń) na konsoli

```

C:\Users\kjkiewicz\jdk\temurin-19.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.2\lib\idea_rt.jar=51184:C:\Program Files\
KursAkcji [spolka=Apple, market=NASDAQ, data=2001-09-05, kursOtwarcia=18.24, wartoscMax=18.95, wartoscMin=18.12, kursZamknienia=18.55, obrot=1.28592E7]
KursAkcji [spolka=CocaCola, market=NYSE, data=2001-09-05, kursOtwarcia=48.8, wartoscMax=50.7, wartoscMin=48.8, kursZamknienia=50.45, obrot=6899200.0]
KursAkcji [spolka=Disney, market=NYSE, data=2001-09-05, kursOtwarcia=25.46, wartoscMax=25.84, wartoscMin=25.1, kursZamknienia=25.36, obrot=4642100.0]
KursAkcji [spolka=Ford, market=NYSE, data=2001-09-05, kursOtwarcia=20.03, wartoscMax=20.33, wartoscMin=19.82, kursZamknienia=19.97, obrot=5149200.0]
KursAkcji [spolka=Honda, market=NYSE, data=2001-09-05, kursOtwarcia=73.5, wartoscMax=73.79, wartoscMin=72.5, kursZamknienia=73.37, obrot=27600.0]

. . . . .
KursAkcji [spolka=Disney, market=NYSE, data=2001-09-20, kursOtwarcia=17.9, wartoscMax=18.35, wartoscMin=15.5, kursZamknienia=16.98, obrot=8.10676E7]
KursAkcji [spolka=Ford, market=NYSE, data=2001-09-20, kursOtwarcia=16.11, wartoscMax=16.61, wartoscMin=15.46, kursZamknienia=15.49, obrot=1.1257E7]
KursAkcji [spolka=Honda, market=NYSE, data=2001-09-20, kursOtwarcia=61.99, wartoscMax=62.0, wartoscMin=57.51, kursZamknienia=58.15, obrot=96000.0]
KursAkcji [spolka=IBM, market=NASDAQ, data=2001-09-20, kursOtwarcia=94.1, wartoscMax=95.75, wartoscMin=92.85, kursZamknienia=93.4, obrot=1.51829E7]
KursAkcji [spolka=Intel, market=NASDAQ, data=2001-09-20, kursOtwarcia=21.46, wartoscMax=22.27, wartoscMin=20.5, kursZamknienia=20.67, obrot=7.03388E7]
KursAkcji [spolka=Microsoft, market=NASDAQ, data=2001-09-20, kursOtwarcia=52.35, wartoscMax=52.61, wartoscMin=50.67, kursZamknienia=50.76, obrot=1.179832E8]
KursAkcji [spolka=Oracle, market=NASDAQ, data=2001-09-20, kursOtwarcia=10.79, wartoscMax=11.54, wartoscMin=10.74, kursZamknienia=11.31, obrot=5.72201E7]
KursAkcji [spolka=PepsiCo, market=NYSE, data=2001-09-20, kursOtwarcia=49.01, wartoscMax=49.28, wartoscMin=48.05, kursZamknienia=48.76, obrot=1.35873E7]
KursAkcji [spolka=Yahoo, market=NASDAQ, data=2001-09-20, kursOtwarcia=9.95, wartoscMax=10.34, wartoscMin=9.81, kursZamknienia=9.97, obrot=1.61032E7]

Process finished with exit code 0

```

## Stworzenie strumienia zdarzeń

6. Aby rozpocząć pracę ze środowiskiem przetwarzania strumieni danych *Esper*, musimy dodać odpowiednie biblioteki do pliku `pom.xml`.

7. Dodaj dwie własności do wnętrza elementu `<properties>`

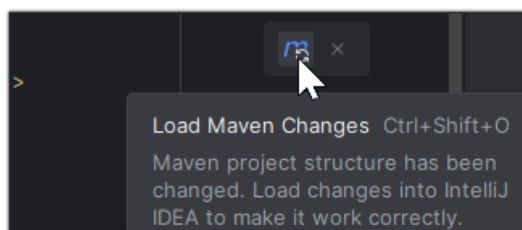
```
<esper.version>8.9.0</esper.version>
<slf4j.version>2.0.12</slf4j.version>
```

8. Wstaw poniższy fragment do pliku `pom.xml` po elemencie `<properties>`

```
<dependencies>
  <dependency>
    <groupId>com.espertech</groupId>
    <artifactId>esper-common</artifactId>
    <version>${esper.version}</version>
  </dependency>
  <dependency>
    <groupId>com.espertech</groupId>
    <artifactId>esper-compiler</artifactId>
    <version>${esper.version}</version>
  </dependency>
  <dependency>
    <groupId>com.espertech</groupId>
    <artifactId>esper-runtime</artifactId>
    <version>${esper.version}</version>
  </dependency>

  <dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-nop</artifactId>
    <version>${slf4j.version}</version>
  </dependency>
</dependencies>
```

9. Następnie zaaplikuj zmiany.



10. Czas na zmiany w klasie `CreateInputStream`. Zadaniem znajdującej się w niej metody `generuj` jest tworzenie strumienia zdarzeń. Będzie ona generowała ten strumień do określonego silnika zdarzeń – obiektu klasy `EPEventService`. Dlatego:

- a. Zmodyfikuj nagłówek metody `generuj`

```
public void generuj(EPEventService eventService) throws IOException {
```

- b. Dodaj brakujący import.

```
import com.espertech.esper.runtime.client.EPEventService;
```

- c. Popraw zawartość wnętrza pętli **zamieniając polecenie wyświetlania zawartości obiektu na polecenie umieszczające obiekt w strumieniu silnika zdarzeń.**

```
144         Float.valueOf(splitResult[5].trim());
145         eventService.sendEventBean(kurs,kurs.getClass().getName());|
146     }
147 }
```

11. Przejdź do definicji klasy `Main`

- a. Na początku metody `main` dodaj

```
Configuration configuration = new Configuration();
configuration.getCommon().addEventType(KursAkcji.class);
EPRuntime epRuntime = EPRuntimeProvider.getDefaultRuntime(configuration);
```

- b. Dodaj brakujące importy

```
import com.espertech.esper.common.client.configuration.Configuration;
import com.espertech.esper.runtime.client.EPRuntime;
import com.espertech.esper.runtime.client.EPRuntimeProvider;
```

- a. Popraw wywołanie metody `generuj`.

```
inputStream.generuj(epRuntime.getEventService());
```

12. Klasa `Main` powinna wyglądać następująco

13. Sprawdź czy aplikacja się uruchamia (konsola powinna być teraz pusta)

```
1  import com.espertech.esper.common.client.configuration.Configuration;
2  import com.espertech.esper.runtime.client.EPRuntime;
3  import com.espertech.esper.runtime.client.EPRuntimeProvider;
4
5  import java.io.IOException;
6
7  * kjankiewicz *
8  public class Main {
9      * kjankiewicz *
10     public static void main(String[] args) throws IOException {
11         Configuration configuration = new Configuration();
12         configuration.getCommon().addEventType(KursAkcji.class);
13         EPRuntime epRuntime = EPRuntimeProvider.getDefaultRuntime(configuration);
14
15         CreateInputStream inputStream = new CreateInputStream();
16         inputStream.generuj(epRuntime.getEventService());
17     }
18 }
```

## Definiowanie relacji

Strumienie zdarzeń przetwarzane są przez zapytania tworzące relacje, których zawartość jest zmienna w czasie. Aby silnik przetwarzał określone zapytanie należy je w jego wnętrzu zarejestrować. Rejestracja zapytań z reguły odbywa się przed etapem zasilania silnika strumieniem zdarzeń wejściowych.

14. We wnętrzu klasy Main dodaj funkcję pomocniczą do rejestrowania zapytań.

```
public static EPDeployment compileAndDeploy(EPRuntime epRuntime, String epl) {
    EPDeploymentService deploymentService = epRuntime.getDeploymentService();

    CompilerArguments args =
        new CompilerArguments(epRuntime.getConfigurationDeepCopy());
    EPDeployment deployment;
    try {
        EPCompiled epCompiled = EPCompilerProvider.getCompiler().compile(epl, args);
        deployment = deploymentService.deploy(epCompiled);
    } catch (EPCompileException | EPDeployException e) {
        throw new RuntimeException(e);
    }

    return deployment;
}
```

15. Uzupełnij importy.

```
import com.espertech.esper.common.client.EPCompiled;
import com.espertech.esper.compiler.client.CompilerArguments;
import com.espertech.esper.compiler.client.EPCompileException;
import com.espertech.esper.compiler.client.EPCompilerProvider;
import com.espertech.esper.runtime.client.*;
```

16. Następnie korzystając ze stworzonej funkcji, zaraz po tworzeniu obiektu `EPRuntime epRuntime`, utwórz obiekt zapytania. W poniższym kodzie wykorzystano własność *Block text* dostępną od wersji Java 15. Jeśli nie możesz z niej skorzystać popraw poniższy fragment kodu tradycyjnie składając ciągi znaków z wielu fragmentów za pomocą operatora „+”.

```
EPDeployment deployment = compileAndDeploy(epRuntime, ""
    select istream spolka as X, kursOtwarcia as Y
    from KursAkcji#length(3);""");
```

Zapytanie opiera się na oknie o długości trzech zdarzeń. Generuje ono w wyniku zarówno strumień zdarzeń wstawianych jak i zdarzeń usuwanych. Postać zdarzeń w strumieniach wynikowych jest inna niż zdarzeń w strumieniu wejściowym.

## Podłączanie się do strumienia wynikowego relacji

Aby podłączyć się do zapytania należy stworzyć klasę, której obiekty będą odbiorcami wyników. Można w tym celu wykorzystać klasę implementującą interfejs `UpdateListener`.

17. Utwórz nową klasę o nazwie `ProstyListener`. Dodaj deklarację implementacji interfejsu `UpdateListener`. Dołóż brakujący import.

```
1  import com.espertech.esper.runtime.client.UpdateListener;
2
3  public class ProstyListener implements UpdateListener {
4
5  }
```

18. Następnie dodaj brakujące, z punktu widzenia implementowanego interfejsu, metody.



```
nts UpdateListener {
    Implement methods
    Make 'ProstyListener' abstract >
}

@Override
public void update(EventBean[] eventBe
}
```

19. Jako treść dodanej metody `update` wprowadź poniższą zawartość.

```
if (newEvents != null) {
    for (EventBean newEvent : newEvents) {
        System.out.println("ISTREAM : " + newEvent.getUnderlying());
    }
}
if (oldEvents != null) {
    for (EventBean oldEvent : oldEvents) {
        System.out.println("RSTREAM : " + oldEvent.getUnderlying());
    }
}
```

20. Zmień nazwy parametrów metody, tak aby pasowały do powyższej zawartości (`newEvents` jako nazwa pierwszego parametru, a `oldEvents` drugiego).



21. Porównaj swoją klasę z postacią klasy przedstawioną poniżej i ewentualnie dokonaj niezbędnych poprawek. Zapisz zmiany.

```

1  import com.espertech.esper.common.client.EventBean;
2  import com.espertech.esper.runtime.client.EPRuntime;
3  import com.espertech.esper.runtime.client.EPStatement;
4  import com.espertech.esper.runtime.client.UpdateListener;
5
6  no usages new *
7  public class ProstyListener implements UpdateListener {
8      new *
9      @Override
10     public void update(EventBean[] newEvents, EventBean[] oldEvents, EPStatement epStatement, EPRuntime epRuntime) {
11         if (newEvents != null) {
12             for (EventBean newEvent : newEvents) {
13                 System.out.println("ISTREAM : " + newEvent.getUnderlying());
14             }
15         }
16         if (oldEvents != null) {
17             for (EventBean oldEvent : oldEvents) {
18                 System.out.println("RSTREAM : " + oldEvent.getUnderlying());
19             }
20         }
21     }
22 }

```

22. Skoro mamy już klasę przeznaczoną do odbioru strumienia wynikowego (dowolnego zapytania), to przyszedł czas na zarejestrowanie obiektu tej klasy w utworzonym wcześniej zapytaniu.

W klasie Main, bezpośrednio pod poleceniem tworzącym obiekt zapytania utwórz obiekt listenera.

```
ProstyListener prostyListener = new ProstyListener();
```

23. Następnie dodaj polecenia rejestrujące utworzony obiekt prostyListener dla wdrożonych zapytań.

```

for (EPStatement statement : deployment.getStatements()) {
    statement.addListener(prostyListener);
}

```

24. Sprawdź czy postać metody main w Twoim programie jest analogiczna do poniższego przykładu

```

11  public static void main(String[] args) throws IOException {
12      Configuration configuration = new Configuration();
13      configuration.getCommon().addEventType(KursAkcji.class);
14      EPRuntime epRuntime = EPRuntimeProvider.getDefaultRuntime(configuration);
15
16      EPDeployment deployment = compileAndDeploy(epRuntime, epl: """
17          select istream spolka as X, kursOtwarcia as Y
18          from KursAkcji#length(3);""");
19
20      ProstyListener prostyListener = new ProstyListener();
21
22      for (EPStatement statement : deployment.getStatements()) {
23          statement.addListener(prostyListener);
24      }
25
26      CreateInputStream inputStream = new CreateInputStream();
27      inputStream.generuj(epRuntime.getEventService());
28  }

```

25. Uruchom program i sprawdź czy otrzymane przez Ciebie wyniki (zdarzenia wstawiane i usuwane) są dla Ciebie zrozumiałe.

```
ISTREAM : {X=Apple, Y=18.24}
ISTREAM : {X=CocaCola, Y=48.8}
ISTREAM : {X=Disney, Y=25.46}
ISTREAM : {X=Ford, Y=20.03}
RSTREAM : {X=Apple, Y=18.24}
. . .
RSTREAM : {X=Honda, Y=61.99}
ISTREAM : {X=Oracle, Y=10.79}
RSTREAM : {X=IBM, Y=94.1}
ISTREAM : {X=PepsiCo, Y=49.01}
RSTREAM : {X=Intel, Y=21.46}
ISTREAM : {X=Yahoo, Y=9.95}
RSTREAM : {X=Microsoft, Y=52.35}
```

## Testowanie różnych postaci zapytania

Wprowadzimy teraz kilka zmian do naszego zapytania celem obserwacji efektów tych zmian. Obecnie zapytanie ma następującą postać

```
16 EPDeployment deployment = compileAndDeploy(epRuntime, epl: """
17     select irstream spolka as X, kursOtwarcia as Y
18     from KursAkcji#length(3);""");
```

26. Ogranicz, za pomocą klauzuli WHERE, wynikowe zdarzenia jedynie do tych, które dotyczą spółki Oracle. Dlaczego w odróżnieniu od poprzedniego przypadku, bezpośrednio po pierwszym zdarzeniu w strumieniu zdarzeń wstawianych pojawiły zdarzenia w strumieniu zdarzeń usuwanych?

```
ISTREAM : {X=Oracle, Y=12.25}
RSTREAM : {X=Oracle, Y=12.25}
ISTREAM : {X=Oracle, Y=11.82}
RSTREAM : {X=Oracle, Y=11.82}
. . .
ISTREAM : {X=Oracle, Y=11.23}
RSTREAM : {X=Oracle, Y=11.23}
ISTREAM : {X=Oracle, Y=10.79}
```

27. Dodaj do postaci zdarzeń wynikowych datę kursu. Usuń także aliasy kolumn, zmieniające nazwy atrybutów zdarzeń wynikowych. Pozwoli nam to jednoznacznie identyfikować zdarzenia.

```
ISTREAM : {data=Wed Sep 05 00:00:00 CEST 2001, kursOtwarcia=12.25, spolka=Oracle}
RSTREAM : {data=Wed Sep 05 00:00:00 CEST 2001, kursOtwarcia=12.25, spolka=Oracle}
ISTREAM : {data=Thu Sep 06 00:00:00 CEST 2001, kursOtwarcia=11.82, spolka=Oracle}
RSTREAM : {data=Thu Sep 06 00:00:00 CEST 2001, kursOtwarcia=11.82, spolka=Oracle}
. . .
ISTREAM : {data=Wed Sep 19 00:00:00 CEST 2001, kursOtwarcia=11.23, spolka=Oracle}
RSTREAM : {data=Wed Sep 19 00:00:00 CEST 2001, kursOtwarcia=11.23, spolka=Oracle}
ISTREAM : {data=Thu Sep 20 00:00:00 CEST 2001, kursOtwarcia=10.79, spolka=Oracle}
```

28. Zmień zapytanie tak, aby już na poziomie konstruowania okna były brane pod uwagę tylko kursy spółki Oracle. Znowu coś zmieniło. Dlaczego pojawiły się trzy zdarzenia wstawiane przed pierwszym usuwanym?

```
ISTREAM : {data=Wed Sep 05 00:00:00 CEST 2001, kursOtwarcia=12.25, spolka=Oracle}
ISTREAM : {data=Thu Sep 06 00:00:00 CEST 2001, kursOtwarcia=11.82, spolka=Oracle}
ISTREAM : {data=Fri Sep 07 00:00:00 CEST 2001, kursOtwarcia=10.86, spolka=Oracle}
ISTREAM : {data=Mon Sep 10 00:00:00 CEST 2001, kursOtwarcia=10.89, spolka=Oracle}
RSTREAM : {data=Wed Sep 05 00:00:00 CEST 2001, kursOtwarcia=12.25, spolka=Oracle}
ISTREAM : {data=Mon Sep 17 00:00:00 CEST 2001, kursOtwarcia=10.29, spolka=Oracle}
. . .
RSTREAM : {data=Mon Sep 17 00:00:00 CEST 2001, kursOtwarcia=10.29, spolka=Oracle}
```

29. Zamień w zapytaniu operator `irstream` na operator `istream`, wyłączając generowanie wynikowego strumienia zdarzeń usuwanych.

```
ISTREAM : {data=Wed Sep 05 00:00:00 CEST 2001, kursOtwarcia=12.25, spolka=Oracle}
ISTREAM : {data=Thu Sep 06 00:00:00 CEST 2001, kursOtwarcia=11.82, spolka=Oracle}
ISTREAM : {data=Fri Sep 07 00:00:00 CEST 2001, kursOtwarcia=10.86, spolka=Oracle}
ISTREAM : {data=Mon Sep 10 00:00:00 CEST 2001, kursOtwarcia=10.89, spolka=Oracle}
ISTREAM : {data=Mon Sep 17 00:00:00 CEST 2001, kursOtwarcia=10.29, spolka=Oracle}
ISTREAM : {data=Tue Sep 18 00:00:00 CEST 2001, kursOtwarcia=10.95, spolka=Oracle}
ISTREAM : {data=Wed Sep 19 00:00:00 CEST 2001, kursOtwarcia=11.23, spolka=Oracle}
ISTREAM : {data=Thu Sep 20 00:00:00 CEST 2001, kursOtwarcia=10.79, spolka=Oracle}
```

30. Zmień zapytanie tak, aby każdego dnia zwracało maksymalne kursy otwarcia dla spółki Oracle z każdych kolejnych 5 ostatnich notowań.

```
ISTREAM : {data=Wed Sep 05 00:00:00 CEST 2001, max(kursOtwarcia)=12.25, spolka=Oracle}
ISTREAM : {data=Thu Sep 06 00:00:00 CEST 2001, max(kursOtwarcia)=12.25, spolka=Oracle}
ISTREAM : {data=Fri Sep 07 00:00:00 CEST 2001, max(kursOtwarcia)=12.25, spolka=Oracle}
ISTREAM : {data=Mon Sep 10 00:00:00 CEST 2001, max(kursOtwarcia)=12.25, spolka=Oracle}
ISTREAM : {data=Mon Sep 17 00:00:00 CEST 2001, max(kursOtwarcia)=12.25, spolka=Oracle}
ISTREAM : {data=Tue Sep 18 00:00:00 CEST 2001, max(kursOtwarcia)=11.82, spolka=Oracle}
ISTREAM : {data=Wed Sep 19 00:00:00 CEST 2001, max(kursOtwarcia)=11.23, spolka=Oracle}
ISTREAM : {data=Thu Sep 20 00:00:00 CEST 2001, max(kursOtwarcia)=11.23, spolka=Oracle}
```

31. Zamiast maksymalnego kursu otwarcia z ostatnich 5 notowań, umieść w zdarzeniach wynikowych informację o różnicy pomiędzy bieżącym kursem otwarcia a maksymalnym kursem otwarcia z ostatnich 5 notowań. Czy rozumiesz jak działa funkcja `max`? Czy różni się to działanie od funkcji `max` w języku SQL?

```
ISTREAM : {data=Wed Sep 05 00:00:00 CEST 2001, spolka=Oracle, roznica=0.0}
ISTREAM : {data=Thu Sep 06 00:00:00 CEST 2001, spolka=Oracle, roznica=-0.4300003}
ISTREAM : {data=Fri Sep 07 00:00:00 CEST 2001, spolka=Oracle, roznica=-1.3900003}
ISTREAM : {data=Mon Sep 10 00:00:00 CEST 2001, spolka=Oracle, roznica=-1.3599997}
ISTREAM : {data=Mon Sep 17 00:00:00 CEST 2001, spolka=Oracle, roznica=-1.96}
ISTREAM : {data=Tue Sep 18 00:00:00 CEST 2001, spolka=Oracle, roznica=-0.8699999}
ISTREAM : {data=Wed Sep 19 00:00:00 CEST 2001, spolka=Oracle, roznica=0.0}
ISTREAM : {data=Thu Sep 20 00:00:00 CEST 2001, spolka=Oracle, roznica=-0.43999958}
```

32. Napisz zapytanie, które będzie generowało do wynikowego strumienia zdarzeń wstawianych jedynie te przypadki, w których kurs otwarcia spółki Oracle w bieżącym notowaniu **wzrósł** w stosunku do kursu otwarcia tej samej spółki w poprzednim notowaniu. Jeśli chcesz możesz oprzeć się na poprzednim rozwiązaniu.

*Podpowiedź: skorzystaj z klauzuli HAVING – tylko tam możesz zdefiniować warunek operujący na funkcjach agregujących.*

```
ISTREAM : {data=Mon Sep 10 00:00:00 CEST 2001, spolka=Oracle, roznica=0.030000687}
ISTREAM : {data=Tue Sep 18 00:00:00 CEST 2001, spolka=Oracle, roznica=0.65999985}
ISTREAM : {data=Wed Sep 19 00:00:00 CEST 2001, spolka=Oracle, roznica=0.27999973}
```

Dane źródłowe spółki Oracle są następujące:

```
ISTREAM : {data=Wed Sep 05 00:00:00 CEST 2001, kursOtwarcia=12.25, spolka=Oracle}
ISTREAM : {data=Thu Sep 06 00:00:00 CEST 2001, kursOtwarcia=11.82, spolka=Oracle}
ISTREAM : {data=Fri Sep 07 00:00:00 CEST 2001, kursOtwarcia=10.86, spolka=Oracle}
ISTREAM : {data=Mon Sep 10 00:00:00 CEST 2001, kursOtwarcia=10.89, spolka=Oracle}
ISTREAM : {data=Mon Sep 17 00:00:00 CEST 2001, kursOtwarcia=10.29, spolka=Oracle}
ISTREAM : {data=Tue Sep 18 00:00:00 CEST 2001, kursOtwarcia=10.95, spolka=Oracle}
ISTREAM : {data=Wed Sep 19 00:00:00 CEST 2001, kursOtwarcia=11.23, spolka=Oracle}
ISTREAM : {data=Thu Sep 20 00:00:00 CEST 2001, kursOtwarcia=10.79, spolka=Oracle}
```

Czy uzyskany przez Ciebie wynik jest poprawny? Czy faktycznie uzyskane przez Ciebie zostały tylko wzrosty?