

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Кафедра МСС

Гилевич Павел Геннадьевич

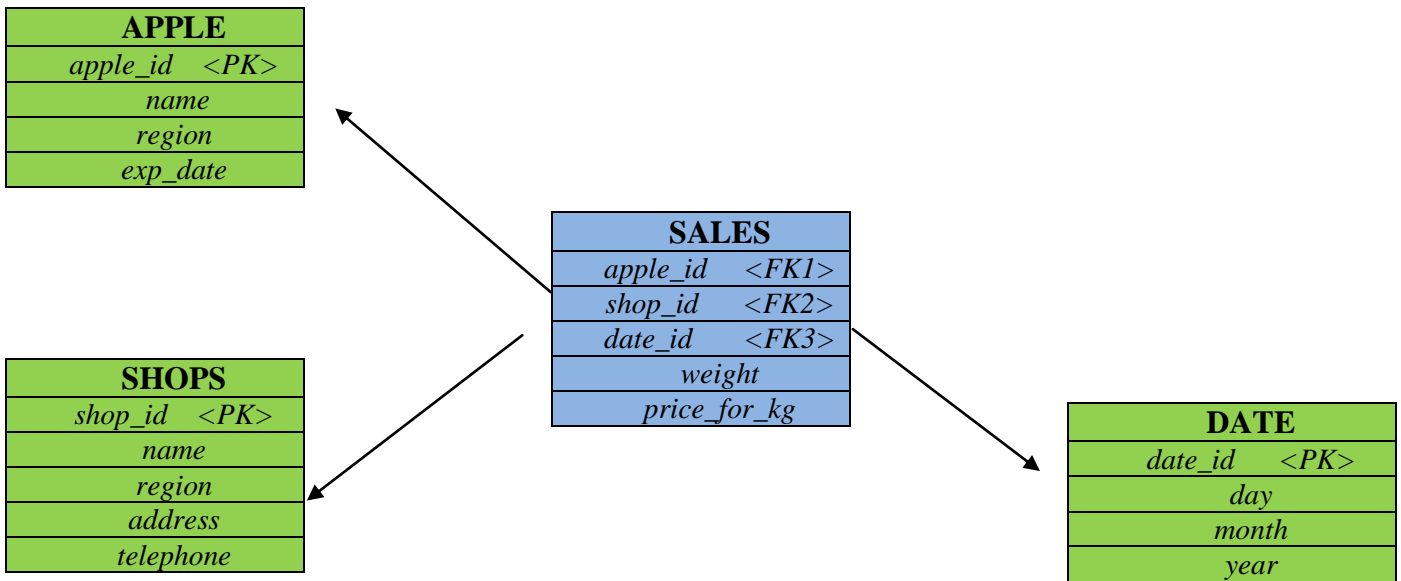
Отчет по лабораторной работе
студента 3 курса 12 группы

Преподаватель
Довнар С.Е.

Минск 2013

Отчет

В начале работы была создана схема звездного соединения - специальная организация таблиц, удобная для хранения многомерных показателей. Схема звезды включает в себя одну таблицу фактов (таблица Sales) и три таблицы измерений (таблицы Apples, Shops, Date).



структура схемы звезды

Таблицы измерений расшифровывают ключи, на которые ссылается таблица фактов (в данном случае - поля *apple_id*, *shop_id*, *date_id*), содержат более подробные сведения.

На следующем этапе с помощью SQLite была создана база данных, соответствующая приведенной выше схеме.

Name	Object	Type	Schema
Sales	table		CREATE TABLE Sales (apple_id INTEGER REFERENCES apple (apple_ID), shop_id INTEGER REFERENCES Shops (shop_ID), date_id INTEGER REFERENCES Date (date_ID), weight NUMERIC, price_for_kg NUMERIC)
apple	table		CREATE TABLE apple (ID INTEGER PRIMARY KEY, name TEXT, region TEXT, exp_date Date)
Shops	table		CREATE TABLE Shops (ID INTEGER PRIMARY KEY, name TEXT, region TEXT, address TEXT, telephone TEXT)
Date	table		CREATE TABLE Date (ID INTEGER PRIMARY KEY, day INTEGER, month TEXT, year INTEGER)

Далее эта база данных была заполнена конкретными значениями.

Table: apple					Table: Shops				
apple_ID	name	region	exp_date		shop_ID	name	region	address	telephone
1	1 Antonovka	Grodno	2014-01-05		1	Evroopt	Minsk	Montazhnikov str. 2	215-16-13
2	2 Belyi Naliv	Logoisk	2014-04-11		2	Almi	Molodechno	V.Gostinets str. 120	74-33-18
3	3 Papirovka	Soligorsk	2014-05-14		3	Belmarket	Mogilev	Kosmonavtov ave.7	24-22-12
4	4 Imrus	Molodechno	2014-03-06		4	Sosedi	Grodno	Fomicheva str. 16	54-00-27
5	5 Alesya	Myadel	2013-12-25		5	ProStore	Minsk	Pobeditelei ave.80	207-17-45
6	6 Darunak	Ivye	2014-02-23						
7	7 Pospeh	Voronovo	2014-03-04						

Table: Date

	date ID	day	month	year
1	1	25	September	2013
2	2	26	September	2013
3	3	27	September	2013
4	4	30	September	2013
5	5	1	October	2013
6	6	2	October	2013
7	7	3	October	2013
8	8	4	October	2013

Table: Sales

	apple id	shop id	date id	weight	price for kg
1	2	1	5	20.3	680
2	1	2	6	16.1	859
3	4	3	1	14	770
4	3	3	2	22.3	900
5	5	2	8	18.4	890
6	4	4	4	21.6	910
7	2	5	2	17	719
8	7	4	5	17.9	1010
9	3	5	3	25.6	929
10	1	1	7	33.4	840
11	6	1	1	31.5	1120
12	5	4	4	20	889
13	7	5	1	15.2	1000
14	1	4	3	28	860
15	6	5	7	37.8	1099
16	3	2	4	29.6	910
17	1	3	1	17.1	870
18	1	5	2	29.9	890
19	2	3	1	41	710
20	2	4	6	18.9	750
21	3	1	2	31.6	990
22	7	2	2	10	1100
23	6	4	3	29.5	1179
24	5	5	2	11.6	810
25	7	3	6	14.8	1050

1. Измерение Apple

представляет информацию о различных сортах яблок

- ID: уникальный ключ
- name: название сорта (например, Антоновка)
- region: регион, где яблоки данного сорта были собраны (например, Гродно)
- exp_date: срок годности

2. Измерение Shops

представляет информацию о магазинах, в которых ведется продажа различных сортов яблок

- ID: уникальный ключ магазина
- name: название магазина (например, «Евроопт»)
- region: название населенного пункта, где данный магазин расположен (например, Минск)
- address: адрес магазина (например, улица Мира, 17)
- telephone: телефон

3. Измерение Date

описывает дату

- ID: уникальный ключ даты
- day: день (например, 28)
- month: название месяца (например, Октябрь)
- year: год (например, 2013)

4. Таблица фактов Sales

представляет информацию о всех совершенных операциях продажи яблок

- apple_id: внешний ключ, код яблока
- shop_id: внешний ключ, код магазина
- date_id: внешний ключ, код даты
- weight: общий вес яблок определенного сорта, проданных за день в данном магазине
- price_for_kg: цена за килограмм яблок

Описание базы данных

Были созданы 2 XML-документа, описывающие соответственно таблицу фактов и таблицы измерений.

Документ *Facts.xml*, описывающий структуру таблицы фактов:

<Report_Facts>

```

=<CubeInfo>
<name>Apples' sales</name>
<ID>1</ID>
</CubeInfo>
=<Datatable>
<name>Продажи</name>
<real_name>Sales</real_name>
=<Fields>
=<Field>
<ID>1</ID>
<name>ID яблока</name>
<real_name>apple_id</real_name>
<dimension>1</dimension>
</Field>
=<Field>
<ID>2</ID>
<name>ID магазина</name>
<real_name>shop_id</real_name>
<dimension>2</dimension>
</Field>
=<Field>
<ID>3</ID>
<name>ID даты</name>
<real_name>date_id</real_name>
<dimension>3</dimension>
</Field>
=<Field>
<ID>4</ID>
<name>Вес</name>
<real_name>weight</real_name>
<dimension />
</Field>
=<Field>
<ID>5</ID>
<name>Цена за килограмм</name>
<real_name>price_for_kg</real_name>
<dimension />
</Field>
</Fields>
</Datatable>
</Report_Facts>

```

1. Тег *CubeInfo* содержит информацию о «кубике»

- тег *name* - имя «кубика»
- тег *ID* - ID «кубика»

2. Тег *Datatable* представляет информацию о самой таблице.

- тег *name* - имя таблицы, под которым она представляется пользователю
- тег *real_name* - имя таблицы в базе данных
- тег *Fields* - информация о полях таблицы:
 - тег *ID* - ID поля
 - тег *name* - имя поля для пользователя
 - тег *real_name* - имя поля в базе данных
 - тег *dimension* - ID измерения, с которым связано это поле (если поле не является внешним ключом, то информации в теге нет)

Часть документа *Dimensions.xml*, описывающего таблицы измерений:

```

<Report_Dimensions>
=<CubeInfo>
<name>Apples' sales</name>
<ID>1</ID>

```

```

        </CubeInfo>
    = <Dimensions>
    = <Dimension>
    = <DimensionInfo>
        <ID>1</ID>
        <name>Яблоки</name>
</info>Информация о различных сортах яблок</info>
    </DimensionInfo>
    = <Attributes>
    = <Attribute>
        <ID>1</ID>
        <name>ID яблока</name>
        <real_name>ID</real_name>
        <type>integer</type>
    </Attribute>
    = <Attribute>
        <ID>2</ID>
        <name>Название сорта</name>
        <real_name>name</real_name>
        <type>text</type>
    </Attribute>
    = <Attribute>
        <ID>3</ID>
        <name>Место сбора</name>
        <real_name>region</real_name>
        <type>text</type>
    </Attribute>
    = <Attribute>
        <ID>4</ID>
        <name>Срок годности</name>
        <real_name>exp_date</real_name>
        <type>Date</type>
    </Attribute>
    </Attributes>
    = <TableInfo>
        <name>apple</name>
        <PrimaryKey>ID</PrimaryKey>
    </TableInfo>
    </Dimension>
    ...
</Dimensions>
</Report_Dimensions>

```

1. Тег *CubeInfo* содержит информацию о «кубике»
 - тег *name* - имя «кубика»
 - тег *ID* - ID «кубика»
2. В теге *Dimensions* представляется информация обо всех измерениях (каждое измерение представлено отдельным тегом *Dimension*)
 - тег *DimensionInfo*
 - тег *ID* - ID измерения
 - тег *name* - имя измерения, под которым его видит пользователь
 - тег *info* - дополнительная информация об измерении
 - тег *Attributes* представляет информацию об атрибутах (каждый атрибут представлен отдельным тегом *Attribute*)
 - тег *ID* - ID атрибута
 - тег *name* - имя, под которым атрибут представляется пользователю
 - тег *real_name* - имя атрибута в базе данных
 - тег *type* - тип данных атрибута
 - тег *TableInfo* представляет информацию о таблице в базе данных
 - тег *name* - имя таблицы в базе данных

➤ тег *PrimaryKey* - имя поля, являющегося первичным ключом

Файлы отчетов

На данном этапе была написана утилита, формирующая отчеты в виде XML-файлов. Она позволяет зафиксировать одно измерение, а также выбрать, какое измерение будет выводиться в строках, а какое - в столбцах. В проекте утилита представлена функцией *GenerateXML*:

```
public static void GenerateXML(List<Dimensions> allDimensions, SQLiteConnection connection)
{
    string nameXML = "D:\\UNIVER\\3 курс\\Спецкурс\\";
    Console.WriteLine("Print the name of XML Document: ");
    nameXML += Console.ReadLine();
    nameXML += ".xml";

    List<Dimensions> copyList = new List<Dimensions>(allDimensions); //список
измерений

    int dimensionColumn = 0, dimensionRow = 0, dimensionFixed = 0;

    using (XmlWriter writer = XmlWriter.Create(nameXML))
    {
        writer.WriteStartDocument();
        writer.WriteStartElement("Report");

        int key = 0;
        do
        {
            //выбираем измерение по столбцам
            Console.WriteLine("Select dimension on column\n 1 - {0}\n 2 - {1}\n 3 -
{2}\nInput key: ", copyList[0].UserName, copyList[1].UserName, copyList[2].UserName);
            key = Convert.ToInt32(Console.ReadLine());
            if (key != 1 & key != 2 & key != 3)
            {
                Console.WriteLine("Wrong key!");
                key = 0;
            }

            else
            {
                dimensionColumn = copyList[key - 1].ID;
                copyList.RemoveAt(key - 1);
            }
        } while (key == 0);

        do
        {
            //аналогично выбираем измерение по строкам
        } while (key == 0);

        dimensionFixed = copyList[0].ID; //фиксированное измерение
        writer.WriteElementString("DimensionByColumn", allDimensions[dimensionColumn-
1].ID.ToString());
        writer.WriteElementString("DimensionByRow", allDimensions[dimensionRow-
1].ID.ToString());
        writer.WriteStartElement("Fixed");
        writer.WriteElementString("Dimension", allDimensions[dimensionFixed-
1].ID.ToString());

        writer.WriteStartElement("ID");
        Selection(allDimensions[dimensionFixed-1], connection, writer);
        writer.WriteEndElement();
        writer.WriteEndElement();

        writer.WriteStartElement("Selection");

        writer.WriteStartElement("Column");
        writer.WriteStartElement("ID");
        Selection(allDimensions[dimensionColumn-1], connection, writer);
        writer.WriteEndElement();
```

```

writer.WriteEndElement();

writer.WriteStartElement("Row");
writer.WriteStartElement("ID");
Selection(allDimensions[dimensionRow-1], connection, writer);
writer.WriteEndElement();
writer.WriteEndElement();

writer.WriteEndElement();
writer.WriteEndElement();

```

```

}
}

```

Функция *Selection* используется при формировании отчета для определения зафиксированных полей в каждом измерении и включении их в XML-документ:

```

public static void Selection(Dimensions dimension, SQLiteConnection connection, XmlWriter
writer)
{
    Console.WriteLine("-----\nSelect fixed ID in dimension {0}:",
dimension.UserName);
    string command = "SELECT * FROM ";
    command += dimension.TableName;           //запрос для получения из БД всех полей
измерения dimension

    SQLiteCommand comm = new SQLiteCommand(command, connection);
    SQLiteDataReader reader = comm.ExecuteReader();
    foreach (DbDataRecord record in reader)
        Console.WriteLine(record[0] + " " + record[1] + " " + record[2]);

    Console.WriteLine("-----");
    string result = Console.ReadLine(); //вводим значения ID для тех полей, которые
хотим зафиксировать
    writer.WriteString(result);         //записываем эти значения в XML
}

```

Представим пример полученного отчета:

```

<?xml version="1.0" encoding="utf-8" ?>
    = <Report>
<DimensionByColumn>1</DimensionByColumn>
    <DimensionByRow>3</DimensionByRow>
    = <Fixed>
        <Dimension>2</Dimension>
        <ID>1</ID>
    </Fixed>
    = <Selection>
    = <Column>
        <ID>2 4 5</ID>
    </Column>
    = <Row>
        <ID>2 8</ID>
    </Row>
    </Selection>
    </Report>

```

1. Тег *DimensionByColumn* - id измерения по столбцам (id из файла Dimensions.xml)
2. Тег *DimensionByRow* - id измерения по строкам (id из файла Dimensions.xml)
3. Блок *Fixed* представляет информацию о зафиксированном измерении
 - тег *Dimension* - id зафиксированного измерения (из файла Dimensions.xml)
 - тег *ID* - зафиксированное значение первичного ключа
4. Блок *Selection* представляет ключи, выбранные из таблиц первых двух тегов
 - блок *Column* в теге *ID* представляет значения ключа для измерения по столбцам
 - блок *Row* в теге *ID* представляет значения ключа для измерения по строкам

В заключение представим скриншоты работы утилиты.

```
C:\Windows\system32\cmd.exe
1 - Create XML
2 - Open report
0 - Exit
Select action: 1
Print the name of XML Document: fixdate

Select dimension on column
1 - Яблоки
2 - Магазины
3 - Дата
Input key: 2

Select dimension on row
1 - Яблоки
2 - Дата
Input key: 1

Fixed dimension is Дата

Select fixed ID in dimension Дата:
1 - 25 September
2 - 26 September
3 - 27 September
4 - 30 September
5 - 1 October
6 - 2 October
7 - 3 October
8 - 4 October
-----
4

Select fixed ID in dimension Магазины:
1 - Euroopt Minsk
2 - Almi Molodechno
3 - Belmarket Mogilev
4 - Sosedni Grodno
5 - ProStore Minsk
-----
1 2

Select fixed ID in dimension Яблоки:
1 - Antonovka Grodno
2 - Belyi Maliv Logoisk
3 - Papirouka Soligorsk
4 - Inrus Molodechno
5 - Alesya Myadel
6 - Darunak Iyye
7 - Pospesh Voronovo
-----
3 4 5
```

Описание работы программы

Были созданы классы:

- *Dimensions*, для описания таблицы измерений

```
class Dimensions
{
    int dimensionID;           //id измерения
    String userName;           //имя для пользователя
    String tableName;          //имя таблицы
    String PrimaryKey;         //имя поля-первичного ключа
    List<Attribute> Attributes; //список атрибутов

    ...
}
```

Также представим класс *Attributes*

```
class Attribute
{
    protected int attributeID; //id атрибута
    protected String name;     //имя для пользователя
    protected String realName; //имя в таблице

    ...
}
```

- *Facts*, для описания таблицы фактов

```
class Facts
{
    String name;           //имя таблицы для пользователя
    String realName;       //имя таблицы в БД
    int measure;           //id поля, в котором находится мера
    List<Field> fields;     //список полей в таблице

    ...
}
```

Класс *Field*, являющийся наследником класса *Attribute*:

```
class Field : Attribute
{
    int ReferenceToDimension; //связь с измерением (внешний ключ)

    ...
}
```


Класс *Element* используется для хранения ID измерений по столбцам/строкам и первичных ключей, которые будут использоваться при выполнении среза «кубика».

```
class Element
{
    int DimensionID;           //id измерения
    public List<int> values;    //значения ключей

    public Element()
    {
        values = new List<int>();
    }
    ...
};
```

Класс *FixedElement* выполняет ту же задачу, но для фиксированного измерения.

```
class FixedElement
{
    int DimensionID;           //id измерения
    int FixedID;               //значение зафиксированного первичного ключа
    ...
};
```

Далее приведем некоторые фрагменты исходного кода приложения.

```
//создание экземпляров классов Element и FixedElement
FixedElement fixedElement = new FixedElement();
Element dimensionColumn = new Element();
Element dimensionRow = new Element();

//создание списка измерений, а также экземпляра класса таблицы фактов
List<Dimensions> allDimensions = new List<Dimensions>();
Facts facts = new Facts();

//заполнение полей экземпляра таблицы фактов и списка измерений
ReadFacts(facts);
ReadDimensions(allDimensions);
```

Для примера приведем код функции *ReadFacts (Facts)*, принцип работы функции *ReadDimensions(List<Dimensions>)* схожий:

```
public static void ReadFacts(Facts facts)
{
    XmlDocument xml = new XmlDocument();
    xml.Load("D:\\UNIVER\\3 курс\\Спецкурс\\Facts.xml");
    XmlNode report = xml.SelectSingleNode("Report_Facts");
    XmlNode datatable = report.SelectSingleNode("Datatable");
    facts.Name = datatable.SelectSingleNode("name").InnerText;
    facts.RealName = datatable.SelectSingleNode("real_name").InnerText;
    facts.Measure = Convert.ToInt32(datatable.SelectSingleNode("measure").InnerText);
    XmlNode fields = datatable.SelectSingleNode("Fields");

    foreach (XmlNode field in fields)
    {
        Field concrete_field = new Field();
        concrete_field.ID = Convert.ToInt32(field.SelectSingleNode("ID").InnerText);
        concrete_field.Name = field.SelectSingleNode("name").InnerText;
        concrete_field.RealName = field.SelectSingleNode("real_name").InnerText;
        String reference = field.SelectSingleNode("dimension").InnerText;
        if (reference.Equals(""))
            concrete_field.Reference = 0;
        else
            concrete_field.Reference = Convert.ToInt32(reference);

        facts.AddField(concrete_field);
    }
}
```

Пример формирования и выполнения запроса для получения названий столбцов таблицы (запрос для получения название строк аналогичен)

```
//формируем запрос для получения названий строк
```

```

string command_Row = "SELECT * FROM " + allDimensions[dimensionRow.DimID-1].TableName
+ " WHERE " + allDimensions[dimensionRow.DimID-1].PK + " = " + dimensionRow.values[0].ToString();
for (int i = 1; i < dimensionRow.values.Count; i++)
{
    command_Row += " OR ID = ";
    command_Row += dimensionRow.values[i].ToString();
}
SQLiteCommand comm_Row = new SQLiteCommand(command_Row, connection);
SQLiteDataReader reader_Row = comm_Row.ExecuteReader();

//запрос для получения данных о фиксированном измерении
string command = "SELECT * FROM " + allDimensions[fixedElement.DimID-1].TableName + "
WHERE " + allDimensions[fixedElement.DimID-1].PK + " = " + fixedElement.FixID.ToString();
SQLiteCommand comm = new SQLiteCommand(command, connection);
SQLiteDataReader reader = comm.ExecuteReader();

```

Вывод данных в таблицу:

```

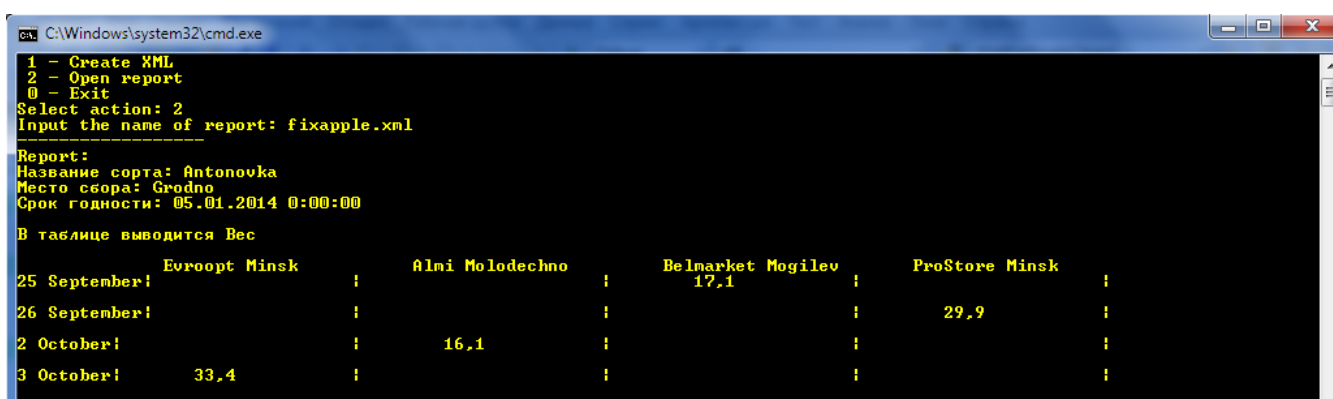
foreach (DbDataRecord record_Row in reader_Row)
{
    string selection = "SELECT " + facts.GetField(facts.Measure-1).RealName + " FROM "
+ facts.RealName + " WHERE " + facts.GetField(dimFixed.ID-1).RealName + " = " +
fixedElement.FixID.ToString() + " AND " + facts.GetField(dimRow.ID-1).RealName + " = " +
dimensionRow.values[order] + " AND ";

    Console.WriteLine("{0} {1}|", record_Row[1], record_Row[2]);
    for (int l = 0; l < dimensionColumn.values.Count; l++)
    {
        string finalSelection = selection + facts.GetField(dimColumn.ID-1).RealName +
" = " + dimensionColumn.values[l].ToString();
        SQLiteCommand result = new SQLiteCommand(finalSelection, connection);
        SQLiteDataReader weights = result.ExecuteReader();

        Console.WriteLine("\t");
        foreach (DbDataRecord weight in weights)
            Console.WriteLine("{0}", weight[0]);
        Console.WriteLine("\t\t|");
    }
    order++;
    Console.WriteLine("\n-----");
}

```

Скриншот выведенной в результате таблицы:



Report:
 Название сорта: Antonovka
 Место сбора: Grodno
 Срок годности: 05.01.2014 0:00:00

В таблице выводится Вес

	Evroopt Minsk	Almi Molodechno	Belmarket Mogilev	ProStore Minsk
25 September!			17,1	
26 September!				29,9
2 October!		16,1		
3 October!	33,4			