**Контрольное домашнее задание (CW)**
**Алгоритмы и структуры данных 2020**

# Invisible Join

Авторы задания: команда курса «Алгоритмы и структуры данных 2020».
Вопросы по заданию направлять:
Бузулуков Алексей Максимович ambuzulukov@edu.hse.ru
или в канал MS Teams “**Контрольное домашнее задание CW**”

## Отправка решения

Результатом решения должен быть один файл на языке C# (.cs), а также файлы тестов (входные и выходные данные). Именно файл на языке C# (.cs) и тесты будут оцениваться. Решение нужно загрузить в LMS в проект **Контрольное домашнее задание CW** по дисциплине Алгоритмы и структуры данных 2020 уч. год Б 2 курс (код 131940, ОП: М090304ПИНЖ)
*При выявлении плагиата будет выставлена оценка 0 баллов*

**Дедлайн. 22 октября 2020 г. 23:59 МСК**
Пожалуйста, загружайте решения заранее, чтобы не возникало технических препятствий в последний момент. Время на сервере LMS и Вашем устройстве может отличаться. Пожалуйста, настройте доступ в ЛМС (даже если Вы проходите курс по ИУП): архивы с решениями не принимаются по электронной почте.

## Что нужно сделать?

1. Задание разбито на 3 уровня сложности. Выберите **только один уровень** сложности. Выполнение более сложной задачи не гарантирует автоматически более высокую оценку.
2. Решите задание (один файл .cs).
3. Напишите 5 своих тестов (пары файлов входных и выходных данных) к задаче, пронумеруйте их соответственно. Разместите файлы с входными данными в директории input, а с выходными в директории output.
4. Создайте директорию с решением по шаблону <Фамилия и Имя>_<группа>_<уровень>, например BuzulukovAlexey_161_3. Разместите внутри нее директории input и output с тестами и один файл .cs с решением задачи. Внимание: номер выбранного Вами уровня задания находится в имени директории (в примере BuzulukovAlexey_161_3 — это уровень 3).
5. Упакуйте корневую директорию (BuzulukovAlexey_161_3) в zip-архив и отправьте в LMS (размер файла должен быть не более 1МБ).

**Внимание**: не публикуйте свое решение в открытом доступе, например на GitHub либо в других открытых источниках. Это действие будет приравниваться к плагиату и повлечет за собой оценку 0 баллов за домашнее задание (в том числе при обнаружении такового факта уже после выставления оценки).

## Мотивация

В эпоху Data Science данные являются ключевым инструментом для работы бизнеса и их объем растет экспоненциально. Данные загружаются в хранилища, причем в специальном виде, т.к. их организация напрямую влияет на эффективность последующей работы. Один из самых популярных способов организации таблиц данных - *схема типа "Звезда" (Star Schema)*. В ней имеется одна главная *таблица фактов (Fact Table)*, хранящая наблюдения и события, и *таблицы измерений (Dimension Table)*, описывающие бизнес-сущности, связанные с главной таблицей фактов через столбец уникальных ключей. Запросы направляются к таблице фактов, применяя различные фильтрации по данным из таблиц измерений или таблицы фактов. Таким образом, таблицы соединяются (Join) в одну, по которой и будут применены указанные фильтры. Но в эпоху больших данных такие соединения таблиц бывают сильно время- и ресурсозатратные, поэтому изобретают все новые алгоритмы реализации соединений таблиц. Один из них - *Invisible Join*, который применяется во многих современных СУБД.

Вам, как программным инженерам, необходимо реализовать этот алгоритм, чтобы помочь аналитикам быстрее получать ответы на их запросы к хранилищу, основываясь на имеющейся модели данных.

## Модель данных задания

Данные основаны на датасете *AdventureWorksWH*, опубликованные Microsoft в своем репозитории по лицензии MIT: https://github.com/microsoft/sql-server-samples.

*Adventure Works Cycles* - вымышленная компания, которая производит и продает велосипеды на коммерческих рынках Северной Америки, Европы и Азии. Компания стремится расширить свою долю на рынке, ориентируя продажи на своих лучших клиентов, расширяя доступность своей продукции через магазины-реселлеры.

В модели данных имеются основная таблица фактов - FactResellerSales и связанные с ней таблицы измерений с префиксом Dim (*рисунок 1*). Гарантируется, что первое поле (ключ) Dim-таблиц является уникальным значением в столбце таблицы. Имена таблиц с их полями на маппинг типов C# доступен в *листинге 1*.

В архиве с заданием в директории *data* вы найдете файлы с данными, на основе которых будете выдавать ответы на запросы. Таблица фактов (FactResellerSales) разбита по столбцам в отдельные файлы, формат именования: '*FactResellerSales.<имя столбца>.csv*'. Данные таблиц измерений не разбиты и доступны в соответствующем файле таблицы, формат именования: '*<имя таблицы измерений>.csv*'. Порядок полей (столбцов) в файле соответствует порядку полей из *листинга 1* (из этого же листинга можно узнать, какие типы данных представляют собой nvarchar и другие типы, используемые в схеме базы данных). Значения полей в файлах разделены знаком '|', строки разделены символом конца строки '\n'. Конец файла - пустая строка.

**Модель данных задания и порядок колонок меняться не будут!**

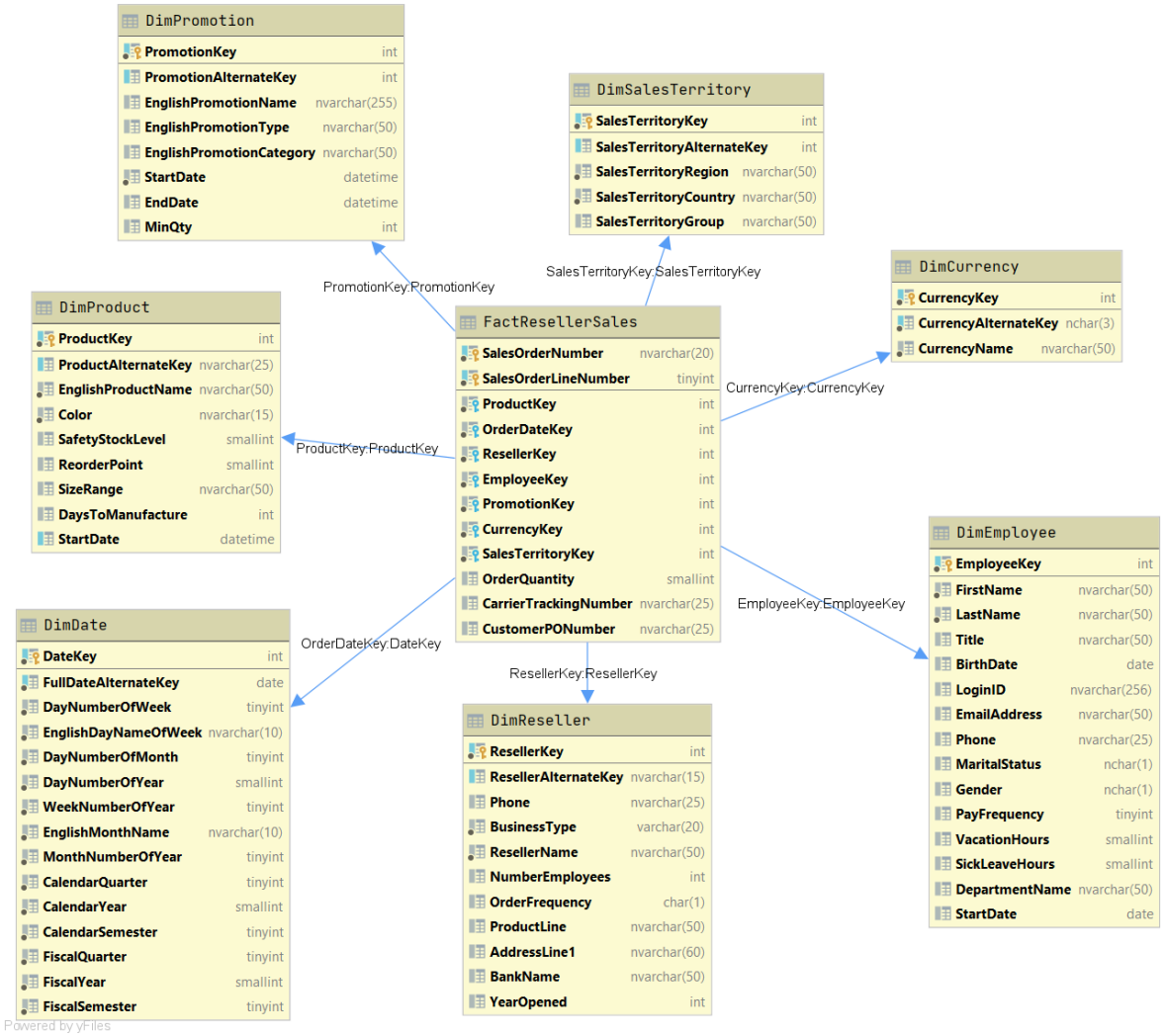


*Рисунок 1. Модель данных задания*

*Листинг 1. Таблица и список полей на маппинг C# типов*

**FactResellerSales:**

```
ProductKey int,
OrderDateKey int,
ResellerKey int,
EmployeeKey int,
PromotionKey int,
CurrencyKey int,
SalesTerritoryKey int,
SalesOrderNumber string,
SalesOrderLineNumber byte,
OrderQuantity short,
CarrierTrackingNumber string,
CustomerPONumber string
```

**DimProduct:**
```
ProductKey int,
ProductAlternateKey string,
EnglishProductName string,
Color string,
SafetyStockLevel short,
ReorderPoint short,
SizeRange string,
DaysToManufacture int,
StartDate string
```

**DimReseller:**
```
ResellerKey int,
ResellerAlternateKey string,
Phone string,
BusinessType string,
ResellerName string,
NumberEmployees int,
OrderFrequency string,
ProductLine string,
AddressLine1 string,
BankName string,
YearOpened int
```

**DimCurrency:**
```
CurrencyKey int,
CurrencyAlternateKey string,
CurrencyName string
```

**DimPromotion:**
```
PromotionKey int,
PromotionAlternateKey int,
EnglishPromotionName string,
EnglishPromotionType string,
EnglishPromotionCategory string,
StartDate string,
EndDate string,
MinQty int
```

**DimSalesTerritory:**
```
SalesTerritoryKey int,
SalesTerritoryAlternateKey int,
SalesTerritoryRegion string,
SalesTerritoryCountry string,
SalesTerritoryGroup string
```

**DimEmployee:**

```
EmployeeKey int,
FirstName string,
LastName string,
Title string,
BirthDate string,
LoginID string,
EmailAddress string,
Phone string,
MaritalStatus string,
Gender string,
PayFrequency byte,
VacationHours short,
SickLeaveHours short,
DepartmentName string,
StartDate string
```

**DimDate:**

```
DateKey int,
FullDateAlternateKey string,
DayNumberOfWeek byte,
EnglishDayNameOfWeek string,
DayNumberOfMonth byte,
DayNumberOfYear short,
WeekNumberOfYear byte,
EnglishMonthName string,
MonthNumberOfYear byte,
CalendarQuarter byte,
CalendarYear short,
CalendarSemester byte,
FiscalQuarter byte,
FiscalYear short,
FiscalSemester byte
```

## Условие задания

1) Необходимо описать абстрактный класс Bitmap, содержащий абстрактные функции:

`void And(Bitmap other)` - выполняет операцию логического И текущего Bitmap с Bitmap, переданным в качестве аргумента функции. Результат выполнения операции помещается в вызываемый (текущий) объект.

`void Set(int i, bool value)` - устанавливает бит по индексу i в значение value.

`bool Get(int i)` - возвращает значение бита по индексу i.

*(Можно дополнять этот абстрактный класс по необходимости, однако за странные/ненужные члены класса будет снижена оценка – например, если реализация класса перестает быть bitmap как таковой)*

2) В зависимости от выбранного уровня сложности задания (см. Уровни оценивания) необходимо реализовать свой класс, наследующий абстрактный класс Bitmap (название выбираете сами).

В реализации запрещается использовать сторонние библиотеки и готовые битовые структуры данных! (Массив типа bool тоже считается запрещенным)

3) Реализовать алгоритм Invisible Join, использующий вашу реализацию через абстрактный класс Bitmap (полиморфизм). Алгоритм принимает на вход текстовый файл с запросом (см. Формат входных данных) и, на основе загружаемых таблиц данных (см. Модель данных задания), записывает строки в выходной текстовый файл (см. Формат выходных данных). В зависимости от выбранного уровня сложности реализуется одна из вариаций алгоритма.

**Вариация алгоритма Invisible Join #1**

На фазе 1 (этап фильтрации строк таблиц измерений по предикатам/условиям) для хранения отфильтрованных первичных ключей таблиц измерений можно использовать множество (HashSet).

На фазе 2 необходимо использовать Вашу собственную реализацию Bitmap.

Для данной вариации алгоритма существует ограничение входных данных: гарантируется, что список запрашиваемых полей выходной таблицы состоит только из множества полей таблицы фактов.

Следовательно, фазу 3 алгоритма Invisible Join реализовывать не нужно.

**Вариация алгоритма Invisible Join #2**

На фазе 1 (этап фильтрации строк таблиц измерений по предикатам/условиям) для хранения отфильтрованных первичных ключей таблиц необходимо использовать Вашу собственную реализацию Bitmap (посредством полиморфизма) - Roaring Bitmap.

На фазе 2 необходимо использовать Вашу собственную реализацию Bitmap - Roaring Bitmap.

Для данной вариации алгоритма список запрашиваемых полей выходной таблицы может включать поля из любых таблиц (включая таблицу фактов) модели данных задания.

Следовательно, необходимо реализовать фазу 3 алгоритма Invisible Join.

На фазе 3 необходимо использовать Вашу собственную реализацию Bitmap - Roaring Bitmap.

## Уровни оценивания

**Уровень 1** (оценка до 6 баллов) - Вариация алгоритма Invisible Join #1 и обычный массив бит. Один элемент массива должен содержать несколько бит (напр., если используется 4-байтовый целочисленный тип данных, то один элемент массива должен содержать 32 бита; необходимо использовать побитовые операции для работы с отдельными битами). Реализованный массив бит должен использоваться на фазе 2 реализации алгоритма Invisible Join.

**Уровень 2** (оценка до 8 баллов) - Вариация алгоритма Invisible Join #1 и Roaring Bitmap. Реализованный Roaring Bitmap должен использоваться на фазе 2 реализации алгоритма Invisible Join.

**Уровень 3** (оценка до 10 баллов) - Вариация алгоритма Invisible Join #2 и Roaring Bitmap. Реализованный Roaring Bitmap должен использоваться на всех фазах алгоритма Invisible Join.

## Формат входных данных

На первой строке входного файла - список полей в выходной таблице (<имя таблицы>.<имя поля>) через запятую без пробела. Гарантируется, что список состоит только из полей таблиц модели данных, т.е. несуществующих полей нет.

На второй строке находится натуральное число N (0 <= N <= 1000) - количество условий фильтрации данных (все условия - логическое И).

Далее N строк - условия фильтрации в формате:
<поле таблицы><пробел><оператор><пробел><значение>

где <поле таблицы>:
<имя таблицы>.<имя поля>

<значение> строкового типа оборачивается в одинарные кавычки: 'string'
Гарантируется, что тип поля таблицы совпадает с типом сравниваемого значения!
Операторы сравнения условий:
**<** - строго меньше
**>** - строго больше
**<=** - меньше либо равно,
**>=** - больше либо равно,
**=** - строгое равенство,
**<>** - строгое неравенство
Для строк существуют только операторы = и <> (чувствительные к регистру), таким образом строки сравниваются через `string1.Equals(string2)`

## Формат выходных данных

В выходной файл записываются строки после отработки запроса со значениями полей в том же порядке, что и на первой строке входного файла. Значения полей в файле разделены знаком '|', строки разделены символом конца строки '\n'. Последняя строка в файле должна быть пустой.

## Пример для вариации алгоритма Invisible Join #1

| Входные данные | Выходные данные |
|---|---|
| FactResellerSales.SalesOrderLineNumber,FactResellerSales.CarrierTrackingNumber<br>7<br>DimPromotion.MinQty <> 0<br>DimReseller.BankName = 'International Bank'<br>DimProduct.Color <> 'Silver'<br>DimCurrency.CurrencyAlternateKey = 'USD'<br>DimDate.CalendarYear < 2012<br>DimDate.DayNumberOfWeek >= 3<br>DimDate.DayNumberOfMonth = 29 | 3\|BA78-4228-89<br>1\|6F2B-45A4-9C<br>15\|29D0-4A7E-9E<br>10\|68A1-47A8-9A |

## Пример для вариации алгоритма Invisible Join #2

| Входные данные | Выходные данные |
|---|---|
| FactResellerSales.SalesOrderLineNumber,DimReseller.ResellerName<br>7<br>DimPromotion.MinQty <> 0<br>DimReseller.BankName = 'International Bank'<br>DimProduct.Color <> 'Silver'<br>DimCurrency.CurrencyAlternateKey = 'USD'<br>DimDate.CalendarYear < 2012<br>DimDate.DayNumberOfWeek >= 3<br>DimDate.DayNumberOfMonth = 29 | 3\|The Gear Store<br>1\|Preferred Bikes<br>15\|Bicycle Outfitters<br>10\|This Area Sporting Goods |

## Как будет оцениваться работа?

Решения будут проверяться на наборах тестов, каждый тест — это два файла, один для входных данных, другой для выходных. Таким образом, ввод и вывод данных происходит через файлы.

Будут оценены: корректность кода, составленные тесты (граничные условия, сложные случаи и т.п.), декомпозиция, аккуратность кода. Возможны устные собеседования при сомнении в самостоятельности выполнения работы.

В приложенных к условию задачи директориях расположено по 5 тестов к каждой вариации алгоритма, на которых можно проверить работоспособность решения, однако стоит учитывать, что прохождение приложенных тестов не гарантирует полное отсутствие ошибок в решении (решения будут также проверяться и на других, расширенных тестах). Файлы входных данных именуются по принципу '*test<номер места>.txt*', файлы выходных данных (или ответов) именуются 'answer<номер теста>.txt'.

Названия файлов для входных данных и выходных данных должны быть указаны через командную строку. Кроме того, в параметрах командной строки могут указываться не только имена файлов и директорий, но и относительные/абсолютные пути к ним. Например, так будет выглядеть тестирование программы BuzulukovAlexey_161_3.cs:

csc BuzulukovAlexey_161_3.cs

BuzulukovAlexey_161_3 data test1.txt answer1.txt

*либо*

BuzulukovAlexey_161_3 path/to/data/ path/to/test1.txt path/to/answer1.txt

где

1-ый параметр – путь до директории с данными (csv-файлы),

2-ой параметр – путь к входному файлу,
3-ий параметр – путь к выходному файлу.

Пример использования аргументов командной строки в C#:
https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/main-and-commandargs/command-line-arguments

## Литература


- Лекция от 06 октября 2020 года по курсу Алгоритмы и структуры данных
- Abadi, D. J. (2008). Query execution in column-oriented database systems (Doctoral dissertation, Massachusetts Institute of Technology). (**invisible join**)
- Chambi, S., Lemire, D., Kaser, O., & Godin, R. (2016). Better bitmap performance with roaring bitmaps. Software: practice and experience, 46(5), 709-719. (**Roaring bitmaps**)