Введение

- ❏ Репозиторий для сдачи: http://gitlab.atp-fivt.org/hobod2021/XXXXX-nosql
- ❏ Ветки: **nosqltask1** (для writer'a)**, nosqltask2** (для reader'a)**.**

Тестов для данного задания нет, проверка будет осуществляться вручную с помощью запуска файла run.sh.

Сроки

- ➢ Soft deadline: 11.05, 23:59.
- ➢ Hard deadline: 16.05, 23:59.

Общее описание

Для решения задач следует использовать **Apache HBase или Cassandra**. Программы могут быть реализованы: на Java API или Python (библиотека HappyBase для HBase).

Данные необходимо преобразовать с помощью Hadoop или Spark, положить в удобном для решения задачи виде в базу HBase или Cassandra и уметь получать из базы результат. Таким образом, решение состоит из двух программ:

- **[1 балл]** обработчика (writer), который преобразует исходные данные и сохраняет результат. Т.к. пишем мы реже, чем читаем, то основная логика должна быть реализована во writer'e.
- **[0,5 балла]** клиент (reader) для отображения результата. При этом клиент может содержать дополнительную логику по преобразованию данных (если необходимо).

Обе программы – консольные приложения, reader выводит результат на консоль.

Важно! Настолько, что это является необходимым условием сдачи задания. Выбирайте наиболее удобную структуру таблицы в базе для вашей задачи. Используйте то свойство, что семейств колонок обычно немного, а колонок в семействе может быть очень много. Причем у разных строк могут быть разные колонки.

Входные данные

Логи матчей игры PUBG, подробное описание здесь.

Путь в HDFS: **`/data/hobod/pubg`**.

Описание полей в датасете

Поле	Описание
killed_by	Тип оружия, которым был убит игрок.
killer_name	Кто убил игрока
killer_placement	Рейтинг в матче (по кол-ву фрагов).
killer_position_x, killer_position_y	Где был убит игрок.

map	На какой карте проходит матч (для задач неактуально).
match_id	Уникальный ID матча.
time	Время от начала матча (в секундах).
victim_name, victim_placement, victim_position_x, victim_position_y	Аналогичные поля, но для убитого игрока.

Задания

Вариант определяется по формуле: `int('hob2021XX'[3:]) % 4 +1`.

Задание 1

Клиент должен уметь выводить TOP-10 областей по кол-ву киллов за первые 10 минут в заданном матче (матч задается по ID). Позиция задаётся парой (killer_position_x, killer_position_y). Областью в данном случае назовём круговой участок карты, в котором:

$$A \leq (x - 400000)^2 + (y - 400000)^2 \leq A + 100$$

ID области вычисляется по числу A. Число A изменяется с шагом 100. Т.е. если A=0, то у области будет ID=0, если A=100, то у области будет ID=1, и так далее. Области отсортировать по количеству киллов.

Задание 2

Клиент должен уметь для каждого матча выводить TOP-10 типов оружия, с которых больше всего убивали игроков за определённый период времени. Период времени задаётся парой (start, end).

Задание 3

Клиент должен уметь выводить TOP-10 типов оружия, с которых больше всего убивали игроков в каждом матче в определённой области. Область в данном случае задаётся 4 параметрами: x_min, y_min, x_max, y_max. По полям killer_position_x и killer_position_y можно определить, где произошёл kill.

Задание 4

Клиент должен уметь выводить распределение количества убийств игроков по типам оружия в указанный период времени (в секундах от начала матча).

Техническая информация

Для HBase

Разрабатывайте и запускайте программы на `mipt-client.atp-fivt.org`. Там все настроено и установлено (доступ к HBase, happybase).

Себе в .profile в домашней директории на сервере добавьте:

```
export
HADOOP_CLASSPATH=/etc/hbase/conf:/opt/cloudera/parcels/CDH/lib/hbase/lib/core-3.1.1.jar
:/opt/cloudera/parcels/CDH/lib/hbase/lib/guava-12.0.1.jar:/opt/cloudera/parcels/CDH/lib
/hbase/hbase-protocol.jar:/opt/cloudera/parcels/CDH/lib/hbase/lib/htrace-core.jar:/opt/
cloudera/parcels/CDH/lib/hbase/lib/protobuf-java-2.5.0.jar:/opt/cloudera/parcels/CDH/li
b/hbase/hbase-client.jar:/opt/cloudera/parcels/CDH/lib/hbase/hbase-common.jar:/opt/clou
dera/parcels/CDH/lib/hbase/lib/zookeeper.jar:/opt/cloudera/parcels/CDH/lib/hbase/hbase-
server.jar:/opt/cloudera/parcels/CDH/lib/hbase/hbase-hadoop-compat.jar:/opt/cloudera/pa
rcels/CDH/lib/hbase/lib/high-scale-lib-1.1.1.jar:/opt/cloudera/parcels/CDH/lib/hbase/li
b/metrics-core-2.2.0.jar
```

Это нужно, чтобы HBase классы были доступны при выполнении команды hadoop. Потому что если вы пишете код, который сначала запускает MapReduce задачу, а потом складывает результат в HBase (а этот вариант предпочтителен), то это будет код, похожий на задание 1 и запускаться он будет с помощью hadoop. Просто по окончании расчета будет обращение к HBase.

Если в программе нет взаимодействия с Hadoop, то достаточно запустить так:

```
HBASE_CLASSPATH=$HBASE_CLASSPATH:./build hbase client.PutExample
```

При работе с Java API не путайте объекты Configuration - один будет для Hadoop, другой для HBase (HBaseConfiguration). Это разные объекты, первый - для создания Job, другой - для соединения с HBase.

Полезные материалы

- Разбор доп. задачи из семинара по HBase.
- Книга HBase The Definitive Guide - в ресурсах курса (вкладка Resources вверху страницы). В книжке есть ссылка на примеры: https://github.com/larsgeorge/hbase-book/. Примеры непосредственно из текста книги тоже можно использовать с поправкой на то, что HBase Java API с момента издания менялось, поэтому сверяйтесь с актуальным: http://hbase.apache.org/apidocs/
- Документация по HappyBase (Python модуль для работы с HBase): http://happybase.readthedocs.org/en/latest/
- Примеры программ также есть на кластере в папке `/home/velkerr/seminars/hbase` (локальная файловая система).
    - Пример записи в HBase написан на Java API,
    - Пример чтения с фильтрами - на Python с использованием библиотеки happybase.

Для Cassandra

- Примеры работы с Cassandra из Spark
- Материалы семинара по Cassandra
- Документация по Cassandra query language (CQL).