## Задания для оборочного тура по Хакатону «Большие данные» в МФТИ

За каждое задание, выполненное даже частично, начисляются баллы. **Необязательно решить все приведенные задания**. Отбор будет осуществляться по набранным баллам. Задание «Подготовка к Хакатону» желательно для выполнения, для эффективного использования времени на Хакатоне.

Формат решения:

По каждому пункту

1. Работающая программа/скрипт
2. Исходный код и все линкующиеся библиотеки, если есть
	1. Инструкции по сборке, если нужна; сборка должна осуществляться успешно.
3. Эксплуатационная документация (может быть просто: запустить скрипт, параметры, …)
	1. С описанием допущений и ограничений
4. Когда необходимо: тестовые данные и/или скрипты создающие тестовые данные (схемы, таблицы, данные, …)

#### Основное задание по Базам Данных

**Задание 0.1 Дубли**

В таблице две колонки с натуральными числами – на SQL найти дубли.

**Задание 0.2 Выборка клиентов без счетов**

Есть таблица с клиентами с id\_client и таблица со счетами, связанная с таблицей клиентов по id\_client.

1. Предложить от одного до трех вариантов запроса на SQL для поиска и удаления счетов, у которых нет клиентов.
2. В таблице клиентов 100 млн. записей, в таблице счетов 1 млрд.
	1. Выбрать и оптимизировать самый производительный запрос.

**Задание 0.3 3. Колодцы на планете**

1. Спроектировать БД для хранения карты планеты, разбитой на сектора (заданного размера, например 1км\*1км)
2. Сектор с колодцем помечается признаком – подготовить тестовые данные
3. Разработать несколько вариантов поиска ближайшего колодца (расстояние по прямой) (на вход: сектор текущего местоположения)
	1. Выбрать самый оптимальный из вариантов – обосновать
	2. Оптимизировать выбранный алгоритм
	3. Оптимизировать структуру хранения

#### Последующие задания необязательные к решению, но приносят дополнительные баллы.

#### Задание 1 Разработка Хранилищ Данных

**Задание 1.1 Версионный контроль вообще**

Написать программу, которая для определенной ветки *Git (любой системы контроля версий)*:

определяет и выдает, в виде отчета, список всех удаленных, измененных и добавленных файлов.

(если файл был сначала добавлен, потом удален – его выдавать не надо).

Придумать: как отображать переименованные файлы.

Можно сделать два варианта – либо за всю историю ветки, либо за какой-то интервал дат.

**Задание 1.2 Версионный контроль Баз Данных**

Есть таблица *TAB* в схеме *PROM*, содержит миллион записей. Есть таблица *TAB* в схеме *DEV*, с более новой структурой, с данными, которые переносить на *PROM* не следует.

Необходимо написать программу или скрипт, которая по системному словарю СУБД будет находить различия между таблицами и генерировать скрипт преобразования структуры из старой в новую.

В скрипте должны быть команды:

- ALETER TABLE TAB CHANGE COLUMN…

- ALETER TABLE TAB ADD COLUMN…

- команды (пере)создания индексов и ключей;

- команды переноса данных;

- ALETER TABLE TAB DROP COLUMN

- …

Данными командами не обязательно ограничиваться, и можно предложить альтернативные.

Обязательно учесть следующий случай:

Расщепляется поле FULL\_PHONE на PHONE\_PREFIX и PHONE\_NUMBER:

ALTER TABLE TAB ADD COLUMN PHONE\_PREFIX VARCHAR(10);

ALTER TABLE TAB ADD COLUMN PHONE\_NUMBER VARCHAR(20);

UPDATE TAB SET PHONE\_PREFIX = SUBSTR(FULL\_PHONE, 1, 10), PHONE\_NUMBER = SUBSTR(FULL\_PHONE, 11) WHERE FULL\_PHONE is not null;

ALTER TABLE TAB DROP COLUMN FULL\_PHONE;

Дополнительные баллы: написать скрипт переноса данных с PROM на DEV. На DEV потенциально изменённая структура БД. Учесть изменение типа поля и расщепление полей (в ту и обратную сторону).

Дополнительные баллы: написать скрипт проверки переноса данных.

**Задание 1.3 Версионный контроль Баз Данных – сложная**

Есть несколько команд, разрабатывающих одновременно, как DDL, так и DML, хранимые процедуры, …

Предложить процесс организации стендов разработки и версионного контроля кода.

И реализовать скрипты (за каждый скрипт и его функциональность – дополнительные баллы):

- merge DDL и прочего кода;

- поиска конфликтов;

- автоматического решения конфликтов ;

- генерации alter скриптов с безопасной миграцией данных на новую версию БД;

- контроль корректности перехода и переноса данных на новую версию БД;

- какие еще скрипты для автоматизации рутинных задач, ускорения разработки и повышения качества можете предложить (дополнительные баллы).

Описать допущения и требования к организации процесса и функционирования скриптов.

**Задание 1.4 Версионный контроль Хранилищ Данных**

В дополнение к контролю версий Базы данных, разработать подход и необходимые скрипты для проверок, переноса, merge, … кода для компонент:

* ETL типа Informatica Power Center, формат файлов: xml;
	+ ETL/ELT на хранимых процедурах;
* скрипты на Python;
* jar, разработанные на Java;
* отчеты на SAP Business Object (можно использовать его API);
	+ предложить реализацию сравнения/выявления изменений между biar и между universe разных версий
* отчеты на Qlik View и Sense.

Скрипты для создания инкрементальных патчей (дистрибутивов).

Скрипты для создания кумулятивных патчей (дистрибутивов).

Описать допущения и требования к организации процесса и функционирования скриптов.

#### Задание 2 Модульное тестирование Хранилищ Данных

**Задание 2.1 Организация модульного тестирования Хранилищ Данных**

* Предложить подходы к тестированию хранилища данных, а именно:
* Какие виды тестов можно реализовать для основных сущностей хранилища: таблиц, представлений, процедур?
* Какие виды тестов для загрузки данных (ETL), расчёта витрин и построения отчетов?
* По какому принципу разделять хранилище данных на модули?
* Осуществить разделение на модули:
 перечислить модули и основную функцию, а также какие есть связи с каким модулями.
	+ Над какими модулями, их интерфейсами, и на каком этапе, проводить модульное тестирование?
	+ Упорядочить по убыванию приоритета необходимости тестирования.
* Организация стендов и контроля версий для разработки хранилища данных несколькими командами; для модульного тестирования, интеграционного тестирования, …
* С чем (какими данными, эталонами, прототипами и т.д.) проводить сравнение полученных результатов при проведении модульных тестов?
	+ Будут ли различия при тестировании нового функционала или изменении старого?
* Как можно протестировать датафикс (только изменение данных) в хранилище данных?

**Задание 2.2 Практическая реализация модульного тестирования Хранилищ Данных**

* Предложить и реализовать фреймворк для модульного тестирования.
	+ Необходимо ли закупить какое-то ПО? Реализовать на демо версии.
	+ На основе OpenSource.
	+ Разработать свой.
* Реализовать фреймворк тестирования datafix.
* Автоматизировать регрессионное интеграционное тестирование.
* Дополнительный уровень: фреймворк должен предоставлять возможности для хранения с версионным контролем: скриптов, их параметров, расписания запуска, журналов запусков и результатов, …

Описать допущения и требования к организации процесса и функционирования фреймворков.

#### Задание 3 Скрининг оборотов

1. На основе операций по счетам клиентов (юр.лица) построить индикатор оттока.

Тут можно проверять сигналы из вложения: 

Можно упростить или сделать меньшее кол-во, или предложить свои.

На вход

• Одна широкая таблица с платежными операциями клиентов

На выходе

• Выдавать клиентов, для которых сработало не менее 2-х сигналов

Задача требует подготовки обезличенного, но репрезентативного набора данных.

#### Задание 4 Проверка данных

Реализация автоматической проверки данных при ежедневной инкрементальной загрузке остатков и проводок.

 На вход

• Таблица t1 с остатками по всем балансовым счетам (остатки хранятся периодами, реализована историчность, актуальность данных T-1)

• файл или stg-таблица s1 с остатками на дату T

• файл или stg-таблица s2 с проводками, совершенными за дату T

На выходе получить

• скрипт загрузки данных из s1 в t1.

• Должна производиться сверка: исходящий остаток по всем активам = исходящий остаток по всем пассивам.

Если сверка выполняется успешно, то производится загрузка. В противном случае – произвести анализ проводок (из них выделить обороты в разрезе каждого счета), в итоге выдать сообщение, какие проводки приводят к ошибке.

Задача требует подготовки обезличенного, репрезентативного набора данных.

#### Задание 5 Подготовка к хакатону

Спроектировать и подготовить скрипты создания Базы данных, а также генерации и наполнения ее репрезентативными данными для разработки и тестирования.

БД: нормализованная реляционная БД (3-я или 4-я форма).

Пример таблиц и связей: *клиенты*-*М-к-М*->*договора*-*М-к-М*->*счета*->*проводки*.

Если нужны: вспомогательные таблицы типа: *остатки*, *обороты*, …

Справочные таблицы типа: *регионы*, *города*, *филиалы*, *отделения*.

Данные историчны (встречается как история «на дату», так и «за период»).
*Историчность – это возможность написать запрос к данным на дату и/или за период.*

Поля таблиц содержат как значения (даты, числа, строки и т.д.), так и справочные значения (ссылки на выделенные таблицы-справочники).

Порядок чисел записей – от сотни тысяч до миллиона строк.

### Средства и технологии

**Серверное и клиентское ПО:** должно быть установлено и настроено на ноутбуках команд.
Oracle server:

<http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html>
Oracle client:
<http://www.oracle.com/technetwork/database/enterprise-edition/downloads/112010-win32soft-098987.html>
нужно скачать Oracle Client для своей архитектуры (32/64 бит).

Можно использовать и другие СУБД.

Возможны предустановленные дополнительные инструменты для разработки и тестирования.