**Задание 1. Bash-скрипты.**

Необходимо написать параметризированный bash-скрипт «search\_by\_path.sh», основной задачей которого является получение информации о txt-файлах в указанной директории.

**Входящие параметры bash-скрипта:**

1. PATH - путь к директории, в которой находятся папки и файлы, обязательный
2. DATETIME – дата и время последнего изменения файла, обязательный
3. OUT\_CSV\_FILE\_NAME – путь и название csv-файла с результатами обработки, обязательный

**Функциональные требования к bash-скрипту:**

1. В указанной директории (PATH) найти все txt-файлы, которые были изменены или созданы начиная с даты (DATETIME) включительно, и вывести их в консоль терминала в формате: полный путь к файлу (path, название, расширение)
2. Получить указанные ниже атрибуты для каждого txt-файла и записать их в результирующий файл OUT\_FILE\_NAME в формате CSV. Порядок атрибутов соответствует порядку столбцов в результирующем файле, название столбцов не указывать:

- дата сканирования;

- путь до файла;

- название файла;

- дата-время последнего изменения файла;

- дата-время последнего доступа к файлу;

- размер файла(Mb);

- число строк в файле;

Желательно выполнить задание с использованием пользовательских функций и дескрипторов ввод-вывод.

**Требования к результату:**

В качестве результата ожидается файл с названием <Фамилия Имя Отчество>\_task01.txt, в котором будет содержаться полный текст скрипта и команда для его запуска с краткими комментариями.

**Задание 2. Регулярные выражения.**

На входе вашего ETL-процесса, есть txt-файл (case02\_file.txt), полученный от системы хранения метаинформации. Система имеет свой формат выгрузки данных, которые изменить невозможно.



Ваша задача – написать bash-скрипт/команду для получения из файла всех текстовых идентификаторов систем в виде очищенного списка.

Текстовым идентификатором систем считается значение, заключенное в квадратные скобки, идущее после комбинации символов “ID: ”.

*Пример:* [ID: WAREHOUSE], значение WAREHOUSE.

**Требования к результату:**

В качестве результата ожидается файл <Фамилия Имя Отчество>\_task02.txt, в котором будет содержаться полный текст bash-скрипта или команда с краткими комментариями.

**Задание 3. Работа с Python.**

В качестве источника данных у вас есть созданный сотрудником компании excel-файл case03\_input\_file.xlsx, содержащий в себе коэффициент эффективности партнёрской сети в стандартной для всех партнеров объемов продаж.



К сожалению, данные в файле структурированы не самым оптимальным для загрузки в hdfs образом. См. лист data.

**Описание файла:**

1. Размер таблиц всегда одинаковый (9 строк, 9 столбцов).
2. В одном файле содержится только один лист с данными и в нем может быть более 10 000 таблиц.
3. Таблицы всегда разделены пустой строкой.
4. Значения «диапазон», всегда одинаковы: 0 – 10, 100 – 500, 500 - 1 000, 1 000 - 5 000, 5 000 - 10 000, 10 000 - 100 000, > 100 000
5. Название региона, всегда содержится в первой строке – в первом столбце таблицы

Ваша задача - написать скрипт обработки данного файла с использованием языка Python (3.х). Результатом обработки должен быть csv файл, состоящий из следующих столбцов:

1. File\_Name
2. Region
3. Partner
4. Range,Value

Имена столбцов должны содержаться в первой строке файла. Пример см. на листе «Result».

**Требования к результату:**

В качестве результата ожидается файл <Фамилия Имя Отчество>\_task03.py, в котором будет содержаться код с краткими комментариями

**Задание 4. SQL Запросы. Генерация данных.**

Для решения заданий Вам потребуется скачать и установить любую бесплатную РСУБД (SQL Server Developer, PostgreSQL, MySQL, Oracle XE и т.п.)

В базе-источнике, есть три таблицы:

1. Таблица с данными о магазинах сети - «**stores**»:

* **store\_id** – id магазина (целочисленный тип)
* **store\_name** – Название магазина
* **store\_region** – регион магазина
* **store\_id** - primary key таблицы stores

1. Таблица с данными о продажах во всех магазинах сети - «**sales**»:

* **check\_num** - Номер Чека (целочисленный тип)
* **sales\_date** – год, месяц, день продажи, без времени.
* **store\_id** – id магазина, foreign key stores.store\_id
* **good\_name** – Название товара
* **s\_count** – количество товара в позиции (целочисленный тип)
* **s\_sum** – сумма продажи (целочисленный тип, для упрощения)
* primary key таблицы sales – это набор полей [check\_num, sales\_date, store]

1. Таблица с данными о входе-выходе сотрудников из магазинах сети - «**store\_acs**»:

* **store\_id** - id магазина, foreign key stores.store\_id
* **employee\_id** сотрудник (целочисленный тип)
* **event\_ts** – дата-время входа/выхода сотрудника из магазина.
* **event\_type** – тип события ( 1 – вход сотрудника, -1 – выход сотрудника)

С помощью SQL-запросов, используя циклы и рекурсивные CTE, сгенерируйте данные в таблицах, удовлетворяющие следующим условиям:

1. «**stores**»: 1-3 магазина в 5 разных регионах. Форматы: Регион01,Регион02…Регион05, Магазин01, Магазин02…
2. «**sales**»: 20 уникальных товаров в формате: товар01, товар02… товар20.
3. «**sales**»: продажи за 3 месяца, продажи каждого из 20-ти товаров не менее 5 штук в день в каждом магазине.
4. Таблицу «**store\_acs**» достаточно заполнить инструкцией INSERT за одну дату для двух магазинов. Ограничения: рабочий день с 9 до 21 – для всех магазинов, приход – уход не регламентирован, каждый сотрудник минимум один раз выходит из магазина.

**Требования к результату:**

В качестве результата ожидается файл <Фамилия Имя Отчество>\_task04.sql, в котором будет содержаться код создания таблиц, генерации данных и краткие комментарии.

**Задание 5. SQL Запросы. Продажи товар01 и товар02.**

Для решения задания Вам потребуются созданные в задании 4 таблицы.

Напишите один запрос, который вернёт полную сумму чеков в разрезе по месяцам (строки) для следующих 4 –х условий (столбцы):

1. Одновременно продан товар01 и товар02.
2. Продан только товар01 (вместе с ним могут быть любые другие товары, кроме товар02)
3. Продан только товар02
4. Ни товар01, ни товар01 не присутствуют в чеке.

Для удобства тестирования, рекомендуется добавить тестовые записи в таблицу **sales**.

**Требования к результату:**

В качестве результата ожидается файл <Фамилия Имя Отчество>\_task05.sql, в котором будет содержаться код и краткие комментарии.

**Задание 6. SQL Запросы. Учет рабочего времени.**

Для решения задания Вам потребуется созданная в задании 4 таблица **store\_acs**. Внимательно ознакомьтесь с требованиями к заполнению таблицы данными.

Напишите запрос, который предоставит информацию по количеству сотрудников на каждый час рабочего дня, в каждом магазине за выбранную дату. В качестве упрощения, используйте только те магазины, для которых вы сгенерировали данные. В комментариях необходимо указать, какое условие нахождения сотрудника в магазине вы используете при подсчёте.

**Требования к результату:**

В качестве результата ожидается файл <Фамилия Имя Отчество>\_task06.sql, в котором будет содержаться код и краткие комментарии.

**Задание 7. SQL Запросы. Аналитика продаж.**

Для решения задания Вам потребуются созданные в задании 4 таблицы **stores** и **sales**.

Напишите один запрос, который вернет информацию о ТОП 3 товарах по сумме продаж для каждого региона в разрезе месяцев, а также:

* % от суммы продаж каждого товара в данном регионе за год
* % от суммы продаж каждого товара по всей сети за месяц
* % от суммы продаж всех товаров за месяц в данном регионе.

**Требования к результату:**

В качестве результата ожидается файл <Фамилия Имя Отчество>\_task07.sql, в котором будет содержаться код и краткие комментарии.