In [2]:
```
import pandas as pd
```

In [11]:
```
df = pd.read_csv(r'shedulers.csv', encoding='cp1251', sep=';')
df.head(10)
```

Out[11]:

	ФИО	Расписание	Дата начала расписания	Дата окончания расписания
0	Поставщик 1	дддвсвнн	01.01.2019 0:00	10.01.2019 0:00
1	Поставщик 1	ннвннв	11.01.2019 0:00	15.01.2019 0:00
2	Поставщик 1	св	16.01.2019 0:00	20.01.2019 0:00
3	Поставщик 2	свсвсв	01.01.2019 0:00	07.01.2019 0:00
4	Поставщик 2	днвсв	08.01.2019 0:00	14.01.2019 0:00
5	Поставщик 2	ннддвсв	15.01.2019 0:00	31.12.9999 0:00
6	Поставщик 3	нвнвнв	01.01.2019 0:00	01.02.2019 0:00
7	Поставщик 3	двдвдвдв	02.02.2019 0:00	31.12.9999 0:00

Пример - есть расписания поставщиков заданное видом ДДННССВВ, где Д - дневная смена, Н - ночная смена, С - суточная смена, В - выходной

Считать что дневная смена с 08:00 по 20:00 Считать что ночная смена с 20:00 по 08:00 Считать что суточная смена с 08:00 по 08:00 завтрашнего дня

Описание работы расписания ДНСВ: 1 день - Д дневная смена 2 день - Н ночная смена 3 день - С суточная смена 4 день - В выходной день 5 день - Д дневная смена и т.д. циклично

Задание 1

Создать таблицу (T_CONTRACTOR_SHERULER) под расписание и заполнить его с файла schedulers.csv. (Использовать любую библиотеку для заливки в СУБД (как вариант MS SQL)) Таблицу следует нормализовать.

- ID_NAME - идентификтор поставщика
- NAME - название поставщика
- SHEDULE - расписание
- DATE_BEGIN - дата начала действия расписания
- DATE_END - дата окончания действия расписания

Пример записи без нормализации:

- Поставщик 1 ДВС 01.01.2019 04.01.2019
- Поставщик 2 НВС 05.01.2019 31.12.2019

Вводные:

- Связку полей FIO, DATE_BEGIN считать уникальной.
- DATE_BEGIN не может привышать DATE_END.
- Можете продемонстрировать работу с ключами/ограничениями.

Задание 2

Создать таблицу (T_CONTRACTOR_WORK_DAY) выходов на работу сотрудников.

Таблица должна иметь следующий вид

- ID - идентификатор записи
- NAME - название поставщика
- DATE_BEGIN - Начало рабочего дня (datetime)
- DATE_END - Конец рабочего дня (datetime)

Задание 3

Создать процедуру расчета рабочих дней.

Входящие параметры:

- Дата начала периода расчета
- Дата окончания периода расчета.

Ожидаемый результать выполнения хранимой процедуры - заполнение таблицы T_CONTRACTOR_WORK_DAY рабочими днями согласно расписания работы поставщиков из таблицы T_CONTRACTOR_SHERULER Выходные дни (В) не должны попадать в таблицу T_CONTRACTOR_WORK_DAY

Пример выполнения для Поставщика 1 (Из примера записи таблицы T_CONTRACTOR_SHERULER) с параметрами '01.01.2019' - '08.01.2019' таблица T_CONTRACTOR_WORK_DAY заполнится следующими данными:

- 1 Поставщик 1 01.01.2019 08:00 01.01.2019 20:00
- 2 Поставщик 1 03.01.2019 08:00 04.01.2019 08:00
- 3 Поставщик 1 04.01.2019 08:00 04.01.2019 08:00
- 4 Поставщик 1 05.01.2019 20:00 06.01.2019 08:00
- 5 Поставщик 1 07.01.2019 08:00 08.01.2019 08:00
- 6 Поставщик 1 08.01.2019 20:00 09.01.2019 08:00

Пояснение: для записей с 01.01.2019 по 04.01.2019 берется расписание ДВС

- 1 - Д - дневная смена далее следует выходной В - запись о выходном дне не попадает в таблицу
- 2 - С - суточная смена
- 3 - расписание закончилось, поэтому оно циклично начинается с начала (Д - дневная смена)
- 4 - С 05.01.2019 начинает действовать новое расписание - НВС берется Н - ночная смена далее следует выходной В - запись о выходном дне не попадает в таблицу
- 5 - С - суточная смена
- 6 - Снова Н - ночная смена

Задание 4

С помощью SQL запросов:

- Сделать выборку содержащую сколько рабочих дней было у каждого поставщика
- Сделать выборку поставщиков, у которыйх было больше 10 рабочих дней за январь 2019 года
- Сделать выборку поставщиков, кто работал 14, 15 и 16 января 2019 года

Задание 5

Выполните задачи 3, 4 с помощью python.

Формат ответа:

- Необходимо вернуть .ipynb с кодом заливки данных и скрипты .sql для создания всех объектов и выполнения всех запросов. Также можно приложить ссылку на github.
- Выполнение заданий будет оцениваться с точки зрения лаконичности/элегантности кода и глубины Вашего знания sql.