Задание 1. «Отчёт о продажах в Телеграм»

Задачи:

1. Выберите 3 метрики, которые помогут руководителю отдела продаж контролировать ситуацию ежедневно — всё ли идёт нормально. Объясните свой выбор.
2. Напишите Телеграм Бота (скрипт), который будет отправлять ежедневный отчёт по этим метрикам в Телеграм руководителю. Чтобы показать, как работает бот вставьте его код в файл с ответом и прикрепите скриншот отправленного им сообщения, чтобы было видно от кого это. Под названием отправителя должно быть написано Бот, как на скриншоте ниже.

Задача 1.

При выборе метрик в первую очередь будем учитывать ситуацию в которой оказался Руководитель продаж, а именно:

- отсутствие сквозной аналитики
- отсутствие централизованного DWH или сложного BI
- необходимо выбрать только 3 метрики

Учитывая условие выбора только 3х метрик и отсутствия какой-либо аналитики у Руководителя продаж, выбранные метрики должны в первую очередь отвечать на один вопрос. И это будет вопрос: сколько? Соответственно это должны быть простые низкоуровневые метрики, помогающие Руководителю продаж держать руку на пульсе отдела.

Исходя из этого выбор метрик будет следующий:

1. Количество продаж

   Ключевой показатель, показывающий итог работы отдела продаж.

2. Средний чек

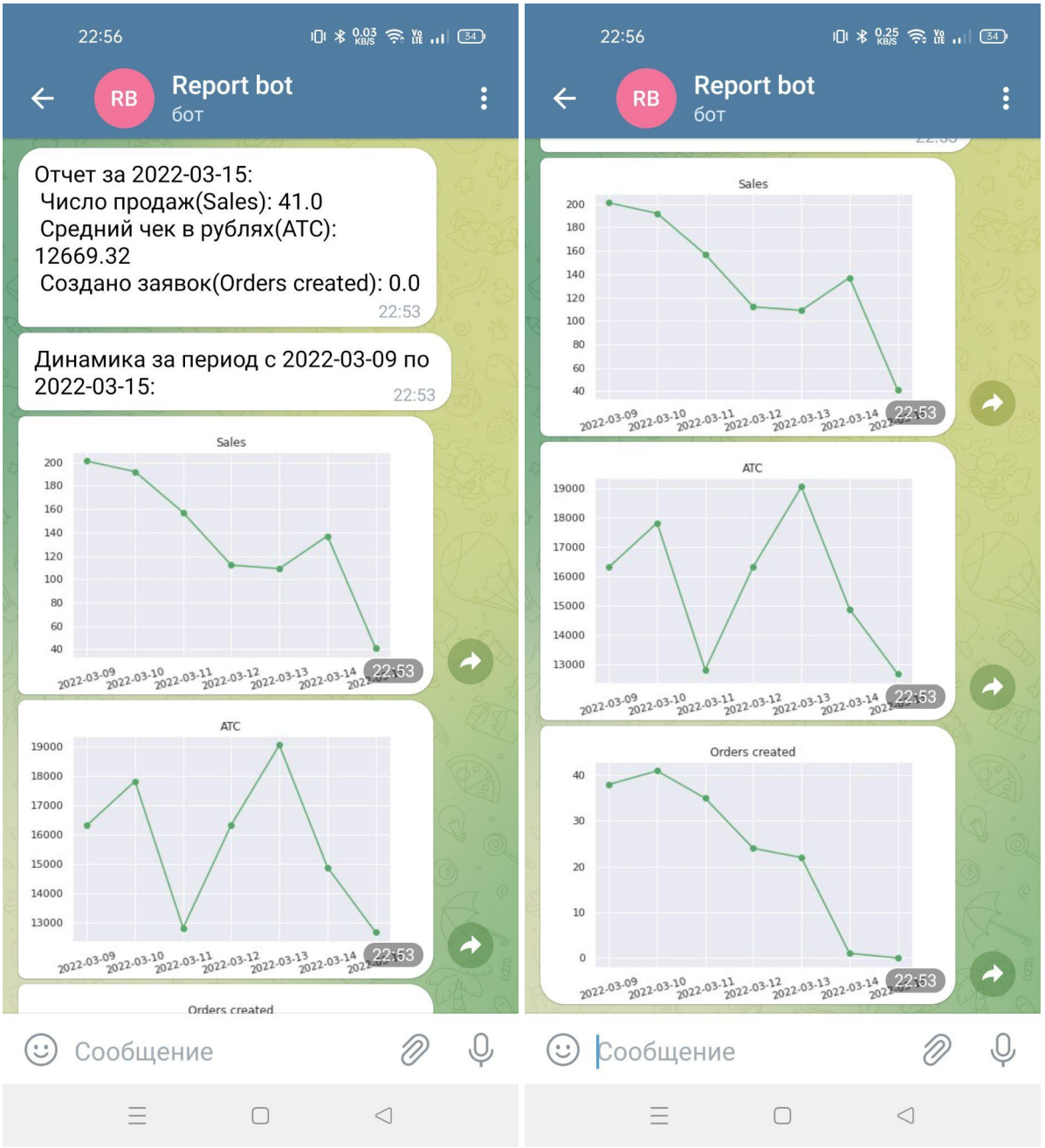
   Полагая, что в стартапе в области английского языка продают клиентам пакеты курсов различной стоимости. Данная метрика позволит понять какие пакеты пользуются большим спросом. Какая сумма для клиентов является более комфортной для заключения сделки. Также снижение среднего чека может является причиной нарушения работы по скрипту менеджеров по продажам.

3. Количество созданных заявок

   Данная метрика позволяет оценить насколько мощная формируется воронка продаж. Какие усилия предпринимают сотрудники для привлечения новых клиентов. Также позволяет оценить соотношение между количеством продаж и количеством поданных заявок.

Задача 2.

- Скриншот сообщения в телеграм, отправленного ботом:



Автоматизация отправки:

Для автоматизации отправки отчета в telegram, зальем код в GitLab и настроим ежедневную отправку отчета через CI/CD.

- Код:

```
# !pip install telegram
# !pip install python-telegram-bot
from sqlalchemy import create_engine
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import telegram
import io

sns.set()

def report(chat=None):
    ''' Функция создает запрос в БД и формирует отчет для отправки базовых
метрик в telegram '''
    chat_id = chat or **********
    # bot = telegram.Bot(token=os.environ.get("REPORT_BOT_TOKEN")) # вариант
шировка токена в GitLab
    bot = telegram.Bot(token='**********************************')

    # Подключение с параметрами:
    param = ('postgresql://{username}:{password}@{ipaddress}:{port}/{dbname}'
                                          .format(username='rouser',
                                                  password='ZI6MVnmi',
                                                  ipaddress='178.62.242.91',
                                                  port='5433',
                                                  dbname='quest-db'))

    # Метрика 1. Количество покупок клиентов в день
    query = '''
    SELECT DATE(transaction_created_at) as date,
          COUNT(amount) as count_buy
    FROM payments
    GROUP BY date
    ORDER BY date DESC
    '''
    # Датафрейм
    df_count_buy = pd.read_sql_query(query, create_engine(param))
    df_count_buy.set_index('date',inplace=True)


    # Метрика 2. Средний размер покупки в день
    query = '''
    SELECT date(transaction_created_at), currency, amount
    FROM payments
    '''
    # Датафрейм
```

```
    df_payments = pd.read_sql_query(query, create_engine(param))
    df_payments['amount'] = df_payments['amount'].astype('float')
    # курсы валют на 23.03.2022
    exc_rate = dict(zip(['RUB', 'EUR', 'GBP', 'PLN', 'USD', 'JPY', 'CLP',
'KRW', 'PEN', 'UAH', 'MXN', 'UYU'],
                        [1, 114.7 ,138 ,24.5, 104, 86, 0.13, 85.4, 27.5, 35,
5.1, 2.4]))
    # колонка с курсами валют
    df_payments['exc_rate'] = df_payments['currency'].map(exc_rate)
    # величина транзакций при переводе в рубли
    df_payments['amount_rub'] = df_payments['amount']*df_payments['exc_rate']

    # Средний размер покупки
    df_avg_buy = (df_payments.groupby('date')['amount_rub'].mean()
                                .sort_index(ascending=False)
                                .to_frame()
                                .rename(columns={'amount_rub':'avg_buy'}))


    # Метрика 3. Количество созданных заявок
    query = '''
    SELECT date(happened_at),
          _name as create_order
    FROM events
    LEFT JOIN events_dict
    ON events.event_id = events_dict.id
    WHERE _name = 'student_created_order'
    '''
    # Датафрейм
    df_creat_ord = pd.read_sql_query(query, create_engine(param))
    # Количество созданных заявок
    df_creat_ord = (df_creat_ord.groupby('date')
                              .count()
                              .sort_index(ascending=False))


    # Итоговый датафрейм с метриками
    df_common = (df_count_buy.merge(df_avg_buy, on='date', how='outer')
                           .merge(df_creat_ord, on='date', how='outer')
                           .fillna(0)
                           .rename(columns={'count_buy':'Sales',
                                            'avg_buy':'ATC',
                                            'create_order':'Orders created'}))


    # Отправка базовых метрик за последний день
    msg1 = '''Отчет за {day}:
\tЧисло продаж(Sales): {count_buy}
\tСредний чек в рублях(ATC): {avg_buy}
```

```
\tСоздано заявок(Orders created): {create_order}
                            '''.format(day = df_common.index[0],
                                count_buy = df_common.Sales[0],
                                avg_buy = round(df_common.ATC[0], 2),
                                create_order = df_common['Orders created'][0])
    bot.sendMessage(chat_id=chat_id, text= msg1)


    # Отправка недельных графиков по базовым метрикам
    msg2 = '''Динамика за период с {first} по {last}:'''.format(
                                                    first = df_common.index[6],
                                                    last = df_common.index[0])
    bot.sendMessage(chat_id=chat_id, text= msg2)
    n = 0
    for metric in df_common.columns:
        y = df_common.iloc[:7, n]
        x = df_common.index[:7]
        plt.title(metric)
        plt.xticks(rotation=15)
        plt.plot(x,y,'go-')
        n += 1

        plot_object = io.BytesIO()
        plt.savefig(plot_object)
        plot_object.name = 'plot.png'
        plot_object.seek(0)
        bot.sendPhoto(chat_id = chat_id, photo = plot_object)
        plt.close()


try:
    report()
except Exception as e:
    print(e)
```