Первоначально мною было задумано создание JSON формата с вложенными объектами вместо создания (одноступенчатых) json файлов для каждой из трех таблиц. По этой причине, я создала JSON для одной таблицы - REC\_EMPLOYEES, и обратное преобразование также только для данной таблицы.

Результат моего JSON преобразования – json\_result.txt.

Результат моих вставок в таблицу (JSON to SQL) – result\_jsonToSQL.jpg.

**ТАБЛИЦЫ ДЛЯ СИМУЛЯЦИИ**

Создание таблиц и заполнение их данными предоставлены в файле tables.sql (папка sqlToJSON). Данные таблицы были использованы для тестов и ориентира при написании скриптов.

**JSON ФОРМАТ**

Следующий формат был использован для формирования JSON строки для таблицы REC\_EMPLOYEES:

[

{

"dur": 5,

"transaction": {

"id": 3,

"start\_date": "20-MAY-2020,10:06:14",

"end\_date": "20-MAY-2020,10:10:04",

"amount": 30000,

"client\_id": 120,

"recycler": {

"start\_date": "20-APR-2020",

"office": "",

"city": "Moskow",

"rcl\_number": "30",

"id": 3

},

"process": "deposit",

"product": "kaspi gold"

},

"process": "deposit",

"screen\_name": "screen1",

"id": 1

},

…

]

Один объект "rec\_employees" представляет одну запись таблицы. Поле "tran\_id" развернуто с данными из таблицы "transaction". В свою очередь, поле "rcl\_id" из таблицы "transaction" развернуто согласно данным из таблицы "recycler".

**SQL to JSON**

Ни одно из готовых решений прямого преобразования не было выполнено успешно, т.к. подобные функции в PL/SQL действуют с версии Oracle 12.1.0.2.0. Таким образом, был создан собственный пакет для преобразования таблиц в JSON формат на PL/SQL (sqlToJson.sql).

Основные функции и процедуры пакета REC\_SCREEN\_JSON:

1. Функция "get\_recscr": принимает rec\_screen id, возвращает JSON строку CLOB для одной записи "rec\_screen".
2. Функция "get\_transaction": принимает transaction id, возвращает JSON строку CLOB для одной записи "transaction".
3. Функция "get\_recycler": принимает recycler id, возвращает JSON строку CLOB для одной записи "recycler".
4. Процедура "serialize\_table": процедура, которая создает JSON строку для всей таблицы "rec\_screen".

Процедура для функций "get\_recscr", "get\_transaction", "get\_recycler" практически идентична, но в функциях "get\_recscr" и "get\_transaction" применяется явный вызов зависимых функций для создания вложенного объекта ("get\_recscr" вызывает "get\_transaction", "get\_transaction" вызывает "get\_recycler").

Тип данных DATE записывается в двух форматах: 'DD-MON-YYYY', 'DD-MON-YYYY,HH24:MI:SS'. Последний формат был использован только для таблицы "transaction" (для полей start\_date, end\_date).

**ЗАМЕТКИ ДЛЯ PL/SQL СКРИПТА (SQL to JSON)**

Поля с NULL значениями заменены следующим образом:

1. Для строковых данных (date, varchar2, etc) null => "" (пустая строка)
2. Для числовых данных (number) null => 0. Мною предполагалось, что нумерация ID начинается с 1.

В моей реализации есть баг, связанный с присваиванием 0 к числовым значениям с null. Успешное присвоение происходит только для данных "rec\_screen", но не для вложенных объектов ("transaction", "recycler").

**JSON to SQL QUERY**

Обратная операция была выполнена на Python 3.6.

Вставка значений DATE в формате 'DD-MON-YYYY,HH24:MI:SS' не проходила успешно на Python после серий попыток, поэтому в файле data1.json (JSON строка, полученная после выполнения скрипта на PL/SQL) значения дат в данном формате были вручную обрезаны до формата 'DD-MON-YYYY'.

Дальнейшее описание представлено в скрипте jsonToSql.py.