# Задача №3

### Контекст

Представьте, что вы работаете в компании специализирующиеся на высокочастотной торговли. Текущий проект, на котором вы задействованы, предоставляет безопасную платформу для торгов в криптовалюте.

Непосредственно ваша текущая задача это миграция данных из одного хранилища в другое. Системой-источником выбран был [MongoDb](https://medium.com/nuances-of-programming/%D1%80%D1%83%D0%BA%D0%BE%D0%B2%D0%BE%D0%B4%D1%81%D1%82%D0%B2%D0%BE-%D0%BF%D0%BE-mongodb-6e844437b0de), а система-приемник у вас Postgres. Архитекторами проекта было решено воспользоваться встроенными механизмами переноса данных, а не писать свой отдельный сервис.

### Постановка

В Postgres имеется 3 таблицы, в которые делается вставка: \* auctioneer; \* attachment; \* bet.

Данные в источнике хранятся в коллекции bettings, примеры записей из этой коллекции:

*// запись № X:*  
{  
 "auctioneer": {  
 "firstname": "john",  
 "lastname": "doe",  
 "nickname": "mr.notknown",  
 "email": "john.doe@email.com"  
 },  
 "attachment": {  
 "filename": "key.token1",  
 "location": {  
 "datacenter": "AWS\_EU",  
 "localname": "iD1234sdv23r23rtfwv"  
 }  
 },  
 "bet": {  
 "volume": 0.00001,  
 "ts": 1672520400  
 }  
}  
  
*// запись № (X+1):*  
{  
 "auctioneer": {  
 "firstname": "john",  
 "lastname": "doe",  
 "nickname": "mr.notknown",  
 "email": "john.doe@email.com"  
 },  
 "attachment": {  
 "filename": "key.token2",  
 "location": {  
 "datacenter": "AWS\_US",  
 "localname": "iDasd@-asfasf23@3s"  
 }  
 },  
 "bet": {  
 "volume": 0.00085,  
 "ts": 1676928960  
 }  
}

**Небольшое объяснение**

Пользователи платформы (информация о пользователе — таблица auctioneer) делают ставки на аукционе (таблица bet). Из соображений безопасности во время произведения ставки пользователь должен приложить файл, полученный во время регистрации на аукцион (таблица attachment).

**Технические ограничения**

Внутренний механизм миграции: 1. делает вставку БД по одному документу из коллекции; 2. не имеет прав на вставку в таблицы auctioneer, attachment, bet; 3. может вставлять документы “как есть” в виде строки.

В силу этих ограничений вставка происходит через view v\_auction\_payload.

### Ожидаемый формат ответа

Ваше решение должно быть в виде дополнения к скрипту:

CREATE TABLE auctioneer (  
 event\_id INTEGER,  
 firstname TEXT,  
 lastname TEXT,  
 nickname TEXT,  
 email TEXT  
);  
CREATE TABLE attachment (  
 event\_id INTEGER,  
 filename TEXT,  
 datacenter TEXT,  
 localname TEXT  
);  
CREATE TABLE bet (  
 event\_id INTEGER,  
 volume NUMERIC,  
 ts TIMESTAMP  
);  
  
CREATE OR REPLACE FUNCTION auctioneer\_to\_json(event\_id INTEGER)  
 RETURNS TEXT  
 LANGUAGE plpgsql  
AS $$  
DECLARE  
 firstname\_ TEXT;  
 lastname\_ TEXT;  
 nickname\_ TEXT;  
 email\_ TEXT;  
BEGIN  
 SELECT  
 firstname, lastname, nickname, email  
 INTO  
 firstname\_, lastname\_, nickname\_, email\_  
 FROM  
 auctioneer  
 WHERE  
 auctioneer.event\_id = $1;  
 RETURN format(  
 '{"firstname": %I, "lastname": %I, "nickname": %I, "email": %I}',  
 firstname\_, lastname\_, nickname\_, email\_  
 );  
END;  
$$;  
  
CREATE OR REPLACE FUNCTION attachment\_to\_json(event\_id INTEGER)  
 RETURNS TEXT  
 LANGUAGE plpgsql  
AS $$  
DECLARE  
 filename\_ TEXT;  
 datacenter\_ TEXT;  
 localname\_ TEXT;  
BEGIN  
 SELECT  
 filename, datacenter, localname  
 INTO  
 filename\_, datacenter\_, localname\_  
 FROM  
 attachment  
 WHERE  
 attachment.event\_id = $1;  
 RETURN format(  
 '{"filename": %I, location": {"datacenter": %I, "localname": %I}}',  
 filename\_, datacenter\_, localname\_  
 );  
END;  
$$;  
  
CREATE OR REPLACE FUNCTION bet\_to\_json(event\_id INTEGER)  
 RETURNS TEXT  
 LANGUAGE plpgsql  
AS $$  
DECLARE  
 volume\_ NUMERIC;  
 ts\_ TIMESTAMP;  
BEGIN  
 SELECT  
 volume, ts  
 INTO  
 volume\_, ts\_  
 FROM  
 bet  
 WHERE  
 bet.event\_id = $1;  
 RETURN format(  
 '{"volume": %s, "ts": %s}',  
 volume\_, EXTRACT(EPOCH FROM ts\_)::BIGINT  
 );  
END;  
$$;  
  
  
CREATE OR REPLACE VIEW v\_auction\_payload(payload) AS  
 SELECT  
 format(  
 '{"auctioneer": %s, "attachment": %s, "bet": %s}',  
 auctioneer\_to\_json(bet.event\_id),  
 attachment\_to\_json(bet.event\_id),  
 bet\_to\_json(bet.event\_id)  
 )  
 FROM  
 auctioneer  
 JOIN  
 attachment  
 ON  
 auctioneer.event\_id = attachment.event\_id  
 JOIN  
 bet  
 ON  
 attachment.event\_id = bet.event\_id;

### Пример

Ваше дополнение к скрипту должно позволить получить следующий эффект

INSERT INTO v\_auction\_payload(payload) VALUES (  
'{  
 "auctioneer": {  
 "firstname": "john",  
 "lastname": "doe",  
 "nickname": "mr.notknown",  
 "email": "john.doe@email.com"  
 },  
 "attachment": {  
 "filename": "key.token1",  
 "location": {  
 "datacenter": "AWS\_EU",  
 "localname": "iD1234sdv23r23rtfwv"  
 }  
 },  
 "bet": {  
 "volume": 0.00001,  
 "ts": 1672520400  
 }  
}');  
INSERT INTO v\_auction\_payload(payload) VALUES (  
'{  
 "auctioneer": {  
 "firstname": "john",  
 "lastname": "doe",  
 "nickname": "mr.notknown",  
 "email": "john.doe@email.com"  
 },  
 "attachment": {  
 "filename": "key.token2",  
 "location": {  
 "datacenter": "AWS\_US",  
 "localname": "iDasd@-asfasf23@3s"  
 }  
 },  
 "bet": {  
 "volume": 0.00085,  
 "ts": 1676928960  
 }  
}');

SELECT \* FROM auctioneer;  
  
--  
  
1,john,doe,mr.notknown,john.doe@email.com  
2,john,doe,mr.notknown,john.doe@email.com

SELECT \* FROM attachment;  
  
---  
  
1,key.token1,AWS\_EU,iD1234sdv23r23rtfwv  
2,key.token2,AWS\_US,iDasd@-asfasf23@3s

SELECT \* FROM bet;  
  
---  
  
1,0.00001,2022-12-31 21:00:00.000000  
2,0.00085,2023-02-20 21:36:00.000000

SELECT \* FROM v\_auction\_payload;  
  
---  
  
"{""auctioneer"": {""firstname"": john, ""lastname"": doe, ""nickname"": ""mr.notknown"", ""email"": ""john.doe@email.com""}, ""attachment"": {""filename"": ""key.token1"", location"": {""datacenter"": ""AWS\_EU"", ""localname"": ""iD1234sdv23r23rtfwv""}}, ""bet"": {""volume"": 0.00001, ""ts"": 1672520400}}"  
"{""auctioneer"": {""firstname"": john, ""lastname"": doe, ""nickname"": ""mr.notknown"", ""email"": ""john.doe@email.com""}, ""attachment"": {""filename"": ""key.token2"", location"": {""datacenter"": ""AWS\_US"", ""localname"": ""iDasd@-asfasf23@3s""}}, ""bet"": {""volume"": 0.00085, ""ts"": 1676928960}}"

### Подсказки

* [Ознакомтесь с работой с JSON в Postgres](https://www.postgresql.org/docs/9.3/functions-json.html)
* Для нумерации событий стоит воспользоваться с ранее изученным объектом из предыдущего задания.
* Нумерация должна начинаться с 1.
* Для получения текущего момента времени можно воспользоваться функцией now().