Министерство науки и высшего образования РФ

федеральное государственное автономное  
образовательное учреждение высшего образования

«Омский государственный технический университет»

Факультет информационных технологий и компьютерных систем

Кафедра «Прикладная математика и фундаментальная информатика»

|  |
| --- |
| **Лабораторная работа №4** |
| по дисциплине **Системы управления базами данных**  **Тема: РАБОТА С МАССИВАМИ И ТИПАМИ JSON** |

|  |  |  |  |
| --- | --- | --- | --- |
| Студента | Кузиной Варвары Михайловны | | |
| Курс | 3 | Группа | МО-231 |
| Направление | 02.03.03 Математическое обеспечение и администрирование информационных систем | | |
| Руководитель | доц., к.н. | | |
|  | Моисеева Н.А. | | |
| Выполнил | 16.10 | | |
| Проверил |  | | |

Омск 2025

**Цель работы:**

Расширить предметную область *rgr\_store*:

1) добавить таблицу со столбцом-массивом;

2) добавить таблицу со столбцом JSON/JSONB;

3) заполнить данными;

4) выполнить демонстрационные запросы и проверки ограничений.

**Исходная модель:**

БД *rgr\_store*: employees, suppliers, product\_catalog, orders, order\_composition, receipts, check\_composition, invoices, invoice\_composition.  
Связи реализованы внешними ключами (с каскадами), есть CHECK/NOT NULL/PK.

**Задание 1. Таблица с массивом значений**

Создаём product\_media: для каждого товара — массив фотографий и набор тэгов. Типы text[]; CHECK гарантирует, что фото не пустой массив. Связь с product\_catalog через FK с каскадом удаления.

**SQL запросы:**

-- === Таблица с массивами: фото и тэги товаров ===

CREATE TABLE IF NOT EXISTS public.product\_media (

product\_code integer PRIMARY KEY

REFERENCES public.product\_catalog(product\_code)

ON DELETE CASCADE ON UPDATE CASCADE,

photos text[] NOT NULL, -- массив путей/URL фото

tags text[] DEFAULT '{}'::text[], -- массив тэгов (по умолчанию пустой)

CONSTRAINT pm\_photos\_not\_empty CHECK (cardinality(photos) > 0)

);

-- Наполнение

INSERT INTO public.product\_media(product\_code, photos, tags) VALUES

(100, ARRAY['/img/100-1.jpg','/img/100-2.jpg'], ARRAY['fruits','fresh'])

ON CONFLICT (product\_code) DO NOTHING;

INSERT INTO public.product\_media(product\_code, photos, tags) VALUES

(101, ARRAY['/img/101-1.jpg'], ARRAY['fruits','import'])

ON CONFLICT (product\_code) DO NOTHING;

INSERT INTO public.product\_media(product\_code, photos, tags) VALUES

(200, ARRAY['/img/200-1.jpg','/img/200-2.jpg'], ARRAY['dairy'])

ON CONFLICT (product\_code) DO NOTHING;

**Пояснения:**

* text[] — массив строк; ARRAY[...] — литерал массива.
* DEFAULT '{}'::text[] — по умолчанию пустой массив (каст :: → к типу text[]).
* cardinality(photos) > 0 — фотографий минимум одна.
* ON DELETE CASCADE — удаляешь товар → удаляются и его медиа-записи.

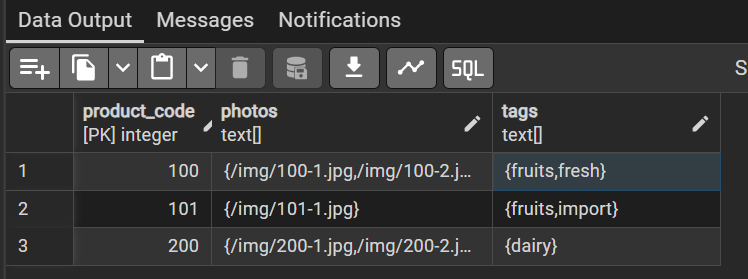


Рисунок 1— Наполнение таблицы продуктов

**Проверочные запросы (рисунок 2-4):**

-- Товары, у которых среди тэгов есть 'fruits'

SELECT pm.\*

FROM public.product\_media pm

WHERE pm.tags @> ARRAY['fruits'];

-- Развернуть фото по строкам

SELECT product\_code, UNNEST(photos) AS photo

FROM public.product\_media;

-- Количество фото на товар

SELECT product\_code, array\_length(photos, 1) AS photos\_count

FROM public.product\_media;

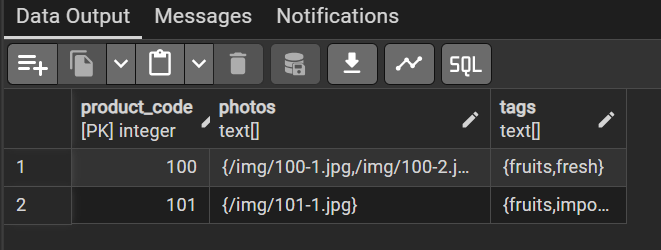
****

Рисунок 2 — Результат создания и вставки строк с массивами

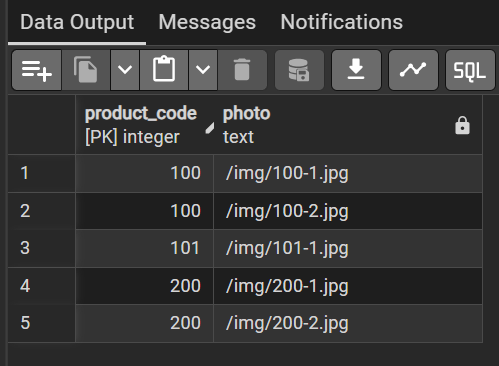


Рисунок 3 — Просмотр путей к фото по каждому товару в виде столбца

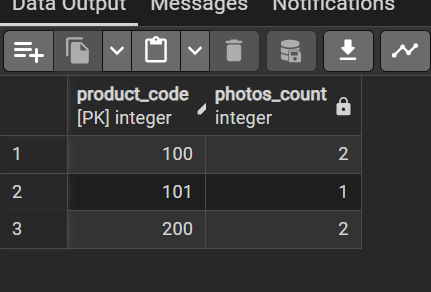


Рисунок 4 — Проверка соответствия строк ограничению, объявленному при создании таблицы

**Задание 2. Таблица со значениями JSONB**

Создаём receipt\_payments: параметры оплаты чека в формате jsonb. Мини-валидация «схемы» через CHECK: payment — объект, обязательно есть method и amount, method ∈ {cash, card, qr}, amount ≥ 0.

**SQL запросы:**

-- === Таблица с JSONB: параметры платежа по чеку ===

CREATE TABLE IF NOT EXISTS public.receipt\_payments (

receipt\_number integer PRIMARY KEY

REFERENCES public.receipts(receipt\_number)

ON DELETE CASCADE ON UPDATE CASCADE,

payment jsonb NOT NULL,

CONSTRAINT rp\_is\_object CHECK (jsonb\_typeof(payment) = 'object'),

CONSTRAINT rp\_has\_required CHECK (payment ? 'method' AND payment ? 'amount'),

CONSTRAINT rp\_method\_enum CHECK ((payment->>'method') IN ('cash','card','qr')),

CONSTRAINT rp\_amount\_non\_neg CHECK (((payment->>'amount')::numeric) >= 0)

);

-- Наполнение

INSERT INTO public.receipt\_payments(receipt\_number, payment) VALUES

(70001, jsonb\_build\_object(

'method','card',

'amount', 214.50,

'card\_last4','1234',

'bank','SBER'))

ON CONFLICT (receipt\_number) DO NOTHING;

INSERT INTO public.receipt\_payments(receipt\_number, payment) VALUES

(70002, jsonb\_build\_object(

'method','cash',

'amount', 328.90,

'change\_given', 71.10))

ON CONFLICT (receipt\_number) DO NOTHING;

**Пояснения:**

* jsonb — бинарный формат JSON (операторы, индексация).
* payment ? 'method' — ключ присутствует.
* payment->>'method' — текстовое значение по ключу method.
* (payment->>'amount')::numeric — приводим текст к числу для сравнения.

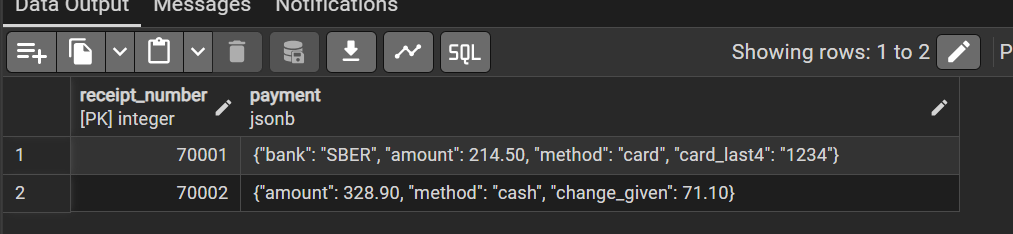


Рисунок 5 — Просмотр способов оплаты

**Проверочные запросы (рисунок 6-8):**

-- Чеки, оплаченные картой, с суммой

SELECT r.receipt\_number,

p.payment->>'method' AS method,

(p.payment->>'amount')::numeric AS amount

FROM public.receipt\_payments p

JOIN public.receipts r USING (receipt\_number)

WHERE p.payment->>'method' = 'card';

-- Вывод некоторых полей JSON

SELECT receipt\_number,

payment->>'method' AS method,

(payment->>'amount')::numeric AS amount,

payment->>'bank' AS bank,

payment->>'card\_last4' AS last4

FROM public.receipt\_payments;

-- Чеки с суммой больше 300

SELECT receipt\_number

FROM public.receipt\_payments

WHERE (payment->>'amount')::numeric > 300;

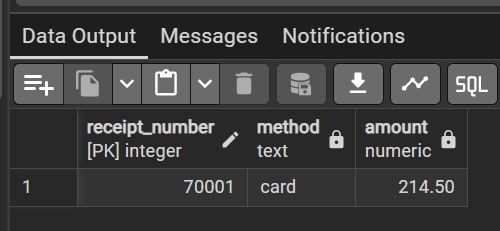
****

Рисунок 6 — Номера чеков, оплаченных картой

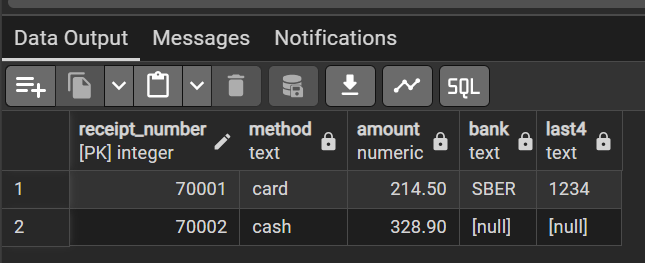


Рисунок 7 — Все строки с JSONB

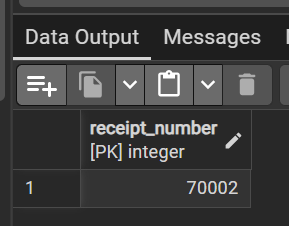


Рисунок 8 — Номера чеков с суммой больше 300

**Проверка ограничений (негативные тесты) (рисунок 9-10):**

-- product\_media: запрещён пустой массив фотографий

INSERT INTO public.product\_media(product\_code, photos, tags)

VALUES (300, ARRAY[]::text[], ARRAY['bread']); -- ОЖИДАЕМ ERROR: pm\_photos\_not\_empty

-- receipt\_payments: нарушение схемы JSON (неверный method и отрицательная сумма)

INSERT INTO public.receipt\_payments(receipt\_number, payment)

VALUES (70001, '{"method":"bitcoin","amount":-5}'::jsonb); -- ОЖИДАЕМ ERROR по rp\_method\_enum/rp\_amount\_non\_neg

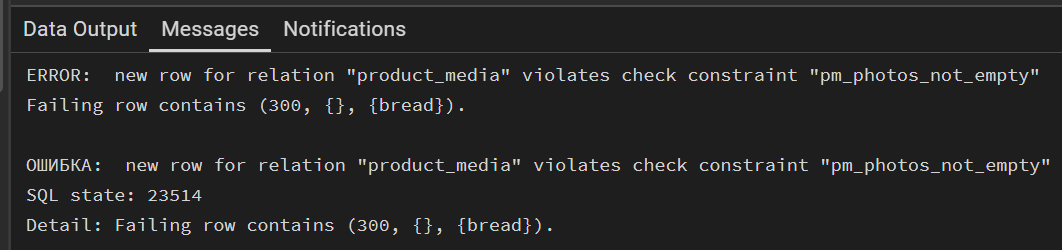
****

Рисунок 9 — Проверка ограничения на размер массива (не может быть 0)

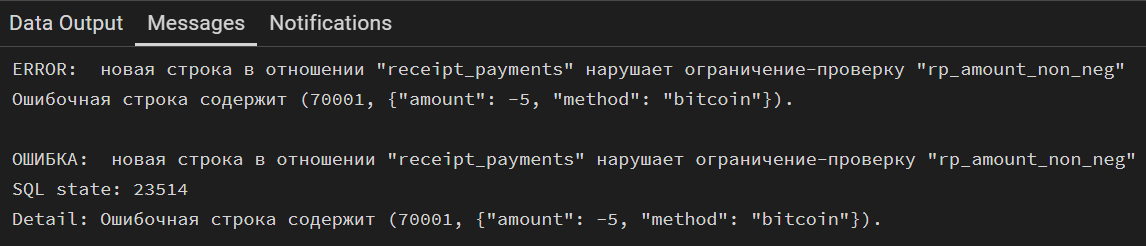


Рисунок 10 — Проверка на отрицательную сумму и неправильный тип оплаты

**Посмотрим обновленную схему и убедимся что появились новые таблицы (рисунок 11):**

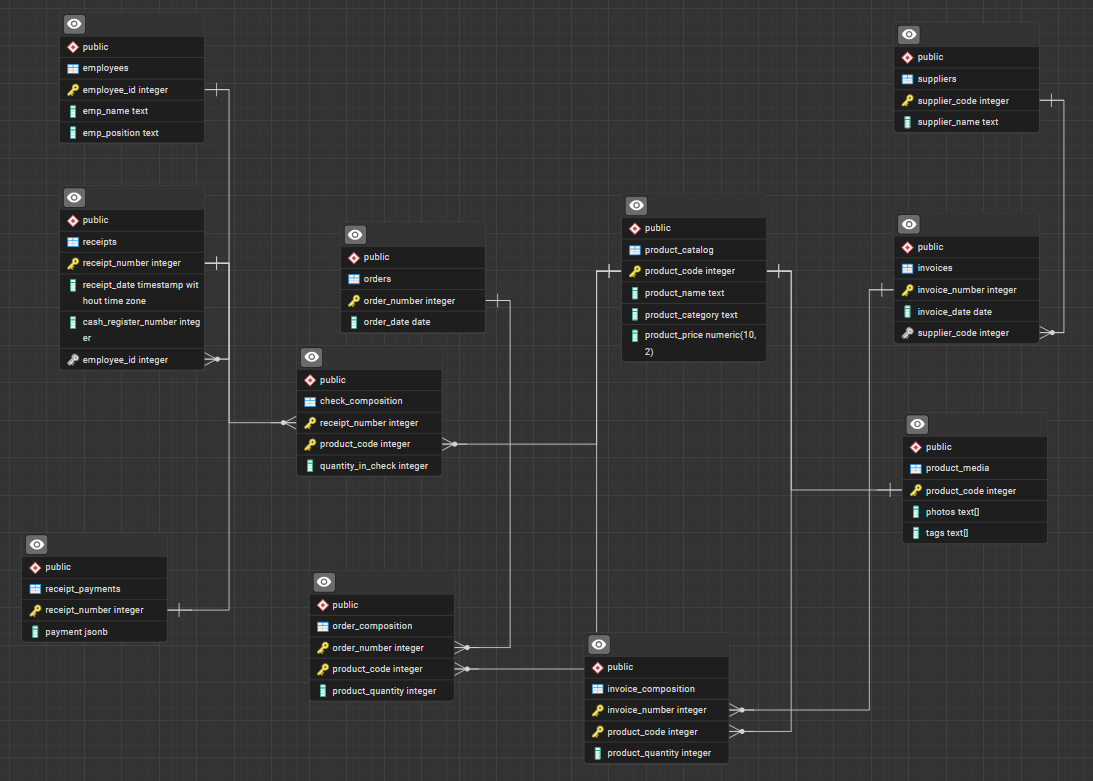


Рисунок 11 — Проверка на наличие новых таблиц в схеме

**Заключение**

БД *rgr\_store* расширена двумя таблицами, использующими типы text[] и jsonb. Таблицы наполнены тестовыми данными; показаны запросы с массивами и JSON; проведены проверки CHECK-ограничений и приведены рекомендуемые индексы. Результаты подтверждены скриншотами выполнения SQL в pgAdmin.