МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Институт математики и информационных систем
Факультет автоматики и вычислительной техники
Кафедра электронных вычислительных машин

Дата сдачи на проверку:

«__»_____ 2025 г.

Проверено:

«__»_____ 2025 г.

Основы DML-запросов в PostgreSQL.

Отчёт по лабораторной работе №2.1

по дисциплине

«Управление данными»

Разработал студент гр. ИВТб-2301-05-00 _____/Черкасов А. А./

(подпись)

Старший Преподователь _____/Клюкин В. Л./

(подпись)

Работа защищена «__»_____ 2025 г.

Киров

2025

Цели лабораторной работы

- освоить основные варианты DML-запросов в PostgreSQL;
- научиться создавать SQL-скрипты для заполнения таблиц данными;
- научиться использованию команд `UPDATE` и `DELETE`;
- научиться работать с представлениями.

Задание

1. Создать и выполнить SQL-скрипт, который будет заполнять таблицы данными (не менее 3–5 строк в каждую таблицу).
2. Создать представления для нескольких таблиц, в которых собираются данные из самой таблицы и других, на которые она ссылается. Хотя бы одно из представлений должно быть сделано с использованием соединений (`JOIN`).
3. Для таблицы, содержащей столбец с числовыми данными, создать представление, отражающее статистику по этому столбцу (минимум, максимум, среднее, сумма).

Тема БД: «Система управления умным домом»

База данных предназначена для хранения информации о пользователях, хабах и устройствах умного дома, а также событий, которые генерируют эти устройства. Она обеспечивает:

- регистрацию и управление пользователями;
- хранение информации о хабах и их местоположении;
- классификацию устройств по типам и отслеживание их состояния;
- фиксацию событий устройств для мониторинга и анализа.

Заполнение таблиц данными

Для заполнения таблиц были использованы команды `INSERT`. Для обеспечения ссылочной целостности использовались возвращаемые значения первичных ключей через `RETURNING`. Ниже приведён фрагмент скрипта `init_data.sql`.

Представления

Были созданы три представления, описанные в файле `views.sql`.

1. Полная информация об устройствах

Представление `device_full_info` объединяет данные из таблиц `devices`, `device_types`, `hubs` и `users`, предоставляя полную информацию об устройстве, его типе, хабе и владельце.

ID	Устройство	Тип	Хаб	Владелец
2	"ThermoSensor"	"Thermometer"	"Living Room Hub"	"lisachenko"
3	"Kitchen Lamp"	"Lamp"	"Kitchen Hub"	"lisachenko"
4	"Security Camera"	"Camera"	"Bedroom Hub"	"bobby"
1	"Ceiling Lamp"	"Lamp"	"Living Room Hub"	"lisachenko"

Таблица 1: Полная информация об устройствах

2. Информация о событиях

Представление `event_info` показывает события вместе с именем устройства и хаба, к которому оно относится.

ID события	Тип события	Данные	Устройство	Хаб
1	"switch_on"	`"{power: on}"`	"Ceiling Lamp"	"Living Room Hub"
2	"temperature_change"	`"{temperature: 23.0}"`	"ThermoSensor"	"Living Room Hub"
4	"motion_detected"	`"{movement: true}"`	"Security Camera"	"Bedroom Hub"

Таблица 2: Информация о событиях

3. Статистика по числовому столбцу

Представление `device_id_stats` отображает минимальное, максимальное, среднее значение и сумму по столбцу `id` в таблице `devices`.

Метрика	Значение	ID записи
"Минимальное значение"	"1"	1
"Максимальное значение"	"4"	4
"Среднее значение"	"2.50"	NULL
"Сумма значений"	"10"	NULL

Таблица 3: Статистика по идентификаторам устройств

Вывод

В ходе выполнения лабораторной работы №2_1 были освоены основные DML-запросы в PostgreSQL. Был создан и выполнен скрипт для заполнения таблиц тестовыми данными с соблюдением ссылочной целостности. Реализованы команды `UPDATE` и `DELETE` для изменения и удаления записей. Также были созданы три представления: два с использованием `JOIN` для получения полной информации об устройствах и событиях, и одно — для отображения статистики по числовому столбцу. Работа позволила закрепить навыки написания SQL-скриптов и подготовила основу для последующих лабораторных работ.

Приложение А1. Исходный код

```
# Общий Containerfile
FROM docker.io/library/postgres:alpine3.22

ENV POSTGRES_USER=pozordom_user
ENV POSTGRES_PASSWORD=pozordom_pass
ENV POSTGRES_DB=pozordom

# Лаб 1 - Создание таблиц
COPY lab1/code/init.sql /docker-entrypoint-initdb.d/
# Лаб 2.1 - Наполнение таблиц и представления
COPY lab2_1/code/init_data.sql /docker-entrypoint-initdb.d/
COPY lab2_1/code/views.sql /docker-entrypoint-initdb.d/
```

Приложение А2. SQL-скрипты

```
-- Лаб 2.1 - наполнение БД

-- Добавляем пользователей
insert into users(username, email, password_hash)
values
('lisachenko', 'me@lsnko.com', 'hash1'),
('bobby', 'bob@example.com', 'hash2'),
('alice', 'alice@example.com', 'hash3');

-- Добавляем хабы
insert into hubs(user_id, name, location, serial_number)
values
(1, 'Living Room Hub', 'Living Room', 'HUB-001'),
(1, 'Kitchen Hub', 'Kitchen', 'HUB-002'),
(2, 'Bedroom Hub', 'Bedroom', 'HUB-003');

-- Типы устройств
insert into device_types(type_name)
values
('Lamp'),
```

```
('Thermometer'),
('Camera');

-- Устройства
insert into devices(hub_id, type_id, name, status)
values
(1, 1, 'Ceiling Lamp', '{"power": "off"}'),
(1, 2, 'ThermoSensor', '{"temperature": 22.5}'),
(2, 1, 'Kitchen Lamp', '{"power": "on"}'),
(3, 3, 'Security Camera', '{"status": "active"}');

-- События
insert into events(device_id, event_type, event_data)
values
(1, 'switch_on', '{"power": "on"}'),
(2, 'temperature_change', '{"temperature": 23.0}'),
(3, 'switch_off', '{"power": "off"}'),
(4, 'motion_detected', '{"movement": true}');

-- UPDATE-примеры

-- пользователь "alice"
update users
set email = 'alice@newmail.com'
where username = 'alice';

-- Включим лампу в гостиной
update devices
set status = '{"power": "on"}'
where name = 'Ceiling Lamp';

-- DELETE-примеры

-- Удалим выключение
delete from events
where event_type = 'switch_off';
```

Приложение А3. Представления

```sql
-- 1. Представление устройств с их хабами и владельцами
create view device_full_info as
select d.id as device_id,
       d.name as device_name,
       dt.type_name,
       h.name as hub_name,
       u.username as owner
from devices d
join device_types dt on d.type_id = dt.id
join hubs h on d.hub_id = h.id
join users u on h.user_id = u.id;

-- 2. Представление событий с инфо об устройствах
create view event_info as
select e.id as event_id,
       e.event_type,
       e.event_data,
       d.name as device_name,
       h.name as hub_name
from events e
join devices d on e.device_id = d.id
join hubs h on d.hub_id = h.id;

-- 3. Статистика по числовому столбцу (id устройств)
create view device_id_stats as
select 'Минимальное значение' as metric,
       min(id)::text as value,
       (select id from devices order by id asc limit 1) as ref_id
from devices
union all
select 'Максимальное значение',
       max(id)::text,
       (select id from devices order by id desc limit 1)
from devices
union all
```

```
select 'Среднее значение',
       round(avg(id),2)::text,
       null
from devices
union all
select 'Сумма значений',
       sum(id)::text,
       null
from devices;
```