# Лабораторная работа №4

# «Postgres partitioning»

**Задание.**

Продемонстрировать работу инструмента секционирования (partitioning) на примере базы данных PostgresSQL.

**Ход работы.**

Секционированием данных называется разбиение одной большой логической таблицы на несколько меньших физических секций.

PostgreSQL предлагает поддержку следующих видов секционирования:

***Секционирование по диапазонам:*** Таблица секционируется по «диапазонам», определённым по ключевому столбцу или набору столбцов, и не пересекающимся друг с другом. Например, можно секционировать данные по диапазонам дат или по диапазонам идентификаторов определённых бизнес-объектов.

***Секционирование по списку:*** Таблица секционируется с помощью списка, явно указывающего, какие значения ключа должны относиться к каждой секции.

***Секционирование по хешу:*** Таблица секционируется по определённым модулям и остаткам, которые указываются для каждой секции. Каждая секция содержит строки, для которых хеш-значение ключа разбиения, делённое на модуль, равняется заданному остатку.

В качестве образца воспользуемся официальными материалами предлагаемыми postgres: <https://postgrespro.ru/docs/postgresql/11/ddl-partitioning>.

Создадим пример таблицы для компании торгующей мороженым:

CREATE TABLE measurement (

city\_id int not null,

logdate date not null,

peaktemp int,

unitsales int

)PARTITION BY RANGE (logdate);

При создании таблица указывается как секционированная. Выбран метод секционирования RANGE а в качестве столбцов образующих ключ разбиения «logdate».

Затем создадим несколько секций:

CREATE TABLE measurement\_y2006m02 PARTITION OF measurement

FOR VALUES FROM ('2006-02-01') TO ('2019-11-01');

CREATE TABLE measurement\_y2006m03 PARTITION OF measurement

FOR VALUES FROM ('2006-03-01') TO ('2019-12-01');

В результате «под капотом» таблица measurement оказывается разделенной на секции, и соответствующие записи помещаются именно в них. Таблица теперь не имеет общей индексации, а каждая секция имеет свой отдельный индекс. При обращении к таблице postgres работает с соответствующими секциями исходя из ключа разбиения. Такой подход может принести следующую пользу:

* В определённых ситуациях кардинально увеличивает быстродействие, особенно когда большой процент часто запрашиваемых строк таблицы относится к одной или лишь нескольким секциям.
* Когда в выборке или изменении данных задействована большая часть одной секции, последовательное сканирование этой секции может выполняться гораздо быстрее, чем случайный доступ по индексу к данным, разбросанным по всей таблице.
* Массовую загрузку и удаление данных можно осуществлять, добавляя и удаляя секции, если это было предусмотрено при проектировании секционированных таблиц. Операция ALTER TABLE DETACH PARTITION или удаление отдельной секции с помощью команды DROP TABLE выполняются гораздо быстрее, чем массовая обработка. Эти команды также полностью исключают накладные расходы, связанные с выполнением VACUUM после DELETE.
* Редко используемые данные можно перенести на более дешёвые и медленные носители.

Всё это обычно полезно только для очень больших таблиц. Какие именно таблицы выиграют от секционирования, зависит от конкретного приложения, хотя, как правило, это следует применять для таблиц, размер которых превышает объём ОЗУ сервера.