АО «Троник»

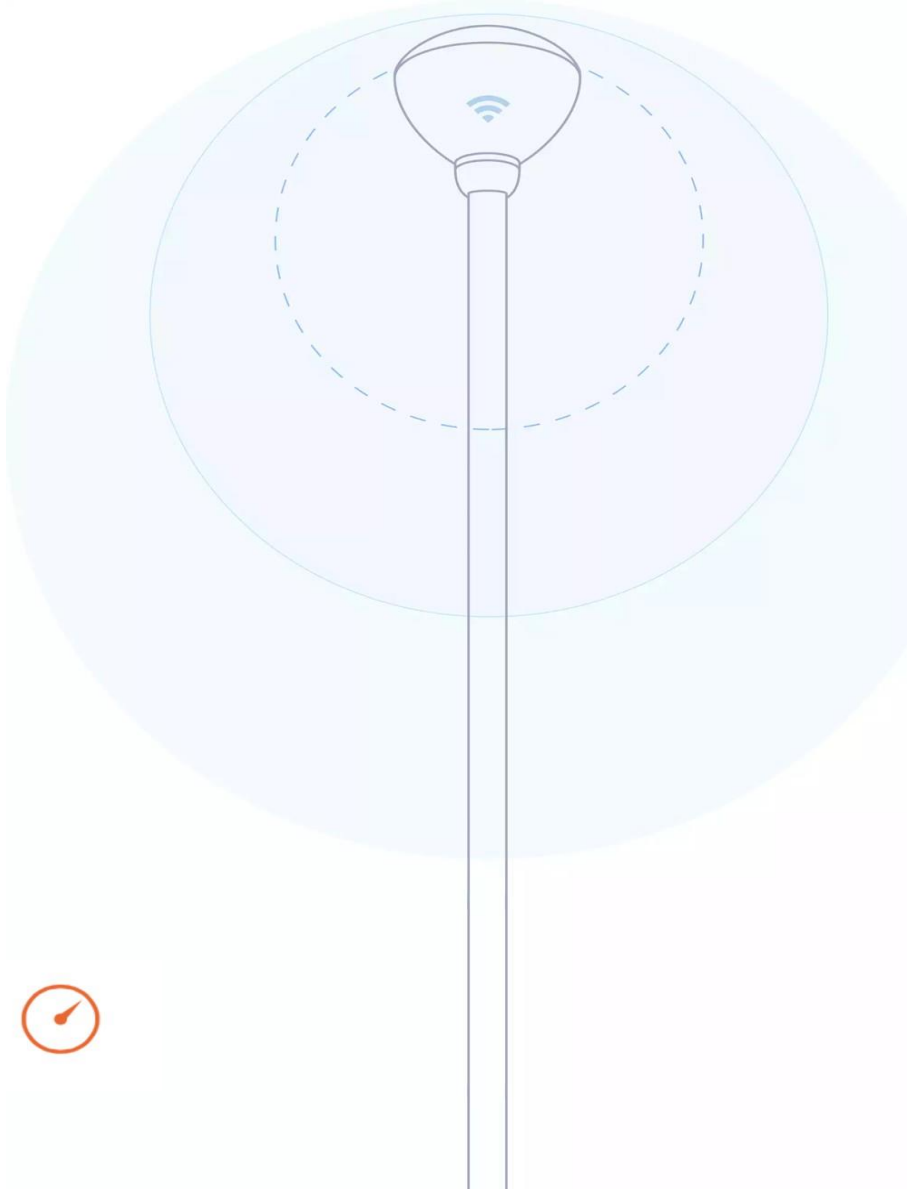
Оптимизация обработки данных аналитических отчётов

Методика обработки данных, поступающих ежесекундно от распределённой сети сенсоров

Исламов Камиль Фаритович

Проект «Умный столб»

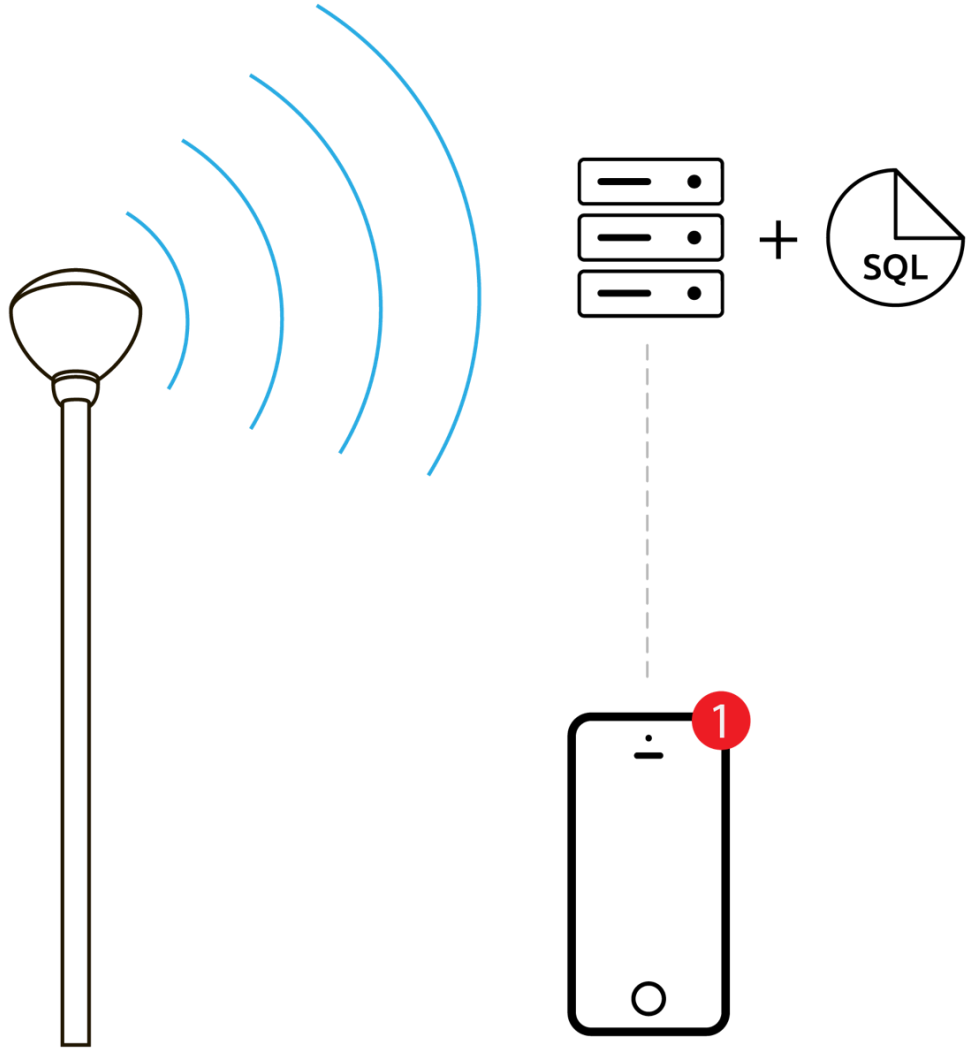
- Встроенная Wi-Fi точка доступа;
- система датчиков и сенсоров;
- управление освещением.



Общая схема

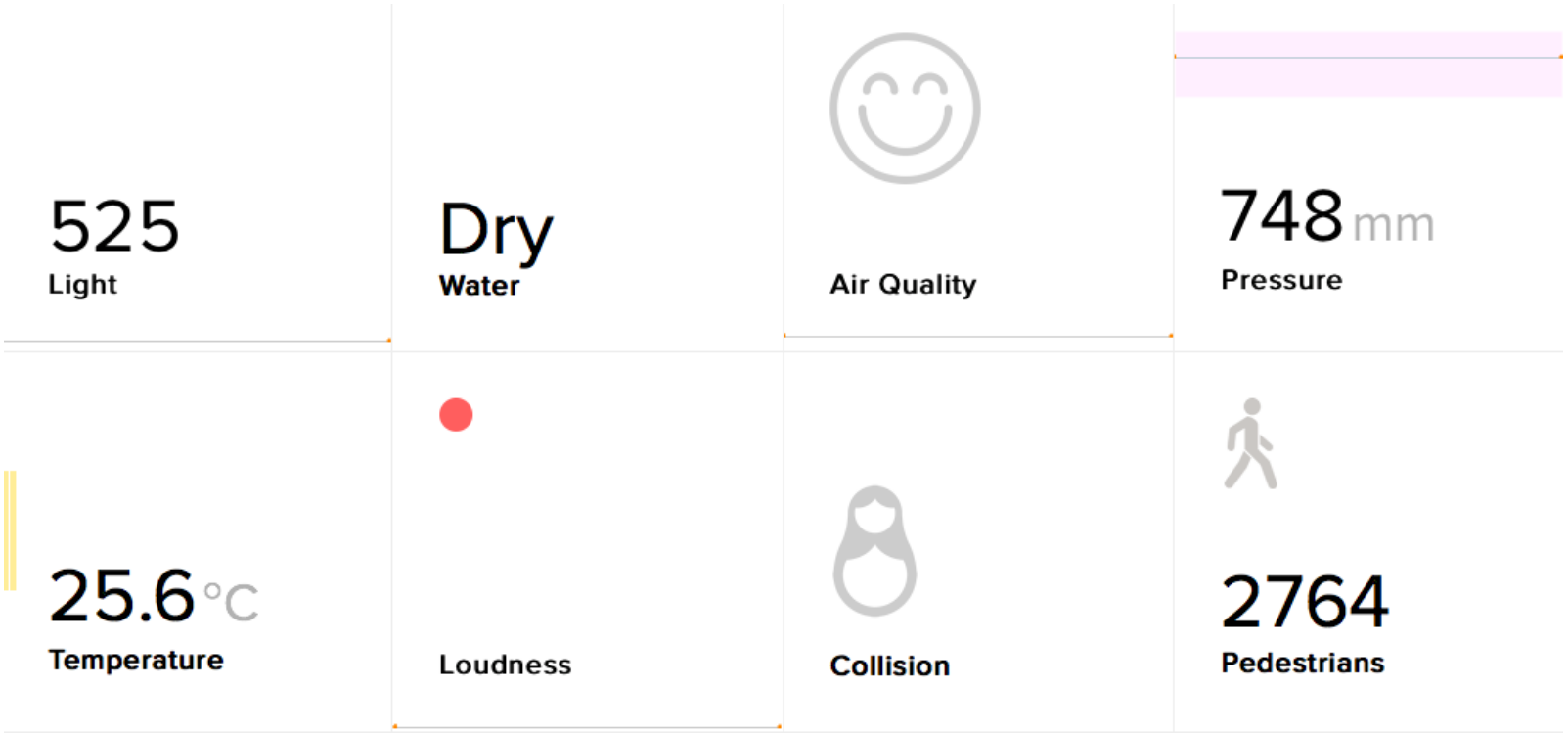
Клиент-серверное решение

- Столб;
- сервер + БД;
- мобильный клиент;
- web-клиент.

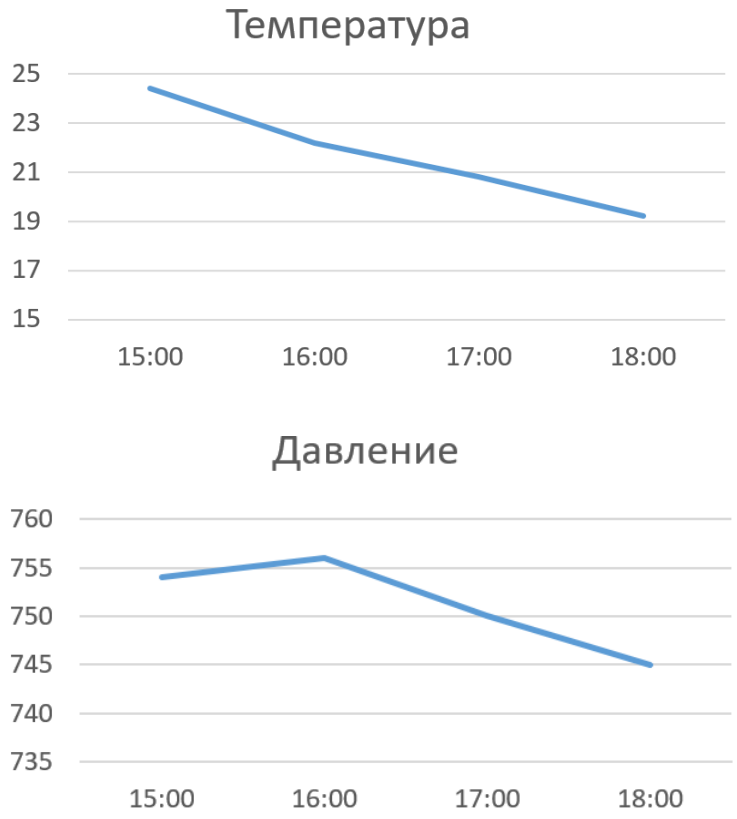


Web-клиент

Актуальная информация о сенсорах



Текущие показатели счётчиков

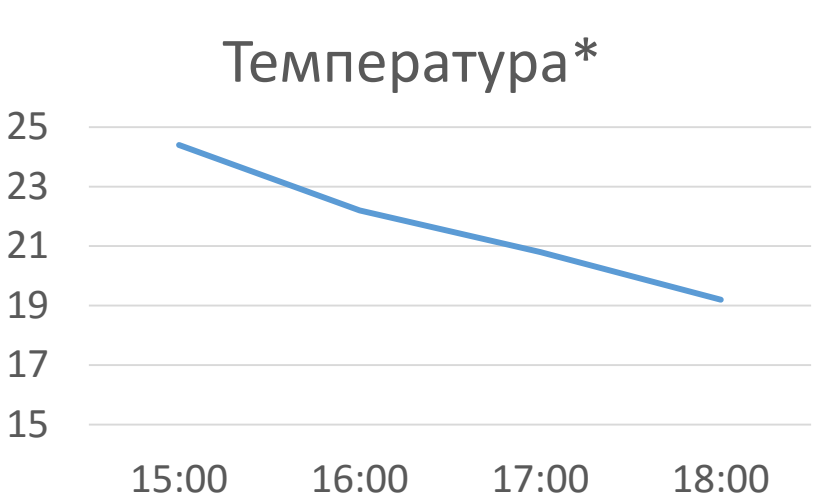


Аналитические отчёты

Поступающие от сенсоров данные

Время	Столб	Сенсор	Номер датчика	Значение
01.05.2015 10:00:07	Пр. Мира, 12, 123	Гигрометр	1	60
01.05.2015 10:00:08	Пр. Мира, 12, 123	Гигрометр	1	59
01.05.2015 10:00:09	Пр. Мира, 12, 123	Гигрометр	1	60
01.05.2015 10:00:10	Пр. Мира, 12, 123	Гигрометр	1	58
01.05.2015 10:00:11	Пр. Мира, 12, 123	Гигрометр	1	57
…	…	…	…	…
10.10.2015 15:12:20	Пр. Мира, 12, 123	ТермоБарометр	1	24.1
10.10.2015 15:12:20	Пр. Мира, 12, 123	ТермоБарометр	2	750
10.10.2015 15:12:23	Пр. Мира, 12, 123	ТермоБарометр	1	24.2
10.10.2015 15:12:23	Пр. Мира, 12, 123	ТермоБарометр	2	751
10.10.2015 15:12:23	Пр. Мира, 12, 123	ТермоБарометр	1	750
10.10.2015 15:12:28	Пр. Мира, 12, 123	ТермоБарометр	2	24.1

Пример отчёта web-интерфейса



* Почасовой график.
Максимальные значения

Время	Столб	Сенсор	Номер датчика	Значение
10.10.2015 15:00	Пр. Мира, 12, 123	ТермоБарометр	1	24.4
10.10.2015 16:00	Пр. Мира, 12, 123	ТермоБарометр	1	22.2
10.10.2015 17:00	Пр. Мира, 12, 123	ТермоБарометр	1	20.8
10.10.2015 18:00	Пр. Мира, 12, 123	ТермоБарометр	1	19.2
10.10.2015 19:00	Пр. Мира, 12, 123	ТермоБарометр	1	18.2
10.10.2015 20:00	Пр. Мира, 12, 123	ТермоБарометр	1	18.1

Решение #1 – SUM, MAX, AVG, MIN

SQL функции агрегации

+ Простота реализации

+ Лёгкость сопровождения

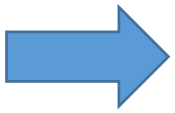
+ Лёгкость доработки

- Низкая производительность web

Решение #2. Предварительная подготовка

Обработка результатов Решения #1:

Выполнить запросы

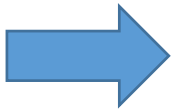


Сохранить Все результаты

+ Высокая производительность web
+ Простота реализации
+ Лёгкость сопровождения
- Усложнение доработки
- Ресурсы на предобработку
- Неактуальные данные

Решение #3. Материализованные представления

Написать запросы

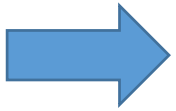


Материализовать результаты

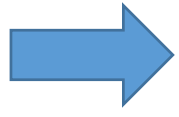
+ Высокая производительность web
+ Простота реализации
+ Лёгкость сопровождения
- Невозможно изменить данные в снимке
- Неактуальные данные

Решение #4. Триггеры

Поступление данных



Обработка записи триггером



Вычисленные результаты

+ Высокая производительность web
+ Наилучшая актуальность данных
+ Лёгкость сопровождения
- Снижение производительности добавления данных
- Сложность запуска и отладки

Решение #5. Порционная обработка

Обработка только поступающих данных:

Получить данные Пересчитать Сохранить результаты

+ Высокая производительность web
+ Высокая скорость обработки новых данных
+ Лёгкость сопровождения
+ Лёгкость доработки
+ Актуальность данных
- Сложность запуска и отладки

Принцип порционной обработки

Первоначальная обработка

Запомнить её время

Для следующих порций данных:

Обработка поступивших данных

Показания приборов учёта без агрегации – «сырые данные»

Время	Сенсор	Номер датчика	Значение
1 мая 10:00:01	555	1	499
1 мая 10:00:01	555	2	350
1 мая 10:00:02	888	1	**9**
1 мая 10:00:03	555	1	490
1 мая 10:00:03	555	2	**380**
1 мая 10:00:03	888	1	9
1 мая 10:00:07	555	1	**500**
1 мая 10:00:07	555	2	400
1 мая 10:00:07	888	1	**10**
1 мая 10:00:08	555	1	**488**
1 мая 10:00:08	555	2	**400**
1 мая 10:00:08	888	1	10
1 мая 10:00:09	555	1	500
1 мая 10:00:09	555	2	400
1 мая 10:00:10	888	1	10

Данные на момент актуальности снимка

Вновь поступившая порция данных

Вычисление
SUM, **MAX**, **MIN**, COUNT, AVG
у вновь поступивших данных

Обработка максимального значения

Время	Сенсор	Значение до
1 мая 10:00	555 / 1	**500**
1 мая 10:00	555 / 2	**400**
1 мая 10:00	888	**10**

Время	Сенсор	Новое значение
1 мая 10:00	555 / 1	**510**
1 мая 10:00	555 / 2	390
1 мая 10:00	888	9

GREATEST

Предыдущие значения

Новая порция

Время	Сенсор	Значение до	Новое	Значение после
1 мая 10:00	555	500	**510**	**510**
1 мая 10:00	555	**400**	390	**400**
1 мая 10:00	888	**10**	9	**10**

Новые значения

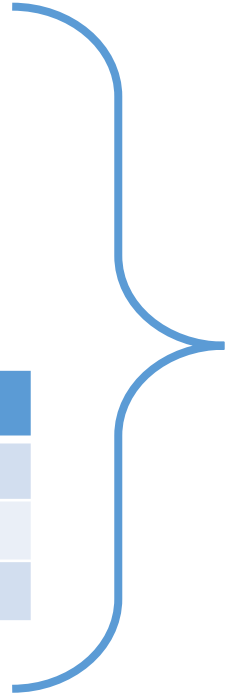
Обработка максимального в БД

```
UPDATE cache_SensorData prev

   SET sensor_value =


GREATEST (prev.sensor_value, curr.sensor_value),

read_count=prev.read_count+curr.read_count


WHERE prev.read_date = curr.read_date AND prev.sensor_id =

curr.sensor_id
```

Обработка минимального значения

Время	Сенсор	Значение до
1 мая 10:00	555 / 1	**500**
1 мая 10:00	555 / 2	**400**
1 мая 10:00	888	**10**

Время	Сенсор	Новое значение
1 мая 10:00	555 / 1	500
1 мая 10:00	555 / 2	**390**
1 мая 10:00	888	**9**



LEAST

Предыдущие значения

Новая порция

Время	Сенсор	Значение до	Новое	Значение после
1 мая 10:00	555	**500**	510	**500**
1 мая 10:00	555	400	**390**	**390**
1 мая 10:00	888	10	**9**	**9**

Новые значения

Обработка минимального в БД

```
UPDATE cache_SensorData prev

    SET sensor_value =


LEAST (prev.sensor_value, curr.sensor_value),

Read_count=prev.read_count+curr.read_count


WHERE prev.read_date = curr.read_date AND prev.sensor_id =

curr.sensor_id
```

Обработка среднеарифметического значения

Время	Сенсор	Значение до **oVal**	Кол-во показаний **oCnt**
1 мая 10:00	555 / 1	**500**	10
1 мая 10:00	555 / 2	**400**	200
1 мая 10:00	888	**10**	100

Время	Сенсор	Новое значение **nVal**	Кол-во показаний **nCnt**
1 мая 10:00	555 / 1	510	1
1 мая 10:00	555 / 2	390	1
1 мая 10:00	888	9	1

Предыдущие значения

$$\frac{oVal \cdot \mathrm{oCnt} + \mathrm{nVal} \cdot \mathrm{nCnt}}{\mathrm{oCnt} + \mathrm{nCnt}}$$

Новая порция

Время	Сенсор	Значение до	Новое	Значение после
1 мая 10:00	555	500	510	**500.91**
1 мая 10:00	555	400	390	**399.95**
1 мая 10:00	888	10	9	**9.99**

Новые значения

Обработка среднеарифметического в БД

```
UPDATE cache_SensorData prev
   SET sensor_value =

(prev.sensor_value * prev.read_count
  + curr.sensor_value * curr.read_count)
                / (st.read_count + a.read_count),
read_count =prev.read_count+curr.read_count

WHERE prev.read_date = curr.read_date AND prev.sensor_id =
curr.sensor_id
```

$$\frac{oVal \cdot \mathrm{oCnt} + \mathrm{nVal} \cdot \mathrm{nCnt}}{\mathrm{oCnt} + \mathrm{nCnt}}$$

Обновление агрегированных данных

Максимальные по часам до поступления

Время	Сенсор	Значение до
1 мая 10:00	555 / 1	500
1 мая 10:00	555 / 2	400
1 мая 10:00	888	10

Максимальные по часам в новой порции

Время	Сенсор	Значение до
1 мая 10:00	555 / 1	450
1 мая 10:00	555 / 2	420
1 мая 10:00	888	10

>	<	=

Максимальные по часам после актуализации

Время	Сенсор	Значение до
1 мая 10:00	555 / 1	500
1 мая 10:00	555 / 2	420
1 мая 10:00	888	10

Изменение

Схема хранения данных

«Сырые» данные

Время	Сенсор	Номер датчика	Значение
1 мая 10:00:01	555	1	499
1 мая 10:00:01	555	2	350
1 мая 10:00:01	888	1	9
1 мая 10:00:03	555	1	490
1 мая 10:00:03	555	2	380
1 мая 10:00:03	888	1	9
1 мая 10:00:07	555	1	**500**
1 мая 10:00:07	555	2	400
1 мая 10:00:07	888	1	**10**
1 мая 10:00:08	555	1	488
1 мая 10:00:08	555	2	**400**
1 мая 10:00:08	888	1	10
1 мая 10:00:09	555	1	500
1 мая 10:00:09	555	2	400
1 мая 10:00:10	888	1	10
…			

Максимальные по часам

Время	Сенсор	Значение
1 мая 10:00	555 / 1	**500**
1 мая 10:00	555 / 2	**400**
1 мая 10:00	888	**10**

Время предыдущего обновления

Результаты внедрения порционной обработки

- Быстрая вставка данных

- Быстрая выдача данных

- Управляемая актуальность

- Минимальная нагрузка на сервер

- Масштабируемость

- Простота развития

Использование SQL представлений

Единая функция, выполняющая обновление кэша

1. Разработать представление, вычисляющее аналитику.

2. Используя системную функцию ***pg_get_viewdef*** , получить SQL запрос, вычисляющий аналитику.

3. На его основе сформировать динамический SQL, обновляющий кэш.

Представление, вычисляющее аналитику

Стандартные функции агрегации

```
CREATE VIEW sensorhour (
    sensor_id,
    sensor_value,
    date,
    read_count)
AS
SELECT sensor_id,
    max(sensor_value) AS sensor_value,
    date_trunc('hour', read_date) AS date,
    count(*) AS read_count
FROM sensorvalue
GROUP BY sensor_id, date_trunc('hour', read_date)
```

Формирование динамического SQL

1. Оператор *Update* к вычисленным значениям.

```
UPDATE cache_sensorhour as cache
```

2. Актуализация вычисленного значения.

```
SET sensor_value = GREATEST (portion.sensor_value, cache.sensor_value)
```

3. Текст SQL от представления + **фильтр** новой порции.

```
FROM (<view sql> WHERE read_date < $1 AND read_date>= $2) as portion
```

4. Выбор строк из кэша, относящихся к новой порции.

```
where  cache.date      = portion.date
and    cache.sensor_id = portion.sensor_id
```

Итоговый динамический SQL

```
1  UPDATE cache_sensorhour cache

2  SET sensor_value =
   GREATEST (cache.sensor_value, portion.sensor_value),
     read_count=portion.read_count + cache.read_count

3  FROM(
    SELECT sensor_id,
       MAX(sensor_value) AS sensor_value,
       date_trunc('hour', read_date) AS date,
       COUNT(*) AS read_count
      FROM sensorvalue
     WHERE read_date < $1 AND read_date >= $2
    GROUP BY sensor_id, date_trunc('hour', read_date)
   ) portion

4  where  cache.date      = portion.date
   AND    cache.sensor_id = portion.sensor_id
```

Результаты внедрения динамического SQL

- Использование стандартных представлений.

- Упрощение добавления новых кэш-таблиц.

- Совместимые функции:
  - SUM, MAX, MIN, COUNT – без усложнений;
  - AVG – с дополнительным вычислением;
  - другие возможные функции – используя принципы AVG.

Спасибо за внимание

Вопросы


ikamil

kam@kambox.ru