**Задание 1:  Разработать структуру хранения данных и функции, предоставляющие определенную выборку из этих данных.**

**Структуры данны**х:

Используя БД **PostgreSQL,** необходимо продумать и реализовать архитектуру хранения данных под следующую задачу:

Каждый час, поступает информация о людях, живущих в 5 крупных городах\*, и 1700 различных уникально названных метрик по каждому человеку. Предусмотреть возможность расширения количества метрик до 2700 шт.. Все метрики представляют собой числа. Для хранения каждой метрики необходимо использовать отдельную колонку. Общее количество граждан данных городов - 5 миллионов. Они распределены не равномерно: 4млн.; 500 тыс.; 300 тыс.; 150 тыс.; 50 тыс..(можно сгенерировать в пропорции 45 людей в одном городе и т.д.) Идентификаторами гражданина является фамилия и имя\*\*. В разных городах могут встречаться полные тёски, в одном городе нет полных тёсок.

*\*Имена городов взять как имена любых реально существующих городов.*

*\*\*Имена и фамилия можно сгенерировать случайным набором символов.*

*\*\*\*Для проверки корректности функций с опцией агрегирования должны соблюдаться следующие правила:*

* *В данных должны быть минимум несколько часов за одни сутки.*
* *В данных должны быть минимум несколько суток за одну неделю.*
* *В данных должны быть несколько недель.*
* *В данных должны быть несколько месяцев.*

**Функции:**

Необходимо реализовать несколько максимально быстрых функций возвращающих курсоры на данные. Функции должны быть написаны с учетом возможного расширения количества метрик в заложенной архитектуре.

* Функция поиска гражданина по фамилии или по фамилии и имени.

Возвращает курсор на город, фамилию, имя человека, а так же указывает две даты: время самой ранней информации о гражданине и время самой последней информации о нём.

* Функция, предоставляющая исходные данные по конкретному гражданину.

Возвращает курсор на запрашиваемые метрики\* по конкретному гражданину за весь период времени или за указанный промежуток времени. Опционально может передаваться параметр агрегирования на большие периоды(*DY – день, WK- неделя, MO - месяц*). В этом случае необходимо представить суммарное значение каждой метрики или формулы\*\* из метрик.

*\*передаются как имена колонок с соответствующим названием с разделителем запятая.*

* Функция, предоставляющая суммарные данные по городам.

Возвращает курсор на запрашиваемые метрики\* в разрезе городов за весь период времени или за указанный промежуток времени. Опционально может передаваться параметр агрегирования на большие периоды(*DY – день, WK- неделя, MO - месяц*). В этом случае необходимо представить суммарное значение каждой метрики или формулы\*\* из метрик.

*\*передаются как имена колонок с соответствующим названием с разделителем запятая.*

Результатом задания должно стать описание архитектуры таблиц с DDL SQL архитектуры. Файл(ы) с кодом функций. И демонстрационные выборки в виде скриншотов с разнообразными, в том числе опциональными, входными параметрами вызова функций.

Общее описание схемы данных и её назначение.

**Основным объектом является customer (покупатель) вокруг него строится схема customer.**

**Необходимо хранить метрики по персонам и предоставлять аналитику по метрикам в различных разрезах, разрезах персон, городов с возможностью фильтрации и агрегации временных периодов.**

Подробное описание каждой таблицы (поля, индексы, ключи).

#### Таблица: customer

* **Назначение**: Хранение информации о клиентах.
* **Структура**:

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| customer\_id | integer | Уникальный идентификатор клиента. |
| name | text | Имя клиента. |
| email | text | Электронная почта клиента. |
| created\_at | timestamp | Дата и время создания записи. |

* **Индексы**:
  + customer\_pkey (Primary Key): customer\_id

#### Таблица: city

* **Назначение**: Справочник городов.
* **Структура**:

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| city\_id | integer | Уникальный идентификатор города. |
| name | text | Название города. |

* **Индексы**:
  + customer\_pkey (Primary Key): city\_id

#### Таблица: metric

* **Назначение**: Справочник метрик.
* **Структура**:

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| metric \_id | integer | Уникальный идентификатор метрики. |
| name | text | Название столбца метрики. |
| explain | text | Объяснение/назначение метрики. |
| data\_type | text | Тип данных метрики. |
| table\_name | text | Название таблицы, в которой хранится метрика |

* **Индексы**:
  + customer\_pkey (Primary Key): metric \_id

#### Ниже описание таблицы метрики ,оно универсально для любого порядкового номера таблицы 1 или 2, такой подход выбран потому что нельзя создать таблицу в PosgreSQL у которой было бы больше 1600 столбцов.

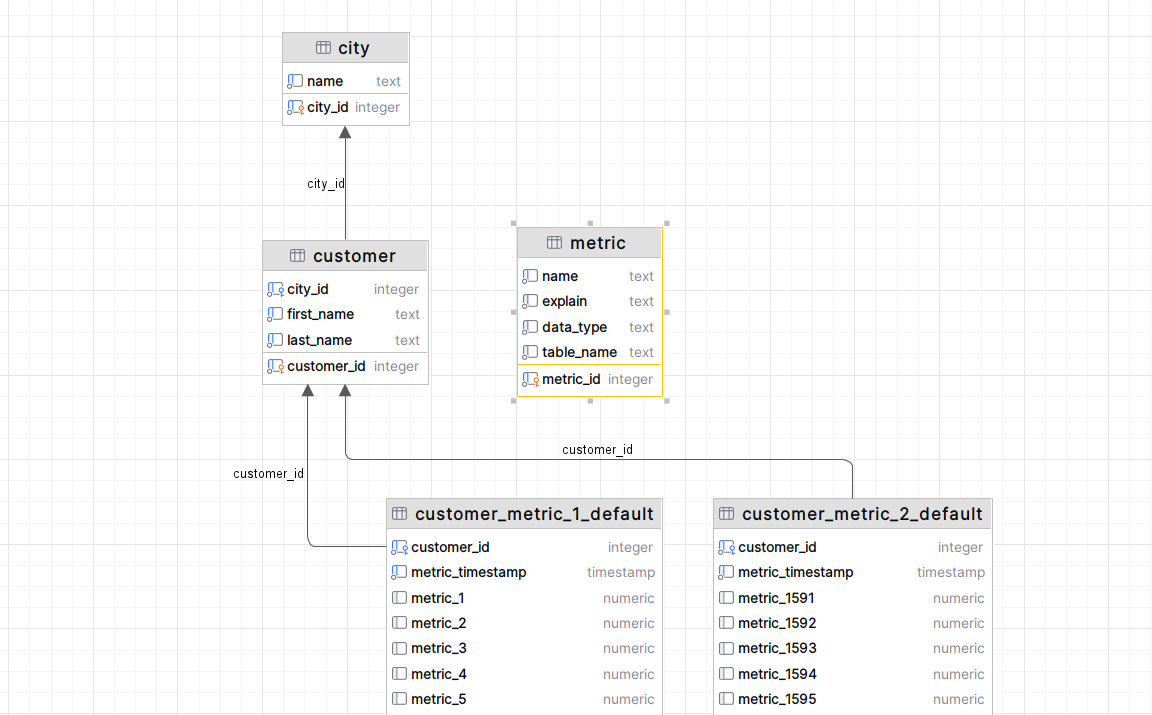
#### Таблица: customer\_metric\_default\_(порядковый номер таблицы)

* **Назначение**: Таблица метрик (порядковый номер таблицы).
* **Структура**:

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| customer\_id | integer | Уникальный идентификатор метрики. |
| metric\_timestamp | TIMESTAMP | Момент времени к которму относятся метрики |
| metric\_(здесь любое число) | numeric | Значение метрики |

* **Индексы**:
  + customer\_metric\_idx\_customer\_id\_metric\_timestamp\_(порядковый номер таблицы) on (customer\_id, metric\_timestamp)
  + customer\_metric\_idx\_\_timestamp\_day\_(порядковый номер таблицы) on (date\_trunc('day'::text, metric\_timestamp))
  + customer\_metric\_idx\_\_timestamp\_ week\_(порядковый номер таблицы) on (date\_trunc(' week'::text, metric\_timestamp))
* **Партиционирование**:
  + Таблица партиционирована по месяцам на основе поля metric\_timestamp (создано партиционирование на 4 года вперед, по правильному нужно создавать партиционирование через триггер).

Общий вид схемы customer



# Весь SQL код представлен в файле ниже

Так же приложен к сообщению под названием «Тестовое задание Postgres (1) (7) (2) (1) (1)\_Грехов\_Святослав.sql»

# DDL описан в файле в регионе «-- region DDL создания схемы customer с заполнением ее данными»:

# Функции описаны в регионе «-- region Функции и их проверка»

# Демонстрационные выборки в виде скриншотов с разнообразными, в том числе опциональными, входными параметрами вызова функций

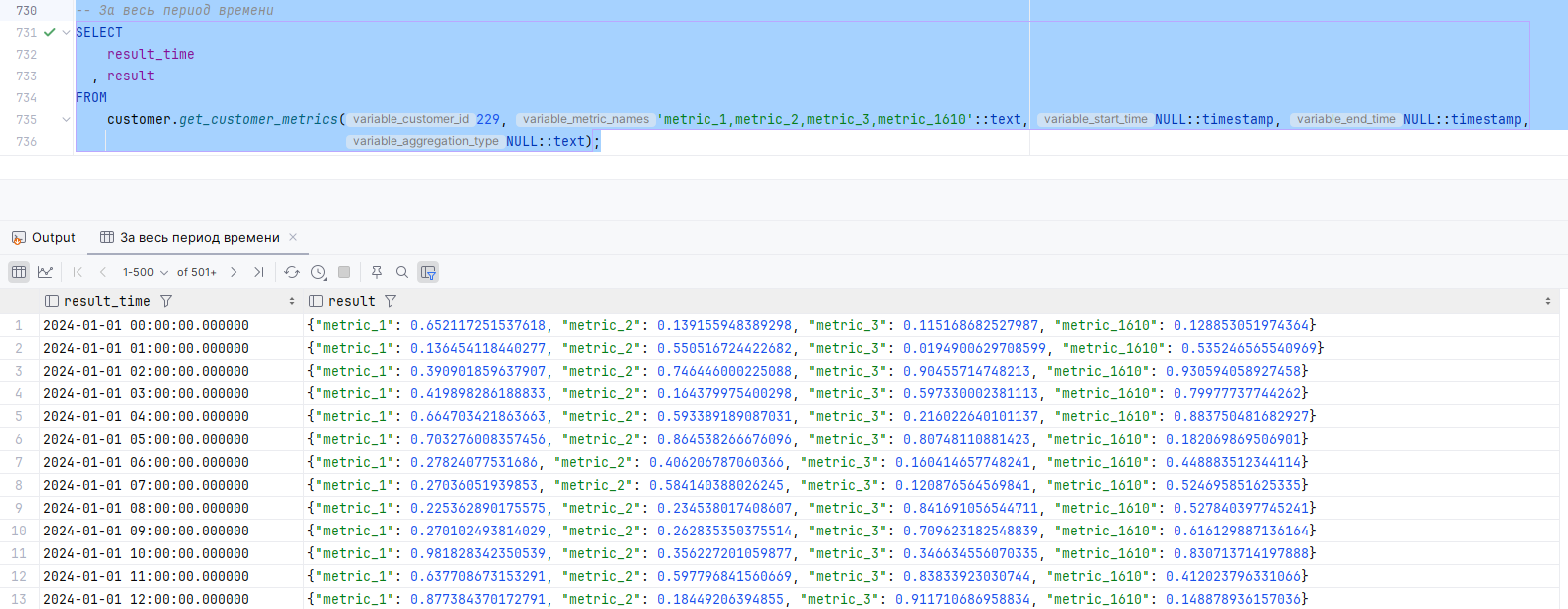
Функция поиска человека по фамилии и имени



Функция поиска человека по фамилии



Функция, предоставляющая исходные данные по конкретному гражданину. За весь период времени



Функция, предоставляющая исходные данные по конкретному гражданину. За весь указанный промежуток времени



Функция, предоставляющая исходные данные по конкретному гражданину. За весь указанный промежуток времени c агрегацией по месяцам.  


Функция, предоставляющая исходные данные по конкретному гражданину. За весь указанный промежуток времени c агрегацией по дням.



Функция, предоставляющая исходные данные по конкретному гражданину. За весь указанный промежуток времени c агрегацией по дням и ограничением времени с одной стороны временного отрезка.



Функция, предоставляющая суммарные данные по городам.

*Здесь добавил возможность суммировать метрики за весь период по городам, флаг ALL для этого нужно использовать*

Функция, предоставляющая суммарные данные по городам. В разрезе городов за весь период времени



Функция, предоставляющая суммарные данные по городам. В разрезе городов за указанный промежуток времени



Функция, предоставляющая суммарные данные по городам. В разрезе городов с агрегацией по месяцам

*Представлена только Москва потому, что сгенерил только 1 млн записей с метриками, если больше, то появятся и другие города.*



Функция, предоставляющая суммарные данные по городам. В разрезе городов суммирует показатели за промежуток заданный на входе

*Представлена только Москва потому, что сгенерил только 1 млн записей с метриками, если больше, то появятся и другие города.*

*Здесь добавил возможность суммировать метрики за весь период по городам, флаг ALL для этого нужно использовать*

