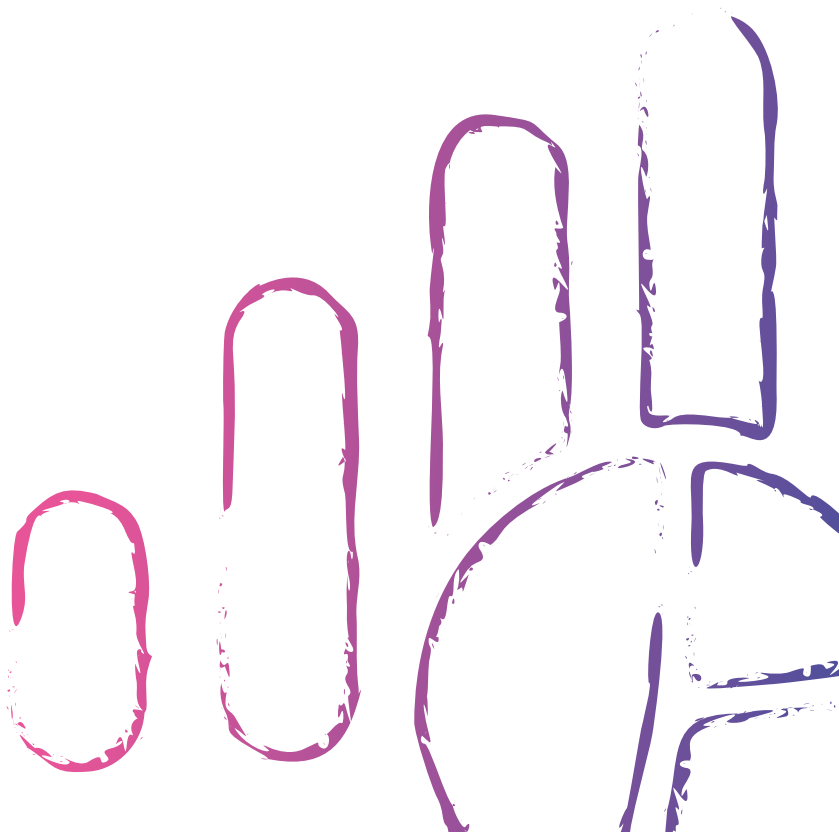
Luxms BI

Руководство по конфигурации дэшлетов

JSON КОНФИГУРАЦИИ



2023-02-20

Оглавление

1 Расширенные настройки виджетов Luxms BI

Дэшлет (**дэш**) - зарезервированное место на дэшборде какого-либо из датасетов, являющееся контейнером для конкретного визеля, но не равный ему

Визель - компонент React, по умолчанию рисующий график на основе входных данных и характеристик и особенностей самого типа графика. В одном дэше может быть отрисовано более одного визеля.

Настройки дэша включают в себя тип дэша и JSON структуру, описывающую содержимое дэша.

Тип дэша (view_class) - это строка, указывающая тип визеля, который нужно отобразить в данном дэшлете.

На данный момент существует следующий список типов дэшей, доступных “из коробки”:

1. Текстовая метка (text)

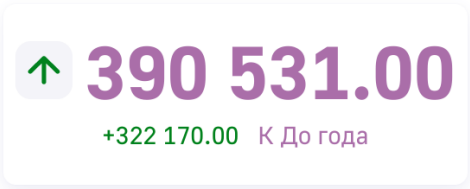


Рис. 1.1 **Значение**

Показывает одно значение на текущий момент в виде цифры. По умолчанию выводится с символом роста/падения и абсолютным отклонением от прошлого периода.

2. Список (list)

Показывает несколько значений на текущий момент в виде списка.

3. Спидометр (circle)

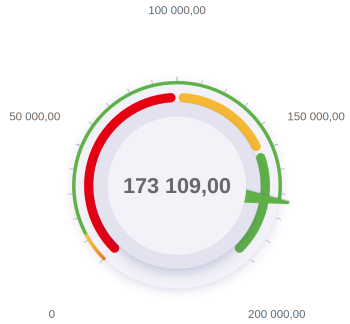


Рис. 1.2 **Спидометр**

Представляет данные на текущий момент в соответствии с нормативными значениями.

4. Полукруглый спидометр (semicircle)

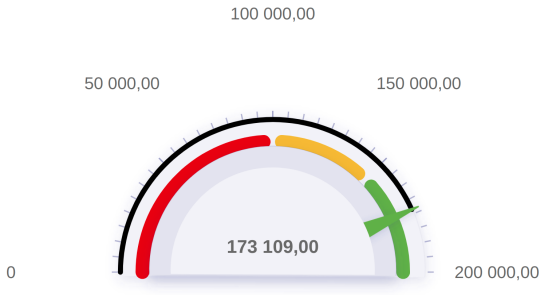


Рис. 1.3 **Полукруглый спидометр**

Представляет данные на текущий момент в соответствии с нормативными значениями.

5. Термометр (thermometer)

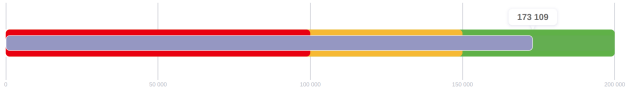


Рис. 1.4 **Термометр**

Представляет данные на текущий момент в соответствии с нормативными значениями.

6. Пирог (pie)

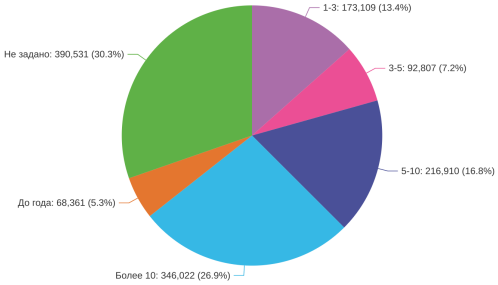


Рис. 1.5 **Пирог**

Показывает долевое отношение нескольких значений на текущий момент.

7. Радиальная диаграмма (radar)

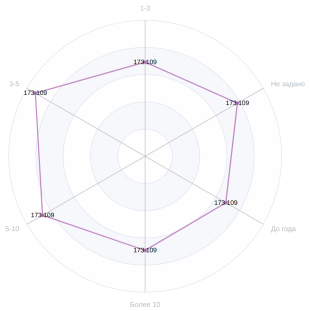


Рис. 1.6 **Радар**

Представляет данные на текущий момент в виде радиальной диаграммы.

8. Воронка (funnel)

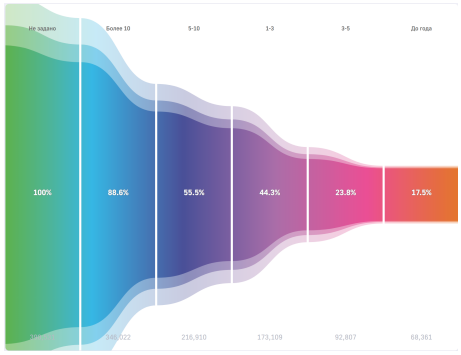


Рис. 1.7 **Воронка**

Представляет данные на текущий момент в виде воронки.

10. Весы (scales)

Рис. 1.8 **Весы**

Показывает отношение двух значений на текущий момент.

11. Столбики (column)

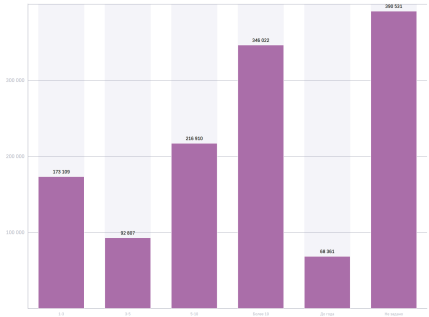


Рис. 1.9 **Столбики**

Представляет данные в виде столбчатого графика.

12. Линии (line)

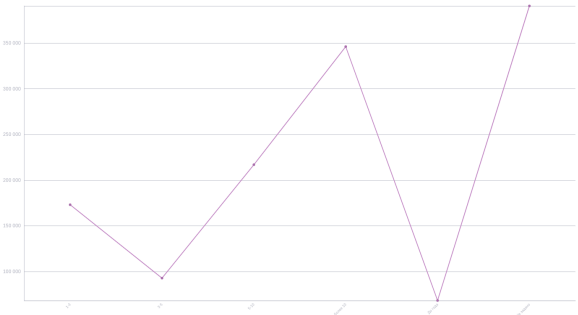


Рис. 1.10 **Линия**

Представляет данные в виде линейного графика.

13. Области (area)

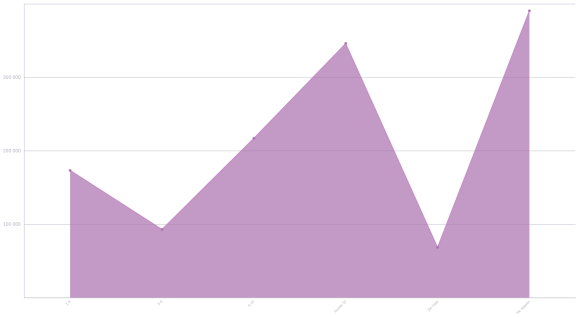


Рис. 1.11 **Области**

14. Штабели (stacked-column)

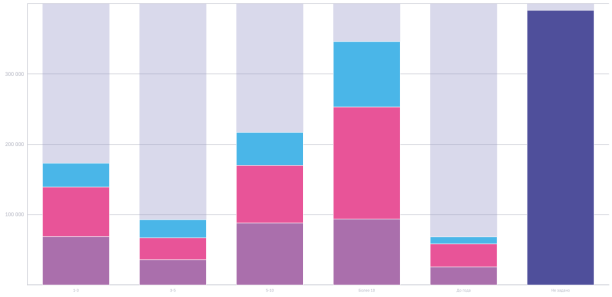


Рис. 1.12 **Штабели вертикальные**

Представляет данные в виде столбчатого графика с накоплением.

15. Горизонтальные столбики (bar)

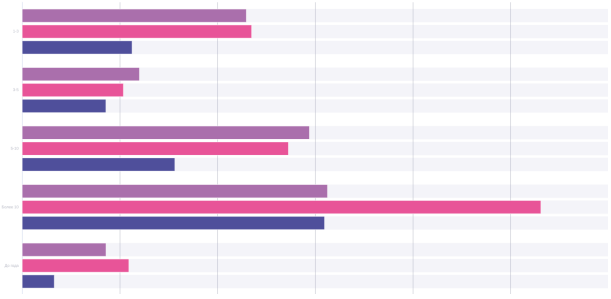


Рис. 1.13 **Столбики горизонтальные**

Представляет данные в виде горизонтального столбчатого графика.

16. Горизонтальные штабели (stacked-bar)

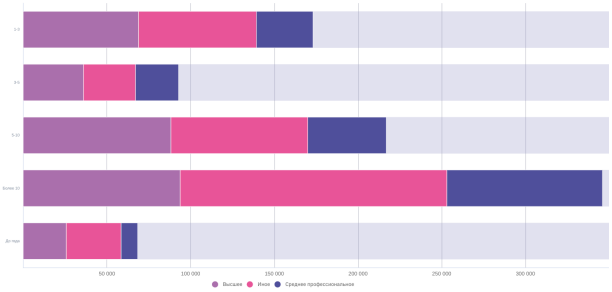


Рис. 1.14 **Штабели горизонтальные**

Представляет данные в виде горизонтального столбчатого графика с накоплением.

17. Таблица (table)

	Высшее	Иное	Среднее професси...
1-3	68863	70485	33761
3-5	36011	31080	25716
5-10	88238	81785	46887
Более 10	93784	159344	92894
До года	25747	32764	9850

Рис. 1.15 **Таблица**

Представляет данные в табличном виде.

18. Плавный линейчатый график (spline)

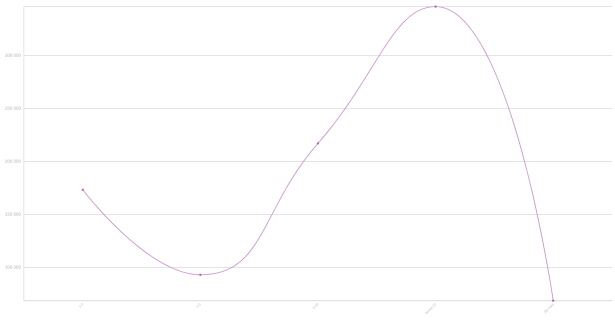


Рис. 1.16 **Сплайн**

Представляет данные виде плавного линейного графика.

19. Комбинированный график

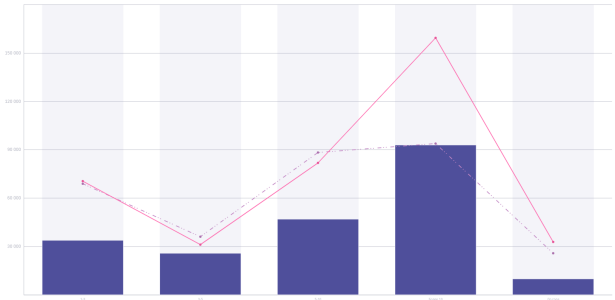


Рис. 1.17 **Комбинирование**

Объединяет два типа визуализации на одном графике, например, столбики и линии (для каждой величины задается свой тип графика через настройку widgetType в блоке style секции dataSource).

20. Картинка

Статичная картинка.

21. Водопад (waterfall)

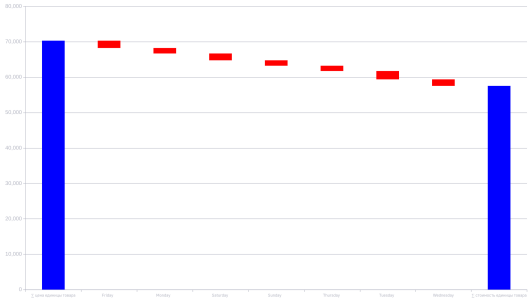


Рис. 1.18 **Водопад**

Визуальное представление для факторного анализа.

22. Схема (plan)

Представляет данные в виде кастомизированной графики с возможностью детализации.

23. Карта (map)

Позволяет визуально сравнить географические области по значению показателя или по выполнению норматива (требуется добавить в конфиг деша типа map опции “fillAreasByData” - окрашивание, “fillLegend” - легенда для него).

24. board – позволяет задать для отображения внутри деша более одного визеля через список оных в поле children.
25. tableP – пивот-таблица.

experience	Высшее	Иное	Среднее профессионально
1-3	68863	70485	33761
3-5	36011	31080	25716
5-10	88238	81785	46887
Более 10	93784	159344	92894
До года	25747	32764	9850
Общий итог	312643	375458	209108

Рис. 1.19 **tableP**

26. lookup-table – таблица, отображающая результат запрос в кастомную таблицу БД.
27. scatter – точечный график (множество точек, не соединенных между собой).

Рис. 1.20 **Точки**

28. html - Дэш, используемый для отображения статичного текста с использованием html-верстки. В поле `htmlText` указывается html-верстка. Верстка прописывается в кавычках. В примере ниже продемострировано использование `mailto`, стилей, и тега `p` в дэше html:

```
htmlText: "<p class='text'>Данные носят предварительный характер
<br>(имеют накопительный эффект)<br>и будут обновляться</p><p class=
  'text'>Подробности по
<a href='mailto:admin@luxms.ru'>электронной почте</a></p>
   <style>.text{text-align: center; color: red; font-size: 1.1rem;}</style>",
view_class: 'html',
```

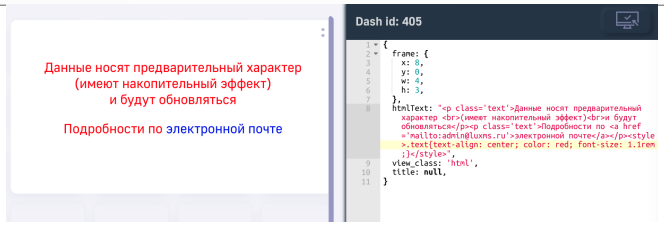


Рис. 1.21 **Отображение html-дэша**

29. tabs - указание нескольких визелей внутри одного дэша путем переключения. Пример:

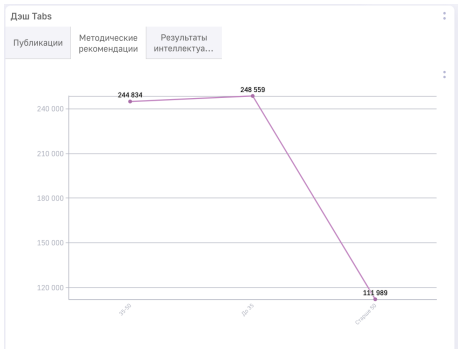


Рис. 1.22 **Дэш tabs**

30. treemap - древовидная карта

Рис. 1.23 **Древовидная карта**

31. sankey - Диаграмма Санкей

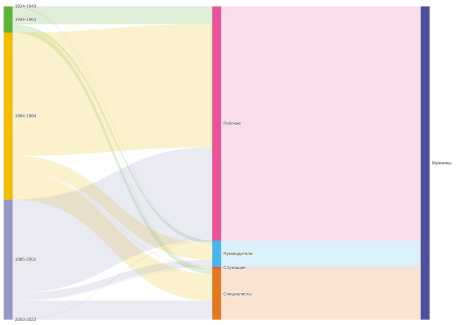


Рис. 1.24 **Диаграмма Санкей**

30. classified-bar – горизонтальные столбики для одномерного случая.
31. classified-column – вертикальные столбики для одномерного случая.

JSON структура описания деша на верхнем уровне состоит из следующих секций и опций:

- frame
- onClick
- dataSource
- display
- url
- bgImage
- options
- hierarchy
- saveAbilities
- onClickDataPoint
- children
- view_class
- title

1.1 Секция frame

Данная секция используется для размещения создаваемых дэшей на дэшборде

Формат:

```
"frame": {
  "h": Integer,
  "w": Integer,
  "x": Integer,
  "y": Integer,
}
```

Где

- `h` - указание высоты в относительных единицах;
- `w` - указание ширины блока в относительных единицах;
- `x` - указание расположения дэша по оси X в относительных единицах;
- `y` - указание расположения дэша по оси Y в относительных единицах;

Дэш занимает все свободное пространство, в случае указания

```
"frame": {
  "h": 0,
  "w": 0,
  "x": 0,
  "y": 0,
}
```

По мере увеличения количества дэшей на дэшборде, расположение будет меняться, маневрируя показателями в секции `frame`

1.2 Секция dataSource

Описывает источник данных для деша, измерения (dimension) и меры (measures)

Формат:

```
{
 "dataSource": {
   "koob": "String",
   "dimensions": [],
   "measures": [],
   "xAxis": "String",
   "yAxis": "String",
   "filters": {} или [],
  "sortBy": "String",
   "style": {
     "String": {
       "String": {
       "title": "String",
       "color": "String"
       }
```

```
    },
    "measures": {
      "String": {
        "title": "String",
        "color": "String",
        "unit_id": Integer,
        "format": "String",
        "widgetType": "String"
      }
    }
  }
 }
}
```

1.2.1 Опции секции dataSource

“koob”:
- тип STRING, имя куба. Указание куба происходит с указанием источника данных и куба. Например:

```
koob: "название_источника.название_куба"
```

“dimensions” : [ ]

- Принимает массив STRING через запятую, имена столбцов у куба (должны быть известны заранее)
- Обязателен, для загрузки данных по этим столбцам
- Через двоеточие можно задать псевдоним. Если у id столбца в конце поставить двоеточие и условное слово, к примеру, newId. Дальнейшее обращение можно будет сделать через этот идентификатор.
- Принимает формулы или функции, где обязательно нужно будет задавать новый id через двоеточие в конце
- Пример: `"dimension":["dt", "sex","age", "concat(age, sex):id2"]`
- У размерности типа “период” есть возможность указания формата отображения. Для этого вы можете использовать функции форматирования даты своего источника. в Postgres для этого используется `TO_CHAR`. Пример: `"dimension":["TO_CHAR(dt, MM.DD):new_date", "sex","age", "concat(age, sex):id2"]`

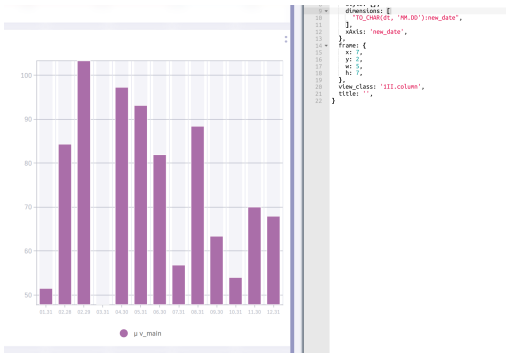


Рис. 1.25 **Форматирование даты**

“measures” : [ ]

- Тип массив STRING через запятую, формула которая применяется к столбцам куба.
- Обязателен, для получения значений
- Если у id столбца в конце поставить двоеточие и условное слово, к примеру, newId. Дальнейшее обращение можно будет сделать через этот идентификатор.
- Пример: `"measures": ["sum(v_main):id1", sum(v_main):newId","count(v_main)"]`.

Также в подсекцию **“measures”** можно включить условие ее формирования, например, `"if (sum(pers)=0, 0, sum(fin)/sum(pers)):d"`. В результате на сервере формируется SELECT. Для данного примера - `CASE WHEN sum(pers) = 0 THEN 0 ELSE sum(fin) / sum(pers) END as "d"`.

- Для сводной таблицы (pivot/table) реализована возможность отображения по умолчанию не всех фактов. Для этого в конфигурации необходимо прописать массив `innerMeasures`. Те факты, которые будут указаны в данном массиве будут включены по-умолчанию.
- Пример: `"innerMeasures": ["id1", "newId"]`.

“xAxis”, “yAxis” : ““

- Тип String, набор id dimensions
- id записываются в строку через точку запятой
- Есть спец id **’measures’,** при котором на оси будут находиться все id measures

Пример: `"xAxis": "sex;age", "yAxis": "measures;dt"`.

“filters” : {} или []

- Тип OBJECT или массив
- Тип OBJECT содержит ключи id измерений (dimension)
- Тип массив содержит массив строк - измерения (dimension), по которым работает фильтрация из управляющего деша
- Для варианта с типом OBJECT значения id ключа могут быть:

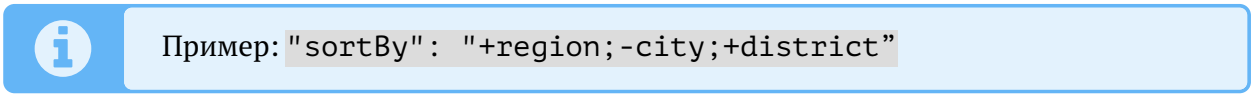
  1. **true** - данные по этому фильтру будет подтягиваться из управляющего деша 2. массив STRING, установка определенного фильтра, значения из управляющего деша будут проигнорированы

Пример: `"filters": {"sex": true, dt:["=","2020-02-31"]}`

- Для варианта с типом Массив > Пример: `"filters": ["region": “city”, “district”]` {.is-info}

“sortBy”

- тип STRING, в строке поочередно указываются id размерностей с указанием знака сортировки



“style” : {}

- тип OBJECT, принимает id размерностей и спец. ключ **“measures”**
- **ключ id “dimension” :**

  1. тип OBJECT
  2. принимает id ключ из массива **dataSource.dimensions**

  - тип OBJECT - ключ **“title”**: тип STRING, меняет заголовок dimension - ключ **“color”**: тип STRING, меняет цвет dimension

  3. Пример : `style:{"sex":{"Мужчина":{ "title": "myyyy" }}}`

- **спец ключ “measures”:**

  1. тип OBJECT
  2. Принимает id ключ из массива **dataSource.measures**

  - тип OBJECT - ключ **“title”**: тип STRING, меняет заголовок measures - ключ **“color”**: тип STRING, меняет цвет measures. Используется значение #HEX для указания цвета - ключ **“unit_id”**: тип NUMBER, добавляет единицу измерения из таблицы unit (каждый unit имеет свой id). **unit_id** необходимо указывать в схеме выбранного датасета в таблице units - ключ **“format”**: ниже подробное описание **- ключ** ”widgetType”**: тип STRING, меняет тип графика для выбранного факта / размерности. Имеет следующие типы значений:

  a. `bar` - Отображение данного показателя в виде столбиков b. `line` - Отображение данного показателя в виде линии c. `spline` - Отображение данного показателя в виде плавной линии (сплайн) - ключ **“strokeStyle”** в случае, если для дэша выбран тип отображения “Линии”, “Сплайн” или “Области”, то дэш меняет свое отображение в зависимости от выбранного значения. Ниже представлен список доступных значений:

```

a. `Solid` - сплошная линия
 b. `ShortDot` - линия отображена точками, расположенными близко
   относительно друг друга
 c. `ShortDash` - линия отображена черточками небольшой длины
 d. `ShortDashDot` - линия отображена черточками небольшой длины и точками

 e. `ShortDashDotDot` - линия отображена черточками небольшой длины и двумя
   точками
 f. `Dot` - отображение точками
 g. `Dash` - линия отображена черточками
 h. `LongDash` - линия отображена длинными черточками(тире)
```

```
i. `DashDot` - линия отображена черточками и точками
j. `LongDashDot` - линия отображена длинными черточками(тире) и точкой
k. `LongDashDotDot` - линия отображена длинными черточками(тире) и двумя точками
```

Пример:

```
"style": {
  "measures": {
    "newId":{
       "title": "мой новый заголовок",
       "color": "#f2f2f2",
       "unit_id": 2,
       "format": "# ###",
       "widgetType": "spline"
    }
  }
}
```

Для указания стилей всем фактам разом, в конфигурации вместо названия факта указывается символ `*`. Пример:

```
"style": {
  "measures": {
    "*":{
       "title": "мой новый заголовок",
       "color": "#f2f2f2",
       "format": "# ###",
    }
  }
}
```

Аналогичным способом можно указать стили для всех показателей размерности:

```
"style": {
  "category": {
    "*":{
       "color": "#f2f2f2",
       "format": "# ###",
    }
  }
}
```

В случае необходимости сортировки данных по размерности, которую выводить не нужно можно размерности прописать **title** с пустой строкой.

Пример:

```
"style": {
  "month_id": {
    "*":{
       "title": ''
    }
```

```
  }
}
```

Заголовок для подытогов задается внутри объекта, имеющего в названии символ суммы Σ (U+2211) и наименование размерности:

Пример:

```
"style": {
  "category": {
    "Σcategory":{
       "title": 'Подытог'
    }
  }
}
```

“limit”

- тип NUMBER, ограничение по количеству выводимых данных
- Обрезает ось-X на заданное зн-ие

Пример: `"limit": 3`

Только для koob-table-simple:

1. Принимает id ключ из массива **dataSource.measures**
2. Создаем любые переменные в виде ключ : выражение
3. backgroundColor: будет либо цвета:
   a. если myConst1 = true, backgroundColor: ‘#00b9ac’
   b. если myConst2 = true, backgroundColor: ‘#fed450’
   c. если (myConst1 && myConst2) = false, ‘#93CAFE’
   d. Пример: `"newId": {"myConst1":" done_date != null && delivery_target_date > due_date "}`

Пример:

```
"style": {
  "measures": {
    "newId":{
      "myConst1": используется lpe-выражение
      (true || false),
      "myConst2": используется lpe-выражение
      (true||false),
      "backgroundColor":"lpe:when(lpe(myConst1), '#00b9ac',
      lpe(myConst1),'#fed450', '#93CAFE')"
    }
  }
}
```

“subtotals” : “dimensions” Только для таблицы tableP и сводной таблицы (pivot/table)

- тип STRING, принимает список размерностей, для которых необходимо выводить подытог

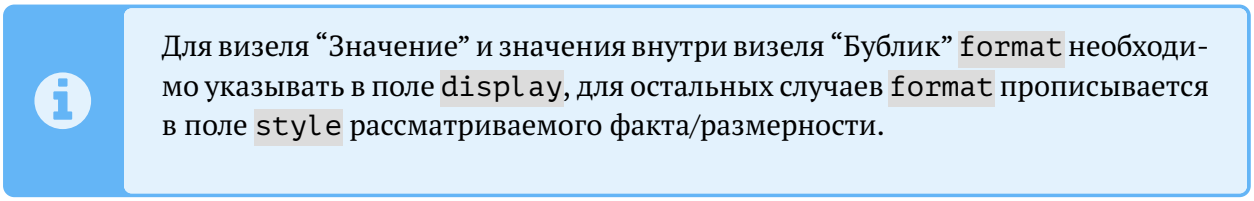
- работает для источника данных `PostgreSQL`

Пример : `subtotals: "sex;age;degree"`

“format” :

- тип STRING
- Используется в типах деша: **text**
- формат задается исходя из документации (по данной ссылке)
- Примеры:

1. `"format": "-# ###,0%"` - значение будет выводится с один знаком после запятой с символом “%”
2. `"format":"#[ / 2]"` - отображаемое значение в тултипе и подписях будет поделено на 2
3. `"format": "# ### [тыс, млн, млрд, тера]")` - значение будет отображаться в зависимости от количества цифр в числе. Пример: число “220 000” будет выводится как “220 тыс”, число “200 000 000” будет выводится как “220 млн” и т.д.
4. `"format": "△ #"` округляет значение в большую сторону, а `"▽ #"` в меньшую



1.3 Секция display

Описывает отображение деша

Формат:

```
{
 "display": {
   "limit": Integer,
   "range": Integer,
   "format": "String",
   "maxFontSize": Integer,
   "minFontSize" : Integer,
   "bgColor": "String",
   "color": "String",
   "customValue": "String",
   "stoplight": {
     "lights": [
       {
```

```
      "name": "String",
      "color": "String",
      "limit": [
        Integer, ...],

      "bgColor": "String"
    }]

  }
 }
}
```

1.3.1 Опции секции display

“limit” :

- тип NUMBER
- Обрезает ось-X на заданное зн-ие
- Пример `"limit":20`

“range” :

- Принимает массив значений, типа NUMBER
- Применяется на типе визеля с лимитами
- Пример: `"range": [0,30000]`
- range можно также указать для конкретной единицы измерения. Для этого необходимо перейти в раздел единиц измерения, редактировать или создать новую единицу измерения, и прописать в ее конфигурации `range`

“stoplight”:

- тип OBJECT

“lights”:

- Принимает массив OBJECT, ключи которого:
  - **name** : имя лимита (STRING)
  - **color :** цвет (STRING)
  - **limit :** промежуток цвета (массив NUMBER)
  - **bgColor :** цвет фона у показателя (STRING)
- Пример:

```
"lights": [ {
  "name": "red",
  "color": "#ff4d4d",
  "limit": [20000,3000],
  "bgColor" :#d4f6dc
```

```
    },
{...},{...} ],
```

“maxFontSize” :

- тип NUMBER || STRING
- Используется в типах деша: **text**
- Максимальный размер текста
- Пример: `"maxFontSize": "30"`

“minFontSize” :

- тип NUMBER || STRING
- Используется в типах деша: **text**
- Минимальный размер текста
- Пример: `"minFontSize": "30"`

“cmpTitle” :

- тип STRING
- Используется в типах деша: **gauge**
- Указание текстовой подписи под значением
- Пример: `"cmpTitle": "Нижний заголовок"`

“gap” : - тип NUMBER || STRING - Используется в типах деша: **board** - Указание отступов между дочерними элементами - Пример: `gap: "5"`

“selectedColor” : - тип STRING - Используется в типах деша: **map** - Указание цвета обводки при включенной множественной фильтрации на карте - Пример: `selectedColor: "#000"`

1.3.2 Визуализация тултипов

Визуализация тултипов возможна с использованием опции display. В данном разделе будет рассмотрено два варианта конфигурации отображения тултипов: отображение в тултипе визеля и отображение в тултипе математических формул. Для данных действий используется опция `tooltip`. Для отображения визеля внутри тултипа необходимо `tooltip` указать как объект и прописать внутри него конфигурацию. Пример:

```
"display": {
 "tooltip": {
  "dataSource": {
    "xAxis": "measures",
    "yAxis": "sex",
    "dimensions": ["sex"],
   },
  "view_class": "pie",
 },
},
```

Отображение данного тултипа продемонстрировано ниже:

Рис. 1.26 **Отображение визеля в тултипе**

Для отображения формул `tooltip` необходимо задавать как строку с формулой LaTex. Ниже представлен пример указания формулы:

```
"display": {
  "tooltip": "latex:%y: $f_x(%x) = %v^2$",
}
```

где `%y` - наименование показателя, отложенного на оси Y, `%x` - наименование показателя, отложенного на оси X, `%v` - значение

Ниже пример отображения тултипа с использованием данной функции:

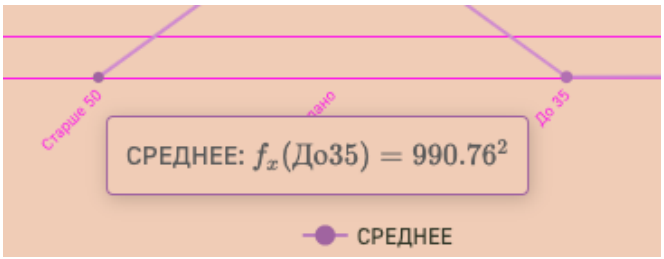


Рис. 1.27 **Математические формулы в тултипе**

В тултипах также можно указать собственную подпись, выводить значения и наименования показателей используя HTML верстку. В случае необходимости указания общего тултипа для всех показателей, его необходимо прописать в блоке `display`:

```
"display": {
"tooltip": "html:общий формат тултипов с заменой %x %y %v <b>%percent %marker"
```

```
},
```

где **%x** и **%y** - наименования показателей, отложенные на **xAxis** и **yAxis** соответственно, **%v** - значение показателя, **%percent** - Вывод показателя в процентах (для дэшлетов “Пирог” и “Донат”), **%marker** - отображения цвета показателя в круге

В случае необходимости указания тултипа для каждого показателя индивидуально, **tooltip** указывается в блоке **style**, находящийся в блоке **dataSource**:

```
"dataSource": {
"style": {
"measures|<dimension>": {
"tooltip": "html:<b>Тултип для конкретной серии с заменой %x %y %v %percent %marker</b>"
}
}
}
```

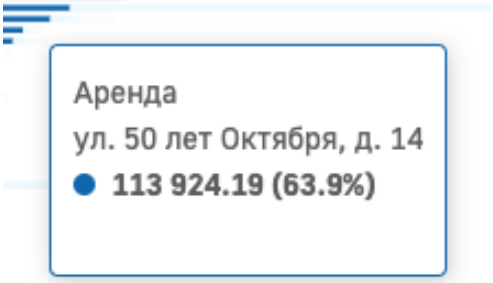


Рис. 1.28 **Пример отображения HTML-тултипа**

1.4 Опция url

- тип STRING
- Используется в типах деша: **external, internal**
- Является ссылкой на html файл (для **external**) или на js файл с компонентом написанном на React и содержащем export default (для **internal**)

url: путь к файлу из раздела resources датасета (текущего или любого другого существующего).

Пример: ссылка на файл template.html, который находится в разделе resources того же датасета, где и дешлет:

`"res:template.html"`

Ссылка на тот же файл, но если он находится не в текущем датасете, а в датасете с `schema_name = "ds_20"`:

`"res:ds_20:template.html"`

1.5 Опция bgImage

“bgImage” : ““

- тип STRING
- Вставляет фон-картинку для визеля
- В качестве зн-ия принимает ссылку на ресурсы
- Пример: `"bgImage": "res:round.png"`

1.6 Секция options

Описывает различные дополнительные настройки при отображении деша

“options” : [],

- массив STRING;
- Пример `"options": ["DisplayAllBadges","HideLegend"]`.

1. `DisableLoadData` - Запрещает загрузку у кубовых дешей
2. `DisplayAllBadges` - Показать все подписи на графиках без учета пересечений значений
3. `DisplayAllVeryBadges` - Показать все подписи принудительно
4. `!TopBarMenu` - Скрыть меню визеля
5. `!TopBar` - Скрыть заголовок дэша
6. `HideLegend` - Скрыть легенду на графике
7. `!DisplayOverall` - Для дэша типа “таблица” убирает итоговые строки внизу таблицы
8. `EastPanel` - дублирует дэш в правой выдвигающейся панели (рекомендуется использовать только для управляющего дэша)
9. `Chat` В меню визеля (колесико) появляется чат, привязан к этому визелю
10. `bgContain` - Фон-картинку подстраивает под контейнер визеля
11. `EastPanel` - Разместить виджет в левом меню
12. `!DisplayAxisYMarks` - Скрыть ось Y у дэша (работает для двумерных дэшей)
13. `!DisplayAxisXMarks` - Скрыть ось X у дэша (работает для двумерных дэшей)

14. `TooltipXAxisTitle` - Отображение полного наименования показателя размерности в всплывающей подсказке (тултипе)
15. `!DisplayAxis` - Скрытие сетки и осей у двумерных дэшей
16. `!ShowBackground` - Скрытие теней у столбцов
17. `!DisplaySplitLines` - Скрытие разделительных линий у двумерных дэшей
18. `!DisplayTicks` - Скрытие рисок/тиков осей у двумерных дэшей
19. `Transparent` - фон визелей “Значение” и “Внутренний” соответствует фону подложки дэшборда
20. `XAxisValue` - переключение оси X с категориальной на численный вариант отображения (применяется для двумерных дэшей)

Помимо общих опций, есть опции для конкретный пользователей. Указываются данные опции в блоке style конкретной размерности. Ниже представлен пример указания опции для конкретной размерности:

```
"dataSource":{
 "koob":"ch.max_example",
 "style":{
  "category":{
   "Специалисты":{
      "color":"red",
      "options":[
        "AlwaysLast",],

   },
  },
 },
}
```

Ниже представлены опции для конкретных размерностей:

1. `AlwaysLast` - Независимо от включенной сортировки набора данных, данный показатель будет отображаться последним. Реализовано для двумерных (столбики) и одномерных дэшей (бублик и пирог)
2. `DisableLegend` - По умолчанию данный показатель будет отображаться после нажатия на легенду. (реализовано для двумерных дэшей)

1.7 Секция hierarchy

Описывает иерархию измерений, например для реализации функции DrillDown

“hierarchy” : [],

- тип массив STRING;
- создание иерархии у кубов; используется для описания каскадных фильтров в управляющем дэше
- Пример: `"hierarchy": ["sex=>age=>dt", "category=>degree=>sex"]`.

1.8 Секция onClickDataPoint

Поле `onClickDataPoint` описывает поведение клиентской части LuxmsBI при нажатии на один из указанных показателей в LPE-выражении, используемом для описания поведения.

Для изменения поведения можно пользоваться следующими функциями:

1. `setKoobFilters` - фильтрация дэшей по значению размерности при нажатии на данное значение размерности. Функция имеет три аргумента: `название_источника_данных.название_куба`, `название размерности (по которой необходимо отфильтровать данные)`, `массив с указанием условия фильтрации` - первым элементом массива должен быть указан знак условия фильтрации (`=`, `!=`, `<`, `>`, `<=`, `>=`) после этого указывается размерность, по нажатию на которую, в массив подставится значение размерности. Указать можно как одну размерность для фильтрации, так и несколько: `setKoobFilters('источник.куб', 'ключ',['=',значение], 'ключ', ['=',значение], ...)` Пример: при нажатии на сегмент дэша со значением размерности “Мужчины” фильтр для дэшей будет установлен в следующее положение:

```
"filters": {
  "указанное_вторым_параметром_название_размерности": [
    "=",
    "Мужчины"  ]

}
```

2. `navigate` - переключение на указанный в выражении дэшборд/датасет. Функция имеет минимум два аргумента: `Элемент для перехода` - Возможные значения: `dboard` - для перехода на другой дэшбод текущего датасета, `segmentId` - для перехода на другой датасет (в случае, если в аргументах указан только переход на датасет, то переход будет осуществлен на дэшборд с номером, аналогичным изначальному) `ID элемента для перехода` - Для перехода к датасету необходимо указывать схему датасета, для дэшборда номер.

Пример использования LPE-выражения с вышеописанными функциями представлен ниже:

На изображении ниже представлен дэш “Данные” и его конфигурационный файл:

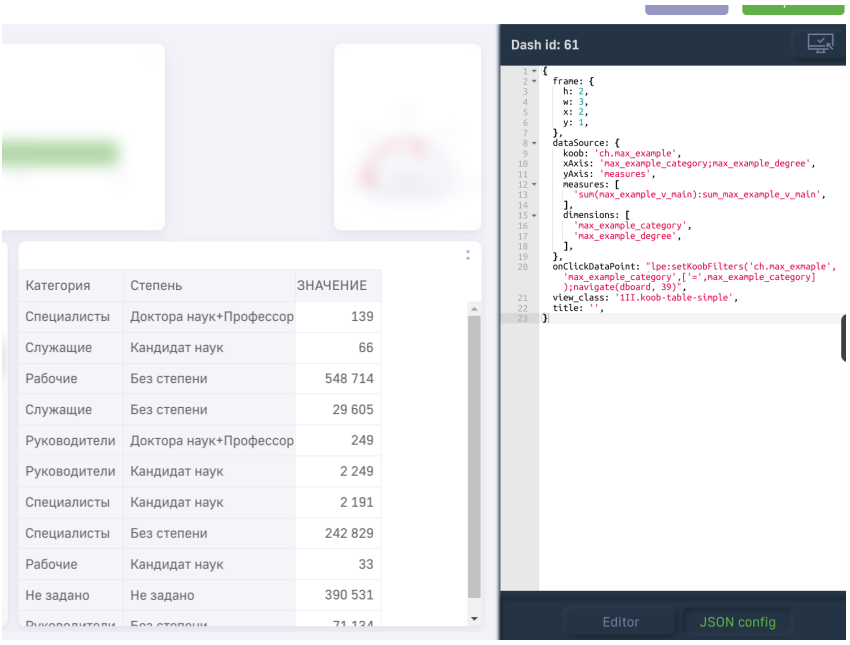


Рис. 1.29 **Дэш “Данные” и конфигурационный файл**

Ниже представлен сам конфигурационный файл дэша:

```
{
 "frame":{
  "h":2,
  "w":3,
  "x":2,
  "y":1,
 },
 "dataSource":{
   "koob":"ch.max_example",
   "xAxis":"max_example_category;max_example_degree",
   "yAxis":"measures",
   "measures":[
     "sum(max_example_v_main):sum_max_example_v_main",],

   "dimensions":[
    "max_example_category",
    "max_example_degree",],

 },
 "onClickDataPoint":"lpe:setKoobFilters('ch.max_example',
 'max_example_category',['=',max_example_category]);navigate(dboard,39)",
 "view_class":"1II.koob-table-simple",
 "title":"",
}
```

В данном примере, при нажатии на ячейку таблицы, происходит переход на дэшборд 39 и на данном дэшборде все дэши фильтруются, у которых в конфигурации указан фильтр по размерности, указанной в LPE-выражении и используется указанный куб.

Существует также возможность в функции `navigate` указать 4 аргумента, для перехода к конкретному датасету на конкретный дэшборд. Пример функции представлен ниже:

```
navigate(segmentId, ds_demo117, dboard, 1)
```

3. `openModal` - при нажатии на точку (сегмент) дэша откроется модальное окно с дэшем, id которого указано в конфигурационном файле. Пример использования данной функциональности представлен ниже:

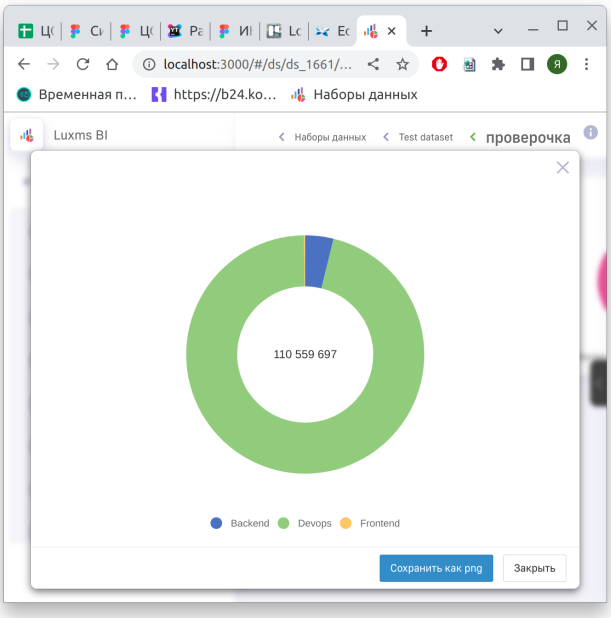


Рис. 1.30 **Открытое модальное окно с дэшем “Донат”**

Конфигурационный файл с примером использования функции `openModal`:

```
{
 "frame": {
   "h": 4,
   "w": 6,
   "x": 0,
   "y": 0,
 },
 "dataSource": {
   "koob": "my_new_excels.my_new_excels",
   "xAxis": "department",
   "yAxis": "measures;sex",
   "measures": [
     "avg(height):avg_height",  ],

   "dimensions": [
     "department",
     "sex",  ],

 },
 "onClickDataPoint": "lpe:openModal(dashlet(210))",
 "view_class": "1II.stacked-column",
 "title": "",
}
```

Для использования `openModal` необходимо аргументом передать функцию `dashlet` и id необходимого для отображения дашлета (в случае на примере выбран дашлет с id = 210)

4. `navigateUrl` - используется для перехода на внешний источник с подстановкой значения из таблицы. Пример использования данной функциональности представлен ниже:

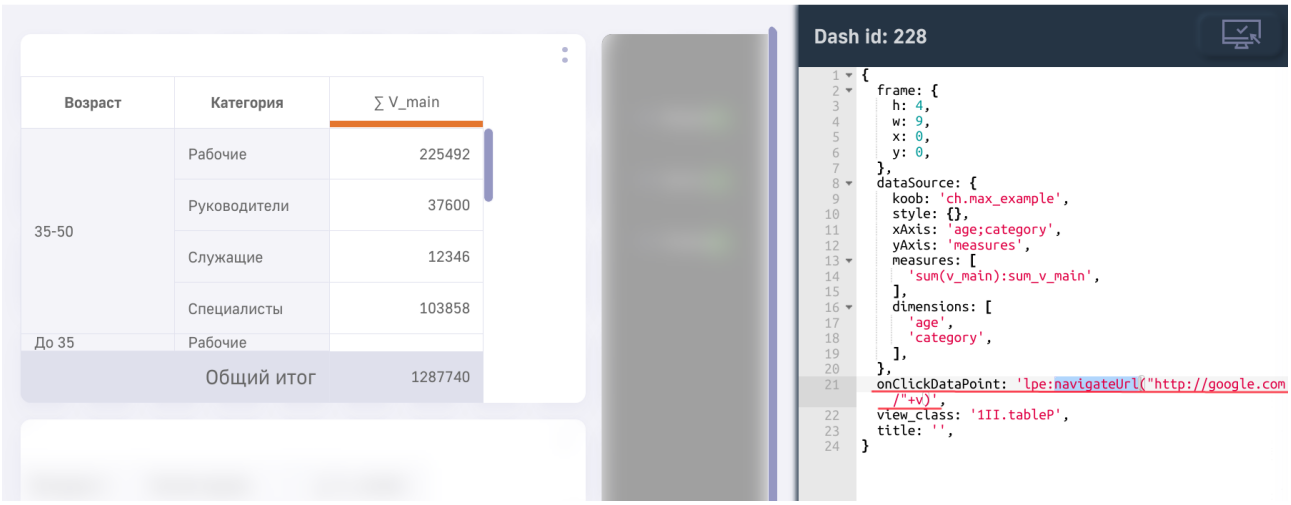


Рис. 1.31 **navigateUrl для таблицы tableP**

Конфигурационный файл с примером использования функции `navigateUrl`:

```
{
 "frame": {
   "h": 4,
   "w": 9,
   "x": 0,
   "y": 0,
 },
 "dataSource": {
   "koob": "ch.max_example",
   "style": {},
   "xAxis": "age;category",
   "yAxis": "measures",
   "measures": [
     "sum(v_main):sum_v_main",  ],

   "dimensions": [
     "age",
     "category",  ],

 },
 "onClickDataPoint": "lpe:navigateUrl('http://google.com/'+v)",
 "view_class": "1II.tableP",
 "title": "",
}
```

Параметры, необходимые для функции `navigateUrl`:

- в кавычках указывается url для перехода
- `+v` указывается в случае необходимости подставновки значения из таблицы в url и переходе с учетом `id` из таблицы

На примере, продемонстрированном выше, в случае нажатия на ячейку со значением `37600`, браузер откроет страницу по следующему адресу `http://google.com/37600`

Данный функционал реализован только для дэшей типа “koob-table-simple” и “tableP”

5. `menuItem` - используется для совместного использования **hierarchy** и функций **onClickDataPoint**.

Ниже представлен пример добавления фильтрации и вызова lookup-таблицы в выпадающий список по клику:

```
"onClickDataPoint": "lpe:showDrilldownMenu(menuItem('Доп.пункт.меню',
  setKoobFilters('luxmsbi.orders_ful','country',['=',country])),attachment(1))",
```

В `menuItem` аргументами являются все команды, которые необходимо указать в списке. Данную команду можно прописать следующим образом:

```
"onClickDataPoint": [
'showDrilldownMenu',
[
'menuItem',
'Доп.пункт.меню',
"setKoobFilters('luxmsbi.orders_ful','country',['=',country])",    ],

[
'attachment',
1,    ],  ],
```

6. `toggleKoobFilters` - множественный выбор для фильтрации на дэше “Карта”. Работает для всех слоев.

Пример:

```
"onClickDataPoint": "lpe:toggleKoobFilters('luxmsbi.custom_final','region_id',
  ['=',region_id])",
```

Для выбора цвета обводки выбранных объектов используется функция `selectedColor` в блоке `display`

1.9 Секция children

Позволяет задать дочерние деши

“children” :[]

- Используется в типах деша: **board**, **tabs**
- Принимает массив ВИЗЕЛЕЙ
- Пример:

```
"children": [{
  "view_class": "text"
  "frame": {},
  "display": {},
  "options": [],
  "dataSource": {}
  }, {...},{...}]
```

В случае необходимости отображения меню дэша у дочерних элементов, у них во view_class необходимо прописать `dashlet/название_дэшлета`

1.10 настройка isHidden

- скрывает данный дэш из отображения.

1.11 Опции для визеля Pie

Вывод значений для дэша “Пирог” можно реализовать различными способами, для этого реализованы следующие опции (ввод значения производится в массиве `options`):

```
"options": [
  "DisplayAllBadges",
  ...]
```

1. `DisplayAllBadges` - Вывод наименование показателя, его абсолютное и относительное значение
2. `DisplayBadgesPercent` - Выводится только относительное значение (в процентах)
3. `DisplayBadgesValue` - Выводится только абсолютное значение

В случае необходимости вывода абсолютного и относительного значений без наименования показателя, необходимо указать опции `DisplayBadgesPercent` и `DisplayBadgesValue`.

1.12 Опции для визеля koob-table-simple/tableP

Для визеля koob-table-simple существует возможность указывать стили CSS для столбцов таблицы. Реализованы следующие опции: 1. `color` - Указание цвета значений в столбце (указывается код цвета `#HEX`) 2. `fontStyle` - Указание стилизации отображения значений столбца (указание значения `italic` отображает текст курсивом) 2. `fontSize` - Указание размера шрифта значений в столбце (указываются относительные значения в процентах относительно значения по умолчанию) 3. `fontWeight` - Установка насыщенности шрифта значений (указание значения `bold` отображает текст полужирным начертанием) 4. `textDecoration` - Добавление оформление текста с использованием линии (указание значения `underline` подчеркивает текст) 5. `backgroundColor` - Указание цвета фона значений в столбце (указывается код цвета `#HEX`) 6. `minWidth` - Указание минимального значения ширины столбца в относительных единицах 7. `whiteSpace` - Опция, для переноса текста в ячейке, в случае, когда он не помещается в ячейке (указать значение `wrap`)

Для вышеописанных опций присутствует также возможность использования LPE-выражений для различного отображения ячеек в зависимости Пример использования данных ключей конфигурации представлен ниже:

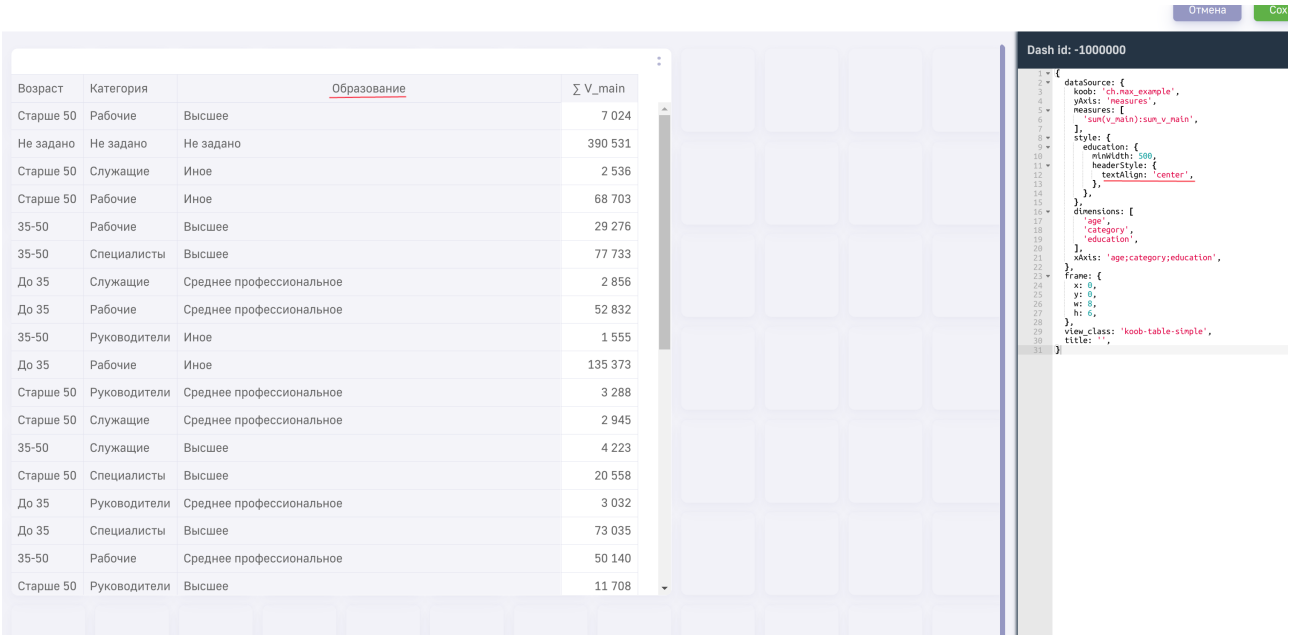
```
"style": {
    "measures": {
      "b": {
        "color": "#a6c497",
        "minWidth": 20,
        "backgroundColor": "#771111",
      },
      "c": {
        "color": "#4ab6e8",
        "fontSize": "lpe:if(c>400000, '120%', '80%')",
        "minWidth": 10,
        "fontStyle": "lpe:if(c>400000, 'italic', '')",
        "fontWeight": "lpe:if(c>400000, 'bold', 'normal')",
        "textDecoration": "lpe:if(c>400000, 'underline', '80%')",
        "backgroundColor": "lpe:if(c > 400000, '#ffaabb', 'transparent')",
      },
    },
  },
```

Для таблицы `tableP` стилизация строк реализована аналогично стилизации столбцов

Для стилизации заголовков таблицы, внутри объекта стилизации факта/размерности необходимо указать объект `headerStyle` и внутри него прописывать необходимые стили. Это может понадобится для отцентровки заголовка столбца (`textAlign: "center"`). Ниже представлен пример конфигурации для стилизации заголовка таблицы:

```
"style": {
   "education": {
     "minWidth": 500,
     "headerStyle": {
```

```
      "textAlign": "center",
    },
  },
},
```

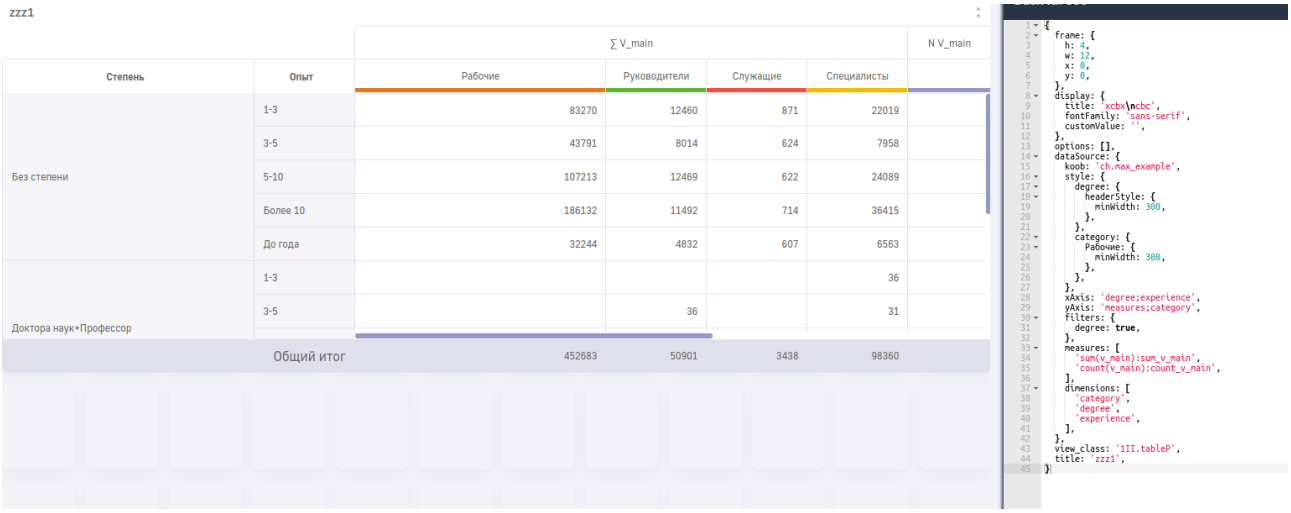


Возраст	Категория	Образование	∑ V_main
Старше 50	Рабочие	Высшее	7 024
Не задано	Не задано	Не задано	390 531
Старше 50	Служащие	Иное	2 536
Старше 50	Рабочие	Иное	68 703
35-50	Рабочие	Высшее	29 276
35-50	Специалисты	Высшее	77 733
До 35	Служащие	Среднее профессиональное	2 856
До 35	Рабочие	Среднее профессиональное	52 832
35-50	Руководители	Иное	1 555
До 35	Рабочие	Иное	135 373
Старше 50	Руководители	Среднее профессиональное	3 288
Старше 50	Служащие	Среднее профессиональное	2 945
35-50	Служащие	Высшее	4 223
Старше 50	Специалисты	Высшее	20 558
До 35	Руководители	Среднее профессиональное	3 032
До 35	Специалисты	Высшее	73 035
35-50	Рабочие	Среднее профессиональное	50 140
Старше 50	Руководители	Высшее	11 708

Рис. 1.32 **Стилизация заголовка у таблицы koob-table-simple**

Указание ширины для столбцов, отложенных на оси X в tableP производится в headerStyle:

```
"degree": {
  "headerStyle": {
    "minWidth": 300,
  },
},
```



Степень	Опыт	∑ V_main				N V_main
		Рабочие	Руководители	Служащие	Специалисты	
Без степени	1-3	83270	12460	871	22019	
	3-5	43791	8014	624	7958	
	5-10	107213	12469	622	24089	
	Более 10	186132	11492	714	36415	
	До года	32244	4832	607	6563	
Доктора наук+Профессор	1-3				36	
	3-5		36		31	
Общий итог		452683	50901	3438	98360	

Рис. 1.33 **Указание ширины для столбца с размерностями**

Таблицу koob-table-simple также можно использовать для редактирования значений в базе

данных. Для этого необходимо совершить ряд действий: 1. Открыть список кубов в административной панели 2. Открыть режим редактирования куба, в котором необходимо проводить редактирование значений 3. Открыть вкладку “Конфигурация” и прописать в конфигурацию следующее:

```

"table": "maxexample", // таблица и схема, куда пишутся
"schema": "public",
"primary_key": [  "id"  ]  // список ключей этой таблицы
```

4. Сохранить изменения
5. Перейти в клиентскую часть и в конфигурации дэша **koob-table-simple** прописать:

```
"onClickDataPoint": 'edit'
```

Для поля `koob` должен быть указан куб, конфигурация которого была отредактирована на шаге 3. После этого вы можете редактировать значения в таблице нажав на ячейку, которую необходимо редактировать. Ниже представлен пример редактирования таблицы:

Категория	Ученая степень	id	Анкеты
0Рабочие	Без степени!@#$	165	10
1Специалисты	Без степени 3	42100009	140 005 615
Не задано	Не задано	15 738	14 553
Не задано	Не задано	3 090	279
Не задано	Не задано	13 506	27 922

Рис. 1.34 **Редактирование значения в koob-table-simple**

Помимо этого, можно дать доступ на редактирование только одному из столбцов. Для этого, необходимо указать `"onClickDataPoint": 'edit'` в блоке style для конкретного факта/размерности.

Также существует возможность скрытия одного из столбцов в koob-table-simple. Для этого необходимо в его стилях прописать `isHidden: true`. Пример:

```

style: {
measures: {
sum_v_main: {
isHidden: true,
},
},
},
```

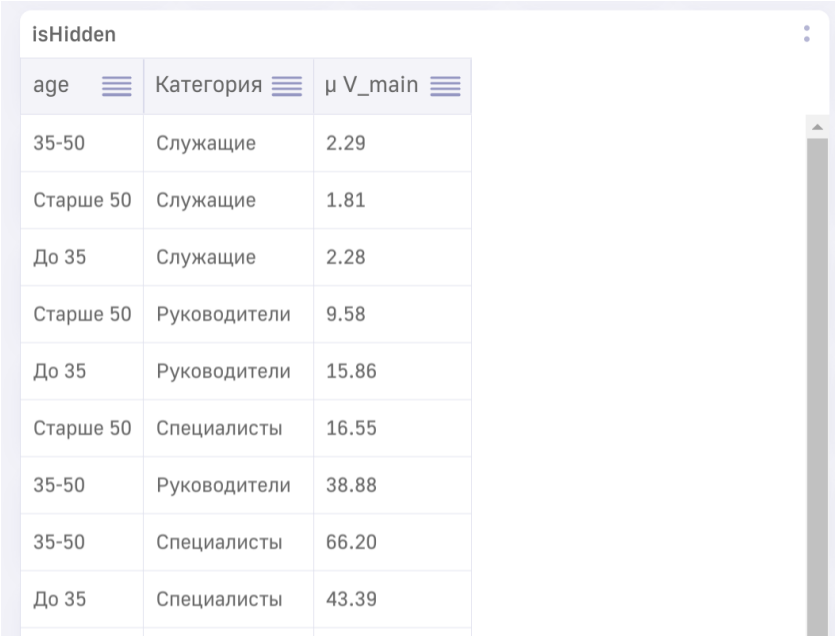


Рис. 1.35 **Скрытия столбца в таблице koob-table-simple**

1.12.1 Секция saveAbilities

Отображает кнопку для экспорта и сохранения данных, на которых строится деш

“saveAbilities” : [ ]

- тип массив STRING;
- форматы для сохранения, появляется кнопка в меню с сохранением формата;
- Пример: `"saveAbilities": ["csv","parquet","arrow"]`.

1.13 Конфигурационные опции для визеля “Значение”

Для визеля “значение” (`label`/`text`) существует ряд конфигурационных опций, работающих только для данного визеля. Ниже представлен список опций с возможными значениями:

- `title` - указание заголовка для дэша. Заголовок указывается над значением, может использоваться для описания значения. Ниже представлено изображения примера работы опции `title`

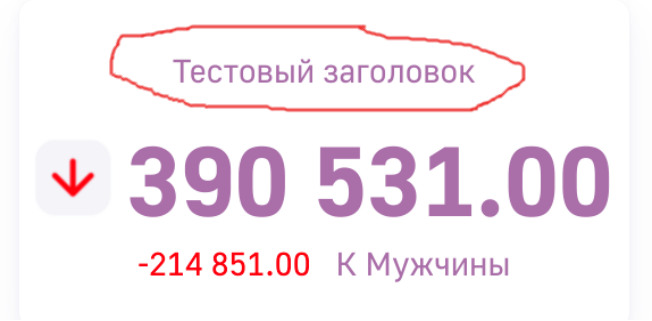


Рис. 1.36 **Стилизация дэша “Значение”**

- `bgColor` - Указание цвета фона дэша (указывается код цвета `#HEX`)
- `color` - Указание цвета значений в столбце (указывается код цвета `#HEX`)
- `fontFamily` - Указания конкретного шрифта для дэша. Существует возможность выбрать из следующих вариантов шрифта:
  - Golos UI
  - Arial
  - Times New Roman
  - Helvetica
  - Courier New
  - Courier
  - Verdana
  - Georgia
  - Garamond
  - Bookman
  - Trebuchet
  - Tahoma
  - Arial Black
  - Comic Sans MS
  - Impact
- `fontSize` - указание размера шрифта в относительных единицах. Указывается типом значения NUMBER
- `customValue` - Данная опция используется для указания прописанного текста в значении данного ключа. При установке данного ключа данные не отображаются, отображается только введенный текст в значение `customValue`

1.14 Конфигурация lookup-таблицы

1.14.1 Настройка ширины столбцов

В конфигурации lookup-таблицы существует возможность использования определенных стилей CSS для оформления отображения столбцов таблицы. Ниже приведены примеры, в которых продемонстрированы ключи для работы lookup-таблицы:

1. `width` - установка ширины столбца в lookup-таблице
2. `white-space` - управление обработкой пробельных символов внутри элемента. Возможные варианты значений:

- `normal` - Последовательности пробелов объединяются в один пробел. Символы новой строки в источнике обрабатываются, как отдельный пробел. Применение данного значения при необходимости разбивает строки для того, чтобы заполнить строчные боксы.
- `nowrap` - Объединяет последовательности пробелов в один пробел, как значение `normal`, но не переносит строки (оборачивание текста) внутри текста.
- `pre` - Последовательности пробелов сохраняются так, как они указаны в источнике.
- `pre-line` - Последовательности пробелов объединяются в один пробел.

2 Примеры конфигурации дэшей в LuxmsBI

После создания дэша на дэшборде с использованием в административной панели раздела “Дэшборд” или с использованием self-service конструктора LuxmsBI на экране будет отображен созданный дэш. Ниже представлен вариант отображения дэша и его файл конфигурации после создания:

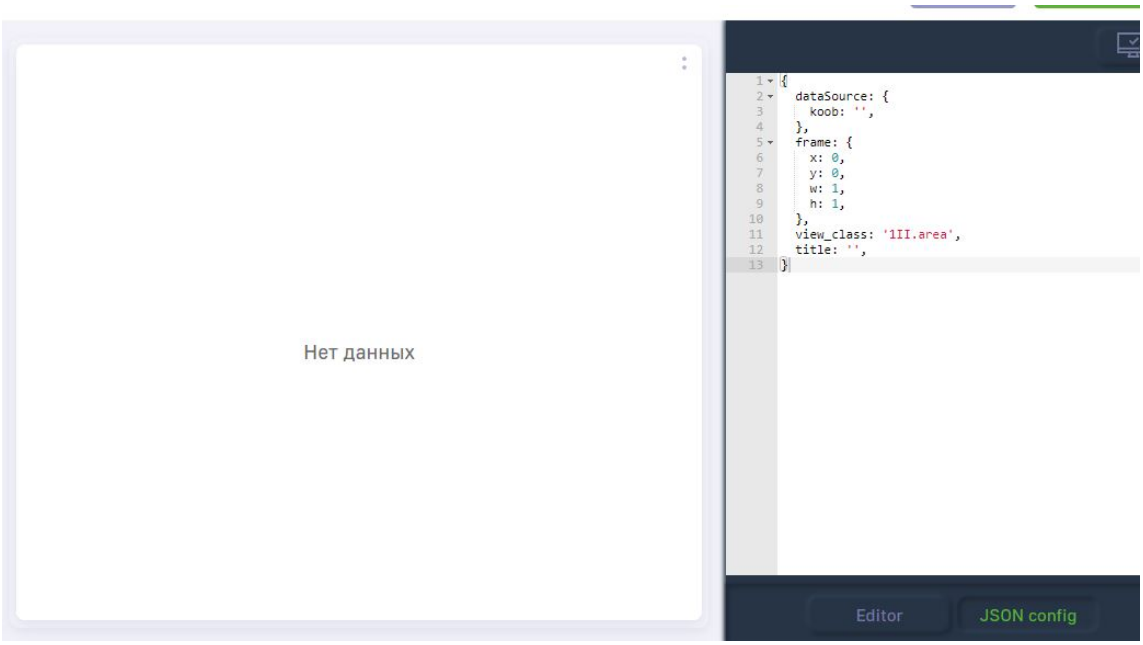


Рис. 2.1 **Дэш после создания**

2.1 Поле title

В поле `title` указывается заголовок дэша. Заголовок указывается в кавычках. Ниже представлен пример использования `title`:

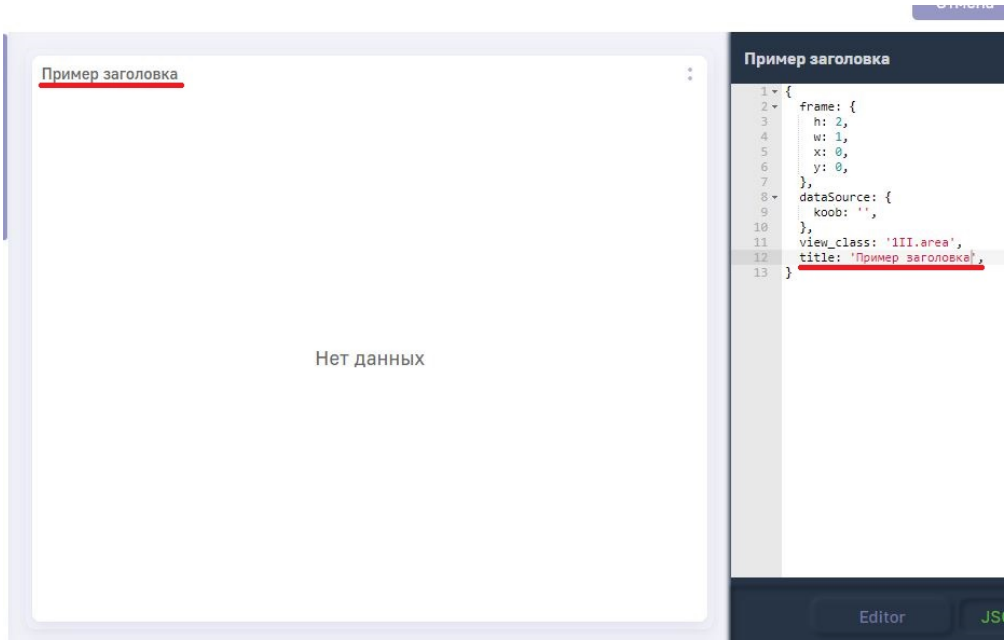


Рис. 2.2 **Поле `title`**

2.2 Поле view_class

Поле `view_class` используется для отображения выбора варианта отображения данных (визеля). После выбора дэша и переносе его на дэшборд используя административную панель / конструктор self-service существует возможность заменить вариант отображения, заменив значение `view_class`. Список доступных ключей представлен ниже:

1. `area` - Области
2. `stacked-area` - Области-штабели
3. `column` - Столбики вертикальные
4. `stacked-column` - Штабели вертикальные
5. `bar` - Столбики горизонтальные
6. `stacked-bar` - Штабели горизонтальные
7. `line` - Линии
8. `scatter` - Точки
9. `spline` - Плавная линия (сплайн)
10. `table` - Таблица
11. `tableP` - Таблица с подитогом/итогом
12. `koob-table-simple` - Таблица фильрации данных
13. `waterfall1d` - Водопад
14. `correlationnew` - Пузырьки (корреляция)
15. `gauge` - Спидометр
16. `semicircle` - Полуспидометр
17. `thermometer` - Термометр
18. `radar` - Радар
19. `pie` - Пирог
20. `bublik` - Бублик

21. `scales` - Весы
22. `funnel` - Воронка
23. `label / value / text` - Значение
24. `list` - Список
25. `VizelKoobControl` - Управляющий дэш
26. `pivot/table` - Сводная таблица
27. `html` - Дэш html
28. `image` - Изображение
29. `internal` - Внутренний
30. `external` - Внешний

2.3 Секция frame

Секция `frame` используется для указания местоположения и размеров конфигурируемого дэша. По умолчанию, созданные дэши занимают все доступное пространство для отображения. Для отображения дэша на экране используются следующие ключи: 1. `x` - расположение дэша на дэшборде по оси X При увеличении значения для ключа `x` дэш сдвигается вправо на дэшборде. На изображении ниже представлен пример дэша, с указанным значением для 1:

Рис. 2.3 **Изменение расположения дэша по оси X**

Дэш сдвинут таким образом, что в левой части дэшборда поместится дэш со значениями `w=1/h=1` и значением `x=0`. Также можно указывать значения с числами после запятой, для более тонкой настройки расположения дэша. Ниже представлен пример отображения дэша со значением `x=3.5`

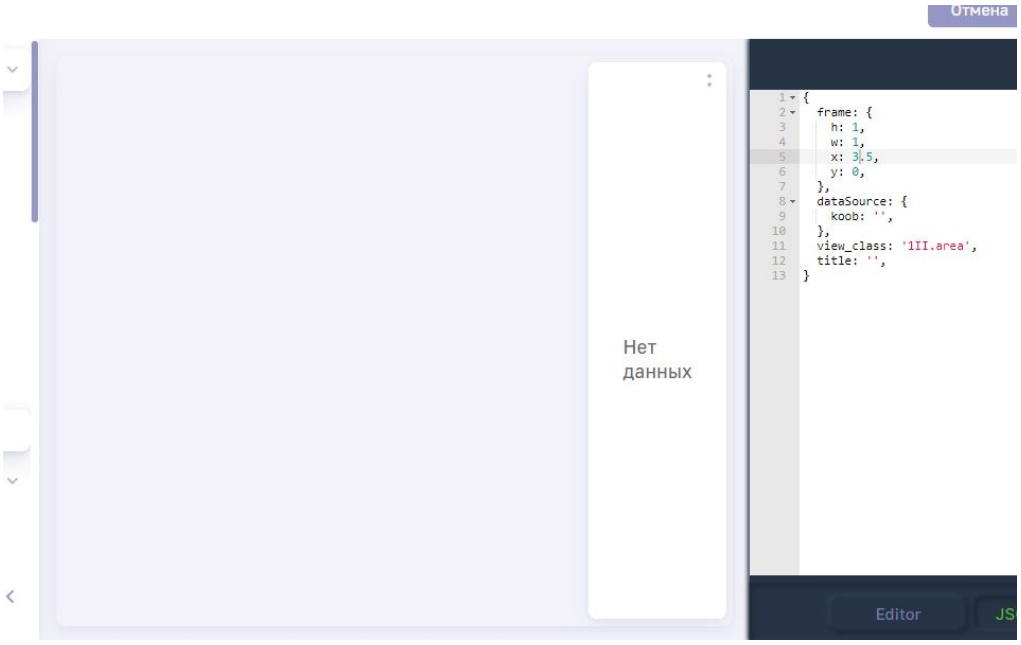


Рис. 2.4 **Изменение расположения дэша по оси X**

Как можно заметить, в левой пустой части дэшборда, можно разместить три дэша со значениями `w=1` и один дэш со значением ширины `w=0.5` Аналогичная логика работы при изменения расположения дэша по оси Y. Ниже представлены изображения для `y=1`:

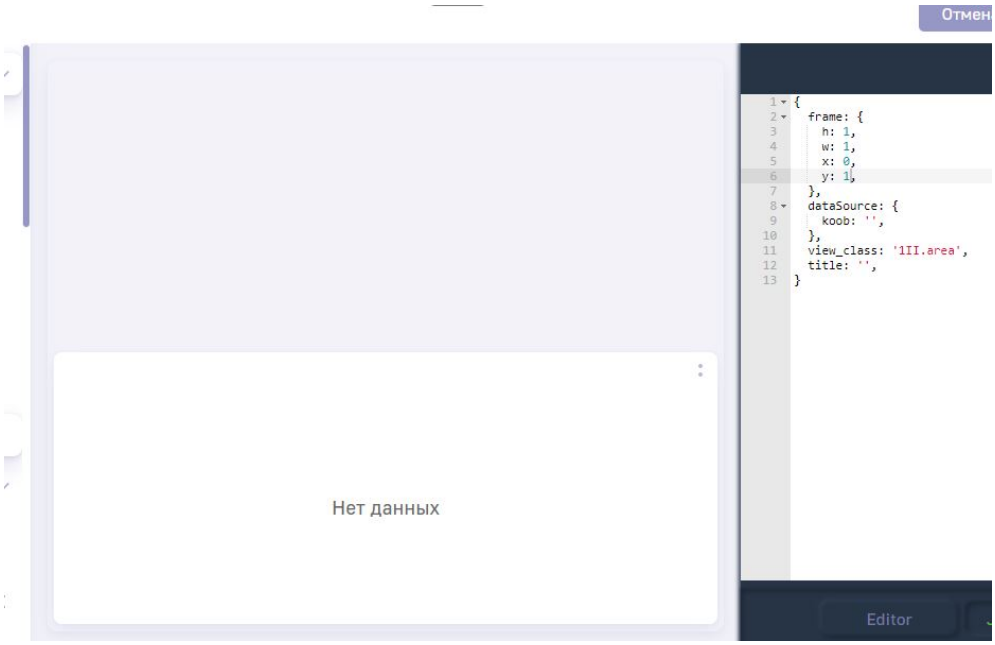


Рис. 2.5 **Изменение расположения дэша по оси Y**

и `y=3.5` соответственно:

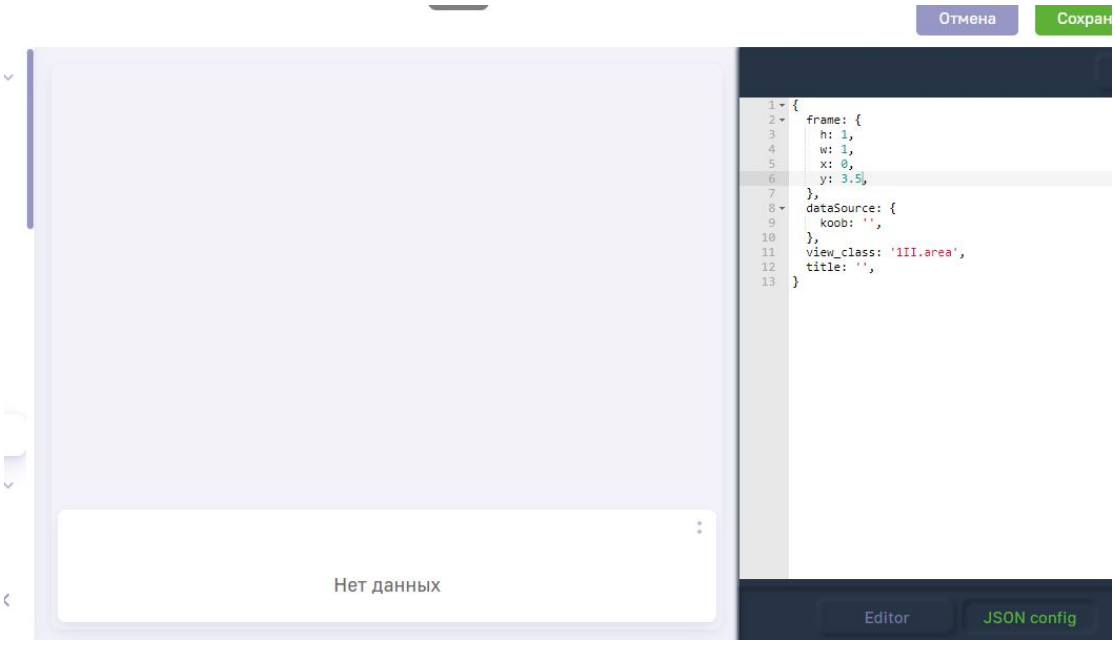


Рис. 2.6 **Дэш, с указанной координатой Y**

Логика аналогичная как и в случае изменения `x` - в верхней пустой части дэшборда, можно разместить три дэша со значениями `h=1` и один дэш со значением ширины `h=0.5`, либо один дэш со значением `h=2` и один дэш со значением ширины `h=1.5`, ниже представлен пример отображения трех дэшей на дэшборде с вышеописанными примерами:

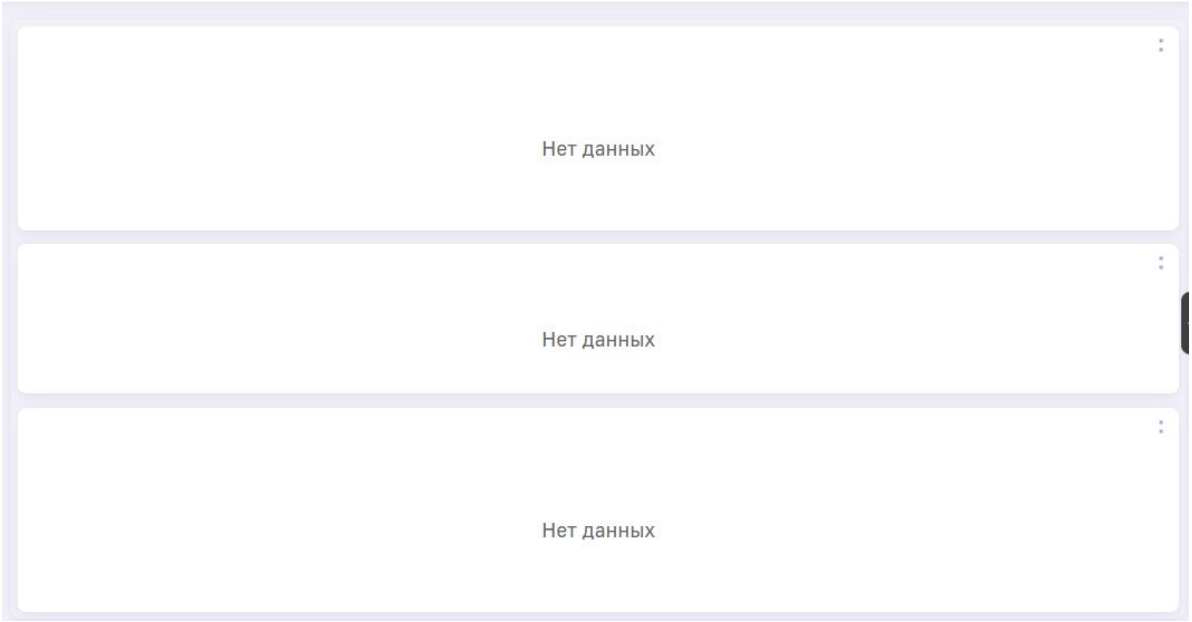


Рис. 2.7 **Три дэша, расположенные по вертикали**

Секции frame в файле-конфигурации для дэшей, представленных на изображении выше:

- Дэш 1:

```
"frame": {
   "h": 2,
   "w": 1,
   "x": 0,
   "y": 0,
}
```

- Дэш 2:

```
"frame": {
   "h": 1.5,
   "w": 1,
   "x": 0,
   "y": 2,
}
```

- Дэш 3:

```
"frame": {
   "h": 2,
   "w": 1,
   "x": 0,
   "y": 3.5,
}
```

Используя секцию `frame` и изменяя вышеуказанные показатели в относительных единицах, вы сможете отобразить дэши на дэшборде в любом удобном для вас варианте.

`frame` можно также указать и в строковом варианте:

```
frame: "(x1,y1),(x2,y2)"
```

где **x1 = x y1 = y x2 = x + w y2 = y + h**

2.4 Секция dataSouce

Секция `dataSouce` используется для указания источника данных, куба, редактирования варианта отображения данных. Ниже будут представлены подробности работы с разделом.

2.4.1 Управление данными

Рассмотрим конфигурацию дэша, только что созданную на дэшборде с использованием административной панели / конструктора:

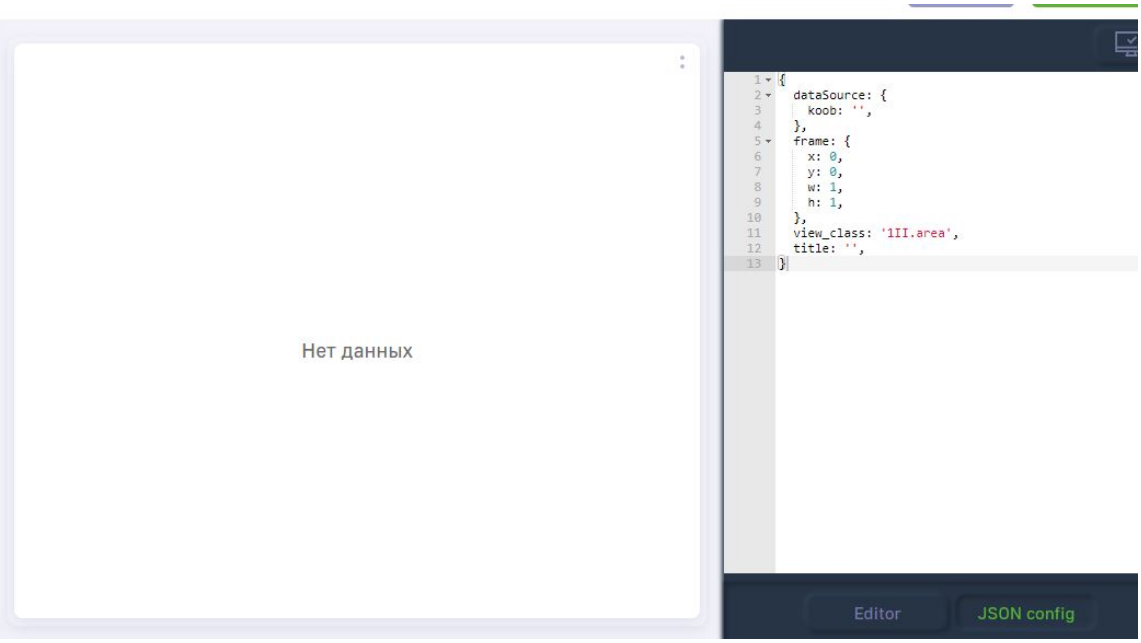


Рис. 2.8 **Пустой дэш**

В секции `dataSource` автоматически вставлен ключ `koob`, который предназначен для выбора источника данных и куба.

Для указания куба и отображения данных в дэше необходимо в поле `koob` прописать следующее значение:

```
"koob": "название_источника.название_куба"
```

Внимание! Для корректной работы LuxmsBI, используйте для названия кубов и источников данных латинские буквы, цифры и нижнее подчеркивание (но не первым символом).

Пример заполнения данного поля представлен ниже:

```
"koob": 'clickHouse.test2'
```

где `clickHouse` - название источника данных, `test2` - название куба

После этого, в секции `dataSource` необходимо указать факты (`measures`) и размерности (`dimensions`). Для этого необходимо прописать в массив `measures` - факты следующим образом:

```
"measures":[
"агрегационная_функция(показатель):название",
...]
```

Аналогичным образом необходимо прописать и список размерностей:

```
"dimensions":[
"название_размерности",
```

```
... ]
```

Ниже представлен список доступных агрегационных функций для фактов:

1. `sum`
2. `avg`
3. `count`
4. `min`
5. `max`
6. `var_pop`
7. `var_samp`
8. `stddev_samp`
9. `mode`
10. `stddev_pop`
11. `median`

Над фактами можно проводить различные арфиметические операции, ниже представлен пример подсчета среднего значения факта:

```

"measures": [
'sum(v_main)/count(v_main):avg_v_main'    ],
```

Ниже представлен пример заполнения массивов `measures` и `dimensions`:

```
"dataSource": {
  "koob": "ch.test2",
  "measures": [
    "sum(v_main):sum_v_main",
    "count(v_main):count_v_main",],

  "dimensions": [
    "degree",
    "education",
    "experience",
    "age",],

},
```

Для отображения выбранных данных из куба, необходимо их отложить на оси X и Y. Для этого используются поля `xAxis` и `yAxis`. Все факты всегда отображаются на одной оси, поэтому для них используется обобщенный заголовок `measures`. Размерности можно отображать на различных осях.

В случае работы с одномерными дэшами (радар, пирог, донат) все данные для отображения необходимо перечислять на оси Y (`yAxis`)

Ниже представлены варианты отображения данных в зависимости от расположения на осях:

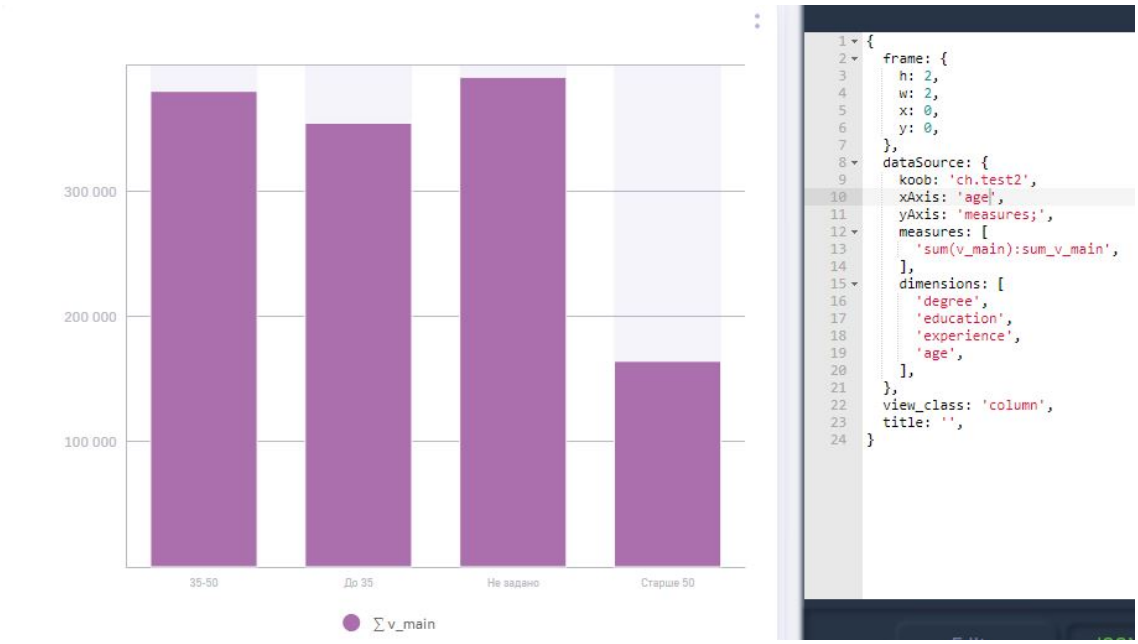


Рис. 2.9 **Дэш, с отложенными размерностями на оси X**

В случае, если размерность, будет отложена на оси X, то данные по этой размерности будут отображаться как одним цветом.

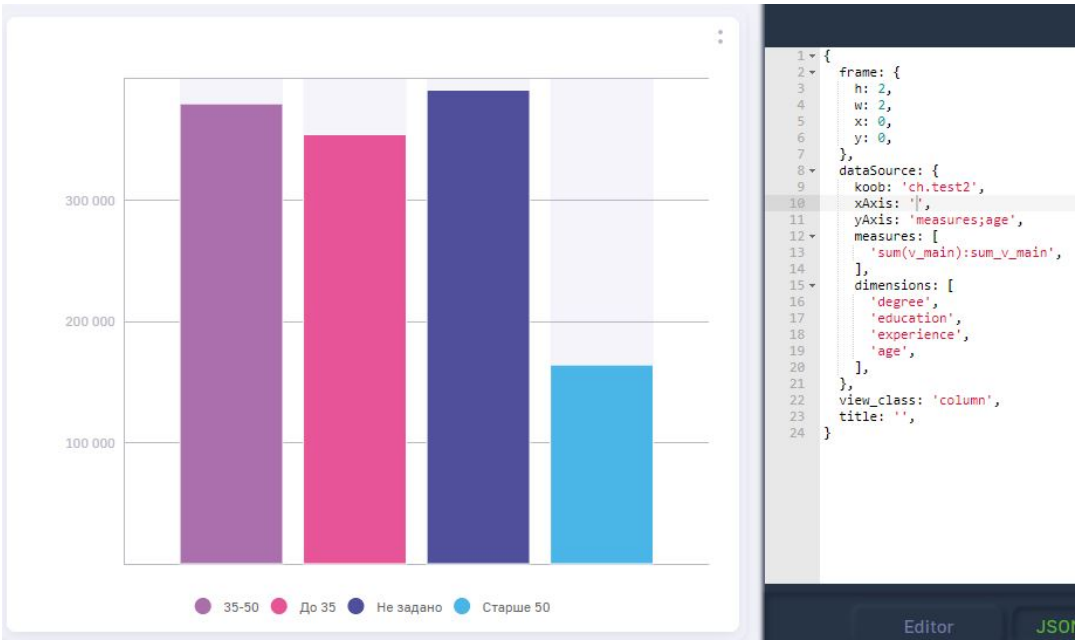


Рис. 2.10 **Дэш, с отложенными размерностями на оси Y**

В случае, если размерность, будет отложена на оси Y, то каждый показатель данной размерности будет отображен индивидуальным цветом. Ниже представлен диаграмма, у которой один показатель отложен по оси X, а другой по оси Y:

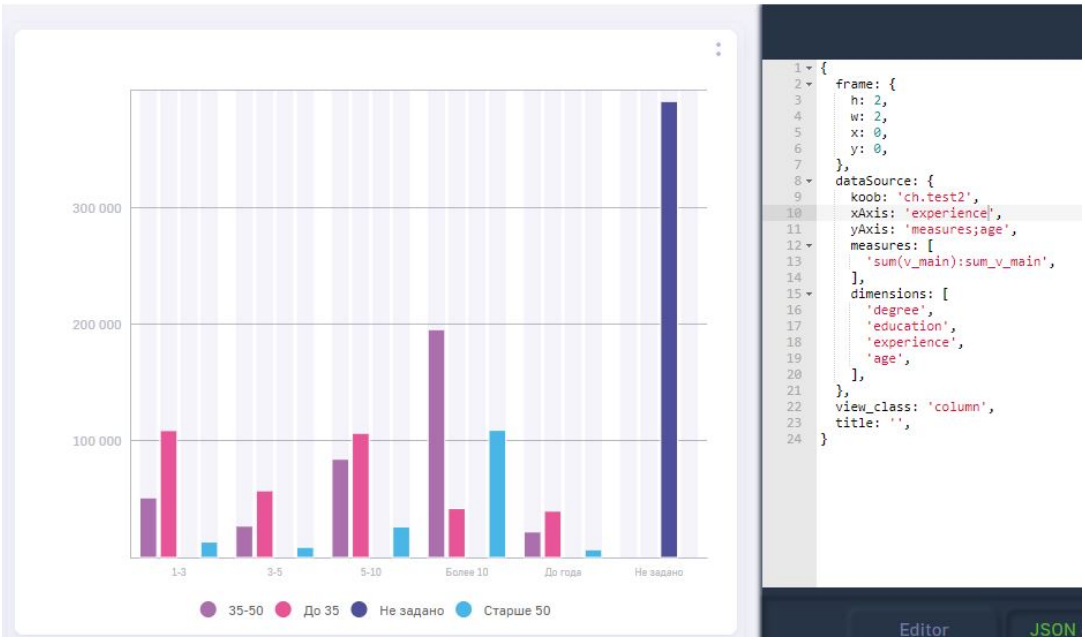


Рис. 2.11 **Дэш, с отложенными размерностями на оси X и на оси Y**

Из примера видно, что каждый цвет отображает один из рассматриваемых диапазонов возраста, которые сгруппированы по размерности `"experience"` - показателю, отложенному по оси X.

Существуют случаи, когда необходимо отфильтровать значения. На примере выше видно, что присутствует показатель “Не задано”. Из-за большого количества значений по данному показателю, дэш отображен некорректно и не передает полноценно аналитические данные. Для фильтрации данных необходимо использовать поле `filters`.

Поле `filters` можно записать в файле в двух представлениях: в виде массива и в виде объекта.

Поле `filters` в формате массива используется для указания размерностей, фильтрация которых будет воспроизводится только с использованием управляющего дэша (о конфигурации управляющего дэша описано ниже) Пример записи поля `filters` в виде массива представлен ниже:

```
"filters":[
    "degree",
    "education",
    "experience",
    "age", ]
```

Внутри массива перечислены размерности, которые будут реагировать на фильтрацию управляющим дэшем. Аналогичное действие можно совершить задав поле `filters` как объект:

```
"filters":{
    "degree": true,
    "education": true,
    "experience": true,
```

```
    "age": true,
}
```

В случае, когда нам необходимо отфильтровать какой-то показатель сразу, независимо от управляющего дэша, нам необходимо использовать объект `filters`. Вернемся вышеописанному примеру:

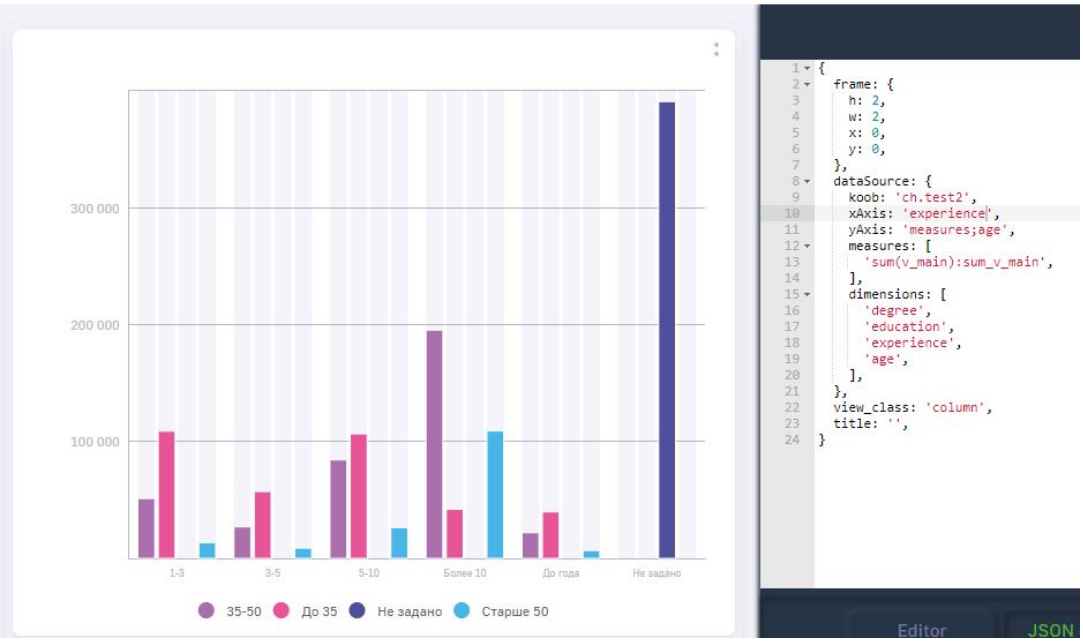


Рис. 2.12 **Дэш с неотфильтрованными данными**

Для того, чтобы отфильтровать значения “Не задано”, необходимо прописать в `filters` следующее:

```
"filters":{
 "experience": [
   "!=",
    "Не задано"]

}
```

где `experience` - наименование размерности `!=` - условие фильтрации, говорящее о том, что необходимо оставлять только показатели неравные нижеперечисленным `"Не задано"` - фильтруемый показатель

Ниже приведен пример дэша после фильтрации:

Рис. 2.13 **Фильтрация данных по условию “!=”**

Как видим, показатель “Не задано” отфильтрован и не отображен на дэше. Аналогичным образом можно оставить только один показатель прописав объект следующим образом:

```
"filters":{
 "experience": [
   "!=",
   "Не задано"]

}
```

Ниже представлен результат данной фильтрации:

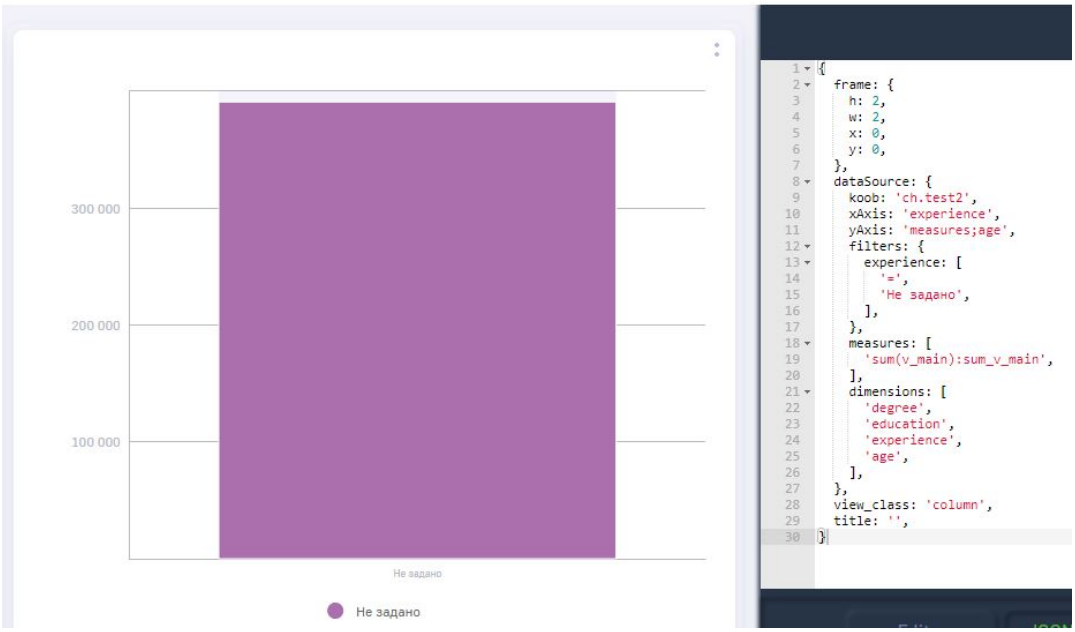


Рис. 2.14 **Фильтрация данных по условию “=”**

Как мы видим из рисунка выше, на дэше отображается только показатель, указанный в конфигурационном файле.

В случае, когда в кубе вместо пустых значений указан null, BI будет воспринимать эти данные как “подытог”. Для скрытия “подытога” необходимо отфильтровать данные по null (“!=”, “null”)

2.4.2 Стилизация дэшей

Иногда возникает необходимость стилизации дэшей: указать конкретный цвет для показателя / размерности, поменять заголовок и т.д. Для таких случаев используется объект `style`. Рассмотрим случай, когда у нас два факта и одна размерность:

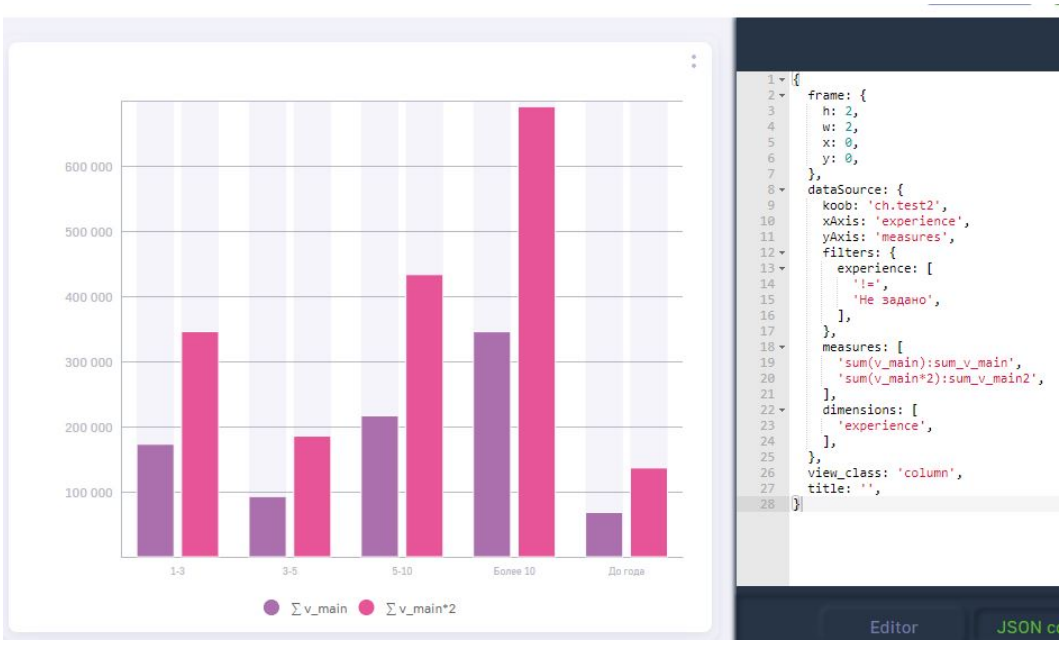


Рис. 2.15 **Дэш “Столбики” для двух фактов и одной размерности**

Объект `style` для стилизации дэшей составляется следующим образом:

```
"style":{
 "sum_v_main":{
  "title": "Факт1",
   "color": "#3b3fb8",
   "vizelType": "line",
   "strokeDash": "ShortDash",
   "unit_id": 1
  },
  "sum_v_main2":{
   ...
  }
}
```

где `sum_v_main/sum_v_main2` - названия стилизуемых фактов `title` - поле указания заголовка для факта `color` - указание цвета факта `vizelType` - указание варианта отображения факта (возможные варианты: **line** - линии, **scatter** - точки, **bar** - стоолбцы) `strokeDash` - вариант отображения факта, в случае его отображения в виде “линии”. Ниже представлен список доступных значений:

1. `Solid` - сплошная линия
2. `ShortDot` - линия отображена точками, расположенными близко относительно друг друга
3. `ShortDash` - линия отображена черточками небольшой длины
4. `ShortDashDot` - линия отображена черточками небольшой длины и точками
5. `ShortDashDotDot` - линия отображена черточками небольшой длины и двумя точками
6. `Dot` - отображение точками
7. `Dash` - линия отображена черточками
8. `LongDash` - линия отображена длинными черточками(тире)
9. `DashDot` - линия отображена черточками и точками
10. `LongDashDot` - линия отображена длинными черточками(тире) и точкой
11. `LongDashDotDot` - линия отображена длинными черточками(тире) и двумя точками;

`unit_id` - указание единицы измерения факта, в случае, если он задан в схеме датасета в таблице **units**

Аналогичную стилизацию можно проводить и с показателями размерностей, в случае если они отложени на оси Y. Ниже представлен пример стилизации:

Рис. 2.16 **Стилизованные показатели в дэше “Столбики”**

Пример конфигурации стилей для размерности приведен ниже:

```
"style":{
```

```
 "experience":{
   "1-3":{
     "color": "#4426dd",
      "title": "Начальный уровень"
    },
    "3-5":{
   ...
    },
    ...
  }
}
```

Объект `style` записывается в объект `dataSource` для всех типов дэша, за исключением типа “Значение”, описание которого продемонстрировано ниже.

2.5 Секция display

Секция `display` используется для редактирования отображения данных в дэше (сортировка, ограничения количества выводимых данных и т.д.)

Рассматрим вышеописанный пример сконфигурированного дэша “Столбики”:

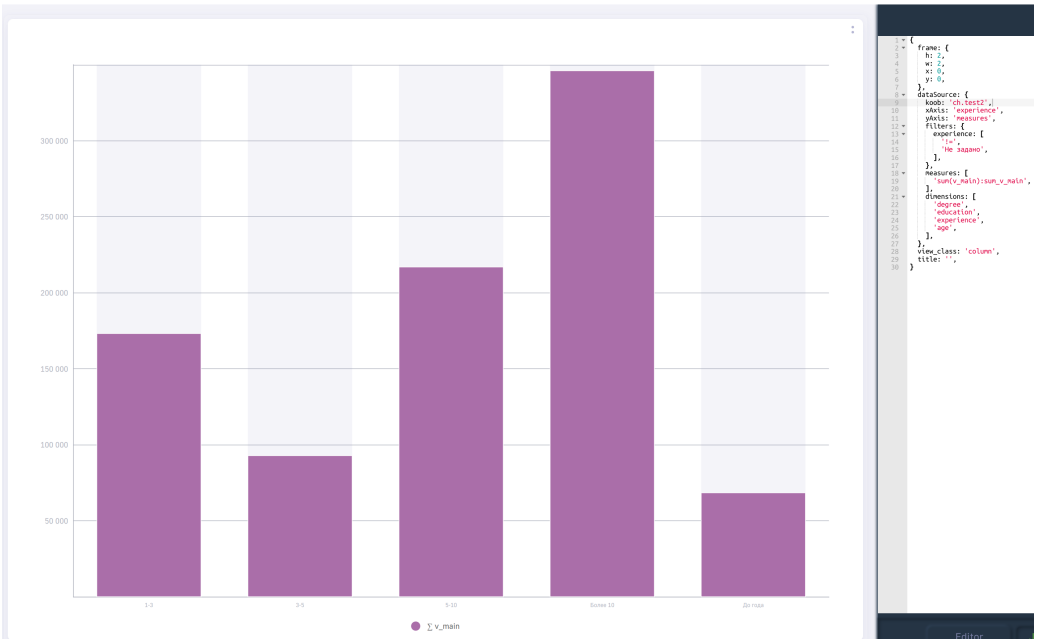


Рис. 2.17 **Дэш “Столбики”**

В случае необходимости сортировки столбцов по возрастанию/убыванию, в секцию `display` необходимо ввести следующие поля:

```
"display":{
  "sortBy": "sum_v_main"
```

```
}
```

где `'sum_v_main'` - заголовок факта, относительно которого необходимо сортировать значения.

После ввода вышеописанного блока в конфигурационный файл, данные будут отсортированы по возрастанию:

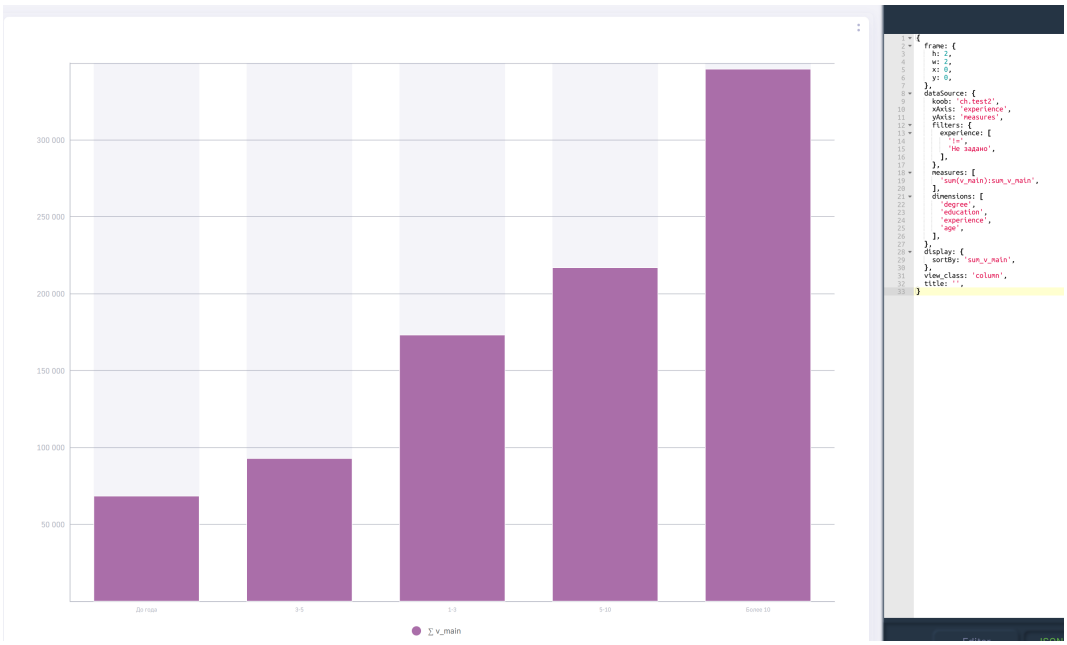


Рис. 2.18 **Дэш с отсортированными столбцами по возрастанию**

Для сортировки по убыванию для текущего факта необходимо дописать поле `sort` со значением `DESC`, следовательно, блок `display` будет описан следующим образом:

```
"display":{
  "sortBy": "sum_v_main",
  "sort": "DESC"
}
```

Ниже представлен пример отображения с вышеописанными полями.

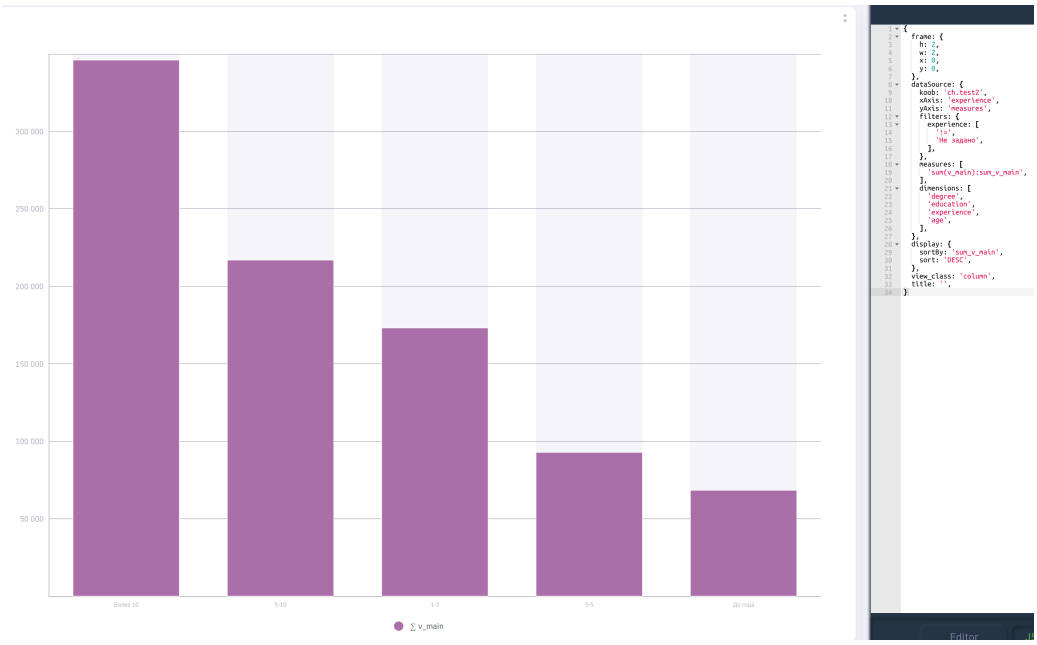


Рис. 2.19 **Дэш с отсортированными столбцами по убыванию**

Для случаев, когда размерность, имеющая тип “строка”, имеет показатели с большим количеством символов, в случае ограниченного количества места на дэшборде, названия показателей будут налезать друг на друга:

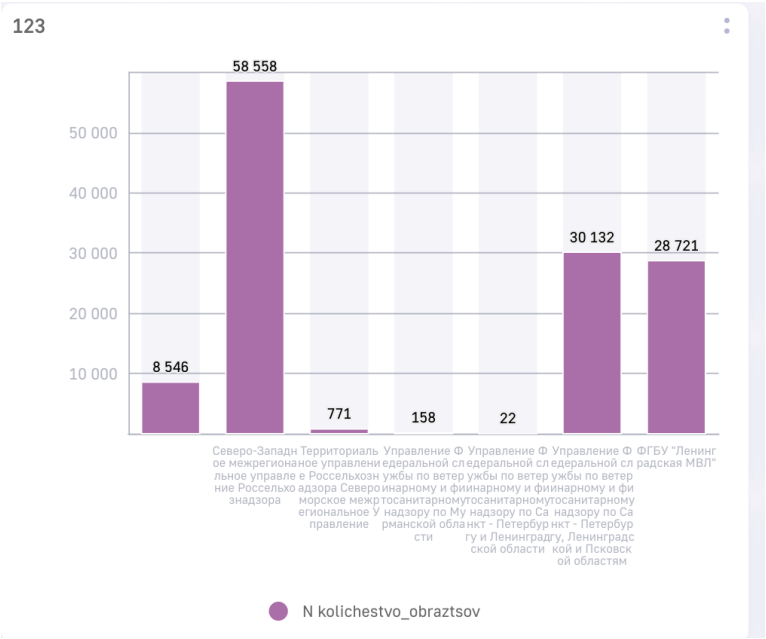


Рис. 2.20 **Наложение подписей по оси X**

Для таких случаев существуют поля `xAxisLabelLimit` - для ограничения вывода подписей по осям X и Y соответственно. Ниже представлены примеры использования данных полей:

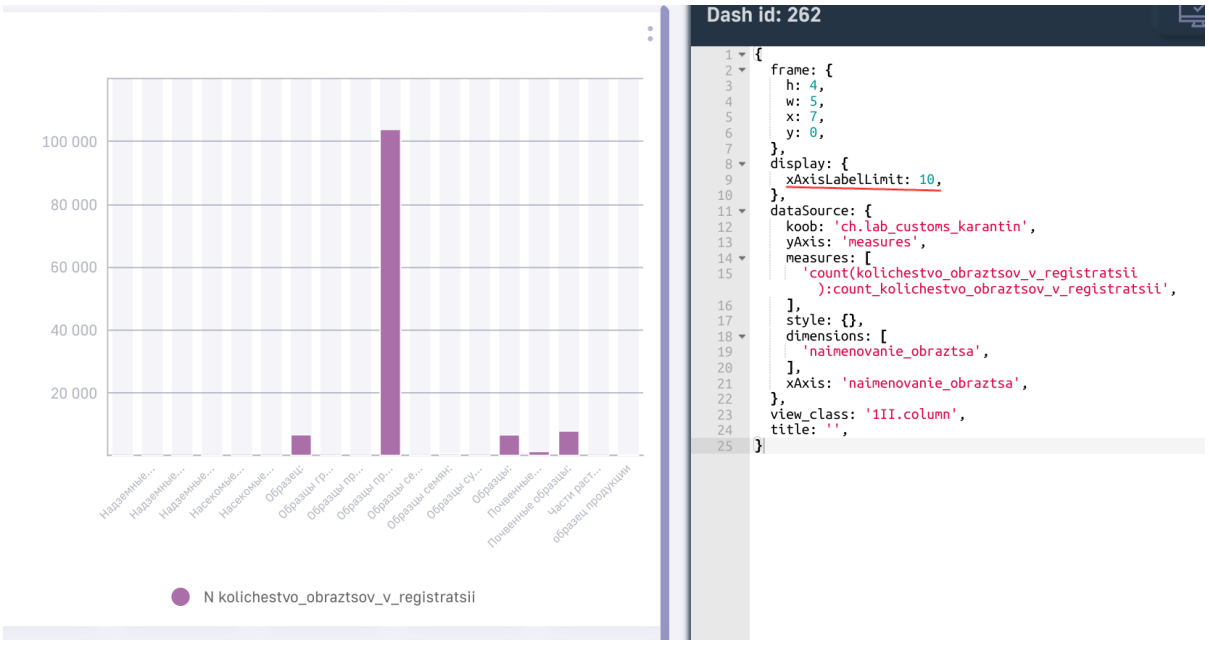


Рис. 2.21 **Использование xAxisLabelLimit для дэша “Вертикальные столбики”**

Данные поля ограничивают отображение текста на оси тем количеством символов, которое указано в конфигурационном файле. Для отображения полного наименования в всплывающей подсказке (тултипе) необходимо указать в массиве `options` опцию `TooltipXAxisTitle`.

Помимо этого, существует возможность менять угол отображения наименования показателей на оси, используя поле `rotateXLabel` со значением угла поворота. Ниже представлен пример дэша с использованием данного функционала:

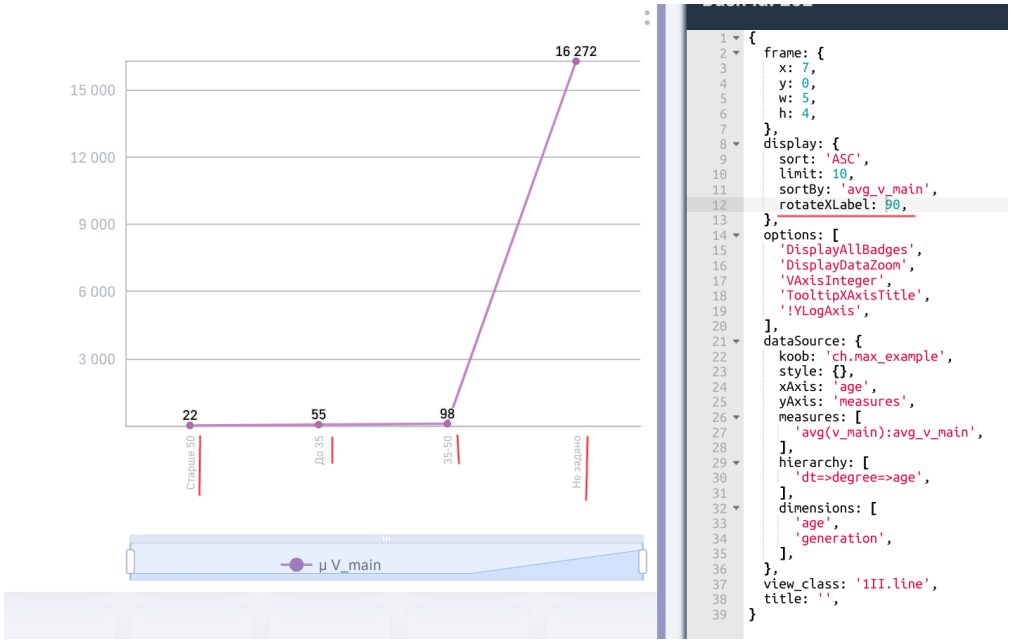


Рис. 2.22 **Использование rotateXLabel для дэша “Линии”**

Все вышеописанные действия (включая объект `style` внутри блока `dataSource`) применимы для двумерных типов дэшей (столбики, линии, точки, штабели, и т.д.)

Далее будут представлены опции для нольмерных типов дэшей: значение, термометр, спидометры и т.д.

2.5.1 Стилизация типа дэша “Значение”

Для стилизации типа дэша значение (`label`, `text`, `value`) необходимо использовать секцию `display` и использовать поля, созданные специально для рассматриваемого дэша. Ниже приведен простой пример конфигурациий дэша “Значение:

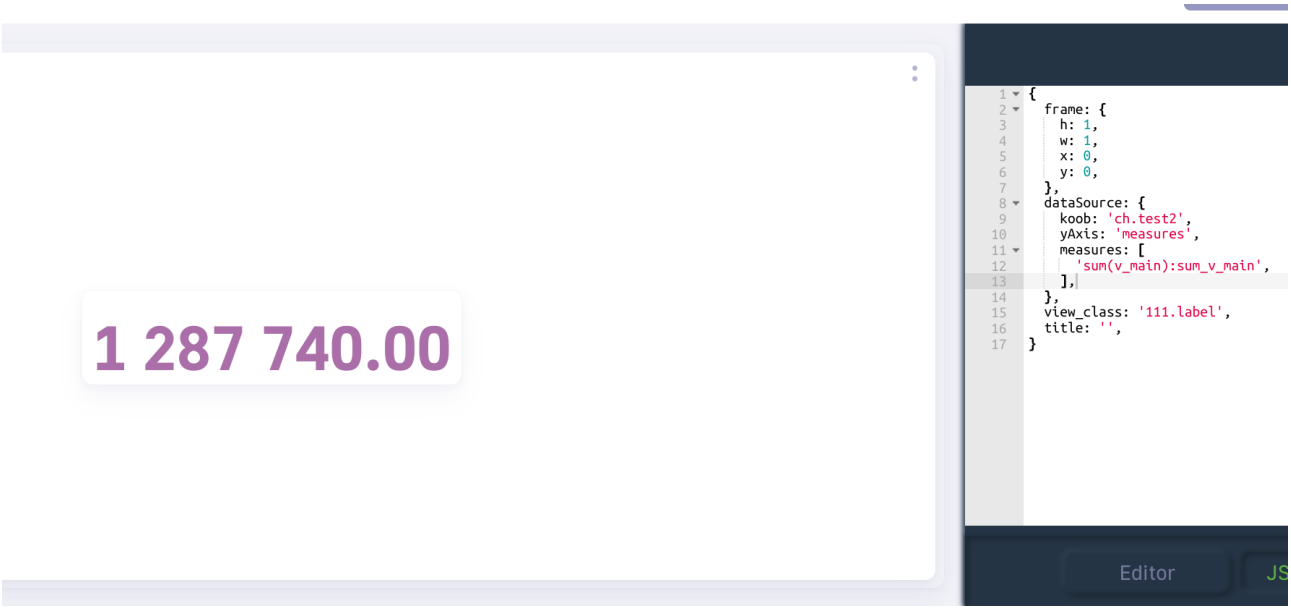


Рис. 2.23 **Дэш “Значение”**

На скриншоте в конфигурационном файле представлены все поля, описанные выше.

Данное значение существует возможность подписать сверху, для этого используется поле `title`. Для этого добавим в конфигурационный файл объект `display` с полем `title`:

```
"display": {
  "title": "Заголовок значения",
}
```

Ниже представлено отображения дэша после сохранения изменений в конфигурационном дэше:

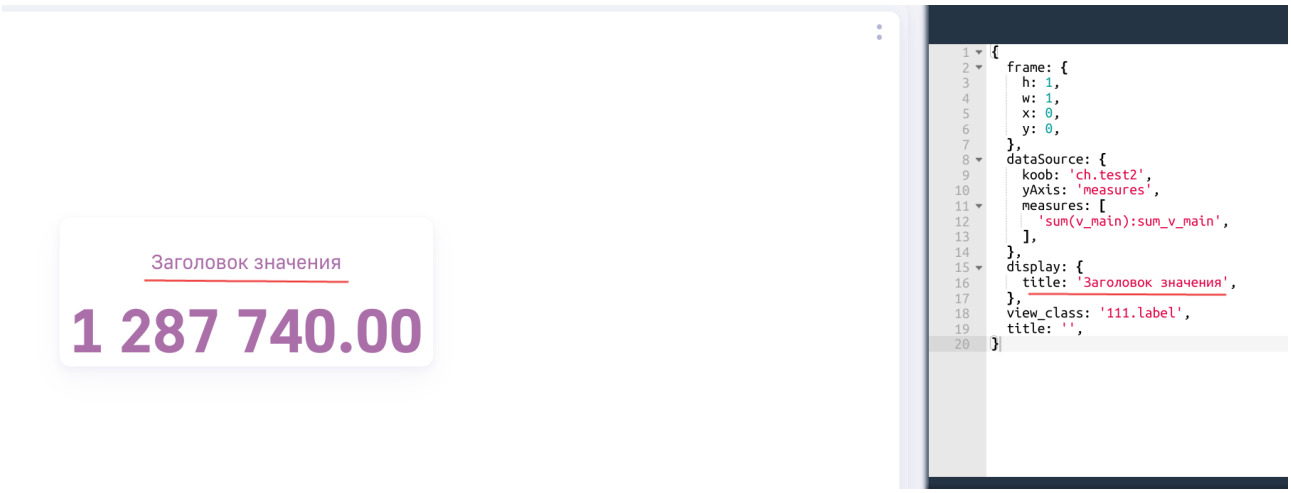


Рис. 2.24 **Указание заголовка значения**

Для изменения цвета значения и заголовка используется поле `color` с указанием шестнадцатеричного кода цвета. Ниже представлен пример изменения цвета после указания его в конфигурационном файле:

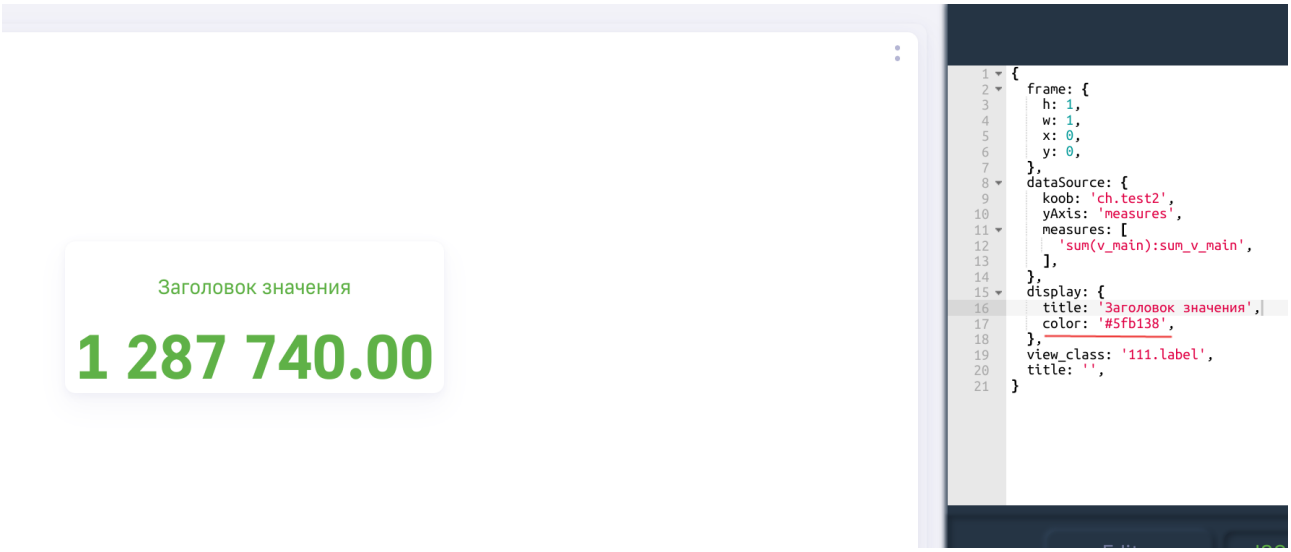


Рис. 2.25 **Изменение цвета значения**

Аналогичным образом, для дэша “Значение” существует возможность поменять цвет фона, используя поле `bgColor`:

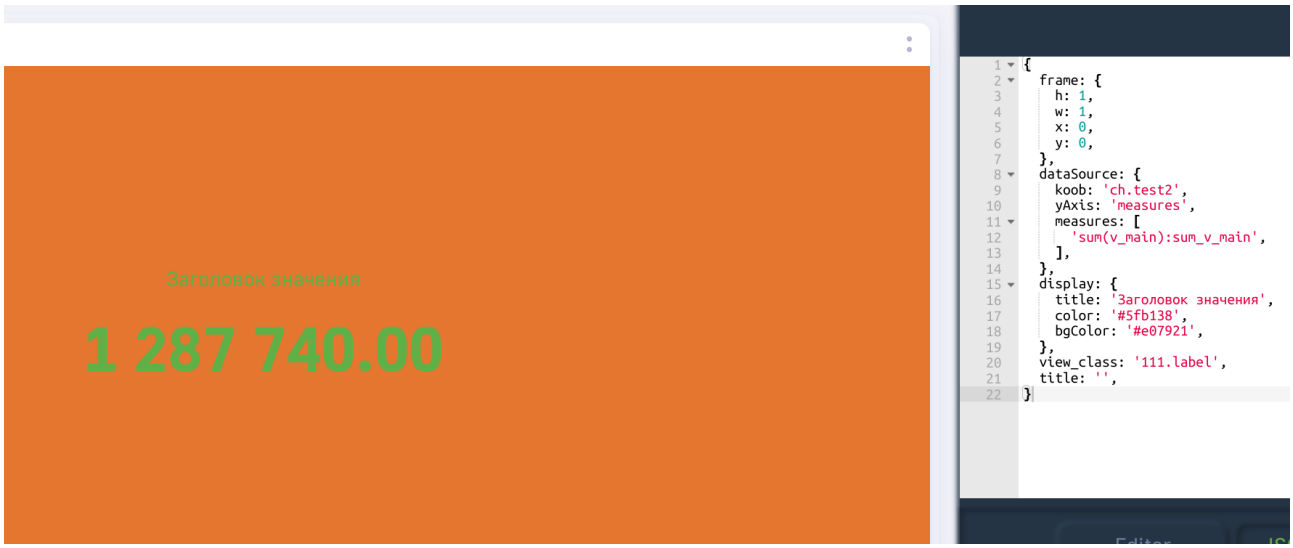


Рис. 2.26 **Изменение цвета фона для дэша “Значение”**

Также, для данного дэша присутсвует возможность указания конкретного шрифта, используя поле `fontFamily`. Полный набор вариантов шрифтов представлен в предудщем разделе.

Ниже представлен пример “Значения” с установленным шрифтов `"Times New Roman"`:

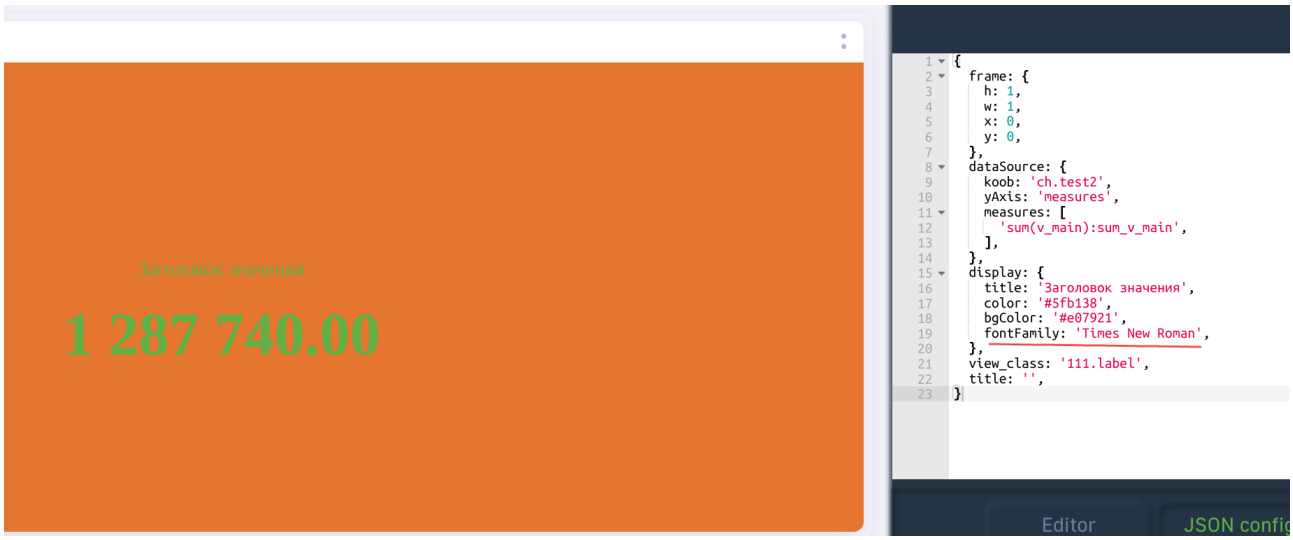


Рис. 2.27 **Установка шрифта**

Помимо этого, сущесвует возможность указать размер минимальный/максимальный или конкретный размер шрифта для значения. Для этого используются поля `maxFontSize`, `minFontSize`, `fontSize` соотвественно. Размер шрифта задается в относительных единицах. Пример использования представлен ниже:

Рис. 2.28 **Использование поля `fontSize`**

В случае, когда нам необходимо указать статичный текст в дэше, используется поле `customValue`. Ниже представлен пример использования нижеописанного поля:

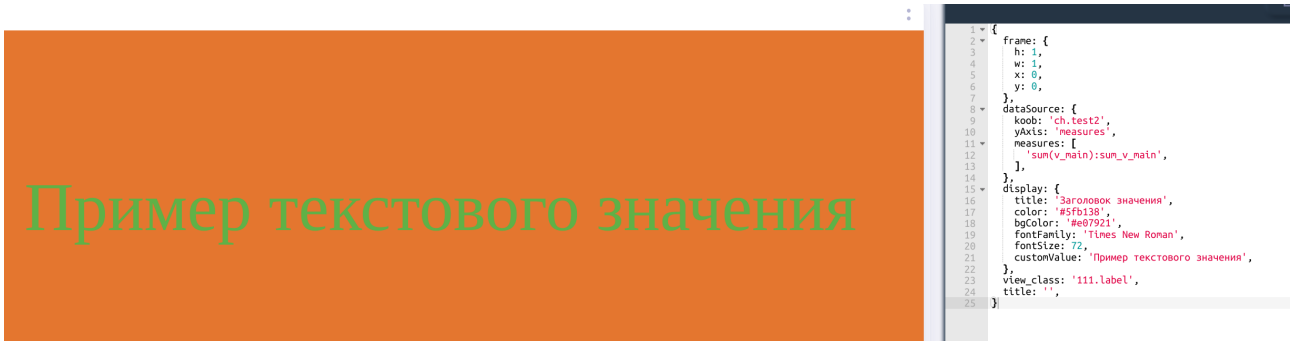


Рис. 2.29 **Поле `customValue` для дэша “Значение”**

Ниже представлен итоговый конфигурационный файл, полученный для дэша “Значение” в ходе примеров:

```
{
  "frame": {
    "h": 1,
    "w": 1,
    "x": 0,
    "y": 0,
  },
  "dataSource": {
    "koob": "ch.test2",
    "yAxis": "measures",
    "measures": [
      "sum(v_main):sum_v_main",],

  },
  "display": {
    "title": "Заголовок значения",
    "color": "#5fb138",
    "bgColor": "#e07921",
    "fontFamily": "Times New Roman",
```

```
    "fontSize": 72,
    "customValue": "Пример текстового значения",
  },
  "view_class": "111.label",
  "title": "",
}
```

2.5.2 Объект stoplights / массив range

Опции, описанные в данном подразделе, используются перовоочередно в таких типах дэша как: спидометр, полуспидометр, термометр. Для примера возьмем дэш “Спидометр” и отобразим его используя вышеописанные поля:

```
{
  "frame": {
    "h": 1,
    "w": 1,
    "x": 0,
    "y": 0,
  },
  "dataSource": {
    "koob": "ch.test2",
    "yAxis": "measures",
    "measures": [
      "sum(v_main):sum_v_main",],

    "dimensions": [
      "degree",],

  },
  "view_class": "gauge",
  "title": "",
}
```

Используя вышепредставленную конфигурацию, мы получим спидометр следующего вида:

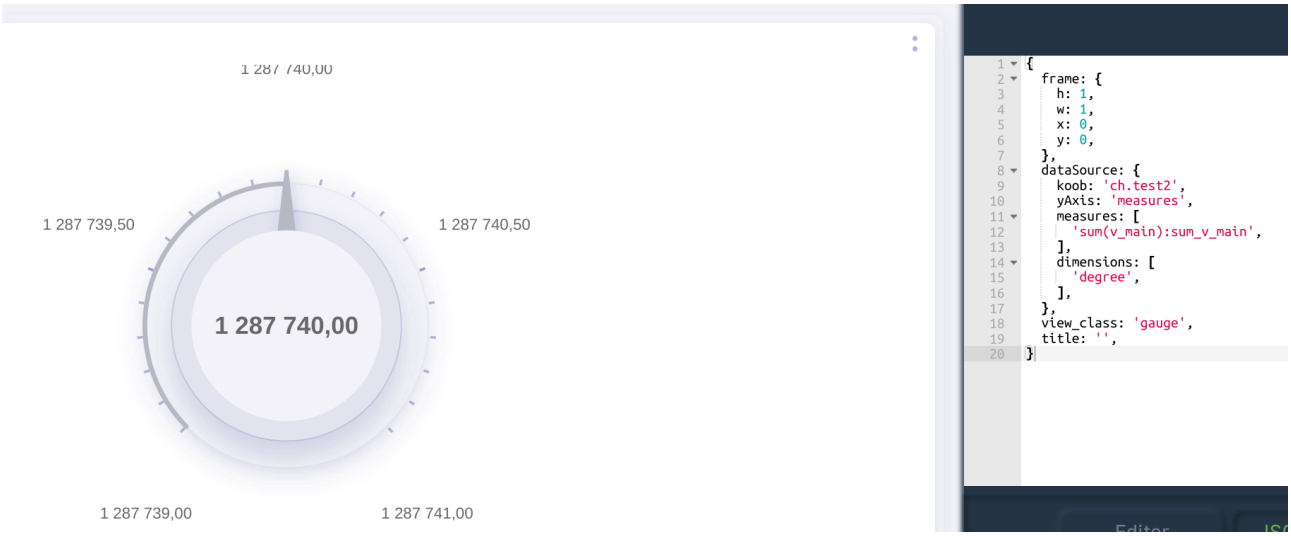


Рис. 2.30 **Дэш “Спидометр”**

Как можно заметить, в состоянии по-умолчанию спидометр отображает значения в диапазоне от `значение - 1` до `значение + 1`. С помощью массива `range` в блоке `display` можно указать отображаемый диапазон значений для спидометров/термометра. Ниже представлено изображение спидометра с указанным диапазоном от миллиона до двух миллионов:

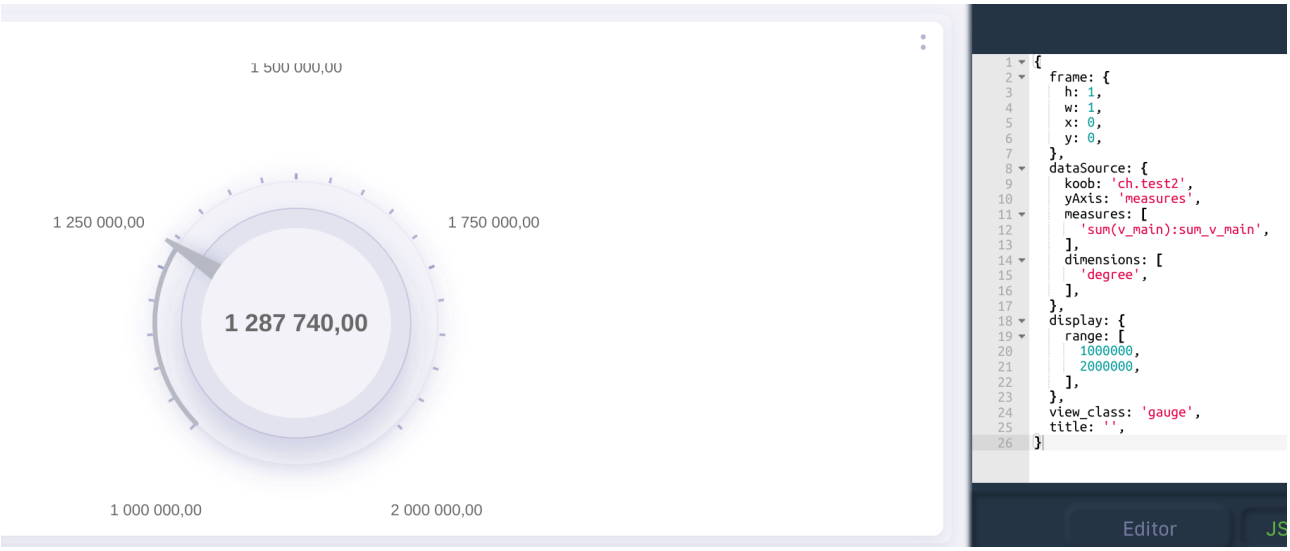


Рис. 2.31 **Спидометра с указанным диапазоном в `range`**

В большинстве случаев для спидометр используется для отображения того, что значение показателя находится в нужном диапазоне или нет (например, показать выполнение плана и т.д.). Для таких случаев необходимо использовать объект `stoplights` внутри которого массив объектов `lights`. Ниже приведен пример конструкции объекта `stoplights`:

```
"stoplight": {
    "lights": [
      {
        "name": "Плохо",
        "color": "#f05045",
        "limit": [-
          Infinity,
          1500000,],

      },
      {
        "name": "Средне",
        "color": "#f2bb05",
        "limit": [
          1500000,
          1750000,],

      },
      {
        "name": "Хорошо",
        "color": "#5fb138",
        "limit": [
          1750000,
```

```
          Infinity,],

      },
      ...],

},
```

массив `lights` состит из объектов, включающие в себя следующие поля: `name` - название зоны `color` - цвет зоны, указывается #HEX цвета `limit` - массив, в котором указывается два значения - диапазон, в котором будет отрисован цвет (`-Infinity/Infinity` используется для указания, что диапазон начинается с `-бесконечности` или до `бесконечности`)

Диапазоны, указанные в `stoplights` будут отрисованы на дэше в диапазоне, указанном в `range`

Ниже представлен пример использования `stoplights` и `range` для дэша спидометр:

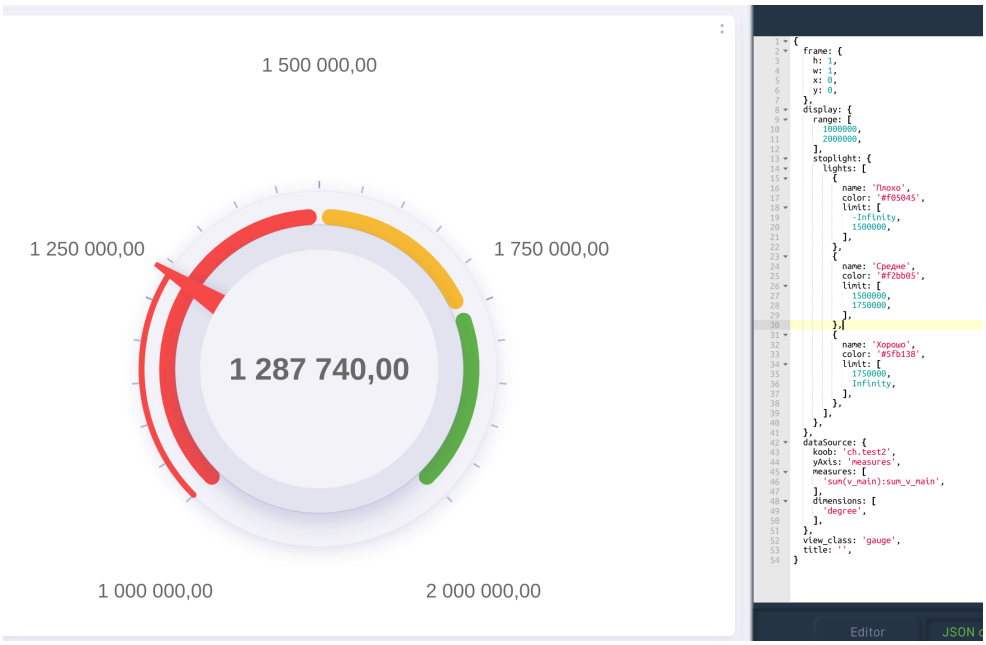


Рис. 2.32 **`stoplights` и `range` для дэша спидометр**

Конфигурационный файл данного дэша выглядит следующим образом:

```
{
  "frame": {
    "h": 1,
    "w": 1,
    "x": 0,
    "y": 0,
  },
  "display": {
    "range": [
      1000000,
      2000000,],
```

```
    "stoplight": {
     "lights": [
      {
       "name": "Плохо",
       "color": "#f05045",
       "limit": [-
        Infinity,
        1500000,],

      },
      {
       "name": "Средне",
       "color": "#f2bb05",
       "limit": [
        1500000,
        1750000,],

      },
      {
       "name": "Хорошо",
       "color": "#5fb138",
       "limit": [
        1750000,
        Infinity,],

      },],

    },
  },
  "dataSource": {
    "koob": "ch.test2",
    "yAxis": "measures",
    "measures": [
     "sum(v_main):sum_v_main",],

    "dimensions": [
     "degree",],

  },
  "view_class": "gauge",
  "title": "",
}
```

Объект `stoplights` также может использоваться и для двумерных дэшей. Ниже представлен пример использования `stoplights` для дэша “Столбики горизонтальные”:

Рис. 2.33 **`stoplights` для дэша “Столбики горизонтальные”**

Конфигурационный файл примера:

```
{
  "frame": {
    "h": 1,
    "w": 1,
    "x": 0,
    "y": 0,
  },
  "display": {
    "stoplight": {
      "lights": [
        {
          "name": "Плохо",
          "color": "#f05045",
          "limit": [-
            Infinity,
            100000,],

        },
        {
          "name": "Средне",
          "color": "#f2bb05",
          "limit": [
            100000,
            500000,],

        },
        {
          "name": "Хорошо",
          "color": "#5fb138",
          "limit": [
```

```
          500000,
          Infinity,],

      },],

    },
  },
  "dataSource": {
    "koob": "ch.test2",
    "yAxis": "measures",
    "measures": [
      "sum(v_main):sum_v_main",],

    "dimensions": [
      "degree",],

    "xAxis": "degree",
  },
  "view_class": "bar",
  "title": "",
}
```

Помимо указания конкретных значений в `stoplights` существует возможность градиентного их представления с использованием констант. Для этого необходимо использовать поле `stoplight`. Ниже представлен список констант для указания в `stoplight`:

1. `XL_GREEN_YELLOW_RED` - переход зеленый - желтый - красный
2. `XL_RED_YELLOW_GREEN` - переход красный - желтый - зеленый
3. `XL_GREEN_WHITE_RED` - переход зеленый - белый - красный
4. `XL_RED_WHITE_GREEN` - переход красный - белый - зеленый
5. `XL_BLUE_WHITE_RED` - переход синий - белый - красный
6. `XL_RED_WHITE_BLUE` - переход красный - белый - синий
7. `XL_WHITE_RED` - переход белый - красный
8. `XL_RED_WHITE` - переход красный - белый
9. `XL_GREEN_WHITE` - переход зеленый - белый
10. `XL_WHITE_GREEN` - переход белый - зеленый
11. `XL_GREEN_YELLOW` - переход зеленый - желтый
12. `XL_YELLOW_GREEN` - переход желтый - зеленый

Ниже представлен пример отображение горизонтальных столбцов с использованием `stoplight`:

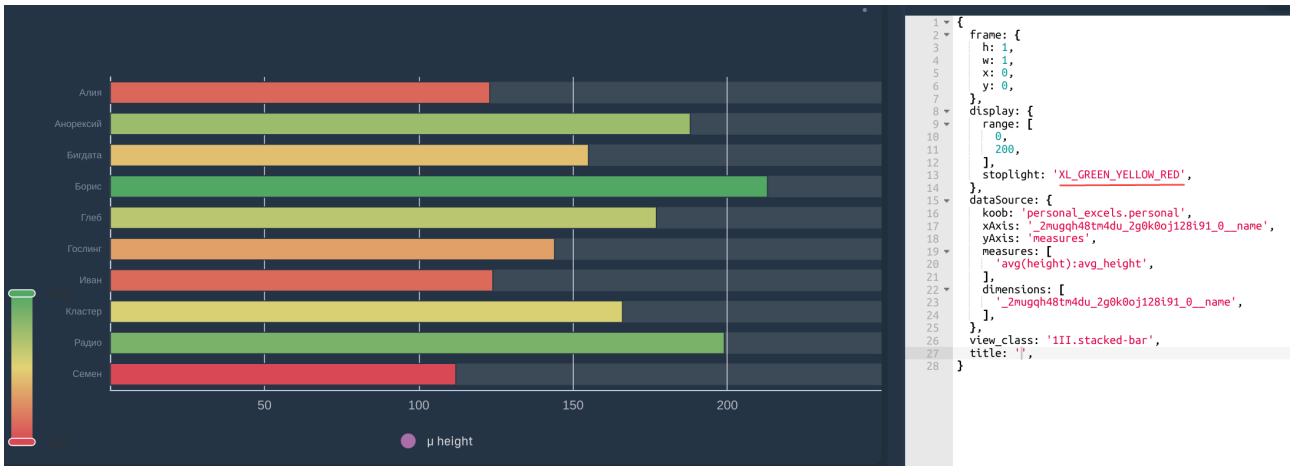


Рис. 2.34 **Использование опции stoplight**

2.6 Секция options

При конфигурации дэшей, существует набор опций, указывая которые можно добавлять-/удалять различные артефакты дэша. Данные опции прописываются в массиве `options` в слеюудщем формате:

```
"options": [
  "опция_1",
  "опция_2",
  ...]
```

Ниже представлен пример использования опций для дэша “Столбики”:

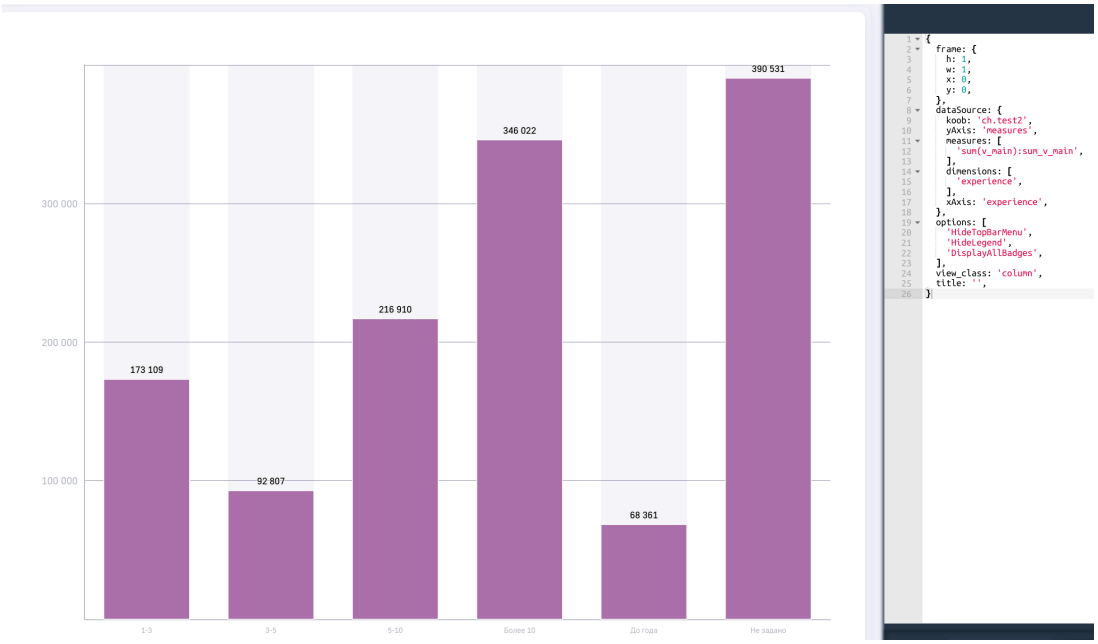


Рис. 2.35 **Пример использования опций для дэша “Столбики”**

Конфигурация дэша:

```
{
  "frame": {
    "h": 1,
    "w": 1,
    "x": 0,
    "y": 0,
  },
  "dataSource": {
    "koob": "ch.test2",
    "yAxis": "measures",
    "measures": [
      "sum(v_main):sum_v_main",],

    "dimensions": [
      "experience",],

    "xAxis": "experience",
  },
  "options": [
    "HideTopBarMenu",
    "HideLegend",
    "DisplayAllBadges",],

  "view_class": "column",
  "title": "",
}
```

где `HideTopBarMenu` - скрыть меню дэша `HideLegend` - скрыть легенду дэша `DisplayAllBadges` - отобразить все значения на графике

2.7 Пример конфигурации визеля map

Для отображения данных на карте, используется тип дэша “Карта”, имеющий инидивидуальные поля в конфигурационном файле. Ниже представлен пример конфигурирования “Карты”:

```
{
  "frame": {
    "h": 1,
    "w": 1,
    "x": 0,
    "y": 0,
  },
  "children": [
    {
      "title": "Точки",
      "display": {
        "stoplight": "XL_BLUE_WHITE_RED",
 "defaultActive": false
      },
```

```
    dataSource: {
      "koob": "sourceWithCountryAndFo.goodKoobMap",
      "style": {
        "measures": {
          "v": {
            "title": "Показатель4",
          },
          "a1": {
            "title": "Показатель3",
          },
          "val2": {
            "title": "Показатель2",
          },
        },
      },
      "xAxis": "name;lng;lat",
      "yAxis": "measures",
      "filters": {
        lat: [
          ">",
          60,],

      },
      "measures": [
        "sum(val):v",
        "sum(val2)",
        "concat("Бонусная-", sum(val2)):a1",],

      "dimensions": [
        "name",
        "lng",
        "lat",
        "concat("Координаты", lng, "x", lat):aa",],

    },
    "view_class": "mapdots",
  },
  {
    "title": "Области",
    "display": {
      "stoplight": "XL_RED_YELLOW_GREEN",
    },
    "dataSource": {
      "koob": "testMAP.cub",
      "style": {
        "measures": {
          "v": {
            "title": "Показатель3",
          },
          "val2": {
            "title": "Показатель4",
          },
        },
```

```
      },
      "xAxis": "name;lng;lat;region_id",
      "yAxis": "measures",
      "measures": [
        "sum(val):v",
        "sum(val2)",],

      "dimensions": [
        "name",
        "lng",
        "lat",
        "region_id",],

      "pathToMapApi": "glossary.russia_region_borders",
    },
    "view_class": "mapareas",
  },],

  "view_class": "map",
  "title": "Заголовок карты",
}
```

Секции `frame` и `title` работают аналогично вышеописанным дэшам (отображение дэша на дэшборде и указание заголовка дэша соотвественно). `view_class` необходимо указать `'map'`. Далее указывается массив `children`, в котором каждый элемент массива является отдельным слоем с возможностью скрытия его на карте. Внутри каждого элемента массива `children` используются стандарные опции конфигурации дэшей. В примере выше в массиве `children` указано два элемента: для отображения точек (`'mapdots'`) и для окраски областей карты (`'mapareas'`).

В массиве `dimensions` необходимо указать минимум три размерности:

1. `name` - Название локации
2. `lng` - долгота
3. `lat` - широта

И отложить их на оси X:

```
"xAxis": 'name;lng;lat',
```

Вот время как рассматриваемые факты откладываются по оси Y:

```
"yAxis": 'measures',
```

Стилизация слоев работает по правилам, аналогичным другим дэшам. В примере выше представлены варианты стилизации (указание заголовков фактам, `stoplight` и т.д.)

`title` указанный для объекта `children` (слоя) указывает название кнопки на карте, демонстрирующий/скрывающий слой на карте

Для `view_class: 'mapareas'` также предусмотрено поле `pathToMapApi` - указывающий путь для выделения областей и их координат (wkt). В случае, если `pathToMapApi` не указан, то по умолчанию используется таблица `'glossary.russia_region_borders'` из базы данных LuxmsBI с картой Российской Федерации, поделенной на регионы и федеральные округа.

В случае необходимости создания собственной таблицы для областей на карте, в таблицы должны присутствовать следующие столбцы: 1. type - в зависимости от уровня разбиения карты (доступные значения в таблице **glossary.russia_region_borders**: region/fdistrict) 2. id - уникальный номер области 3. title - название области 4. wkt - координаты области в формате wkt

В случае использования типов отображения “Точки” и “Графики” в кубе с данными должны быть прописаны также: - ширина - долгота - Название показателя на карте (например название городов, стран, областей и т.д.)

Пример полученного отображения на карте представлен выше:

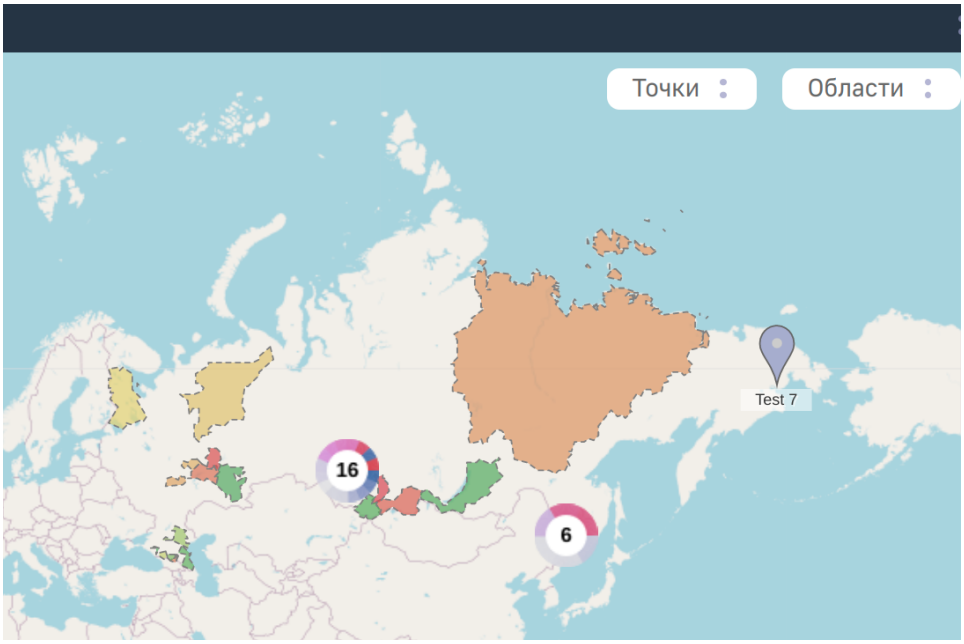


Рис. 2.36 **Сконфигурированный дэш “Карта”**

В случае необходимости скрытия слоя по умолчанию в `display` необходимо указать `defaultActive` со значением `false`.

Для карты присутсвует опция `HoverTooltip` - при нажатии на элемент карты (точку или область) выведется информация со значениями фактов по данному элементу.

В случае указания onClickDataPoint для слоя, опция `HoverTooltip` будет отображать информацию при наведении на элемент

Функции из блока `display` также работают в карте, например: стилизация тултипа, указания формата выводимых данных, указания зон (`stoplights`) и т.д.

Для указания размера модального окна для слоев “Графики” можно использовать `displayDrilldown` в блоке `display`.

Пример:

```
displayDrilldown: {
top: 'calc(50% - 150px)',
left: 'calc(50% - 200px)',
width: '400px',
height: '300px',
},
```

где, **top** - ориентация модального окна по вертикали, **left** - ориентация модального окна по горизонтали, **width** - ширина модального окна, **height** - высота модального окна

Указывать значения можно в условных единицах, пикселах, процентах, с использованием функции **calc** и т.д.

2.8 Дэш axes-selector и работа с его конфигурационным файлом

В системе LuxmsBI существует возможность отображения одного дэша с различным набором данных. Для этого можно использовать `axes-selector`.

Рассмотрим функционал данного дэша на примере дэша “Донат”.

Рис. 2.37 **Дэш “Донат”**

Для добавления возможности переключения необходимо в поле `view_class` указать `axes-selector/` + название визеля для отображения (в нашем случае `axes-selector/bublik`). После указания `axes-selector` дэш визуально не помелся. Для добавления выпадающего списка на дэше необходимо в конфигурационном файле указать массивы `xAxes/yAxes`, в

которых перечисляется список для отображения переключения данных, откладываемых по оси X и Y соответственно. После добавления массивов `xAxes/yAxes` и сохранения изменений, дэш выглядит следующим образом.

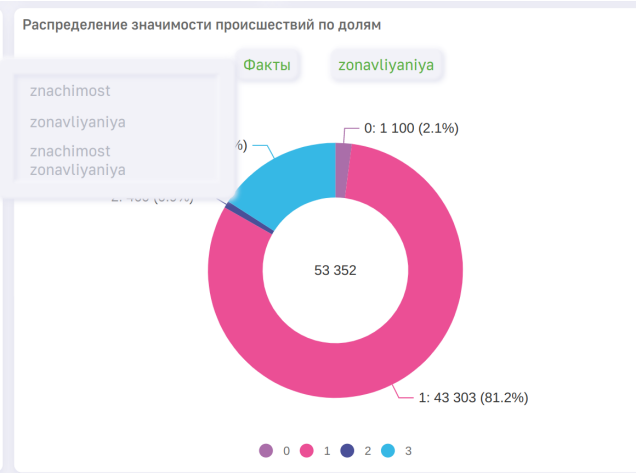


Рис. 2.38 **Дэш “Донат” с возможностью смены фактов/размерностей для отображения**

Конфигурационный файл данного дэша представлен ниже:

```
{
  "frame": {
    "h": 3,
    "w": 5,
    "x": 8,
    "y": 1,
  },
  "options": [
    "DisplayAllBadges",],

  "dataSource": {
    "koob": "luxmsbi.adm_confi",
    "xAxes": [
      "znachimost",
      "zonavliyaniya",
      "znachimost;zonavliyaniya",],

    "xAxis": "measures",
    "yAxes": [
      "znachimost",
      "zonavliyaniya",
      "znachimost;zonavliyaniya",],

    "yAxis": "znachimost",
    "measures": [
      "count(kol):count_kol",],

    "dimensions": [
      "znachimost",
      "zonavliyaniya",],
```

```
  },
  "view_class": "axes-selector/bublik",
  "title": "Распределение значимости происшествий по долям",
}
```

2.9 Конфигурация управляющего дэша

Для управляющего дэша большинство полей конфигурируется аналогичным образом, как и остальных типов визелей (frame, dataSource, view_class, title) за исключением отсутствия для него массива measures. По-умолчанию в управляющем дэше строковые размерности отображены как checkbox'ы,

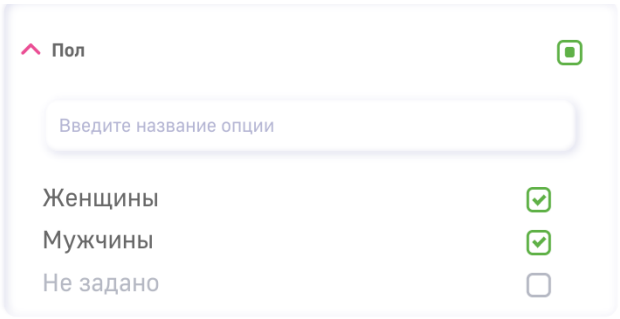


Рис. 2.39 **Размерность с типом "строка" в управляющем дэше**

размерности типа "период" отображены календарем с выбором диапазона дат для отображения,

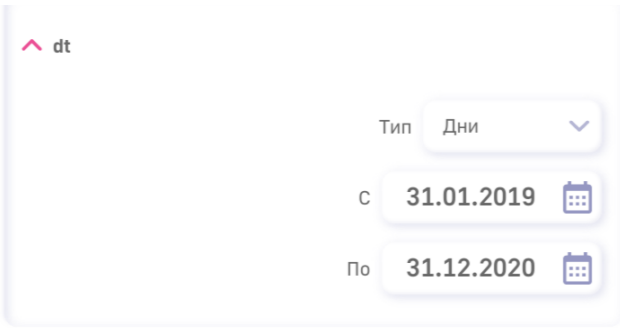


Рис. 2.40 **Размерность с типом "период" в управляющем дэше**

для численных размерностей используется ползунок, для выбора численного диапазона для отображения.

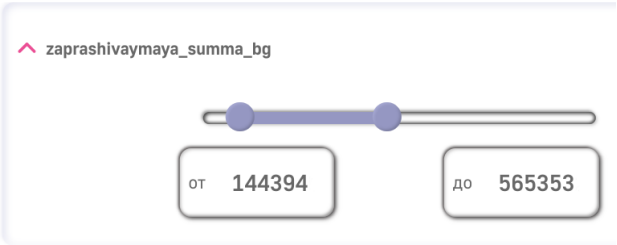


Рис. 2.41 **Размерность с типом "число" в управляющем дэше**

В случае необходимости указания одного показателя размерности, в поле style в для необходимой размерности указать `type: 'radio'` для представления данной размерности в управляющем дэше в стиле **radiobutton** (возможен выбор только одного показателя)

Для указания конкретной работы радиобаттонов необходимо в конфигурации куба указать defaultValue с конкретным показателем

Также существует возможность стилизации шапки и тела управляющего дэша. Для указания цвета фона управляющего дэша необходимо указать ключ `bgColor` со значением HEX цвета. Для стилизации шапки управляющего дэша в объекте `display` необходимо указать поле `headerStyle`, в котором можно прописать следующие поля:

- color - указание цвета заголовка
- backgroundColor - указание цвета фона шапки
- textAlign - ориентация заголовка в шапке (допустимые значения: left, right, center)

Также для размерности можно указать опцию `PanelOpened` - опция, позволяющая раскрыть список показателей для фильтрации по умолчанию.

Для реализации каскадных фильтров в конфигурации управляющего дэша необходимо прописать массив `hierarchy` с указанием иерархии.

Для размерностей типа “период” также существуют дополнительные опции, прописываемые в блоке `style`:

- `types` - массив с перечислением доступных типов периода: день(day), неделя(week), месяц(month), квартал(quarter), год(year).
- single - выбор одного периода, вместо диапазона (по умолчанию - false)
- defaultType - включенный по умолчанию тип периода

Пример конфигурации управляющего дэша представлен ниже:

```
{
  "frame": {
    "h": 5,
    "w": 4,
    "x": 7,
    "y": 0,
  },
  "display": {
    "headerStyle": {
      "color": 'blue',
      "textAlign": 'center',
      "backgroundColor": 'red',
    },
  },
  "bgColor": 'blue',
  "dataSource": {
    "koob": 'luxmsbi.public_maxexample',
    "style": {
      "sex": {
```

```
      "type": 'radio',
      "options": [
        "PanelOpened"]

    },
    "dt":{
     "types": [
       'month',],

      "single": true,
      "defaultType": 'month',
    }
   },
"hierarchy": [
  'region=>gorod=>address',    ],

   "xAxis": 'sex;dt',
   "yAxis": 'measures',
   "dimensions": [
     'sex',
     'dt'],

  },
  "view_class": 'VizelKoobControl',
  "title": '',
}
```

2.10 Конфигурация дэша what-if

Если необходимо отображать список переменных для анализа на нескольких дэшах, необходимо использовать массив `vars`.

```
vars:[
 'a1',
'a2',
....]
```

указанные в данном массиве переменные будут отображены в дэше.

3 Использование LPE-выражений в виджетах

Существуют случаи, когда нам необходимо в конфигурационном файле использовать не конкретное значение, а функции, выполняющие условие / действие, в зависимости от приходящих значений или действий пользователя. Для этого необходимо использовать LPE-выражения. Ниже представлен пример-псевдокод использования LPE-выражений:

```
поле_конфигурационного_файла: "lpe:название_функции(аргумент1, аргумент2, ...)"
```

В случае, когда необходимо использовать несколько функций одновременно, существует возможность указания нескольких функций внутри одного LPE-выражения:

```
поле_конфигурационного_файла: "lpe:название_функции1(аргумент1, аргумент2, ...);
  название_функции2(аргумент1, аргумент2, ...)..."
```

Более конкретные примеры использования данных выражений представлено ниже.

3.1 Использование LPE-выражений для стилизации дэшей

В случае, когда нам необходимо стилизовать дэши в зависимости от приходящих значений, существует возможность использования условных LPE-выражений. Для использования условных выражений используются следующие функции:

1. `when` - используется для неограниченного количества условий в выражении. Пример:

```
"lpe:when(lpe(public_mv_doc_base1_amount_discount > 0),'#00b9ac',  '#93CAFE')"
```

В данном примере представлен случай, когда LPE-выражение представлено внутри LPE-выражения

2. `if` - используется для одного условного выражения. Необходимые для указания аргументы:
   2.1 условие, 2.2 значение параметра, если условие истинно 2.3 значение параметра, если условие ложно

Ниже представлен пример использования условных выражений с оператором `if`:

```
fontSize: "lpe:if(value<10, '120%', '80%')"
```

В данном примере, в случае, если значение будет меньше 10, то оно отобразится на 20% больше стандартного размере, в противном случае, на 20% меньше.

Данные условные выражения можно использовать для различных полей стилизации. Ниже представлен файл конфигурации дэша “Данные” с использованием условных LPE-выражений:

```
{
  frame: {
    h: 4,
    w: 7,
    x: 0,
    y: 0,
  },
  dataSource: {
    koob: "luxmsbi.public_mv_doc_base1",
    style: {
      measures: {
        ttt: {
          color: "lpe:when(lpe(public_mv_doc_base1_amount_discount > 0),'#ff0000', '#00ff00')",
          title: "Индктр",
        },
        public_mv_doc_base1_azs_name: {
          backgroundColor: "lpe:when(lpe(public_mv_doc_base1_amount_discount > 0),'#00b9ac',  '#93CAFE')",
        },
        public_mv_doc_base1_add_info_quantity: {
          fontSize: "lpe:if(value<10, '120%', '80%')",
          minWidth: 300,
          fontStyle: "lpe:if(value<10, 'italic', '')",
          fontWeight: "lpe:if(value>40, 'bold', 'normal')",
          textDecoration: "lpe:if(value>40, 'underline', '80%')",
          backgroundColor: "lpe:if(value = 2, '#ffaabb', 'transparent')",
        },
      },
    },
    yAxis: "measures;",
    sortBy: "public_mv_doc_base1_by_day_s,public_mv_doc_base1_trans_time",
    filters: {},
    measures: [
      "concat('Бонусная-', public_mv_doc_base1_card_type):a1",
      "if(public_mv_doc_base1_rc='P', 'Платиновая', if(public_mv_doc_base1_rc='J','Золотая',if(public_mv_doc_base1_rc='R','Серебренная','Отсутствует'))):a10",
      "if(public_mv_doc_base1_amount_discount > 0,  '↑','↓'):ttt",
      "public_mv_doc_base1_add_info_quantity",
      "public_mv_doc_base1_amount_discount",],

    dimensions: [
      "public_mv_doc_base1_goods_code_type",],

  view_class: "koob-table-simple",
}
```

Ниже на изображении представлен дэш с вышеописанным конфигурационным файлом:

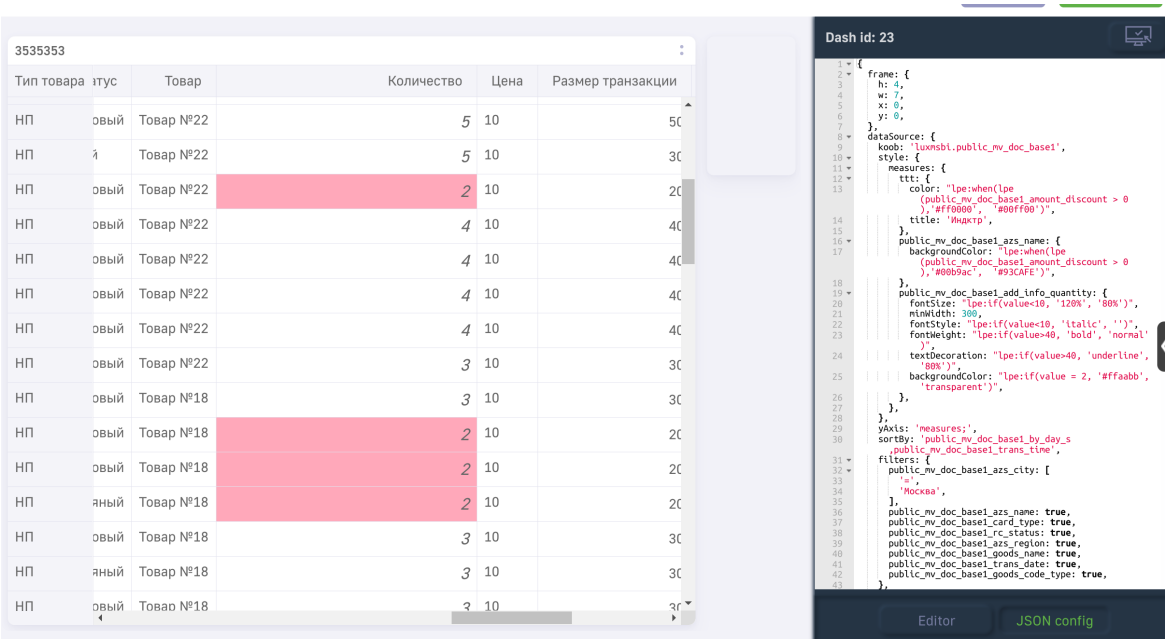


Рис. 3.1 **Дэш “Данные” со стилизацией столбцов с использованием LPE-выражений**

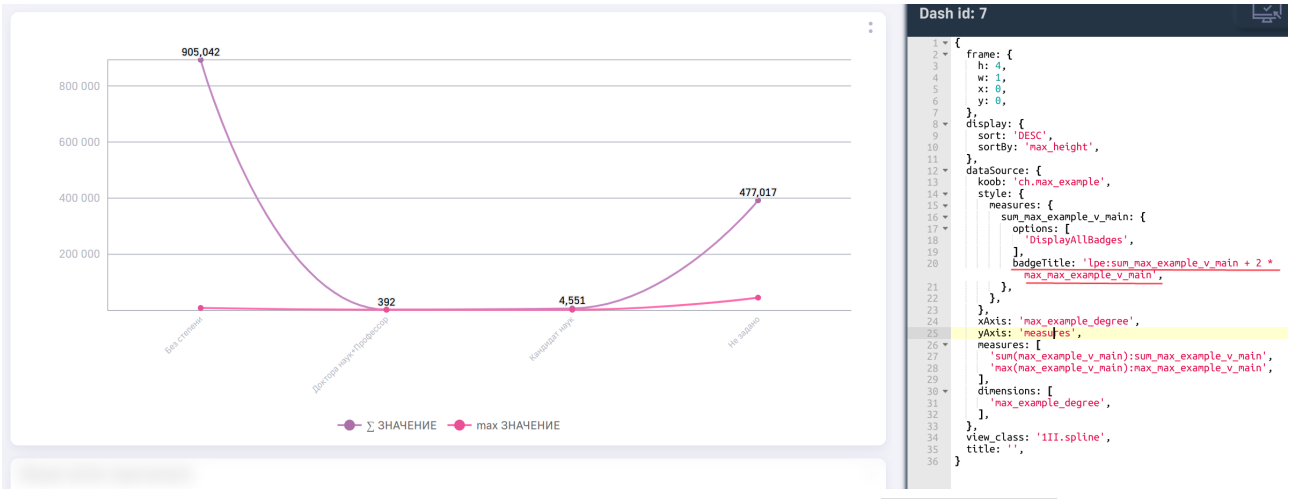


Рис. 3.2 **Использование LPE-выражения в поле `badgeTitle`**

На примере выше, в выводимых значениях предомострировано значение, рассчитанное по формуле: `1_факта + 2 * 2_факт`

3.1.1 Функции, доступные внутри выражений if и when

Внутри выражений вы можете использовать:

- математические операнды `+`, `-`, `*`, `/`, `(`, `)`;
- операторы сравнения `<`, `>`, `<=`, `>=`, `=`, `!=`;
- Значения `true`, `false`, `#t`(true), `#f`(false), `null`, `NIL`(null);
- Логические операторы `and`, `or`, &&(and), `||`(or), `!`
- Функцию `cond()`; Пример:

```
color: 'lpe:when(cond(1>null), "#ff0000", "#00ff00")'
```

- специальную переменную `value` чтобы изменить параметры только некоторых значений. Пример:

```
color: 'lpe:if(value>989, "#00f0f0", "#0000ff")'
```

Результат такой настройки

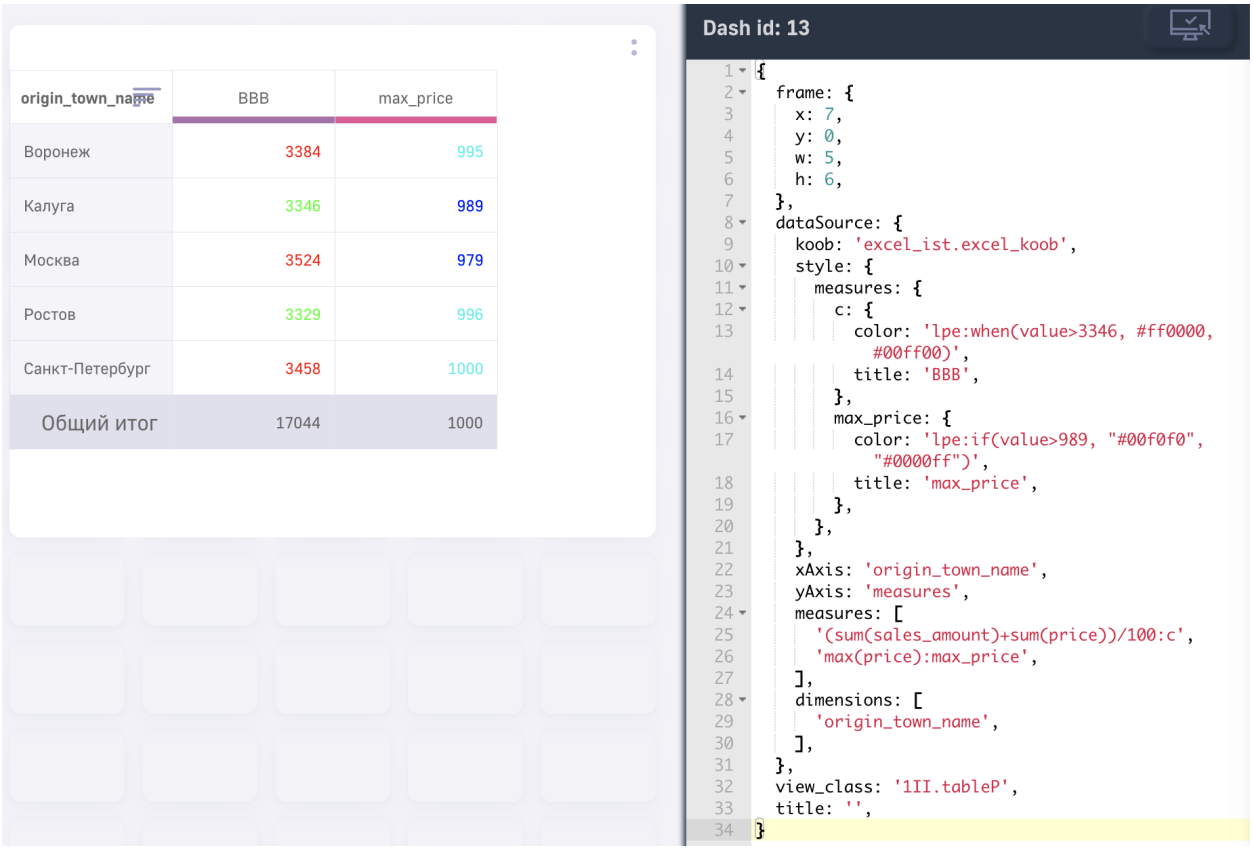


Рис. 3.3 **использование `value` в лпе в цвете дэша**

3.2 Использование LPE-выражений для вычислений

Для вычислений на лету можно использовать LPE-выражения прямо в конфигурационном поле виджета. Пример:

```
measures: [
"(sum(sales_amount)+sum(price))/100:c",
"max(price):max_price",    ],
```

При добавлении своей формулы, необходимо после нее через `:` указать название для этого факта.

Помимо алгебраических выражений можно также использовать условные конструкции:

```
measures: [
"if(avg(price)>558, 5, 'много'):c",      ],
```

Такая конфигурация заполнит столбец таблицы значениями по условию. Можно использовать как числовые значение, так и строковые.

3.2.1 Список специальных агрегационных функций.

В LPE выражениях поддерживаются все аггрегационные функции которые возможны в вашей базе. Список поддерживаемых баз:

- Clickhouse;
- Greenplum;
- MS SQL;
- Oracle > 9;
- PostgreSQL;
- SAP HANA, поддержка добавлена в версии `luxmsbi-pg` v8.10.18. Не поддерживается функция `mode()`.
- Teradata;
- Vertica, поддержка добавлена в версии `luxmsbi-pg` v8.10.9. Не поддерживаются функции `mode()` и `median()`.

Название функции	Имя функции
статистическая дисперсия	var_pop
выборочная дисперсия	var_samp
выборочное отклонение	stddev_samp
статистическое отклонение	stddev_pop
мода	mode
медиана	median

3.2.1.1 Синтаксис агрегационных функций

Все функции и операторы в LPE записываются следующим образом:

```
<имя функции>(<параметры функции>)
```

Несколько примеров:

1. `'count(distinct(town_name))'`
2. `'sum(price)'`
3. `"if(avg(price) >= 10000, 'Больше 10 000', 'Меньше 10 000')"`