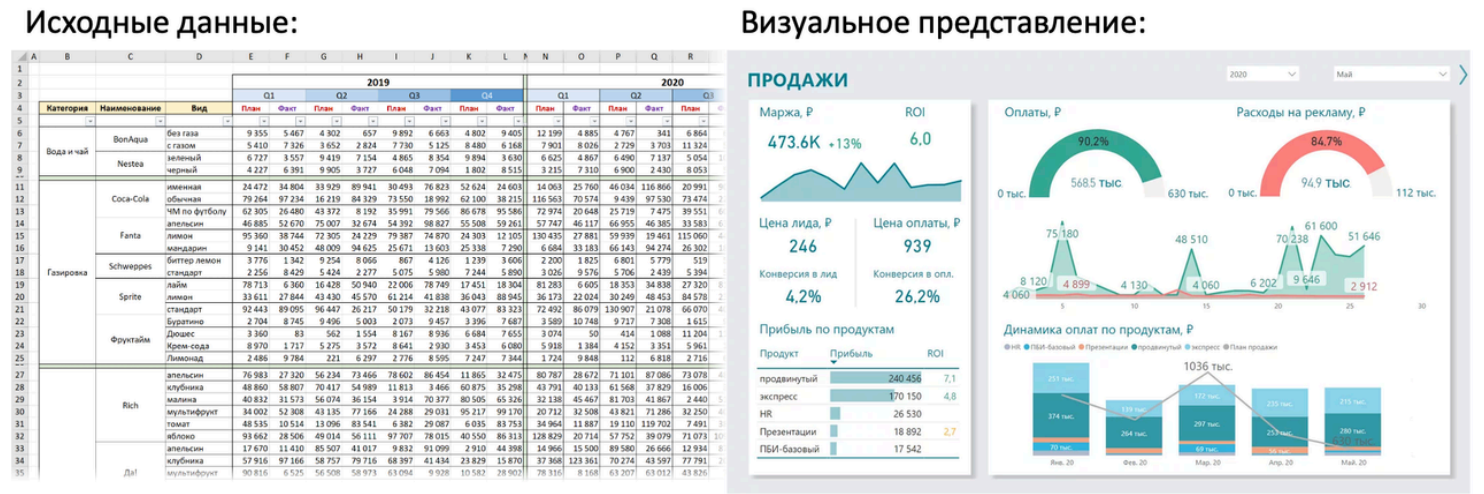


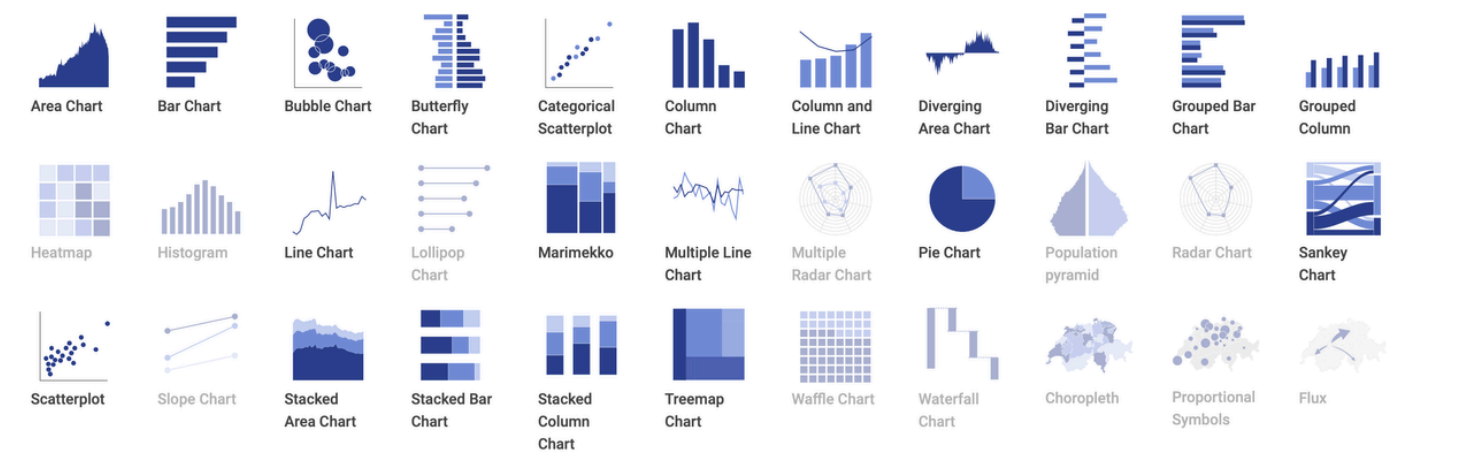
# Инструменты визуализации: виды графиков и их применение.

**Визуализация данных** – это процесс использования визуальных элементов, таких как диаграммы, графики или карты, для представления данных. Он переводит сложные, масштабные или числовые данные в визуальное представление, которое легче обрабатывать. Инструменты визуализации данных улучшают и автоматизируют процесс визуальной передачи данных для обеспечения точности и детализации. Ниже представлен пример как исходные данные могут выглядеть и какую форму приобретать при использовании средств визуализации:



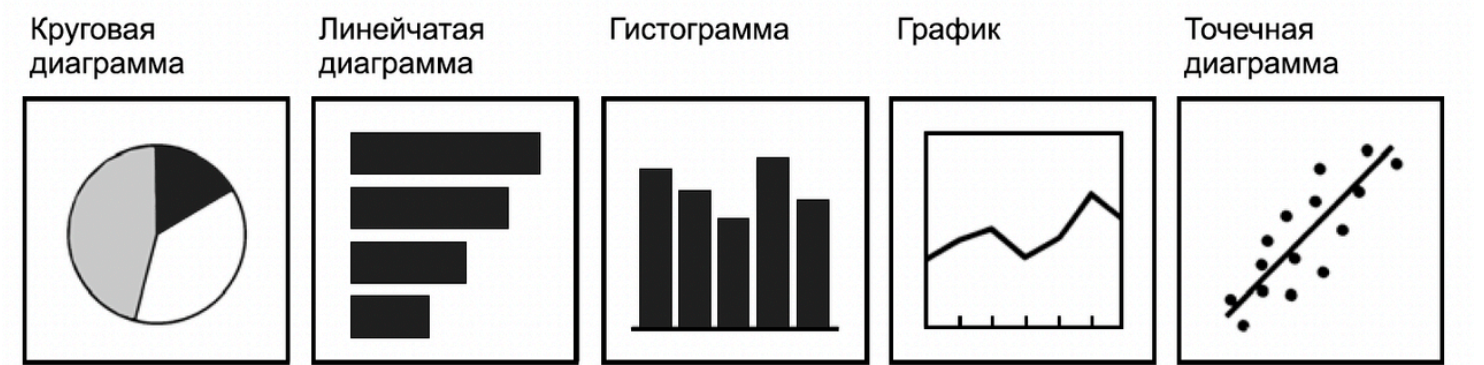
(<https://postimg.cc/WtNgMQj1>)

На данный момент существует огромное количество видов диаграмм:



(<https://postimg.cc/8fzZ9c6y>)

Но все они сводятся к 5-ти видам:




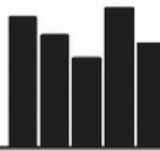
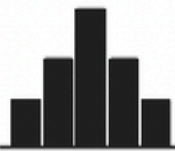
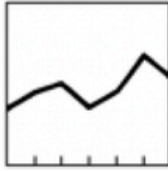
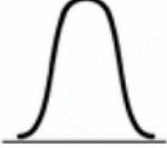



(<https://postimg.cc/SYfgt8RY>)

Визуализация данных помогает обращать внимание на особенности в данных, например, на графике акций это может быть взлет/спад цены, на круговой диаграмме - доля самого продаваемого товара и т.д. При выборе метода графика для визуализации важно учитывать, что неграмотно презентованный материал может исказить представление о реальности и ввести в заблуждение, что повлечет за собой неверные выводы и принятые решения (например, спад цены на акции будет трудно увидеть на круговой диаграмме). Поэтому для выражения аспектов данных используют пять основных типов сравнения:

- 1. Покомпонентный,
- 2. Позиционный,
- 3. Временной,
- 4. Частотный,
- 5. Корреляционный.

Также к каждому типу сравнения рекомендуют применять определенный вид визуализации:

ТИПЫ СРАВНЕНИЯ					
ОСНОВНЫЕ ТИПЫ ДИАГРАММ	ПОКОМПОНЕНТНОЕ	ПОЗИЦИОННОЕ	ВРЕМЕННОЕ	ЧАСТОТНОЕ	КОРРЕЛЯЦИОННОЕ
					
					
					
					
					

(<https://postimg.cc/qzLk25J4>)

Рассмотрим каждый тип на примерах:

При **покомпонентном сравнении** мы прежде всего показываем размер каждого компонента в процентах от некоего целого. Например:

- В ноябре продажи ёлочных игрушек составили наибольшую долю в общем объеме продаж магазина.
- Доля рынка яблок составляет менее 10% рынка фруктов.
- Почти половина студентов посетили занятие.

Ключевые слова для покомпонентного сравнения: «доля», «проценты от целого», «составило X%».

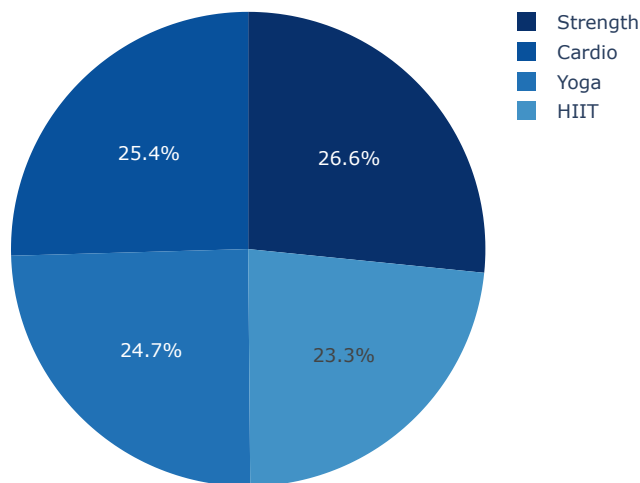
```
In [1]: import pandas as pd #Импортируем библиотеку для работы с таблицами
df = pd.read_csv('gym_members_exercise_tracking.csv') #Загружаем датасет с данными о посещениях спор
#зала
# Описание датасета: https://www.kaggle.com/datasets/valakhorasani/gym-members-exercise-dataset
df.head() #Выводим первые строки датасета для проверки
```

Out[1]:

	Age	Gender	Weight (kg)	Height (m)	Max_BPM	Avg_BPM	Resting_BPM	Session_Duration (hours)	Calories_Burned	Workout_Type	Fat_Percentage
0	56	Male	88.3	1.71	180	157	60	1.69	1313.0	Yoga	12.6
1	46	Female	74.9	1.53	179	151	66	1.30	883.0	HIIT	33.9
2	32	Female	68.1	1.66	167	122	54	1.11	677.0	Cardio	33.4
3	25	Male	53.2	1.70	190	164	56	0.59	532.0	Strength	28.8
4	38	Male	46.1	1.79	188	158	68	0.64	556.0	Strength	29.2

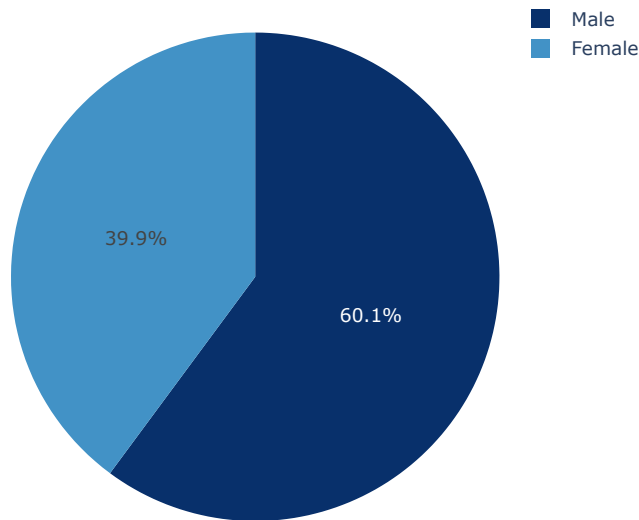
```
In [2]: import plotly.express as px # Для построения визуализации будем использовать библиотеку plotly
# Она помогает строить интерактивные графики и обладает большим спектром настроек
# Построим пример графика покомпонентного сравнения – круговую диаграмму
fig = px.pie(df, values='Session_Duration (hours)', names='Workout_Type',
             width=500, height=500, title = 'Доля затраченного времени на тренировку по типам',
             color_discrete_sequence=px.colors.sequential.Blues[:-1])
# fig – объект создаваемый библиотекой с информацией о графике, через него осуществляется более гиб
кая настройка и отображение
# Используем функцию px.pie, в которую передаем наименование датафрейма, значения для подсчета сумм
ы и разбиения по категориям
# Параметры width и height регулируют размер графика, title – отображает наименование
# color_discrete_sequence – выбирает цветовую палитру
fig.show() # Отображение диаграммы
# Например, по данному графику можно сделать вывод, что занятия йогой составляют 25% от общего числ
а занятий
```

Доля затраченного времени на тренировку по типам



```
In [3]: # Построим график со сравнением потребляемой воды во время тренировки по полу (мужчина/женщина)
fig = px.pie(df, values='Water_Intake (liters)', names='Gender',
             color_discrete_sequence=px.colors.sequential.Blues[::-3])
fig.update_layout( width=500, height=500, title = 'Доля потребления воды по полу')
# Параметр update_layout также позволяет изменять свойства графиков
fig.show()
# По данной диаграмме можем сделать вывод, что мужчины во время тренировки потребляют на 20 п.п. (процентных пунктов) больше воды, чем женщины
```

Доля потребления воды по полу



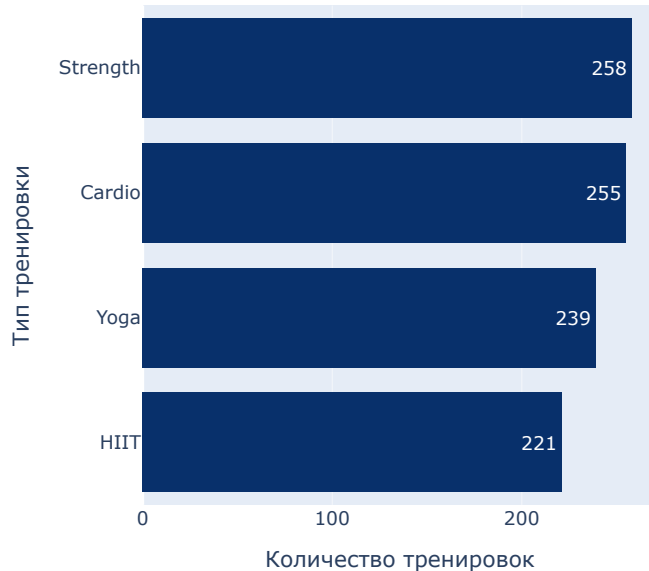
При **позиционном сравнении** мы выявляем, как объекты соотносятся друг с другом: одинаковы ли они, больше или меньше других. Например:

- В декабре продажи ёлок превысили продажи фикусов и орхидей.
- Выручка клиента от продаж находится на четвертом месте.
- Посещение занятий в шести группах примерно одинакова.

Ключевыми словами для позиционного сравнения являются следующие: «больше чем», «меньше чем», «равно ».

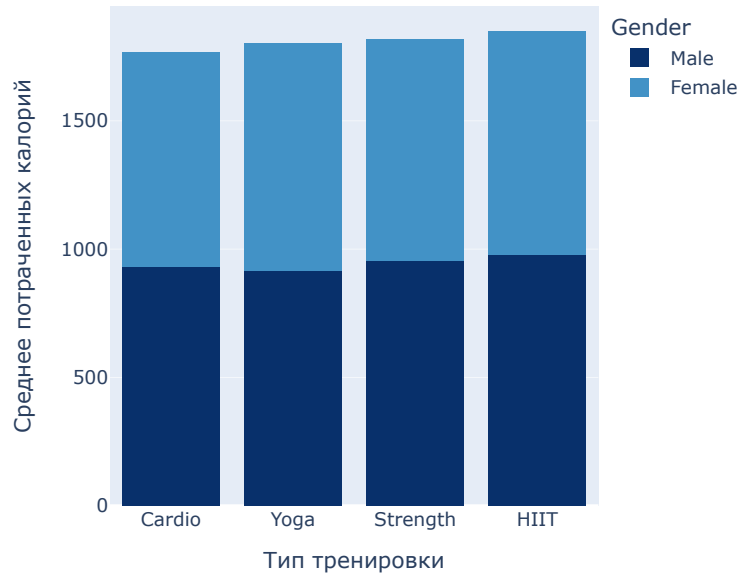
```
In [4]: fig = px.histogram(df, y="Workout_Type",
                           text_auto=True, width=500,height=500, title = 'Количество тренировок по типам',
                           color_discrete_sequence=px.colors.sequential.Blues[::-1])
# Для построения гистограмм используем функцию px.histogram, y= – помогает из привычной диаграммы с
# делать линейчатую,
# что упрощает восприятие при просмотре отсортированных столбцов
# Параметр text_auto=True добавляет подписи к столбцам
fig.update_yaxes(categoryorder='total ascending', title = 'Тип тренировки')
fig.update_xaxes(title = 'Количество тренировок')
# Методы update_yaxes и update_xaxes настраивают отображение осей x и y
# параметр categoryorder сортирует столбцы
fig.show()
# По данному графику можем сделать вывод, что растяжкой занимаются чаще всего, но категории имеют п
# риблизительно одинаковое количество занятий
```

Количество тренировок по типам



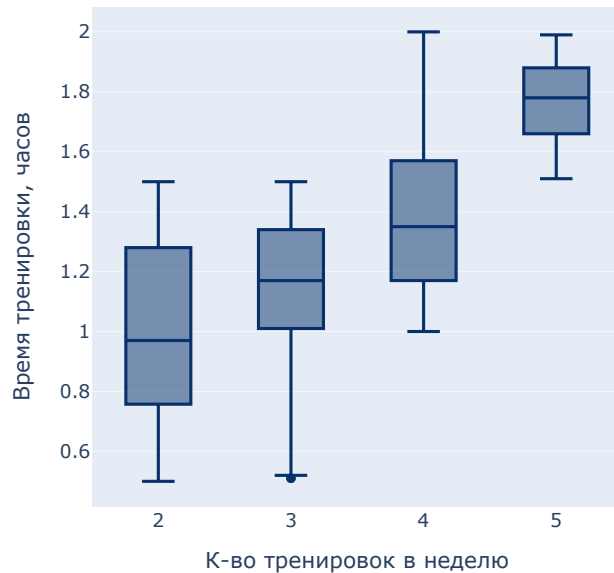
```
In [5]: fig = px.histogram(df, x="Workout_Type", y="Calories_Burned", color="Gender",
                          width=500, height=500, title="Среднее распределение калорий по типу и полу" ,
                          histfunc='avg',
                          color_discrete_sequence=px.colors.sequential.Blues[::-3])
# Также в px.histogram можно разбивать данные по определенному признаку при помощи параметра color=
# Параметр histfunc меняет агрегирующую функцию для столбцов (по умолчанию стоит функция подсчета
# количества)
fig.update_xaxes(categoryorder='total_ascending', title = 'Тип тренировки')
fig.update_yaxes(title = 'Среднее потраченных калорий')
fig.show()
# По данному графику заметим, что среднее значение сожженных калорий выше всего у интервальных трени
# ровок (при этом этим видом спорта реже всего занимаются)
```

## Среднее распределение калорий по типу и полу



```
In [6]: # Также группы можно сравнивать при помощи графика ящика с усами, который учитывает не только количество в группе, но и распределение внутри
fig = px.box(df, x="Workout_Frequency (days/week)", y="Session_Duration (hours)",
             width=500, height=500, title= 'Распределение времени тренировок по к-ву сессий',
             color_discrete_sequence=px.colors.sequential.Blues[::3])
# Для построения используется функция px.box, x= задает распределения на группы
fig.update_xaxes(title = 'К-во тренировок в неделю')
fig.update_yaxes(title = 'Время тренировки, часов')
fig.show()
# По графику можем заметить, что чем чаще посетитель занимается спортом, тем больше его среднее время тренировки
```

Распределение времени тренировок по к-ву сессий



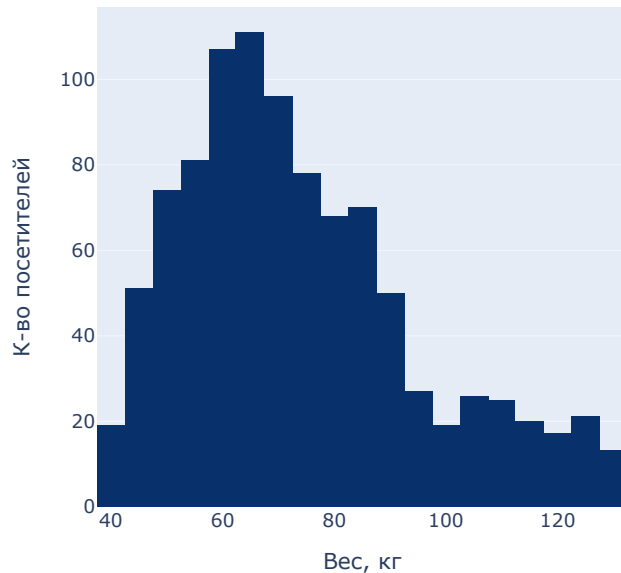
**Частотный вид сравнения** помогает определить, сколько объектов попадает в определенные последовательные области числовых значений. Например, частотное сравнение используется для того, чтобы показать, сколько работников зарабатывает менее чем 30 тыс. руб., сколько — 30—60 тыс. руб. и т. д.; сколько жителей относится к возрастной группе до 10 лет, сколько — от 10 до 20, от 20 до 30 и т. д. Примеры типичных формулировок такого вида:

- В ноябре снег в основном выпадал от 5 до 10 дней.
- Выполнение курсовой работы у студентов в большинстве случаев занимает от 10 до 15 дней.
- У жителей Москвы в основном есть 1-2 домашних питомца.

Термины, характерные для этого вида сравнения: «в диапазоне от  $x$  до  $y$ », «концентрация», «частотность» и «распределение».

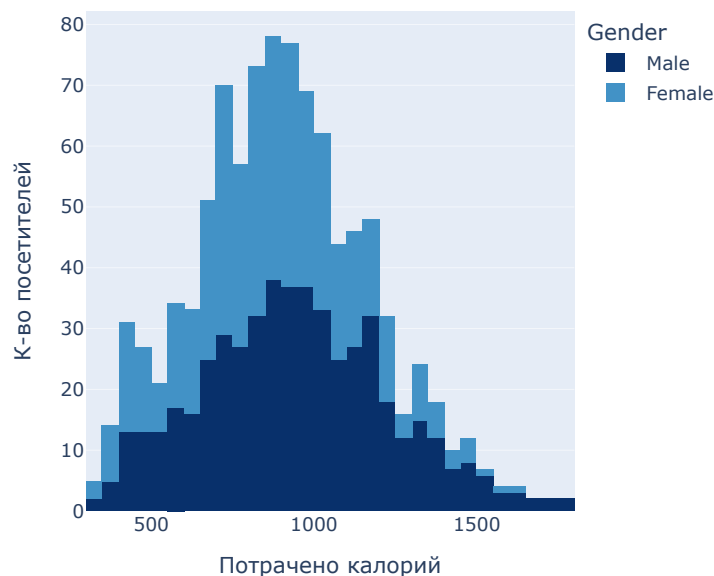
```
In [7]: fig = px.histogram(df, x="Weight (kg)", nbins=30,
                        width=500, height=500, title = 'Вес посетителей спортзала',
                        color_discrete_sequence=px.colors.sequential.Blues[::-1])
# В данном примере в функции px.histogram используем числовой тип данных, nbins изменяет к-во разбиений столбцов
fig.update_xaxes(title = 'Вес, кг')
fig.update_yaxes(title = 'К-во посетителей')
fig.show()
# По данному графику можем сделать вывод, что в спортзал чаще всего приходят посетители с весом от 60 до 80 кг
```

Вес посетителей спортзала



```
In [8]: fig = px.histogram(df, x="Calories_Burned", color="Gender",
                        width=500, height=500, title = 'Распределение потраченных калорий по полу',
                        color_discrete_sequence=px.colors.sequential.Blues[::-3])
# При помощи параметра color можно показывать распределение по типам
fig.update_xaxes(title = 'Потрачено калорий')
fig.update_yaxes(title = 'К-во посетителей')
fig.show()
# По диаграмме можем сделать вывод, что мужчины и женщины тратят за тренировку в основном от 800 до 1000 калорий
```

Распределение потраченных калорий по полу





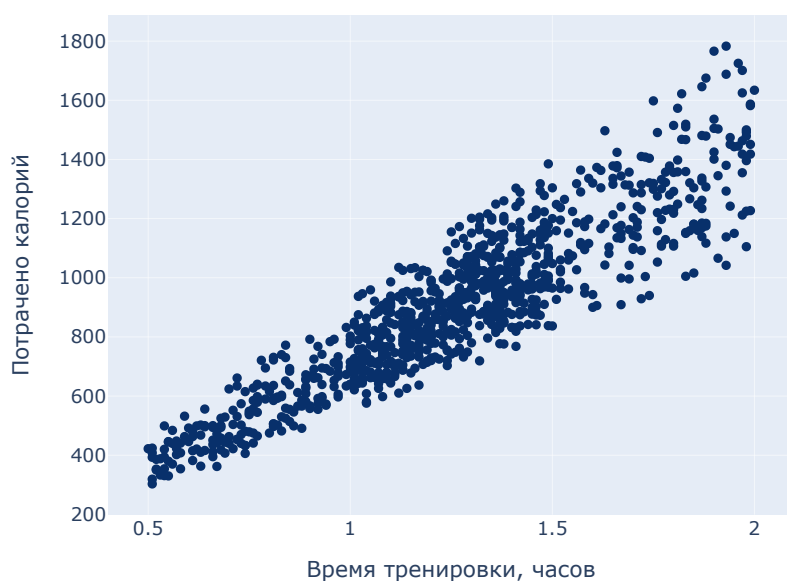
**Корреляционное сравнение** показывает наличие (или отсутствие) зависимости между двумя переменными. Например, обычно ожидается, что при увеличении объемов продаж возрастает прибыль или что при увеличении скидок возрастают объемы продаж. Например:

- Результаты продажи мороженого демонстрируют зависимость с температурой на улице
- Средний балл по предмету не зависит от к-ва машин во дворе
- Зарплата возрастает при увеличении опыта.

Ключевые слова для корреляционного типа сравнения: «относится к», «возрастает при (в случае)», «снижается при (в случае)», «меняется при (в случае)» или, наоборот, «не возрастает при (в случае)» и т. д.

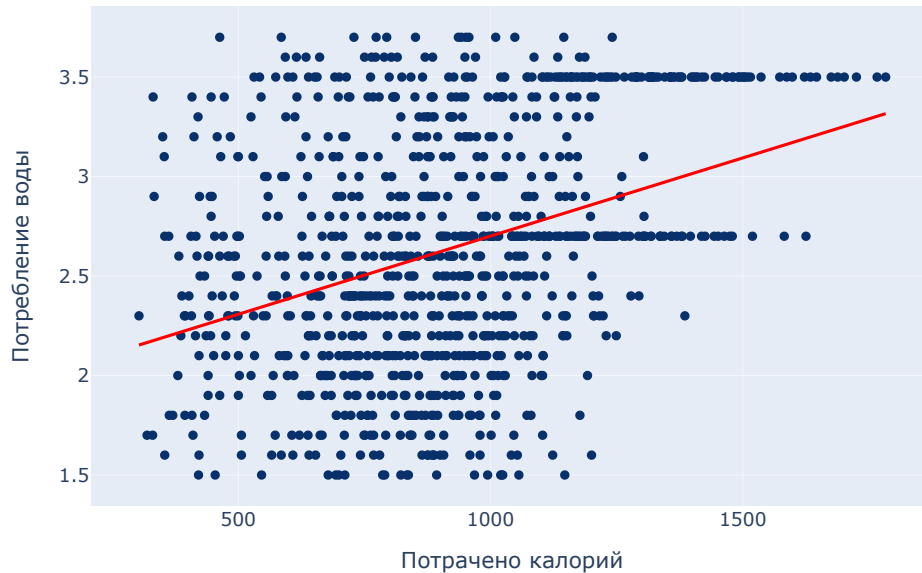
```
In [9]: fig = px.scatter(df, x='Session_Duration (hours)', y='Calories_Burned',
                        width=600, height=500, title = 'Зависимость потраченных калорий от времени тренировки',
                        color_discrete_sequence=px.colors.sequential.Blues[::-1])
# Для отображения зависимостей используем функцию px.scatter (диаграмма рассеяния)
fig.update_xaxes(title = 'Время тренировки, часов')
fig.update_yaxes(title = 'Потрачено калорий')
fig.show()
# По графику можем заметить явную связь между временем тренировки и количеством потраченных калорий
```

Зависимость потраченных калорий от времени тренировки



```
In [10]: # Для более ясного понимания вывода можно прописывать его в названии диаграммы
# Например, мы хотим показать отсутствие зависимости потраченных калорий и количеством выпитой воды
fig = px.scatter(df, x="Calories_Burned", y="Water_Intake (liters)",
                 width=700, height=500, title = 'Потраченные калории не зависят от количества потребляемой воды',
                 trendline="ols", trendline_color_override="red",
                 color_discrete_sequence=px.colors.sequential.Blues[::-1])
# Также для наглядности можно показать линию тренда параметром trendline="ols", trendline_color_override – задает цвет линии
fig.update_xaxes(title = 'Потрачено калорий')
fig.update_yaxes(title = 'Потребление воды')
fig.show()
```

Потраченные калории не зависят от количества потребляемой воды



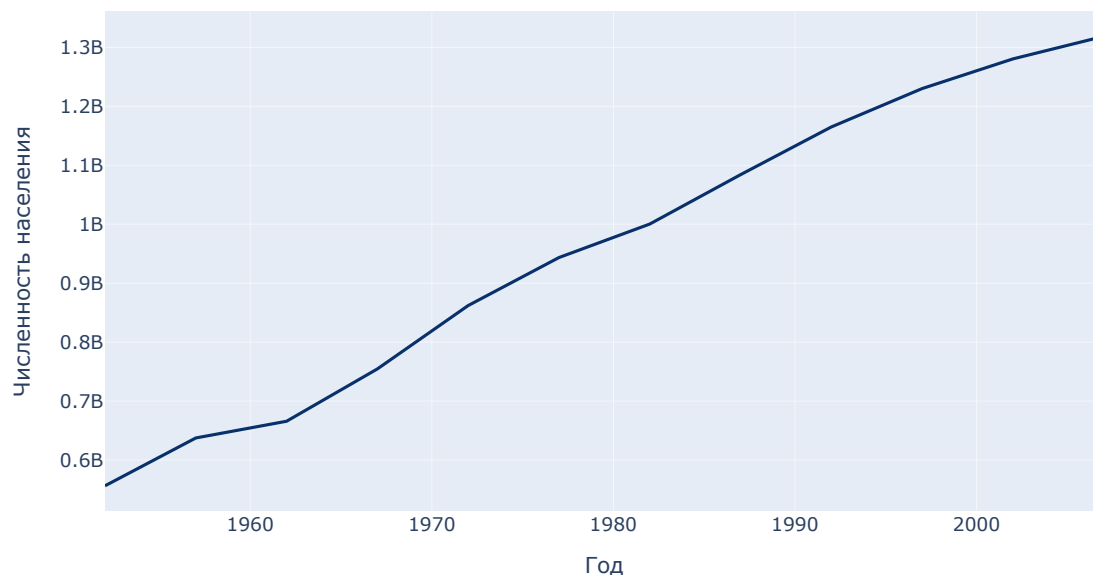
**При временном сравнении** нас интересует не размер каждой доли в сравнении с целым, не соотношение долей, а то, как они изменяются во времени — что происходит с определенными показателями на протяжении недель, месяцев, кварталов, лет: возрастают ли они, снижаются, колеблются или остаются неизменными. Например:

- Продажи в ноябре постоянно росли
- Цена акций за год резко упала
- Спрос на хлеб не изменился

Ключевые слова в данном случае: «изменяться», «расти», «убывать», «возрастать», «снижаться», «колебаться» и т. д.

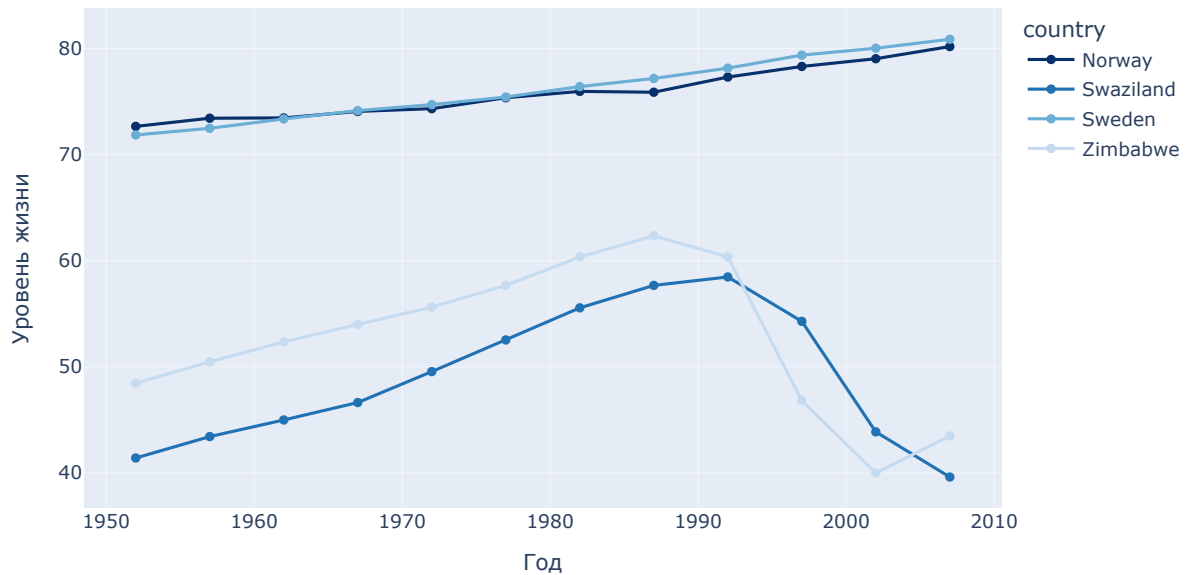
```
In [11]: df_country = px.data.gapminder() # Загружаем новый датафрейм о населении мира из библиотеки
fig = px.line(df_country[df_country['country']=='China'], x='year', y="pop",
              width=800, height=500, title = 'Рост численности населения в Китае',
              color_discrete_sequence=px.colors.sequential.Blues[::-1])
# График для временного сравнения строится при помощи функции px.line
# для построения отбираем численность населения Китая df_country[df_country['country']=='China']
fig.update_xaxes(title = 'Год')
fig.update_yaxes(title = 'Численность населения')
fig.show()
# Делаем вывод что численность Китая неуклонно растет с 1952 года
```

Рост численности населения в Китае



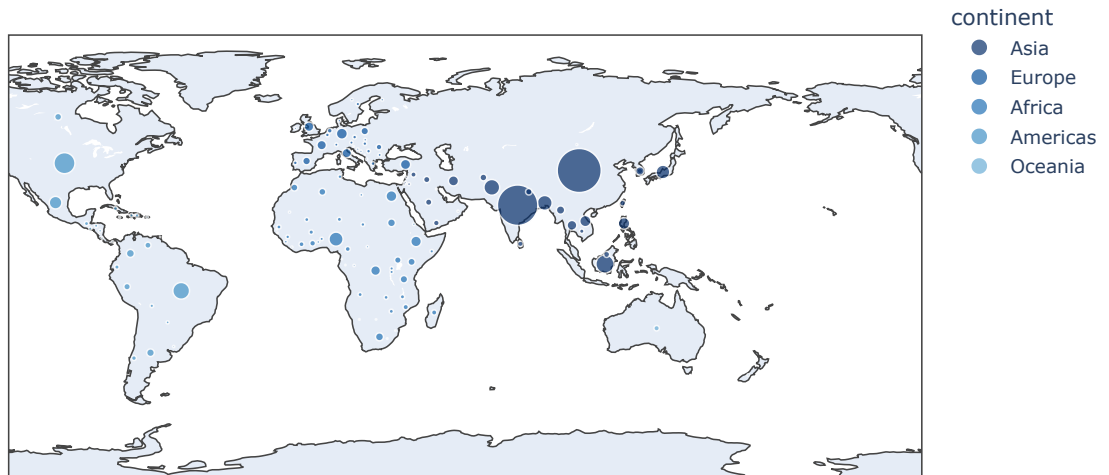
```
In [12]: fig = px.line(df_country[(df_country['country'].isin(['Zimbabwe','Swaziland','Norway','Sweden']))], x='year', y='lifeExp', color='country',
                    width=800,height=500, title = 'Изменение уровня жизни в скандинавских странах и странах Африки',
                    color_discrete_sequence=px.colors.sequential.Blues[::-2], markers=True)
# для отображения по нескольким категориям используем параметр color
# параметр markers добавляет маркеры на график
fig.update_xaxes(title = 'Год')
fig.update_yaxes(title = 'Уровень жизни')
fig.show()
# По данному графику можем сделать вывод, что уровень жизни в скандинавских странах растет, а в африканских начинает резко снижаться с 1990 года
```

Изменение уровня жизни в скандинавских странах и странах Африки



```
In [13]: # Также plotly позволяет строить интерактивные карты
fig = px.scatter_geo(df_country[df_country['year']==2007], locations="iso_alpha", color="continent", size="pop",
                    width=800,height=500, title = 'Численность населения мира в 2007 году',
                    projection="equiangular",
                    color_discrete_sequence=px.colors.sequential.Blues[::-1])
# В параметр locations передается код страны, size регулирует размер точек (в данном случае по численности населения)
# projection – вид карты
fig.show()
```

## Численность населения мира в 2007 году



## Задание 1

Самостоятельно изучить понятие "**дашборд**"

## Задание 2

Найти датафрейм на любую тему (например, на <https://www.kaggle.com> (<https://www.kaggle.com>)) и построить дашборд по данным (дашборд можно строить в Экселе, сервисах Datalens, Power BI или при помощи библиотеки plotly и метода plotly.subplots). Обосновать выбор визуализации и сделать по 2 вывода к каждой диаграмме.