**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ (ИУ5)

О Т Ч Е Т

**по лабораторной работе**

по дисциплине: Технологии машинного обучения

на тему: Разведочный анализ данных. Исследование и визуализация данных

\_

\_

\_

\_

\_

\_

\_

\_

Студент РТ5-61Б **А.C. Акушко**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель

**Ю.Е. Гапанюк**

(Подпись, дата) (И.О.Фамилия)

*2020 г.*

# Лабораторная работа №1

## Текстовое описание набора данных

В качестве набора данных мы будем использовать набор данных Diabets dataset <https://scikit-learn.org/stable/datasets/index.html#toy-datasets>Для каждого из n = 442 больных сахарным диабетом были получены десять исходных переменных, возраст, пол, индекс массы тела, среднее артериальное давление и шесть измерений сыворотки крови, а также интересующая нас реакция - количественная мера прогрессирования заболевания через год после исходного уровня.

In [7]:

**import numpy as np import pandas as pd import seaborn as sns**

**import matplotlib.pyplot as plt**

%**matplotlib** inline sns.set(style="ticks")

data = pd.read\_csv('data/diabetes.tab.txt', sep="**\t**")

# Основные характеристики датасета

In [9]:

*# Первые 5 строк датасета*

data.head()

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Out[9]: |  | | | | | | | | | | | |
|  |  | **AGE** | **SEX** | **BMI** | **BP** | **S1** | **S2** | **S3** | **S4** | **S5** | **S6** | **Y** |
|  | **0** | 59 | 2 | 32.1 | 101.0 | 157 | 93.2 | 38.0 | 4.0 | 4.8598 | 87 | 151 |
|  | **1** | 48 | 1 | 21.6 | 87.0 | 183 | 103.2 | 70.0 | 3.0 | 3.8918 | 69 | 75 |
|  | **2** | 72 | 2 | 30.5 | 93.0 | 156 | 93.6 | 41.0 | 4.0 | 4.6728 | 85 | 141 |
|  | **3** | 24 | 1 | 25.3 | 84.0 | 198 | 131.4 | 40.0 | 5.0 | 4.8903 | 89 | 206 |
|  | **4** | 50 | 1 | 23.0 | 101.0 | 192 | 125.4 | 52.0 | 4.0 | 4.2905 | 80 | 135 |

In [10]:

*# Размер датасета - 442 строки, 11 колонок*

data.shape

Out[10]: (442, 11)

In [11]:

total\_count = data.shape[0]

print('Всего строк: **{}**'.format(total\_count))

Всего строк: 442

In [12]:

*# Список колонок*

data.columns

Out[12]: Index(['AGE', 'SEX', 'BMI', 'BP', 'S1', 'S2', 'S3', 'S4', 'S5', 'S6', 'Y'], dtype='object')

In [13]:

*# Список колонок с типами данных*

data.dtypes

Out[13]: AGE int64

SEX int64 BMI float64 BP float64

S1 int64

S2 float64

S3 float64

S4 float64

S5 float64

S6 int64

Y int64 dtype: object

In [14]:

*# Проверим наличие пустых значений*

*# Цикл по колонкам датасета*

**for** col **in** data.columns:

*# Количество пустых значений - все значения заполнены* temp\_null\_count = data[data[col].isnull()].shape[0] print('**{}** - **{}**'.format(col, temp\_null\_count))

AGE - 0

SEX - 0

BMI - 0

BP - 0

S1 - 0

S2 - 0

S3 - 0

S4 - 0

S5 - 0

S6 - 0

Y - 0

In [15]:

*# Основные статистические характеристки набора данных*

data.describe()

Out[15]:

**AGE SEX BMI BP S1 S2 S3**

**count** 442.000000 442.000000 442.000000 442.000000 442.000000 442.000000 442.000000 442.00

**mean** 48.518100 1.468326 26.375792 94.647014 189.140271 115.439140 49.788462 4.07

**std** 13.109028 0.499561 4.418122 13.831283 34.608052 30.413081 12.934202 1.29

**min** 19.000000 1.000000 18.000000 62.000000 97.000000 41.600000 22.000000 2.00

**25%** 38.250000 1.000000 23.200000 84.000000 164.250000 96.050000 40.250000 3.00

**50%** 50.000000 1.000000 25.700000 93.000000 186.000000 113.000000 48.000000 4.00

**75%** 59.000000 2.000000 29.275000 105.000000 209.750000 134.500000 57.750000 5.00

**max** 79.000000 2.000000 42.200000 133.000000 301.000000 242.400000 99.000000 9.09

In [21]:

*# Определим уникальные значения для целевого признака*

data['SEX'].unique()

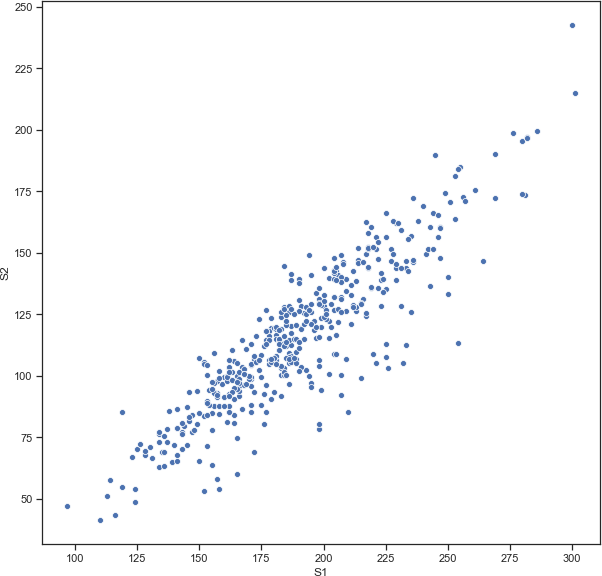
Out[21]: array([2, 1], dtype=int64)

# Визуальное исследование датасета

In [38]:

fig, ax = plt.subplots(figsize=(10,10)) sns.scatterplot(ax=ax, x='S1', y='S2', data=data)

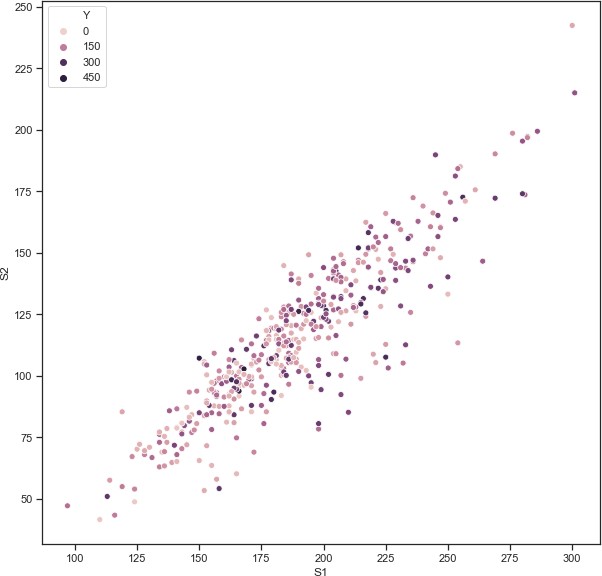
Out[38]: <matplotlib.axes.\_subplots.AxesSubplot at 0xe70c610>



In [42]:

fig, ax = plt.subplots(figsize=(10,10)) sns.scatterplot(ax=ax, x='S1', y='S2', data=data, hue='Y')

Out[42]: <matplotlib.axes.\_subplots.AxesSubplot at 0xfd81e70>

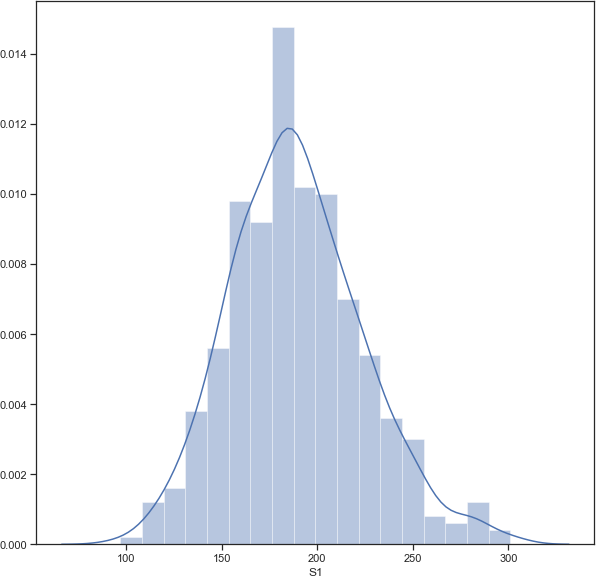


## Гистограмма

In [43]:

fig, ax = plt.subplots(figsize=(10,10)) sns.distplot(data['S1'])

Out[43]: <matplotlib.axes.\_subplots.AxesSubplot at 0xfd816b0>



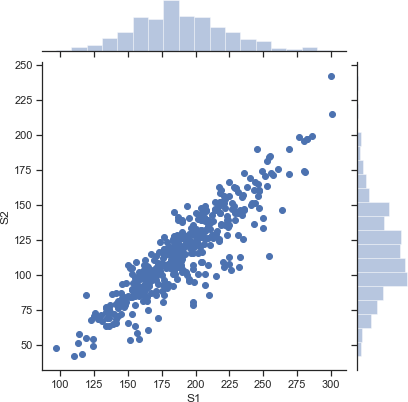
### Jointplot

Комбинация гистограмм и диаграмм рассеивания.

In [44]:

sns.jointplot(x='S1', y='S2', data=data)

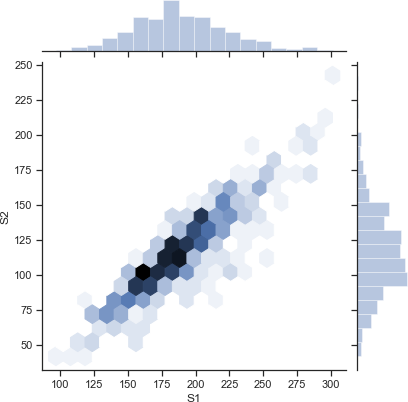
Out[44]: <seaborn.axisgrid.JointGrid at 0xfd663b0>



In [48]:

sns.jointplot(x='S1', y='S2', data=data, kind="hex")

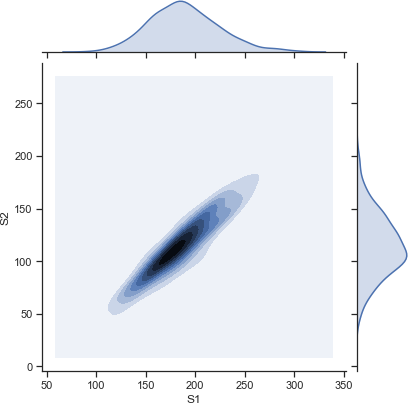
Out[48]: <seaborn.axisgrid.JointGrid at 0x1041bab0>



In [49]:

sns.jointplot(x='S1', y='S2', data=data, kind="kde")

Out[49]: <seaborn.axisgrid.JointGrid at 0x1079f450>

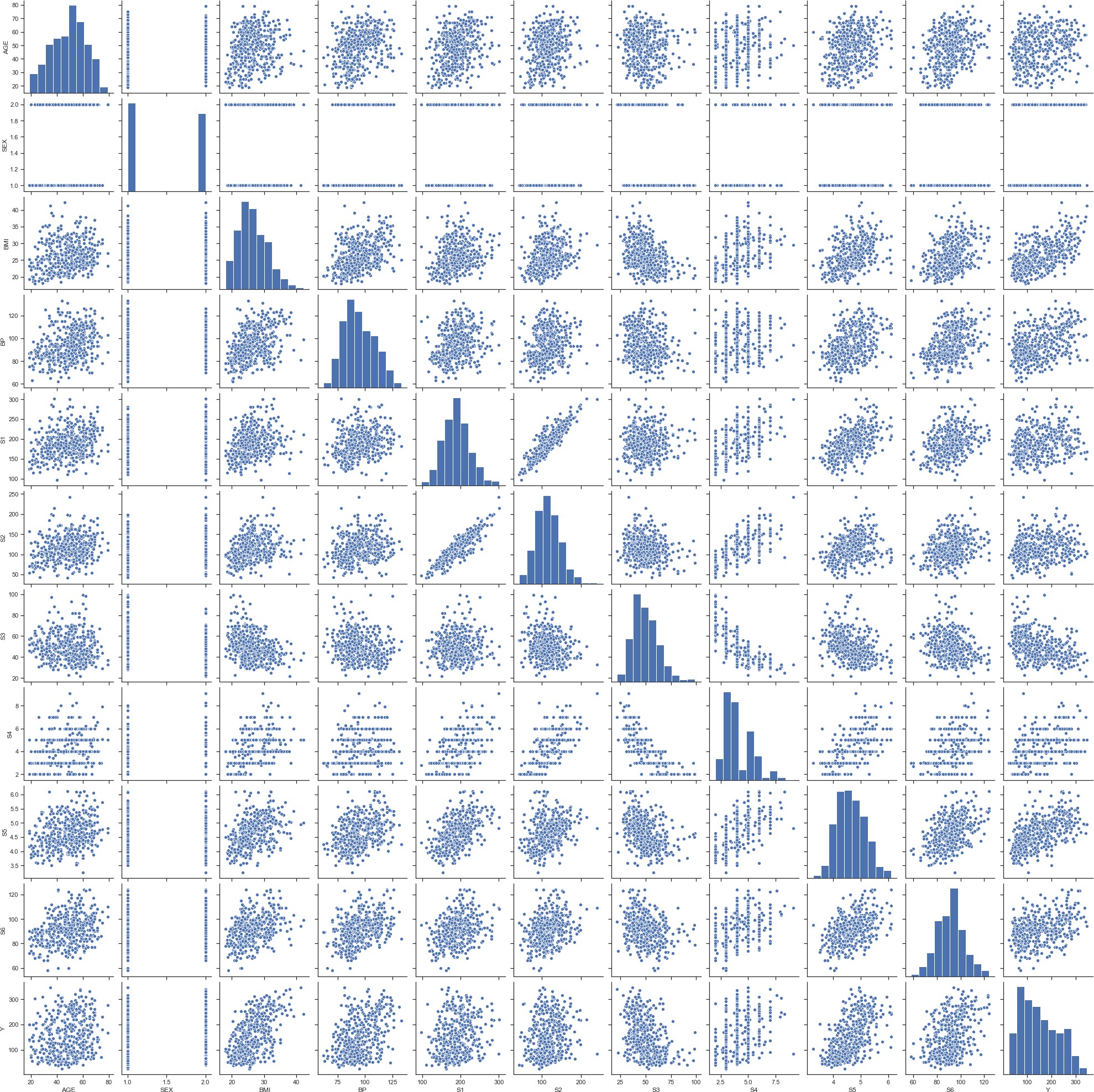


## "Парные диаграммы"

In [54]:

sns.pairplot(data)

Out[54]: <seaborn.axisgrid.PairGrid at 0x207eb810>



In [ ]:

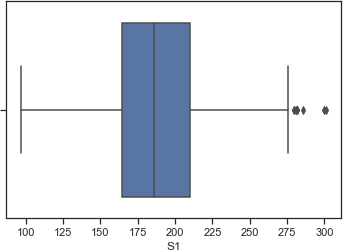
### Ящик с усами

Отображает одномерное распределение вероятности.

In [56]:

sns.boxplot(x=data['S1'])

Out[56]: <matplotlib.axes.\_subplots.AxesSubplot at 0x3066f290>

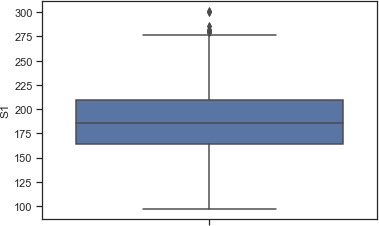


In [57]:

*# По вертикали*

sns.boxplot(y=data['S1'])

Out[57]: <matplotlib.axes.\_subplots.AxesSubplot at 0x3095a350>

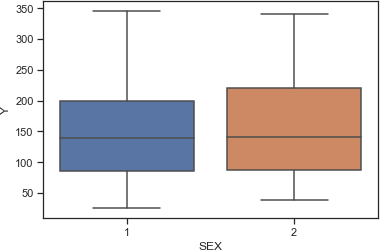


In [61]:

*# Распределение параметра S1 сгруппированные по Y.*

sns.boxplot(x='SEX', y='Y', data=data)

Out[61]: <matplotlib.axes.\_subplots.AxesSubplot at 0x301089d0>



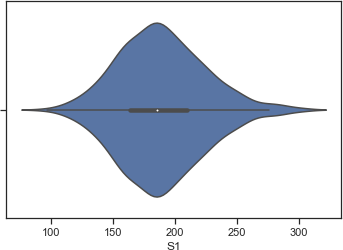
### Violin plot

Похоже на предыдущую диаграмму, но по краям отображаются распределения плотности

In [59]:

sns.violinplot(x=data['S1'])

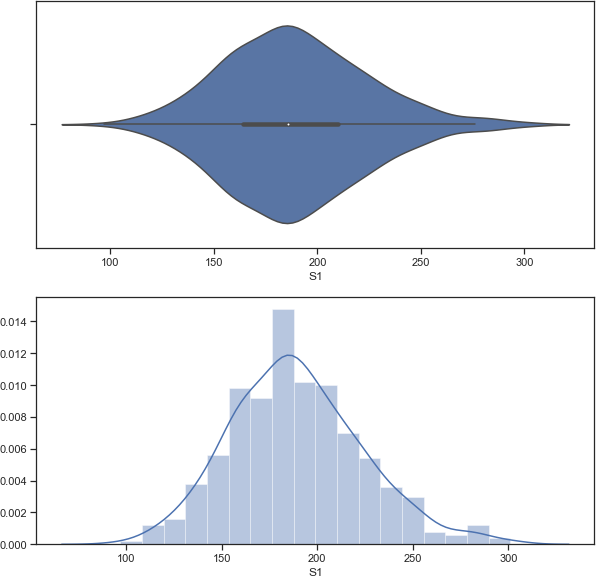
Out[59]: <matplotlib.axes.\_subplots.AxesSubplot at 0x313a38d0>



In [62]:

fig, ax = plt.subplots(2, 1, figsize=(10,10)) sns.violinplot(ax=ax[0], x=data['S1']) sns.distplot(data['S1'], ax=ax[1])

Out[62]: <matplotlib.axes.\_subplots.AxesSubplot at 0x2e5efe10>

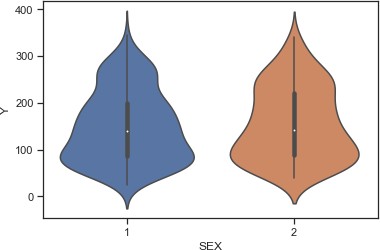


In [65]:

*# Распределение параметра Humidity сгруппированные по Occupancy.*

sns.violinplot(x='SEX', y='Y', data=data)

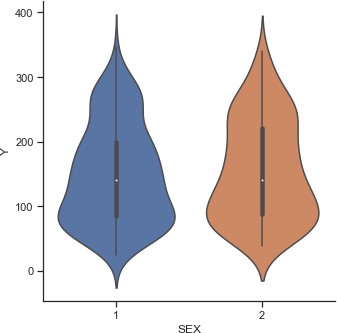
Out[65]: <matplotlib.axes.\_subplots.AxesSubplot at 0x305b53b0>



In [79]:

sns.catplot(y='Y', x='SEX', data=data, kind="violin", split=**True**)

Out[79]: <seaborn.axisgrid.FacetGrid at 0x340379b0>



# Информация о корреляции признаков

In [68]:

data.corr()

Out[68]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **AGE** | **SEX** | **BMI** | **BP** | **S1** | **S2** | **S3** | **S4** |  |
| **AGE** | 1.000000 | 0.173737 | 0.185085 | 0.335428 | 0.260061 | 0.219243 | -0.075181 | 0.203841 | 0.2707 |
| **SEX** | 0.173737 | 1.000000 | 0.088161 | 0.241010 | 0.035277 | 0.142637 | -0.379090 | 0.332115 | 0.1499 |
| **BMI** | 0.185085 | 0.088161 | 1.000000 | 0.395411 | 0.249777 | 0.261170 | -0.366811 | 0.413807 | 0.4461 |
| **BP** | 0.335428 | 0.241010 | 0.395411 | 1.000000 | 0.242464 | 0.185548 | -0.178762 | 0.257650 | 0.3934 |
| **S1** | 0.260061 | 0.035277 | 0.249777 | 0.242464 | 1.000000 | 0.896663 | 0.051519 | 0.542207 | 0.5155 |
| **S2** | 0.219243 | 0.142637 | 0.261170 | 0.185548 | 0.896663 | 1.000000 | -0.196455 | 0.659817 | 0.3183 |
| **S3** | -0.075181 | -0.379090 | -0.366811 | -0.178762 | 0.051519 | -0.196455 | 1.000000 | -0.738493 | -0.3985 |
| **S4** | 0.203841 | 0.332115 | 0.413807 | 0.257650 | 0.542207 | 0.659817 | -0.738493 | 1.000000 | 0.6178 |
| **S5** | 0.270774 | 0.149916 | 0.446157 | 0.393480 | 0.515503 | 0.318357 | -0.398577 | 0.617859 | 1.0000 |
| **S6** | 0.301731 | 0.208133 | 0.388680 | 0.390430 | 0.325717 | 0.290600 | -0.273697 | 0.417212 | 0.4646 |
| **Y** | 0.187889 | 0.043062 | 0.586450 | 0.441482 | 0.212022 | 0.174054 | -0.394789 | 0.430453 | 0.5658 |

In [69]:

data.corr(method='pearson')

Out[69]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **AGE** | **SEX** | **BMI** | **BP** | **S1** | **S2** | **S3** | **S4** |  |
| **AGE** | 1.000000 | 0.173737 | 0.185085 | 0.335428 | 0.260061 | 0.219243 | -0.075181 | 0.203841 | 0.2707 |
| **SEX** | 0.173737 | 1.000000 | 0.088161 | 0.241010 | 0.035277 | 0.142637 | -0.379090 | 0.332115 | 0.1499 |
| **BMI** | 0.185085 | 0.088161 | 1.000000 | 0.395411 | 0.249777 | 0.261170 | -0.366811 | 0.413807 | 0.4461 |
| **BP** | 0.335428 | 0.241010 | 0.395411 | 1.000000 | 0.242464 | 0.185548 | -0.178762 | 0.257650 | 0.3934 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S1** | 0.260061 | 0.035277 | 0.249777 | 0.242464 | 1.000000 | 0.896663 | 0.051519 | 0.542207 | 0.5155 |
| **S2** | 0.219243 | 0.142637 | 0.261170 | 0.185548 | 0.896663 | 1.000000 | -0.196455 | 0.659817 | 0.3183 |
| **S3** | -0.075181 | -0.379090 | -0.366811 | -0.178762 | 0.051519 | -0.196455 | 1.000000 | -0.738493 | -0.3985 |
| **S4** | 0.203841 | 0.332115 | 0.413807 | 0.257650 | 0.542207 | 0.659817 | -0.738493 | 1.000000 | 0.6178 |
| **S5** | 0.270774 | 0.149916 | 0.446157 | 0.393480 | 0.515503 | 0.318357 | -0.398577 | 0.617859 | 1.0000 |
| **S6** | 0.301731 | 0.208133 | 0.388680 | 0.390430 | 0.325717 | 0.290600 | -0.273697 | 0.417212 | 0.4646 |
| **Y** | 0.187889 | 0.043062 | 0.586450 | 0.441482 | 0.212022 | 0.174054 | -0.394789 | 0.430453 | 0.5658 |

In [70]:

data.corr(method='kendall')

Out[70]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **AGE** | **SEX** | **BMI** | **BP** | **S1** | **S2** | **S3** | **S4** |  |
| **AGE** | 1.000000 | 0.146580 | 0.136535 | 0.242111 | 0.182220 | 0.153612 | -0.073846 | 0.160898 | 0.1805 |
| **SEX** | 0.146580 | 1.000000 | 0.080424 | 0.215733 | 0.022809 | 0.110208 | -0.326188 | 0.297335 | 0.1431 |
| **BMI** | 0.136535 | 0.080424 | 1.000000 | 0.281770 | 0.194171 | 0.198583 | -0.249831 | 0.335625 | 0.3447 |
| **BP** | 0.242111 | 0.215733 | 0.281770 | 1.000000 | 0.188067 | 0.140253 | -0.131014 | 0.205948 | 0.2688 |
| **S1** | 0.182220 | 0.022809 | 0.194171 | 0.188067 | 1.000000 | 0.717229 | 0.010695 | 0.393367 | 0.3562 |
| **S2** | 0.153612 | 0.110208 | 0.198583 | 0.140253 | 0.717229 | 1.000000 | -0.133332 | 0.503579 | 0.2422 |
| **S3** | -0.073846 | -0.326188 | -0.249831 | -0.131014 | 0.010695 | -0.133332 | 1.000000 | -0.638633 | -0.3117 |
| **S4** | 0.160898 | 0.297335 | 0.335625 | 0.205948 | 0.393367 | 0.503579 | -0.638633 | 1.000000 | 0.4854 |
| **S5** | 0.180544 | 0.143172 | 0.344720 | 0.268863 | 0.356268 | 0.242250 | -0.311775 | 0.485410 | 1.0000 |
| **S6** | 0.201784 | 0.168199 | 0.266373 | 0.264566 | 0.227139 | 0.194082 | -0.200545 | 0.307397 | 0.3162 |
| **Y** | 0.130709 | 0.030630 | 0.391195 | 0.289352 | 0.154016 | 0.129665 | -0.278884 | 0.324734 | 0.4089 |

In [71]:

data.corr(method='spearman')

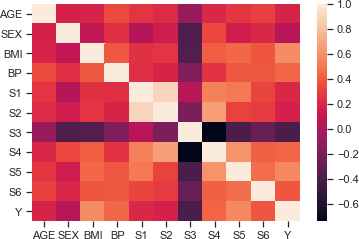
Out[71]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **AGE** | **SEX** | **BMI** | **BP** | **S1** | **S2** | **S3** | **S4** |  |
| **AGE** | 1.000000 | 0.177463 | 0.200554 | 0.350859 | 0.262524 | 0.221711 | -0.106973 | 0.221017 | 0.2651 |
| **SEX** | 0.177463 | 1.000000 | 0.098079 | 0.261508 | 0.027790 | 0.134695 | -0.394584 | 0.337524 | 0.1746 |
| **BMI** | 0.200554 | 0.098079 | 1.000000 | 0.397985 | 0.287829 | 0.295494 | -0.371172 | 0.459068 | 0.4916 |
| **BP** | 0.350859 | 0.261508 | 0.397985 | 1.000000 | 0.275224 | 0.205638 | -0.191033 | 0.280799 | 0.3960 |
| **S1** | 0.262524 | 0.027790 | 0.287829 | 0.275224 | 1.000000 | 0.878793 | 0.015308 | 0.520674 | 0.5128 |
| **S2** | 0.221711 | 0.134695 | 0.295494 | 0.205638 | 0.878793 | 1.000000 | -0.197435 | 0.652283 | 0.3499 |
| **S3** | -0.106973 | -0.394584 | -0.371172 | -0.191033 | 0.015308 | -0.197435 | 1.000000 | -0.789694 | -0.4504 |
| **S4** | 0.221017 | 0.337524 | 0.459068 | 0.280799 | 0.520674 | 0.652283 | -0.789694 | 1.000000 | 0.6403 |
| **S5** | 0.265176 | 0.174625 | 0.491609 | 0.396071 | 0.512864 | 0.349947 | -0.450420 | 0.640390 | 1.0000 |
| **S6** | 0.296235 | 0.203277 | 0.384664 | 0.381219 | 0.332173 | 0.286483 | -0.290863 | 0.413700 | 0.4530 |
| **Y** | 0.197822 | 0.037401 | 0.561382 | 0.416241 | 0.232429 | 0.195834 | -0.410022 | 0.448931 | 0.5894 |

In [72]:

sns.heatmap(data.corr())

Out[72]: <matplotlib.axes.\_subplots.AxesSubplot at 0x31bc55d0>

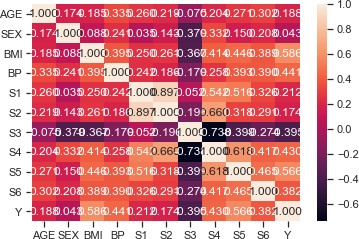


In [74]:

*# Вывод значений в ячейках*

sns.heatmap(data.corr(), annot=**True**, fmt='.3f')

Out[74]: <matplotlib.axes.\_subplots.AxesSubplot at 0x2ccbdf70>

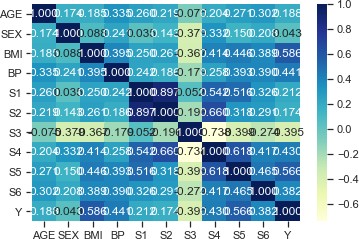


In [75]:

*# Изменение цветовой гаммы*

sns.heatmap(data.corr(), cmap='YlGnBu', annot=**True**, fmt='.3f')

Out[75]: <matplotlib.axes.\_subplots.AxesSubplot at 0x31c7b790>



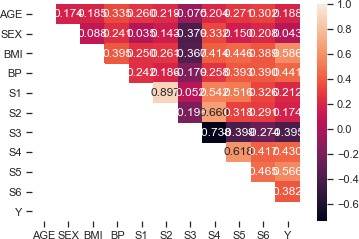
In [76]:

*# Треугольный вариант матрицы*

mask = np.zeros\_like(data.corr(), dtype=np.bool) *# чтобы оставить нижнюю часть матрицы # mask[np.triu\_indices\_from(mask)] = True*

*# чтобы оставить верхнюю часть матрицы* mask[np.tril\_indices\_from(mask)] = **True** sns.heatmap(data.corr(), mask=mask, annot=**True**, fmt='.3f')

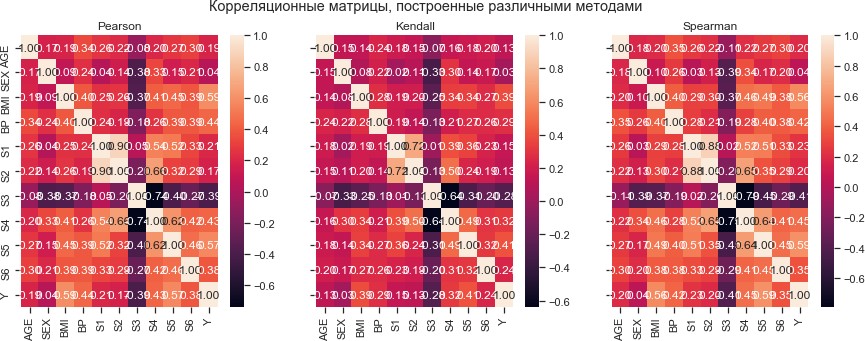
Out[76]: <matplotlib.axes.\_subplots.AxesSubplot at 0x31ae5f10>



In [77]:

fig, ax = plt.subplots(1, 3, sharex='col', sharey='row', figsize=(15,5)) sns.heatmap(data.corr(method='pearson'), ax=ax[0], annot=**True**, fmt='.2f') sns.heatmap(data.corr(method='kendall'), ax=ax[1], annot=**True**, fmt='.2f') sns.heatmap(data.corr(method='spearman'), ax=ax[2], annot=**True**, fmt='.2f') fig.suptitle('Корреляционные матрицы, построенные различными методами') ax[0].title.set\_text('Pearson')

ax[1].title.set\_text('Kendall') ax[2].title.set\_text('Spearman')



In [ ]: