```
In [2]: import os
        import math
        import numpy as np
        import pandas as pd
        import plotly.express as px
        import plotly.graph_objects as go
```

```
In [3]: red_color="#E93F36"
        blue_color = "#04104E"
```

```
In [6]: data_path = "report_builder/data"
        data_file_name = "Benchling_updated.csv"

        df = pd.read_csv(os.path.join(data_path,data_file_name))
```

Сводная информация по всей БД

```
In [7]: df.head()
```

Out[7]:

	Run	Sample sheet_Sample_ID	Clinical/RnD	Support	Clinical QC Required	Is it top off seq?	Is it reseq?	Sample target	FlowCell	Project	...	Extraction kit	Library construction kit	Library prob set kit	Case_ID (patient ID)	Specimen_ID
0	230512_NovaD_XTHS2	WES-normal-230512_NovaD_Sample_1	RnD	No	No	NaN	NaN	NaN	S1	Coverage map	...	Maxwell RSC Stabilized Saliva DNA Kit	Agilent XTHS2 RNA	V8+UTR	PT003656	NaN
1	230512_NovaD_XTHS2	WES-normal-230512_NovaD_Sample_2	RnD	No	No	NaN	NaN	NaN	S1	Coverage map	...	Maxwell RSC Stabilized Saliva DNA Kit	Agilent XTHS2 RNA	V8+UTR	PT003657	NaN
2	230512_NovaD_XTHS2	WES-normal-230512_NovaD_Sample_3	RnD	No	No	NaN	NaN	NaN	S1	Coverage map	...	Maxwell RSC Stabilized Saliva DNA Kit	Agilent XTHS2 RNA	V8+UTR	PT003658	NaN
3	230512_NovaD_XTHS2	WES-normal-230512_NovaD_Sample_4	RnD	No	No	NaN	NaN	NaN	S1	Coverage map	...	Maxwell RSC Stabilized Saliva DNA Kit	Agilent XTHS2 RNA	V8+UTR	PT003659	NaN
4	230512_NovaD_XTHS2	WES-normal-230512_NovaD_Sample_5	RnD	No	No	NaN	NaN	NaN	S1	Coverage map	...	Maxwell RSC Stabilized Saliva DNA Kit	Agilent XTHS2 RNA	V8+UTR	PT003660	NaN

5 rows × 24 columns

Количество уникальных образцов

```
In [8]: df["Sample sheet_Sample_ID"].drop_duplicates().shape
```

```
Out[8]: (2213,)
```

Сводная информация по каждому столбцу

```
In [9]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2213 entries, 0 to 2212
Data columns (total 24 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Run                     2213 non-null   object
 1   Sample sheet_Sample_ID  2213 non-null   object
 2   Clinical/RnD            2213 non-null   object
 3   Support                 2213 non-null   object
 4   Clinical QC Required    2199 non-null   object
 5   Is it top off seq?      0 non-null      float64
 6   Is it reseq?            0 non-null      float64
 7   Sample target           0 non-null      float64
 8   FlowCell                2213 non-null   object
 9   Project                 2213 non-null   object
 10  Source                  2213 non-null   object
 11  Tumor/Normal            2213 non-null   object
 12  Input material type     2213 non-null   object
 13  Extraction
place         466 non-null    object
 14  Extraction kit          2213 non-null   object
 15  Library construction kit  1756 non-null   object
 16  Library prob set kit    892 non-null    object
 17  Case_ID (patient ID)    1101 non-null   object
 18  Specimen_ID             525 non-null    object
 19  Library Lab_ID
(unique)  2213 non-null   object
 20  Extract BG_ID           2213 non-null   object
 21  Patient BG_ID           2213 non-null   object
 22  Sample BG_ID            2213 non-null   object
 23  Library Concentration   2213 non-null   int64
dtypes: float64(3), int64(1), object(20)
memory usage: 415.1+ KB
```

Столбцы с пропущенными значениями

```
In [10]: df["Extraction kit"] = df["Extraction kit"].replace("Unknown",np.nan)
```

```
In [11]: nan_columns = [i for i in df.columns if df[i].isnull().any()]
         nan_columns
```

```
Out[11]: ['Clinical QC Required',
          'Is it top off seq?',
          'Is it reseq?',
          'Sample target',
          'Extraction \nplace',
          'Extraction kit',
          'Library construction kit',
          'Library prob set kit',
          'Case_ID (patient ID)',
          'Specimen_ID']
```

Аналитика столбцов участвующих в отчете

Top-off/Reseq

Процент образцов по каждому проекту, не требующих Top-off или Reseq

```
In [12]: df["sample_by_project"] = df.groupby("Project")[
             "Sample sheet_Sample_ID"
         ].transform("nunique")

         df["sample_by_extract"] = df.groupby(["Project","Extract BG_ID"])[
             "Sample sheet_Sample_ID"
         ].transform("nunique")

         df["sample_by_library"] = df.groupby(
             ["Project","Extract BG_ID", "Library Lab_ID \n(unique)"]
         )["Sample sheet_Sample_ID"].transform("nunique")
```

```
In [13]: df["good_sample_flag"] = np.select(
             [
                 df["sample_by_extract"] == 1
             ],[True],default=False
         )
```

```
In [14]: sample_report = (
             df.groupby(["Project","sample_by_project","good_sample_flag"])
             .agg({"Sample sheet_Sample_ID":"nunique"})
             .reset_index()
         )
```

```
In [15]: sample_report_pivoted = pd.pivot(
             sample_report,
             index=["Project","sample_by_project"],
             columns="good_sample_flag",
             values="Sample sheet_Sample_ID"
         ).fillna(0).astype(int).reset_index()

         sample_report_pivoted["good_sample %"] = round(
             sample_report_pivoted[True] / sample_report_pivoted["sample_by_project"] * 100,2
         )
```

```
In [16]: sample_report_pivoted[["Project","sample_by_project","good_sample %"]].sort_values(by="good_sample %").set_index("Project")
```

Out[16]:

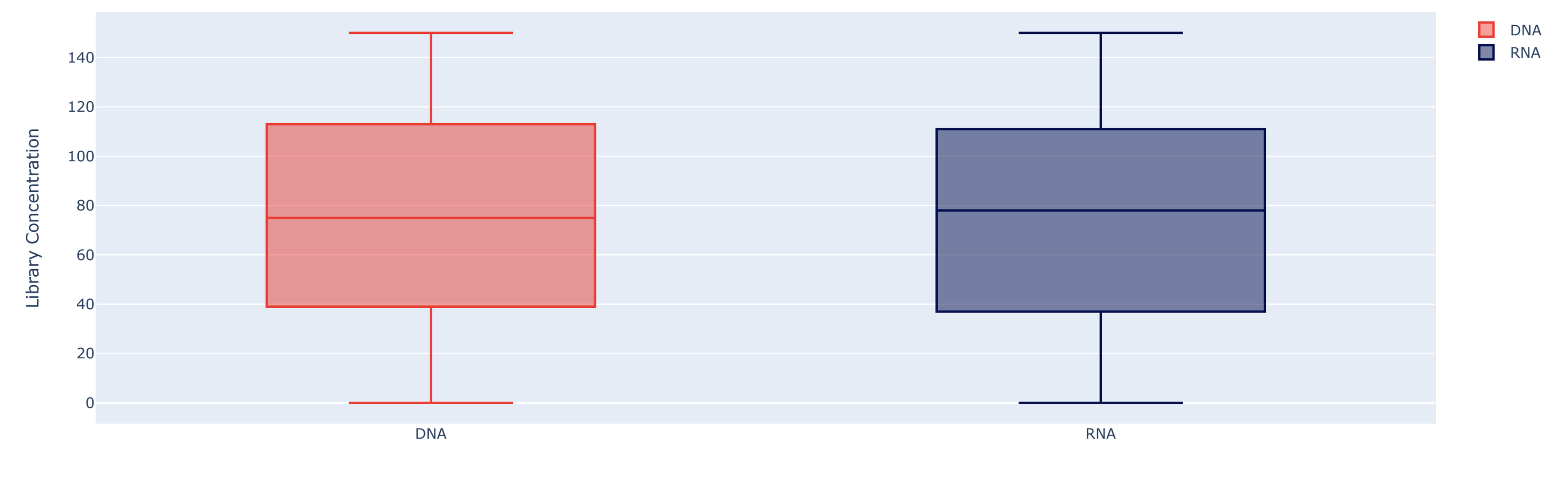
good_sample_flag	sample_by_project	good_sample %
Project		
cfRNA Deconvolution	166	0.00
cfDNA	12	0.00
V8+UTR validation	94	0.00
UM.Valdes.BC.cfDNA	2	0.00
Heme validation	4	0.00
cfDNA TMB references	48	0.00
cfDNA Fusions	905	0.55
MDACC_Yam_Artemis	195	6.15
FEASY_trial	17	17.65
Immune Status	193	45.08
MGH.David Ting.PDAC	25	52.00
Other	93	59.14
WCMC_CUP_Sternberg_Retrospective	53	60.38
NCI_FL_Roschewski	79	64.56
Mayo_DIAL_Villasboas	32	68.75
cfDNA Plasma	16	87.50
MDA.Heymach.SCLC	93	97.85
MDA.Flowers.PTLD	31	100.00
HRS validation	1	100.00
HRD validation	54	100.00
WU.Huang.Meningioma	18	100.00
Coverage map	10	100.00
Cell Atlas	19	100.00
MDA.Strati.FL.Blood	8	100.00
Bravo Validation	45	100.00

Результаты QC

Распределение Library Concentration по типу образцов (DNA/RNA)

```
In [17]: df["sample_type"] = np.select(
             [
                 df["Extract BG_ID"].str[0] == "D"
             ],["DNA"],default="RNA"
         )
```

```
In [18]: fig = go.Figure()

         fig.add_trace(
             go.Box(
                 y=df.query("sample_type == 'DNA'")["Library Concentration"],
                 name='DNA',
                 marker_color = red_color
             )
         )

         fig.add_trace(
             go.Box(
                 y=df.query("sample_type == 'RNA'")["Library Concentration"],
                 name='RNA',
                 marker_color = blue_color
             )
         )

         fig.update_layout(
             yaxis_title='Library Concentration'
         )

         fig.show()
```



Распределение Library Concentration по типу образцов (DNA/RNA) с границами QC

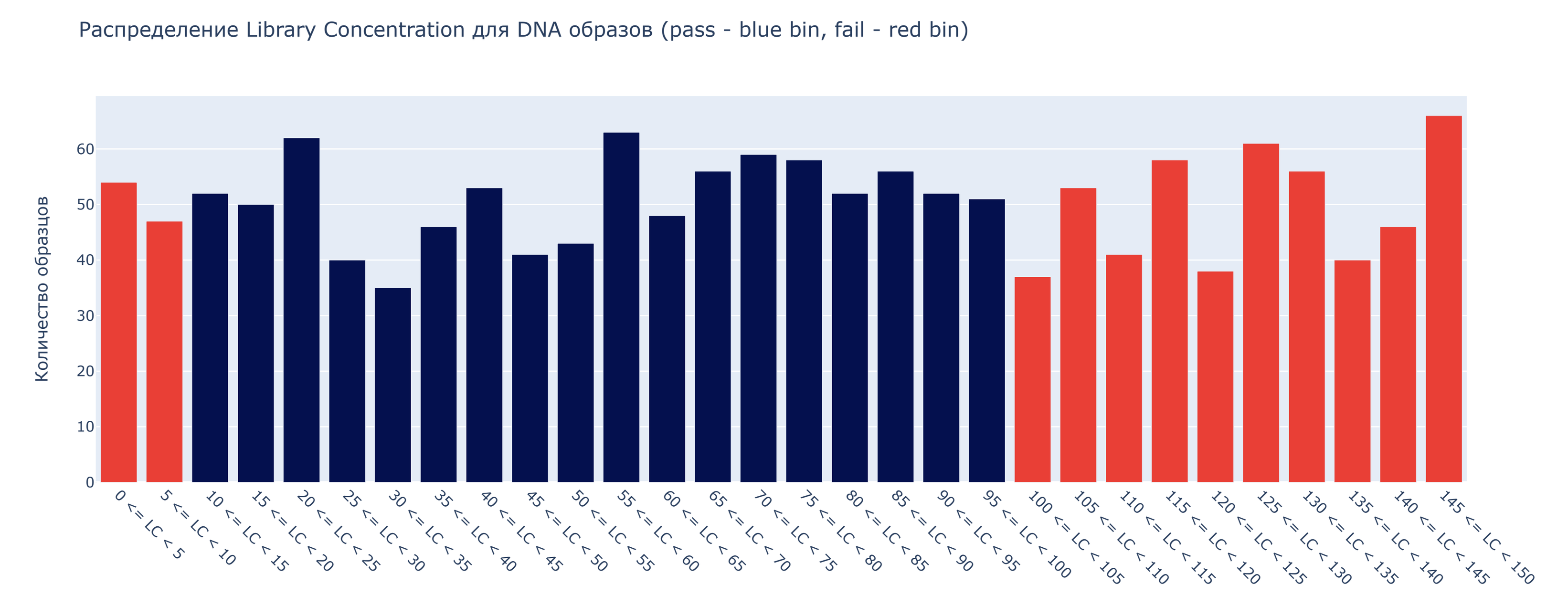
```
In [19]: bin_number = math.ceil((df["Library Concentration"].max() - df["Library Concentration"].min()) / 5)
```

```
In [20]: df["Library Concentration_bin"] = pd.cut(df["Library Concentration"],bin_number,right=False)
```

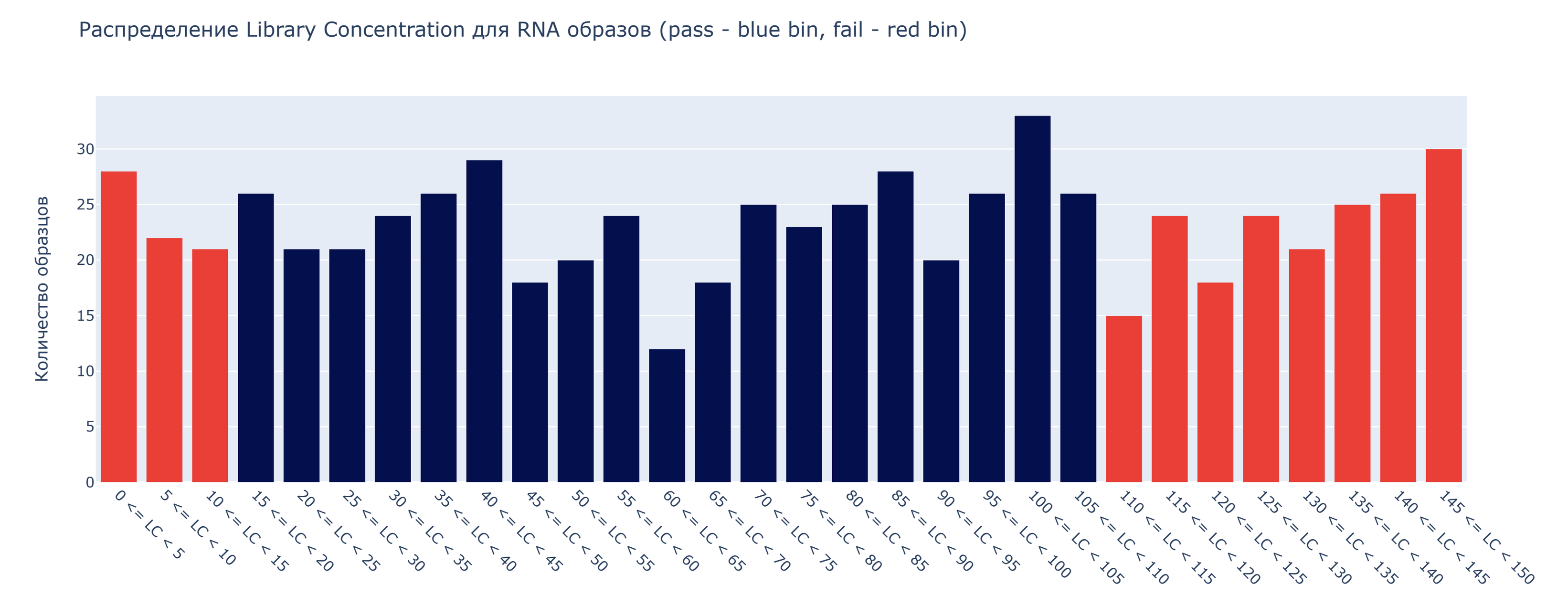
```
In [21]: library_concentration_report = pd.pivot_table(
             df.groupby(["Library Concentration_bin","sample_type"]).agg({"Sample sheet_Sample_ID":"nunique"}),
             index="Library Concentration_bin",
             columns="sample_type",
             values="Sample sheet_Sample_ID"
         ).reset_index()

         library_concentration_report["right_border"] = library_concentration_report["Library Concentration_bin"].apply(
             lambda x: int(x.right)
         )

         library_concentration_report["left_border"] = library_concentration_report["Library Concentration_bin"].apply(
             lambda x: int(x.right)
         )
```

```
In [22]: library_concentration_report["Library Concentration_bin_label"] = (
             library_concentration_report["Library Concentration_bin"].apply(
                 lambda x: f"{int(x.left)} <= LC < {int(x.right)}"
             )
         )
```

```
In [23]: library_concentration_report["Library Concentration_bin_color_dna"] = np.select(
             [
                 (library_concentration_report["left_border"] <= 10)
                 | (library_concentration_report["right_border"] > 100)
             ],[red_color],default=blue_color
         )

         fig = go.Figure()

         fig.add_trace(
             go.Bar(
                 x=library_concentration_report["Library Concentration_bin_label"],
                 y=library_concentration_report["DNA"],
                 marker_color=library_concentration_report["Library Concentration_bin_color_dna"]
             )
         )

         fig.update_layout(
             title_text="Распределение Library Concentration для DNA образцов (pass - blue bin, fail - red bin)",
             yaxis_title="Количество образцов"
         )

         fig.update_xaxes(tickangle=45)

         fig.show()
```




```
In [24]: library_concentration_report["Library Concentration_bin_color_rna"] = np.select(
             [
                 (library_concentration_report["left_border"] <= 15)
                 | (library_concentration_report["right_border"] > 110)
             ],[red_color],default=blue_color
         )

         fig = go.Figure()

         fig.add_trace(
             go.Bar(
                 x=library_concentration_report["Library Concentration_bin_label"],
                 y=library_concentration_report["RNA"],
                 marker_color=library_concentration_report["Library Concentration_bin_color_rna"]
             )
         )

         fig.update_layout(
             title_text="Распределение Library Concentration для RNA образцов (pass - blue bin, fail - red bin)",
             yaxis_title="Количество образцов"
         )

         fig.update_xaxes(tickangle=45)

         fig.show()
```



Исключениеобразцов

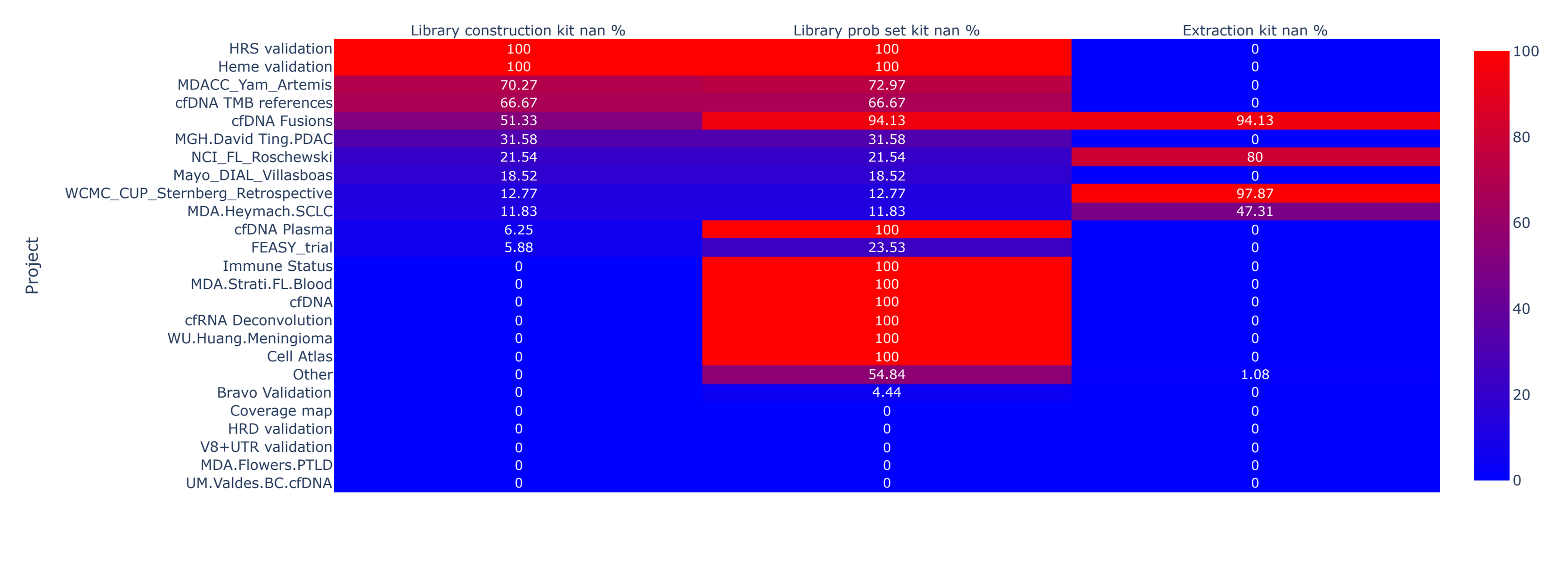
Процент библиотек с пропущенными значениями указанных для фильтра столбцов по проектам

```
In [25]: columns_for_analysis = [
             "Library construction kit",
             "Library prob set kit",
             "Extraction kit",
         ]
```

```
In [26]: df["total_library_number"] = df.groupby("Project")["Library Lab_ID \n(unique)"].transform("nunique")
```

```
In [27]: empty_columns_report = pd.melt(
             df,
             id_vars=["Project","total_library_number","Library Lab_ID \n(unique)"],
             value_vars=columns_for_analysis
         )

         empty_columns_report = empty_columns_report[empty_columns_report["value"].isna()].drop_duplicates()

         empty_columns_report_pivoted = pd.pivot_table(
             (
                 empty_columns_report
                 .groupby(["Project","total_library_number","variable"])
                 .agg({"Library Lab_ID \n(unique)":"nunique"})
             ).reset_index(),
             index=["Project","total_library_number"],
             columns="variable",
             values="Library Lab_ID \n(unique)"
         ).fillna(0).astype(int).reset_index()

         for col in columns_for_analysis:
             empty_columns_report_pivoted[f"{col} nan %"] = round(
                 empty_columns_report_pivoted[col] / empty_columns_report_pivoted["total_library_number"] * 100, 2
             )
         columns_for_analysis_in_percent = list(map(lambda col: f"{col} nan %",columns_for_analysis))

         total_projects = df[["Project"]].drop_duplicates()

         final_report = total_projects.merge(
             empty_columns_report_pivoted[
                 ["Project"] + columns_for_analysis_in_percent
             ],
             how="left",
             on="Project"
         ).set_index("Project").fillna(0).sort_values(by=columns_for_analysis_in_percent,ascending=False)
```

```
In [28]: fig = px.imshow(
             final_report,
             text_auto=True,
             aspect='auto',
             color_continuous_scale='Bluered'
         )

         fig.update_xaxes(side="top")
         fig.show()
```




```
In [ ]:
```