Hughes_Trevor_DA201_Assignment_Notebo

Data Preparation

Import Required Libraries

In [66]:
```
# Import the necessary libraries.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as mticker
import matplotlib.dates as mdates
import matplotlib.patches as mpatches
import matplotlib.lines as mlines
from matplotlib.gridspec import GridSpec
from matplotlib.lines import Line2D
import seaborn as sns
from textblob import TextBlob

# Optional - Ignore warnings.
import warnings
warnings.filterwarnings('ignore')
```

Adding data showing 'population size' of each sub ICB

based on https://digital.nhs.uk/data-and-information/publications/statistical/patients-registered-at-a-gp-practice/january-2022

In [2]:
```
nhs_stats = pd.read_csv("gp-reg-pat-prac-all.csv")
nhs_stats.head(3)
```

Out[2]:

	PUBLICATION	EXTRACT_DATE	TYPE	CCG_CODE	ONS_CCG_CODE	CODE	POST
0	GP_PRAC_PAT_LIST	01Jan2022	GP	00L	E38000130	A84002	NE65
1	GP_PRAC_PAT_LIST	01Jan2022	GP	00L	E38000130	A84005	NE2
2	GP_PRAC_PAT_LIST	01Jan2022	GP	00L	E38000130	A84006	NE6

In [3]:
```
# Aggregate NUMBER_OF_PATIENTS by ONS_CCG_CODE - (note these are sub-ICB codes).
patient_population = (
    nhs_stats.groupby("ONS_CCG_CODE")["NUMBER_OF_PATIENTS"].sum().reset_index()
    .rename(columns={"NUMBER_OF_PATIENTS": "total_patients"})
    .sort_values("total_patients", ascending=False)
```

```
)
# Find the total number of registered patients
total_patients = nhs_stats["NUMBER_OF_PATIENTS"].sum()
print(f"Total registered patients: {total_patients:,}")
patient_population.head(3)
```

```
Total registered patients: 61,469,262
```

Out[3]:

	ONS_CCG_CODE	total_patients
104	E38000256	2751843
103	E38000255	2346009
92	E38000244	2047004

Data checking/cleaning/preparation of actual_duration

In [4]:
```
# Import and sense-check the actual_duration.csv data set as ad.
ad_raw = pd.read_csv("actual_duration.csv")

# View the DataFrame.
print(ad_raw.shape)
ad_raw.head(3)
```

```
(137793, 8)
```

Out[4]:

	sub_icb_location_code	sub_icb_location_ons_code	sub_icb_location_name	icb_ons_cod
0	00L	E38000130	NHS North East and North Cumbria ICB - 00L	E5400005(
1	00L	E38000130	NHS North East and North Cumbria ICB - 00L	E5400005(
2	00L	E38000130	NHS North East and North Cumbria ICB - 00L	E5400005(

In [5]:
```
# Determine whether there are missing values.
ad_raw_na = ad_raw[ad_raw.isna().any(axis=1)]
ad_raw_na.shape
```

Out[5]: (0, 8)

In [6]:
```
# Check for rows that are identical across all columns
ad_raw_duplicate_rows = ad_raw[ad_raw.duplicated(keep=False)]

# Count how many duplicates exist
ad_raw_duplicate_count = ad_raw.duplicated().sum()
print(f"Total duplicate rows: {ad_raw_duplicate_count}")
```

```
# Sort by all columns to group identical rows together
ad_raw_ordered_duplicates = ad_raw_duplicate_rows.sort_values(by=ad_raw.columns.

# Display the result
ad_raw_ordered_duplicates
```

```
Total duplicate rows: 0
```

Out[6]:

	sub_icb_location_code	sub_icb_location_ons_code	sub_icb_location_name	icb_ons_code

In [7]:
```
# Determine the metadata of the data set.
print(ad_raw.dtypes)
```

```
sub_icb_location_code        object
sub_icb_location_ons_code    object
sub_icb_location_name        object
icb_ons_code                 object
region_ons_code              object
appointment_date             object
actual_duration              object
count_of_appointments         int64
dtype: object
```

In [8]:
```
# Determine date range of the date set.
# Change the date format of ad['appointment_date'].
ad_raw['appointment_date'] = pd.to_datetime(ad_raw['appointment_date'])
ad_raw_min_date = ad_raw['appointment_date'].min()
ad_raw_max_date = ad_raw['appointment_date'].max()

print(f"Earliest date: {ad_raw_min_date}")
print(f"Latest date: {ad_raw_max_date}")
```

```
Earliest date: 2021-12-01 00:00:00
Latest date: 2022-06-30 00:00:00
```

In [9]:
```
# merging with population data
# Join to appointments_regional table
ad_merged = ad_raw.merge(
    patient_population,
    left_on="sub_icb_location_ons_code",
    right_on="ONS_CCG_CODE",
    how="left"
)

# Drop the redundant ONS_CCG_CODE column after join
ad_merged = ad_merged.drop(columns=["ONS_CCG_CODE"])

# Check for any ICB codes that didn't match
unmatched = ad_merged[ad_merged["total_patients"].isna()]["sub_icb_location_code
print(f"Unmatched ICB codes: {len(unmatched)}")
if len(unmatched) > 0:
    print(unmatched)

pop = ad_merged.groupby(['sub_icb_location_code', 'sub_icb_location_name'])['tot
print(f"Total population (all sub-ICBs):  {pop['total_patients'].sum():,.0f}")
```

```
Unmatched ICB codes: 6
['01Y' '06H' '15E' '15M' '78H' 'D2P2L']
Total population (all sub-ICBs):  55,468,820
```

In [10]:
```
# Manually map missing data
# Sum GP practice populations up to sub-ICB level
gp_totals = nhs_stats.groupby('CCG_CODE')['NUMBER_OF_PATIENTS'].sum().reset_inde
gp_totals.columns = ['sub_icb_location_code', 'total_patients_gp']

# Identify the 6 sub-ICBs with missing population
missing_codes = ['01Y', '06H', '15E', '15M', '78H', 'D2P2L']

# Match missing data
ad_merged = ad_merged.merge(gp_totals, on='sub_icb_location_code', how='left')
ad_merged.loc[ad_merged['sub_icb_location_code'].isin(missing_codes),
    'total_patients'] = ad_merged.loc[ad_merged['sub_icb_location_code'].isin(mi
ad_merged = ad_merged.drop(columns='total_patients_gp')

# --- Verification ---
pop = ad_merged.groupby(['sub_icb_location_code', 'sub_icb_location_name'])['tot
print(f"Total population (all sub-ICBs):  {pop['total_patients'].sum():,.0f}")
```

```
Total population (all sub-ICBs):  61,469,262
```

In [11]:
```
# create df for further analysis
ad = ad_merged
ad.head(3)
ad.to_csv('cleaned_ad.csv', index=False)
```

Data checking/cleaning/preparation of appointments_regional

In [12]:
```
# Import and sense-check the appointments_regional.csv data set as ar.
ar_raw = pd.read_csv("appointments_regional.csv")

# View the DataFrame.
print(ar_raw.shape)
ar_raw.head(3)
```

```
(596821, 7)
```

Out[12]:

	icb_ons_code	appointment_month	appointment_status	hcp_type	appointment_mode
0	E54000034	2020-01	Attended	GP	Face-to-Face
1	E54000034	2020-01	Attended	GP	Face-to-Face
2	E54000034	2020-01	Attended	GP	Face-to-Face

In [13]:
```
# Determine whether there are missing values.
ar_raw_na = ar_raw[ar_raw.isna().any(axis=1)]
ar_raw_na.shape
```

Out[13]: (0, 7)

In [14]:
```
# Determine whether there are any duplicate values.
ar_raw_duplicate_rows = ar_raw[ar_raw.duplicated(keep=False)]

# Counts only the extra copies (ignores the first occurrence)
ar_raw_duplicate_count = ar_raw.duplicated().sum()
print(f"Total duplicate rows: {ar_raw_duplicate_count}")

# Sort by all columns to group identical rows together
ar_raw_ordered_duplicates = ar_raw_duplicate_rows.sort_values(by=ar_raw.columns.

# Display the result
ar_raw_ordered_duplicates
```

```
Total duplicate rows: 21604
```

Out[14]:

	icb_ons_code	appointment_month	appointment_status	hcp_type	appointment
496290	E54000008	2020-01	Attended	GP	Face-
505707	E54000008	2020-01	Attended	GP	Face-
516594	E54000008	2020-01	Attended	GP	Face-
487191	E54000008	2020-01	Attended	GP	Face-
522453	E54000008	2020-01	Attended	GP	Face-
...	...	...	...	...	
294829	E54000061	2022-06	Unknown	Unknown	Face-
289282	E54000061	2022-06	Unknown	Unknown	Tel
307348	E54000061	2022-06	Unknown	Unknown	Tel
294838	E54000061	2022-06	Unknown	Unknown	Tel
307349	E54000061	2022-06	Unknown	Unknown	Tel

40135 rows × 7 columns

In [15]:
```
# Remove duplicated rows and recheck
ar_cleaned = ar_raw.drop_duplicates()

# Counts only the extra copies (ignores the first occurrence)
ar_cleaned_duplicate_count = ar_cleaned.duplicated().sum()
print(f"Total duplicate rows after cleaning: {ar_cleaned_duplicate_count}")
print(ar_cleaned.shape)
```

```
Total duplicate rows after cleaning: 0
(575217, 7)
```

In [16]:
```
# Determine the metadata of the data set.
ar_cleaned.dtypes
```

Out[16]:
```
icb_ons_code                          object
appointment_month                     object
appointment_status                    object
hcp_type                              object
appointment_mode                      object
time_between_book_and_appointment     object
count_of_appointments                  int64
dtype: object
```

In [17]:
```
# Determine the minimum and maximum dates in the ar DataFrame.
# Use appropriate docstrings.
ar_cleaned_min_date = ar_cleaned['appointment_month'].min()
ar_cleaned_max_date = ar_cleaned['appointment_month'].max()

print(f"Earliest date: {ar_cleaned_min_date}")
print(f"Latest date: {ar_cleaned_max_date}")
```

```
Earliest date: 2020-01
Latest date: 2022-06
```

In [18]:
```
# filter for only dates after 08-2021 to match NC file.
ar_cleaned['appointment_month'] = pd.to_datetime(ar_cleaned['appointment_month']
start_date = '2021-08-01'
end_date = '2022-06-30'

# Create a new DataFrame based on the condition
ar_filtered = ar_cleaned[(ar_cleaned['appointment_month'] >= start_date) & (ar_c
ar_min_date = ar_filtered['appointment_month'].min()
ar_max_date = ar_filtered['appointment_month'].max()
print(f"Earliest date: {ar_min_date}")
print(f"Latest date: {ar_max_date}")
ar_filtered.shape
```

```
Earliest date: 2021-08-01 00:00:00
Latest date: 2022-06-01 00:00:00
```

Out[18]:
```
(215402, 7)
```

In [19]:
```
# merging with population data using appointments duration which has both ICB an

# Keep only the ALL/ALL summary rows and sum by ICB ONS code
patients_by_icb = (
    nhs_stats[(nhs_stats["SEX"] == "ALL") & (nhs_stats["AGE"] == "ALL")]
    .groupby("ONS_CCG_CODE")["NUMBER_OF_PATIENTS"]
    .sum()
    .reset_index()
    .rename(columns={"ONS_CCG_CODE": "sub_icb_ons_code", "NUMBER_OF_PATIENTS": "
)

# Then join sub-ICB to ICB mapping from your appointments duration dataframe
icb_mapping = ad[["sub_icb_location_ons_code", "icb_ons_code"]].drop_duplicates(

patients_by_icb = patients_by_icb.merge(
    icb_mapping,
    left_on="sub_icb_ons_code",
    right_on="sub_icb_location_ons_code",
    how="left"
).groupby("icb_ons_code")["total_patients"].sum().reset_index()

# Now merge onto your appointments regional data
ar_merged = ar_filtered.merge(patients_by_icb, on="icb_ons_code", how="left")
```

```
# Check total number of patients is still correct
total = ar_merged.drop_duplicates(subset=["icb_ons_code"])["total_patients"].sum
print(f"Total patients: {total:,.0f}")
```

```
Total patients: 55,468,820
```

In [20]:
```
# Correct for missing patients
missing_codes = {'E38000026', 'E38000250', 'E38000182', 'E38000229', 'E38000220'

missing_patients = (
    nhs_stats[(nhs_stats["SEX"] == "ALL") & (nhs_stats["AGE"] == "ALL") & (nhs_s
    .groupby("ONS_CCG_CODE")["NUMBER_OF_PATIENTS"]
    .sum()
)
print(missing_patients)

# Find the codes for the missing patients
print(nhs_stats[nhs_stats["ONS_CCG_CODE"].isin(missing_codes)][["ONS_CCG_CODE",

# Find the parent codes for the missing sub-ICBs
ccg_codes = ['01Y', '06H', '15E', '15M', '78H', 'D2P2L']

print(ad[ad["sub_icb_location_code"].isin(ccg_codes)]
    [["sub_icb_location_code", "sub_icb_location_ons_code", "icb_ons_code"]]
    .drop_duplicates())

# Use manual mapping to fill missing data
manual_mapping = {
    'E38000182': 'E54000057',
    'E38000026': 'E54000056',
    'E38000220': 'E54000055',
    'E38000229': 'E54000058',
    'E38000242': 'E54000059',
    'E38000250': 'E54000062',
}

# Get the missing patients with their ICB mapped
missing_df = (
    nhs_stats[(nhs_stats["SEX"] == "ALL") & (nhs_stats["AGE"] == "ALL") & (nhs_s
    .groupby("ONS_CCG_CODE")["NUMBER_OF_PATIENTS"]
    .sum()
    .reset_index()
    .assign(icb_ons_code=lambda x: x["ONS_CCG_CODE"].map(manual_mapping))
    .groupby("icb_ons_code")["NUMBER_OF_PATIENTS"]
    .sum()
    .reset_index()
    .rename(columns={"NUMBER_OF_PATIENTS": "total_patients"})
)

# Combine into patients_by_icb and recheck patient count
patients_by_icb = (
    pd.concat([patients_by_icb, missing_df])
    .groupby("icb_ons_code")["total_patients"]
    .sum()
    .reset_index()
)

print(f"Total patients: {patients_by_icb['total_patients'].sum():,.0f}")
```

```
ar_merged = ar_filtered.merge(patients_by_icb, on="icb_ons_code", how="left")

# Verify - check for any remaining nulls or zeros
print(ar_merged[ar_merged["total_patients"].isna() | (ar_merged["total_patients"
print(f"\nTotal patients check: {ar_merged.drop_duplicates(subset='icb_ons_code'
```

```
ONS_CCG_CODE
E38000026    1022493
E38000182     253133
E38000220    1348610
E38000229    1074187
E38000242     798944
E38000250    1503075
Name: NUMBER_OF_PATIENTS, dtype: int64
     ONS_CCG_CODE CCG_CODE
641     E38000182      01Y
1472    E38000026      06H
2596    E38000220      15E
2846    E38000229      15M
4209    E38000242      78H
5551    E38000250    D2P2L
       sub_icb_location_code sub_icb_location_ons_code icb_ons_code
25082                    01Y                 E38000263    E54000057
58539                    06H                 E38000260    E54000056
88096                    15E                 E38000258    E54000055
91055                    15M                 E38000261    E54000058
109026                   78H                 E38000262    E54000059
127931                 D2P2L                 E38000259    E54000062
Total patients: 61,469,262
Empty DataFrame
Columns: [icb_ons_code, appointment_month, appointment_status, hcp_type, appointm
ent_mode, time_between_book_and_appointment, count_of_appointments, total_patient
s]
Index: []

Total patients check: 61,469,262
```

In [21]:
```
#create final df for further analysis
ar = ar_merged
ar.head(3)
ar.to_csv('cleaned_ar.csv', index=False)
```

In [ ]:
```

```

Data checking/cleaning/preparation of national_categories

In [22]:
```
# Import and sense-check the national_categories.xlsx data set as nc.
nc_raw = pd.read_excel("national_categories.xlsx")

# View the DataFrame.
print(nc_raw.shape)
nc_raw.head(3)
```

```
(817394, 8)
```

Out[22]:

	appointment_date	icb_ons_code	sub_icb_location_name	service_setting	context_type
0	2021-08-02	E54000050	NHS North East and North Cumbria ICB - 00L	Primary Care Network	Care Related Encounter
1	2021-08-02	E54000050	NHS North East and North Cumbria ICB - 00L	Other	Care Related Encounter
2	2021-08-02	E54000050	NHS North East and North Cumbria ICB - 00L	General Practice	Care Related Encounter

In [23]:
```
# Determine whether there are missing values.
nc_raw_na = nc_raw[nc_raw.isna().any(axis=1)]
nc_raw_na.shape
```

Out[23]:
```
(0, 8)
```

In [24]:
```
# Determine whether there are any duplicate values.
nc_raw_duplicate_rows = nc_raw[nc_raw.duplicated(keep=False)]

# Counts only the extra copies (ignores the first occurrence)
nc_raw_duplicate_count = nc_raw.duplicated().sum()
print(f"Total duplicate rows: {nc_raw_duplicate_count}")

# Sort by all columns to group identical rows together
nc_raw_ordered_duplicates = nc_raw_duplicate_rows.sort_values(by=nc_raw.columns.

# Display the result
nc_raw_ordered_duplicates
```

```
Total duplicate rows: 0
```

Out[24]:

	appointment_date	icb_ons_code	sub_icb_location_name	service_setting	context_type

In [25]:
```
# Determine the metadata of the data set.
nc_raw.dtypes
```

Out[25]:
```
appointment_date         datetime64[ns]
icb_ons_code                     object
sub_icb_location_name            object
service_setting                  object
context_type                     object
national_category                object
count_of_appointments             int64
appointment_month                object
dtype: object
```

In [26]:
```
# Determine the minimum and maximum dates in the nc DataFrame.
# Use appropriate docstrings.
nc_raw_min_date = nc_raw['appointment_date'].min()
```

```
nc_raw_max_date = nc_raw['appointment_date'].max()

print(f"Earliest date: {nc_raw_min_date}")
print(f"Latest date: {nc_raw_max_date}")
```

```
Earliest date: 2021-08-01 00:00:00
Latest date: 2022-06-30 00:00:00
```

In [27]:
```
# merge population data
nc_merged = nc_raw.merge(patients_by_icb, on="icb_ons_code", how="left")

# Verify - check for any remaining nulls or zeros
print(nc_merged[nc_merged["total_patients"].isna() | (nc_merged["total_patients"
print(f"\nTotal patients check: {nc_merged.drop_duplicates(subset='icb_ons_code'
```

```
Empty DataFrame
Columns: [appointment_date, icb_ons_code, sub_icb_location_name, service_setting,
context_type, national_category, count_of_appointments, appointment_month, total_
patients]
Index: []

Total patients check: 61,469,262
```

In [28]:
```
# create a final df for further analysis
nc = nc_merged
nc.to_csv('cleaned_nc.csv', index=False)
nc.head(3)
```

Out[28]:

	appointment_date	icb_ons_code	sub_icb_location_name	service_setting	context_type
0	2021-08-02	E54000050	NHS North East and North Cumbria ICB - 00L	Primary Care Network	Care Relatec Encounte
1	2021-08-02	E54000050	NHS North East and North Cumbria ICB - 00L	Other	Care Relatec Encounte
2	2021-08-02	E54000050	NHS North East and North Cumbria ICB - 00L	General Practice	Care Relatec Encounte

In [ ]:
```

```

Data checking/cleaning/preparation of tweets.csv

In [29]:
```
# Import and sense-check the national_categories.xlsx data set as nc.
tw = pd.read_csv("tweets.csv")

# View the DataFrame.
print(tw.shape)
tw.head(3)
```

```
(1174, 10)
```

Out[29]:

	tweet_id	tweet_full_text	tweet_entities	tweet_entities_hashtags	
0	1567629223795527681	As Arkansas' first Comprehensive Stroke Certif...	{'hashtags': [{'text': 'Healthcare', 'indices'...	#Healthcare	{'is
1	1567582846612553728	RT @AndreaGrammer: Work-life balance is at the...	{'hashtags': [{'text': 'PremiseHealth', 'indic...	#PremiseHealth, #hiring	{'is
2	1567582787070304256	RT @OntarioGreens: $10 billion can go a long w...	{'hashtags': [{'text': 'Healthcare', 'indices'...	#Healthcare	{'is

In [30]:
```
# Determine whether there are missing values.
tw_na1 = tw[tw.isna().any(axis=1)]
tw_na2 = tw['tweet_entities_hashtags'].isnull().sum()
print(tw_na1.shape, tw_na2)
```

```
(167, 10) 167
```

In [31]:
```
# Determine whether there are any duplicate values.
tw_duplicate_rows = tw[tw.duplicated(keep=False)]

# Counts only the extra copies (ignores the first occurrence)
tw_duplicate_count = tw.duplicated().sum()
print(f"Total duplicate rows: {tw_duplicate_count}")

# Sort by all columns to group identical rows together
tw_ordered_duplicates = tw_duplicate_rows.sort_values(by=tw.columns.tolist())

# Display the result
tw_ordered_duplicates
```

```
Total duplicate rows: 0
```

Out[31]:

	tweet_id	tweet_full_text	tweet_entities	tweet_entities_hashtags	tweet_metadata	twe

In [32]:
```
#removing unneeded columns
tw_cleaned = tw[["tweet_id","tweet_full_text","tweet_entities_hashtags","tweet_r
                 "tweet_favorite_count"]]
print(tw_cleaned.shape)
tw_cleaned.head(5)
```

```
(1174, 5)
```

Out[32]:

	tweet_id	tweet_full_text	tweet_entities_hashtags	tweet_retweet_coun
0	1567629223795527681	As Arkansas' first Comprehensive Stroke Certif...	#Healthcare	
1	1567582846612553728	RT @AndreaGrammer: Work-life balance is at the...	#PremiseHealth, #hiring	
2	1567582787070304256	RT @OntarioGreens: $10 billion can go a long w...	#Healthcare	3
3	1567582767625428992	RT @modrnhealthcr: 🚨 #NEW: 🚨 Insurance companies...	#NEW	
4	1567582720460570625	ICYMI: Our recent blogs on Cybersecurity in Ac...	#blogs, #digitaltransformation, #cybersecurity...	

In [33]:
```
# Determine the metadata of the data set.
tw_cleaned.dtypes
```

Out[33]:
```
tweet_id                  int64
tweet_full_text          object
tweet_entities_hashtags  object
tweet_retweet_count       int64
tweet_favorite_count      int64
dtype: object
```

In [34]:
```
# create df for further analysis
tw = tw_cleaned
tw.to_csv('cleaned_tw.csv', index=False)
tw.head(3)
```

Out[34]:

	tweet_id	tweet_full_text	tweet_entities_hashtags	tweet_retweet_coun
0	1567629223795527681	As Arkansas' first Comprehensive Stroke Certif...	#Healthcare	
1	1567582846612553728	RT @AndreaGrammer: Work-life balance is at the...	#PremiseHealth, #hiring	
2	1567582787070304256	RT @OntarioGreens: $10 billion can go a long w...	#Healthcare	3

Data Exploration - Actual Duration File

Actual Duration File

Question 1: How many locations are there in the data set?

In [35]:
```
# Determine the number of locations.
sub_icb_count = ad['sub_icb_location_ons_code'].nunique()
icb_count = ad['icb_ons_code'].nunique()
region_count = ad['region_ons_code'].nunique()
print(f"There are {region_count} regions, made up of {icb_count} icbs and {sub_i
```

```
There are 7 regions, made up of 42 icbs and 106 sub icbs
```

Question 2: What are the five locations with the highest number of appointments?

In [36]:
```
# Aggregate by Sub-ICB
grouped = ad.groupby(['sub_icb_location_code', 'sub_icb_location_name']).agg(
    count_of_appointments=('count_of_appointments', 'sum')
    ).reset_index()

# Top 10 by Count of Appointments
top10_count = (
    grouped
    .nlargest(10, 'count_of_appointments')
    [['sub_icb_location_code', 'sub_icb_location_name', 'count_of_appointments']]
    .reset_index(drop=True)
)
top10_count.index += 1

print("=" * 70)
print("LIST 1 — TOP 10 SUB-ICBs BY COUNT OF APPOINTMENTS")
print("=" * 70)
print(top10_count.to_string())
```

```
print()
```

```
======================================================================
LIST 1 — TOP 10 SUB-ICBs BY COUNT OF APPOINTMENTS
======================================================================
   sub_icb_location_code                          sub_icb_location_name  count_of_a
ppointments
1                  W2U3Z            NHS North West London ICB - W2U3Z
6976986
2                  A3A8R            NHS North East London ICB - A3A8R
5341883
3                    91Q                NHS Kent and Medway ICB - 91Q
5209641
4                  D9Y0V  NHS Hampshire and Isle Of Wight ICB - D9Y0V
4712737
5                    72Q              NHS South East London ICB - 72Q
4360079
6                    15N                          NHS Devon ICB - 15N
4255338
7                    36L              NHS South West London ICB - 36L
4014321
8                  D2P2L                NHS Black Country ICB - D2P2L
3901431
9                    93C           NHS North Central London ICB - 93C
3795250
10                   15E        NHS Birmingham and Solihull ICB - 15E
3600087
```

Question 3: Consider fair share of appointments - how does this change?

In [37]:
```
# Create a ratio for appointments based on patient population

# Key parameters
DAILY_NATIONAL_THRESHOLD = 1_200_000
ad['appointment_date'] = pd.to_datetime(ad['appointment_date'], format='mixed',
num_days = ad['appointment_date'].nunique()
total_national_appointments = DAILY_NATIONAL_THRESHOLD * num_days  # 1.2m x 212

# Aggregate by sub-ICB
grouped = ad.groupby(['sub_icb_location_code', 'sub_icb_location_name']).agg(
    actual_appointments=('count_of_appointments', 'sum'),
    population=('total_patients', 'first')
).reset_index()

total_population = grouped['population'].sum()  # 61,469,262

# Fair share calculation
# Step 1: each sub-ICB's share of the national population
grouped['population_share'] = grouped['population'] / total_population

# Step 2: expected appointments based on population share over the period
grouped['fair_share_appointments'] = (grouped['population_share'] * total_nation

# Step 3: actual vs fair share ratio (1.0 = exactly as expected)
grouped['fair_share_ratio'] = (grouped['actual_appointments'] / grouped['fair_sh

#  Print Summary stats
print(f"Dataset spans:                    {num_days} days")
```

```
print(f"National threshold over period:   {total_national_appointments:,}")
print(f"Total population:                 {total_population:,.0f}")
print(f"Total actual appointments:        {grouped['actual_appointments'].sum():
print()

# --- Top 10 by fair share ratio ---
top10 = grouped.nlargest(10, 'fair_share_ratio').reset_index(drop=True)
top10.index += 1
print("=" * 80)
print("TOP 10 — HIGHEST FAIR SHARE RATIO")
print("=" * 80)
print(top10[['sub_icb_location_code', 'sub_icb_location_name',
             'population', 'actual_appointments',
             'fair_share_appointments', 'fair_share_ratio']].to_string())
print()

# --- Full table sorted by ratio ---
result = grouped.sort_values('fair_share_ratio', ascending=False).reset_index(dr
result.index += 1
print("=" * 80)
print("ALL SUB-ICBs — SORTED BY FAIR SHARE RATIO (high to low)")
print("=" * 80)
print(result[['sub_icb_location_code', 'sub_icb_location_name',
              'population', 'actual_appointments',
              'fair_share_appointments', 'fair_share_ratio']].to_string())
```

```
Dataset spans:                    212 days
National threshold over period:   254,400,000
Total population:                 61,469,262
Total actual appointments:        167,980,692

================================================================================
TOP 10 — HIGHEST FAIR SHARE RATIO
================================================================================
   sub_icb_location_code                          sub_icb_location_name  populat
ion  actual_appointments  fair_share_appointments  fair_share_ratio
1                    03H        NHS Humber and North Yorkshire ICB - 03H    17153
6.0               627632                   709928              0.88
2                    36J                  NHS West Yorkshire ICB - 36J    64803
0.0              2348208                  2681972              0.88
3                    84H      NHS North East and North Cumbria ICB - 84H    56103
9.0              2035860                  2321946              0.88
4                    11N  NHS Cornwall and The Isles Of Scilly ICB - 11N    59724
5.0              2097913                  2471790              0.85
5                    12F           NHS Cheshire and Merseyside ICB - 12F    33934
2.0              1151537                  1404419              0.82
6                    01H      NHS North East and North Cumbria ICB - 01H    32901
6.0              1098237                  1361683              0.81
7                    15N                            NHS Devon ICB - 15N   126991
5.0              4255338                  5255739              0.81
8                    00L      NHS North East and North Cumbria ICB - 00L    33305
0.0              1096360                  1378379              0.80
9                    03L                  NHS South Yorkshire ICB - 03L    26659
4.0               885621                  1103340              0.80
10                   18C  NHS Herefordshire and Worcestershire ICB - 18C    81435
6.0              2696703                  3370338              0.80

================================================================================
ALL SUB-ICBs — SORTED BY FAIR SHARE RATIO (high to low)
================================================================================
    sub_icb_location_code                                              sub_icb_loc
ation_name  population  actual_appointments  fair_share_appointments  fair_share_
ratio
1                    03H                          NHS Humber and North Yorkshire
ICB - 03H    171536.0              627632                   709928
0.88
2                    84H                       NHS North East and North Cumbria
ICB - 84H    561039.0             2035860                  2321946
0.88
3                    36J                                  NHS West Yorkshire
ICB - 36J    648030.0             2348208                  2681972
0.88
4                    11N                    NHS Cornwall and The Isles Of Scilly
ICB - 11N    597245.0             2097913                  2471790
0.85
5                    12F                             NHS Cheshire and Merseyside
ICB - 12F    339342.0             1151537                  1404419
0.82
6                    15N                                                NHS Devon
ICB - 15N   1269915.0             4255338                  5255739
0.81
7                    01H                       NHS North East and North Cumbria
ICB - 01H    329016.0             1098237                  1361683
0.81
8                    03L                                  NHS South Yorkshire
ICB - 03L    266594.0              885621                  1103340
```

```
0.80
9                    00L                          NHS North East and North Cumbria
ICB - 00L    333050.0             1096360                  1378379
0.80
10                   18C                    NHS Herefordshire and Worcestershire
ICB - 18C    814356.0             2696703                  3370338
0.80
11                   42D                            NHS Humber and North Yorkshire
ICB - 42D    436201.0             1446056                  1805285
0.80
12                   03R                                     NHS West Yorkshire
ICB - 03R    387856.0             1266542                  1605202
0.79
13                   26A                                NHS Norfolk and Waveney
ICB - 26A   1081228.0             3490647                  4474828
0.78
14                   03W               NHS Leicester Leicestershire and Rutland
ICB - 03W    344852.0             1108701                  1427223
0.78
15                   02M                          NHS Lancashire and South Cumbria
ICB - 02M    182369.0              576099                   754762
0.76
16                   11M                                      NHS Gloucestershire
ICB - 11M    673541.0             2127909                  2787553
0.76
17                   04V               NHS Leicester Leicestershire and Rutland
ICB - 04V    408253.0             1281775                  1689618
0.76
18                   03K                            NHS Humber and North Yorkshire
ICB - 03K    183593.0              577527                   759828
0.76
19                   15M                                NHS Derby and Derbyshire
ICB - 15M   1074187.0             3295790                  4445688
0.74
20                   11J                                                 NHS Dorset
ICB - 11J    819004.0             2513652                  3389574
0.74
21                   92G  NHS Bath and North East Somerset Swindon and Wiltshire
ICB - 92G    978455.0             3012568                  4049487
0.74
22                   02T                                     NHS West Yorkshire
ICB - 02T    222304.0              676331                   920039
0.74
23                   02Q                         NHS Nottingham and Nottinghamshire
ICB - 02Q    121699.0              371555                   503670
0.74
24                 X2C4Y                                     NHS West Yorkshire I
CB - X2C4Y    449573.0              1374398                  1860627
0.74
25                   97R                                                 NHS Sussex
ICB - 97R    566568.0             1711061                  2344829
0.73
26                   00R                          NHS Lancashire and South Cumbria
ICB - 00R    175989.0              532424                   728358
0.73
27                   71E                                      NHS Lincolnshire
ICB - 71E    808124.0             2457468                  3344546
0.73
28                   16C                          NHS North East and North Cumbria
ICB - 16C    716200.0             2139714                  2964104
```

```
0.72
29                    06T                        NHS Suffolk and North East Essex
ICB - 06T    365026.0              1076051                   1510716
0.71
30                    07K                        NHS Suffolk and North East Essex
ICB - 07K    261825.0               767971                   1083603
0.71
31                    06L                        NHS Suffolk and North East Essex
ICB - 06L    418775.0              1215736                   1733165
0.70
32                    01W                                 NHS Greater Manchester
ICB - 01W    320079.0               929497                   1324696
0.70
33                    11X                                           NHS Somerset
ICB - 11X    593933.0              1722566                   2458083
0.70
34                    15F                                      NHS West Yorkshire
ICB - 15F    908914.0              2625961                   3761680
0.70
35                    52R                       NHS Nottingham and Nottinghamshire
ICB - 52R   1117719.0              3199202                   4625852
0.69
36                    78H                                    NHS Northamptonshire
ICB - 78H    798944.0              2276102                   3306553
0.69
37                    06H                      NHS Cambridgeshire and Peterborough
ICB - 06H   1022493.0              2937987                   4231745
0.69
38                    03N                                      NHS South Yorkshire
ICB - 03N    619466.0              1774620                   2563755
0.69
39                    04C               NHS Leicester Leicestershire and Rutland
ICB - 04C    425863.0              1219839                   1762500
0.69
40                    02Y                           NHS Humber and North Yorkshire
ICB - 02Y    307774.0               862258                   1273770
0.68
41                  M2L0M                      NHS Shropshire Telford and Wrekin I
CB - M2L0M    518272.0              1459819                    2144948
0.68
42                    02X                                      NHS South Yorkshire
ICB - 02X    327561.0               915790                   1355662
0.68
43                    01V                                NHS Cheshire and Merseyside
ICB - 01V    126530.0               358060                    523664
0.68
44                    99C                        NHS North East and North Cumbria
ICB - 99C    224094.0               631499                    927448
0.68
45                    70F                                             NHS Sussex
ICB - 70F    921366.0              2587912                   3813215
0.68
46                    00T                                 NHS Greater Manchester
ICB - 00T    319373.0               884380                   1321774
0.67
47                    01K                        NHS Lancashire and South Cumbria
ICB - 01K    353699.0               983389                   1463838
0.67
48                    07G                                NHS Mid and South Essex
ICB - 07G    184224.0               512496                    762439
```

```
0.67
49                  D9Y0V                        NHS Hampshire and Isle Of Wight I
CB - D9Y0V   1687493.0              4712737                  6983949
0.67
50                    02P                                   NHS South Yorkshire
ICB - 02P    265170.0               724617                  1097447
0.66
51                    10Q      NHS Buckinghamshire Oxfordshire and Berkshire West
ICB - 10Q    792917.0              2153717                  3281609
0.66
52                    15E                             NHS Birmingham and Solihull
ICB - 15E   1348610.0              3600087                  5581430
0.65
53                    02G                          NHS Lancashire and South Cumbria
ICB - 02G    115055.0               311713                   476173
0.65
54                    13T                          NHS North East and North Cumbria
ICB - 13T    538215.0              1439487                  2227486
0.65
55                    01J                             NHS Cheshire and Merseyside
ICB - 01J    169846.0               455569                   702934
0.65
56                    02E                             NHS Cheshire and Merseyside
ICB - 02E    223195.0               603158                   923727
0.65
57                    04Y                    NHS Staffordshire and Stoke-on-Trent
ICB - 04Y    135943.0               367587                   562621
0.65
58                    91Q                                    NHS Kent and Medway
ICB - 91Q   1958865.0              5209641                  8107064
0.64
59                    06Q                                NHS Mid and South Essex
ICB - 06Q    404251.0              1072222                  1673055
0.64
60                  D4U1Y                                           NHS Frimley I
CB - D4U1Y    809260.0              2155885                  3349247
0.64
61                    05D                    NHS Staffordshire and Stoke-on-Trent
ICB - 05D    147758.0               393144                   611519
0.64
62                    27D                             NHS Cheshire and Merseyside
ICB - 27D    795167.0              2122585                  3290921
0.64
63                    05Q                    NHS Staffordshire and Stoke-on-Trent
ICB - 05Q    217805.0               577208                   901420
0.64
64                    99E                                NHS Mid and South Essex
ICB - 99E    285327.0               758049                  1180870
0.64
65                    99A                             NHS Cheshire and Merseyside
ICB - 99A    564635.0              1485630                  2336829
0.64
66                    15C    NHS Bristol North Somerset and South Gloucestershire
ICB - 15C   1057251.0              2756491                  4375596
0.63
67                    00P                          NHS North East and North Cumbria
ICB - 00P    285825.0               749173                  1182931
0.63
68                    05V                    NHS Staffordshire and Stoke-on-Trent
ICB - 05V    151851.0               394531                   628459
```

```
0.63
69                    15A      NHS Buckinghamshire Oxfordshire and Berkshire West
ICB - 15A    567247.0              1489285                   2347639
0.63
70                    00Q                          NHS Lancashire and South Cumbria
ICB - 00Q    180958.0               473422                    748923
0.63
71                    01Y                                  NHS Greater Manchester
ICB - 01Y    253133.0               664830                   1047630
0.63
72                  B2M3M                           NHS Coventry and Warwickshire I
CB - B2M3M   1048561.0              2751699                   4339631
0.63
73                  D2P2L                                    NHS Black Country I
CB - D2P2L   1503075.0              3901431                   6220707
0.63
74                  M1J4Y             NHS Bedfordshire Luton and Milton Keynes I
CB - M1J4Y   1072162.0              2791385                   4437307
0.63
75                    03Q                           NHS Humber and North Yorkshire
ICB - 03Q    368694.0               961867                   1525897
0.63
76                    05G                    NHS Staffordshire and Stoke-on-Trent
ICB - 05G    218149.0               557573                    902843
0.62
77                    01F                               NHS Cheshire and Merseyside
ICB - 01F    135183.0               348396                    559476
0.62
78                    06K                          NHS Hertfordshire and West Essex
ICB - 06K    616463.0              1579395                   2551327
0.62
79                    07H                          NHS Hertfordshire and West Essex
ICB - 07H    324282.0               836025                   1342091
0.62
80                    03F                             NHS Humber and North Yorkshire
ICB - 03F    306590.0               773239                   1268870
0.61
81                    00X                          NHS Lancashire and South Cumbria
ICB - 00X    187827.0               473443                    777351
0.61
82                    01E                          NHS Lancashire and South Cumbria
ICB - 01E    220319.0               552951                    911824
0.61
83                  W2U3Z                                 NHS North West London I
CB - W2U3Z   2751843.0              6976986                  11388926
0.61
84                    92A                                    NHS Surrey Heartlands
ICB - 92A   1122836.0              2789898                   4647030
0.60
85                    00N                          NHS North East and North Cumbria
ICB - 00N    158713.0               394613                    656858
0.60
86                    01A                          NHS Lancashire and South Cumbria
ICB - 01A    394894.0               982794                   1634330
0.60
87                    01G                                  NHS Greater Manchester
ICB - 01G    299603.0               740590                   1239953
0.60
88                    14Y      NHS Buckinghamshire Oxfordshire and Berkshire West
ICB - 14Y    577367.0              1429206                   2389522
```

```
0.60
89                    99F                                NHS Mid and South Essex
ICB - 99F    186104.0              458133                   770220
0.59
90                    09D                                         NHS Sussex
ICB - 09D    329346.0              808508                  1363049
0.59
91                    06N                     NHS Hertfordshire and West Essex
ICB - 06N    668490.0             1611539                  2766649
0.58
92                    01X                          NHS Cheshire and Merseyside
ICB - 01X    200086.0              478518                   828087
0.58
93                    00Y                            NHS Greater Manchester
ICB - 00Y    262089.0              626289                  1084696
0.58
94                    14L                            NHS Greater Manchester
ICB - 14L    694270.0             1645845                  2873343
0.57
95                    01D                            NHS Greater Manchester
ICB - 01D    241955.0              573448                  1001368
0.57
96                    99G                                NHS Mid and South Essex
ICB - 99G    190375.0              439058                   787896
0.56
97                    10R                       NHS Hampshire and Isle Of Wight
ICB - 10R    230397.0              530103                   953533
0.56
98                    36L                            NHS South West London
ICB - 36L   1726248.0             4014321                  7144343
0.56
99                  A3A8R                            NHS North East London I
CB - A3A8R   2346009.0             5341883                  9709319
0.55
100                   02H                            NHS Greater Manchester
ICB - 02H    337650.0              757429                  1397416
0.54
101                   05W                  NHS Staffordshire and Stoke-on-Trent
ICB - 05W    297506.0              664341                  1231274
0.54
102                   93C                          NHS North Central London
ICB - 93C   1742487.0             3795250                  7211551
0.53
103                   01T                          NHS Cheshire and Merseyside
ICB - 01T    156997.0              340895                   649756
0.52
104                   72Q                            NHS South East London
ICB - 72Q   2047004.0             4360079                  8471841
0.51
105                   02A                            NHS Greater Manchester
ICB - 02A    245330.0              465791                  1015336
0.46
106                   00V                            NHS Greater Manchester
ICB - 00V    208532.0              362242                   863042
0.42
```

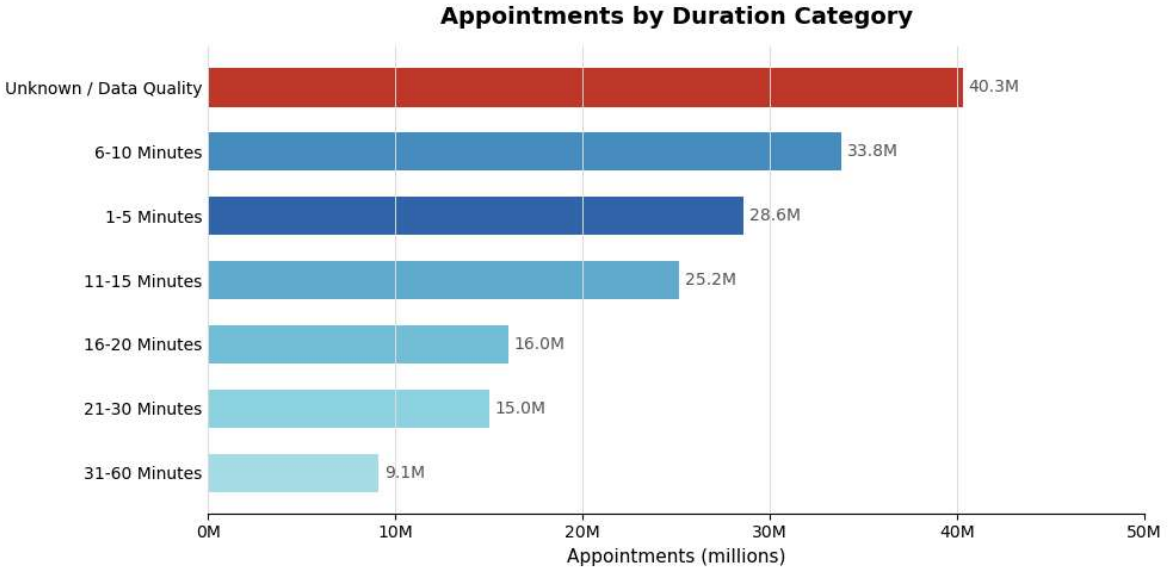
Question 4: What is the distribution of appointment durations?

In [38]:
```
# Examine appointment durations
# Aggregate data
```

```
duration_categories = (ad.groupby("actual_duration")["count_of_appointments"].su
    .sort_values(ascending=False))
duration_categories
```

Out[38]:
```
actual_duration
Unknown / Data Quality    40284086
6-10 Minutes              33800815
1-5 Minutes               28600865
11-15 Minutes             25160882
16-20 Minutes             16004247
21-30 Minutes             15026365
31-60 Minutes              9103432
Name: count_of_appointments, dtype: int64
```

In [39]:
```
# Create a bar chart to show this

# Fix Colours
colours = [
    "#c0392b",  # Unknown / Data Quality
    "#4a8fc2",  # 6-10 Minutes
    "#3266ad",  # 1-5 Minutes
    "#5faad0",  # 11-15 Minutes
    "#73c2da",  # 16-20 Minutes
    "#8dd5e1",  # 21-30 Minutes
    "#a8e0e8",  # 31-60 Minutes
]

# Plot Horizontal Bar Chart
fig, ax = plt.subplots(figsize=(10, 5))
bars = ax.barh(duration_categories.index, duration_categories.values / 1e6,
               color=colours, edgecolor="none", height=0.6)

# Value labels
for bar, val in zip(bars, duration_categories.values):
    ax.text(
        bar.get_width() + 0.3,
        bar.get_y() + bar.get_height() / 2,
        f"{val / 1e6:.1f}M",
        va="center", ha="left", fontsize=10, color="#555",
    )

ax.set_xlabel("Appointments (millions)", fontsize=11)
ax.set_title("Appointments by Duration Category", fontsize=14, fontweight="bold"
ax.xaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f"{x:.0f}M"))
ax.invert_yaxis()
ax.spines[["top", "right", "left"]].set_visible(False)
ax.tick_params(axis="y", length=0)
ax.set_xlim(0, 50)
ax.grid(axis="x", color="#e0e0e0", linewidth=0.7)

plt.tight_layout()
plt.savefig("appointments_by_duration.png", dpi=150, bbox_inches="tight")
plt.show()
```



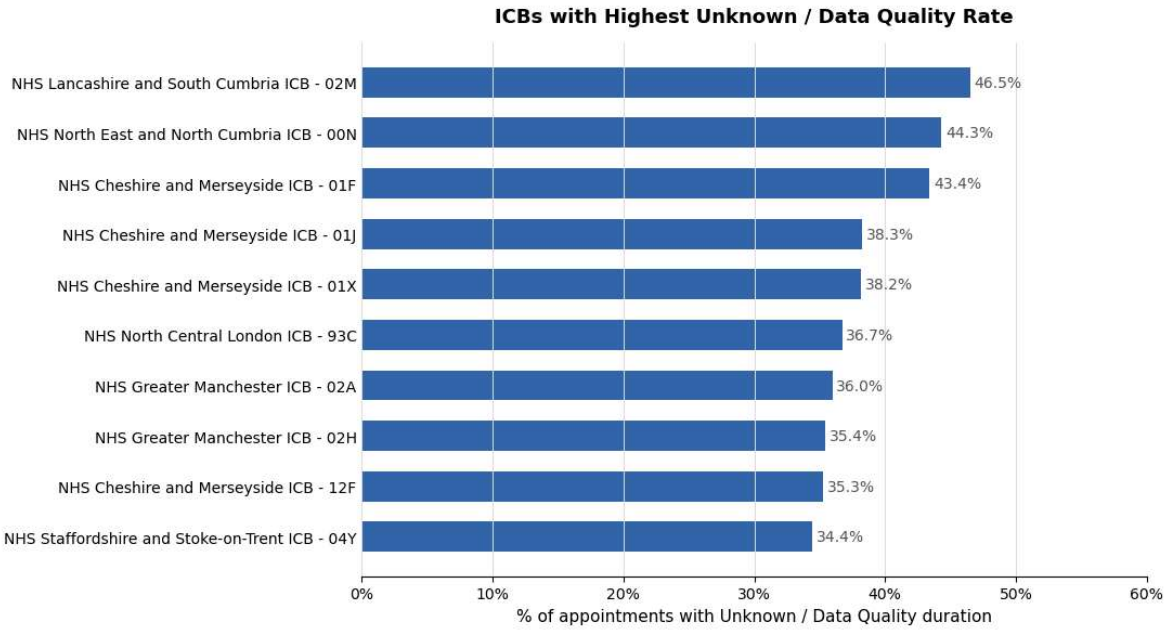
Question 5: Which are the worst ICBs for Unknown/Data Quality Issues?

In [40]:
```
worst_icbs = ad.groupby(["sub_icb_location_name", "actual_duration"])["count_of_
worst_icbs["total"] = worst_icbs.sum(axis=1)
worst_icbs["unknown_pct"] = worst_icbs["Unknown / Data Quality"] / worst_icbs["t

top10 = worst_icbs["unknown_pct"].sort_values(ascending=False).head(10)
top10
```

Out[40]:
```
sub_icb_location_name
NHS Lancashire and South Cumbria ICB - 02M        46.529329
NHS North East and North Cumbria ICB - 00N        44.327480
NHS Cheshire and Merseyside ICB - 01F             43.408650
NHS Cheshire and Merseyside ICB - 01J             38.250408
NHS Cheshire and Merseyside ICB - 01X             38.198772
NHS North Central London ICB - 93C                36.712022
NHS Greater Manchester ICB - 02A                  35.980515
NHS Greater Manchester ICB - 02H                  35.440021
NHS Cheshire and Merseyside ICB - 12F             35.268168
NHS Staffordshire and Stoke-on-Trent ICB - 04Y    34.430217
Name: unknown_pct, dtype: float64
```

In [41]:
```
# Plot these on a horizontal bar chart
fig, ax = plt.subplots(figsize=(11, 6))

bars = ax.barh(top10.index, top10.values, color="#3266ad", edgecolor="none", hei

for bar, val in zip(bars, top10.values):
    ax.text(
        bar.get_width() + 0.3,
        bar.get_y() + bar.get_height() / 2,
        f"{val:.1f}%",
        va="center", ha="left", fontsize=10, color="#555",
    )

ax.set_xlabel("% of appointments with Unknown / Data Quality duration", fontsize
ax.set_title("ICBs with Highest Unknown / Data Quality Rate", fontsize=13, fontw
ax.xaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f"{x:.0f}%"))
ax.invert_yaxis()
ax.spines[["top", "right", "left"]].set_visible(False)
ax.tick_params(axis="y", length=0)
```

```
ax.set_xlim(0, 60)
ax.grid(axis="x", color="#e0e0e0", linewidth=0.7)

plt.tight_layout()
plt.savefig("unknown_rate_by_icb.png", dpi=150, bbox_inches="tight")
plt.show()
```



Question 5: What do fair-share ratios look like on a daily basis?

In [42]:

```
# Constants
DAILY_NATIONAL_THRESHOLD = 1_200_000
total_population = 61_469_262

# Total population (fixed per sub-ICB, used to calculate population share)
pop_lookup = ad.groupby('sub_icb_location_code')['total_patients'].first()
total_population = pop_lookup.sum()  # 61,469,262

# Step 1: aggregate actual appointments by sub-ICB and date
# (summing across all duration bands for each sub-ICB/date combination)
daily = ad.groupby(['sub_icb_location_code', 'sub_icb_location_name', 'appointme
    'count_of_appointments'
].sum().reset_index()

# Step 2: add population and calculate daily fair share
daily['population'] = daily['sub_icb_location_code'].map(pop_lookup)
daily['daily_fair_share'] = (daily['population'] / total_population) * DAILY_NAT

# Step 3: calculate daily ratio
daily['daily_ratio'] = daily['count_of_appointments'] / daily['daily_fair_share'

# Step 4: count days where ratio exceeded 1.0 per sub-ICB
days_over = daily[daily['daily_ratio'] > 1.0].groupby(
    ['sub_icb_location_code', 'sub_icb_location_name']
).size().reset_index(name='days_over_1')

# Step 5: join back to get total days in range for each sub-ICB
total_days = daily.groupby(
    ['sub_icb_location_code', 'sub_icb_location_name']
)['appointment_date'].nunique().reset_index(name='total_days')
```

```
result = total_days.merge(days_over, on=['sub_icb_location_code', 'sub_icb_locat
result['days_over_1'] = result['days_over_1'].fillna(0).astype(int)
result['pct_days_over'] = (result['days_over_1'] / result['total_days'] * 100).r

# Sort by days over 1.0 descending
result = result.sort_values('days_over_1', ascending=False).reset_index(drop=Tru
result.index += 1

# Output
total_date_range = daily['appointment_date'].nunique()
print(f"Total days in dataset:     {total_date_range}")
print(f"Total population:          {total_population:,.0f}")
print(f"Daily national threshold:  {DAILY_NATIONAL_THRESHOLD:,}")
print()
print("=" * 85)
print("DAYS WHERE DAILY FAIR SHARE RATIO EXCEEDED 1.0 — BY SUB-ICB")
print("=" * 85)
print(result[['sub_icb_location_code', 'sub_icb_location_name',
              'total_days', 'days_over_1', 'pct_days_over']].to_string())
```

```
Total days in dataset:     212
Total population:          61,469,262
Daily national threshold:  1,200,000

================================================================================
====
DAYS WHERE DAILY FAIR SHARE RATIO EXCEEDED 1.0 — BY SUB-ICB
================================================================================
====
    sub_icb_location_code                                          sub_icb_loc
ation_name  total_days  days_over_1  pct_days_over
1                    36J                                  NHS West Yorkshire
ICB - 36J         185          142           76.8
2                    84H                      NHS North East and North Cumbria
ICB - 84H         212          142           67.0
3                    00L                      NHS North East and North Cumbria
ICB - 00L         178          141           79.2
4                    11N                    NHS Cornwall and The Isles Of Scilly
ICB - 11N         210          141           67.1
5                    03H                        NHS Humber and North Yorkshire
ICB - 03H         183          141           77.0
6                    18C                    NHS Herefordshire and Worcestershire
ICB - 18C         199          140           70.4
7                    42D                        NHS Humber and North Yorkshire
ICB - 42D         205          140           68.3
8                    15N                                           NHS Devon
ICB - 15N         212          139           65.6
9                    26A                                NHS Norfolk and Waveney
ICB - 26A         212          138           65.1
10                   12F                             NHS Cheshire and Merseyside
ICB - 12F         212          138           65.1
11                   03L                                  NHS South Yorkshire
ICB - 03L         153          137           89.5
12                   01H                      NHS North East and North Cumbria
ICB - 01H         191          136           71.2
13                   11M                                  NHS Gloucestershire
ICB - 11M         198          133           67.2
14                   04V                NHS Leicester Leicestershire and Rutland
ICB - 04V         205          133           64.9
15                   03R                                   NHS West Yorkshire
ICB - 03R         204          133           65.2
16                   03W                NHS Leicester Leicestershire and Rutland
ICB - 03W         206          131           63.6
17                   92G  NHS Bath and North East Somerset Swindon and Wiltshire
ICB - 92G         189          128           67.7
18                   02M                      NHS Lancashire and South Cumbria
ICB - 02M         168          128           76.2
19                   03K                        NHS Humber and North Yorkshire
ICB - 03K         202          126           62.4
20                   97R                                           NHS Sussex
ICB - 97R         212          126           59.4
21                   11J                                           NHS Dorset
ICB - 11J         212          126           59.4
22                   16C                      NHS North East and North Cumbria
ICB - 16C         212          122           57.5
23                   15M                              NHS Derby and Derbyshire
ICB - 15M         211          121           57.3
24                   71E                                  NHS Lincolnshire
ICB - 71E         210          121           57.6
25                   02T                                  NHS West Yorkshire
```

```
ICB - 02T         208          118           56.7
26                  X2C4Y                                  NHS West Yorkshire I
CB - X2C4Y         189           118           62.4
27                    02Q                    NHS Nottingham and Nottinghamshire
ICB - 02Q         179          114           63.7
28                    00R                        NHS Lancashire and South Cumbria
ICB - 00R         173          112           64.7
29                    07K                        NHS Suffolk and North East Essex
ICB - 07K         172          105           61.0
30                    06T                        NHS Suffolk and North East Essex
ICB - 06T         212          103           48.6
31                    06L                        NHS Suffolk and North East Essex
ICB - 06L         179           93           52.0
32                    03N                                 NHS South Yorkshire
ICB - 03N         180           90           50.0
33                    01W                              NHS Greater Manchester
ICB - 01W         176           89           50.6
34                    15F                                  NHS West Yorkshire
ICB - 15F         212           88           41.5
35                    11X                                       NHS Somerset
ICB - 11X         173           86           49.7
36                    01V                          NHS Cheshire and Merseyside
ICB - 01V         178           84           47.2
37                    52R                    NHS Nottingham and Nottinghamshire
ICB - 52R         203           83           40.9
38                    06H                     NHS Cambridgeshire and Peterborough
ICB - 06H         212           83           39.2
39                    78H                              NHS Northamptonshire
ICB - 78H         208           82           39.4
40                    04C                NHS Leicester Leicestershire and Rutland
ICB - 04C         196           78           39.8
41                    02Y                       NHS Humber and North Yorkshire
ICB - 02Y         197           76           38.6
42                    99C                        NHS North East and North Cumbria
ICB - 99C         180           68           37.8
43                    02X                                 NHS South Yorkshire
ICB - 02X         172           67           39.0
44                  M2L0M                     NHS Shropshire Telford and Wrekin I
CB - M2L0M         206            65           31.6
45                    70F                                       NHS Sussex
ICB - 70F         212           64           30.2
46                    00T                              NHS Greater Manchester
ICB - 00T         182           63           34.6
47                    02P                                 NHS South Yorkshire
ICB - 02P         149           61           40.9
48                    01K                        NHS Lancashire and South Cumbria
ICB - 01K         178           61           34.3
49                  D9Y0V                        NHS Hampshire and Isle Of Wight I
CB - D9Y0V         205            61           29.8
50                    07G                              NHS Mid and South Essex
ICB - 07G         200           59           29.5
51                    02E                          NHS Cheshire and Merseyside
ICB - 02E         176           51           29.0
52                    04Y                    NHS Staffordshire and Stoke-on-Trent
ICB - 04Y         193           51           26.4
53                    01J                          NHS Cheshire and Merseyside
ICB - 01J         158           49           31.0
54                    05D                    NHS Staffordshire and Stoke-on-Trent
ICB - 05D         157           46           29.3
55                    01Y                              NHS Greater Manchester
```

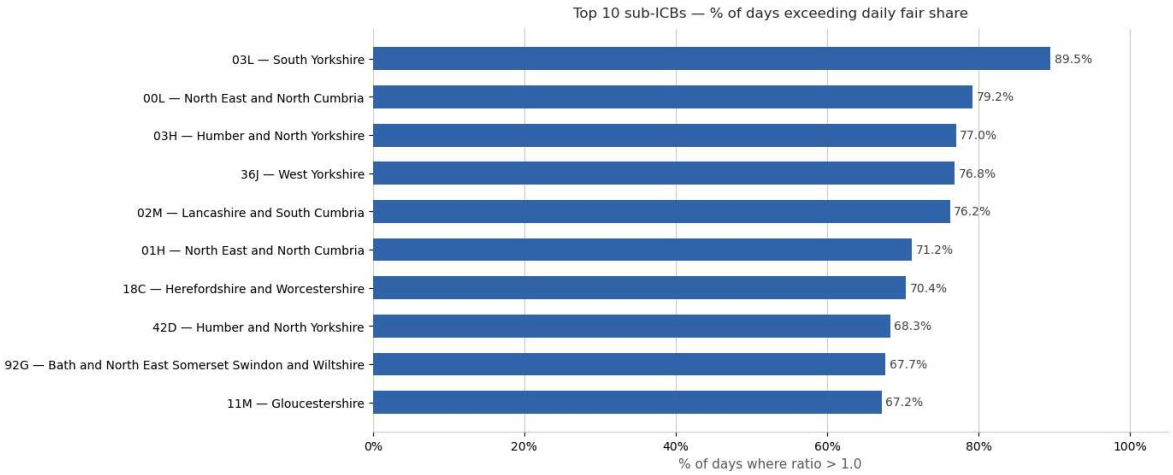
```
ICB - 01Y         202           43           21.3
56                    13T                       NHS North East and North Cumbria
ICB - 13T         178           39           21.9
57                    10Q      NHS Buckinghamshire Oxfordshire and Berkshire West
ICB - 10Q         183           39           21.3
58                    15E                            NHS Birmingham and Solihull
ICB - 15E         212           38           17.9
59                    02G                       NHS Lancashire and South Cumbria
ICB - 02G         167           38           22.8
60                    99A                            NHS Cheshire and Merseyside
ICB - 99A         177           37           20.9
61                  B2M3M                        NHS Coventry and Warwickshire I
CB - B2M3M         189           37           19.6
62                    05Q                     NHS Staffordshire and Stoke-on-Trent
ICB - 05Q         204           35           17.2
63                    06Q                                 NHS Mid and South Essex
ICB - 06Q         204           34           16.7
64                    27D                            NHS Cheshire and Merseyside
ICB - 27D         206           32           15.5
65                    00P                       NHS North East and North Cumbria
ICB - 00P         149           31           20.8
66                  D4U1Y                                        NHS Frimley I
CB - D4U1Y         211           30           14.2
67                    01F                            NHS Cheshire and Merseyside
ICB - 01F         146           30           20.5
68                    05G                     NHS Staffordshire and Stoke-on-Trent
ICB - 05G         210           29           13.8
69                    99E                                 NHS Mid and South Essex
ICB - 99E         179           29           16.2
70                    91Q                                     NHS Kent and Medway
ICB - 91Q         212           28           13.2
71                  D2P2L                                     NHS Black Country I
CB - D2P2L         212           25           11.8
72                    00Q                       NHS Lancashire and South Cumbria
ICB - 00Q         182           23           12.6
73                    03Q                          NHS Humber and North Yorkshire
ICB - 03Q         197           23           11.7
74                    15C    NHS Bristol North Somerset and South Gloucestershire
ICB - 15C         208           23           11.1
75                    15A      NHS Buckinghamshire Oxfordshire and Berkshire West
ICB - 15A         210           22           10.5
76                    05V                     NHS Staffordshire and Stoke-on-Trent
ICB - 05V         172           21           12.2
77                    07H                       NHS Hertfordshire and West Essex
ICB - 07H         193           19            9.8
78                    01E                       NHS Lancashire and South Cumbria
ICB - 01E         208           18            8.7
79                  M1J4Y              NHS Bedfordshire Luton and Milton Keynes I
CB - M1J4Y         212           18            8.5
80                    06K                       NHS Hertfordshire and West Essex
ICB - 06K         202           15            7.4
81                    14Y      NHS Buckinghamshire Oxfordshire and Berkshire West
ICB - 14Y         182           14            7.7
82                    92A                                   NHS Surrey Heartlands
ICB - 92A         211           13            6.2
83                    99F                                 NHS Mid and South Essex
ICB - 99F         201           12            6.0
84                    00X                       NHS Lancashire and South Cumbria
ICB - 00X         211            8            3.8
85                    00N                       NHS North East and North Cumbria
```

```
ICB - 00N         177            8            4.5
86                    03F                          NHS Humber and North Yorkshire
ICB - 03F         212            4            1.9
87                    01G                                NHS Greater Manchester
ICB - 01G         201            2            1.0
88                    01A                   NHS Lancashire and South Cumbria
ICB - 01A         173            2            1.2
89                  W2U3Z                            NHS North West London I
CB - W2U3Z         212            1            0.5
90                    06N                   NHS Hertfordshire and West Essex
ICB - 06N         212            1            0.5
91                    01X                         NHS Cheshire and Merseyside
ICB - 01X         194            1            0.5
92                    99G                            NHS Mid and South Essex
ICB - 99G         200            0            0.0
93                  A3A8R                            NHS North East London I
CB - A3A8R         212            0            0.0
94                    02H                           NHS Greater Manchester
ICB - 02H         174            0            0.0
95                    93C                       NHS North Central London
ICB - 93C         212            0            0.0
96                    72Q                            NHS South East London
ICB - 72Q         205            0            0.0
97                    00V                           NHS Greater Manchester
ICB - 00V         179            0            0.0
98                    36L                            NHS South West London
ICB - 36L         202            0            0.0
99                    00Y                           NHS Greater Manchester
ICB - 00Y         204            0            0.0
100                   01D                           NHS Greater Manchester
ICB - 01D         175            0            0.0
101                   01T                         NHS Cheshire and Merseyside
ICB - 01T         158            0            0.0
102                   10R                     NHS Hampshire and Isle Of Wight
ICB - 10R         185            0            0.0
103                   09D                                        NHS Sussex
ICB - 09D         212            0            0.0
104                   05W                   NHS Staffordshire and Stoke-on-Trent
ICB - 05W         190            0            0.0
105                   02A                           NHS Greater Manchester
ICB - 02A         199            0            0.0
106                   14L                           NHS Greater Manchester
ICB - 14L         212            0            0.0
```

In [43]:

```python
# Put that info in a chart

# Top 10 and never-exceeded
top10 = result.nlargest(10, 'pct_days_over').copy()
never = result[result['days_over_1'] == 0].copy()

# Shorten name for labels
def short_label(code, name):
    name = name.replace('NHS ', '').replace(' ICB', '')
    if ' - ' in name:
        name = name.split(' - ')[0]
    return f"{code} — {name}"

top10['label'] = top10.apply(lambda r: short_label(r['sub_icb_location_code'], r
never['label'] = never.apply(lambda r: short_label(r['sub_icb_location_code'], r
```

```
# Sort top10 so highest is at top of horizontal bar chart
top10 = top10.sort_values('pct_days_over', ascending=True)

# Figure layout
fig = plt.figure(figsize=(12, 10))
gs = GridSpec(2, 1, figure=fig, height_ratios=[3, 1], hspace=0.45)

# Top chart: top 10 bar chart
ax1 = fig.add_subplot(gs[0])

bars = ax1.barh(top10['label'], top10['pct_days_over'],
                color='#3266ad', height=0.6, zorder=3)

for bar, val in zip(bars, top10['pct_days_over']):
    ax1.text(bar.get_width() + 0.5, bar.get_y() + bar.get_height() / 2,
             f'{val:.1f}%', va='center', ha='left', fontsize=10, color='#444441'

ax1.set_xlim(0, 105)
ax1.set_xlabel('% of days where ratio > 1.0', fontsize=11, color='#5F5E5A')
ax1.set_title('Top 10 sub-ICBs — % of days exceeding daily fair share', fontsize
ax1.tick_params(axis='both', labelsize=10)
ax1.xaxis.set_major_formatter(plt.FuncFormatter(lambda x, _: f'{x:.0f}%'))
ax1.set_axisbelow(True)
ax1.grid(axis='x', color='#D3D1C7', linewidth=0.8, zorder=1)
ax1.spines[['top', 'right', 'left']].set_visible(False)
ax1.spines['bottom'].set_color('#D3D1C7')

# Bottom panel: never-exceeded list
ax2 = fig.add_subplot(gs[1])
ax2.axis('off')

ax2.set_title('Sub-ICBs that never exceeded their daily fair share (0 days)',
              fontsize=11, color='#2C2C2A', pad=8, loc='left')

never_labels = never.sort_values('sub_icb_location_code')['label'].tolist()
n = len(never_labels)
cols = 3
rows = -(-n // cols)  # ceiling division

for i, label in enumerate(never_labels):
    col = i % cols
    row = i // cols
    x = col / cols
    y = 1 - (row + 1) / (rows + 0.5)
    ax2.text(x, y, f'• {label}', transform=ax2.transAxes,
             fontsize=9.5, color='#5F5E5A', va='top')

plt.savefig('daily_fair_share_chart.png', dpi=150, bbox_inches='tight', facecolo
plt.show()
```



Sub-ICBs that never exceeded their daily fair share (0 days)

- 00V — Greater Manchester
- 00Y — Greater Manchester
- 01D — Greater Manchester
- 01T — Cheshire and Merseyside
- 02A — Greater Manchester
- 02H — Greater Manchester
- 05W — Staffordshire and Stoke-on-Trent
- 09D — Sussex
- 10R — Hampshire and Isle Of Wight
- 14L — Greater Manchester
- 36L — South West London
- 72Q — South East London
- 93C — North Central London
- 99G — Mid and South Essex
- A3A8R — North East London

Data Exploration - Appointments Regional File

Question 1: What do we know about missed appointments?

In [44]:
```
# count appointment status
status_counts = ar.groupby('appointment_status')['count_of_appointments'].sum().
status_counts.columns = ['appointment_status', 'total_appointments']
status_counts = status_counts.sort_values('total_appointments', ascending=False)
status_counts
```

Out[44]:

	appointment_status	total_appointments
0	Attended	270617423
1	DNA	13285796
2	Unknown	12037388

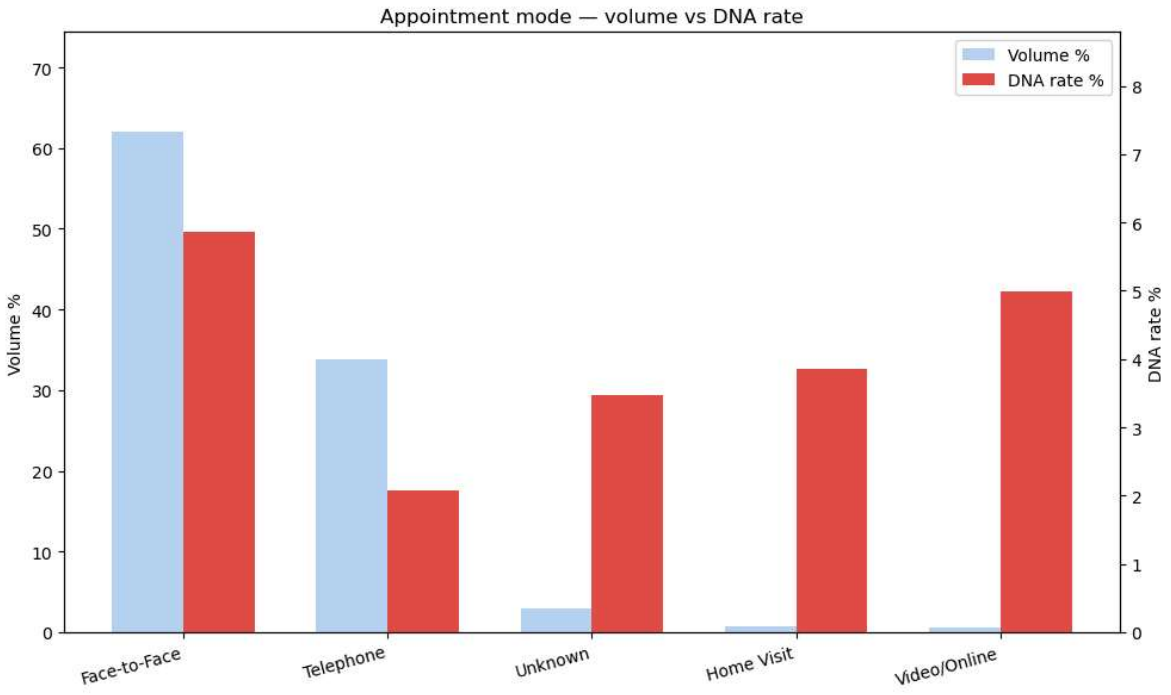
Question 2: How does that compare across appointment modes?

In [45]:
```
mode_total = ar.groupby('appointment_mode')['count_of_appointments'].sum()
mode_dna = ar[ar['appointment_status'] == 'DNA'].groupby('appointment_mode')['co

mode_summary = pd.DataFrame({
    'total_appointments': mode_total,
    'dna_appointments': mode_dna
}).fillna(0)

mode_summary['dna_rate_%'] = (mode_summary['dna_appointments'] / mode_summary['t
mode_summary['volume_%'] = (mode_summary['total_appointments'] / mode_summary['t

mode_summary = mode_summary.sort_values('total_appointments', ascending=False)
```

```
print(mode_summary)
```

appointment_mode	total_appointments	dna_appointments	dna_rate_%	volume_%
Face-to-Face	183457603	10747016	5.86	61.99
Telephone	100237680	2081545	2.08	33.87
Unknown	8729565	304109	3.48	2.95
Home Visit	1975204	76306	3.86	0.67
Video/Online	1540555	76820	4.99	0.52

In [46]:
```
# Turn this into a chart
fig, ax1 = plt.subplots(figsize=(10, 6))

modes = mode_summary.index.tolist()
x = np.arange(len(modes))
width = 0.35

bars1 = ax1.bar(x - width/2, mode_summary['volume_%'], width, label='Volume %',
ax1.set_ylabel('Volume %')
ax1.set_ylim(0, max(mode_summary['volume_%']) * 1.2)

ax2 = ax1.twinx()
bars2 = ax2.bar(x + width/2, mode_summary['dna_rate_%'], width, label='DNA rate
ax2.set_ylabel('DNA rate %')
ax2.set_ylim(0, max(mode_summary['dna_rate_%']) * 1.5)

ax1.set_xticks(x)
ax1.set_xticklabels(modes, rotation=15, ha='right')
ax1.set_title('Appointment mode — volume vs DNA rate')

lines1, labels1 = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()
ax1.legend(lines1 + lines2, labels1 + labels2, loc='upper right')

plt.tight_layout()
plt.show()
```

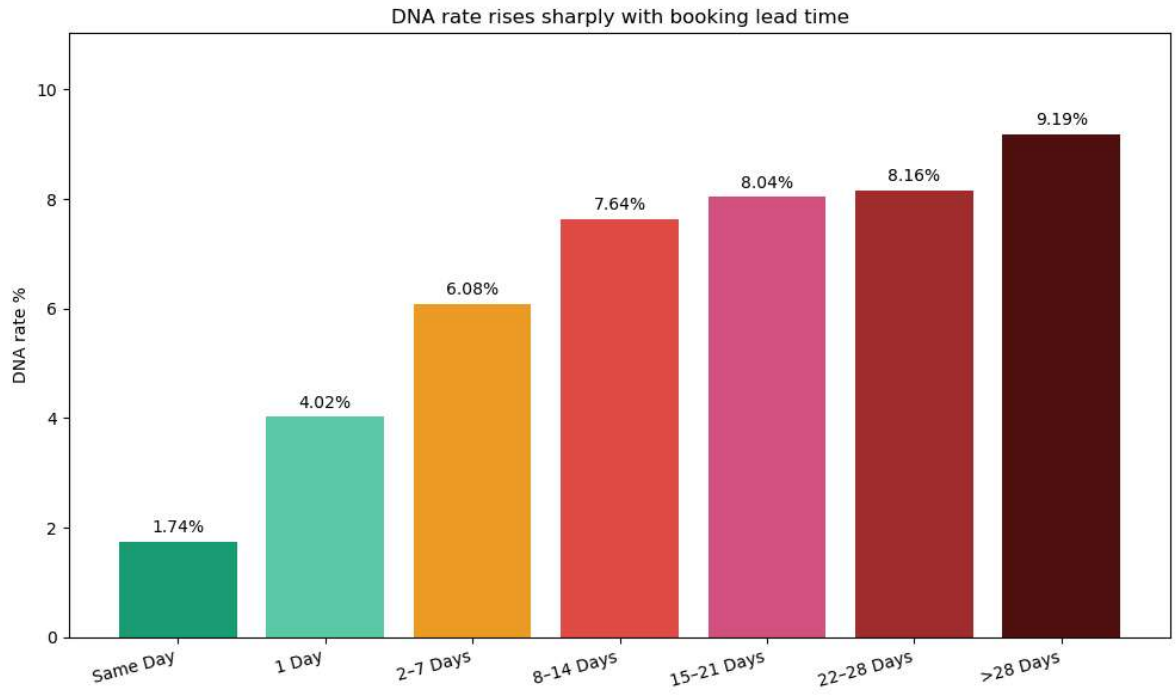


Question 3: How does that compare across time between booking and appointment?

In [47]:
```
lead_time_order = ['Same Day', '1 Day', '2 to 7 Days', '8  to 14 Days',
                   '15  to 21 Days', '22  to 28 Days', 'More than 28 Days']

lead_total = ar.groupby('time_between_book_and_appointment')['count_of_appointme
lead_dna = ar[ar['appointment_status'] == 'DNA'].groupby('time_between_book_and_

lead_summary = pd.DataFrame({
    'total_appointments': lead_total,
    'dna_appointments': lead_dna
}).fillna(0)

lead_summary['dna_rate_%'] = (lead_summary['dna_appointments'] / lead_summary['t
lead_summary = lead_summary.loc[lead_time_order]

print(lead_summary)
```

```
                                  total_appointments  dna_appointments  \
time_between_book_and_appointment
Same Day                                   130964160           2284169
1 Day                                       25854059           1039523
2 to 7 Days                                 60720229           3689732
8  to 14 Days                               37490409           2864860
15  to 21 Days                              19437514           1563712
22  to 28 Days                              11417519            931458
More than 28 Days                            9868508            907394

                                  dna_rate_%
time_between_book_and_appointment
Same Day                                1.74
1 Day                                   4.02
2 to 7 Days                             6.08
8  to 14 Days                           7.64
15  to 21 Days                          8.04
22  to 28 Days                          8.16
More than 28 Days                       9.19
```

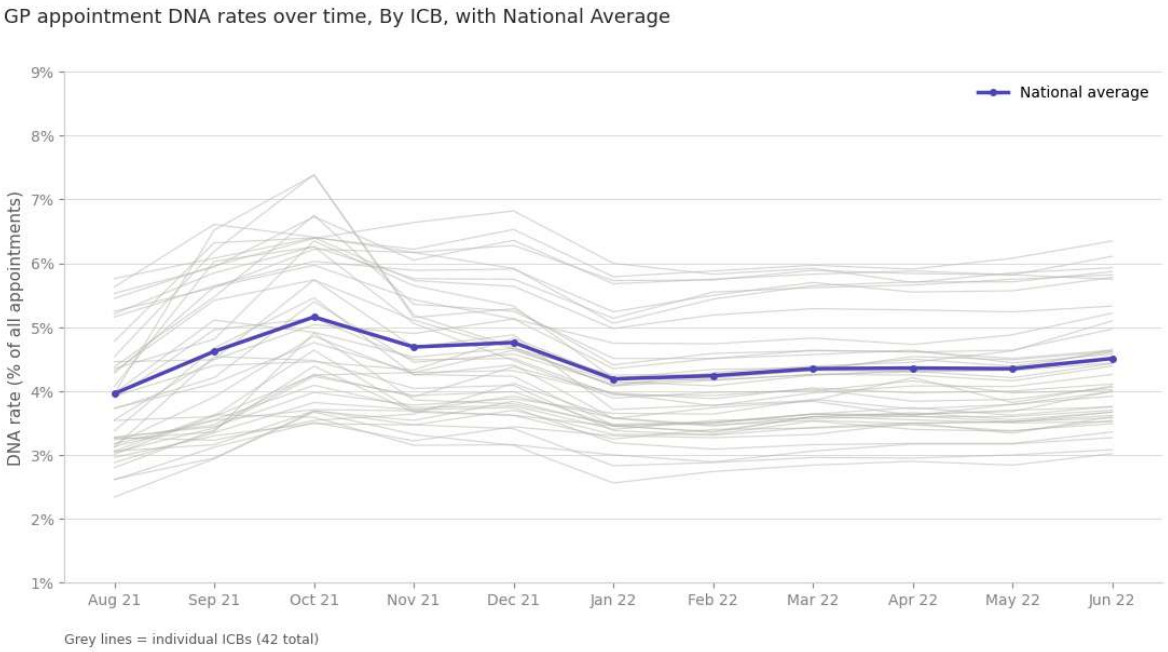
In [49]:
```
# Turn into a chart
colors = ['#1D9E75', '#5DCAA5', '#EF9F27', '#E24B4A', '#D4537E', '#A32D2D', '#50
labels = ['Same Day', '1 Day', '2–7 Days', '8–14 Days', '15–21 Days', '22–28 Day

fig, ax = plt.subplots(figsize=(10, 6))

bars = ax.bar(labels, lead_summary['dna_rate_%'], color=colors)

ax.set_ylabel('DNA rate %')
ax.set_title('DNA rate rises sharply with booking lead time')
ax.set_ylim(0, max(lead_summary['dna_rate_%']) * 1.2)

for bar, val in zip(bars, lead_summary['dna_rate_%']):
    ax.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 0.1,
            f'{val}%', ha='center', va='bottom', fontsize=10)

plt.xticks(rotation=15, ha='right')
plt.tight_layout()
plt.show()
```



Question 4: What does DNA look like over time, and for ICBS

In [50]:
```
# --- DNA % by ICB and month ---
dna = ar[ar['appointment_status'] == 'DNA']

icb_dna = dna.groupby(['icb_ons_code', 'appointment_month'])['count_of_appointme
total_appts = ar.groupby(['icb_ons_code', 'appointment_month'])['count_of_appoin

icb = icb_dna.merge(total_appts, on=['icb_ons_code', 'appointment_month'], suffi
icb['dna_pct'] = (icb['count_of_appointments_dna'] / icb['count_of_appointments_

# --- National average by month ---
national_dna = dna.groupby('appointment_month')['count_of_appointments'].sum().r
national_total = ar.groupby('appointment_month')['count_of_appointments'].sum().
national = national_dna.merge(national_total, on='appointment_month', suffixes=(
national['dna_pct'] = (national['count_of_appointments_dna'] / national['count_o
national = national.sort_values('appointment_month')

# --- Month labels ---
month_labels = [pd.to_datetime(m).strftime('%b %y') for m in national['appointme
# --- Plot ---
fig, ax = plt.subplots(figsize=(11, 6))

fig.suptitle(
    'GP appointment DNA rates over time, By ICB, with National Average',
    fontsize=13,
    fontweight=500,
    color='#2C2C2A',
    x=0.01,
    ha='left',
    y=1.01
)
# Individual ICB lines
for icb_code, group in icb.groupby('icb_ons_code'):
    group = group.sort_values('appointment_month')
    ax.plot(
        month_labels,
        group['dna_pct'].values,
```

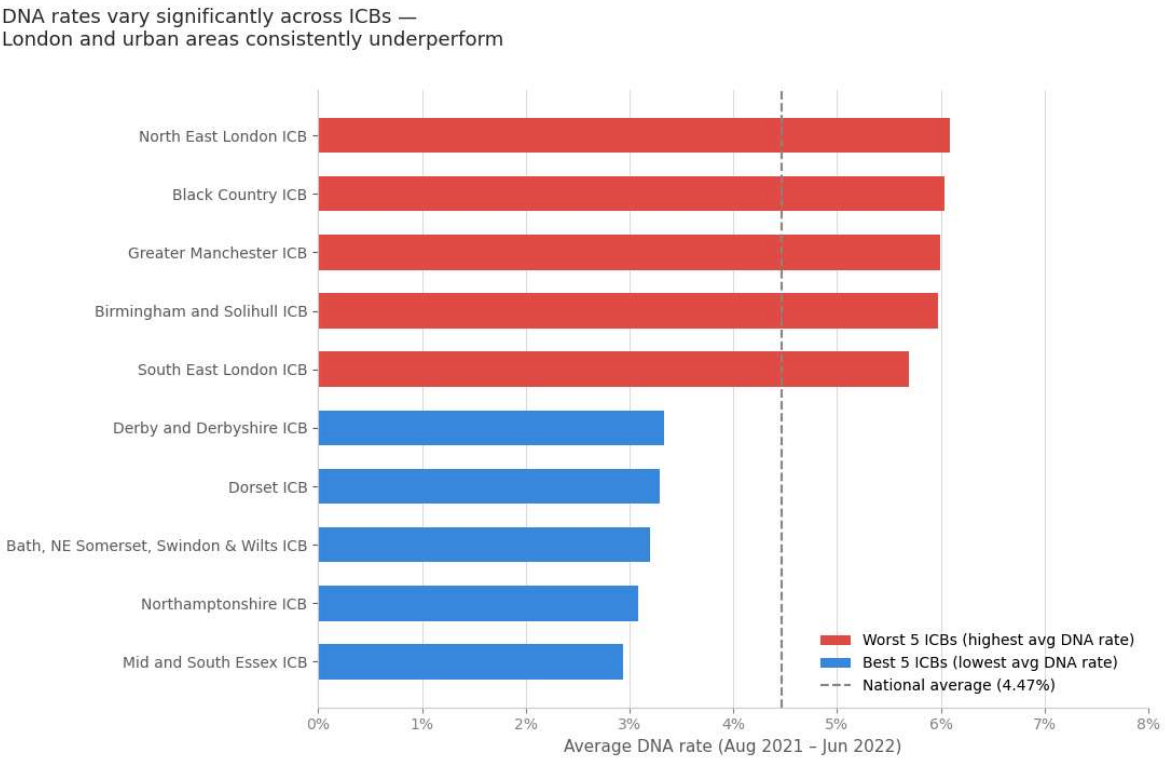
```
        color='#B4B2A9',
        linewidth=0.8,
        alpha=0.5,
        zorder=1
    )
# National average line
ax.plot(
    month_labels,
    national['dna_pct'].values,
    color='#534AB7',
    linewidth=2.5,
    marker='o',
    markersize=4,
    zorder=2,
    label='National average'
)

# Formatting
ax.set_ylim(1, 9)
ax.yaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f'{x:.0f}%'))
ax.set_ylabel('DNA rate (% of all appointments)', color='#5F5E5A', fontsize=11)

for spine in ['top', 'right']:
    ax.spines[spine].set_visible(False)
ax.spines['bottom'].set_color('#D3D1C7')
ax.spines['left'].set_color('#D3D1C7')
ax.yaxis.grid(True, color='#D3D1C7', linewidth=0.5, zorder=0)
ax.set_axisbelow(True)
ax.tick_params(colors='#888780', labelsize=10)

ax.legend(fontsize=10, frameon=False)

# Footnote
ax.annotate(
    'Grey lines = individual ICBs (42 total)',
    xy=(0, -0.12), xycoords='axes fraction',
    fontsize=9, color='#5F5E5A'
)

plt.tight_layout()
plt.savefig('dna_trend_national_icb.png', dpi=150, bbox_inches='tight')
plt.show()
```



Question 5: Which are the worst (and best) ICBs for DNA?

In [51]:
```
# --- Build ICB name lookup from appointments duration file ---
icb_lookup = ad[['icb_ons_code', 'sub_icb_location_name']].drop_duplicates()
icb_lookup['icb_name'] = icb_lookup['sub_icb_location_name'].str.replace(r' - \w
icb_lookup = icb_lookup[['icb_ons_code', 'icb_name']].drop_duplicates(subset='ic

# --- Calculate average DNA rate per ICB across all months ---
dna = ar[ar['appointment_status'] == 'DNA']

icb_dna = dna.groupby(['icb_ons_code', 'appointment_month'])['count_of_appointme
total_appts = ar.groupby(['icb_ons_code', 'appointment_month'])['count_of_appoin

icb = icb_dna.merge(total_appts, on=['icb_ons_code', 'appointment_month'], suffi
icb['dna_pct'] = (icb['count_of_appointments_dna'] / icb['count_of_appointments_

avg = icb.groupby('icb_ons_code')['dna_pct'].mean().round(2).reset_index()
avg.columns = ['icb_ons_code', 'avg_dna_pct']
avg = avg.merge(icb_lookup, on='icb_ons_code', how='left')
avg = avg.sort_values('avg_dna_pct', ascending=False)

# --- Select best 5 and worst 5 ---
worst5 = avg.head(5)
best5 = avg.tail(5).sort_values('avg_dna_pct', ascending=False)
plot_df = pd.concat([worst5, best5]).reset_index(drop=True)

# Shorten long ICB names for display
plot_df['icb_name'] = plot_df['icb_name'].str.replace('NHS ', '', regex=False)
plot_df['icb_name'] = plot_df['icb_name'].str.replace(
    'Bath and North East Somerset Swindon and Wiltshire',
    'Bath, NE Somerset, Swindon & Wilts', regex=False
)

# --- National average ---
national_dna = dna.groupby('appointment_month')['count_of_appointments'].sum()
national_total = ar.groupby('appointment_month')['count_of_appointments'].sum()
national_avg = (national_dna / national_total * 100).mean().round(2)

# --- Plot ---
```

```
fig, ax = plt.subplots(figsize=(11, 7))

colors = ['#E24B4A'] * 5 + ['#378ADD'] * 5

bars = ax.barh(
    plot_df['icb_name'],
    plot_df['avg_dna_pct'],
    color=colors,
    height=0.6
)

# National average line
ax.axvline(x=national_avg, color='#888780', linewidth=1.5, linestyle='--', label

# Formatting
ax.set_xlim(0, 8)
ax.xaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f'{x:.0f}%'))
ax.set_xlabel('Average DNA rate (Aug 2021 – Jun 2022)', color='#5F5E5A', fontsiz
ax.invert_yaxis()

for spine in ['top', 'right']:
    ax.spines[spine].set_visible(False)
ax.spines['bottom'].set_color('#D3D1C7')
ax.spines['left'].set_color('#D3D1C7')

ax.xaxis.grid(True, color='#D3D1C7', linewidth=0.5, zorder=0)
ax.set_axisbelow(True)
ax.tick_params(colors='#888780', labelsize=10)
ax.tick_params(axis='y', colors='#5F5E5A')

# Legend
from matplotlib.patches import Patch
legend_elements = [
    Patch(facecolor='#E24B4A', label='Worst 5 ICBs (highest avg DNA rate)'),
    Patch(facecolor='#378ADD', label='Best 5 ICBs (lowest avg DNA rate)'),
    plt.Line2D([0], [0], color='#888780', linewidth=1.5, linestyle='--', label=f
]
ax.legend(handles=legend_elements, fontsize=10, frameon=False, loc='lower right'

# Title
fig.suptitle(
    'DNA rates vary significantly across ICBs —\nLondon and urban areas consiste
    fontsize=13,
    fontweight=500,
    color='#2C2C2A',
    x=0.01,
    ha='left',
    y=1.01
)

plt.tight_layout()
plt.savefig('dna_best_worst_icbs.png', dpi=150, bbox_inches='tight')
plt.show()
```



Question 6: How have other staff been utilised over time?

In [52]:
```
df_known = ar[ar["hcp_type"] != "Unknown"]

national = (
    df_known.groupby(["appointment_month", "hcp_type"])["count_of_appointments"]
    .sum()
    .unstack(fill_value=0)
)
national["total"] = national.sum(axis=1)
national["pct_other"] = (national["Other Practice staff"] / national["total"] *
national = national.reset_index()[["appointment_month", "pct_other"]]
national["label"] = pd.to_datetime(national["appointment_month"]).dt.strftime("%

# Chart
fig, ax = plt.subplots(figsize=(10, 4.5))
fig.patch.set_facecolor("white")

BLUE_DARK = "#185FA5"
GREY_LINE = "#888780"

x = range(len(national))
ax.fill_between(x, national["pct_other"], alpha=0.15, color=BLUE_DARK, zorder=1)
ax.plot(x, national["pct_other"], color=BLUE_DARK, linewidth=2, zorder=2)
ax.scatter(x, national["pct_other"], color=BLUE_DARK, s=40, zorder=3)
ax.set_xticks(list(x))
ax.set_xticklabels(national["label"], fontsize=9.5, color="#2C2C2A")
ax.set_ylim(44, 54)
ax.yaxis.set_major_formatter(mticker.FuncFormatter(lambda v, _: f"{v:.0f}%"))
ax.tick_params(axis="y", labelsize=9, colors=GREY_LINE)
ax.tick_params(axis="x", length=0)
ax.spines[["top", "right", "left"]].set_visible(False)
ax.spines["bottom"].set_color("#D3D1C7")
ax.grid(axis="y", color="#D3D1C7", linewidth=0.5, zorder=0)
ax.set_axisbelow(True)
```

```
ax.set_title(
    "% of appointments with Other Practice Staff — national average",
    fontsize=12, fontweight="normal", color="#2C2C2A", pad=12,
)
ax.set_ylabel("% Other Practice Staff", fontsize=9, color=GREY_LINE)
fig.text(
    0.01, -0.03,
    "Excludes appointments where HCP type is 'Unknown'.",
    fontsize=8, color=GREY_LINE,
)

plt.tight_layout()
plt.savefig("national_avg_chart.png", dpi=150, bbox_inches="tight", facecolor="w
```

Question 7 What does that look like at ICB level?

In [53]:
```
# National monthly trend
# Exclude 'Unknown' HCP type so % is GP vs Other Practice Staff only
df_known = ar[ar["hcp_type"] != "Unknown"]

national = (
    df_known.groupby(["appointment_month", "hcp_type"])["count_of_appointments"]
    .sum()
    .unstack(fill_value=0)
)
national["total"] = national.sum(axis=1)
national["pct_other"] = (national["Other Practice staff"] / national["total"] *
national_trend = national.reset_index()[["appointment_month", "pct_other"]]
print("=== National monthly % Other Practice Staff ===")
print(national_trend.to_string(index=False))

# ICB monthly %
icb_monthly = (
    df_known.groupby(["appointment_month", "icb_ons_code", "hcp_type"])["count_o
    .sum()
    .unstack(fill_value=0)
)
icb_monthly["total"] = icb_monthly.sum(axis=1)
icb_monthly["pct_other"] = (icb_monthly["Other Practice staff"] / icb_monthly["t
icb_df = icb_monthly.reset_index()[["appointment_month", "icb_ons_code", "pct_ot

# Aug 2021 and Jun 2022 snapshots
```

```
aug21 = (
    icb_df[icb_df["appointment_month"] == "2021-08-01"]
    .set_index("icb_ons_code")["pct_other"]
    .rename("aug21")
)
jun22 = (
    icb_df[icb_df["appointment_month"] == "2022-06-01"]
    .set_index("icb_ons_code")["pct_other"]
    .rename("jun22")
)
# Summary table
summary = pd.concat([aug21, jun22], axis=1)
summary["change_pp"] = (summary["jun22"] - summary["aug21"]).round(2)
summary = summary.merge(icb_lookup, left_index=True, right_on="icb_ons_code")
summary = summary.sort_values("jun22", ascending=False).reset_index(drop=True)

print("\n=== All ICBs ranked by Jun 2022 % Other Practice Staff ===")
print(summary[["icb_name", "aug21", "jun22", "change_pp"]].to_string(index=False

# Top 5 and bottom 5
top5 = summary.head(5)
bottom5 = summary.tail(5)

print("\n=== Top 5 ICBs (Jun 2022) ===")
print(top5[["icb_name", "aug21", "jun22", "change_pp"]].to_string(index=False))

print("\n=== Bottom 5 ICBs (Jun 2022) ===")
print(bottom5[["icb_name", "aug21", "jun22", "change_pp"]].to_string(index=False
```

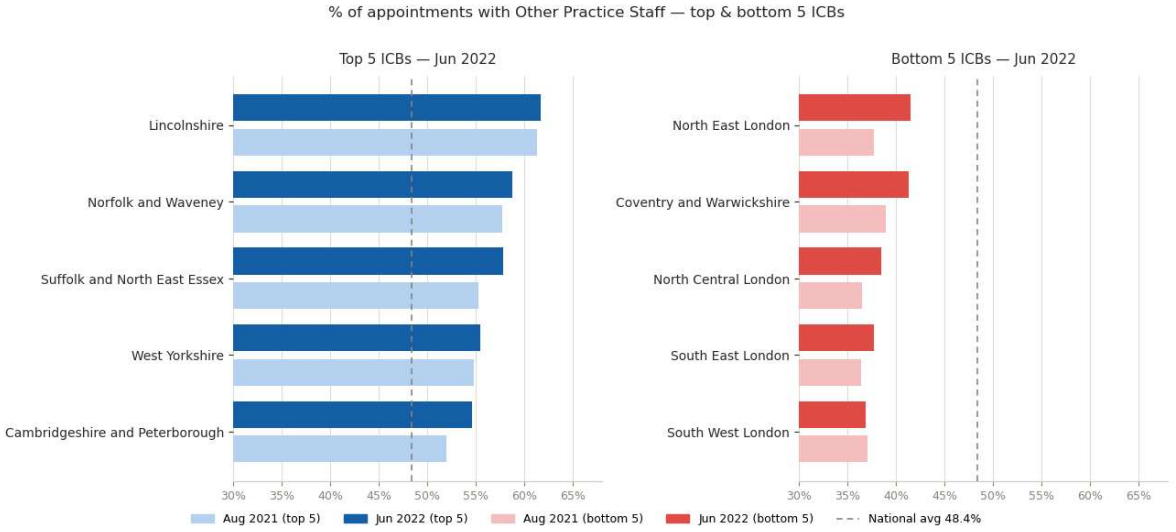
```
=== National monthly % Other Practice Staff ===
appointment_month  pct_other
       2021-08-01      46.74
       2021-09-01      47.54
       2021-10-01      51.10
       2021-11-01      49.21
       2021-12-01      47.85
       2022-01-01      47.12
       2022-02-01      47.59
       2022-03-01      47.21
       2022-04-01      48.50
       2022-05-01      48.17
       2022-06-01      48.43

=== All ICBs ranked by Jun 2022 % Other Practice Staff ===
                                                  icb_name  aug21  jun22  change_
pp
                                     NHS Lincolnshire ICB  61.33  61.67       0.
34
                                NHS Norfolk and Waveney ICB  57.71  58.74       1.
03
                        NHS Suffolk and North East Essex ICB  55.25  57.81       2.
56
                                    NHS West Yorkshire ICB  54.80  55.44       0.
64
                     NHS Cambridgeshire and Peterborough ICB  51.98  54.64       2.
66
NHS Bath and North East Somerset Swindon and Wiltshire ICB  51.85  53.49       1.
64
                          NHS Humber and North Yorkshire ICB  53.00  53.29       0.
29
                                  NHS South Yorkshire ICB  53.19  52.89      -0.
30
                    NHS Cornwall and The Isles Of Scilly ICB  50.24  52.86       2.
62
                        NHS North East and North Cumbria ICB  51.47  52.54       1.
07
                                 NHS Northamptonshire ICB  51.70  52.04       0.
34
                                           NHS Devon ICB  50.81  52.02       1.
21
                                          NHS Dorset ICB  49.34  51.98       2.
64
                        NHS Lancashire and South Cumbria ICB  49.77  51.68       1.
91
                NHS Leicester Leicestershire and Rutland ICB  48.70  51.37       2.
67
                                       NHS Somerset ICB  49.75  51.36       1.
61
                                NHS Mid and South Essex ICB  47.88  51.13       3.
25
                    NHS Staffordshire and Stoke-on-Trent ICB  48.89  51.12       2.
23
                                  NHS Gloucestershire ICB  47.20  50.64       3.
44
                NHS Bedfordshire Luton and Milton Keynes ICB  48.21  50.39       2.
18
                           NHS Hampshire and Isle Of Wight ICB  49.72  49.73       0.
01
  NHS Bristol North Somerset and South Gloucestershire ICB  47.20  49.70       2.
```

```
50
                                NHS Kent and Medway ICB  46.91  49.56       2.
65
                  NHS Nottingham and Nottinghamshire ICB  47.58  48.73       1.
15
                                      NHS Sussex ICB  46.62  48.60       1.
98
                   NHS Shropshire Telford and Wrekin ICB  47.92  48.59       0.
67
                              NHS Derby and Derbyshire ICB  47.45  48.46       1.
01
                                 NHS North West London ICB  44.60  46.82       2.
22
                NHS Herefordshire and Worcestershire ICB  45.01  46.72       1.
71
    NHS Buckinghamshire Oxfordshire and Berkshire West ICB  43.30  45.36       2.
06
                                   NHS Black Country ICB  42.42  45.22       2.
80
                           NHS Cheshire and Merseyside ICB  42.68  44.48       1.
80
                                NHS Greater Manchester ICB  43.26  44.25       0.
99
                           NHS Birmingham and Solihull ICB  39.89  42.84       2.
95
                     NHS Hertfordshire and West Essex ICB  40.22  41.91       1.
69
                                 NHS Surrey Heartlands ICB  39.10  41.88       2.
78
                                          NHS Frimley ICB  39.45  41.82       2.
37
                                 NHS North East London ICB  37.73  41.48       3.
75
                         NHS Coventry and Warwickshire ICB  38.96  41.29       2.
33
                              NHS North Central London ICB  36.53  38.47       1.
94
                                 NHS South East London ICB  36.37  37.69       1.
32
                                 NHS South West London ICB  37.06  36.92      -0.
14

=== Top 5 ICBs (Jun 2022) ===
                               icb_name  aug21  jun22  change_pp
                  NHS Lincolnshire ICB  61.33  61.67       0.34
            NHS Norfolk and Waveney ICB  57.71  58.74       1.03
   NHS Suffolk and North East Essex ICB  55.25  57.81       2.56
                NHS West Yorkshire ICB  54.80  55.44       0.64
NHS Cambridgeshire and Peterborough ICB  51.98  54.64       2.66

=== Bottom 5 ICBs (Jun 2022) ===
                         icb_name  aug21  jun22  change_pp
        NHS North East London ICB  37.73  41.48       3.75
NHS Coventry and Warwickshire ICB  38.96  41.29       2.33
     NHS North Central London ICB  36.53  38.47       1.94
        NHS South East London ICB  36.37  37.69       1.32
        NHS South West London ICB  37.06  36.92      -0.14
```

Question 7 continued: Building a chart for non-GP staff at ICB level

In [54]:
```
# Shorten labels for the chart
def shorten(name):
    return name.replace("NHS ", "").replace(" Integrated Care Board", "").replac

top5["label"] = top5["icb_name"].apply(shorten)
bottom5["label"] = bottom5["icb_name"].apply(shorten)

# Colours
BLUE_LIGHT  = "#B5D4F4"
BLUE_DARK   = "#185FA5"
RED_LIGHT   = "#F7C1C1"
RED_DARK    = "#E24B4A"
GREY_LINE   = "#888780"
NATIONAL_AVG = 48.4

# Chart
fig, axes = plt.subplots(1, 2, figsize=(13, 5.5), sharey=False)
fig.patch.set_facecolor("white")
bar_height = 0.35
gap = 0.1

def draw_panel(ax, df, light_col, dark_col, title):
    n = len(df)
    y = np.arange(n)

    ax.barh(y + bar_height / 2 + gap / 2, df["aug21"], height=bar_height,
            color=light_col, label="Aug 2021", zorder=2)
    ax.barh(y - bar_height / 2 - gap / 2, df["jun22"], height=bar_height,
            color=dark_col, label="Jun 2022", zorder=2)

    ax.axvline(NATIONAL_AVG, color=GREY_LINE, linewidth=1.2,
               linestyle=(0, (4, 3)), zorder=3)

    ax.set_yticks(y)
    ax.set_yticklabels(df["label"], fontsize=10)
    ax.set_xlim(30, 68)
    ax.xaxis.set_major_formatter(plt.FuncFormatter(lambda v, _: f"{v:.0f}%"))
    ax.tick_params(axis="x", labelsize=9, colors=GREY_LINE)
    ax.tick_params(axis="y", labelsize=10, colors="#2C2C2A")
    ax.set_title(title, fontsize=11, fontweight="normal", pad=10, color="#2C2C2A
    ax.spines[["top", "right", "left"]].set_visible(False)
    ax.spines["bottom"].set_color("#D3D1C7")
    ax.grid(axis="x", color="#D3D1C7", linewidth=0.5, zorder=0)
    ax.set_axisbelow(True)
    ax.invert_yaxis()

draw_panel(axes[0], top5,    BLUE_LIGHT, BLUE_DARK, "Top 5 ICBs — Jun 2022")
draw_panel(axes[1], bottom5, RED_LIGHT,  RED_DARK,  "Bottom 5 ICBs — Jun 2022")

# Shared legend
legend_handles = [
    mpatches.Patch(color=BLUE_LIGHT, label="Aug 2021 (top 5)"),
    mpatches.Patch(color=BLUE_DARK,  label="Jun 2022 (top 5)"),
    mpatches.Patch(color=RED_LIGHT,  label="Aug 2021 (bottom 5)"),
    mpatches.Patch(color=RED_DARK,   label="Jun 2022 (bottom 5)"),
    plt.Line2D([0], [0], color=GREY_LINE, linewidth=1.2,
               linestyle=(0, (4, 3)), label=f"National avg {NATIONAL_AVG}%"),
]
fig.legend(handles=legend_handles, loc="lower center", ncol=5,
```

```
            fontsize=9, frameon=False, bbox_to_anchor=(0.5, -0.04))

fig.suptitle(
    "% of appointments with Other Practice Staff — top & bottom 5 ICBs",
    fontsize=12, fontweight="normal", color="#2C2C2A", y=1.01
)
plt.tight_layout()
plt.savefig("icb_top_bottom5_chart.png", dpi=150, bbox_inches="tight", facecolor
```



In [ ]:

Data Exploration - National Categories File

Question 1: How many service settings, context types, national categories, and appointment statuses are there?

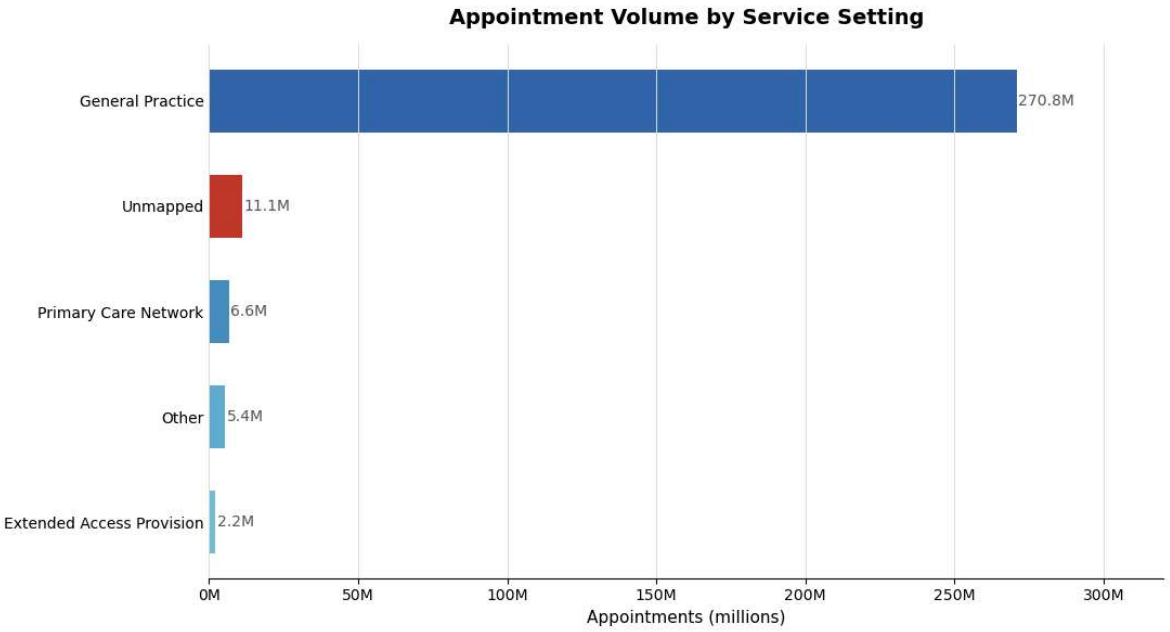
In [55]:
```
# Determine the number of service settings.
service_settings = nc['service_setting'].value_counts()
print(service_settings)

# Determine the number of service settings.
context_types = nc['context_type'].value_counts()
print(context_types)

# Determine the number of national categories.
national_categories = nc['national_category'].value_counts()
print(national_categories)
```

```
service_setting
General Practice             359274
Primary Care Network         183790
Other                        138789
Extended Access Provision    108122
Unmapped                      27419
Name: count, dtype: int64
context_type
Care Related Encounter    700481
Inconsistent Mapping       89494
Unmapped                   27419
Name: count, dtype: int64
national_category
Inconsistent Mapping                                                    89494
General Consultation Routine                                            89329
General Consultation Acute                                              84874
Planned Clinics                                                         76429
Clinical Triage                                                         74539
Planned Clinical Procedure                                              59631
Structured Medication Review                                            44467
Service provided by organisation external to the practice               43095
Home Visit                                                              41850
Unplanned Clinical Activity                                             40415
Patient contact during Care Home Round                                  28795
Unmapped                                                                27419
Care Home Visit                                                         26644
Social Prescribing Service                                              26492
Care Home Needs Assessment & Personalised Care and Support Planning     23505
Non-contractual chargeable work                                         20896
Walk-in                                                                 14179
Group Consultation and Group Education                                   5341
Name: count, dtype: int64
```

Question 2: What does a count of appointments against service setting look like?

In [56]:
```
#group by service setting and sum appointments
service = (
    nc.groupby("service_setting")["count_of_appointments"].sum().sort_values(asc

# --- Colours (data quality issues highlighted, blues for clinical) ---
blue_shades = ["#3266ad","#4a8fc2","#5faad0","#73c2da"]
colour_map = {"Unmapped":"#c0392b"}
final_colours = []
blue_idx = 0
for label in service.index:
    if label in colour_map:
        final_colours.append(colour_map[label])
    else:
        final_colours.append(blue_shades[blue_idx])
        blue_idx = min(blue_idx + 1, len(blue_shades) - 1)
# --- Plot ---
fig, ax = plt.subplots(figsize=(11, 6))

bars = ax.barh(service.index, service.values / 1e6, color=final_colours, edgecol
for bar, val in zip(bars, service.values):
    ax.text(
        bar.get_width() + 0.5,
        bar.get_y() + bar.get_height() / 2,
        f"{val / 1e6:.1f}M",
        va="center", ha="left", fontsize=10, color="#555",
```

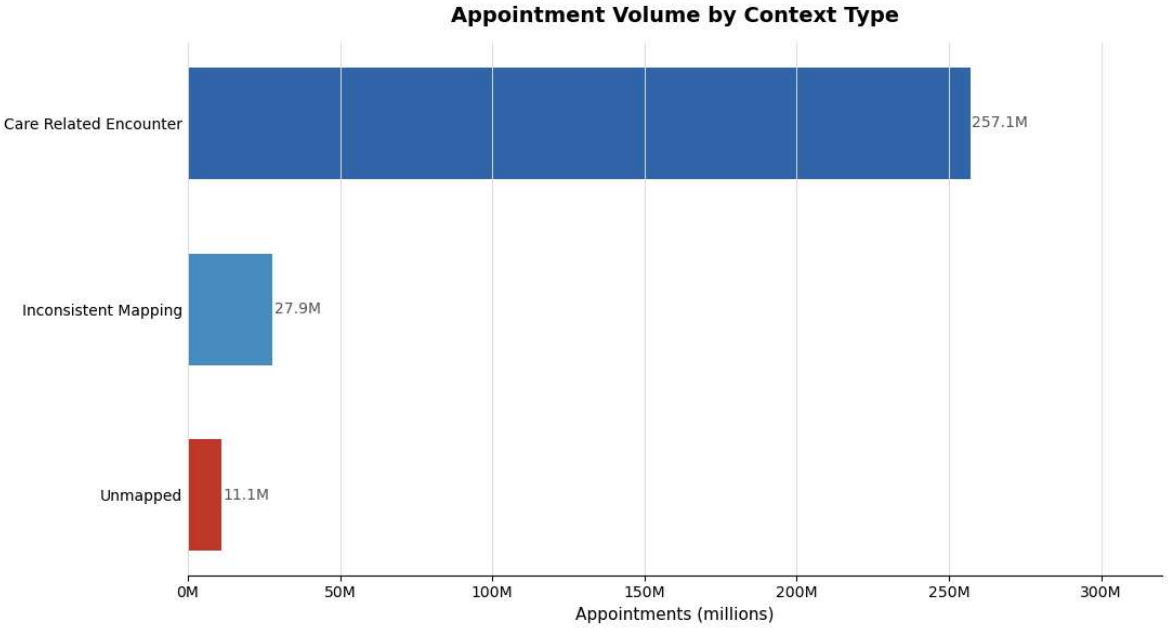
```
    )

ax.set_xlabel("Appointments (millions)", fontsize=11)
ax.set_title("Appointment Volume by Service Setting", fontsize=14, fontweight="b
ax.xaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f"{x:.0f}M"))
ax.invert_yaxis()
ax.spines[["top", "right", "left"]].set_visible(False)
ax.tick_params(axis="y", length=0)
ax.set_xlim(0, 320)
ax.grid(axis="x", color="#e0e0e0", linewidth=0.7)

plt.tight_layout()
plt.savefig("appointments_by_setting.png", dpi=150, bbox_inches="tight")
plt.show()
```



Question 3: What does a count of appointments against context type look like?

In [57]:
```
#group by context type and sum appointments
context = (
    nc.groupby("context_type")["count_of_appointments"].sum().sort_values(ascend

# --- Colours (data quality issues highlighted, blues for clinical) ---
blue_shades = ["#3266ad","#4a8fc2","#5faad0","#73c2da"]
colour_map = {"Unmapped":"#c0392b"}
final_colours = []
blue_idx = 0
for label in context.index:
    if label in colour_map:
        final_colours.append(colour_map[label])
    else:
        final_colours.append(blue_shades[blue_idx])
        blue_idx = min(blue_idx + 1, len(blue_shades) - 1)
# --- Plot ---
fig, ax = plt.subplots(figsize=(11, 6))

bars = ax.barh(context.index, context.values / 1e6, color=final_colours, edgecol
for bar, val in zip(bars, context.values):
    ax.text(
        bar.get_width() + 0.5,
        bar.get_y() + bar.get_height() / 2,
```

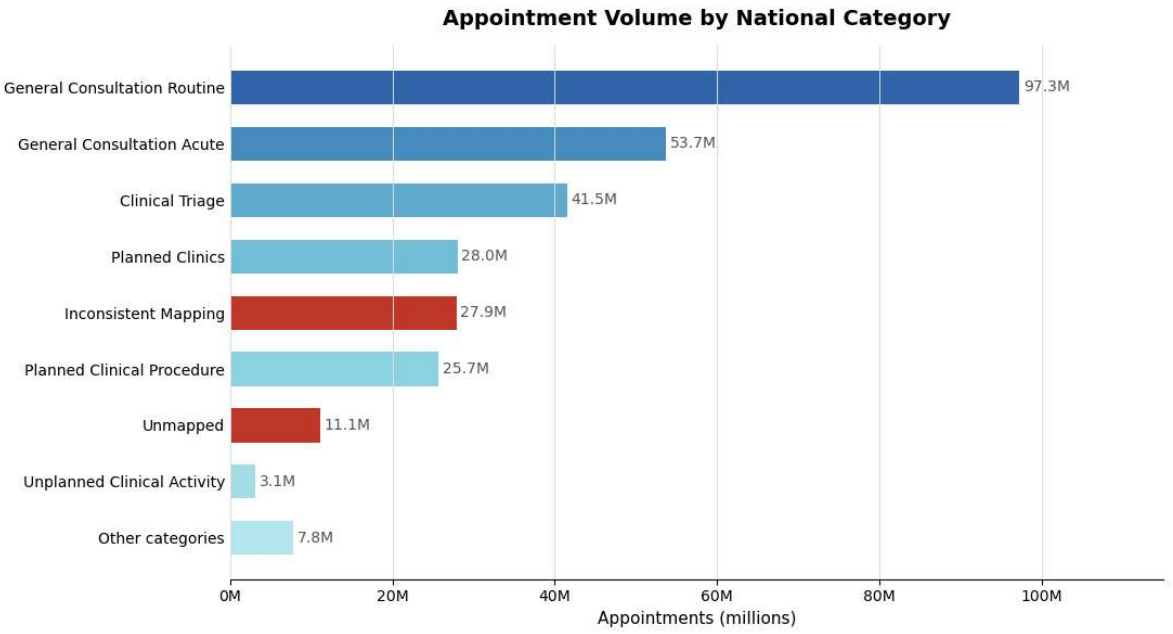
```
        f"{val / 1e6:.1f}M",
        va="center", ha="left", fontsize=10, color="#555",
    )

ax.set_xlabel("Appointments (millions)", fontsize=11)
ax.set_title("Appointment Volume by Context Type", fontsize=14, fontweight="bold
ax.xaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f"{x:.0f}M"))
ax.invert_yaxis()
ax.spines[["top", "right", "left"]].set_visible(False)
ax.tick_params(axis="y", length=0)
ax.set_xlim(0, 320)
ax.grid(axis="x", color="#e0e0e0", linewidth=0.7)

plt.tight_layout()
plt.savefig("appointments_by_context.png", dpi=150, bbox_inches="tight")
plt.show()
```



Question 4: What does a count of appointments against national category look like?

In [58]:
```
#group by category and sum appointments
cat = (
    nc.groupby("national_category")["count_of_appointments"].sum().sort_values(a


# Collapse small categories into "Other categories"
top = cat[cat >= 3e6].copy()
top["Other categories"] = cat[cat < 3e6].sum()

# --- Colours (data quality issues highlighted, blues for clinical) ---
blue_shades = ["#3266ad","#4a8fc2","#5faad0","#73c2da","#8dd5e1","#a8e0e8","#b8e
colour_map = {
    "Inconsistent Mapping": "#c0392b",
    "Unmapped":             "#c0392b",
}
final_colours = []
blue_idx = 0
for label in top.index:
    if label in colour_map:
        final_colours.append(colour_map[label])
    else:
```

```
        final_colours.append(blue_shades[blue_idx])
        blue_idx = min(blue_idx + 1, len(blue_shades) - 1)

# --- Plot ---
fig, ax = plt.subplots(figsize=(11, 6))

bars = ax.barh(top.index, top.values / 1e6, color=final_colours, edgecolor="none
for bar, val in zip(bars, top.values):
    ax.text(
        bar.get_width() + 0.5,
        bar.get_y() + bar.get_height() / 2,
        f"{val / 1e6:.1f}M",
        va="center", ha="left", fontsize=10, color="#555",
    )

ax.set_xlabel("Appointments (millions)", fontsize=11)
ax.set_title("Appointment Volume by National Category", fontsize=14, fontweight=
ax.xaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f"{x:.0f}M"))
ax.invert_yaxis()
ax.spines[["top", "right", "left"]].set_visible(False)
ax.tick_params(axis="y", length=0)
ax.set_xlim(0, 115)
ax.grid(axis="x", color="#e0e0e0", linewidth=0.7)

plt.tight_layout()
plt.savefig("appointments_by_category.png", dpi=150, bbox_inches="tight")
plt.show()
```



Question 5: Using the same 'fair share' analysis as before, examine appoinments per ICB

In [59]:
```
nc['appointment_date'] = pd.to_datetime(nc['appointment_date'], format='mixed',

DAILY_NATIONAL_THRESHOLD = 1_200_000
num_days = nc['appointment_date'].nunique()
total_national_appointments = DAILY_NATIONAL_THRESHOLD * num_days  # 1.2M x 334

grouped = nc.groupby('icb_ons_code').agg(
    actual_appointments=('count_of_appointments', 'sum'),
    population=('total_patients', 'max'),
    icb_name=('sub_icb_location_name', 'first')
```

```
).reset_index()

total_population = grouped['population'].sum()  # 61,469,262

grouped['population_share'] = grouped['population'] / total_population
grouped['fair_share_appointments'] = (grouped['population_share'] * total_nation
grouped['fair_share_ratio'] = (grouped['actual_appointments'] / grouped['fair_sh

grouped = grouped.sort_values('fair_share_ratio', ascending=False).reset_index(d

print(f"Days: {num_days} | Total capacity: {total_national_appointments:,} | Tot
print(grouped[['icb_name', 'population', 'actual_appointments', 'fair_share_appo
```

```
Days: 334 | Total capacity: 400,800,000 | Total population: 61,469,262
                                              icb_name  population  \
0      NHS Cornwall and The Isles Of Scilly ICB - 11N      597245
1                                NHS Devon ICB - 15N     1269915
2      NHS Herefordshire and Worcestershire ICB - 18C      814356
3                  NHS Norfolk and Waveney ICB - 26A     1081228
4                       NHS West Yorkshire ICB - 02T     2616677
5                     NHS Gloucestershire ICB - 11M      673541
6                 NHS Derby and Derbyshire ICB - 15M     1074187
7          NHS North East and North Cumbria ICB - 00L     3146152
8   NHS Bath and North East Somerset Swindon and W...      978455
9                          NHS Lincolnshire ICB - 71E      808124
10  NHS Leicester Leicestershire and Rutland ICB -...     1178968
11                              NHS Dorset ICB - 11J      819004
12                             NHS Somerset ICB - 11X      593933
13           NHS Humber and North Yorkshire ICB - 02Y     1774388
14       NHS Nottingham and Nottinghamshire ICB - 02Q     1239418
15         NHS Suffolk and North East Essex ICB - 06L     1045626
16                     NHS South Yorkshire ICB - 02P     1478791
17      NHS Cambridgeshire and Peterborough ICB - 06H     1022493
18      NHS Shropshire Telford and Wrekin ICB - M2L0M      518272
19                              NHS Sussex ICB - 09D     1817280
20                    NHS Northamptonshire ICB - 78H      798944
21              NHS Cheshire and Merseyside ICB - 01F     2710981
22         NHS Lancashire and South Cumbria ICB - 00Q     1811110
23          NHS Hampshire and Isle Of Wight ICB - 10R     1917890
24                     NHS Kent and Medway ICB - 91Q     1958865
25              NHS Birmingham and Solihull ICB - 15E     1348610
26                     NHS Black Country ICB - D2P2L     1503075
27          NHS Coventry and Warwickshire ICB - B2M3M     1048561
28  NHS Bristol North Somerset and South Glouceste...     1057251
29  NHS Buckinghamshire Oxfordshire and Berkshire ...     1937531
30                            NHS Frimley ICB - D4U1Y      809260
31     NHS Staffordshire and Stoke-on-Trent ICB - 04Y     1169012
32                  NHS Mid and South Essex ICB - 06Q     1250281
33  NHS Bedfordshire Luton and Milton Keynes ICB -...     1072162
34                  NHS North West London ICB - W2U3Z     2751843
35                    NHS Surrey Heartlands ICB - 92A     1122836
36                   NHS Greater Manchester ICB - 00T     3182014
37         NHS Hertfordshire and West Essex ICB - 06K     1609235
38                    NHS South West London ICB - 36L     1726248
39                  NHS North East London ICB - A3A8R     2346009
40                    NHS South East London ICB - 72Q     2047004
41                NHS North Central London ICB - 93C     1742487

    actual_appointments  fair_share_appointments  fair_share_ratio
0               3697369                 3894236              0.95
1               7447758                 8280267              0.90
2               4700180                 5309872              0.89
3               6074027                 7049966              0.86
4              14358371                17061603              0.84
5               3663418                 4391711              0.83
6               5795343                 7004056              0.83
7              16882235                20513956              0.82
8               5242176                 6379852              0.82
9               4253394                 5269237              0.81
10              6242530                 7687263              0.81
11              4340449                 5340178              0.81
12              3079318                 3872640              0.80
13              9125945                11569599              0.79
```

14	6263489	8081417	0.78
15	5335653	6817829	0.78
16	7549415	9642208	0.78
17	5116300	6666994	0.77
18	2568487	3379306	0.76
19	8929398	11849269	0.75
20	3922181	5209380	0.75
21	13250311	17676496	0.75
22	8757248	11809039	0.74
23	9210625	12505280	0.74
24	9286167	12772450	0.73
25	6383746	8793385	0.73
26	7033637	9800548	0.72
27	4863552	6836966	0.71
28	4899508	6893628	0.71
29	8924264	12633346	0.71
30	3754540	5276644	0.71
31	5325790	7622346	0.70
32	5630586	8152247	0.69
33	4838515	6990852	0.69
34	12142390	17942930	0.68
35	4898540	7321264	0.67
36	13857900	20747788	0.67
37	7059966	10492747	0.67
38	7155030	11255710	0.64
39	9588891	15296758	0.63
40	7850170	13347146	0.59
41	6747958	11361594	0.59

Question 6: What if weekends are excluded?

In [60]:
```
nc['appointment_date'] = pd.to_datetime(nc['appointment_date'], format='mixed',

# Remove weekends (Monday=0 ... Sunday=6)
nc_weekdays_only = nc[nc['appointment_date'].dt.dayofweek < 5]

DAILY_NATIONAL_THRESHOLD = 1_200_000
num_days = nc_weekdays_only['appointment_date'].nunique()
total_national_appointments = DAILY_NATIONAL_THRESHOLD * num_days  # 1.2M x 239

grouped = ad.groupby('icb_ons_code').agg(
    actual_appointments=('count_of_appointments', 'sum'),
    population=('total_patients', 'max'),
    icb_name=('sub_icb_location_name', 'first')
).reset_index()

total_population = grouped['population'].sum()  # 61,469,262

grouped['population_share'] = grouped['population'] / total_population
grouped['fair_share_appointments'] = (grouped['population_share'] * total_nation
grouped['fair_share_ratio'] = (grouped['actual_appointments'] / grouped['fair_sh

grouped = grouped.sort_values('fair_share_ratio', ascending=False).reset_index(d

print(f"Days: {num_days} | Total capacity: {total_national_appointments:,} | Tot
print(grouped[['icb_name', 'population', 'actual_appointments', 'fair_share_appo
```

```
Days: 239 | Total capacity: 286,800,000 | Total population: 42,883,400.0
```

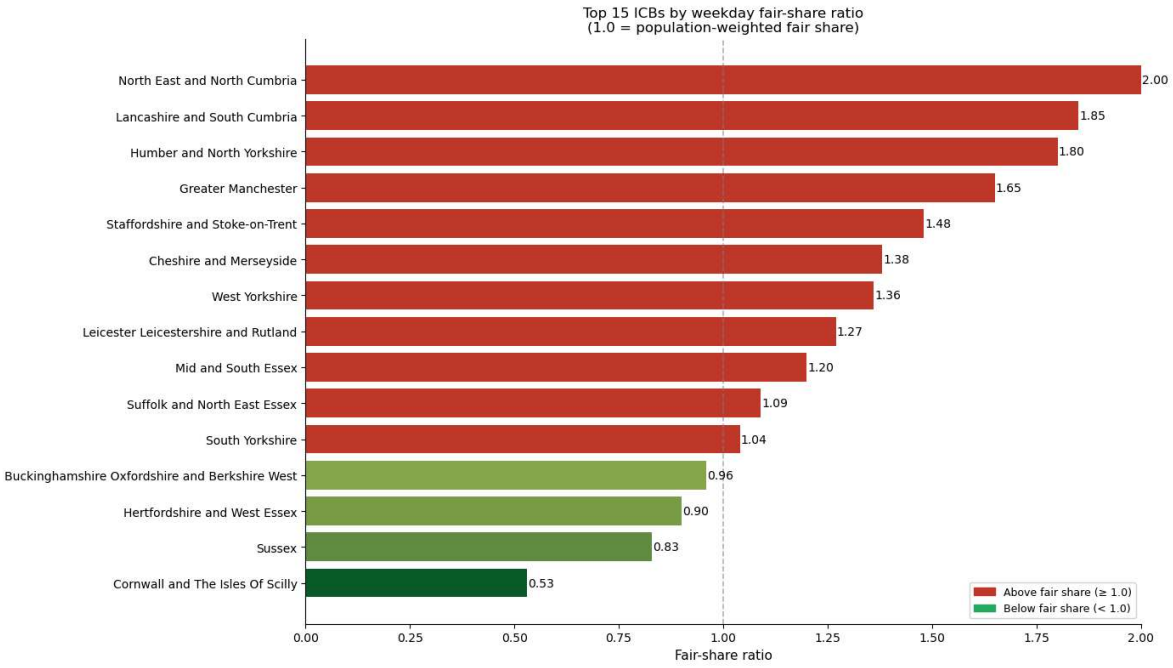
	icb_name	population	actual_appointments	fair_share_appointments	fair_share_ratio
0	NHS North East and North Cumbria ICB - 00L	716200.0	9584943	4789876	2.00
1	NHS Lancashire and South Cumbria ICB - 00Q	394894.0	4886235	2641013	1.85
2	NHS Humber and North Yorkshire ICB - 02Y	436201.0	5248579	2917270	1.80
3	NHS Greater Manchester ICB - 00T	694270.0	7650341	4643210	1.65
4	NHS Staffordshire and Stoke-on-Trent ICB - 04Y	297506.0	2954384	1989691	1.48
5	NHS Cheshire and Merseyside ICB - 01F	795167.0	7344348	5317999	1.38
6	NHS West Yorkshire ICB - 02T	908914.0	8291440	6078728	1.36
7	NHS Leicester Leicestershire and Rutland ICB -...	425863.0	3610315	2848130	1.27
8	NHS Mid and South Essex ICB - 06Q	404251.0	3239958	2703591	1.20
9	NHS Suffolk and North East Essex ICB - 06L	418775.0	3059758	2800726	1.09
10	NHS South Yorkshire ICB - 02P	619466.0	4300648	4142928	1.04
11	NHS Buckinghamshire Oxfordshire and Berkshire ...	792917.0	5072208	5302952	0.96
12	NHS Hertfordshire and West Essex ICB - 06K	668490.0	4026959	4470796	0.90
13	NHS Sussex ICB - 09D	921366.0	5107481	6162006	0.83
14	NHS Cornwall and The Isles Of Scilly ICB - 11N	597245.0			
15	NHS Devon ICB - 15N	1269915.0			
16	NHS Herefordshire and Worcestershire ICB - 18C	814356.0			
17	NHS Nottingham and Nottinghamshire ICB - 02Q	1117719.0			
18	NHS Norfolk and Waveney ICB - 26A	1081228.0			
19	NHS Gloucestershire ICB - 11M	673541.0			
20	NHS Derby and Derbyshire ICB - 15M	1074187.0			
21	NHS Bath and North East Somerset Swindon and W...	978455.0			
22	NHS Dorset ICB - 11J	819004.0			
23	NHS Hampshire and Isle Of Wight ICB - 10R	1687493.0			
24	NHS Lincolnshire ICB - 71E	808124.0			
25	NHS Cambridgeshire and Peterborough ICB - 06H	1022493.0			
26	NHS Northamptonshire ICB - 78H	798944.0			
27	NHS Somerset ICB - 11X	593933.0			
28	NHS Shropshire Telford and Wrekin ICB - M2L0M	518272.0			
29	NHS Frimley ICB - D4U1Y	809260.0			
30	NHS Kent and Medway ICB - 91Q	1958865.0			
31	NHS Birmingham and Solihull ICB - 15E	1348610.0			
32	NHS Bristol North Somerset and South Glouceste...	1057251.0			
33	NHS Bedfordshire Luton and Milton Keynes ICB -...	1072162.0			
34	NHS Coventry and Warwickshire ICB - B2M3M	1048561.0			
35	NHS Black Country ICB - D2P2L	1503075.0			
36	NHS North West London ICB - W2U3Z	2751843.0			
37	NHS Surrey Heartlands ICB - 92A	1122836.0			
38	NHS South West London ICB - 36L	1726248.0			
39	NHS North East London ICB - A3A8R	2346009.0			
40	NHS North Central London ICB - 93C	1742487.0			
41	NHS South East London ICB - 72Q	2047004.0			

14	2097913	3994316	0.53
15	4255338	8493068	0.50
16	2696703	5446334	0.50
17	3570757	7475196	0.48
18	3490647	7231147	0.48
19	2127909	4504577	0.47
20	3295790	7184058	0.46
21	3012568	6543812	0.46
22	2513652	5477419	0.46
23	5242840	11285789	0.46
24	2457468	5404655	0.45
25	2937987	6838334	0.43
26	2276102	5343260	0.43
27	1722566	3972166	0.43
28	1459819	3466153	0.42
29	2155885	5412252	0.40
30	5209641	13100698	0.40
31	3600087	9019372	0.40
32	2756491	7070792	0.39
33	2791385	7170515	0.39
34	2751699	7012674	0.39
35	3901431	10052419	0.39
36	6976986	18404058	0.38
37	2789898	7509418	0.37
38	4014321	11544978	0.35
39	5341883	15689880	0.34
40	3795250	11653583	0.33
41	4360079	13690163	0.32

Question 6 continued: Plot the top twelve ICBS

In [61]:
```
top15 = grouped.head(15).copy()

# Shorten ICB names for display
top15['icb_short'] = (top15['icb_name']
    .str.replace(r'NHS ', '', regex=False)
    .str.replace(r' ICB - .*', '', regex=True)
    .str.replace(r' ICB$', '', regex=True))

# Colour: red if >= 1.0, green gradient if below
def bar_color(ratio):
    if ratio >= 1.0:
        return '#c0392b'
    t = (ratio - top15['fair_share_ratio'].min()) / (1.0 - top15['fair_share_rat
    r = int(10 + t * (150 - 10))
    g = int(92 + t * (174 - 92))
    b = int(42 + t * (80 - 42))
    return f'#{r:02x}{g:02x}{b:02x}'

colors = top15['fair_share_ratio'].apply(bar_color).tolist()

fig, ax = plt.subplots(figsize=(14, 8))

bars = ax.barh(top15['icb_short'], top15['fair_share_ratio'], color=colors, edge

# Fair-share reference line at 1.0
ax.axvline(x=1.0, color='grey', linewidth=1.2, linestyle='--', alpha=0.6)

# Value labels on each bar
for bar, ratio in zip(bars, top15['fair_share_ratio']):
```

```
    ax.text(bar.get_width() + 0.002, bar.get_y() + bar.get_height() / 2,
            f'{ratio:.2f}', va='center', ha='left', fontsize=10)

ax.invert_yaxis()
ax.set_xlim(0, 2)
ax.set_xlabel('Fair-share ratio', fontsize=11)
ax.set_title('Top 15 ICBs by weekday fair-share ratio\n(1.0 = population-weighte
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.tick_params(axis='y', labelsize=10)
ax.tick_params(axis='x', labelsize=10)

red_patch = mpatches.Patch(color='#c0392b', label='Above fair share (≥ 1.0)')
green_patch = mpatches.Patch(color='#27ae60', label='Below fair share (< 1.0)')
ax.legend(handles=[red_patch, green_patch], fontsize=9, loc='lower right')

plt.tight_layout()
plt.savefig('icb_fairshare_top15.png', dpi=150, bbox_inches='tight')
plt.show()
```



Question 7 Investigating inconsistent mapping

In [62]:
```
prob = nc[nc['context_type'].isin(['Inconsistent Mapping', 'Unmapped'])].copy()

icb_prob = prob.groupby(['icb_ons_code', 'context_type'])['count_of_appointments
icb_total = nc.groupby('icb_ons_code')['count_of_appointments'].sum().reset_inde
icb_total.columns = ['icb_ons_code', 'total']
icb_name = nc.groupby('icb_ons_code')['sub_icb_location_name'].first().reset_ind

icb_wide = icb_prob.pivot_table(index='icb_ons_code', columns='context_type', va
icb_wide = icb_wide.merge(icb_total, on='icb_ons_code').merge(icb_name, on='icb_

icb_wide['pct_inconsistent'] = (icb_wide['Inconsistent Mapping'] / icb_wide['tot
icb_wide['pct_unmapped'] = (icb_wide['Unmapped'] / icb_wide['total'] * 100).roun
icb_wide['pct_total'] = icb_wide['pct_inconsistent'] + icb_wide['pct_unmapped']

icb_wide['icb_short'] = (icb_wide['sub_icb_location_name']
    .str.replace(r'NHS ', '', regex=False)
    .str.replace(r' ICB - .*', '', regex=True)
```

```
    .str.replace(r' ICB$', '', regex=True))

top12 = icb_wide.sort_values('pct_total', ascending=False).head(12)
top12
```

Out[62]:

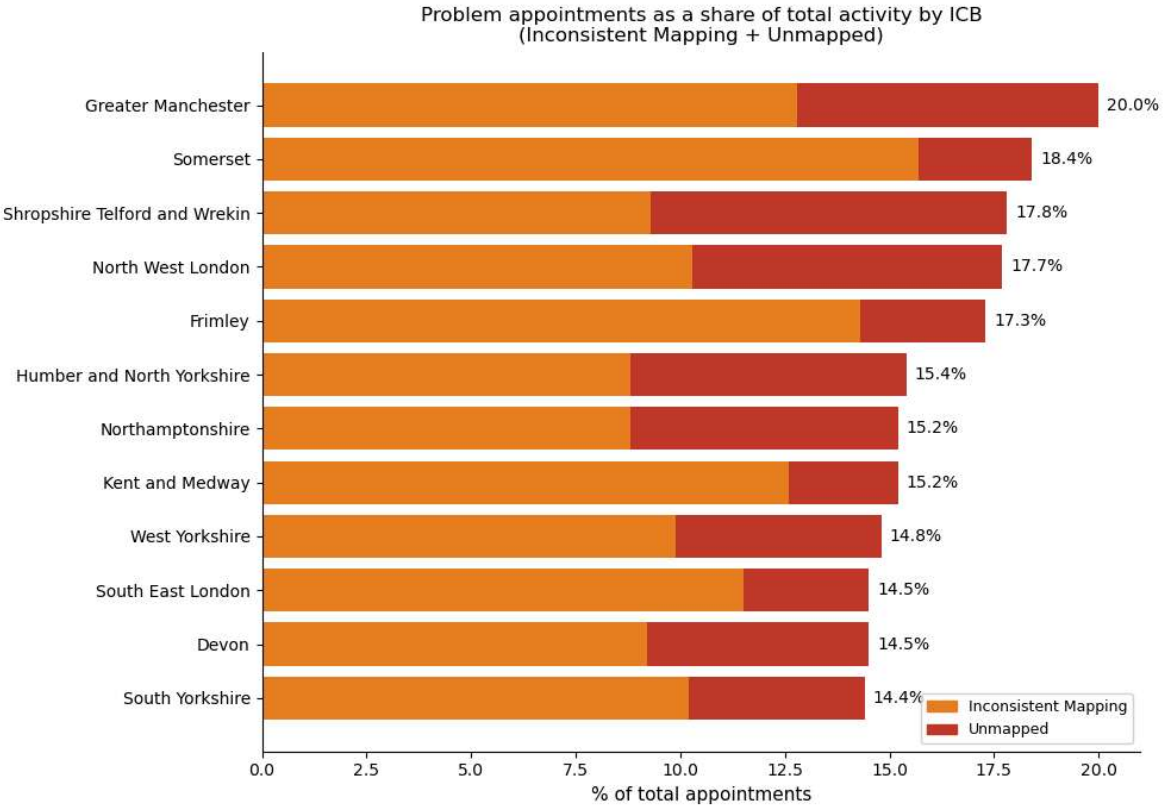
	icb_ons_code	Inconsistent Mapping	Unmapped	total	sub_icb_location_name	pct_incons
36	E54000057	1767555	1000428	13857900	NHS Greater Manchester ICB - 00T	
21	E54000038	482005	82606	3079318	NHS Somerset ICB - 11X	
2	E54000011	239414	219012	2568487	NHS Shropshire Telford and Wrekin ICB - M2L0M	
12	E54000027	1249574	904234	12142390	NHS North West London ICB - W2U3Z	
18	E54000034	536878	114086	3754540	NHS Frimley ICB - D4U1Y	
30	E54000051	801770	602477	9125945	NHS Humber and North Yorkshire ICB - 02Y	
38	E54000059	345912	251121	3922181	NHS Northamptonshire ICB - 78H	
17	E54000032	1171264	239990	9286167	NHS Kent and Medway ICB - 91Q	
33	E54000054	1427714	700194	14358371	NHS West Yorkshire ICB - 02T	
15	E54000030	905129	233787	7850170	NHS South East London ICB - 72Q	
20	E54000037	685957	396621	7447758	NHS Devon ICB - 15N	
40	E54000061	767218	315624	7549415	NHS South Yorkshire ICB - 02P	

Question 7 continued Plotting inconsistent mapping

In [63]:

```
fig, ax = plt.subplots(figsize=(10, 7))
ax.barh(top12['icb_short'], top12['pct_inconsistent'],
        color='#e67e22', edgecolor='none', label='Inconsistent Mapping')
ax.barh(top12['icb_short'], top12['pct_unmapped'],
        left=top12['pct_inconsistent'],
        color='#c0392b', edgecolor='none', label='Unmapped')

for _, row in top12.iterrows():
    ax.text(row['pct_total'] + 0.2,
            top12.index.get_loc(row.name),
            f"{row['pct_total']:.1f}%", va='center', ha='left', fontsize=10)

ax.invert_yaxis()
```

```
ax.set_xlabel('% of total appointments', fontsize=11)
ax.set_title('Problem appointments as a share of total activity by ICB\n(Inconsi
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.tick_params(axis='y', labelsize=10)
ax.tick_params(axis='x', labelsize=10)
orange_patch = mpatches.Patch(color='#e67e22', label='Inconsistent Mapping')
red_patch = mpatches.Patch(color='#c0392b', label='Unmapped')
ax.legend(handles=[orange_patch, red_patch], fontsize=9, loc='lower right')

plt.tight_layout()
plt.savefig('icb_data_quality_pct.png', dpi=150, bbox_inches='tight')
plt.show()
```



Question 8 Are appointments increasing over time?

In [64]:
```
nc['appointment_date'] = pd.to_datetime(nc['appointment_date'], format='mixed',

# Constants
DAILY_NATIONAL_THRESHOLD = 1_200_000

# ICB population (use max total_patients per ICB across all dates)
icb_pop = (
    nc.groupby('icb_ons_code')['total_patients']
    .max()
    .reset_index()
    .rename(columns={'total_patients': 'population'})
)
total_population = icb_pop['population'].sum()
print(f"Total population: {total_population:,}")
print(f"Number of ICBs:   {len(icb_pop)}")

# Daily appointments per ICB
daily_icb = (
    nc.groupby(['appointment_date', 'icb_ons_code'])
```

```
    .agg(actual_appointments=('count_of_appointments', 'sum'))
    .reset_index()
)

# Merge population and calculate fair share & ratio
daily_icb = daily_icb.merge(icb_pop, on='icb_ons_code')
daily_icb['pop_share']  = daily_icb['population'] / total_population
daily_icb['fair_share'] = daily_icb['pop_share'] * DAILY_NATIONAL_THRESHOLD
daily_icb['ratio']      = daily_icb['actual_appointments'] / daily_icb['fair_sha

# Count ICBs with ratio > 1 per day
daily_counts = (
    daily_icb
    .groupby('appointment_date')
    .apply(lambda x: (x['ratio'] > 1).sum())
    .reset_index()
)
daily_counts.columns = ['appointment_date', 'icbs_above_1']
daily_counts = daily_counts.sort_values('appointment_date').reset_index(drop=Tru

# 7-day rolling average
daily_counts['rolling_7'] = (
    daily_counts['icbs_above_1']
    .rolling(window=7, min_periods=1)
    .mean()
    .round(2)
)

# Linear trend (least-squares slope over index)
x = daily_counts.index.values
y = daily_counts['rolling_7'].values
slope = (len(x) * (x * y).sum() - x.sum() * y.sum()) / \
        (len(x) * (x ** 2).sum() - x.sum() ** 2)
intercept = (y.sum() - slope * x.sum()) / len(x)
daily_counts['trend'] = (intercept + slope * x).round(2)

# Summary stats
print(f"\nDate range: {daily_counts['appointment_date'].min().date()} "
      f"to {daily_counts['appointment_date'].max().date()}")
print(f"Total days in dataset:       {len(daily_counts)}")
print(f"Days where zero ICBs > 1:    {(daily_counts['icbs_above_1'] == 0).sum()}
print(f"Days where all 42 ICBs > 1:  {(daily_counts['icbs_above_1'] == 42).sum()
print(f"Mean (non-zero days):         "
      f"{daily_counts.loc[daily_counts['icbs_above_1'] > 0, 'icbs_above_1'].mean
print(f"Trend slope per day:          {slope:.4f} "
      f"({'rising' if slope > 0 else 'falling'})")

filtered_rows = daily_counts[daily_counts['icbs_above_1'] > 40]
print(filtered_rows)
```

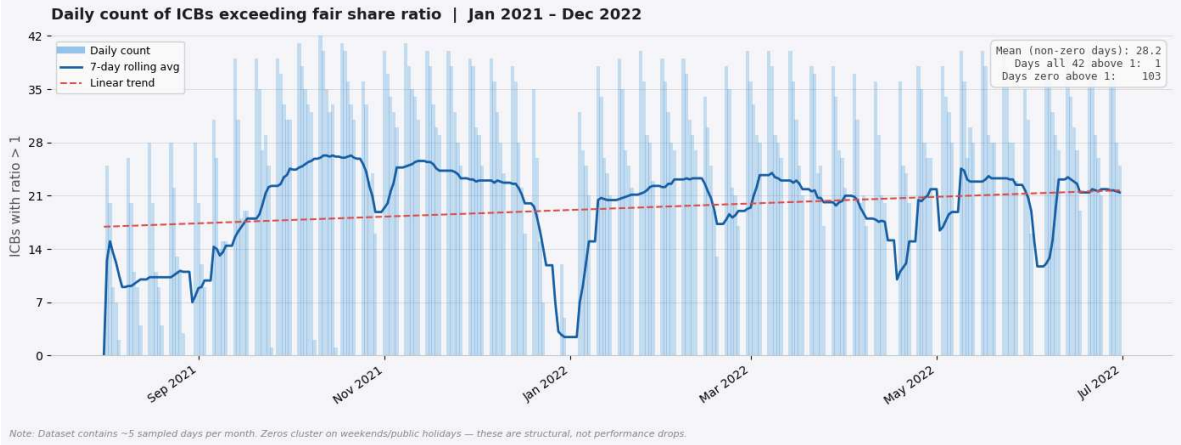
```
Total population: 61,469,262
Number of ICBs:   42

Date range: 2021-08-01 to 2022-06-30
Total days in dataset:       334
Days where zero ICBs > 1:    103
Days where all 42 ICBs > 1:  1
Mean (non-zero days):         28.2
Trend slope per day:          0.0144 (rising)
   appointment_date  icbs_above_1  rolling_7  trend
64       2021-10-04            41      24.71  17.85
71       2021-10-11            42      26.00  17.95
78       2021-10-18            41      26.00  18.05
99       2021-11-08            41      24.86  18.35
```

Question 8 continued Plotting appointments over time

In [67]:
```
# Figure setup
fig, ax = plt.subplots(figsize=(14, 5))
fig.patch.set_facecolor('#F7F8FA')
ax.set_facecolor('#F7F8FA')

# Summary stats for annotation
mean_nonzero = daily_counts.loc[daily_counts['icbs_above_1'] > 0, 'icbs_above_1'
zero_days    = (daily_counts['icbs_above_1'] == 0).sum()
max_days     = (daily_counts['icbs_above_1'] == 42).sum()

# Bar chart: daily count
ax.bar(
    daily_counts['appointment_date'],
    daily_counts['icbs_above_1'],
    width=1.2,
    color='#378ADD',
    alpha=0.25,
    label='Daily count',
    zorder=2
)
# Line: 7-day rolling average
ax.plot(
    daily_counts['appointment_date'],
    daily_counts['rolling_7'],
    color='#185FA5',
    linewidth=2,
    label='7-day rolling avg',
    zorder=3
)

# Line: linear trend
ax.plot(
    daily_counts['appointment_date'],
    daily_counts['trend'],
    color='#E24B4A',
    linewidth=1.5,
    linestyle='--',
    label='Linear trend',
    zorder=4
)
# Axes formatting
ax.set_ylim(0, 42)
ax.set_yticks(range(0, 43, 7))
```

```
ax.set_ylabel('ICBs with ratio > 1', fontsize=11, color='#555')
ax.set_xlabel('')
ax.xaxis.set_major_locator(mdates.MonthLocator(interval=2))
ax.xaxis.set_major_formatter(mdates.DateFormatter('%b %Y'))
plt.xticks(rotation=35, ha='right', fontsize=10)
plt.yticks(fontsize=10)

ax.grid(axis='y', color='#CCCCCC', linewidth=0.6, alpha=0.7)
ax.grid(axis='x', visible=False)
for spine in ['top', 'right', 'left']:
    ax.spines[spine].set_visible(False)
ax.spines['bottom'].set_color('#CCCCCC')

# Title
ax.set_title(
    'Daily count of ICBs exceeding fair share ratio  |  Jan 2021 – Dec 2022',
    fontsize=13, fontweight='bold', color='#222', pad=14, loc='left'
)

# Legend
legend_elements = [
    Line2D([0], [0], color='#378ADD', alpha=0.5, linewidth=6, label='Daily count
    Line2D([0], [0], color='#185FA5', linewidth=2,   label='7-day rolling avg'),
    Line2D([0], [0], color='#E24B4A', linewidth=1.5, linestyle='--', label='Line
]
ax.legend(handles=legend_elements, loc='upper left', fontsize=9,
          framealpha=0.85, edgecolor='#CCCCCC')

# Stats annotation box
stats_text = (
    f"Mean (non-zero days): {mean_nonzero:.1f}\n"
    f"Days all 42 above 1:  {max_days}\n"
    f"Days zero above 1:    {zero_days}"
)
ax.text(
    0.99, 0.97, stats_text,
    transform=ax.transAxes,
    fontsize=9, verticalalignment='top', horizontalalignment='right',
    bbox=dict(boxstyle='round,pad=0.5', facecolor='white', edgecolor='#CCCCCC',
    color='#444', family='monospace'
)
# Note
fig.text(
    0.01, -0.04,
    'Note: Dataset contains ~5 sampled days per month. Zeros cluster on weekends
    'these are structural, not performance drops.',
    fontsize=8, color='#888', style='italic'
)
plt.tight_layout()
plt.savefig("icb_fair_share_timeseries.png", dpi=150, bbox_inches='tight',
            facecolor=fig.get_facecolor())
print("Saved: icb_fair_share_timeseries.png")
print()
print()
plt.show()
```

```
Saved: icb_fair_share_timeseries.png
```

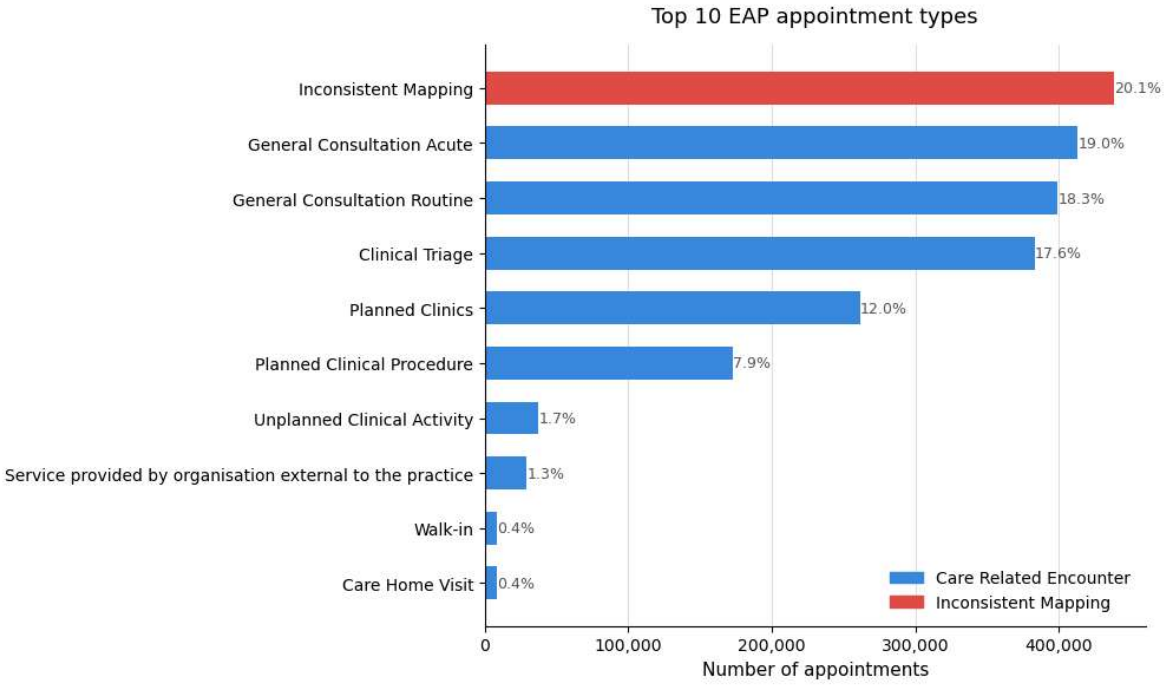


Question 9 Looking at Extended Access - Appointment Types

In [68]:
```
eap = nc[nc['service_setting'] == 'Extended Access Provision']

total_eap = eap['count_of_appointments'].sum()

top10 = (
    eap.groupby('national_category')['count_of_appointments']
    .sum()
    .sort_values(ascending=False)
    .head(10)
    .sort_values(ascending=True)  # ascending so largest bar is at top
)

colors = ['#E24B4A' if cat == 'Inconsistent Mapping' else '#378ADD' for cat in t

fig, ax = plt.subplots(figsize=(10, 6))

bars = ax.barh(top10.index, top10.values, color=colors, height=0.6)

ax.xaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f'{int(x):,}'))
ax.set_xlabel('Number of appointments', fontsize=11)
ax.set_title('Top 10 EAP appointment types', fontsize=13, fontweight='normal', p

for bar in bars:
    w = bar.get_width()
    pct = w / total_eap * 100
    ax.text(w + 80, bar.get_y() + bar.get_height() / 2,
            f'{pct:.1f}%', va='center', fontsize=9, color='#555')

legend_handles = [
    plt.Rectangle((0,0),1,1, color='#378ADD', label='Care Related Encounter'),
    plt.Rectangle((0,0),1,1, color='#E24B4A', label='Inconsistent Mapping'),
]
ax.legend(handles=legend_handles, fontsize=10, frameon=False, loc='lower right')

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.tick_params(axis='y', labelsize=10)
ax.tick_params(axis='x', labelsize=10)
ax.grid(axis='x', color='#e0e0e0', linewidth=0.7)
ax.set_axisbelow(True)

plt.tight_layout()
```

```
plt.savefig('eap_top10_appointment_types.png', dpi=150, bbox_inches='tight')
print('Saved: eap_top10_appointment_types.png')
```

```
Saved: eap_top10_appointment_types.png
```

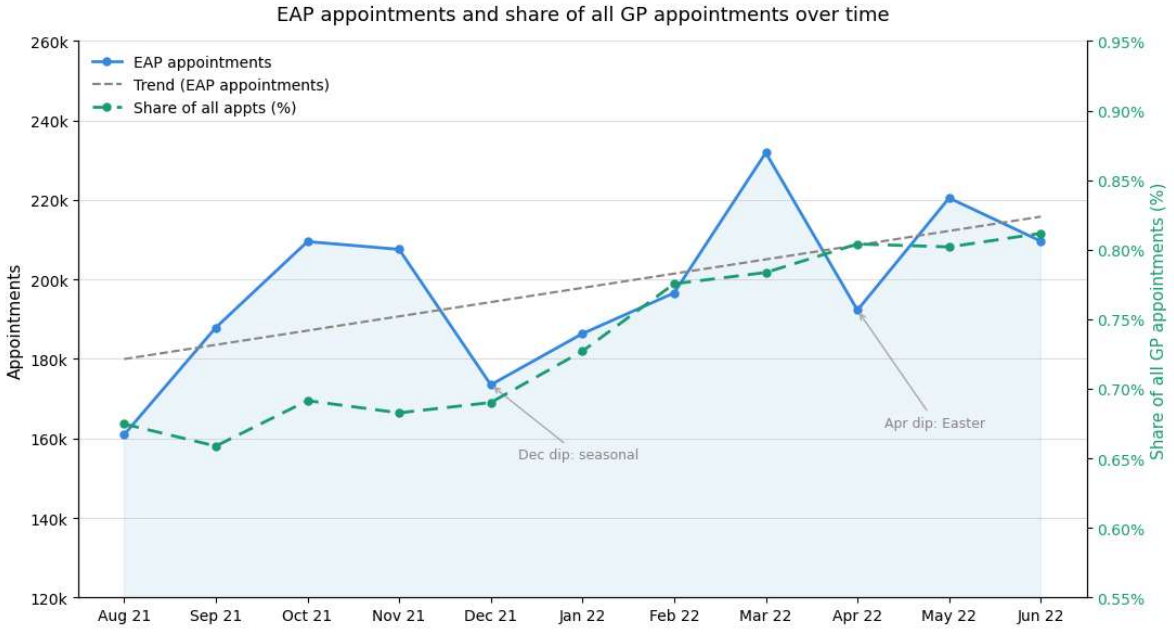


Question 9 continued Extended Access over time

In [69]:
```
monthly = eap.groupby('appointment_month')['count_of_appointments'].sum().reset_
monthly = monthly.sort_values('appointment_month').reset_index(drop=True)

total_monthly = nc.groupby('appointment_month')['count_of_appointments'].sum().r
total_monthly.columns = ['appointment_month', 'total_appointments']
monthly = monthly.merge(total_monthly, on='appointment_month')
monthly['share_pct'] = monthly['count_of_appointments'] / monthly['total_appoint

labels = ['Aug 21','Sep 21','Oct 21','Nov 21','Dec 21','Jan 22','Feb 22','Mar 22
x = np.arange(len(labels))

z = np.polyfit(x, monthly['count_of_appointments'], 1)
trend = np.poly1d(z)(x)

BLUE = '#378ADD'
GREEN = '#1D9E75'
GRAY = '#888780'

fig, ax1 = plt.subplots(figsize=(11, 6))

ax1.plot(x, monthly['count_of_appointments'], color=BLUE, linewidth=2,
         marker='o', markersize=5, label='EAP appointments')
ax1.plot(x, trend, color=GRAY, linewidth=1.5, linestyle='--', label='Trend (EAP
ax1.fill_between(x, monthly['count_of_appointments'], alpha=0.08, color=BLUE)

ax2 = ax1.twinx()
ax2.plot(x, monthly['share_pct'], color=GREEN, linewidth=2,
         marker='o', markersize=5, linestyle=(0, (5, 3)), label='Share of all ap

ax1.set_xticks(x)
ax1.set_xticklabels(labels, fontsize=10)
ax1.set_ylabel('Appointments', fontsize=11)
```

```
ax1.yaxis.set_major_formatter(mticker.FuncFormatter(lambda v, _: f'{int(v/1000)}
ax1.tick_params(axis='y', labelsize=10)
ax1.set_ylim(120000, 260000)

ax2.set_ylabel('Share of all GP appointments (%)', fontsize=11, color=GREEN)
ax2.tick_params(axis='y', labelcolor=GREEN, labelsize=10)
ax2.yaxis.set_major_formatter(mticker.FuncFormatter(lambda v, _: f'{v:.2f}%'))
ax2.set_ylim(0.55, 0.95)

ax1.set_title('EAP appointments and share of all GP appointments over time', fon
              fontweight='normal', pad=14)

ax1.spines['top'].set_visible(False)
ax2.spines['top'].set_visible(False)
ax1.grid(axis='y', color='#e0e0e0', linewidth=0.7)
ax1.set_axisbelow(True)

lines1, labels1 = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()
ax1.legend(lines1 + lines2, labels1 + labels2, fontsize=10, frameon=False, loc='

ax1.annotate('Dec dip: seasonal', xy=(4, monthly['count_of_appointments'].iloc[4
             xytext=(4.3, 155000), fontsize=9, color='#888',
             arrowprops=dict(arrowstyle='->', color='#aaa', lw=1))
ax1.annotate('Apr dip: Easter', xy=(8, monthly['count_of_appointments'].iloc[8])
             xytext=(8.3, 163000), fontsize=9, color='#888',
             arrowprops=dict(arrowstyle='->', color='#aaa', lw=1))

plt.tight_layout()
plt.savefig('eap_growth_over_time.png', dpi=150, bbox_inches='tight')
print('Saved: eap_growth_over_time.png')
```

```
Saved: eap_growth_over_time.png
```



Question 9 continued Extended Access by ICB

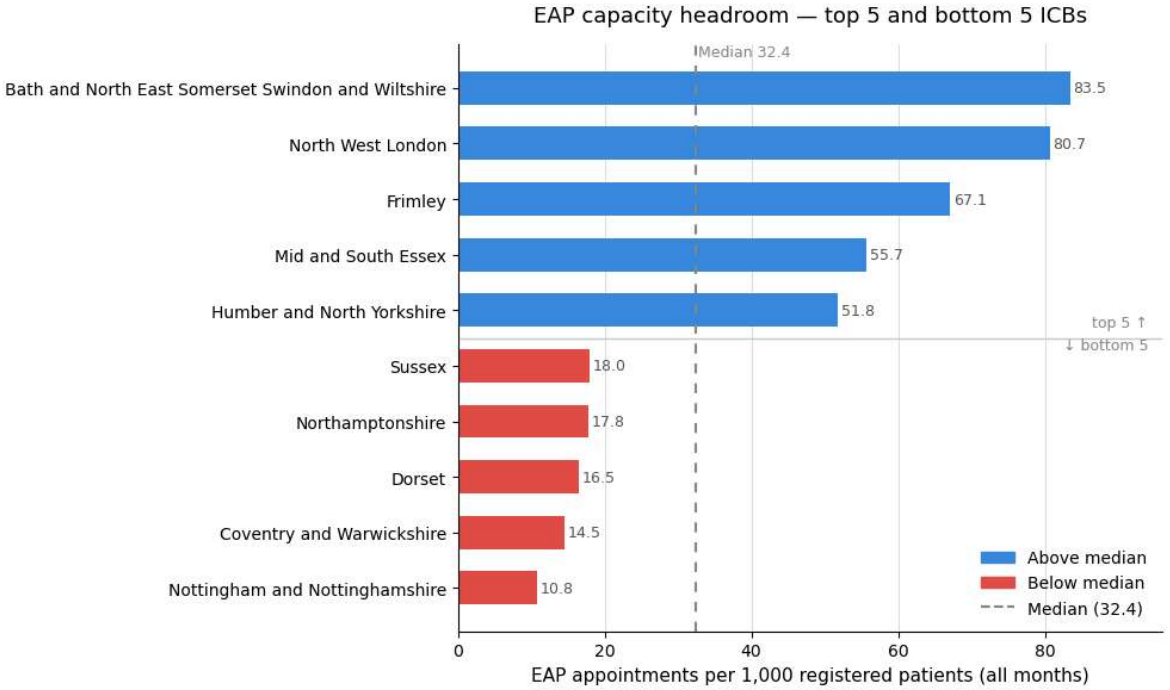
In [70]:

```
eap_by_icb = eap.groupby('icb_ons_code')['count_of_appointments'].sum().reset_in
patients_by_icb = nc.groupby('icb_ons_code')['total_patients'].max().reset_index

icb_names = (
    nc[['icb_ons_code', 'sub_icb_location_name']]
```

```
    .drop_duplicates('icb_ons_code')
    .reset_index(drop=True)
)

merged = eap_by_icb.merge(patients_by_icb, on='icb_ons_code').merge(icb_names, o
merged['appts_per_1000'] = merged['count_of_appointments'] / merged['total_patie
merged = merged.sort_values('appts_per_1000', ascending=False).reset_index(drop=

top5_idx = list(range(5))
bottom5_idx = list(range(len(merged) - 5, len(merged)))
subset = merged.iloc[top5_idx + bottom5_idx].reset_index(drop=True)
subset = subset.sort_values('appts_per_1000', ascending=True).reset_index(drop=T

def short_name(name):
    name = name.split(' ICB')[0].replace('NHS ', '')
    return name

subset['label'] = subset['sub_icb_location_name'].apply(short_name)

median = merged['appts_per_1000'].median()


BLUE = '#378ADD'
RED = '#E24B4A'
GRAY = '#888780'
AMBER = '#BA7517'

colors = [BLUE if v >= median else RED for v in subset['appts_per_1000']]

fig, ax = plt.subplots(figsize=(10, 6))

bars = ax.barh(subset['label'], subset['appts_per_1000'], color=colors, height=0

for bar, val in zip(bars, subset['appts_per_1000']):
    ax.text(val + 0.4, bar.get_y() + bar.get_height() / 2,
            f'{val:.1f}', va='center', fontsize=9, color='#555')

xmax = subset['appts_per_1000'].max() * 1.15
ax.axvline(median, color=GRAY, linewidth=1.5, linestyle=(0, (4, 4)))
ax.text(median + 0.3, ax.get_ylim()[1] if ax.get_ylim()[1] > 0 else 9.3,
        f'Median {median:.1f}', fontsize=9, color=GRAY, va='top')

ax.set_xlim(0, xmax)
ax.set_xlabel('EAP appointments per 1,000 registered patients (all months)', fon
ax.set_title('EAP capacity headroom — top 5 and bottom 5 ICBs', fontsize=13,
             fontweight='normal', pad=14)

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.grid(axis='x', color='#e0e0e0', linewidth=0.7)
ax.set_axisbelow(True)
ax.tick_params(axis='y', labelsize=10)
ax.tick_params(axis='x', labelsize=10)

ax.axhline(y=4.5, color='#cccccc', linewidth=1, linestyle='-')
ax.text(xmax * 0.98, 4.7, 'top 5 ↑', fontsize=9, color='#888', ha='right')
ax.text(xmax * 0.98, 4.3, '↓ bottom 5', fontsize=9, color='#888', ha='right')

legend_handles = [
    plt.Rectangle((0, 0), 1, 1, color=BLUE, label='Above median'),
```

```
    plt.Rectangle((0, 0), 1, 1, color=RED, label='Below median'),
    mlines.Line2D([], [], color=GRAY, linewidth=1.5, linestyle='--', label=f'Med
]
ax.legend(handles=legend_handles, fontsize=10, frameon=False, loc='lower right')

plt.tight_layout()
plt.savefig('eap_capacity_headroom.png', dpi=150, bbox_inches='tight')
print('Saved: eap_capacity_headroom.png')
```

```
Saved: eap_capacity_headroom.png
```



In [ ]:

Data Exploration - Tweets File

Question 1: What are the most common hashtags?

In [71]:

```
# Create a new DataFrame containing only the text.
tw_text = tw[['tweet_full_text']].copy()

# Loop through the messages, and create a list of values containing the # symbol
# Initialize an empty list to store the hashtags
hashtag_list = []

# Loop through each text message in the DataFrame
for text in tw_text['tweet_full_text']:
    # Ensure the row is a string and handle missing values
    if isinstance(text, str):
        # Split the text into individual words
        words = text.split()

        # Filter for words that contain the '#' symbol
        for word in words:
            if '#' in word:
```

```
                hashtag_list.append(word.lower())

# Convert the series to a DataFrame in preparation for visualisation.
hashtags_df = pd.DataFrame(hashtag_list, columns=['hashtag'])

# Group, count, and sort the hashtags for easy visualization
tw_hashtags = hashtags_df['hashtag'].value_counts().reset_index()
tw_hashtags.columns = ['hashtag', 'count']

# Display records where the count is larger than 30.
tw_popular_hashtags = tw_hashtags[tw_hashtags['count'] > 30]

# Display the filtered DataFrame
tw_popular_hashtags
```

Out[71]:

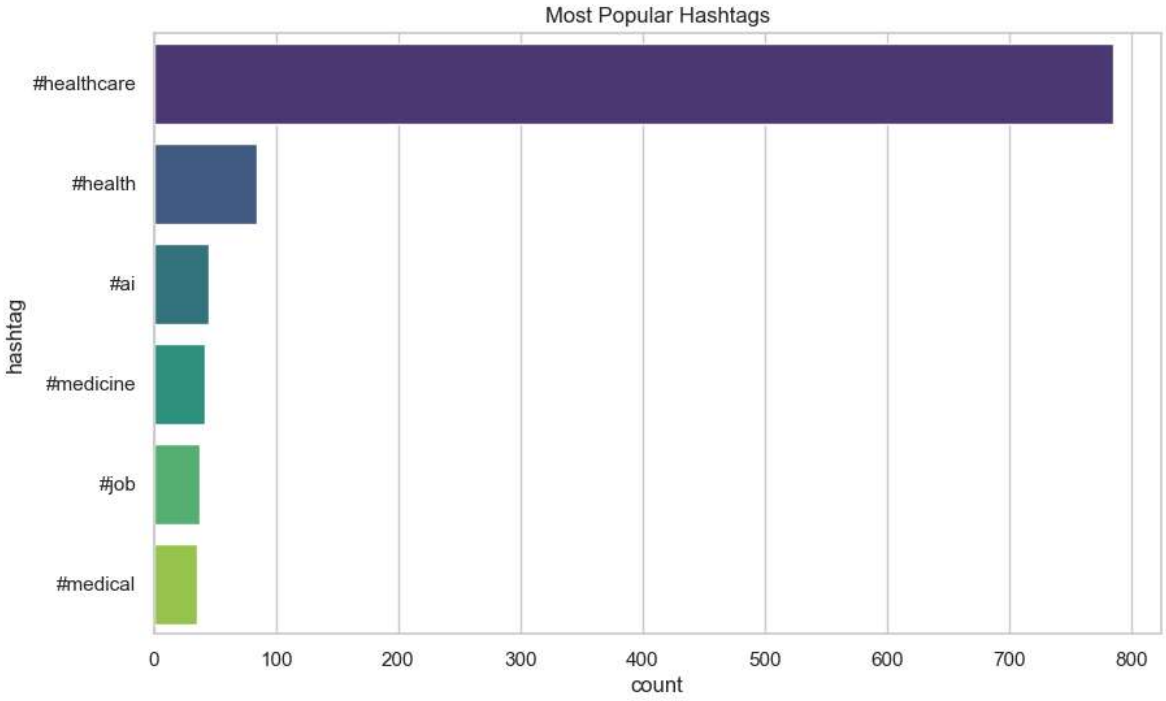
	hashtag	count
0	#healthcare	785
1	#health	84
2	#ai	45
3	#medicine	42
4	#job	38
5	#medical	35

In [72]:

```
# Create a Seaborn barplot indicating records with a count >25 records.

# Set the visual style and figure size
sns.set_theme(style="whitegrid")
plt.figure(figsize=(10, 6))

# Create a horizontal barplot (easier to read hashtag names)
sns.barplot(
    data=tw_popular_hashtags,
    x='count',
    y='hashtag',
    hue='hashtag', # Colors each bar individually
    palette='viridis',
    legend=False
).set_title("Most Popular Hashtags")
```

Out[72]:

```
Text(0.5, 1.0, 'Most Popular Hashtags')
```



Question 2: Which tweets had the highest 'favouriting' ?

In [73]:
```
favourite_tweets = tw["tweet_favorite_count"].value_counts()
favourite_tweets
```

Out[73]:
```
tweet_favorite_count
0     1027
1       91
2       16
3       13
4        7
5        5
6        2
17       1
12       1
10       1
8        1
13       1
11       1
7        1
20       1
28       1
14       1
18       1
9        1
42       1
Name: count, dtype: int64
```

In [74]:
```
# Extract tweets with more than 20 favourites
favourite_tweets = tw[tw["tweet_favorite_count"] > 20]
favourite_tweets = favourite_tweets.reset_index(drop=True)
favourite_tweets
```

Out[74]:

	tweet_id	tweet_full_text	tweet_entities_hashtags	tweet_retwe
0	1567582427719282689	You ready for $JCO @_JennyCo ❤️ \n\n#Healthcare...	#Healthcare	
1	1567583855422611461	Lipid-Lowering Drugs\n\n#TipsForNewDocs #MedEd...	#TipsForNewDocs, #MedEd, #MedTwitter, #medicin...	

Question 3: Which tweets had the highest 'retweeting' ?

In [75]:
```
reposted_tweets = tw["tweet_retweet_count"].value_counts()
reposted_tweets
```

Out[75]:
```
tweet_retweet_count
0      526
1      215
2      114
3       70
5       35
4       27
7       18
12      16
8       15
73      14
9       13
6       12
208     12
35      10
37       6
11       6
10       5
53       5
44       4
150      4
63       4
76       3
85       3
41       3
62       3
207      3
68       3
78       2
23       2
24       2
72       2
16       2
13       1
49       1
48       1
15       1
107      1
14       1
79       1
20       1
39       1
19       1
303      1
57       1
40       1
54       1
169      1
Name: count, dtype: int64
```

In [76]:
```
# Extract tweets with more than 20 reposts
reposted_tweets = tw[tw["tweet_retweet_count"] > 20]
reposted_tweets = reposted_tweets.reset_index(drop=True)
reposted_tweets
```

Out[76]:

	tweet_id	tweet_full_text	tweet_entities_hashtags	tweet_retweet_cou
0	1567582787070304256	RT @OntarioGreens: $10 billion can go a long w...	#Healthcare	3
1	1567581274088566785	RT @khaleejtimes: .@BurjeelHoldings, a private...	NaN	20
2	1567581257823027201	RT @SoniaFurstenau: People in BC are dying of ...	NaN	4
3	1567580783598075905	RT @ricochet_en: Health care in Canada is coll...	NaN	7
4	1567580221037236224	RT @SoniaFurstenau: Germany has one of the top...	#healthcare	8
...	...	...	...	
90	1567585759036669958	RT @khaleejtimes: .@BurjeelHoldings, a private...	NaN	20
91	1567585317498003456	RT @khaleejtimes: .@BurjeelHoldings, a private...	NaN	20
92	1567583678129393665	RT @ricochet_en: Health care in Canada is coll...	NaN	7
93	1567583655689854978	RT @khaleejtimes: .@BurjeelHoldings, a private...	NaN	20
94	1567583062354968576	RT @khaleejtimes: .@BurjeelHoldings, a private...	NaN	20

95 rows × 5 columns

In [77]:

```
# Sort by repost count (highest first)
reposted_tweets = reposted_tweets.sort_values(by='tweet_retweet_count', ascendin

# Remove duplicate tweets, keeping the highest repost version
reposted_unique_tweets = reposted_tweets.drop_duplicates(subset='tweet_full_text

# Optional: reset index
reposted_unique_tweets = reposted_unique_tweets.reset_index(drop=True)
```

```
reposted_unique_tweets
```

Out[77]:

	tweet_id	tweet_full_text	tweet_entities_hashtags	tweet_retweet_cou
0	1567649792897032192	RT @UltimaLionsDen: Temitope is looking to boo...	NaN	3
1	1567594286056603661	RT @khaleejtimes: .@BurjeelHoldings, a private...	NaN	2
2	1567585858546532355	RT @imedverse: I.V Drug Calculations Cheat She...	#TipsForNewDocs, #MedEd, #MedTwitter, #medicin...	1
3	1567586472009601026	RT @Khulood_Almani: 🔟 #Applications of #AI in #...	#Applications, #AI, #healthcare, #digitalhealt...	1
4	1567609182001545221	RT @Khulood_Almani: #Healthcare #DigitalTransf...	#Healthcare, #DigitalTransformation, #digitalh...	1
5	1567577960257056768	RT @SoniaFurstenau: Germany has one of the top...	#healthcare	
6	1567622742580551681	RT @imedverse: Features of Lung Diseases\n\n#p...	#pulmonary, #pulmtwitter, #pulmonology, #pulmo...	
7	1567609206273937411	RT @Khulood_Almani: 🪑 This #wearable Chair from...	#wearable, #wearable	
8	1567577622393270277	RT @AWSCloudIndia: How is #data revolutionizin...	#data, #healthcare	
9	1567621185684242432	RT @healthcare_NFT_: Really excited to be work...	#HealthCare	
10	1567613991202996224	RT @ricochet_en: Health care in Canada is coll...	NaN	
11	1567652191132913664	RT @healthcare_NFT_: In addition to announceme...	NaN	
12	1567618611157630977	RT @healthcare_NFT_: The #HealthCare Heroes jo...	#HealthCare	

	tweet_id	tweet_full_text	tweet_entities_hashtags	tweet_retweet_cou
13	1567620032737464320	RT @holisticchamber: Even Time Magazine has so...	NaN	
14	1567651478332112896	RT @LaurelCoons: Your Medical Data:\n\n👏It's y...	NaN	
15	1567577937519562752	RT @inery_naveen: #Security breaches will cont...	#Security, #healthcare, #centralized	
16	1567618537551691778	RT @Ronald_vanLoon: A global team of scientist...	#AI	
17	1567578091807191040	RT @SoniaFurstenau: Germany's Pop. is 75M, Can...	#healthcare	
18	1567611327002148865	RT @1DavidClarke: 55% are willing to engage wi...	#AI, #Robotics, #Healthcare, #PwC, #4IR, #IoT	
19	1567612691325554688	RT @mark_clinton75: Goofing off onset for a He...	NaN	
20	1567649448230109185	RT @Ronald_vanLoon: Rehabilitation #Robotic Gl...	#Robotic, #MI, #Robotics, #Tech, #Technology, ...	
21	1567654746961416198	RT @KatharineSmart: We need more people on our...	#healthcare, #doctors	
22	1567645217549090816	RT @SoniaFurstenau: People in BC are dying of ...	NaN	
23	1567582787070304256	RT @OntarioGreens: $10 billion can go a long w...	#Healthcare	
24	1567600181666783238	RT @JeffWaltersSask: A good read on what nurse...	#saskatchewan	

	tweet_id	tweet_full_text	tweet_entities_hashtags	tweet_retweet_cou
25	1567616287060234240	RT @CURE_Ecosystem: EXCLUSIVE footage. Just fo...	NaN	
26	1567614270078160897	RT @CurieuxExplorer: Transforming the rehabili...	NaN	
27	1567646962240847873	RT @UPMGlobal: Nanorobots inside our bodies an...	NaN	
28	1567648600133443585	RT @MelissaLMRogers: CANADA is the only countr...	NaN	

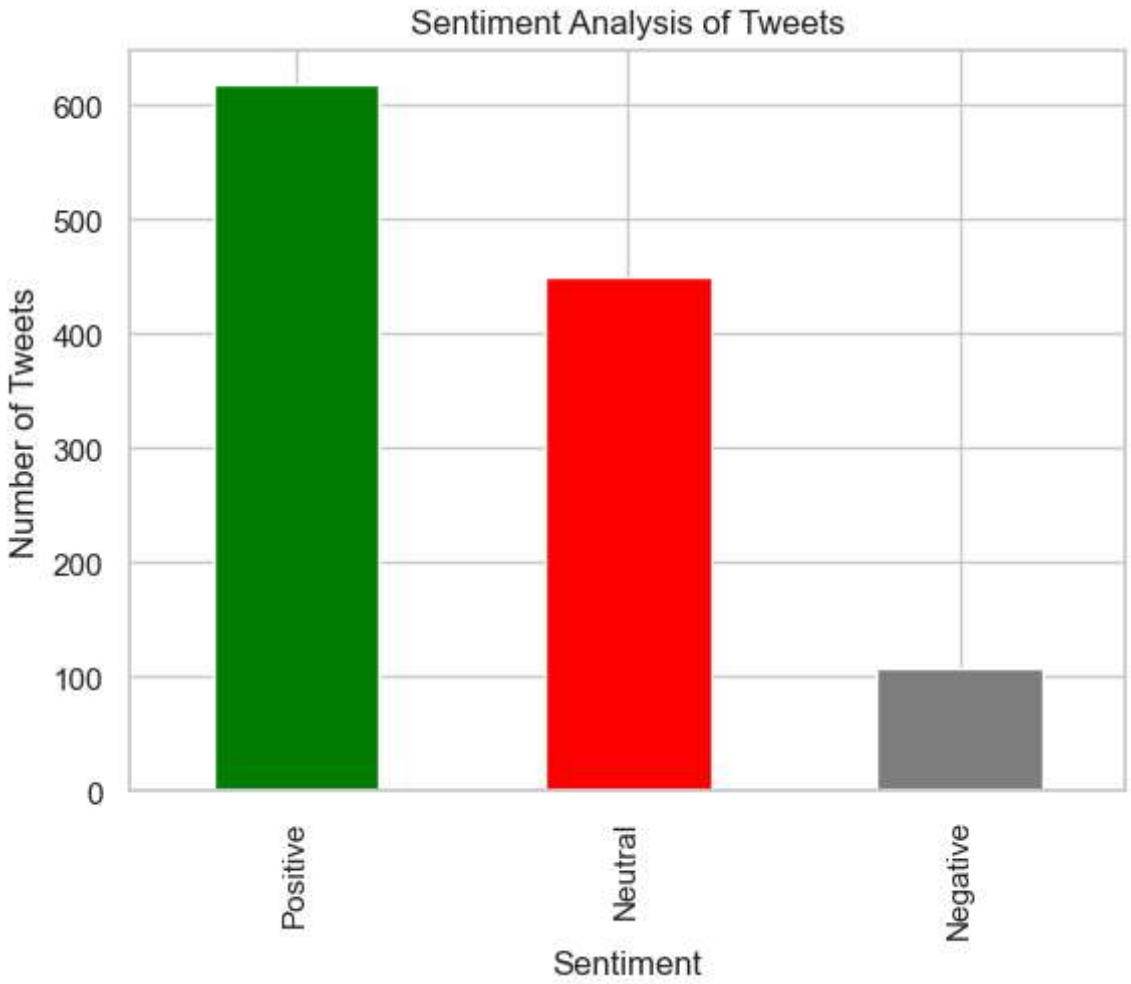
Question 5: Are tweets generally positive or negative?

In [78]:
```
#perform sentinent analysis using textblob
def analyze_sentiment(text):
    if not isinstance(text, str):
        return "Neutral"

    score = TextBlob(text).sentiment.polarity
    if score > 0:
        return "Positive"
    elif score < 0:
        return "Negative"
    else:
        return "Neutral"

# 2. Apply it directly to your column
tw['sentiment'] = tw['tweet_full_text'].apply(analyze_sentiment)
tw.head(5)
```

Out[78]:

	tweet_id	tweet_full_text	tweet_entities_hashtags	tweet_retweet_coun
0	1567629223795527681	As Arkansas' first Comprehensive Stroke Certif...	#Healthcare	
1	1567582846612553728	RT @AndreaGrammer: Work-life balance is at the...	#PremiseHealth, #hiring	
2	1567582787070304256	RT @OntarioGreens: $10 billion can go a long w...	#Healthcare	3
3	1567582767625428992	RT @modrnhealthcr: 🚨 #NEW: 🚨 Insurance companies...	#NEW	
4	1567582720460570625	ICYMI: Our recent blogs on Cybersecurity in Ac...	#blogs, #digitaltransformation, #cybersecurity...	

In [79]:
```
# visualise sentiment analysis

tw['sentiment'].value_counts().plot(kind='bar', color=['green', 'red', 'gray'])
plt.title(f"Sentiment Analysis of Tweets")
plt.xlabel("Sentiment")
plt.ylabel("Number of Tweets")
plt.show()
```


Sentiment Analysis of Tweets
Number of Tweets
0
100
200
300
400
500
600
Positive
Neutral
Negative
Sentiment

In [ ]: