

Predicting Family Planning Demand and Optimizing Service Delivery in Kenya

Project Overview

This project aims to enhance family planning (FP) service delivery efficiency by leveraging data-driven approaches, specifically machine learning (ML) using the CRISP-DM methodology. The goal is to improve maternal and child health outcomes, reduce unmet need for family planning, and contribute to achieving national and global development goals related to reproductive health.

Problem Statement

Kenya faces significant strides in increasing access to family planning services, yet a substantial unmet need for family planning remains. According to the 2022 Kenya Demographic and Health Survey (KDHS), the total unmet need for family planning is 15%, with 10% for spacing and 5% for limiting births. This indicates that a significant portion of the population desires to space or limit births but is not using any contraceptive method. Traditional methods often show imbalances, with a heavy reliance on short-acting methods, leading to unstable uptake of long-acting reversible contraceptives (LARCs) and permanent methods. This disparity can lead to higher discontinuation rates and continued unmet need. Supply chain inefficiencies, commodity stock-outs, inadequate healthcare worker training, and uneven distribution of resources exacerbate these issues, hindering effective service delivery.

Stakeholders

- **Government of Kenya:** Ministry of Health (MoH), especially the National Family Planning Coordinated Implementation Plan (NFPICIP) and Kenya Health Information System (KHIS) initiatives.
- **Policymakers and Donors:** Need evidence-based advocacy for resource mobilization and investment.
- **Healthcare Providers:** Frontline health workers providing FP services.
- **Women of Reproductive Age:** Direct beneficiaries of improved FP services.
- **Local Communities:** Impacted by and involved in FP service delivery.

Key Statistics

- **Total Unmet Need for Family Planning (2022 KDHS):** 15%
 - **Unmet Need for Spacing:** 10%
 - **Unmet Need for Limiting Births:** 5%
- **Married Women Aged 15-49 Rising to 57% in 2022 (DRS 2022):** Modern contraceptive prevalence rate (mCPR).
- **Number of new clients for FP method band and the continuation rate:** Key data points for predicting future demand.

Key Analytics Questions

- How many new clients are expected for injectables in a County next quarter?
- What is the projected demand for different family planning methods (injectables, pills, condoms, implants, IUD, sterilization) at various geographical (e.g., county) and temporal (e.g., quarterly, annual) granularities?
- How can we optimize resource allocation (commodities, equipment, staffing) to meet projected demand and minimize wastage?
- How can predictive analytics identify potential stock-outs or oversupply of specific FP commodities in different locations?
- Which regions or demographics are most underserved in terms of family planning access and uptake?

Objectives

- **Quantitatively forecast the demand for specific family planning methods:** This includes predicting the continuation rates of users for each method at defined geographical and temporal scales.
- **Enable proactive resource allocation:** This involves optimizing the distribution of commodities, equipment, and staffing to reduce stock-outs, minimize wastage, and improve targeted interventions.
- **Improve method continuation:** By understanding demand and improving service delivery, the project aims to reduce discontinuation rates and increase sustained use of FP methods.
- **Provide evidence-based insights:** Support policymakers and donors in making informed decisions regarding resource mobilization and investment in family planning.

Metrics of Importance to Focus On

- **Accuracy of Demand Forecasts:** Measured by comparing predicted demand with actual uptake for various FP methods at different geographical and temporal levels (e.g., Mean Absolute Error, Root Mean Squared Error).
- **Commodity Stock-out Rates:** Reduction in the number or duration of stock-outs for essential family planning commodities.
- **Resource Utilization Efficiency:** Metrics related to optimal allocation and reduced wastage of commodities, equipment, and human resources.
- **Method Continuation Rates:** Increase in the percentage of users who continue using a specific family planning method over a defined period (e.g., 12-month continuation rate).
- **Unmet Need for Family Planning:** Contribution to the reduction of the national unmet need for family planning.
- **Client Satisfaction:** Indirectly improved through better access and availability of preferred methods.
- **Healthcare Worker Productivity:** Optimized allocation of staff to meet demand efficiently.

Preliminaries

Importing Python Libraries

```
In [198]: #Import necessary Libraries for data manipulation, visualization, and machine Learning
# Data handling and manipulation
import pandas as pd                                     # For dataframes and data manipulation
import numpy as np                                      # For numerical operations and arrays

# Visualization libraries
import matplotlib.pyplot as plt                         # For basic plotting
import seaborn as sns                                    # For advanced statistical visualizations

# Suppress warning messages
import warnings                                         # To filter out warnings during execution

# Model selection and evaluation
from sklearn.model_selection import train_test_split, cross_val_score, TimeSeriesSplit
# train_test_split: splits data into training and test sets
# cross_val_score: performs cross-validation
# TimeSeriesSplit: cross-validation for time series data such as this FP data
from sklearn.model_selection import GridSearchCV

# Data preprocessing
from sklearn.preprocessing import StandardScaler, MinMaxScaler, OneHotEncoder
# StandardScaler/MinMaxScaler: scale numerical features
# OneHotEncoder: encode categorical variables

# Metrics for model evaluation
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
# MAE, MSE, R2: performance metrics for regression models

# Combine preprocessing steps
from sklearn.compose import ColumnTransformer # Apply different preprocessing to different columns

# Build machine learning workflows
from sklearn.pipeline import Pipeline          # Chain preprocessing and modeling

# Handle missing data
from sklearn.impute import SimpleImputer       # Fill missing values

# Regression models
from sklearn.linear_model import LinearRegression, Ridge                               # Linear regression
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.multioutput import MultiOutputRegressor
from xgboost import XGBRegressor

# RandomForestRegressor: ensemble method using decision trees
# GradientBoostingRegressor: boosting method for better accuracy
```

Data Loading

```
In [2]: # from google.colab import drive
# drive.mount('/content/drive') # Mount Google Drive

# base_path = "/content/drive/MyDrive/data/"

# # File paths
# population_data_path = base_path + "ke_fp_population_data.csv"
# service_data_path = base_path + "ke_fp_service_data.csv"
# benchmarks_data_path = base_path + "ke_fp_benchmarks_data.csv"
# commodity_data_path = base_path + "ke_fp_commodity_data.csv"

# # Attempt to read with 'Latin1' encoding
# df_population = pd.read_csv(population_data_path, encoding='Latin1')
# df_service = pd.read_csv(service_data_path, encoding='Latin1')
# df_benchmarks = pd.read_csv(benchmarks_data_path, encoding='Latin1')
# df_commodity = pd.read_csv(commodity_data_path, encoding='Latin1')
```

In [3]: # Data Loading

```
try:
    # Define paths
    data_dir = "data/"
    benchmarks_dir = "data/benchmarks/"

    population_data_path = f"{data_dir}ke_fp_population_data.csv"
    service_data_path = f"{data_dir}ke_fp_service_data.csv"
    commodity_data_path = f"{data_dir}ke_fp_commodity_data.csv"

    benchmarks_core_health_workforce_data_path = f"{benchmarks_dir}ke_fp_benchmark_
    benchmarks_demand_satisfied_data_path = f"{benchmarks_dir}ke_fp_benchmarks_De
    benchmarks_mCPR_data_path = f"{benchmarks_dir}ke_fp_benchmarks_mCPR.csv"
    benchmarks_teen_pregnancy_data_path = f"{benchmarks_dir}ke_fp_benchmarks_Teen
    benchmarks_unmet_need_data_path = f"{benchmarks_dir}ke_fp_benchmarks_Total_Ur

    # Load CSVs with fallback encoding
    df_population = pd.read_csv(population_data_path, encoding='latin1')
    df_service = pd.read_csv(service_data_path, encoding='latin1')
    df_commodity = pd.read_csv(commodity_data_path, encoding='latin1')

    # Load benchmark-specific CSVs
    df_core_health_workforce = pd.read_csv(benchmarks_core_health_workforce_data_
    df_demand_satisfied = pd.read_csv(benchmarks_demand_satisfied_data_path, encod
    df_mcpr = pd.read_csv(benchmarks_mCPR_data_path, encoding='latin1')
    df_teenage_pregnancy = pd.read_csv(benchmarks_teen_pregnancy_data_path, encod
    df_unmet_need = pd.read_csv(benchmarks_unmet_need_data_path, encoding='latin1

    # Success Logs
    print("Datasets loaded successfully:")
    print(f"{population_data_path} shape: {df_population.shape}")
    print(f"{service_data_path} shape: {df_service.shape}")
    print(f"{commodity_data_path} shape: {df_commodity.shape}")
    print(f"{benchmarks_core_health_workforce_data_path} shape: {df_core_health_w
    print(f"{benchmarks_demand_satisfied_data_path} shape: {df_demand_satisfied.s
    print(f"{benchmarks_mCPR_data_path} shape: {df_mcpr.shape}")
    print(f"{benchmarks_teen_pregnancy_data_path} shape: {df_teenage_pregnancy.sh
    print(f"{benchmarks_unmet_need_data_path} shape: {df_unmet_need.shape}")

except FileNotFoundError as e:
    print("✖ Error: One or more CSV files were not found.")
    print("🔍 Please ensure all expected files are in their respective folders.")
    print(e)
except Exception as e:
    print("❗ An unexpected error occurred while loading the datasets:")
    print(e)
```

```
Datasets loaded successfully:  
data/ke_fp_population_data.csv shape: (517, 19)  
data/ke_fp_service_data.csv shape: (6204, 60)  
data/ke_fp_commodity_data.csv shape: (2480, 50)  
data/benchmarks/ke_fp_benchmarks_core_health_workforce.csv shape: (47, 8)  
data/benchmarks/ke_fp_benchmarks_Demand_Satisfied.csv shape: (47, 8)  
data/benchmarks/ke_fp_benchmarks_mCPR.csv shape: (47, 8)  
data/benchmarks/ke_fp_benchmarks_Teenage_Pregnancy_rate.csv shape: (47, 8)  
data/benchmarks/ke_fp_benchmarks_Total_Unmet_Need_MW.csv shape: (47, 8)
```

1. Data Understanding

a) Data Cleaning

This involved;

- Standardization of the column names
- Renaming the columns
- Dropping empty and unwanted columns
- Handling missing values, duplicates and outliers

1. ke_fp_service_data.csv

```
In [4]: # Make a copy of the data  
df_service1=df_service.copy()
```

```
In [5]: # from google.colab import drive  
# drive.mount('/content/drive')
```

```
In [6]: # Preview the data  
df_service1.head()
```

Out[6]:

| | periodid | periodname | periodcode | perioddescription | orgunitlevel1 | orgunitlevel2 | organisationunit |
|---|----------|------------|------------|-------------------|---------------|-------------------|------------------|
| 0 | 201404 | April 2014 | 201404 | | Kenya | Turkana County | kphDeKCIF |
| 1 | 201404 | April 2014 | 201404 | | Kenya | Nandi County | t0J75eHKx |
| 2 | 201404 | April 2014 | 201404 | | Kenya | West Pokot County | XWALbfAPa |
| 3 | 201404 | April 2014 | 201404 | | Kenya | Bomet County | HMNARUV2CV |
| 4 | 201404 | April 2014 | 201404 | | Kenya | Nairobi County | jkG3zaihd: |

| | periodid | periodname | periodcode | perioddescription | orgunitlevel1 | orgunitlevel2 | organisationunit |
|---|----------|------------|------------|-------------------|---------------|-------------------|------------------|
| 0 | 201404 | April 2014 | 201404 | | Kenya | Turkana County | kphDeKCIF |
| 1 | 201404 | April 2014 | 201404 | | Kenya | Nandi County | t0J75eHKx |
| 2 | 201404 | April 2014 | 201404 | | Kenya | West Pokot County | XWALbfAPa |
| 3 | 201404 | April 2014 | 201404 | | Kenya | Bomet County | HMNARUV2CV |
| 4 | 201404 | April 2014 | 201404 | | Kenya | Nairobi County | jkG3zaihd: |

5 rows × 60 columns



```
In [7]: # Explore the data  
df_service1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6204 entries, 0 to 6203
Data columns (total 60 columns):
 #   Column          Non-Null Count  Dtype  
--- 
 0   periodid        6204 non-null    int64  
 1   periodname      6204 non-null    object  
 2   periodcode      6204 non-null    int64  
 3   perioddescription 6204 non-null    object  
 4   orgunitlevel1   6204 non-null    float64
 5   orgunitlevel2   6204 non-null    object  
 6   organisationunitid 6204 non-null    object  
 7   organisationunitname 6204 non-null    object  
 8   organisationunitcode 6204 non-null    object  
 9   organisationunitdescription 6204 non-null    float64
 10  Estimated Number of Pregnant Women 6192 non-null    float64
 11  FP Attendance New clients       6204 non-null    float64
 12  FP Attendance Re-visits      6204 non-null    float64
 13  MOH 711 Adolescent 10-14 yrs Receiving FP Services New clients 4095 non-null    float64
 14  MOH 711 Adolescent 10-14 yrs Receiving FP Services Re-visits 3327 non-null    float64
 15  MOH 711 Adolescent 15-19 Yrs Receiving FP Services New clients 4980 non-null    float64
 16  MOH 711 Adolescent 15-19 Yrs Receiving FP Services Re-visits 4956 non-null    float64
 17  MOH 711 Adolescent 20-24 Yrs Receiving FP Services New clients 4972 non-null    float64
 18  MOH 711 Adolescent 20-24 Yrs Receiving FP Services Re-visits 4964 non-null    float64
 19  MOH 711 Client receiving Male condoms New clients       6204 non-null    int64  
 20  MOH 711 Client receiving Male condoms Re-visits      6004 non-null    float64
 21  MOH 711 Clients Counselling Natural Family Planning New clients 5459 non-null    float64
 22  MOH 711 Clients Counselling Natural Family Planning Re-visits 3309 non-null    float64
 23  MOH 711 Clients receiving Female Condoms New clients     4141 non-null    float64
 24  MOH 711 Clients receiving Female Condoms Re-visits      2826 non-null    float64
```

```
25 MOH 711 Emergency contraceptive pill New clients
4837 non-null float64
26 MOH 711 Emergency contraceptive pill Re-visits
3402 non-null float64
27 MOH 711 Pills Combined oral contraceptive New clients
6201 non-null float64
28 MOH 711 Pills Combined oral contraceptive Re-visits
6194 non-null float64
29 MOH 711 Pills progestin only New clients
6193 non-null float64
30 MOH 711 Pills progestin only Re-visits
6165 non-null float64
31 MOH 711 Rev 2020_Adults 25+ receiving FP Services New clients
2415 non-null float64
32 MOH 711 Rev 2020_Adults 25+ receiving FP Services Re-visits
2408 non-null float64
33 MOH 711 Rev 2020_Clients given cycle beads New clients
1188 non-null float64
34 MOH 711 Rev 2020_Clients given cycle beads Re-visits
84 non-null float64
35 MOH 711 Rev 2020_Clients receiving post abortion FP New clients
2007 non-null float64
36 MOH 711 Rev 2020_Clients receiving post abortion FP Re-visits
425 non-null float64
37 MOH 711 Rev 2020_FP Injections DMPA- IM New clients
2465 non-null float64
38 MOH 711 Rev 2020_FP Injections DMPA- IM Re-visits
2469 non-null float64
39 MOH 711 Rev 2020_FP Injections DMPA- SC New clients
2133 non-null float64
40 MOH 711 Rev 2020_FP Injections DMPA- SC Re-visits
2108 non-null float64
41 MOH 711 Rev 2020_IUCD Insertion Hormonal 1st Time Insertion
2015 non-null float64
42 MOH 711 Rev 2020_IUCD Insertion Hormonal Re-insertion
1561 non-null float64
43 MOH 711 Rev 2020_IUCD Insertion Non Hormonal 1st Time Insertion
2332 non-null float64
44 MOH 711 Rev 2020_IUCD Insertion Non Hormonal Re-insertion
2094 non-null float64
45 MOH 711 Rev 2020_Implants insertion 1 Rod 1st Time Insertion
2438 non-null float64
46 MOH 711 Rev 2020_Implants insertion 1 Rod Re-insertion
2389 non-null float64
47 MOH 711 Rev 2020_Implants insertion 2 Rod 1st Time Insertion
2434 non-null float64
48 MOH 711 Rev 2020_Implants insertion 2 Rod Re-insertion
2402 non-null float64
49 MOH 711 Rev 2020_Post partum FP 4weeks to 6weeks New clients
2350 non-null float64
50 MOH 711 Rev 2020_Post partum FP 4weeks to 6weeks Re-visits
1428 non-null float64
51 MOH 711 Rev 2020_Post partum FP within 48 Hours New clients
2208 non-null float64
52 MOH 711 Rev 2020_Post partum FP within 48 Hours Re-visits
1199 non-null float64
53 MOH 711 Rev 2020_Voluntary Surgical Contraception Vasectomy 1st Time Inser
```

```
tion    300 non-null    float64
54 MOH 711 Rev 2020_Voluntary Surgical Contraception Vasectomy Re-insertion
5 non-null    float64
55 MOH 711 Rev 2020_Voluntary surgical contraception BTL Ist Time Insertion
1585 non-null    float64
56 MOH 711 Rev 2020_Voluntary surgical contraception BTL Re-insertion
32 non-null    float64
57 Population Growth Rate
5352 non-null    float64
58 Total Population
6204 non-null    float64
59 Women of childbearing age (15â®49yrs)
6204 non-null    float64
dtypes: float64(51), int64(3), object(6)
memory usage: 2.8+ MB
```

In [8]: # Standardize the column names

```
def standardize_col_labels(df):
    def clean_column(col):
        # Remove redundant prefixes
        col = col.replace('MOH 711 Rev ', '')
        col = col.replace('MOH 711 ', '')

        # Formatting
        col = col.strip().lower()          # Convert to lowercase
        col = col.replace(' ', '_')       # Replace spaces with underscores
        col = col.replace('-', '_')       # Replace hyphen with underscores
        col = col.replace('€', '_')
        col = col.replace('•', '_')
        return col

    df.columns = [clean_column(col) for col in df.columns]
    return df

df_service1 = standardize_col_labels(df_service1)
df_service1.columns
```

```
Out[8]: Index(['periodid', 'periodname', 'periodcode', 'periodeescription',
   'orgunitlevel1', 'orgunitlevel2', 'organisationunitid',
   'organisationunitname', 'organisationunitcode',
   'organisationunitdescription', 'estimated_number_of_pregnant_women',
   'fp_attendance_new_clients', 'fp_attendance_re_visits',
   'adolescent_10_14_yrs_receiving_fp_services_new_clients',
   'adolescent_10_14_yrs_receiving_fp_services_re_visits',
   'adolescent_15_19_yrs_receiving_fp_services_new_clients',
   'adolescent_15_19_yrs_receiving_fp_services_re_visits',
   'adolescent_20_24_yrs_receiving_fp_services_new_clients',
   'adolescent_20_24_yrs_receiving_fp_services_re_visits',
   'client_receiving_male_condoms_new_clients',
   'client_receiving_male_condoms_re_visits',
   'clients_counselled_natural_family_planning_new_clients',
   'clients_counselled_natural_family_planning_re_visits',
   'clients_receiving_female_condoms_new_clients',
   'clients_receiving_female_condoms_re_visits',
   'emergency_contraceptive_pill_new_clients',
   'emergency_contraceptive_pill_re_visits',
   'pills_combined_oral_contraceptive_new_clients',
   'pills_combined_oral_contraceptive_re_visits',
   'pills_progestin_only_new_clients', 'pills_progestin_only_re_visits',
   '2020_adults_25+_receiving_fp_services_new_clients',
   '2020_adults_25+_receiving_fp_services_re_visits',
   '2020_clients_given_cycle_beads_new_clients',
   '2020_clients_given_cycle_beads_re_visits',
   '2020_clients_receiving_post_abortion_fp_new_clients',
   '2020_clients_receiving_post_abortion_fp_re_visits',
   '2020_fp_injections_dmpa_im_new_clients',
   '2020_fp_injections_dmpa_im_re_visits',
   '2020_fp_injections_dmpa_sc_new_clients',
   '2020_fp_injections_dmpa_sc_re_visits',
   '2020_iucd_insertion_hormonal_ist_time_insertion',
   '2020_iucd_insertion_hormonal_re_insertion',
   '2020_iucd_insertion_non_hormonal_ist_time_insertion',
   '2020_iucd_insertion_non_hormonal_re_insertion',
   '2020_implants_insertion_1_rod_ist_time_insertion',
   '2020_implants_insertion_1_rod_re_insertion',
   '2020_implants_insertion_2_rod_ist_time_insertion',
   '2020_implants_insertion_2_rod_re_insertion',
   '2020_post_parturm_fp_4weeks_to_6weeks_new_clients',
   '2020_post_parturm_fp_4weeks_to_6weeks_re_visits',
   '2020_post_parturm_fp_within_48_hours_new_clients',
   '2020_post_parturm_fp_within_48_hours_re_visits',
   '2020_voluntary_surgical_contraception_vasectomy_ist_time_insertion',
   '2020_voluntary_surgical_contraception_vasectomy_re_insertion',
   '2020_voluntary_surgical_contraception_btl_ist_time_insertion',
   '2020_voluntary_surgical_contraception_btl_re_insertion',
   'population_growth_rate', 'total_population',
   'women_of_childbearing_age_(15_49yrs)'],
  dtype='object')
```

In [9]: # Rename column names

```
name_map = {
    'periodcode': 'year_month',
    'orgunitlevel1': 'country',
    'orgunitlevel2': 'county',
    'organisationunitid': 'uid',
    'organisationunitcode': 'county_code',
    'county_cou': 'county_code',
    'dataname': 'population_indicator'
}
df_service1 = df_service1.rename(columns=name_map)
df_service1
```

Out[9]:

| | periodid | periodname | year_month | perioddescription | country | county | uid | orga |
|------|----------|----------------|------------|-------------------|---------|--------|----------------------|--------------|
| 0 | 201404 | April 2014 | 201404 | | NaN | Kenya | Turkana County | kphDeKClFch |
| 1 | 201404 | April 2014 | 201404 | | NaN | Kenya | Nandi County | t0J75eHKxz5 |
| 2 | 201404 | April 2014 | 201404 | | NaN | Kenya | West Pokot County | XWALbfAPa6n |
| 3 | 201404 | April 2014 | 201404 | | NaN | Kenya | Bomet County | HMNARUV2CW4 |
| 4 | 201404 | April 2014 | 201404 | | NaN | Kenya | Nairobi County | jkG3zaihdSs |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 6199 | 202409 | September 2024 | 202409 | | NaN | Kenya | Isiolo County | bzOfj0iwfDH |
| 6200 | 202409 | September 2024 | 202409 | | NaN | Kenya | Trans Nzoia County | mThvosEfIAU |
| 6201 | 202409 | September 2024 | 202409 | | NaN | Kenya | Nakuru County | ob6SxuRcqU4 |
| 6202 | 202409 | September 2024 | 202409 | | NaN | Kenya | Tharaka Nithi County | T4urHM47nlm |
| 6203 | 202409 | September 2024 | 202409 | | NaN | Kenya | Nyeri County | ptWVfaCIldVx |

6204 rows × 60 columns



```
In [10]: # Drop columns where all values are null
df_service1=df_service1.dropna(axis=1, how='all')
df_service1
```

Out[10]:

| | periodid | periodname | year_month | country | county | uid | organisationunitname |
|------|----------|----------------|------------|---------|----------------------|--------------|----------------------|
| 0 | 201404 | April 2014 | 201404 | Kenya | Turkana County | kphDeKClFch | Turkana County |
| 1 | 201404 | April 2014 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | Nandi County |
| 2 | 201404 | April 2014 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | West Pokot County |
| 3 | 201404 | April 2014 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | Bomet County |
| 4 | 201404 | April 2014 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | Nairobi County |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 6199 | 202409 | September 2024 | 202409 | Kenya | Isiolo County | bzOfj0iwfDH | Isiolo County |
| 6200 | 202409 | September 2024 | 202409 | Kenya | Trans Nzoia County | mThvosEfIAU | Trans Nzoia County |
| 6201 | 202409 | September 2024 | 202409 | Kenya | Nakuru County | ob6SxuRcqU4 | Nakuru County |
| 6202 | 202409 | September 2024 | 202409 | Kenya | Tharaka Nithi County | T4urHM47nlm | Tharaka Nithi County |
| 6203 | 202409 | September 2024 | 202409 | Kenya | Nyeri County | ptWVfaCIIdVx | Nyeri County |

6204 rows × 58 columns



```
In [11]: # Drop unwanted columns
df_service1=df_service1.drop(columns=['periodid', 'organisationunitname', 'periodname', 'total_population', 'women_of_childbearing'], axis=1)
```

```
In [12]: # Check for missing values
df_service1.isna().sum().sort_values(ascending=False)
```

| | | |
|----------|--|------|
| Out[12]: | 2020_voluntary_surgical_contraception_vasectomy_re_insertion | 6199 |
| | 2020_voluntary_surgical_contraception_btl_re_insertion | 6172 |
| | 2020_clients_given_cycle_beads_re_visits | 6120 |
| | 2020_voluntary_surgical_contraception_vasectomy_ist_time_insertion | 5904 |
| | 2020_clients_receiving_post_abortion_fp_re_visits | 5779 |
| | 2020_clients_given_cycle_beads_new_clients | 5016 |
| | 2020_post_parturm_fp_within_48_hours_re_visits | 5005 |
| | 2020_post_parturm_fp_4weeks_to_6weeks_re_visits | 4776 |
| | 2020_iucd_insertion_hormonal_re_insertion | 4643 |
| | 2020_voluntary_surgical_contraception_btl_ist_time_insertion | 4619 |
| | 2020_clients_receiving_post_abortion_fp_new_clients | 4197 |
| | 2020_iucd_insertion_hormonal_ist_time_insertion | 4189 |
| | 2020_iucd_insertion_non_hormonal_re_insertion | 4110 |
| | 2020_fp_injections_dmpa_sc_re_visits | 4096 |
| | 2020_fp_injections_dmpa_sc_new_clients | 4071 |
| | 2020_post_parturm_fp_within_48_hours_new_clients | 3996 |
| | 2020_iucd_insertion_non_hormonal_ist_time_insertion | 3872 |
| | 2020_post_parturm_fp_4weeks_to_6weeks_new_clients | 3854 |
| | 2020_implants_insertion_1_rod_re_insertion | 3815 |
| | 2020_implants_insertion_2_rod_re_insertion | 3802 |
| | 2020_adults_25+_receiving_fp_services_re_visits | 3796 |
| | 2020_adults_25+_receiving_fp_services_new_clients | 3789 |
| | 2020_implants_insertion_2_rod_ist_time_insertion | 3770 |
| | 2020_implants_insertion_1_rod_ist_time_insertion | 3766 |
| | 2020_fp_injections_dmpa_im_new_clients | 3739 |
| | 2020_fp_injections_dmpa_im_re_visits | 3735 |
| | clients_receiving_female_condoms_re_visits | 3378 |
| | clients_counselled_natural_family_planning_re_visits | 2895 |
| | adolescent_10_14_yrs_receiving_fp_services_re_visits | 2877 |
| | emergency_contraceptive_pill_re_visits | 2802 |
| | adolescent_10_14_yrs_receiving_fp_services_new_clients | 2109 |
| | clients_receiving_female_condoms_new_clients | 2063 |
| | emergency_contraceptive_pill_new_clients | 1367 |
| | adolescent_15_19_yrs_receiving_fp_services_re_visits | 1248 |
| | adolescent_20_24_yrs_receiving_fp_services_re_visits | 1240 |
| | adolescent_20_24_yrs_receiving_fp_services_new_clients | 1232 |
| | adolescent_15_19_yrs_receiving_fp_services_new_clients | 1224 |
| | clients_counselled_natural_family_planning_new_clients | 745 |
| | client_receiving_male_condoms_re_visits | 200 |
| | pills_progestin_only_re_visits | 39 |
| | estimated_number_of_pregnant_women | 12 |
| | pills_progestin_only_new_clients | 11 |
| | pills_combined_oral_contraceptive_re_visits | 10 |
| | pills_combined_oral_contraceptive_new_clients | 3 |
| | client_receiving_male_condoms_new_clients | 0 |
| | fp_attendance_re_visits | 0 |
| | fp_attendance_new_clients | 0 |
| | county_code | 0 |
| | uid | 0 |
| | county | 0 |
| | country | 0 |
| | year_month | 0 |
| | dtype: int64 | |

Missing values were interpreted as 'no service was provided or dataset missing for the organization unit' and filled with 0

```
In [13]: # Fill the missing values with zeros
df_service1 = df_service1.fillna(0)
```

```
In [14]: # Extract year from 'year_month'
df_service1['year'] = df_service1['year_month'].astype(str).str[:4].astype(int)
df_service1
```

Out[14]:

| | year_month | country | county | uid | county_code | estimated_number_of_pregnant |
|------|------------|---------|----------------------|-------------|--------------|------------------------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | KE_County_23 | 2 |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | KE_County_29 | 4 |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | KE_County_24 | 2 |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | KE_County_36 | 3 |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | KE_County_47 | 15 |
| ... | ... | ... | ... | ... | ... | ... |
| 6199 | 202409 | Kenya | Isiolo County | bzOfj0iwfDH | KE_County_11 | 1 |
| 6200 | 202409 | Kenya | Trans Nzoia County | mThvosEfAU | KE_County_26 | 3 |
| 6201 | 202409 | Kenya | Nakuru County | ob6SxuRcqU4 | KE_County_32 | 7 |
| 6202 | 202409 | Kenya | Tharaka Nithi County | T4urHM47nlm | KE_County_13 | 1 |
| 6203 | 202409 | Kenya | Nyeri County | ptWVfaCldVx | KE_County_19 | 1 |

6204 rows × 53 columns



```
In [15]: # Create a new column (uid_code)
df_service1['uid_code'] = df_service1[
    ['year_month','uid']].astype(str).agg('_'.join, axis=1)

df_service1.head()
```

Out[15]:

| | year_month | country | county | uid | county_code | estimated_number_of_pregnant_women |
|---|------------|---------|-------------------|-------------|--------------|------------------------------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | KE_County_23 | 245 |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | KE_County_29 | 437 |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | KE_County_24 | 211 |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | KE_County_36 | 397 |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | KE_County_47 | 1588 |

5 rows × 54 columns

```
In [16]: # Create a new column (uid_year)
df_service1['uid_year'] = df_service1[
    ['year','uid']].astype(str).agg('_'.join, axis=1)

df_service1.head()
```

Out[16]:

| | year_month | country | county | uid | county_code | estimated_number_of_pregnant_women |
|---|------------|---------|-------------------|-------------|--------------|------------------------------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | KE_County_23 | 245 |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | KE_County_29 | 437 |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | KE_County_24 | 211 |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | KE_County_36 | 397 |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | KE_County_47 | 1588 |

5 rows × 55 columns

In [17]: df_service1.columns

```
Out[17]: Index(['year_month', 'country', 'county', 'uid', 'county_code',
       'estimated_number_of_pregnant_women', 'fp_attendance_new_clients',
       'fp_attendance_re_visits',
       'adolescent_10_14_yrs_receiving_fp_services_new_clients',
       'adolescent_10_14_yrs_receiving_fp_services_re_visits',
       'adolescent_15_19_yrs_receiving_fp_services_new_clients',
       'adolescent_15_19_yrs_receiving_fp_services_re_visits',
       'adolescent_20_24_yrs_receiving_fp_services_new_clients',
       'adolescent_20_24_yrs_receiving_fp_services_re_visits',
       'client_receiving_male_condoms_new_clients',
       'client_receiving_male_condoms_re_visits',
       'clients_counselled_natural_family_planning_new_clients',
       'clients_counselled_natural_family_planning_re_visits',
       'clients_receiving_female_condoms_new_clients',
       'clients_receiving_female_condoms_re_visits',
       'emergency_contraceptive_pill_new_clients',
       'emergency_contraceptive_pill_re_visits',
       'pills_combined_oral_contraceptive_new_clients',
       'pills_combined_oral_contraceptive_re_visits',
       'pills_progestin_only_new_clients', 'pills_progestin_only_re_visits',
       '2020_adults_25+_receiving_fp_services_new_clients',
       '2020_adults_25+_receiving_fp_services_re_visits',
       '2020_clients_given_cycle_beads_new_clients',
       '2020_clients_given_cycle_beads_re_visits',
       '2020_clients_receiving_post_abortion_fp_new_clients',
       '2020_clients_receiving_post_abortion_fp_re_visits',
       '2020_fp_injections_dmpa_im_new_clients',
       '2020_fp_injections_dmpa_im_re_visits',
       '2020_fp_injections_dmpa_sc_new_clients',
       '2020_fp_injections_dmpa_sc_re_visits',
       '2020_iucd_insertion_hormonal_ist_time_insertion',
       '2020_iucd_insertion_hormonal_re_insertion',
       '2020_iucd_insertion_non_hormonal_ist_time_insertion',
       '2020_iucd_insertion_non_hormonal_re_insertion',
       '2020_implants_insertion_1_rod_ist_time_insertion',
       '2020_implants_insertion_1_rod_re_insertion',
       '2020_implants_insertion_2_rod_ist_time_insertion',
       '2020_implants_insertion_2_rod_re_insertion',
       '2020_post_parturm_fp_4weeks_to_6weeks_new_clients',
       '2020_post_parturm_fp_4weeks_to_6weeks_re_visits',
       '2020_post_parturm_fp_within_48_hours_new_clients',
       '2020_post_parturm_fp_within_48_hours_re_visits',
       '2020_voluntary_surgical_contraception_vasectomy_ist_time_insertion',
       '2020_voluntary_surgical_contraception_vasectomy_re_insertion',
       '2020_voluntary_surgical_contraception_btl_ist_time_insertion',
       '2020_voluntary_surgical_contraception_btl_re_insertion', 'year',
       'uid_code', 'uid_year'],
      dtype='object')
```

```
In [18]: # Check columns with float dtypes
float_cols = df_service1.select_dtypes(include=['float', 'float64']).columns
float_cols
```

```
Out[18]: Index(['estimated_number_of_pregnant_women', 'fp_attendance_new_clients',
   'fp_attendance_re_visits',
   'adolescent_10_14_yrs_receiving_fp_services_new_clients',
   'adolescent_10_14_yrs_receiving_fp_services_re_visits',
   'adolescent_15_19_yrs_receiving_fp_services_new_clients',
   'adolescent_15_19_yrs_receiving_fp_services_re_visits',
   'adolescent_20_24_yrs_receiving_fp_services_new_clients',
   'adolescent_20_24_yrs_receiving_fp_services_re_visits',
   'client_receiving_male_condoms_re_visits',
   'clients_counselled_natural_family_planning_new_clients',
   'clients_counselled_natural_family_planning_re_visits',
   'clients_receiving_female_condoms_new_clients',
   'clients_receiving_female_condoms_re_visits',
   'emergency_contraceptive_pill_new_clients',
   'emergency_contraceptive_pill_re_visits',
   'pills_combined_oral_contraceptive_new_clients',
   'pills_combined_oral_contraceptive_re_visits',
   'pills_progestin_only_new_clients', 'pills_progestin_only_re_visits',
   '2020_adults_25+_receiving_fp_services_new_clients',
   '2020_adults_25+_receiving_fp_services_re_visits',
   '2020_clients_given_cycle_beads_new_clients',
   '2020_clients_given_cycle_beads_re_visits',
   '2020_clients_receiving_post_abortion_fp_new_clients',
   '2020_clients_receiving_post_abortion_fp_re_visits',
   '2020_fp_injections_dmpa_im_new_clients',
   '2020_fp_injections_dmpa_im_re_visits',
   '2020_fp_injections_dmpa_sc_new_clients',
   '2020_fp_injections_dmpa_sc_re_visits',
   '2020_iucd_insertion_hormonal_ist_time_insertion',
   '2020_iucd_insertion_hormonal_re_insertion',
   '2020_iucd_insertion_non_hormonal_ist_time_insertion',
   '2020_iucd_insertion_non_hormonal_re_insertion',
   '2020_implants_insertion_1_rod_ist_time_insertion',
   '2020_implants_insertion_1_rod_re_insertion',
   '2020_implants_insertion_2_rod_ist_time_insertion',
   '2020_implants_insertion_2_rod_re_insertion',
   '2020_post_parturm_fp_4weeks_to_6weeks_new_clients',
   '2020_post_parturm_fp_4weeks_to_6weeks_re_visits',
   '2020_post_parturm_fp_within_48_hours_new_clients',
   '2020_post_parturm_fp_within_48_hours_re_visits',
   '2020_voluntary_surgical_contraception_vasectomy_ist_time_insertion',
   '2020_voluntary_surgical_contraception_vasectomy_re_insertion',
   '2020_voluntary_surgical_contraception_btl_ist_time_insertion',
   '2020_voluntary_surgical_contraception_btl_re_insertion'],
  dtype='object')
```

```
In [19]: # Convert float columns to int64
df_service1[float_cols] = df_service1[float_cols].astype('int64')
```

In [20]: df_service1.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6204 entries, 0 to 6203
Data columns (total 55 columns):
 #   Column          Dtype   Nulls on Non-Nu
 11  Count           Dtype   on-null 6204 n
 12  Dtype           Dtype   on-null 6204 n
 13  -----           -----   ----- 6204 n
 14  year_month     int64   on-null 6204 n
 15  country         object  on-null 6204 n
 16  county          object  on-null 6204 n
 17  county_code     object  on-null 6204 n
 18  uid              object  on-null 6204 n
 19  estimated_number_of_pregnant_women int64   on-null 6204 n
 20  fp_attendance_new_clients      int64   on-null 6204 n
 21  fp_attendance_re_visits       int64   on-null 6204 n
 22  adolescent_10_14_yrs_receiving_fp_services_new_clients int64   on-null 6204 n
 23  adolescent_10_14_yrs_receiving_fp_services_re_visits    int64   on-null 6204 n
 24  adolescent_15_19_yrs_receiving_fp_services_new_clients      int64   on-null 6204 n
 25  adolescent_15_19_yrs_receiving_fp_services_re_visits       int64   on-null 6204 n
 26  adolescent_20_24_yrs_receiving_fp_services_new_clients      int64   on-null 6204 n
 27  adolescent_20_24_yrs_receiving_fp_services_re_visits        int64   on-null 6204 n
 28  client_receiving_male_condoms_new_clients      int64   on-null 6204 n
 29  client_receiving_male_condoms_re_visits       int64   on-null 6204 n
 30  clients_counselled_natural_family_planning_new_clients int64   on-null 6204 n
 31  clients_counselled_natural_family_planning_re_visits    int64   on-null 6204 n
 32  clients_receiving_female_condoms_new_clients      int64   on-null 6204 n
 33  clients_receiving_female_condoms_re_visits       int64   on-null 6204 n
 34  emergency_contraceptive_pill_new_clients      int64   on-null 6204 n
 35  emergency_contraceptive_pill_re_visits       int64   on-null 6204 n
 36  pills_combined_oral_contraceptive_new_clients int64   on-null 6204 n
 37  pills_combined_oral_contraceptive_re_visits    int64   on-null 6204 n
 38  pills_progestin_only_new_clients      int64   on-null 6204 n
```

```
25  pills_progestin_only_re_visits          6204 n
on-null  int64
26  2020_adults_25+_receiving_fp_services_new_clients 6204 n
on-null  int64
27  2020_adults_25+_receiving_fp_services_re_visits 6204 n
on-null  int64
28  2020_clients_given_cycle_beads_new_clients 6204 n
on-null  int64
29  2020_clients_given_cycle_beads_re_visits 6204 n
on-null  int64
30  2020_clients_receiving_post_abortion_fp_new_clients 6204 n
on-null  int64
31  2020_clients_receiving_post_abortion_fp_re_visits 6204 n
on-null  int64
32  2020_fp_injections_dmpa_im_new_clients 6204 n
on-null  int64
33  2020_fp_injections_dmpa_im_re_visits 6204 n
on-null  int64
34  2020_fp_injections_dmpa_sc_new_clients 6204 n
on-null  int64
35  2020_fp_injections_dmpa_sc_re_visits 6204 n
on-null  int64
36  2020_iucd_insertion_hormonal_ist_time_insertion 6204 n
on-null  int64
37  2020_iucd_insertion_hormonal_re_insertion 6204 n
on-null  int64
38  2020_iucd_insertion_non_hormonal_ist_time_insertion 6204 n
on-null  int64
39  2020_iucd_insertion_non_hormonal_re_insertion 6204 n
on-null  int64
40  2020_implants_insertion_1_rod_ist_time_insertion 6204 n
on-null  int64
41  2020_implants_insertion_1_rod_re_insertion 6204 n
on-null  int64
42  2020_implants_insertion_2_rod_ist_time_insertion 6204 n
on-null  int64
43  2020_implants_insertion_2_rod_re_insertion 6204 n
on-null  int64
44  2020_post_parturm_fp_4weeks_to_6weeks_new_clients 6204 n
on-null  int64
45  2020_post_parturm_fp_4weeks_to_6weeks_re_visits 6204 n
on-null  int64
46  2020_post_parturm_fp_within_48_hours_new_clients 6204 n
on-null  int64
47  2020_post_parturm_fp_within_48_hours_re_visits 6204 n
on-null  int64
48  2020_voluntary_surgical_contraception_vasectomy_ist_time_insertion 6204 n
on-null  int64
49  2020_voluntary_surgical_contraception_vasectomy_re_insertion 6204 n
on-null  int64
50  2020_voluntary_surgical_contraception_btl_ist_time_insertion 6204 n
on-null  int64
51  2020_voluntary_surgical_contraception_btl_re_insertion 6204 n
on-null  int64
52  year 6204 n
on-null  int32
53  uid_code 6204 n
```

```
on-null    object          54 uid_year
on-null    object
dtypes: int32(1), int64(48), object(6)
memory usage: 2.6+ MB
```

6204 n

2. ke_fp_commodity_data.csv

```
In [21]: # Make a copy of the data
df_commodity1 = df_commodity.copy()
```

```
In [22]: # Preview the data
df_commodity1.head()
```

Out[22]:

| | periodid | periodname | periodcode | perioddescription | orgunitlevel1 | orgunitlevel2 | organisationunit |
|---|----------|------------|------------|-------------------|---------------|------------------|------------------|
| 0 | 201404 | 14-Apr | 201404 | | Kenya | Kirinyaga County | Ulj33KBau: |
| 1 | 201404 | 14-Apr | 201404 | | Kenya | Kisii County | sPkRcDvhGV |
| 2 | 201404 | 14-Apr | 201404 | | Kenya | Kisumu County | tAbBVBbuec |
| 3 | 201404 | 14-Apr | 201404 | | Kenya | Makueni County | BoDytkJQ4 |
| 4 | 201404 | 14-Apr | 201404 | | Kenya | Kwale County | N7YETT3A9 |

5 rows × 50 columns



In [23]: # Explore the data
df_commodity1.info()

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2480 entries, 0 to 2479
Data columns (total 50 columns):
 #   Column           Non-Null Count Dtype
 ---  -- 
 0   periodid        2480 non-null  int64
 1   periodname      2480 non-null  object
 2   periodcode       2480 non-null  int64
 3   perioddescription 0 non-null    float64
 4   orgunitlevel1    2480 non-null  object
 5   orgunitlevel2    2480 non-null  object
 6   organisationunitid 2480 non-null  object
 7   organisationunitname 2480 non-null  object
 8   organisationunitcode 2480 non-null  object
 9   organisationunitdescription 0 non-null    float64
 10  pills_combined_oral_contraceptive_stock_losses 2480 non-null  int64
 11  pills_combined_oral_contraceptive_stock_dispensed 2480 non-null  int64
 12  pills_combined_oral_contraceptive_stock_at_hand 2480 non-null  int64
 13  pills_combined_oral_contraceptive_stock_requested 2480 non-null  int64
 14  pills_combined_oral_contraceptive_stock_received 2480 non-null  int64
 15  pills_emergency_pill_stock_losses 2480 non-null  int64
 16  pills_emergency_pill_stock_dispensed 2480 non-null  int64
 17  pills_emergency_pill_stock_at_hand 2480 non-null  int64
 18  pills_emergency_pill_stock_requested 2480 non-null  int64
 19  pills_emergency_pill_stock_received 2480 non-null  int64
 20  pills_progestin_only_pills_stock_losses 2480 non-null  int64
 21  pills_progestin_only_pills_stock_dispensed 2480 non-null  int64
 22  pills_progestin_only_pills_stock_at_hand 2480 non-null  int64
 23  pills_progestin_only_pills_stock_requested 2480 non-null  int64
 24  pills_progestin_only_pills_stock_received 2480 non-null  int64
 25  condoms_female_condom_stock_losses 2480 non-null  int64
 26  condoms_female_condom_stock_dispensed 2480 non-null  int64
 27  condoms_female_condom_stock_at_hand 2480 non-null  int64
 28  condoms_female_condom_stock_requested 2480 non-null  int64
 29  condoms_female_condom_stock_received 2480 non-null  int64
 30  condoms_male_condom_stock_losses 2480 non-null  int64
 31  condoms_male_condom_stock_dispensed 2480 non-null  int64
 32  condoms_male_condom_stock_at_hand 2480 non-null  int64
 33  condoms_male_condom_stock_requested 2480 non-null  int64
 34  condoms_male_condom_stock_received 2480 non-null  int64
 35  injectables_stock_losses 2480 non-null  int64
 36  injectables_stock_dispensed 2480 non-null  int64
 37  injectables_stock_at_hand 2480 non-null  int64
 38  injectables_stock_requested 2480 non-null  int64
 39  injectables_stock_received 2480 non-null  int64
 40  implants_stock_losses 2480 non-null  int64
 41  implants_stock_dispensed 2480 non-null  int64
 42  implants_stock_at_hand 2480 non-null  int64
 43  implants_stock_requested 2480 non-null  int64
 44  implants_stock_received 2480 non-null  int64
 45  iud_stock_losses 2480 non-null  int64
 46  iud_stock_dispensed 2480 non-null  int64
 47  iud_stock_at_hand 2480 non-null  int64
 48  iud_stock_requested 2480 non-null  int64
 49  iud_stock_received 2480 non-null  int64

```

```
dtypes: float64(2), int64(42), object(6)
memory usage: 968.9+ KB
```

```
In [24]: # Standardize the column names
df_commodity1 = standardize_col_labels(df_commodity1)

df_commodity1.head()
```

```
Out[24]:   periodid periodname periodcode perioddescription orgunitlevel1 orgunitlevel2 organisationunit
0    201404     14-Apr  201404                 NaN      Kenya  Kirinyaga County  Ulj33KBau:
1    201404     14-Apr  201404                 NaN      Kenya  Kisii County   sPkRcDvhGV
2    201404     14-Apr  201404                 NaN      Kenya  Kisumu County  tAbBVBbuec
3    201404     14-Apr  201404                 NaN      Kenya  Makueni County  BoDytkJQ4
4    201404     14-Apr  201404                 NaN      Kenya  Kwale County   N7YETT3A9
```

5 rows × 50 columns



In [25]: # Rename column names

```
df_commodity1 = df_commodity1.rename(columns=name_map)
df_commodity1
```

Out[25]:

| | periodid | periodname | year_month | perioddescription | country | county | uid | orga |
|------|----------|------------|------------|-------------------|---------|--------|------------------|-------------|
| 0 | 201404 | 14-Apr | 201404 | | NaN | Kenya | Kirinyaga County | Ulj33KBau7V |
| 1 | 201404 | 14-Apr | 201404 | | NaN | Kenya | Kisii County | sPkRcDvhGWA |
| 2 | 201404 | 14-Apr | 201404 | | NaN | Kenya | Kisumu County | tAbBVBbueqD |
| 3 | 201404 | 14-Apr | 201404 | | NaN | Kenya | Makueni County | BoDytkJQ4Qi |
| 4 | 201404 | 14-Apr | 201404 | | NaN | Kenya | Kwale County | N7YETT3A9r1 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2475 | 202209 | 22-Sep | 202209 | | NaN | Kenya | Kericho County | ihZsj8alvtb |
| 2476 | 202209 | 22-Sep | 202209 | | NaN | Kenya | Nandi County | t0J75eHKxz5 |
| 2477 | 202209 | 22-Sep | 202209 | | NaN | Kenya | Machakos County | yhCUgGcCcOo |
| 2478 | 202309 | 23-Sep | 202309 | | NaN | Kenya | Nairobi County | jkG3zaihdSs |
| 2479 | 202409 | 24-Sep | 202409 | | NaN | Kenya | Nairobi County | jkG3zaihdSs |

2480 rows × 50 columns



```
In [26]: # Drop columns where all values are null
df_commodity1=df_commodity1.dropna(axis=1, how='all')
df_commodity1
```

Out[26]:

| | periodid | periodname | year_month | country | county | uid | organisationunitname |
|------|----------|------------|------------|---------|------------------|-------------|----------------------|
| 0 | 201404 | 14-Apr | 201404 | Kenya | Kirinyaga County | Ulj33KBau7V | Kirinyaga County |
| 1 | 201404 | 14-Apr | 201404 | Kenya | Kisii County | sPkRcDvhGWA | Kisii County |
| 2 | 201404 | 14-Apr | 201404 | Kenya | Kisumu County | tAbBVBBueqD | Kisumu County |
| 3 | 201404 | 14-Apr | 201404 | Kenya | Makueni County | BoDytkJQ4Qi | Makueni County |
| 4 | 201404 | 14-Apr | 201404 | Kenya | Kwale County | N7YETT3A9r1 | Kwale County |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 2475 | 202209 | 22-Sep | 202209 | Kenya | Kericho County | ihZsj8alvtb | Kericho County |
| 2476 | 202209 | 22-Sep | 202209 | Kenya | Nandi County | t0J75eHKxz5 | Nandi County |
| 2477 | 202209 | 22-Sep | 202209 | Kenya | Machakos County | yhCUGGcCcOo | Machakos County |
| 2478 | 202309 | 23-Sep | 202309 | Kenya | Nairobi County | jkG3zaihdSs | Nairobi County |
| 2479 | 202409 | 24-Sep | 202409 | Kenya | Nairobi County | jkG3zaihdSs | Nairobi County |

2480 rows × 48 columns



In [27]: # Drop unwanted columns

```
df_commodity1.drop(columns='organisationunitname', inplace=True)
df_commodity1
```

c:\Users\User\anaconda3\envs\learn-env\lib\site-packages\pandas\core\frame.py:4
163: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
return super().drop()
```

Out[27]:

| | periodid | periodname | year_month | country | county | uid | county_code | pills_co |
|------|----------|------------|------------|---------|------------------|-------------|--------------|----------|
| 0 | 201404 | 14-Apr | 201404 | Kenya | Kirinyaga County | Ulj33KBau7V | KE_County_20 | |
| 1 | 201404 | 14-Apr | 201404 | Kenya | Kisii County | sPkRcDvhGWA | KE_County_45 | |
| 2 | 201404 | 14-Apr | 201404 | Kenya | Kisumu County | tAbBVBbueqD | KE_County_42 | |
| 3 | 201404 | 14-Apr | 201404 | Kenya | Makueni County | BoDytkJQ4Qi | KE_County_17 | |
| 4 | 201404 | 14-Apr | 201404 | Kenya | Kwale County | N7YETT3A9r1 | KE_County_2 | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2475 | 202209 | 22-Sep | 202209 | Kenya | Kericho County | ihZsJ8alvtb | KE_County_35 | |
| 2476 | 202209 | 22-Sep | 202209 | Kenya | Nandi County | t0J75eHKxz5 | KE_County_29 | |
| 2477 | 202209 | 22-Sep | 202209 | Kenya | Machakos County | yhCUGGcCcOo | KE_County_16 | |
| 2478 | 202309 | 23-Sep | 202309 | Kenya | Nairobi County | jkG3zaihdSs | KE_County_47 | |
| 2479 | 202409 | 24-Sep | 202409 | Kenya | Nairobi County | jkG3zaihdSs | KE_County_47 | |

2480 rows × 47 columns



```
In [28]: # Check for missing values
df_commodity1.isna().sum().sort_values(ascending=False)
```

```
Out[28]: iud_stock_received          0
pills_combined_oral_contraceptive_stock_received 0
pills_progestin_only_pills_stock_requested    0
pills_progestin_only_pills_stock_at_hand      0
pills_progestin_only_pills_stock_dispensed    0
pills_progestin_only_pills_stock_losses       0
pills_emergency_pill_stock_received          0
pills_emergency_pill_stock_requested        0
pills_emergency_pill_stock_at_hand          0
pills_emergency_pill_stock_dispensed        0
pills_emergency_pill_stock_losses           0
pills_combined_oral_contraceptive_stock_requested 0
condoms_female_condom_stock_losses          0
pills_combined_oral_contraceptive_stock_at_hand 0
pills_combined_oral_contraceptive_stock_dispensed 0
pills_combined_oral_contraceptive_stock_losses 0
county_code                                0
uid                                       0
county                                    0
country                                   0
year_month                                 0
periodname                                0
pills_progestin_only_pills_stock_received 0
condoms_female_condom_stock_dispensed     0
iud_stock_requested                       0
injectables_stock_requested              0
iud_stock_at_hand                        0
iud_stock_dispensed                      0
iud_stock_losses                         0
implants_stock_received                 0
implants_stock_requested                0
implants_stock_at_hand                 0
implants_stock_dispensed                0
implants_stock_losses                  0
injectables_stock_received              0
injectables_stock_at_hand               0
condoms_female_condom_stock_at_hand     0
injectables_stock_dispensed             0
injectables_stock_losses                0
condoms_male_condom_stock_received      0
condoms_male_condom_stock_requested     0
condoms_male_condom_stock_at_hand       0
condoms_male_condom_stock_dispensed     0
condoms_male_condom_stock_losses        0
condoms_female_condom_stock_received    0
condoms_female_condom_stock_requested   0
periodid                                  0
dtype: int64
```

The missing values were interpreted as 'no event reported'

In [29]: # Dealing with the missing values

```
df_commodity1 = df_commodity1.fillna(0)
```

In [30]: # Check for duplicates

```
df_commodity1.duplicated().sum()
```

Out[30]: 0

A new column(uid_code) was created by concatenating the year_month column and organisation unit id

In [31]: # Create a new column (uid_code)

```
df_commodity1['uid_code'] = df_commodity1[['year_month', 'uid']].astype(str).agg(lambda x: x[0] + x[1], axis=1)
```

```
df_commodity1.head()
```

Out[31]:

| | periodid | periodname | year_month | country | county | uid | county_code | pills_combi |
|---|----------|------------|------------|---------|------------------|-------------|--------------|-------------|
| 0 | 201404 | 14-Apr | 201404 | Kenya | Kirinyaga County | UIj33KBau7V | KE_County_20 | |
| 1 | 201404 | 14-Apr | 201404 | Kenya | Kisii County | sPkRcDvhGWA | KE_County_45 | |
| 2 | 201404 | 14-Apr | 201404 | Kenya | Kisumu County | tAbBVBbueqD | KE_County_42 | |
| 3 | 201404 | 14-Apr | 201404 | Kenya | Makueni County | BoDytkJQ4Qi | KE_County_17 | |
| 4 | 201404 | 14-Apr | 201404 | Kenya | Kwale County | N7YETT3A9r1 | KE_County_2 | |

5 rows × 48 columns



```
In [32]: # Convert float columns to int64
float_cols_com = df_commodity1.select_dtypes(include=['float', 'float64']).columns
df_commodity1[float_cols_com] = df_commodity1[float_cols_com].astype('int64')
df_commodity1.info()
```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2480 entries, 0 to 2479
Data columns (total 48 columns):
 #   Column           Non-Null Count Dtype
 ---  -- 
 0   periodid        2480 non-null  int64
 1   periodname      2480 non-null  object
 2   year_month     2480 non-null  int64
 3   country         2480 non-null  object
 4   county          2480 non-null  object
 5   uid              2480 non-null  object
 6   county_code     2480 non-null  object
 7   pills_combined_oral_contraceptive_stock_losses 2480 non-null  int64
 8   pills_combined_oral_contraceptive_stock_dispensed 2480 non-null  int64
 9   pills_combined_oral_contraceptive_stock_at_hand 2480 non-null  int64
 10  pills_combined_oral_contraceptive_stock_requested 2480 non-null  int64
 11  pills_combined_oral_contraceptive_stock_received 2480 non-null  int64
 12  pills_emergency_pill_stock_losses    2480 non-null  int64
 13  pills_emergency_pill_stock_dispensed 2480 non-null  int64
 14  pills_emergency_pill_stock_at_hand 2480 non-null  int64
 15  pills_emergency_pill_stock_requested 2480 non-null  int64
 16  pills_emergency_pill_stock_received 2480 non-null  int64
 17  pills_progestin_only_pills_stock_losses 2480 non-null  int64
 18  pills_progestin_only_pills_stock_dispensed 2480 non-null  int64
 19  pills_progestin_only_pills_stock_at_hand 2480 non-null  int64
 20  pills_progestin_only_pills_stock_requested 2480 non-null  int64
 21  pills_progestin_only_pills_stock_received 2480 non-null  int64
 22  condoms_female_condom_stock_losses 2480 non-null  int64
 23  condoms_female_condom_stock_dispensed 2480 non-null  int64
 24  condoms_female_condom_stock_at_hand 2480 non-null  int64
 25  condoms_female_condom_stock_requested 2480 non-null  int64
 26  condoms_female_condom_stock_received 2480 non-null  int64
 27  condoms_male_condom_stock_losses 2480 non-null  int64
 28  condoms_male_condom_stock_dispensed 2480 non-null  int64
 29  condoms_male_condom_stock_at_hand 2480 non-null  int64
 30  condoms_male_condom_stock_requested 2480 non-null  int64
 31  condoms_male_condom_stock_received 2480 non-null  int64
 32  injectables_stock_losses 2480 non-null  int64
 33  injectables_stock_dispensed 2480 non-null  int64
 34  injectables_stock_at_hand 2480 non-null  int64
 35  injectables_stock_requested 2480 non-null  int64
 36  injectables_stock_received 2480 non-null  int64
 37  implants_stock_losses 2480 non-null  int64
 38  implants_stock_dispensed 2480 non-null  int64
 39  implants_stock_at_hand 2480 non-null  int64
 40  implants_stock_requested 2480 non-null  int64
 41  implants_stock_received 2480 non-null  int64
 42  iud_stock_losses 2480 non-null  int64
 43  iud_stock_dispensed 2480 non-null  int64
 44  iud_stock_at_hand 2480 non-null  int64
 45  iud_stock_requested 2480 non-null  int64
 46  iud_stock_received 2480 non-null  int64
 47  uid_code         2480 non-null  object
dtypes: int64(42), object(6)
memory usage: 930.1+ KB

```

3. ke_fp_population_data

```
In [33]: # Make a copy of the data
df_population1 = df_population.copy()
```

```
In [34]: # Standardize column names
df_population1 = standardize_col_labels(df_population1)

# Preview the data
df_population1.head()
```

Out[34]:

| | periodid | periodname | periodcode | perioddescription | orgunitlevel1 | orgunitlevel2 | organisationunit |
|---|----------|------------|------------|-------------------|---------------|------------------------|------------------|
| 0 | 2014 | 2014 | 2014 | | Kenya | Kajiado County | Hsk1YV8kH |
| 1 | 2014 | 2014 | 2014 | | Kenya | Laikipia County | xuFdFy6t9/ |
| 2 | 2014 | 2014 | 2014 | | Kenya | Turkana County | kphDeKCIF |
| 3 | 2014 | 2014 | 2014 | | Kenya | Nandi County | t0J75eHKx |
| 4 | 2014 | 2014 | 2014 | | Kenya | Elgeyo Marakwet County | MqnLxQBigC |



```
In [35]: # Rename column names
df_population1 = df_population1.rename(columns=name_map)

df_population1.head()
```

Out[35]:

| | periodid | periodname | year_month | perioddescription | country | county | uid | organisa |
|---|----------|------------|------------|-------------------|---------|------------------------|-------------|----------|
| 0 | 2014 | 2014 | 2014 | | Kenya | Kajiado County | Hsk1YV8kHkT | K |
| 1 | 2014 | 2014 | 2014 | | Kenya | Laikipia County | xuFdFy6t9AH | L |
| 2 | 2014 | 2014 | 2014 | | Kenya | Turkana County | kphDeKCIFch | Tu |
| 3 | 2014 | 2014 | 2014 | | Kenya | Nandi County | t0J75eHKxz5 | N |
| 4 | 2014 | 2014 | 2014 | | Kenya | Elgeyo Marakwet County | MqnLxQBigG0 | Elg |



In [36]: `# Drop unwanted columns
df_population1.drop(columns=['organisationunitname'], axis=1) # This was dropped`

Out[36]:

| | periodid | periodname | year_month | perioddescription | country | county | uid | cour |
|-----|----------|------------|------------|-------------------|---------|--------|------------------------|------------------|
| 0 | 2014 | 2014 | 2014 | 2014 | NaN | Kenya | Kajiado County | Hsk1YV8kHkT KE_C |
| 1 | 2014 | 2014 | 2014 | 2014 | NaN | Kenya | Laikipia County | xuFdFy6t9AH KE_C |
| 2 | 2014 | 2014 | 2014 | 2014 | NaN | Kenya | Turkana County | kphDeKlFch KE_C |
| 3 | 2014 | 2014 | 2014 | 2014 | NaN | Kenya | Nandi County | t0J75eHKxz5 KE_C |
| 4 | 2014 | 2014 | 2014 | 2014 | NaN | Kenya | Elgeyo Marakwet County | MqnLxQBigG0 KE_C |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 512 | 2024 | 2024 | 2024 | 2024 | NaN | Kenya | Makueni County | BoDytkJQ4Qi KE_C |
| 513 | 2024 | 2024 | 2024 | 2024 | NaN | Kenya | Kajiado County | Hsk1YV8kHkT KE_C |
| 514 | 2024 | 2024 | 2024 | 2024 | NaN | Kenya | Kakamega County | BjC1xL40gHo KE_C |
| 515 | 2024 | 2024 | 2024 | 2024 | NaN | Kenya | Muranga County | ahwTMNAJvrL KE_C |
| 516 | 2024 | 2024 | 2024 | 2024 | NaN | Kenya | Nairobi County | jkG3zaihdSs KE_C |

517 rows × 18 columns



Calculate Women eligible for FP

```
In [37]: # Insert the new column, treating NaN values as 0 during the calculation
df_population1['eligible_fp'] = (
    df_population1['women_of_childbearing_age_(15_49yrs)']
    .add(df_population1['population_10_14_year_old_girls'], fill_value=0)
    .sub(df_population1['estimated_number_of_pregnant_women'], fill_value=0)
)

# Display the updated DataFrame
df_population1.head()
```

Out[37]:

| | periodid | periodname | year_month | perioddescription | country | county | uid | organisat | |
|---|----------|------------|------------|-------------------|---------|--------|------------------------|-------------|-----|
| 0 | 2014 | 2014 | 2014 | | NaN | Kenya | Kajiado County | Hsk1YV8kHkT | K |
| 1 | 2014 | 2014 | 2014 | | NaN | Kenya | Laikipia County | xuFdFy6t9AH | L |
| 2 | 2014 | 2014 | 2014 | | NaN | Kenya | Turkana County | kphDeKClFch | T |
| 3 | 2014 | 2014 | 2014 | | NaN | Kenya | Nandi County | t0J75eHKxz5 | |
| 4 | 2014 | 2014 | 2014 | | NaN | Kenya | Elgeyo Marakwet County | MqnLxQBigG0 | Elg |

4. ke_fp_benchmarks_data.csv

```
In [38]: # Make copies of the benchmarks dataframes
df_core_health_workforce1 = df_core_health_workforce.copy()
df_demand_satisfied1 = df_demand_satisfied.copy()
df_mcpr1 = df_mcpr.copy()
df_teenage_pregnancy1 = df_teenage_pregnancy.copy()
df_unmet_need1 = df_unmet_need.copy()
```

In [39]: # Preview the data

```
df_core_health_workforce1.head()
df_demand_satisfied1.head()
df_mcpr1.head()
df_teenage_pregnancy1.head()
df_unmet_need1.head()
```

Out[39]:

| | organisationunitname | uid | county_cou | benchmark_indicator | Total Unmet Need (Married Women, %) | year | source |
|---|----------------------|-------------|--------------|-------------------------------------|-------------------------------------|------|---------|
| 0 | baringo | vvOK1BxTbet | KE_County_30 | Total Unmet Need (Married Women, %) | 16.6 | 2022 | KDF 20: |
| 1 | bomet | HMNARUV2CW4 | KE_County_36 | Total Unmet Need (Married Women, %) | 16.7 | 2022 | KDF 20: |
| 2 | bungoma | KGHhQ5GLd4k | KE_County_39 | Total Unmet Need (Married Women, %) | 14.6 | 2022 | KDF 20: |
| 3 | busia | Tvf1zgVZ0K4 | KE_County_40 | Total Unmet Need (Married Women, %) | 18.6 | 2022 | KDF 20: |
| 4 | elgeyo_marakwet | MqnLxQBigG0 | KE_County_28 | Total Unmet Need (Married Women, %) | 13.5 | 2022 | KDF 20: |

b) Initial Feature Engineering

CYP computation and grouping

Couple Years of Protection(CYP)-CYP measures the estimated protection provided by FP based on the volume of contraceptive method distribution to clients to help monitor health system performance and track trends and progress over time.

```
In [40]: # Define CYP conversion factors
cyp_factors = {
    'condoms': 0.0083,
    'emergency_pill': 0.05,
    'pills_combined_oral_contraceptives': 0.0067,
    'pills_progestin_only_contraceptives': 0.0833,
    'injections': 0.25,
    'implants_1_rod': 2.5,
    'implants_2_rod': 3.8,
    'iucd_hormonal': 4.8,
    'iucd_non_hormonal': 4.6,
    'surgical': 10.0
}

# Initialize total_cyp column
df_service1['total_cyp'] = 0

# Compute total_cyp based on matching column names
for col in df_service1.columns:
    if 'condom' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['condoms']

    elif 'emergency' in col and 'pill' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['emergency_pill']

    elif 'combined_oral' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['pills_combined']

    elif 'progestin_only' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['pills_progestin']

    elif 'injection' in col or 'dmpa' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['injections']

    elif '1_rod' in col and 'implant' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['implants_1_rod']

    elif '2_rod' in col and 'implant' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['implants_2_rod']

    elif 'iucd' in col and 'hormonal' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['iucd_hormonal']

    elif 'iucd' in col and 'non_hormonal' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['iucd_non_hormonal']

    elif 'surgical' in col or 'vasectomy' in col or 'btl' in col:
        df_service1['total_cyp'] += df_service1[col] * cyp_factors['surgical']
```

In [41]: `df_service1.head()`

Out[41]:

| | year_month | country | county | uid | county_code | estimated_number_of_pregnant_women |
|---|------------|---------|-------------------|-------------|--------------|------------------------------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | KE_County_23 | 2 |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | KE_County_29 | 4 |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | KE_County_24 | 2 |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | KE_County_36 | 3 |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | KE_County_47 | 15 |

5 rows × 56 columns



FP Method Grouping (New vs Revisits)

```
In [42]: # Group FP Methods
df_service1['adolescent_10_24_receiving_fp_new'] = (
    df_service1['adolescent_10_14_yrs_receiving_fp_services_new_clients'] +
    df_service1['adolescent_15_19_yrs_receiving_fp_services_new_clients'] +
    df_service1['adolescent_20_24_yrs_receiving_fp_services_new_clients']
)

df_service1['adolescent_10_24_receiving_fp_revisits'] = (
    df_service1['adolescent_10_14_yrs_receiving_fp_services_re_visits'] +
    df_service1['adolescent_15_19_yrs_receiving_fp_services_re_visits'] +
    df_service1['adolescent_20_24_yrs_receiving_fp_services_re_visits']
)

df_service1['adults_25+_receiving_fp_services_new'] = df_service1['2020_adults_25+_receiving_fp_services_new']
df_service1['adults_25+_receiving_fp_services_revisits'] = df_service1['2020_adults_25+_receiving_fp_services_revisits']

df_service1['condoms_new'] = df_service1['clients_receiving_female_condoms_new_clients']
df_service1['condoms_revisits'] = (
    df_service1['clients_receiving_female_condoms_re_visits'] +
    df_service1['client_receiving_male_condoms_re_visits']
)

df_service1['pills_new'] = (
    df_service1['emergency_contraceptive_pill_new_clients'] +
    df_service1['pills_combined_oral_contraceptive_new_clients'] +
    df_service1['pills_progestin_only_new_clients']
)

df_service1['pills_revisits'] = (
    df_service1['emergency_contraceptive_pill_re_visits'] +
    df_service1['pills_combined_oral_contraceptive_re_visits'] +
    df_service1['pills_progestin_only_re_visits']
)

df_service1['injectable_new'] = (
    df_service1['2020_fp_injections_dmpa_im_new_clients'] +
    df_service1['2020_fp_injections_dmpa_sc_new_clients']
)

df_service1['injectable_revisits'] = (
    df_service1['2020_fp_injections_dmpa_im_re_visits'] +
    df_service1['2020_fp_injections_dmpa_sc_re_visits']
)

df_service1['implants_new'] = (
    df_service1['2020_implants_insertion_1_rod_ist_time_insertion'] +
    df_service1['2020_implants_insertion_2_rod_ist_time_insertion']
)

df_service1['implants_revisits'] = (
    df_service1['2020_implants_insertion_1_rod_re_insertion'] +
    df_service1['2020_implants_insertion_2_rod_re_insertion']
)

df_service1['iucd_new'] = (
    df_service1['2020_iucd_insertion_hormonal_ist_time_insertion'] +
    df_service1['2020_iucd_insertion_non_hormonal_ist_time_insertion']
```

```
)  
  
df_service1['iucd_revisits'] = (  
    df_service1['2020_iucd_insertion_hormonal_re_insertion'] +  
    df_service1['2020_iucd_insertion_non_hormonal_re_insertion'])  
  
df_service1['surgical_new'] = (  
    df_service1['2020_voluntary_surgical_contraception_vasectomy_ist_time_insertion'] +  
    df_service1['2020_voluntary_surgical_contraception_btl_ist_time_insertion'])  
  
df_service1['surgical_revisits'] = (  
    df_service1['2020_voluntary_surgical_contraception_vasectomy_re_insertion'] +  
    df_service1['2020_voluntary_surgical_contraception_btl_re_insertion'])  
  
df_service1['traditional_new'] = (  
    df_service1['2020_clients_given_cycle_beads_new_clients'] +  
    df_service1['clients_counselled_natural_family_planning_new_clients'])  
  
df_service1['traditional_revisits'] = (  
    df_service1['2020_clients_given_cycle_beads_re_visits'] +  
    df_service1['clients_counselled_natural_family_planning_re_visits'])  
)
```



In [43]: df_service1

Out[43]:

| | year_month | country | county | uid | county_code | estimated_number_of_pregnant |
|------|------------|---------|----------------------|-------------|--------------|------------------------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | KE_County_23 | |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | KE_County_29 | |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | KE_County_24 | |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | KE_County_36 | |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | KE_County_47 | |
| ... | ... | ... | ... | ... | ... | ... |
| 6199 | 202409 | Kenya | Isiolo County | bzOfj0iwfDH | KE_County_11 | |
| 6200 | 202409 | Kenya | Trans Nzoia County | mThvosEfAU | KE_County_26 | |
| 6201 | 202409 | Kenya | Nakuru County | ob6SxuRcqU4 | KE_County_32 | |
| 6202 | 202409 | Kenya | Tharaka Nithi County | T4urHM47nlm | KE_County_13 | |
| 6203 | 202409 | Kenya | Nyeri County | ptWVfaCldVx | KE_County_19 | |

6204 rows × 74 columns



FP Method overall banding (pills, condoms, injectables,implants,iucd & surgical)

```
In [44]: # Combined method categories
df_service1['condoms'] = df_service1['condoms_new'] + df_service1['condoms_revisits']
df_service1['pills'] = df_service1['pills_new'] + df_service1['pills_revisits']
df_service1['injectables'] = df_service1['injectable_new'] + df_service1['injectable_revisits']
df_service1['implants'] = df_service1['implants_new'] + df_service1['implants_revisits']
df_service1['iucd'] = df_service1['iucd_new'] + df_service1['iucd_revisits']
df_service1['surgical'] = df_service1['surgical_new'] + df_service1['surgical_revisits']

# Compute Total modern FP methods
df_service1['total_modern_fp'] = (
    df_service1['condoms'] +
    df_service1['pills'] +
    df_service1['injectables'] +
    df_service1['implants'] +
    df_service1['iucd'] +
    df_service1['surgical']
)

# Compute Total traditional FP methods
df_service1['traditional'] = df_service1['traditional_new'] + df_service1['traditional_revisits']
```

```
In [45]: # List of columns to keep
keep_cols = [
    'year_month', 'country', 'county', 'uid', 'uid_code', 'uid_year', 'county_code',
    'adolescent_10_24_receiving_fp_new',
    'adolescent_10_24_receiving_fp_revisits',
    'adults_25+_receiving_fp_services_new',
    'adults_25+_receiving_fp_services_revisits',
    'condoms_new', 'condoms_revisits', 'traditional_new',
    'pills_new', 'pills_revisits', 'traditional_revisits',
    'injectable_new', 'injectable_revisits', 'traditional',
    'implants_new', 'implants_revisits',
    'iucd_new', 'iucd_revisits',
    'surgical_new', 'surgical_revisits',
    'condoms', 'pills', 'injectables', 'implants', 'iucd', 'surgical',
    'total_modern_fp', 'total_cyp'
]

# Keep only the specified columns
df_service1 = df_service1[keep_cols]
```

Drop underlying columns

```
In [46]: df_service1.columns
```

```
Out[46]: Index(['year_month', 'country', 'county', 'uid', 'uid_code', 'uid_year',
       'county_code', 'adolescent_10_24_receiving_fp_new',
       'adolescent_10_24_receiving_fp_revisits',
       'adults_25+_receiving_fp_services_new',
       'adults_25+_receiving_fp_services_revisits', 'condoms_new',
       'condoms_revisits', 'traditional_new', 'pills_new', 'pills_revisits',
       'traditional_revisits', 'injectable_new', 'injectable_revisits',
       'traditional', 'implants_new', 'implants_revisits', 'iucd_new',
       'iucd_revisits', 'surgical_new', 'surgical_revisits', 'condoms',
       'pills', 'injectables', 'implants', 'iucd', 'surgical',
       'total_modern_fp', 'total_cyp'],
      dtype='object')
```

Joining the datasets

A left join was used to merge the four datasets. The service data was used as the base data. Organisation unit id code (uid_code) was used to join the four datasets

```
In [47]: # Merge df_service1 with df_commodity1
fp_service_comm_df = pd.merge(df_service1, df_commodity1, how='left', on='uid_code')

fp_service_comm_df
```

Out[47]:

| | year_month_x | country_x | county_x | uid_x | uid_code | ui |
|------|--------------|-----------|----------------------|-------------|--------------------|-------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | 201404_kphDeKClFch | 2014_kphDe |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | 201404_t0J75eHKxz5 | 2014_t0J75e |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | 201404_XWALbfAPa6n | 2014_XWALb |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | 201404_HMNARUV2CW4 | 2014_HMNARU |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | 201404_jkG3zaihdSs | 2014_jkG3z |
| ... | ... | ... | ... | ... | ... | ... |
| 6199 | 202409 | Kenya | Isiolo County | bzOfj0iwfDH | 202409_bzOfj0iwfDH | 2024_bzOfj |
| 6200 | 202409 | Kenya | Trans Nzoia County | mThvosEfIAU | 202409_mThvosEfIAU | 2024_mThvc |
| 6201 | 202409 | Kenya | Nakuru County | ob6SxuRcqU4 | 202409_ob6SxuRcqU4 | 2024_ob6Sxu |
| 6202 | 202409 | Kenya | Tharaka Nithi County | T4urHM47nlm | 202409_T4urHM47nlm | 2024_T4urHl |
| 6203 | 202409 | Kenya | Nyeri County | ptWVfaCldVx | 202409_ptWVfaCldVx | 2024_ptWVf |

6204 rows × 81 columns



In [48]: # Rename the column

```
fp_service_comm_df = fp_service_comm_df.rename(columns={"county_code_x": "county_code"})

fp_service_comm_df
```

Out[48]:

| | year_month_x | country_x | county_x | uid_x | uid_code | ui |
|------|--------------|-----------|----------------------|-------------|--------------------|-------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | 201404_kphDeKClFch | 2014_kphDe |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | 201404_t0J75eHKxz5 | 2014_t0J75e |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | 201404_XWALbfAPa6n | 2014_XWALb |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | 201404_HMNARUV2CW4 | 2014_HMNARU |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | 201404_jkG3zaihdSs | 2014_jkG3z |
| ... | ... | ... | ... | ... | ... | ... |
| 6199 | 202409 | Kenya | Isiolo County | bzOfj0iwfDH | 202409_bzOfj0iwfDH | 2024_bzOfj |
| 6200 | 202409 | Kenya | Trans Nzoia County | mThvosEfIAU | 202409_mThvosEfIAU | 2024_mThvc |
| 6201 | 202409 | Kenya | Nakuru County | ob6SxuRcqU4 | 202409_ob6SxuRcqU4 | 2024_ob6Sxu |
| 6202 | 202409 | Kenya | Tharaka Nithi County | T4urHM47nlm | 202409_T4urHM47nlm | 2024_T4urH |
| 6203 | 202409 | Kenya | Nyeri County | ptWVfaCldVx | 202409_ptWVfaCldVx | 2024_ptWVf |

6204 rows × 81 columns



In [49]: # Replace non-finite values (NaN, inf, -inf) with 0

```
fp_service_comm_df = fp_service_comm_df.replace([np.inf, -np.inf], np.nan).fillna(0)
```

```
In [50]: # Convert float columns to int64
df_float_cols = fp_service_comm_df.select_dtypes(include=['float', 'float64']).columns
df_float_cols
```

```
Out[50]: Index(['total_cyp', 'periodid', 'year_month_y',
       'pills_combined_oral_contraceptive_stock_losses',
       'pills_combined_oral_contraceptive_stock_dispensed',
       'pills_combined_oral_contraceptive_stock_at_hand',
       'pills_combined_oral_contraceptive_stock_requested',
       'pills_combined_oral_contraceptive_stock_received',
       'pills_emergency_pill_stock_losses',
       'pills_emergency_pill_stock_dispensed',
       'pills_emergency_pill_stock_at_hand',
       'pills_emergency_pill_stock_requested',
       'pills_emergency_pill_stock_received',
       'pills_progestin_only_pills_stock_losses',
       'pills_progestin_only_pills_stock_dispensed',
       'pills_progestin_only_pills_stock_at_hand',
       'pills_progestin_only_pills_stock_requested',
       'pills_progestin_only_pills_stock_received',
       'condoms_female_condom_stock_losses',
       'condoms_female_condom_stock_dispensed',
       'condoms_female_condom_stock_at_hand',
       'condoms_female_condom_stock_requested',
       'condoms_female_condom_stock_received',
       'condoms_male_condom_stock_losses',
       'condoms_male_condom_stock_dispensed',
       'condoms_male_condom_stock_at_hand',
       'condoms_male_condom_stock_requested',
       'condoms_male_condom_stock_received', 'injectables_stock_losses',
       'injectables_stock_dispensed', 'injectables_stock_at_hand',
       'injectables_stock_requested', 'injectables_stock_received',
       'implants_stock_losses', 'implants_stock_dispensed',
       'implants_stock_at_hand', 'implants_stock_requested',
       'implants_stock_received', 'iud_stock_losses', 'iud_stock_dispensed',
       'iud_stock_at_hand', 'iud_stock_requested', 'iud_stock_received'],
      dtype='object')
```

```
In [51]: fp_service_comm_df[df_float_cols] = fp_service_comm_df[df_float_cols].astype('int')
```

In [52]: `fp_service_comm_df.info()`

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6204 entries, 0 to 6203
Data columns (total 81 columns):
 #   Column           Non-Null Count Dtype
 ---  -- 
 0   year_month_x    6204 non-null   int64
 1   country_x       6204 non-null   object
 2   county_x        6204 non-null   object
 3   uid_x           6204 non-null   object
 4   uid_code         6204 non-null   object
 5   uid_year         6204 non-null   object
 6   county_code      6204 non-null   object
 7   adolescent_10_24_receiving_fp_new 6204 non-null   int64
 8   adolescent_10_24_receiving_fp_revisits 6204 non-null   int64
 9   adults_25+_receiving_fp_services_new 6204 non-null   int64
 10  adults_25+_receiving_fp_services_revisits 6204 non-null   int64
 11  condoms_new      6204 non-null   int64
 12  condoms_revisits 6204 non-null   int64
 13  traditional_new 6204 non-null   int64
 14  pills_new        6204 non-null   int64
 15  pills_revisits   6204 non-null   int64
 16  traditional_revisits 6204 non-null   int64
 17  injectable_new   6204 non-null   int64
 18  injectable_revisits 6204 non-null   int64
 19  traditional      6204 non-null   int64
 20  implants_new     6204 non-null   int64
 21  implants_revisits 6204 non-null   int64
 22  iucd_new         6204 non-null   int64
 23  iucd_revisits   6204 non-null   int64
 24  surgical_new    6204 non-null   int64
 25  surgical_revisits 6204 non-null   int64
 26  condoms          6204 non-null   int64
 27  pills            6204 non-null   int64
 28  injectables     6204 non-null   int64
 29  implants         6204 non-null   int64
 30  iucd             6204 non-null   int64
 31  surgical         6204 non-null   int64
 32  total_modern_fp 6204 non-null   int64
 33  total_cyp        6204 non-null   int64
 34  periodid         6204 non-null   int64
 35  periodname       6204 non-null   object
 36  year_month_y    6204 non-null   int64
 37  country_y        6204 non-null   object
 38  county_y         6204 non-null   object
 39  uid_y            6204 non-null   object
 40  county_code_y    6204 non-null   object
 41  pills_combined_oral_contraceptive_stock_losses 6204 non-null   int64
 42  pills_combined_oral_contraceptive_stock_dispensed 6204 non-null   int64
 43  pills_combined_oral_contraceptive_stock_at_hand 6204 non-null   int64
 44  pills_combined_oral_contraceptive_stock_requested 6204 non-null   int64
 45  pills_combined_oral_contraceptive_stock_received 6204 non-null   int64
 46  pills_emergency_pill_stock_losses 6204 non-null   int64
 47  pills_emergency_pill_stock_dispensed 6204 non-null   int64
 48  pills_emergency_pill_stock_at_hand 6204 non-null   int64
 49  pills_emergency_pill_stock_requested 6204 non-null   int64
 50  pills_emergency_pill_stock_received 6204 non-null   int64
 51  pills_progestin_only_pills_stock_losses 6204 non-null   int64
```

```
52 pills_progestin_only_pills_stock_dispensed      6204 non-null    int64
53 pills_progestin_only_pills_stock_at_hand       6204 non-null    int64
54 pills_progestin_only_pills_stock_requested     6204 non-null    int64
55 pills_progestin_only_pills_stock_received      6204 non-null    int64
56 condoms_female_condom_stock_losses             6204 non-null    int64
57 condoms_female_condom_stock_dispensed          6204 non-null    int64
58 condoms_female_condom_stock_at_hand            6204 non-null    int64
59 condoms_female_condom_stock_requested          6204 non-null    int64
60 condoms_female_condom_stock_received           6204 non-null    int64
61 condoms_male_condom_stock_losses              6204 non-null    int64
62 condoms_male_condom_stock_dispensed           6204 non-null    int64
63 condoms_male_condom_stock_at_hand             6204 non-null    int64
64 condoms_male_condom_stock_requested          6204 non-null    int64
65 condoms_male_condom_stock_received           6204 non-null    int64
66 injectables_stock_losses                     6204 non-null    int64
67 injectables_stock_dispensed                  6204 non-null    int64
68 injectables_stock_at_hand                   6204 non-null    int64
69 injectables_stock_requested                 6204 non-null    int64
70 injectables_stock_received                  6204 non-null    int64
71 implants_stock_losses                      6204 non-null    int64
72 implants_stock_dispensed                   6204 non-null    int64
73 implants_stock_at_hand                    6204 non-null    int64
74 implants_stock_requested                  6204 non-null    int64
75 implants_stock_received                   6204 non-null    int64
76 iud_stock_losses                          6204 non-null    int64
77 iud_stock_dispensed                       6204 non-null    int64
78 iud_stock_at_hand                        6204 non-null    int64
79 iud_stock_requested                      6204 non-null    int64
80 iud_stock_received                       6204 non-null    int64
dtypes: int64(70), object(11)
memory usage: 3.9+ MB
```

```
In [53]: # Perform a left merge to retain all rows from fp_service_comm_df
fp_service_comm_pop_df = fp_service_comm_df.merge(
    df_population1[['uid_year', 'eligible_fp']],
    on='uid_year',
    how='left'
)

# Preview the merged DataFrame
fp_service_comm_pop_df.head()
```

Out[53]:

| | year_month_x | country_x | county_x | uid_x | uid_code | uid_y |
|---|--------------|-----------|-------------------|-------------|--------------------|------------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | 201404_kphDeKClFch | 2014_kphDeKCl |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | 201404_t0J75eHKxz5 | 2014_t0J75eHK |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | 201404_XWALbfAPa6n | 2014_XWALbfAPa6n |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | 201404_HMNARUV2CW4 | 2014_HMNARUV2C |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | 201404_jkG3zaihdSs | 2014_jkG3zaihdSs |

5 rows × 82 columns

In [54]: # Merge df_core_health_workforce

```
data_df = fp_service_comm_pop_df.merge(
    df_core_health_workforce[['uid_code', 'core_health_workforce_per_10,000population']],
    on='uid_code', how='left'
)
```

In [55]: # Merge df_demand_satisfied

```
data_df = data_df.merge(
    df_demand_satisfied1[['uid_code', 'Demand_Satisfied_by_Modern_Methods (%)']],
    on='uid_code', how='left'
)
```

In [56]: # Merge df_mcpr

```
data_df = data_df.merge(
    df_mcpr1[['uid_code', 'mCPR (Married Women, %)']],
    on='uid_code', how='left'
)
```

```
In [57]: # Merge df_teenage_pregnancy
data_df = data_df.merge(
    df_teenage_pregnancy1[['uid_code', 'Teenage Pregnancy Rate (15-19, %)']],
    on='uid_code', how='left'
)
```

```
In [58]: # Merge df_unmet_need
data_df = data_df.merge(
    df_unmet_need1[['uid_code', 'Total Unmet Need (Married Women, %)']],
    on='uid_code', how='left'
)
```

```
In [59]: # Rename the columns by removing suffixes
data_df.columns = [
    col.replace('_x_x', '')
    .replace('_y_y', '')
    .replace('_x', '')
    .replace('_y', '')
    for col in data_df.columns
]
```

```
In [60]: # Preview the data to make sure the columns have been renamed
data_df.head()
```

Out[60]:

| | year_month | country | county | uid | uid_code | uidear |
|---|------------|---------|-------------------|-------------|--------------------|------------------|
| 0 | 201404 | Kenya | Turkana County | kphDeKClFch | 201404_kphDeKClFch | 2014_kphDeKClFch |
| 1 | 201404 | Kenya | Nandi County | t0J75eHKxz5 | 201404_t0J75eHKxz5 | 2014_t0J75eHKxz5 |
| 2 | 201404 | Kenya | West Pokot County | XWALbfAPa6n | 201404_XWALbfAPa6n | 2014_XWALbfAPa6n |
| 3 | 201404 | Kenya | Bomet County | HMNARUV2CW4 | 201404_HMNARUV2CW4 | 2014_HMNARUV2CW4 |
| 4 | 201404 | Kenya | Nairobi County | jkG3zaihdSs | 201404_jkG3zaihdSs | 2014_jkG3zaihdSs |

5 rows × 87 columns



```
In [61]: data_df.shape
```

Out[61]: (6204, 87)

```
In [62]: # Remove duplicate columns  
data_df = data_df.loc[:, ~data_df.columns.duplicated()]
```

```
In [63]: # Checking the merged data for duplicates  
data_df.duplicated().sum()
```

```
Out[63]: 0
```

```
In [64]: # Replace " county" Label with blanks in county column  
  
data_df['county'] = data_df['county'].str.replace(' County', '', regex=False)  
print(data_df[['county']].head())
```

```
      county  
0    Turkana  
1      Nandi  
2  West Pokot  
3      Bomet  
4    Nairobi
```

Export .csv file for data_df

```
In [65]: #data_df_sorted = data_df.sort_values(by='year_month', ascending=False)  
  
# Export to CSV  
#data_df_sorted.to_csv("output/data_df.csv", index=False, encoding='utf-8')
```

Composite Calculations

```
In [170]: # Calculate total users receiving FP services
data_df['total_users_receiving_fp'] = (
    data_df['adolescent_10_24_receiving_fp_new'] +
    data_df['adolescent_10_24_receiving_fp_revisits'] +
    data_df['adults_25+_receiving_fp_services_new'] +
    data_df['adults_25+_receiving_fp_services_revisits']
)

# Total actual new (Modern fp)
data_df['total_actual_new_modern_fp_methods'] = data_df[['pills_new', 'condoms_new',
                                                          'injectable_new', 'implants_new',
                                                          'iucd_new', 'surgical_new']]

# Total actual revisits (Modern fp)
data_df['total_actual_revisits_modern_fp_methods'] = data_df[['pills_revisits',
                                                               'injectable_revisits',
                                                               'implants_revisits',
                                                               'surgical_revisits']]

# Total actual traditional fp
data_df['total_actual_traditional_methods'] = data_df[['traditional_revisits', 'traditional_new']]

# Total actual modern fp
data_df['total_actual_modern_fp'] = (
    data_df['total_actual_new_modern_fp_methods'] +
    data_df['total_actual_revisits_modern_fp_methods']
)

# Total FP (both Modern and Traditional)
data_df['total_fp'] = (data_df['total_actual_modern_fp'] +
                      data_df['total_actual_traditional_methods'])

# Proportion adolescents receiving fp
data_df['proportion_adolescents_10_24_yrs_receiving_fp'] = ((data_df['adolescent_10_24_receiving_fp_new'] +
                                                               data_df['adolescent_10_24_receiving_fp_revisits']) /
                                                               data_df['total_users_receiving_fp'])

# Number of months
data_df['number_of_months'] = min(data_df['year_month'].nunique(), 12)

# Actual mCPR
data_df['actual_mcpr'] = data_df['total_actual_modern_fp'] / data_df['eligible_fp']

# Actual mCPR monthly
data_df['actual_mcpr_monthly'] = data_df['total_modern_fp'] / (data_df['eligible_fp'] * 12)

# Number of women(10-49 yrs) with unmet need for fp
data_df['number_women_10_49_yrs_with_unmet_need_for_fp'] = data_df['eligible_fp'] * data_df['proportion_adolescents_10_24_yrs_receiving_fp']

# Unmet need
data_df['actual_unmet_need_for_modern_fp'] = (
    data_df['number_women_10_49_yrs_with_unmet_need_for_fp'] /
    data_df['eligible_fp']
)
```

```

# Actual core health workers-2013
data_df['actual_core_health_workers_2013_10000_eligible_fp'] = (data_df['eligible_10000_fp'] * data_df['actual_mcpr'])

# Actual total demand
data_df['actual_total_demand_for_fp'] = data_df['actual_mcpr'] + data_df['actual_mcpr'] * data_df['actual_core_health_workers_2013_10000_eligible_fp']

# Actual demand satisfied
data_df['actual_demand_satisfied'] = data_df['actual_mcpr'] / data_df['actual_total_demand_for_fp']

# Combine pills (COCs + POCs)
data_df['total_pills_dispensed'] = (
    data_df['pills_combined_oral_contraceptive_stock_dispensed'] +
    data_df['pills_progestin_only_pills_stock_dispensed']
)

# Combine condoms (male + female)
data_df['total_condoms_dispensed'] = (
    data_df['condoms_male_condom_stock_dispensed'] +
    data_df['condoms_female_condom_stock_dispensed']
)

# Total all commodities dispensed
data_df['total_commodities_dispensed'] = (
    data_df['total_pills_dispensed'] +
    data_df['total_condoms_dispensed'] +
    data_df['injectables_stock_dispensed'] +
    data_df['implants_stock_dispensed'] +
    data_df['iud_stock_dispensed']
)

# Calculate average monthly commodities dispensed
data_df['average_monthly_commodities_dispensed'] = (
    data_df['total_commodities_dispensed'] / data_df['number_of_months']
)

```

c) Exploratory Data Analysis (EDA)

Descriptive Statistics

i) FP service data descriptives

In [81]: # List of columns to describe

```
eda_cols = [
    'condoms_new', 'condoms_revisits',
    'pills_new', 'pills_revisits',
    'injectable_new', 'injectable_revisits',
    'implants_new', 'implants_revisits',
    'iucd_new', 'iucd_revisits',
    'surgical_new', 'surgical_revisits'
]

# Basic descriptive statistics
eda_summary = df_service1[eda_cols].describe().T
eda_summary['missing'] = df_service1[eda_cols].isnull().sum()
eda_summary['zeros'] = (df_service1[eda_cols] == 0).sum()
eda_summary['unique'] = df_service1[eda_cols].nunique()
eda_summary['dtype'] = df_service1[eda_cols].dtypes

# Show the summary
print(eda_summary)
```

| | count | mean | std | min | 25% | 50% | \ |
|---------------------|---------|-------------|-------------|-------|--------|-------|---|
| condoms_new | 6204.0 | 1012.755480 | 2179.557079 | 4.0 | 191.75 | 483.0 | |
| condoms_revisits | 6204.0 | 502.967924 | 1201.564246 | 0.0 | 43.00 | 144.0 | |
| pills_new | 6204.0 | 622.910058 | 860.123122 | 0.0 | 183.00 | 387.0 | |
| pills_revisits | 6204.0 | 909.946325 | 1276.593789 | 0.0 | 178.00 | 492.5 | |
| injectable_new | 6204.0 | 426.390071 | 762.405961 | 0.0 | 0.00 | 0.0 | |
| injectable_revisits | 6204.0 | 1470.093649 | 2597.655200 | 0.0 | 0.00 | 0.0 | |
| implants_new | 6204.0 | 427.715506 | 881.975366 | 0.0 | 0.00 | 0.0 | |
| implants_revisits | 6204.0 | 187.019987 | 380.439558 | 0.0 | 0.00 | 0.0 | |
| iucd_new | 6204.0 | 52.249194 | 137.231216 | 0.0 | 0.00 | 0.0 | |
| iucd_revisits | 6204.0 | 17.744842 | 50.077124 | 0.0 | 0.00 | 0.0 | |
| surgical_new | 6204.0 | 3.200838 | 10.788603 | 0.0 | 0.00 | 0.0 | |
| surgical_revisits | 6204.0 | 0.026596 | 0.460010 | 0.0 | 0.00 | 0.0 | |
| | 75% | max | missing | zeros | unique | dtype | |
| condoms_new | 1082.00 | 86502.0 | 0 | 0 | 2194 | int64 | |
| condoms_revisits | 476.25 | 28442.0 | 0 | 176 | 1501 | int64 | |
| pills_new | 703.00 | 8793.0 | 0 | 1 | 1639 | int64 | |
| pills_revisits | 1039.25 | 13292.0 | 0 | 3 | 2186 | int64 | |
| injectable_new | 547.25 | 5757.0 | 0 | 3738 | 1504 | int64 | |
| injectable_revisits | 2294.00 | 17377.0 | 0 | 3734 | 2093 | int64 | |
| implants_new | 487.00 | 11639.0 | 0 | 3750 | 1532 | int64 | |
| implants_revisits | 224.25 | 4972.0 | 0 | 3777 | 1050 | int64 | |
| iucd_new | 42.00 | 1343.0 | 0 | 3819 | 486 | int64 | |
| iucd_revisits | 12.00 | 630.0 | 0 | 4050 | 262 | int64 | |
| surgical_new | 1.00 | 340.0 | 0 | 4594 | 74 | int64 | |
| surgical_revisits | 0.00 | 20.0 | 0 | 6167 | 12 | int64 | |


```
In [82]: # Columns to describe
eda_columns = [
    'year_month', 'country', 'county', 'uid', 'uid_code', 'county_code',
    'adolescent_10_24_receiving_fp_new', 'adolescent_10_24_receiving_fp_revisits',
    'adults_25+_receiving_fp_services_new', 'adults_25+_receiving_fp_services_rev',
    'condoms_new', 'condoms_revisits',
    'pills_new', 'pills_revisits',
    'injectable_new', 'injectable_revisits',
    'implants_new', 'implants_revisits',
    'iucd_new', 'iucd_revisits',
    'surgical_new', 'surgical_revisits',
    'condoms', 'pills', 'injectables', 'implants', 'iucd', 'surgical',
    'total_modern_fp', 'total_cyp'
]

# Subset the DataFrame
df_eda = df_service1[eda_columns]

# 1. General info
print("== Basic Info ==")
print(df_eda.info())

# 2. Missing values
print("== Missing Values ==")
print(df_eda.isnull().sum())

# 3. Descriptive statistics for numeric columns
print("== Summary Statistics ==")
print(df_eda.describe().T)

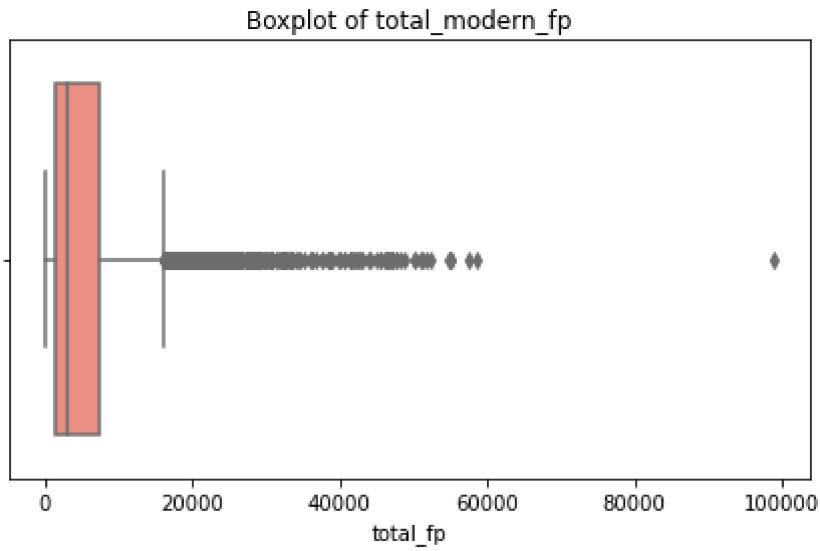
# 4. Unique value counts for categorical columns
cat_cols = ['year_month', 'country', 'county', 'uid', 'uid_code', 'county_code']
print("== Unique Values in Categorical Columns ==")
for col in cat_cols:
    print(f'{col}: {df_eda[col].nunique()} unique values')

# 5. Trends over time (optional)
if pd.api.types.is_datetime64_any_dtype(df_eda['year_month']):
    trend_data = df_eda.groupby('year_month')['total_fp'].sum()
    trend_data.plot(title="Total FP Over Time", figsize=(10, 4))
    plt.ylabel("Total FP")
    plt.xlabel("Month")
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()

# 6. Distribution plots for select variables (optional)
num_vars = ['total_modern_fp', 'total_cyp', 'condoms', 'pills', 'injectables']
for var in num_vars:
    plt.figure(figsize=(6, 3))
    sns.histplot(df_eda[var], bins=30, kde=True)
    plt.title(f'Distribution of {var}')
    plt.tight_layout()
    plt.show()
```

```
==== Basic Info ====
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6204 entries, 0 to 6203
Data columns (total 30 columns):
 #   Column           Non-Null Count Dtype
 ---  ----
 0   year_month      6204 non-null   int64
 1   country          6204 non-null   object
 2   county           6204 non-null   object
 3   uid              6204 non-null   object
 4   uid_code         6204 non-null   object
 5   county_code      6204 non-null   object
 6   adolescent_10_24_receiving_fp_new 6204 non-null   int64
 7   adolescent_10_24_receiving_fp_revisits 6204 non-null   int64
 8   adults_25+_receiving_fp_services_new 6204 non-null   int64
 9   adults_25+_receiving_fp_services_revisits 6204 non-null   int64
 10  condoms_new      6204 non-null   int64
 11  condoms_revisits 6204 non-null   int64
 12  pills_new        6204 non-null   int64
 13  pills_revisits   6204 non-null   int64
```

```
In [83]: # Boxplot of total modern fp
plt.figure(figsize=(6,4))
sns.boxplot(x=data_df['total_modern_fp'].dropna(), color='salmon')
plt.title('Boxplot of total_modern_fp') # Set title
plt.xlabel('total_fp') # Set X_label
plt.tight_layout() # Spacing
plt.show()
```



```
In [84]: # Convert year_month values into datetime objects
data_df['year_month'] = data_df['year_month'].astype(str) # Convert to string

# Split into year and month
data_df['year'] = data_df['year_month'].str[:4].astype(int)
data_df['month'] = data_df['year_month'].str[4:]).astype(int)
```

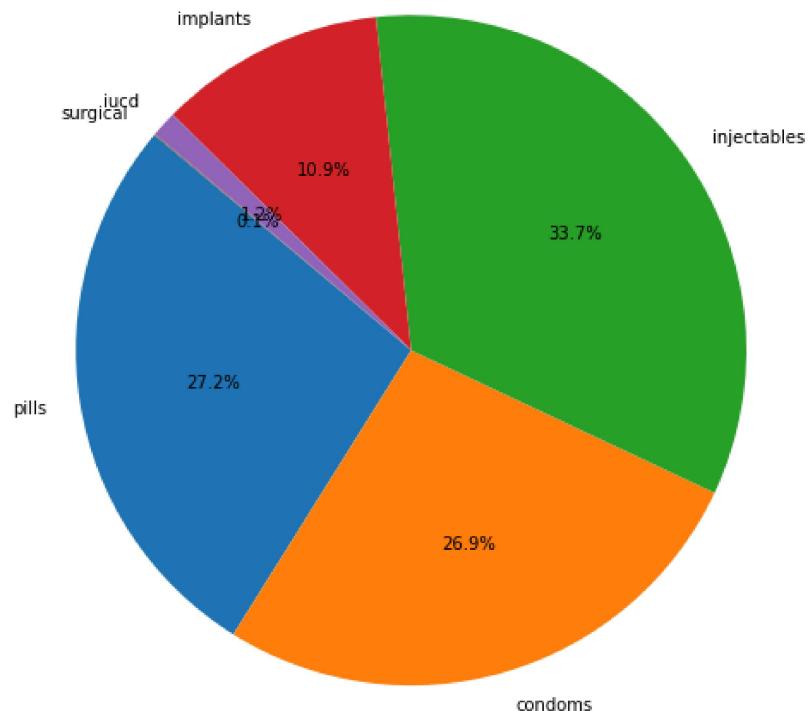
What's the FP method mix composition for the period in focus?

```
In [85]: import matplotlib.pyplot as plt
# Group data by year_month and sum the method counts
method_mix = data_df.groupby('year_month')[['pills', 'condoms', 'injectables', 'implants', 'surgical']].sum()

# Calculate the total for each method across the entire period
total_method_counts = method_mix.sum()

# Plotting the pie chart
plt.figure(figsize=(10, 8))
plt.pie(total_method_counts, labels=total_method_counts.index, autopct='%1.1f%%')
plt.title('Majority of the FP methods provided bewteen 2013 and 2024 were short term methods (pills + condoms + injectables ~ 87.2%)')
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

Majority of the FP methods provided bewteen 2013 and 2024 were short term methods (pills + condoms + injectables ~ 87.2%)



What's the FP method volume by county?

In [86]: # Method Mix volume by county

```
# Sum the total counts for each method by county
county_method_totals = data_df.groupby('county')[['pills', 'condoms', 'injectables', 'implants', 'lucd', 'surgical']].sum()

# Calculate the total for all methods per county for sorting
county_method_totals['total'] = county_method_totals.sum(axis=1)

# Sort counties by total count in descending order
county_method_totals = county_method_totals.sort_values('total', ascending=False)

# Convert values to thousands and format to 0 decimal places
county_method_totals_thousands = county_method_totals / 1000

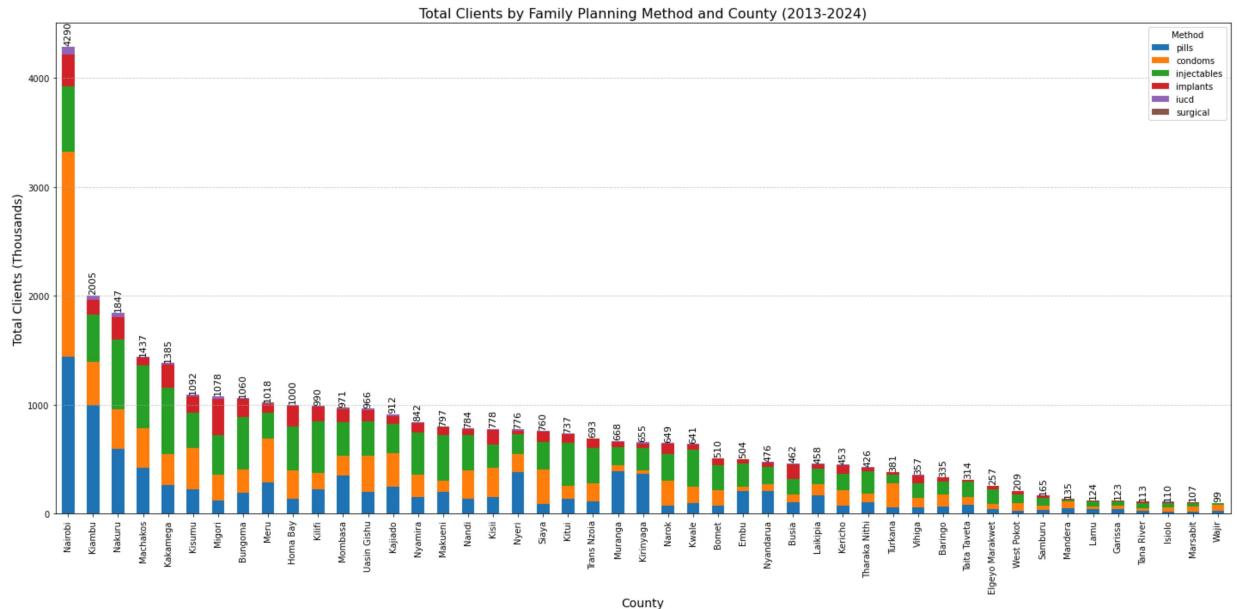
# Plotting the stacked bar chart
plt.figure(figsize=(20, 10))
ax = county_method_totals_thousands.plot(kind='bar', stacked=True, figsize=(20, 10))

# Add Labels and title
plt.xlabel('County', fontsize=14)
plt.ylabel('Total Clients (Thousands)', fontsize=14)
plt.title('Total Clients by Family Planning Method and County (2013-2024)', fontsize=14)
plt.xticks(rotation=90)
plt.legend(title='Method')
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Add total labels on top of bars
for i, county in enumerate(county_method_totals_thousands.index):
    total_value = county_method_totals_thousands.loc[county].sum()
    # Position the text slightly above the bar. Adjust text position as needed.
    ax.text(i, total_value + 10, f'{total_value:.0f}', ha='center', va='bottom', color='black')

plt.tight_layout() # Adjust Layout to prevent labels overlapping
plt.show()
```

<Figure size 1440x720 with 0 Axes>



What are the trends in FP Method uptake over time?

```
In [87]: # Plot FP method uptake trends over time

# Group data by year_month and sum the method counts
method_uptake_time = data_df.groupby('year_month')[['pills', 'condoms', 'injectable']]

# Convert values to thousands
method_uptake_time_thousands = method_uptake_time / 1000

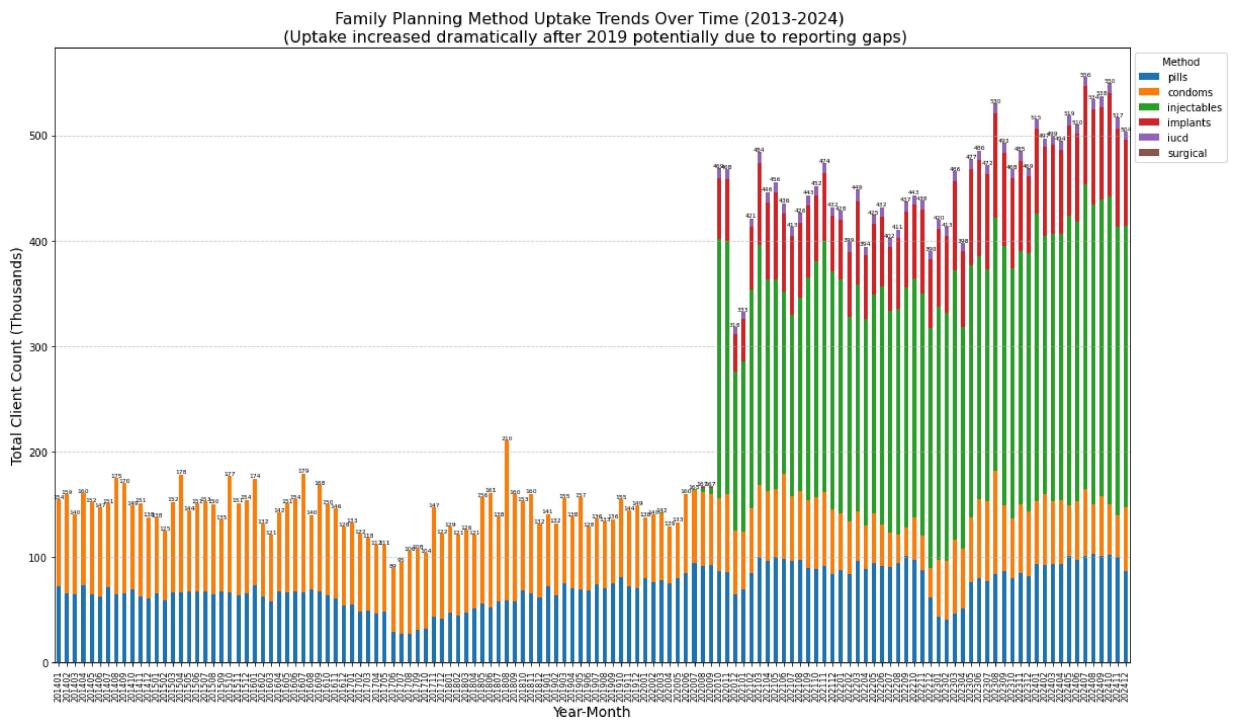
# Plotting the stacked bar chart
plt.figure(figsize=(20, 10))
ax = method_uptake_time_thousands.plot(kind='bar', stacked=True, figsize=(20, 10))

# Add Labels and title
plt.xlabel('Year-Month', fontsize=14)
plt.ylabel('Total Client Count (Thousands)', fontsize=14)
plt.title('Family Planning Method Uptake Trends Over Time (2013-2024) \n (Uptake')
plt.xticks(rotation=90, fontsize=8)
plt.legend(title='Method', loc='upper left', bbox_to_anchor=(1, 1))
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Add total Labels on top of bars
for i, year_month in enumerate(method_uptake_time_thousands.index):
    total_value = method_uptake_time_thousands.loc[year_month].sum()
    # Position the text slightly above the bar. Adjust text position as needed.
    ax.text(i, total_value, f'{total_value:.0f}', ha='center', va='bottom', fontweight='bold')

plt.tight_layout(rect=[0, 0, 0.85, 1]) # Adjust Layout to prevent legend overlap
plt.show()
```

<Figure size 1440x720 with 0 Axes>



Total FP Methods vs Women Eligible for FP Over Time

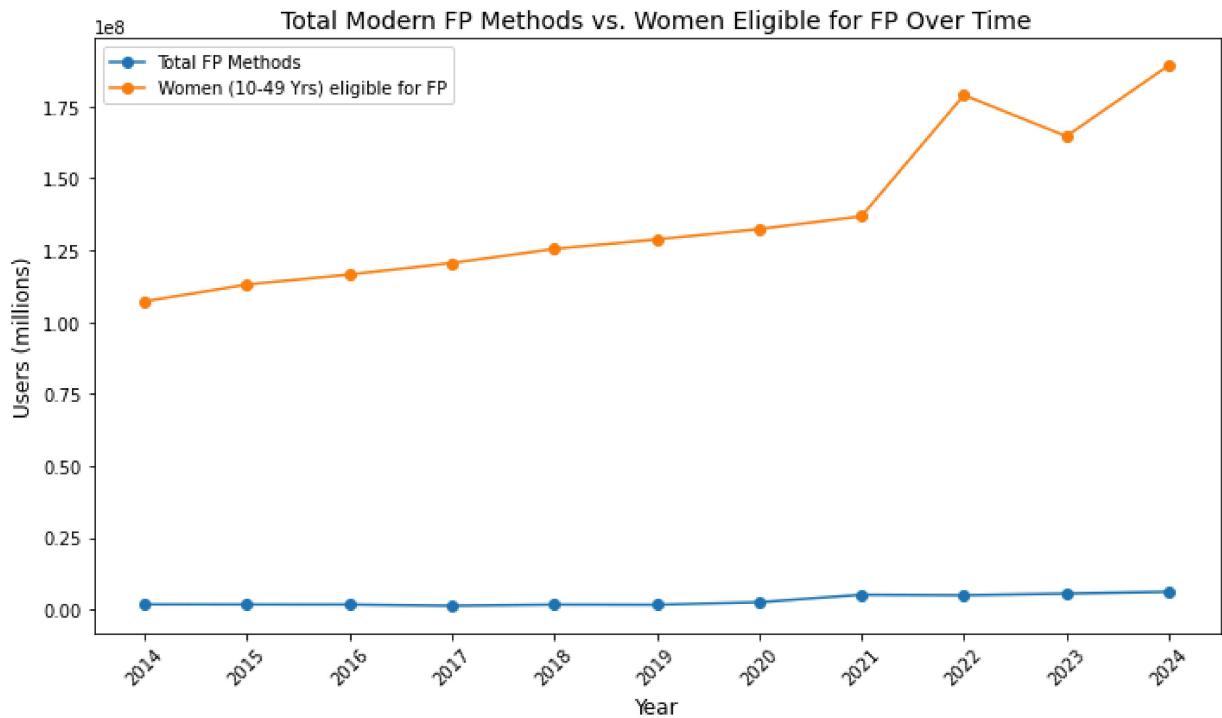
```
In [88]: # Plot Total modern FP Methods vs. Women Eligible for FP Over Time

# Group by year and sum 'total_fp', then get the maximum 'eligible_fp' for each year
trends = data_df.groupby(data_df['year'])[['total_modern_fp', 'eligible_fp']].agg(
    {'total_modern_fp': 'sum', 'eligible_fp': 'sum'}
)

# Plotting
plt.figure(figsize=(10, 6))
plt.plot(trends.index, trends['total_modern_fp'], label='Total FP Methods', marker='o')
plt.plot(trends.index, trends['eligible_fp'], label='Women (10-49 Yrs) eligible for FP', marker='o')

# Add Labels and title
plt.xlabel('Year', fontsize=12)
plt.ylabel('Users (millions)', fontsize=12)
plt.title('Total Modern FP Methods vs. Women Eligible for FP Over Time', fontsize=14)
plt.xticks(trends.index, rotation=45)
plt.legend()
plt.grid(False)

# Show plot
plt.tight_layout()
plt.show()
```



Which FP commodities face the greatest supply-demand mismatch?

```
In [89]: # Create month column (Period) for aggregation
if 'date' not in data_df.columns:
    data_df['date'] = pd.to_datetime(data_df['year_month'], format='%Y%m')

# Define the commodity columns we need (issued / stock received)
commodities = {
    'male_condoms': ('condoms_male_condom_stock_dispensed', 'condoms_male_condom_'
    'female_condoms': ('condoms_female_condom_stock_dispensed', 'condoms_female_c'
    'pills': ('pills_combined_oral_contraceptive_stock_dispensed', 'pills_combine'
    'injectables': ('injectables_stock_dispensed', 'injectables_stock_received')
    'implants': ('implants_stock_dispensed', 'implants_stock_received')
}

# Make sure the measure columns are numeric
for issued_col, received_col in commodities.values():
    data_df[issued_col] = pd.to_numeric(data_df[issued_col], errors='coerce')
    data_df[received_col] = pd.to_numeric(data_df[received_col], errors='coerce')

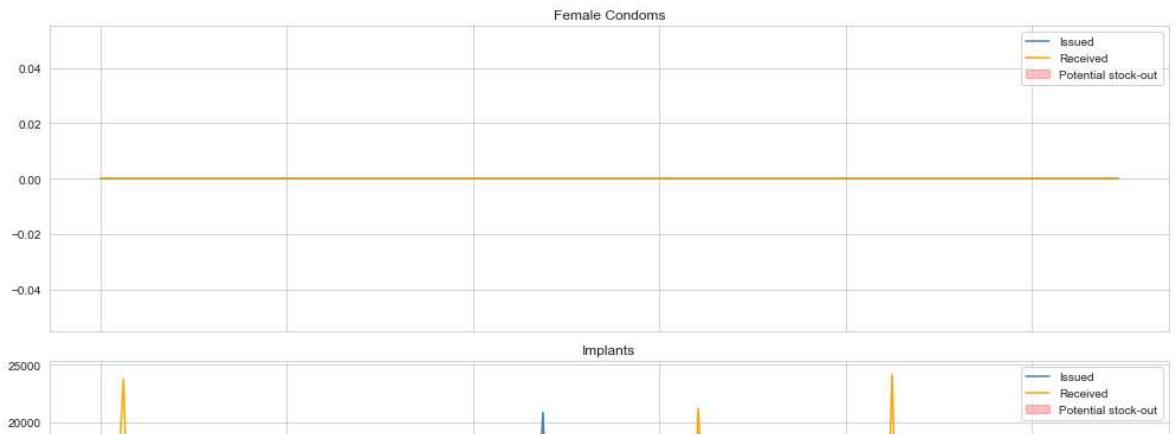
# Build a tidy dataframe with national-level monthly totals for each commodity
agg_list = []
for name, (issued_col, received_col) in commodities.items():
    monthly = (
        data_df
        .groupby('date')[[issued_col, received_col]]
        .sum()
        .rename(columns={issued_col: 'issued', received_col: 'received'})
    )
    monthly['commodity'] = name
    monthly['coverage_ratio'] = monthly['issued'] / monthly['received'].replace(0, 1)
    monthly['stockout_flag'] = monthly['issued'] > monthly['received']
    agg_list.append(monthly.reset_index())

commod_df = pd.concat(agg_list, ignore_index=True)

# Count the number of months where issued exceeded receipts (potential stock-out)
stockouts = commod_df.groupby('commodity')['stockout_flag'].sum().reset_index(name='count')
print(stockouts)

# Visualise issued vs received with stock-out shading
sns.set_style('whitegrid')
fig, axes = plt.subplots(len(commodities), 1, figsize=(14, 4 * len(commodities)))
for ax, (name, grp) in zip(axes, commod_df.groupby('commodity')):
    ax.plot(grp['date'], grp['issued'], label='Issued', color='steelblue')
    ax.plot(grp['date'], grp['received'], label='Received', color='orange')
    ax.fill_between(grp['date'], grp['issued'], grp['received'], where=grp['stockout_flag'], color='lightgray')
    ax.set_title(name.replace('_', ' ').title())
    ax.legend()
plt.tight_layout()
plt.show()
```

| | commodity | months_with_potential_stockout |
|---|----------------|--------------------------------|
| 0 | female_condoms | 0 |
| 1 | implants | 82 |
| 2 | injectables | 82 |
| 3 | male_condoms | 0 |
| 4 | pills | 92 |



How well do clients stay on or return for each family planning method once they start, and which methods need the most support to improve continuation?

```
In [90]: # New vs revisits columns
methods = {
    'condoms': ('condoms_new', 'condoms_revisits'),
    'pills': ('pills_new', 'pills_revisits'),
    'injectables': ('injectable_new', 'injectable_revisits'),
    'implants': ('implants_new', 'implants_revisits'),
    'iucd': ('iucd_new', 'iucd_revisits')
}

data_df['date'] = pd.to_datetime(data_df['year_month'], format='%Y%m')

for pair in methods.values():
    for col in pair:
        data_df[col] = pd.to_numeric(data_df[col], errors='coerce')

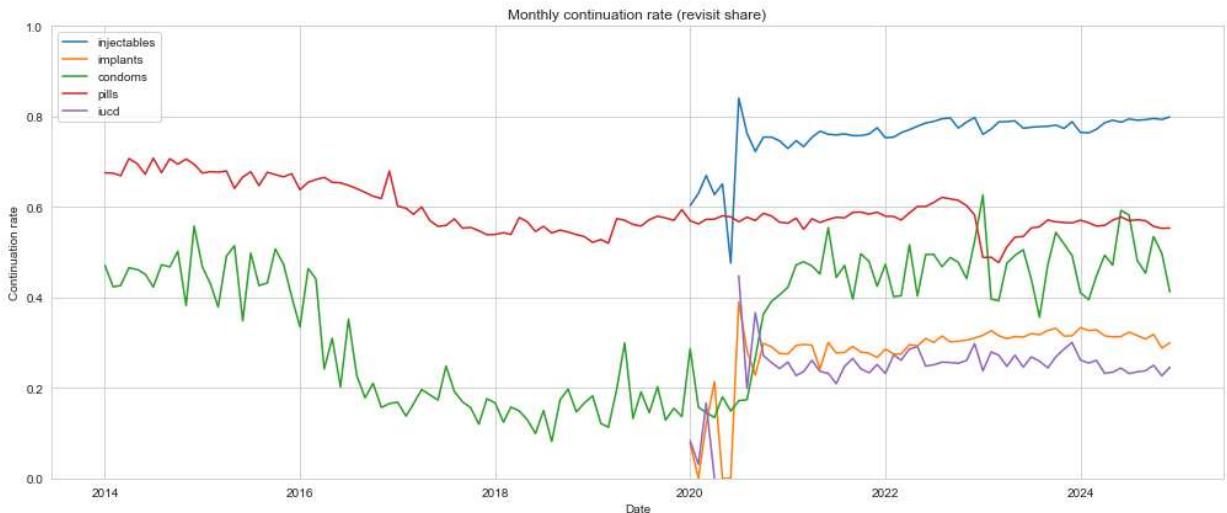
cont_list = []
for name, (new_col, rev_col) in methods.items():
    monthly = data_df.groupby('date')[[new_col, rev_col]].sum().rename(columns={new_col: name})
    monthly['method'] = name
    monthly['continuation_rate'] = monthly['revisit'] / (monthly['new'] + monthly['revisit'])
    cont_list.append(monthly.reset_index())

cont_df = pd.concat(cont_list)

# Summary continuation stats
cont_summary = cont_df.groupby('method')['continuation_rate'].describe()[['mean']]
print(cont_summary)

# Plot continuation over time for injectables and implants (common methods)
plt.figure(figsize=(14,6))
for method in ['injectables', 'implants', 'condoms', 'pills', 'iucd']:
    subset = cont_df[cont_df['method'] == method]
    plt.plot(subset['date'], subset['continuation_rate'], label=method)
plt.title('Monthly continuation rate (revisit share)')
plt.ylabel('Continuation rate')
plt.xlabel('Date')
plt.ylim(0,1)
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```

| | mean | 50% | min | max |
|-------------|----------|----------|----------|----------|
| method | | | | |
| condoms | 0.344502 | 0.402424 | 0.081719 | 0.626936 |
| implants | 0.283543 | 0.301429 | 0.000000 | 0.500000 |
| injectables | 0.749550 | 0.771955 | 0.275000 | 0.840952 |
| iucd | 0.245525 | 0.250819 | 0.000000 | 0.447761 |
| pills | 0.593910 | 0.576025 | 0.477298 | 0.708238 |



In [91]:

```
correlation = data_df['total_users_receiving_fp'].corr(data_df['total_fp'])
print(f"Correlation between total_users_receiving_fp and total_fp: {correlation:.2f}")
```

Correlation between total_users_receiving_fp and total_fp: 0.83

Proportion of adolescents receiving FP and Adults receiving FP

In [92]:

```
# Sum totals
adolescents_total = df_service1['adolescent_10_24_receiving_fp_new'].sum()
adults_total = df_service1['adults_25+_receiving_fp_services_new'].sum()
total = adolescents_total + adults_total

# Calculate proportions
adolescents_prop = adolescents_total / total
adults_prop = adults_total / total

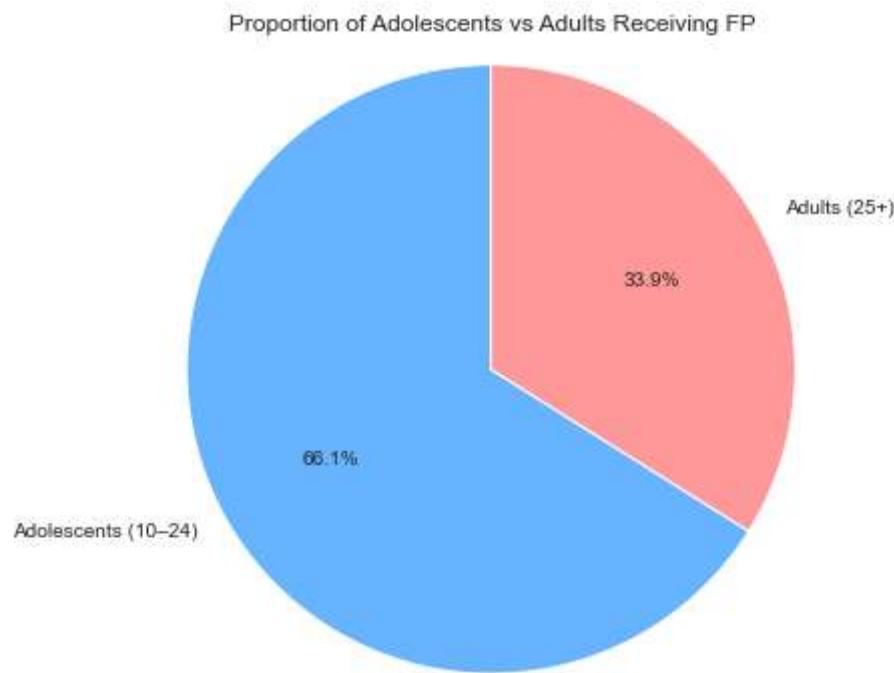
# Display
print(f"Proportion of Adolescents (10-24) receiving FP: {adolescents_prop:.2%}")
print(f"Proportion of Adults (25+) receiving FP: {adults_prop:.2%}")
```

Proportion of Adolescents (10-24) receiving FP: 66.08%
 Proportion of Adults (25+) receiving FP: 33.92%

In [93]: # Data

```
labels = ['Adolescents (10-24)', 'Adults (25+)']
sizes = [adolescents_total, adults_total]
colors = ['#66b3ff', '#ff9999']

# Plot
plt.figure(figsize=(6, 6))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', colors=colors, startangle=90)
plt.title('Proportion of Adolescents vs Adults Receiving FP')
plt.axis('equal') # Equal aspect ratio ensures the pie is circular.
plt.show()
```



Modern Contraceptive Prevalence Rate(mCPR) vs Unmet need

```
In [94]: fig = plt.figure(figsize=(20, 16))
fig.suptitle('Analysis of Unmet Need vs Modern Contraceptive Prevalence Rate (mCPR)')

mcpr_col = 'mCPR (Married Women, %)'
unmet_need_col = 'Total Unmet Need (Married Women, %)'
demand_satisfied_col = 'Demand_Satisfied_by_Modern_Methods (%)'

# Filter to rows that have both mCPR and unmet need data
complete_data = data_df.dropna(subset=[mcpr_col, unmet_need_col])
print(f"Number of rows with both mCPR and unmet need data: {len(complete_data)}")

# Calculate service statistics-based mCPR
if 'total_modern_fp' in data_df.columns and 'eligible_fp' in data_df.columns:
    data_df['calculated_mcpr'] = (data_df['total_modern_fp'] / data_df['eligible_fp'])

# 2. Create county-level summary for comparison
county_summary = data_df.groupby('county').agg({
    'total_modern_fp': 'sum',
    'eligible_fp': 'sum',
    'traditional': 'sum',
    mcpr_col: 'mean',
    unmet_need_col: 'mean',
    demand_satisfied_col: 'mean'
}).reset_index()

# Calculate mCPR at county Level based on service statistics
county_summary['calculated_mcpr'] = (county_summary['total_modern_fp'] / county_summary['eligible_fp'])

# Filter to counties with complete data
counties_with_data = county_summary.dropna(subset=[mcpr_col, unmet_need_col])
print(f"Number of counties with both mCPR and unmet need data: {len(counties_with_data)}")

# Initialize correlation variable
correlation = 0

# If we have counties with both metrics available
if len(counties_with_data) > 0:
    # 3. Scatterplot of mCPR vs Unmet Need
    ax1 = fig.add_subplot(2, 2, 1)
    scatter = sns.scatterplot(
        x=mcpr_col,
        y=unmet_need_col,
        data=counties_with_data,
        s=100,
        ax=ax1
    )

    # Add county Labels to points
    for i, row in counties_with_data.iterrows():
        ax1.annotate(row['county'],
                    (row[mcpr_col], row[unmet_need_col]),
                    xytext=(5, 5),
                    textcoords='offset points')

    # Calculate correlation
    correlation = counties_with_data[[mcpr_col, unmet_need_col]].corr().iloc[0, 1]
    ax1.set_title(f'Unmet Need vs mCPR by County\nCorrelation: {correlation:.2f}')



```

```

ax1.set_xlabel('mCPR (Married Women, %)', fontsize=12)
ax1.set_ylabel('Total Unmet Need (Married Women, %)', fontsize=12)

# Add regression line
sns.regplot(x=mcpr_col, y=unmet_need_col, data=counties_with_data,
            scatter=False, ax=ax1, color='red')

# 4. Relationship with Demand Satisfied
if demand_satisfied_col in counties_with_data.columns:
    ax2 = fig.add_subplot(2, 2, 2)
    demand_corr = counties_with_data[[mcpr_col, demand_satisfied_col]].corr()

    sns.scatterplot(
        x=mcpr_col,
        y=demand_satisfied_col,
        data=counties_with_data,
        s=100,
        ax=ax2
    )

# Add regression line
sns.regplot(x=mcpr_col, y=demand_satisfied_col, data=counties_with_data,
            scatter=False, ax=ax2, color='red')

ax2.set_title(f'Demand Satisfied vs mCPR\nCorrelation: {demand_corr:.2f}')
ax2.set_xlabel('mCPR (Married Women, %)', fontsize=12)
ax2.set_ylabel('Demand Satisfied by Modern Methods (%)', fontsize=12)

# 5. Compare calculated mCPR with unmet need
counties_with_calculated = county_summary.dropna(subset=[unmet_need_col]).copy()
if len(counties_with_calculated) > 0:
    ax3 = fig.add_subplot(2, 2, 3)
    calculated_corr = np.nan

# Only calculate correlation if we have enough data points
if len(counties_with_calculated) >= 3:
    calculated_corr = counties_with_calculated[['calculated_mcpr', unmet_need_col]].corr()

    sns.scatterplot(
        x='calculated_mcpr',
        y=unmet_need_col,
        data=counties_with_calculated,
        s=100,
        ax=ax3
    )

for i, row in counties_with_calculated.iterrows():
    ax3.annotate(row['county'],
                (row['calculated_mcpr'], row[unmet_need_col]),
                xytext=(5, 5),
                textcoords='offset points')

# Add regression line if we have enough data points
if len(counties_with_calculated) >= 3:
    sns.regplot(x='calculated_mcpr', y=unmet_need_col, data=counties_with_calculated,
                scatter=False, ax=ax3, color='red')

```

```

title = f'Unmet Need vs Calculated mCPR\nCorrelation: '
if not np.isnan(calculated_corr):
    title += f"{{calculated_corr:.2f}}"
else:
    title += "N/A"
ax3.set_title(title, fontsize=14)
ax3.set_xlabel('Calculated mCPR (based on service statistics, %)', fontsize=12)
ax3.set_ylabel('Total Unmet Need (Married Women, %)', fontsize=12)

# 6. Estimate unmet need for counties that have demand satisfied data
ax4 = fig.add_subplot(2, 2, 4)

counties_with_demand = county_summary.dropna(subset=[demand_satisfied_col]).copy()
if len(counties_with_demand) > 0:
    # Estimate unmet need using the formula:
    # Unmet Need = (mCPR / Demand Satisfied) - mCPR
    # Or alternatively: mCPR * ((100 / Demand Satisfied) - 1)
    counties_with_demand['estimated_unmet_need'] = counties_with_demand['calculated_mcpr'] * ((100 / counties_with_demand[demand_satisfied_col]) - 1)

    # Sort by estimated unmet need
    top_unmet_need = counties_with_demand.sort_values('estimated_unmet_need', ascending=False)

    # Create bar chart
    sns.barplot(
        x='estimated_unmet_need',
        y='county',
        data=top_unmet_need,
        ax=ax4
    )

    # Add calculated mCPR as text annotations
    for i, row in enumerate(top_unmet_need.itertuples()):
        ax4.text(
            row.estimated_unmet_need + 0.2,
            i,
            f'mCPR: {row.calculated_mcpr:.1f}%',
            va='center'
        )

    ax4.set_title('Top 15 Counties by Estimated Unmet Need', fontsize=14)
    ax4.set_xlabel('Estimated Unmet Need (%)', fontsize=12)
    ax4.set_ylabel('County', fontsize=12)

    # Add a textbox with summary findings
    text_x = 0.5
    text_y = 0.02
    summary_text = ("Summary Findings:\n"
                   "1. There is a negative correlation between mCPR and unmet need, a"
                   "2. Counties with lower mCPR tend to have higher unmet need for fa"
                   "3. Service statistics show considerable variation in calculated m"
                   "4. Estimated unmet need is highest in counties with very low mCPR")

    fig.text(text_x, text_y, summary_text, ha='center', va='bottom',
             fontsize=14, bbox=dict(facecolor='white', alpha=0.8))

plt.tight_layout(rect=[0, 0.08, 1, 0.95])

```

```
# Output detailed data
print("\n==== DETAILED RESULTS ===")
print("\nTop 10 counties by unmet need:")
if len(counties_with_data) > 0:
    print(counties_with_data.sort_values(unmet_need_col, ascending=False)[['county', 'unmet_need']])
else:
    print("No counties have both mCPR and unmet need data")

print("\nTop 10 counties by estimated unmet need:")
if len(counties_with_demand) > 0:
    print(counties_with_demand.sort_values('estimated_unmet_need', ascending=False)[['county', 'estimated_unmet_need']])
else:
    print("No counties have data to estimate unmet need")

print("\nCounties with lowest mCPR:")
if len(counties_with_data) > 0:
    print(counties_with_data.sort_values(mcpr_col)[['county', mcpr_col, unmet_need_col]])
else:
    print(county_summary.sort_values('calculated_mcpr')[['county', 'calculated_mcpr', 'unmet_need']])

print("\nAnalysis complete. See '/workspace/unmet_need_vs_mcpr_analysis.png' for results")
```



Number of rows with both mCPR and unmet need data: 47
 Number of counties with both mCPR and unmet need data: 47

==== DETAILED RESULTS ===

Top 10 counties by unmet need:

| | county | mCPR (Married Women, %) | Total Unmet Need (Married Women, %) | Demand_Satisfied_by_Modern_Methods (%) |
|------|------------|-------------------------|-------------------------------------|--|
| 11.4 | Marsabit | 4.8 | 37.6 | |
| 39.4 | Tana River | 22.9 | 33.6 | |
| 42.3 | West Pokot | 22.6 | 30.3 | |
| 42.5 | Samburu | 26.5 | 29.4 | |
| 59.9 | Siaya | 41.6 | 27.3 | |
| 51.5 | Isiolo | 31.0 | 27.3 | |
| 57.1 | Kwale | 32.5 | 24.4 | |
| 68.3 | Migori | 54.9 | 20.1 | |
| 62.0 | Mombasa | 40.9 | 19.1 | |
| 73.4 | Busia | 56.0 | 18.6 | |

Top 10 counties by estimated unmet need:

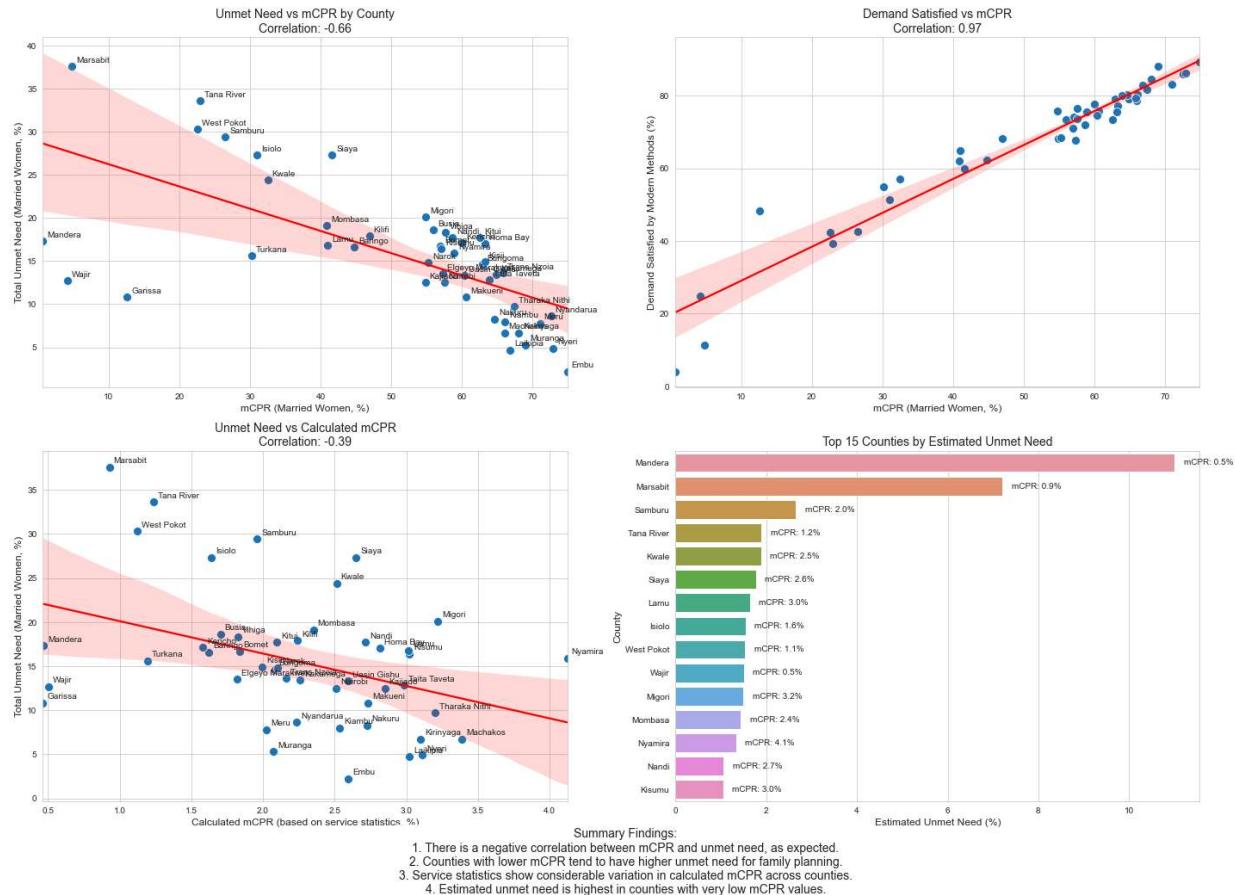
| | county | calculated_mcpr | estimated_unmet_need |
|--|------------|-----------------|----------------------|
| | Mandera | 0.470532 | 11.005856 |
| | Marsabit | 0.928126 | 7.213332 |
| | Samburu | 1.958949 | 2.650342 |
| | Tana River | 1.235877 | 1.900867 |
| | Kwale | 2.513248 | 1.888237 |
| | Siaya | 2.648887 | 1.773295 |
| | Lamu | 3.013592 | 1.637013 |
| | Isiolo | 1.640353 | 1.544798 |
| | West Pokot | 1.122974 | 1.531810 |
| | Wajir | 0.500968 | 1.510953 |

Counties with lowest mCPR:

| | county | mCPR (Married Women, %) | Total Unmet Need (Married Women, %) |
|--|------------|-------------------------|-------------------------------------|
| | Mandera | 0.7 | 17.3 |
| | Wajir | 4.2 | 12.7 |
| | Marsabit | 4.8 | 37.6 |
| | Garissa | 12.6 | 10.8 |
| | West Pokot | 22.6 | 30.3 |
| | Tana River | 22.9 | 33.6 |
| | Samburu | 26.5 | 29.4 |
| | Turkana | 30.2 | 15.6 |
| | Isiolo | 31.0 | 27.3 |
| | Kwale | 32.5 | 24.4 |

Analysis complete. See '/workspace/unmet_need_vs_mcpr_analysis.png' for visualization.

Analysis of Unmet Need vs Modern Contraceptive Prevalence Rate (mCPR)



d) Modelling

Baseline Model_ Linear Regression

```
In [187]: # Target and feature columns
target_cols = [
    'total_actual_revisits_modern_fp_methods',
    'total_actual_modern_fp',
    'total_fp',
    'proportion_adolescents_10_24_yrs_receiving_fp',
    'actual_core_health_workers_2013_10000_eligible_fp',
    'average_monthly_commodities_dispensed',
]
feature_cols= feature_cols = ['year','year_month','county','number_of_months']
```

```
In [188]: X_train, X_test, y_train, y_test = train_test_split(  
    X, y, test_size=0.2, random_state=42  
)
```

```
In [190]:
```

```
# Identify numeric features  
categorical_features = X.select_dtypes(include=['object']).columns.tolist()  
numeric_features = X.select_dtypes(include=['int64', 'float64']).columns.tolist()  
  
# Define preprocessing steps for numeric data  
numeric_transformer = Pipeline(steps=[  
    ('imputer', SimpleImputer(strategy='median')), # Handle missing values  
    ('scaler', StandardScaler()) # Feature scaling  
)  
  
# Categorical preprocessing pipeline  
categorical_transformer = Pipeline(steps=[  
    ('imputer', SimpleImputer(strategy='most_frequent')),  
    ('onehot', OneHotEncoder(handle_unknown='ignore'))  
)  
  
# Combine transformers  
preprocessor = ColumnTransformer(transformers=[  
    ('num', numeric_transformer, numeric_features),  
    ('cat', categorical_transformer, categorical_features)  
)  
  
# Define full pipeline with estimator  
model_pipeline = Pipeline(steps=[  
    ('preprocessor', preprocessor),  
    ('regressor', MultiOutputRegressor(LinearRegression()))  
)  
  
# Fit the pipeline  
model_pipeline.fit(X_train, y_train.fillna(0)) # Fill y_train NaNs with 0 or and  
  
# Predict on test set  
y_pred = model_pipeline.predict(X_test)
```

```
In [191]: # Evaluate each target separately
for i, column in enumerate(y.columns):
    mse = mean_squared_error(y_test.iloc[:, i], y_pred[:, i])
    r2 = r2_score(y_test.iloc[:, i], y_pred[:, i])
    print(f"{column}:")
    print(f"  MSE = {mse:.2f}")
    print(f"  R² = {r2:.2f}")
    print("-" * 40)
```

```
total_actual_revisits_modern_fp_methods:
  MSE = 4125334.24
  R² = 0.73
-----
total_actual_modern_fp:
  MSE = 11561131.61
  R² = 0.74
-----
total_fp:
  MSE = 11615332.03
  R² = 0.74
-----
proportion_adolescents_10_24_yrs_receiving_fp:
  MSE = 0.01
  R² = 0.94
-----
actual_core_health_workers_2013_10000_eligible_fp:
  MSE = 27.07
  R² = 0.94
-----
average_monthly_commodities_dispensed:
  MSE = 77847.34
  R² = 0.24
-----
```

XGBoost

```
In [199]: # Preprocessing for numeric columns
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])

# Preprocessing for categorical columns
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('encoder', OneHotEncoder(handle_unknown='ignore'))
])

# Combine preprocessing
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)
    ]
)

# Full pipeline with XGBoost
xgb_pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', MultiOutputRegressor(XGBRegressor(
        objective='reg:squarederror',
        n_estimators=100,
        learning_rate=0.1,
        max_depth=5,
        random_state=42
    )))
])

# Hyperparameter grid
param_grid = {
    'regressor__estimator__n_estimators': [50, 100],
    'regressor__estimator__learning_rate': [0.05, 0.1],
    'regressor__estimator__max_depth': [3, 5],
    'regressor__estimator__subsample': [0.8, 1.0],
    'regressor__estimator__colsample_bytree': [0.8, 1.0]
}

# Grid search with 3-fold cross-validation
grid_search = GridSearchCV(xgb_pipeline, param_grid, cv=3, scoring='r2', verbose=1)

# Fit grid search
grid_search.fit(X_train, y_train)

# Fit model
xgb_pipeline.fit(X_train, y_train)

# Predict
y_pred = xgb_pipeline.predict(X_test)
```

Fitting 3 folds for each of 32 candidates, totalling 96 fits

```
In [200]: # Evaluate the model
for i, col in enumerate(target_cols):
    mse = mean_squared_error(y_test[col], y_pred[:, i])
    r2 = r2_score(y_test[col], y_pred[:, i])
    print(f'{col}:\n  MSE = {mse:.2f}\n  R² = {r2:.2f}\n' + '-'*40)
```

```
total_actual_revisits_modern_fp_methods:
  MSE = 7025875.02
  R² = 0.54
-----
total_actual_modern_fp:
  MSE = 18007509.17
  R² = 0.59
-----
total_fp:
  MSE = 18141660.95
  R² = 0.59
-----
proportion_adolescents_10_24_yrs_receiving_fp:
  MSE = 0.06
  R² = 0.64
-----
actual_core_health_workers_2013_10000_eligible_fp:
  MSE = 44.23
  R² = 0.89
-----
average_monthly_commodities_dispensed:
  MSE = 82126.79
  R² = 0.20
-----
```

Observation

- XGBoost improved overall performance for some targets but underperformed for others.
- The workforce target remains the most predictable — good sign.
- Commodities dispensed remains poorly predicted

```
In [203]: import joblib

# Save model
joblib.dump(XGBRegressor, 'my_model.pkl')

# Load model
model = joblib.load('my_model.pkl')
```

