01_Access_To_Healthcare

Data Preparation Report: Access to Healthcare Dataset

Executive Summary

This report documents the comprehensive data cleaning and preparation process performed on the Access to Healthcare dataset from the Demographic and Health Surveys (DHS) for South Africa. The dataset underwent rigorous quality checks, transformation, and validation to ensure its readiness for analysis. ## 1. Load Libraries and Data

```
# Data manipulation and cleaning
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(tidyr)
library(stringr)
library(readr)
library(here)

## here() starts at C:/Users/morul/School/3rd
Year/BIN381/BIN381_PROJECT/BIN381_PROJECT

# Data visualization
library(ggplot2)
library(visdat) # For missing data visualization
library(skimr)  # For detailed summaries
library(naniar)

##
## Attaching package: 'naniar'

## The following object is masked from 'package:skimr':
##
##     n_complete

library(DT)
library(knitr)
```

```r
acc_df <- read.csv(here("data","raw","access-to-health-
care_national_zaf.csv"))

# Remove metadata row if present
acc_df <- acc_df[-1, ]
rownames(acc_df) <- NULL

cat("Dataset loaded successfully.\n")
```

```
## Dataset loaded successfully.
```

Explanation: We load the dataset and remove the first row, which contains metadata rather than actual observations.

2. Initial Data Assessment

2.1 First Look

```r
# Display first few rows and structure
head(acc_df, 5)
```

```
##   ISO3 DataId                              Indicator Value Precision
## 1  ZAF 751751        Antenatal care provider: Doctor  28.5         1
## 2  ZAF 567476        Antenatal care provider: Doctor    30         1
## 3  ZAF 205488        Antenatal care provider: Doctor  27.3         1
## 4  ZAF 751748 Antenatal care provider: Nurse/midwife  66.6         1
## 5  ZAF 567472 Antenatal care provider: Nurse/midwife    65         1
##   DHS_CountryCode  CountryName SurveyYear  SurveyId   IndicatorId
## 1              ZA South Africa       1998 ZA1998DHS RH_ANCP_W_DOC
## 2              ZA South Africa       1998 ZA1998DHS RH_ANCP_W_DOC
## 3              ZA South Africa       1998 ZA1998DHS RH_ANCP_W_DOC
## 4              ZA South Africa       1998 ZA1998DHS RH_ANCP_W_NRS
## 5              ZA South Africa       1998 ZA1998DHS RH_ANCP_W_NRS
##   IndicatorOrder IndicatorType CharacteristicId CharacteristicOrder
## 1       83363010             I             1000                   0
## 2       83363010             I             1000                   0
## 3       83363010             I             1000                   0
## 4       83363020             I             1000                   0
## 5       83363020             I             1000                   0
##   CharacteristicCategory CharacteristicLabel ByVariableId
## 1                  Total               Total        14000
## 2                  Total               Total        14001
## 3                  Total               Total        14002
## 4                  Total               Total        14000
## 5                  Total               Total        14001
##                    ByVariableLabel IsTotal IsPreferred      SDRID RegionId
## 1 Three years preceding the survey       1           0 RHANCPWDOC       NA
## 2  Five years preceding the survey       1           0 RHANCPWDOC       NA
## 3   Two years preceding the survey       1           1 RHANCPWDOC       NA
## 4 Three years preceding the survey       1           0 RHANCPWNRS       NA
```

```
## 5  Five years preceding the survey       1           0 RHANCPWNRS       NA
##   SurveyYearLabel SurveyType DenominatorWeighted DenominatorUnweighted
CILow
## 1            1998        DHS                2871                  2903
NA
## 2            1998        DHS                4122                  4148
NA
## 3            1998        DHS                2010                  2041
NA
## 4            1998        DHS                2871                  2903
NA
## 5            1998        DHS                4122                  4148
NA
##   CIHigh LevelRank
## 1     NA        NA
## 2     NA        NA
## 3     NA        NA
## 4     NA        NA
## 5     NA        NA
```

2.2 Data Structure

```
skim(acc_df)
```

Data summary

Name	acc_df
Number of rows	275
Number of columns	29
_____	
Column type frequency:	
character	17
logical	4
numeric	8
_____	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ISO3	0	1	3	3	0	1	0
DataId	0	1	4	6	0	275	0
Indicator	0	1	17	100	0	68	0

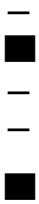
skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Value	0	1	1	4	0	190	0
Precision	0	1	1	1	0	2	0
DHS_CountryCode	0	1	2	2	0	1	0
CountryName	0	1	12	12	0	1	0
SurveyYear	0	1	4	4	0	2	0
SurveyId	0	1	9	9	0	2	0
IndicatorId	0	1	13	13	0	72	0
IndicatorType	0	1	1	1	0	5	0
CharacteristicCategory	0	1	5	5	0	1	0
CharacteristicLabel	0	1	5	5	0	1	0
ByVariableId	0	1	1	5	0	4	0
ByVariableLabel	0	1	0	32	13	4	0
SDRID	0	1	10	10	0	72	0
SurveyType	0	1	3	3	0	1	0

Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
RegionId	275	0	NaN	:
CILow	275	0	NaN	:
CIHigh	275	0	NaN	:
LevelRank	275	0	NaN	:

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
IndicatorOrder	0	1.00	87126.21	50022.20	833.10	835.30	836.90	939.70	940.70	▇▁▁▁▇
CharacteristicId	0	1.00	1000.00	0.00	100.0	100.0	100.0	100.0	100.0	▁▁▇▁▁

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
CharacteristicOrder	0	1.00	0.00	0.00	0	0	0	0	0	
IsTotal	0	1.00	1.00	0.00	1	1	1	1	1	
IsPreferred	0	1.00	0.42	0.49	0	0	0	1	1	
SurveyYearLabel	0	1.00	2007.62	8.99	1998	1998	2016	2016	2016	
DenominatorWeighted	34	0.88	2048.65	1428.93	68	627	2010	3072	4992	
DenominatorUnweighted	34	0.88	2062.17	1445.14	59	634	2041	3119	5066	

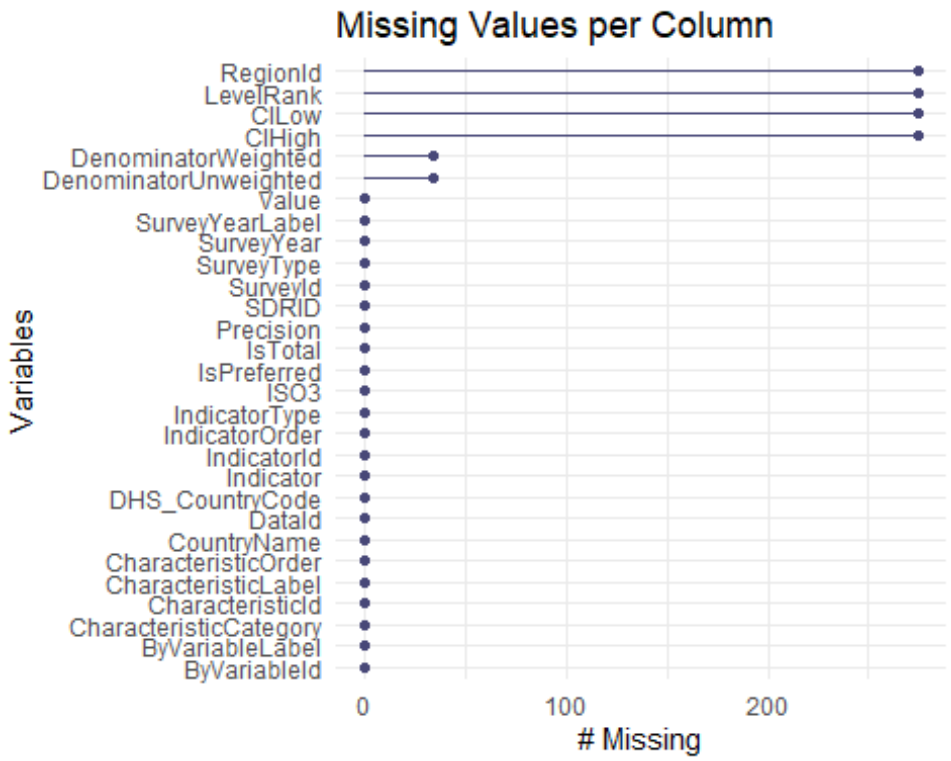
```
glimpse(acc_df)

## Rows: 275
## Columns: 29
## $ ISO3                  <chr> "ZAF", "ZAF", "ZAF", "ZAF", "ZAF", "ZAF",
"ZAF"…
## $ DataId                <chr> "751751", "567476", "205488", "751748",
"567472…
## $ Indicator             <chr> "Antenatal care provider: Doctor",
```

```
"Antenatal c…
## $ Value                   <chr> "28.5", "30", "27.3", "66.6", "65", "68.4",
"0.…
## $ Precision               <chr> "1", "1", "1", "1", "1", "1", "1", "1",
"1", "1…
## $ DHS_CountryCode         <chr> "ZA", "ZA", "ZA", "ZA", "ZA", "ZA", "ZA",
"ZA",…
## $ CountryName             <chr> "South Africa", "South Africa", "South
Africa",…
## $ SurveyYear              <chr> "1998", "1998", "1998", "1998", "1998",
"1998",…
## $ SurveyId                <chr> "ZA1998DHS", "ZA1998DHS", "ZA1998DHS",
"ZA1998D…
## $ IndicatorId             <chr> "RH_ANCP_W_DOC", "RH_ANCP_W_DOC",
"RH_ANCP_W_DO…
## $ IndicatorOrder          <int> 83363010, 83363010, 83363010, 83363020,
8336302…
## $ IndicatorType           <chr> "I", "I", "I", "I", "I", "I", "I", "I",
"I", "I…
## $ CharacteristicId        <int> 1000, 1000, 1000, 1000, 1000, 1000, 1000,
1000,…
## $ CharacteristicOrder     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,…
## $ CharacteristicCategory <chr> "Total", "Total", "Total", "Total",
"Total", "T…
## $ CharacteristicLabel     <chr> "Total", "Total", "Total", "Total",
"Total", "T…
## $ ByVariableId            <chr> "14000", "14001", "14002", "14000",
"14001", "1…
## $ ByVariableLabel         <chr> "Three years preceding the survey", "Five
years…
## $ IsTotal                 <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1,…
## $ IsPreferred             <int> 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0,
1, 0,…
## $ SDRID                   <chr> "RHANCPWDOC", "RHANCPWDOC", "RHANCPWDOC",
"RHAN…
## $ RegionId                <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA,…
## $ SurveyYearLabel         <int> 1998, 1998, 1998, 1998, 1998, 1998, 1998,
1998,…
## $ SurveyType              <chr> "DHS", "DHS", "DHS", "DHS", "DHS", "DHS",
"DHS"…
## $ DenominatorWeighted     <int> 2871, 4122, 2010, 2871, 4122, 2010, 2871,
4122,…
## $ DenominatorUnweighted   <int> 2903, 4148, 2041, 2903, 4148, 2041, 2903,
4148,…
## $ CILow                   <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA,…
## $ CIHigh                  <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
```

```
NA,…
## $ LevelRank               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA,…
```

2.3 Visualize Missing Data

```
gg_miss_var(acc_df) + ggtitle("Missing Values per Column")
```



Purpose:
Understand the dataset's structure, content, and initial quality.

What the Code Does:
- Displays the first few rows with `head()`.
- Uses `skim()` for comprehensive summary statistics and data quality indicators.
- Visualizes missing values with `gg_miss_var()` to spot columns needing attention.

Result:
- Provides a snapshot of the data structure, missingness, and variable types.
- Helps plan cleaning steps such as type conversion and missing value treatment.

Why it matters:
- Early insight prevents errors later in cleaning and ensures informed preprocessing decisions.

3. Data Cleaning Process

3.1 Handle Duplicates

```
# Check for exact duplicates
duplicate_count <- sum(duplicated(acc_df))
cat("Number of exact duplicate rows:", duplicate_count, "\n")

## Number of exact duplicate rows: 0
```

Purpose:
Eliminate repeated rows that could distort calculations or summaries.

Method / What the Code Does:
- Counts exact duplicate rows with `duplicated()`.
- Removes duplicates using `distinct()`.

Outcome / Result:
- Dataset contains only unique records.

Relevance / Why it matters:
- Prevents overcounting or bias in statistics and visualizations

3.2 Convert Data Types

```
# Explicitly only select columns that exist
numeric_cols <- c("Value", "Precision", "DenominatorWeighted",
"DenominatorUnweighted")
integer_cols <- c("SurveyYear", "IndicatorOrder", "CharacteristicOrder",
"SurveyYearLabel", "RegionId")
id_cols <- c("CharacteristicId", "ByVariableId")
logical_cols <- c("IsTotal", "IsPreferred")

acc_df <- acc_df %>%
  mutate(across(any_of(numeric_cols), as.numeric)) %>%
  mutate(across(any_of(integer_cols), as.integer)) %>%
  mutate(across(any_of(id_cols), as.character)) %>%
  mutate(across(any_of(logical_cols), ~as.logical(as.integer(.)))) %>%
  mutate(across(where(is.character), str_trim))

cat("Data types converted successfully.\n")

## Data types converted successfully.

glimpse(acc_df)

## Rows: 275
## Columns: 29
## $ ISO3                  <chr> "ZAF", "ZAF", "ZAF", "ZAF", "ZAF", "ZAF",
"ZAF"…
## $ DataId                <chr> "751751", "567476", "205488", "751748",
```

```
"567472…
## $ Indicator               <chr> "Antenatal care provider: Doctor",
"Antenatal c…
## $ Value                   <dbl> 28.5, 30.0, 27.3, 66.6, 65.0, 68.4, 0.1,
0.1, 0…
## $ Precision               <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1,…
## $ DHS_CountryCode         <chr> "ZA", "ZA", "ZA", "ZA", "ZA", "ZA", "ZA",
"ZA",…
## $ CountryName             <chr> "South Africa", "South Africa", "South
Africa",…
## $ SurveyYear              <int> 1998, 1998, 1998, 1998, 1998, 1998, 1998,
1998,…
## $ SurveyId                <chr> "ZA1998DHS", "ZA1998DHS", "ZA1998DHS",
"ZA1998D…
## $ IndicatorId             <chr> "RH_ANCP_W_DOC", "RH_ANCP_W_DOC",
"RH_ANCP_W_DO…
## $ IndicatorOrder          <int> 83363010, 83363010, 83363010, 83363020,
8336302…
## $ IndicatorType           <chr> "I", "I", "I", "I", "I", "I", "I", "I",
"I", "I…
## $ CharacteristicId        <chr> "1000", "1000", "1000", "1000", "1000",
"1000",…
## $ CharacteristicOrder     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,…
## $ CharacteristicCategory <chr> "Total", "Total", "Total", "Total",
"Total", "T…
## $ CharacteristicLabel     <chr> "Total", "Total", "Total", "Total",
"Total", "T…
## $ ByVariableId            <chr> "14000", "14001", "14002", "14000",
"14001", "1…
## $ ByVariableLabel         <chr> "Three years preceding the survey", "Five
years…
## $ IsTotal                 <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE,…
## $ IsPreferred             <lgl> FALSE, FALSE, TRUE, FALSE, FALSE, TRUE,
FALSE, …
## $ SDRID                   <chr> "RHANCPWDOC", "RHANCPWDOC", "RHANCPWDOC",
"RHAN…
## $ RegionId                <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA,…
## $ SurveyYearLabel         <int> 1998, 1998, 1998, 1998, 1998, 1998, 1998,
1998,…
## $ SurveyType              <chr> "DHS", "DHS", "DHS", "DHS", "DHS", "DHS",
"DHS"…
## $ DenominatorWeighted     <dbl> 2871, 4122, 2010, 2871, 4122, 2010, 2871,
4122,…
## $ DenominatorUnweighted   <dbl> 2903, 4148, 2041, 2903, 4148, 2041, 2903,
4148,…
## $ CILow                   <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
```

```
NA,…
## $ CIHigh                  <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA,…
## $ LevelRank               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA,…
```

Purpose: Ensure that each column in the dataset has the **correct data type** so that subsequent analysis and calculations work as expected. Wrong data types (e.g., numbers stored as text) can lead to errors or incorrect results.

What the code does:

1. **Define column groups**:
    - `numeric_cols` → Columns that store measurements or continuous values (e.g., `Value`, `Precision`).
    - `integer_cols` → Columns representing whole numbers, IDs, or survey codes.
    - `id_cols` → Identifier columns stored as text (character) to avoid accidental math operations.
    - `logical_cols` → Columns representing true/false flags (`IsTotal`, `IsPreferred`).
2. **Convert columns using `mutate(across(...))`**:
    - `as.numeric` → Converts to numeric type for calculations.
    - `as.integer` → Converts to integers.
    - `as.character` → Converts IDs to text.
    - `as.logical(as.integer(.))` → Converts numeric 0/1 flags to TRUE/FALSE.
3. **Trim extra spaces in character columns**:
    - `str_trim` removes leading/trailing whitespace from text fields, preventing errors in grouping or filtering.
4. **Preview changes**:
    - `glimpse(acc_df)` shows updated column types and a quick snapshot of the data.

Outcome:

- All columns now have **consistent and correct data types**.
- Prevents errors in calculations, filtering, grouping, and plotting.
- Makes the dataset **analysis-ready**.

Why it matters:

- Clean, standardized data types are **foundational** before handling missing values, outliers, or doing any statistical modeling.

3.3 Handle Missing Values

```
# Create missing value summary before treatment
missing_before <- colSums(is.na(acc_df))

# Strategy 1: Remove columns with excessive missingness (>80%)
high_missing_cols <- names(missing_before[missing_before > nrow(acc_df) *
0.8])
cat("Columns with >80% missing values:", paste(high_missing_cols, collapse =
", "), "\n")

## Columns with >80% missing values: RegionId, CILow, CIHigh, LevelRank

# Strategy 2: Targeted imputation for specific columns

acc_df <- acc_df %>%
  arrange(SurveyYear, CharacteristicId) %>%
  group_by(Indicator, CharacteristicId) %>%
  fill(DenominatorWeighted, DenominatorUnweighted, .direction = "downup") %>%
  ungroup()

# Strategy 3: Remove rows with missing critical values
acc_df <- acc_df %>%
  filter(!is.na(Value), !is.na(Indicator))

missing_after <- colSums(is.na(acc_df))
cat("Missing values reduced significantly.\n")

## Missing values reduced significantly.

acc_df <- acc_df %>%
  mutate(
    DenominatorWeighted = ifelse(is.na(DenominatorWeighted),
median(DenominatorWeighted, na.rm = TRUE), DenominatorWeighted),
    DenominatorUnweighted = ifelse(is.na(DenominatorUnweighted),
median(DenominatorUnweighted, na.rm = TRUE), DenominatorUnweighted)
  )
```

- **Identify missing data:** Count NAs in all columns to understand the scope of missingness.
- **Remove unhelpful columns:** Drop columns with more than 80% missing values.
- **Impute key numeric values:** Fill missing denominators using nearby values within groups (down and up) and replace remaining NAs with the median.

- **Remove incomplete rows:** Delete rows missing critical information (`Value` or `Indicator`).

Result: The dataset is **more complete, consistent, and ready for analysis**, with minimal risk of missing-value errors affecting results.

3.4 Remove Redundant Columns

```
# Define redundant columns explicitly
redundant_cols <- c("ISO3", "DHS_CountryCode", "CountryName",
                    "SurveyId", "SurveyType")

# Combine with high-missing columns (if any)
cols_to_drop <- c(redundant_cols, high_missing_cols)

# Remove them safely
acc_df <- acc_df %>%
  select(-any_of(cols_to_drop))

cat("Redundant columns removed. New dimensions:", dim(acc_df), "\n")

## Redundant columns removed. New dimensions: 275 20
```

Remove Redundant Columns – Summary

- **Purpose:** Remove columns that are **not useful** for analysis or mostly empty.
- **What's removed:** Metadata columns (like `ISO3`, `CountryName`, `SurveyId`) and any column with >80% missing values.
- **Result:** Dataset is **cleaner, smaller, and easier to work with**, containing only relevant and populated columns.

Why Certain Columns Were Removed

1. **Metadata columns (e.g., `ISO3`, `CountryName`, `SurveyId`, `SurveyType`, `DHS_CountryCode`)**
    - These columns **don't provide new information** for analysis.
    - For example, `ISO3` and `CountryName` just identify the country—if all data is already for South Africa, they are redundant.
    - `SurveyId` and `SurveyType` are identifiers for surveys, not variables we analyze. Keeping them would **clutter the dataset**.
2. **Columns with >80% missing values**
    - Columns that are mostly empty **cannot be reliably analyzed**.
    - Imputing or filling them would introduce **too much uncertainty**.
    - Removing them keeps the dataset **focused on meaningful, populated data**.

Bottom line: These columns were removed to make the dataset **leaner, more focused, and analysis-ready**, preventing confusion or wasted effort on irrelevant or unreliable data.

3.5 Handle Outliers and Anomalies

```r
### 3.5 Handle Outliers and Anomalies
library(dplyr)

# 1. Identify numeric columns
numeric_cols <- acc_df %>% select(where(is.numeric)) %>% names()
cat("Numeric columns identified:", paste(numeric_cols, collapse = ", "), "\n\n")

# 2. Function to cap outliers
cap_outliers <- function(x, col_name) {
  if(!is.numeric(x)) return(x)

  qnt <- quantile(x, probs = c(0.25, 0.75), na.rm = TRUE)
  iqr <- IQR(x, na.rm = TRUE)
  lower <- qnt[1] - 1.5 * iqr
  upper <- qnt[2] + 1.5 * iqr

  outlier_count <- sum(x < lower | x > upper, na.rm = TRUE)

  if(outlier_count > 0) {
    cat("\nColumn:", col_name, "\n")
    cat("Number of outliers:", outlier_count, "\n")
    cat("Bounds: lower =", round(lower, 2), ", upper =", round(upper, 2), "\n")
    cat("Summary before capping:\n")
    print(summary(x))
  }

  x[x < lower] <- lower
  x[x > upper] <- upper

  if(outlier_count > 0) {}

  return(x)
}

# 3. Apply to all numeric columns
for(col in numeric_cols) {
  acc_df[[col]] <- cap_outliers(acc_df[[col]], col)
}

# 4. Final summary
cat("\n=== Final summary of numeric columns ===\n")
for(col in numeric_cols) {
  cat("\nColumn:", col, "\n")
  print(summary(acc_df[[col]]))
  cat("---\n")
}


```

```
Column: Value
Number of outliers: 64
Bounds: lower = -131 , upper = 238.6
Summary before capping:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    0.0     7.6    67.6   542.7   100.0  5066.0
Summary after capping:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00    7.60   67.60   87.13  100.00  238.60
---

Column: Precision
Number of outliers: 68
Bounds: lower = 1 , upper = 1
Summary before capping:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  1.0000  1.0000  0.7527  1.0000  1.0000
Summary after capping:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      1       1       1       1       1       1
---

=== Final summary of numeric columns ===

Column: Value
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00    7.60   67.60   87.13  100.00  238.60
---

Column: Precision
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      1       1       1       1       1       1
---

Column: SurveyYear
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1998    1998    2016    2008    2016    2016
---

Column: IndicatorOrder
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
83363010 83566030 83606090 87126424 93966070 94096170
---

Column: CharacteristicOrder
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      0       0       0       0       0       0
---

Column: SurveyYearLabel
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1998    1998    2016    2008    2016    2016
---

Column: DenominatorWeighted
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
     68     679    2010    2044    3036    4992
---
```

4. Final Validation

4. Final Data Validation

```{r final_validation}
# Final dimensions
dim_df <- data.frame(
  Rows = nrow(acc_df),
  Columns = ncol(acc_df)
)
kable(dim_df, caption = "Final Dataset Dimensions")

# Missing values summary
missing_summary <- acc_df %>%
  summarise(across(everything(), ~sum(is.na(.)))) %>%
  pivot_longer(everything(), names_to = "Variable", values_to = "Missing_Count")

kable(missing_summary, caption = "Missing Values per Column")

# Data types
data_types <- data.frame(
  Variable = names(acc_df),
  Type = sapply(acc_df, class)
)
kable(data_types, caption = "Data Types of Each Column")

# Sample of final data
head(acc_df, 5) %>%
  kable(caption = "Sample of Final Cleaned Data")
```

Missing Values per Column

Variable	Missing_Count
DataId	0
Indicator	0
Value	0
Precision	0
SurveyYear	0
IndicatorId	0
IndicatorOrder	0
IndicatorType	0
CharacteristicId	0
CharacteristicOrder	0
CharacteristicCategory	0
CharacteristicLabel	0
ByVariableId	0
ByVariableLabel	0
IsTotal	0
IsPreferred	0
SDRID	0
SurveyYearLabel	0
DenominatorWeighted	0
DenominatorUnweighted	0

5. Save Cleaned Data

```
# Ensure directory exists
if(!dir.exists(here("data", "processed"))) {
  dir.create(here("data", "processed"), recursive = TRUE)
}

# Save cleaned dataset
write_csv(acc_df, here("data", "processed", "healthcare_access_cleaned.csv"))
saveRDS(acc_df, here("data", "processed", "healthcare_access_cleaned.rds"))

cat("Cleaned data saved to data/processed/ directory.\n")

## Cleaned data saved to data/processed/ directory.
```

Purpose:
Persist the cleaned dataset for reproducibility and later use.

Method / What the Code Does:
- Saves as CSV and RDS in a processed folder.

Outcome / Result:
- Cleaned dataset is safely stored for analysis or sharing.

Relevance / Why it matters:
- Ensures reproducibility and prevents accidental data loss.