Performed a comprehensive data validation process across four key datasets: Books, Users, Ratings, and Reviews. The following steps were included:

- **Null Value Checks**: Ensured there were no missing values post-cleaning.
- **Data Type Verification**: Validated that each column retained its expected data type (e.g., User-ID as integer, rating as numeric).
- **Duplicate Removal**: Verified and removed duplicate rows in all datasets.
- **Referential Integrity**: Checked that User-ID and ISBN in Ratings and Reviews matched valid entries in Users and Books.
- **Format Accuracy**: Ensured there were no corrupted characters, blank strings, or malformed records.
- **Value Range Checks**: Confirmed all ratings were within the valid range of 0 to 10.
- **Row Count Consistency**: Compared row counts before and after cleaning to identify any unexpected data loss.

```
import pandas as pd
import numpy as np
```

```
df_books_raw=pd.read_csv("Books.csv")
df_books_clean=pd.read_csv("books_cleaned.csv")
```

```
C:\Users\ramya\AppData\Local\Temp\ipykernel_18880\3754276022.py:1: DtypeWarning: Columns (3) have mixed types. Specify dtype
option on import or set low_memory=False.
  df_books_raw=pd.read_csv("Books.csv")
```

```
df_users_raw=pd.read_csv("Users.csv")
df_users_clean=pd.read_csv("users_cleaned.csv")
```

```
df_ratings_raw=pd.read_csv("Ratings.csv")
df_ratings_clean=pd.read_csv("ratings_cleaned.csv")
```

```
df_reviews_raw=pd.read_csv("merged_reviews.csv")
df_reviews_clean=pd.read_csv("preprocessed_reviews_cleaned.csv")
```

```
C:\Users\ramya\AppData\Local\Temp\ipykernel_18880\3432971094.py:1: DtypeWarning: Columns (10,11,12) have mixed types. Specif
y dtype option on import or set low_memory=False.
  df_reviews_raw=pd.read_csv("merged_reviews.csv")
C:\Users\ramya\AppData\Local\Temp\ipykernel_18880\3432971094.py:2: DtypeWarning: Columns (10,11,12) have mixed types. Specif
y dtype option on import or set low_memory=False.
  df_reviews_clean=pd.read_csv("preprocessed_reviews_cleaned.csv")
```

```
print("Books - Rows Before:", len(df_books_raw), "| After:", len(df_books_clean))
print("Users - Rows Before:", len(df_users_raw), "| After:", len(df_users_clean))
print("Ratings - Rows Before:", len(df_ratings_raw), "| After:", len(df_ratings_clean))
print("Reviews - Rows Before:", len(df_reviews_raw), "| After:", len(df_reviews_clean))
```

```
Books - Rows Before: 271360 | After: 271360
Users - Rows Before: 278858 | After: 278858
Ratings - Rows Before: 1149780 | After: 1149780
Reviews - Rows Before: 79467 | After: 79446
```

```
def check_nulls(df, name):
    print(f"\n{name} - Null Values:")
    print(df.isnull().sum())

check_nulls(df_books_clean, "Books")
check_nulls(df_users_clean, "Users")
check_nulls(df_ratings_clean, "Ratings")
check_nulls(df_reviews_clean, "Reviews")
```

```
print("\nSample Books Data:")
print(df_books_clean.sample(5))

print("\nSample Users Data:")
print(df_users_clean.sample(5))

print("\nSample Ratings Data:")
print(df_ratings_clean.sample(5))

print("\nSample Reviews Data:")
print(df_reviews_clean.sample(5))
```

```
print(df_books_clean.dtypes)
assert df_books_clean['ISBN'].apply(lambda x: isinstance(x, str) and len(x.strip()) > 0).all()
assert df_ratings_clean['Book-Rating'].between(0, 10).all()
```

```
ISBN                  object
Book-Title            object
Book-Author           object
Year-Of-Publication  float64
Publisher             object
Image-URL-S           object
Image-URL-M           object
Image-URL-L           object
dtype: object
```

```
missing_books = set(df_ratings_clean['ISBN']) - set(df_books_clean['ISBN'])
print("Missing ISBNs in Books:", missing_books)
missing_users = set(df_ratings_clean['User_Id']) - set(df_users_clean['user_id'])
print("Missing user_ids in Users:", missing_users)
missing_books_reviews = set(df_reviews_clean['ISBN']) - set(df_books_clean['ISBN'])
missing_users_reviews = set(df_reviews_clean['User_Id']) - set(df_users_clean['User_Id'])
print("Missing ISBNs in Reviews:", missing_books_reviews)
print("Missing user_ids in Reviews:", missing_users_reviews)
```

```
print("Books - Null values:\n", df_books_clean.isnull().sum())
print("Users - Null values:\n", df_users_clean.isnull().sum())
print("Ratings - Null values:\n", df_ratings_clean.isnull().sum())
print("Reviews - Null values:\n", df_reviews_clean.isnull().sum())
```

```
print("Books - Rows with nulls:")
print(df_books_clean[df_books_clean.isnull().any(axis=1)])

print("\nUsers - Rows with nulls:")
print(df_users_clean[df_users_clean.isnull().any(axis=1)])

print("\nRatings - Rows with nulls:")
print(df_ratings_clean[df_ratings_clean.isnull().any(axis=1)])

print("\nReviews - Rows with nulls:")
print(df_reviews_clean[df_reviews_clean.isnull().any(axis=1)])
```

```
print(df_books_clean['Book-Title'].sample(5))
print(df_books_clean['Book-Title'].str.contains(r'[^\x00-\x7F]', na=False).sum(), "non-ASCII titles")
```

```
46484     clan del oso cavernario, el - los hijos de la ...
258837                                          reflections
17922                                            passing on
268429                              the chaotic miss crispino
86782                                    all the queen's men
Name: Book-Title, dtype: object
6634 non-ASCII titles
```

```
print("Books duplicates:", df_books_clean.duplicated().sum())
print("Users duplicates:", df_users_clean.duplicated().sum())
print("Ratings duplicates:", df_ratings_clean.duplicated().sum())
print("Reviews duplicates:", df_reviews_clean.duplicated().sum())
```

```
Books duplicates: 0
Users duplicates: 0
Ratings duplicates: 0
Reviews duplicates: 0
```

```
if 'rating' in df_ratings_clean.columns:
    print("Invalid ratings:", ~df_ratings_clean['rating'].between(0, 10).sum())
```

```
print("Dangling ISBNs in Ratings:", ~df_ratings_clean['ISBN'].isin(df_books_clean['ISBN']).sum())
print("Dangling User-IDs in Ratings:", ~df_ratings_clean['User-ID'].isin(df_users_clean['User-ID']).sum())
```

```
Dangling ISBNs in Ratings: -1031137
Dangling User-IDs in Ratings: -1149781
```

```
print(df_books_clean.sample(3))
print(df_users_clean.sample(3))
print(df_ratings_clean.sample(3))
print(df_reviews_clean.sample(3))
```

```
print("Empty book titles:", (df_books_clean['Book-Title'].str.strip() == "").sum())
```

```
Empty book titles: 0
```

Week 1: SDLC, Requirement Gathering, and System Design

Day 1: Sprint Planning & Requirements Understanding

- Defined User Stories:
- Understood dataset interactions:
    - books, users, ratings, and reviews.

Day 2–3: Business Understanding & Use Cases

- Defined Business Goals:
- Listed Expected Outcomes:
- Stakeholder Identification:
    - Users (readers), admins (managing insights), devs (modeling team).

Day 4–5: System Design & Data Exploration

- Created ER Diagram:
- Designed Data Flow Diagrams (DFDs):

- Conducted Dataset Analysis:
    - Identified attributes, distributions, missing values, outliers, and relationships.

Day 6–7: Preprocessing Strategy & Feedback

- Finalized Data Cleaning Rules:
    - Remove duplicates, fix nulls, standardize formats (especially dates).

Week 2: Data Cleaning, Text Preprocessing, and Validation

Day 1: Cleaning Books, Users & Ratings

- Removed duplicates, handled missing values logically.
- Standardized publication year and rating formats.

Day 2–3: Cleaning Reviews

- Merged review datasets from multiple sources (Amazon, Goodreads).
- Ensured consistent formatting and no duplication/data loss.

3️⃣ Day 4–5: Text Preprocessing (for NLP)

- Applied:
    - Stopword Removal
    - Special Character Removal
    - Tokenization
    - Lowercase Conversion

Day 6–7: Data Validation & Sprint Review

- **Data Validation Checks**:
    - Row count consistency before/after cleaning.
    - No introduction of missing values.
    - Format validation across datasets.

Challenges:

- Formatting inconsistencies, especially in review data.
- Complex merging of multi-source datasets