**Data Cleaning Tasks**

**1.First Step：**

Load the data into a DataFrame: You can load data into a DataFrame using the read function. For example, to load a CSV file into a DataFrame, you can use the following code:

**df = spark.read.csv("path/to/file.csv", header=True, inferSchema=True)**

**----------------------------------------------------------------------------**

**2.Check for missing values:**

You can use the isNull() or isNotNull() functions to check for missing values in the data. For example, to check for missing values in all columns of the DataFrame, you can use the following code:

**from pyspark.sql.functions import col, sum**

**# Count the number of null values in each column**

**df.select([sum(col(c).isNull().cast("int")).alias(c) for c in df.columns]).show()**

This code uses the isNull() function to check for null values in each column of the DataFrame, and then counts the number of null values in each column using the sum() function.

**----------------------------------------------------------------------------**

**3. Remove missing values:**

You can use the dropna() function to remove rows that contain missing values. For example, to remove rows that contain at least one missing value, you can use the following code:

**# Drop rows with null values**

**df = df.dropna()**

**----------------------------------------------------------------------------**

**4. Check for duplicates:**

You can use the dropDuplicates() function to check for duplicate rows in the data. For example, to check for duplicate rows in the DataFrame, you can use the following code:

**# Count the number of duplicate rows**

**df.count() - df.dropDuplicates().count()**

This code counts the number of duplicate rows in the DataFrame by subtracting the number of unique rows from the total number of rows.

**----------------------------------------------------------------------------**

**5. Correct data types:**

You can use the cast() function to convert the data types of the columns in the DataFrame. For example, to convert a column containing string values to numeric values, you can use the following code:

**# Convert string column to numeric column**

**df = df.withColumn("numeric\_column", df["string\_column"].cast("float"))**

**----------------------------------------------------------------------------**

**6. Correct inconsistent values:**

You can use various functions to correct inconsistent values in the data. For example, to replace all occurrences of a certain value with another value, you can use the replace() function. For example:

**# Replace all occurrences of "foo" with "bar" in a column**

**from pyspark.sql.functions import when**

**df=df.withColumn("column\_name",when(df["column\_name"]=="foo","bar").otherwise(df["column\_name"]))**

**----------------------------------------------------------------------------**

**Snap Shots for Cleaning Procedure**

**HDFS Storage Check:**

文本

描述已自动生成

**JSON file overview:**

图形用户界面, 文本

描述已自动生成

文本

描述已自动生成

图形用户界面, 文本

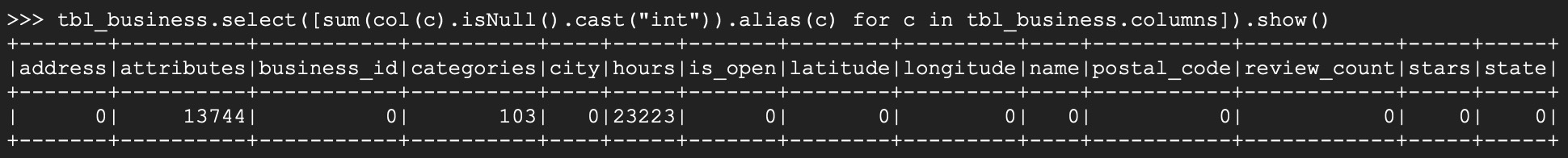
描述已自动生成

图形用户界面, 文本

描述已自动生成 图形用户界面, 文本

描述已自动生成

**1.Data Cleaning for tbl\_business:**



文本

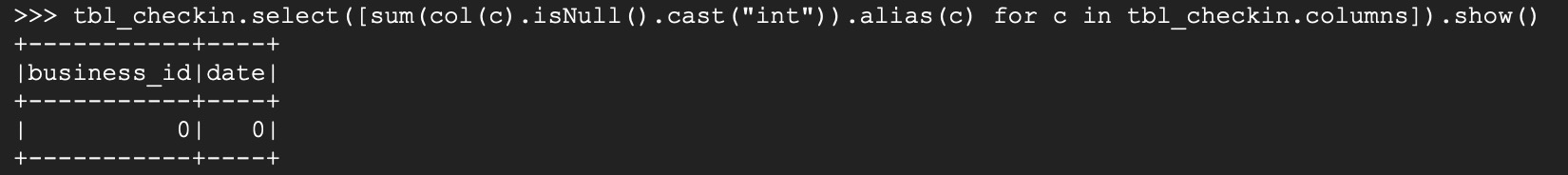
描述已自动生成

**1.1Data after Null value dropping and missing value filling:**

图形用户界面, 文本

描述已自动生成

**2.Data Cleaning for tbl\_checkin:**



文本

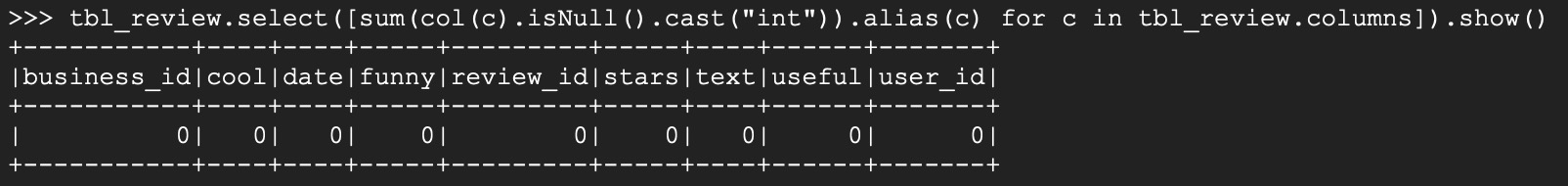
描述已自动生成

Way2: By using JOIN

文本

描述已自动生成

**3.Data Cleaning for tbl\_review:**

****

**4.Data Cleaning for tbl\_tip**

**5.Data Cleaning for tbl\_user**

**图形用户界面

描述已自动生成**

**Loading it to Hive**

文本

描述已自动生成