

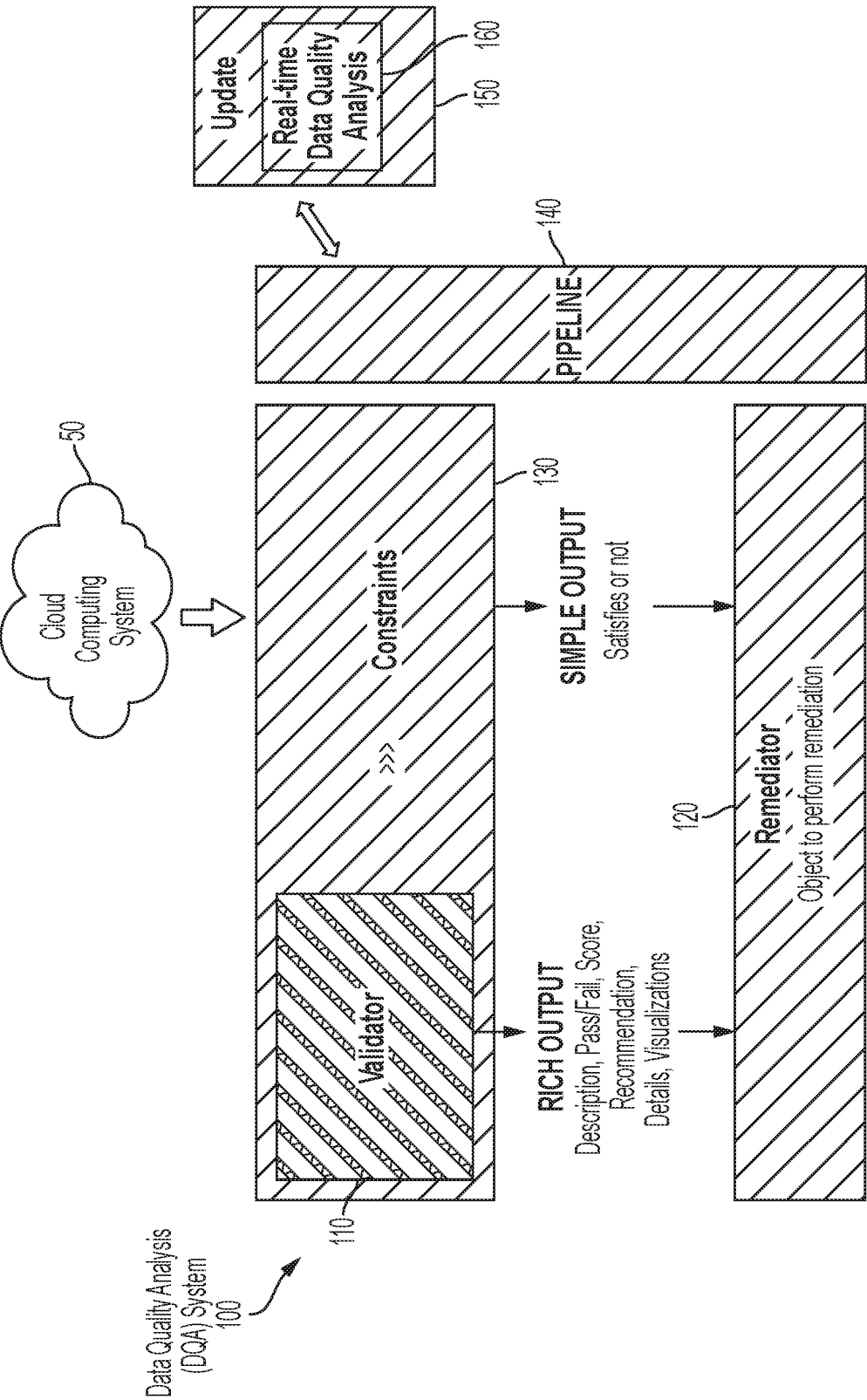


FIG. 1

Example Data Quality Checks in Validator 110

- Null values
- Infinity values
- Duplicate and repeated values
- Uniqueness checks
- Remediation
 - Data imputation of missing values
- Univariate and Multivariate checks
 - In multivariate data, determine if different features are related
- Time series checks, such as
 - Sufficient duration
 - Minimum frequency
 - Duplicate timestamps
 - Missing data
 - Trend, seasonality, outliers
- Other data quality checks

FIG. 2

Time Series
Data
300

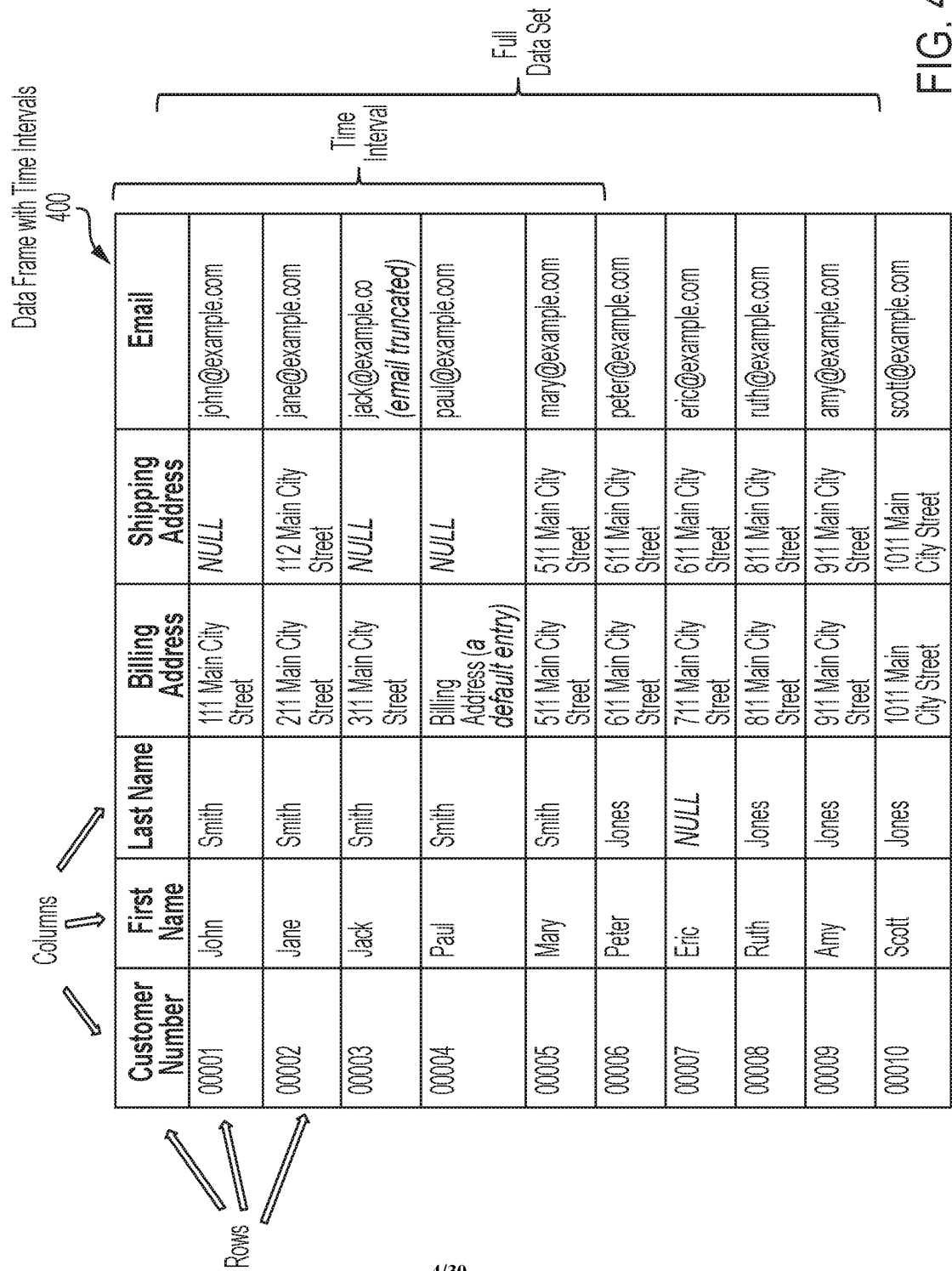


```
import pandas as pd
carbs = ['pizza', 'hamburger', 'rice']
pd.Series(carbs)

0 'pizza'
1 'hamburger'
2 'rice'

dtype: object
```

FIG. 3



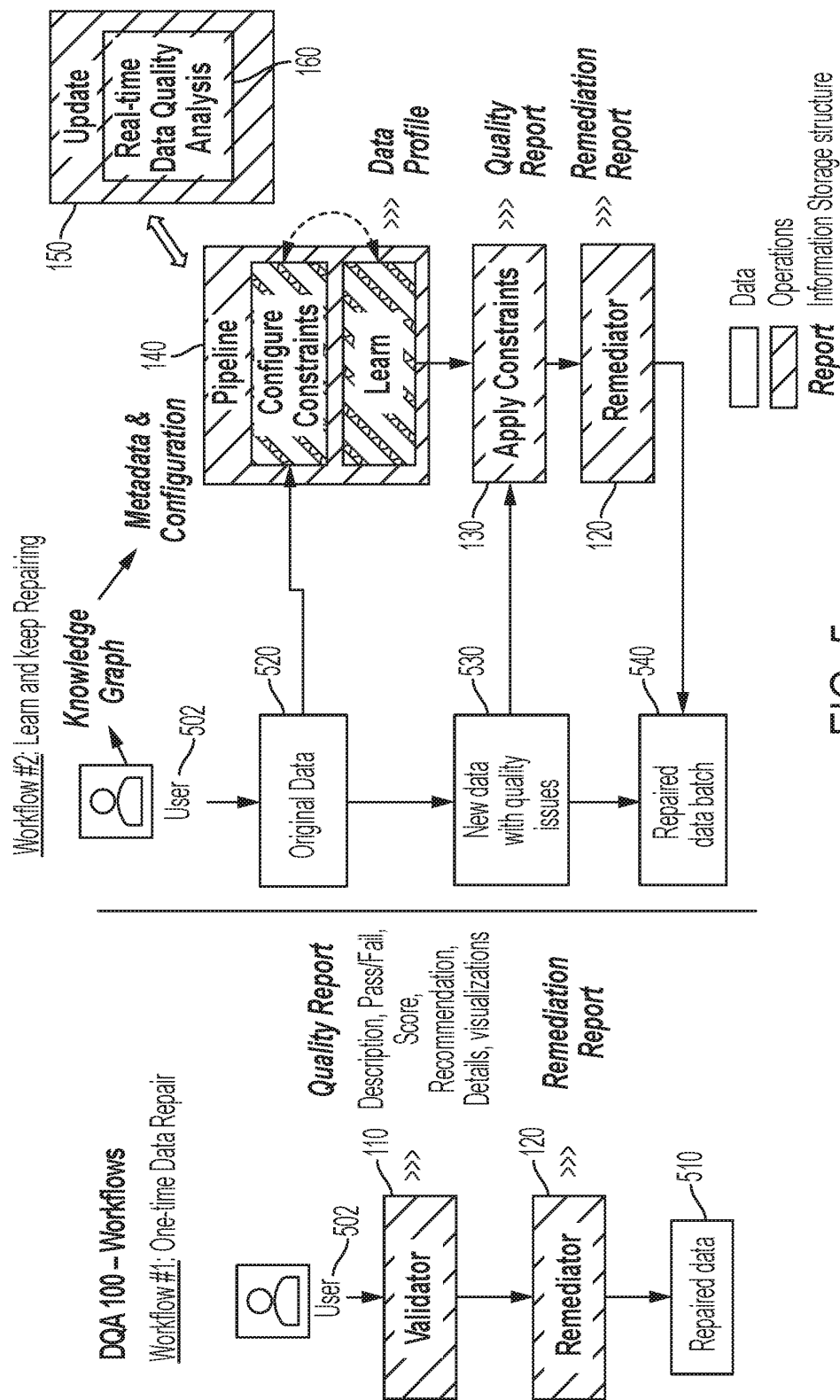


FIG. 5

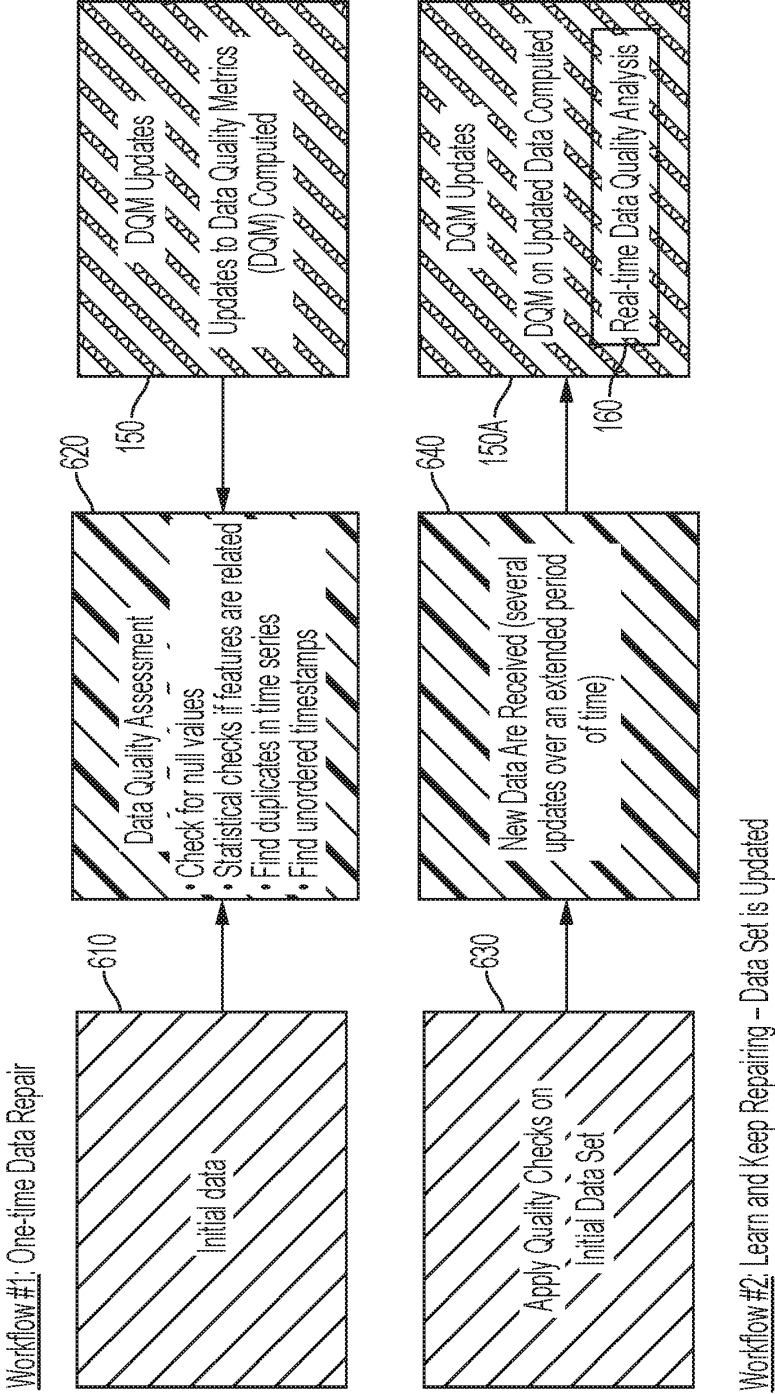


FIG. 6

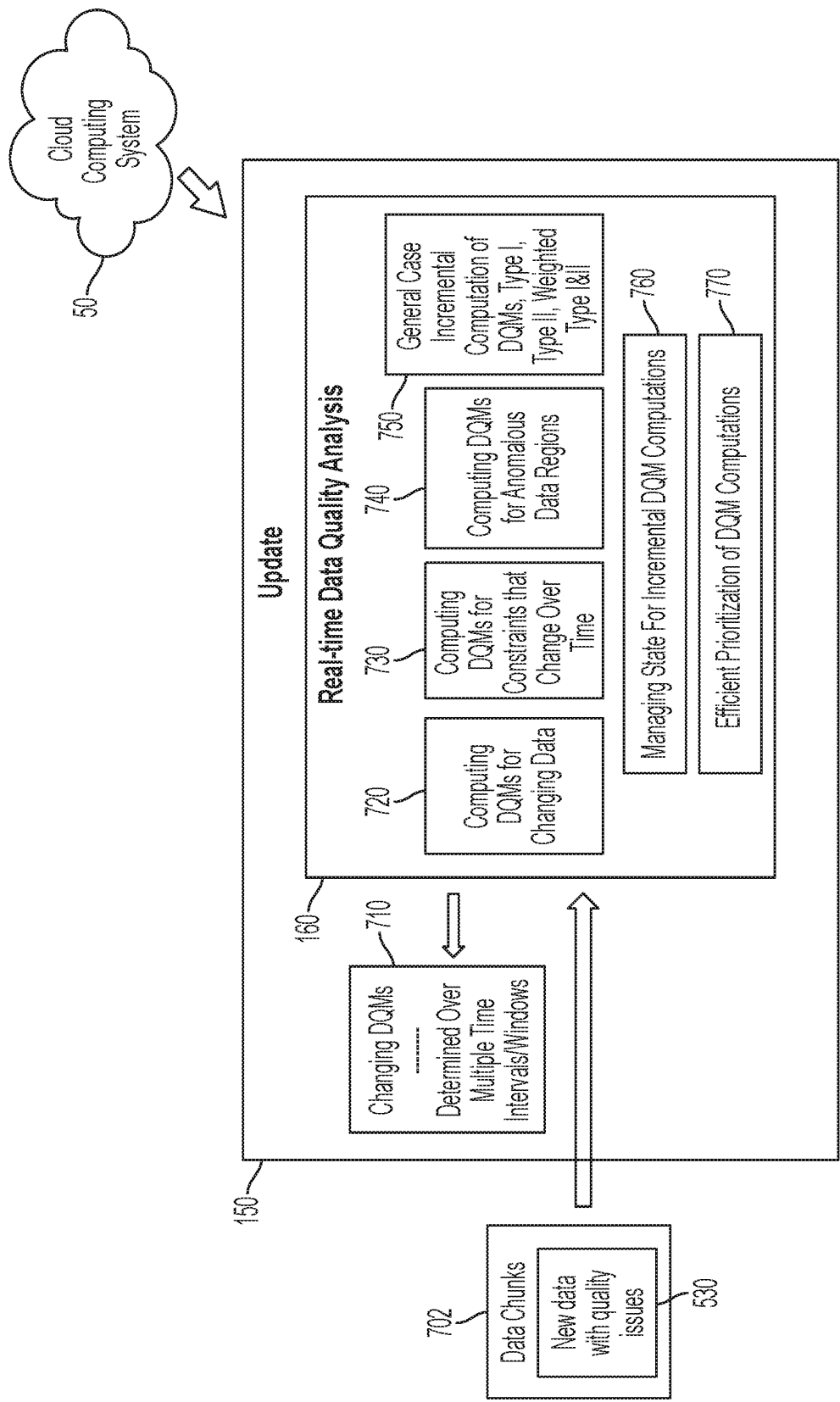


FIG. 7

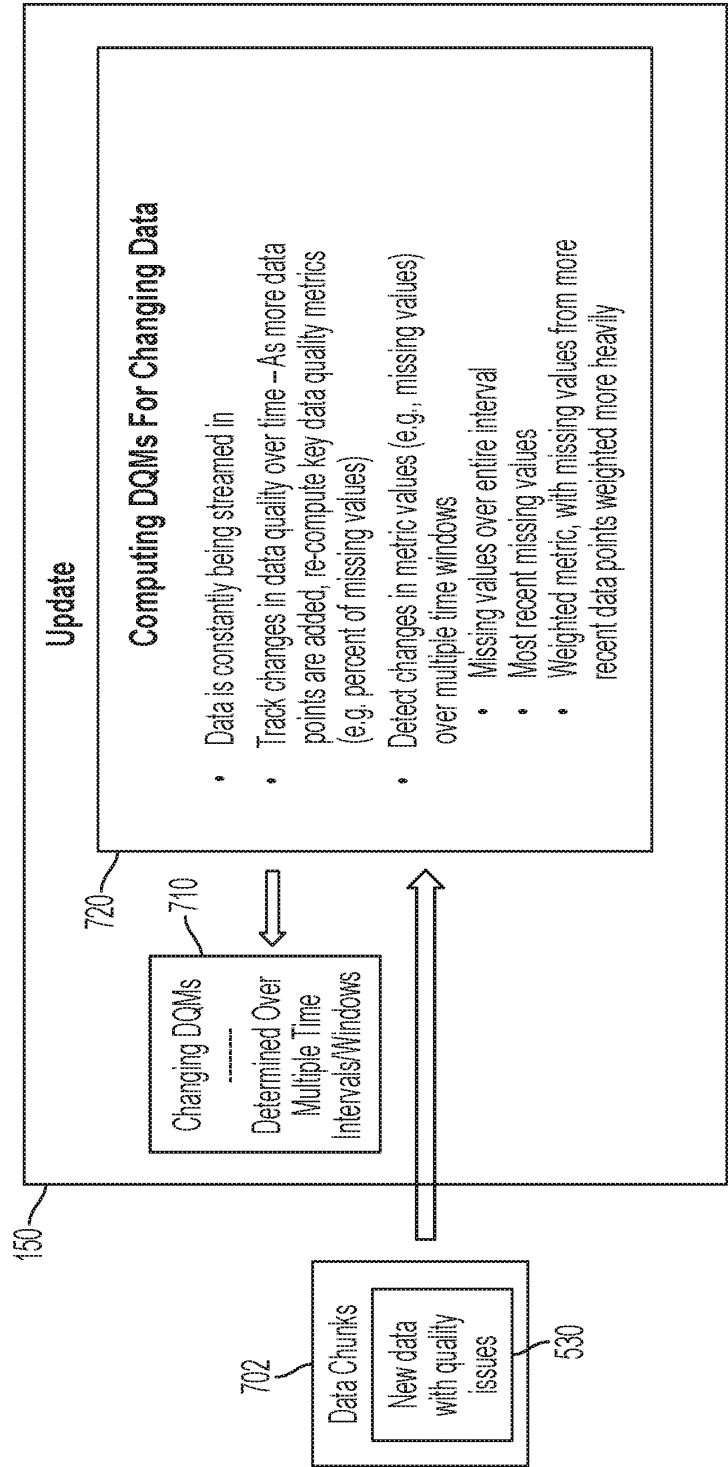


FIG. 8

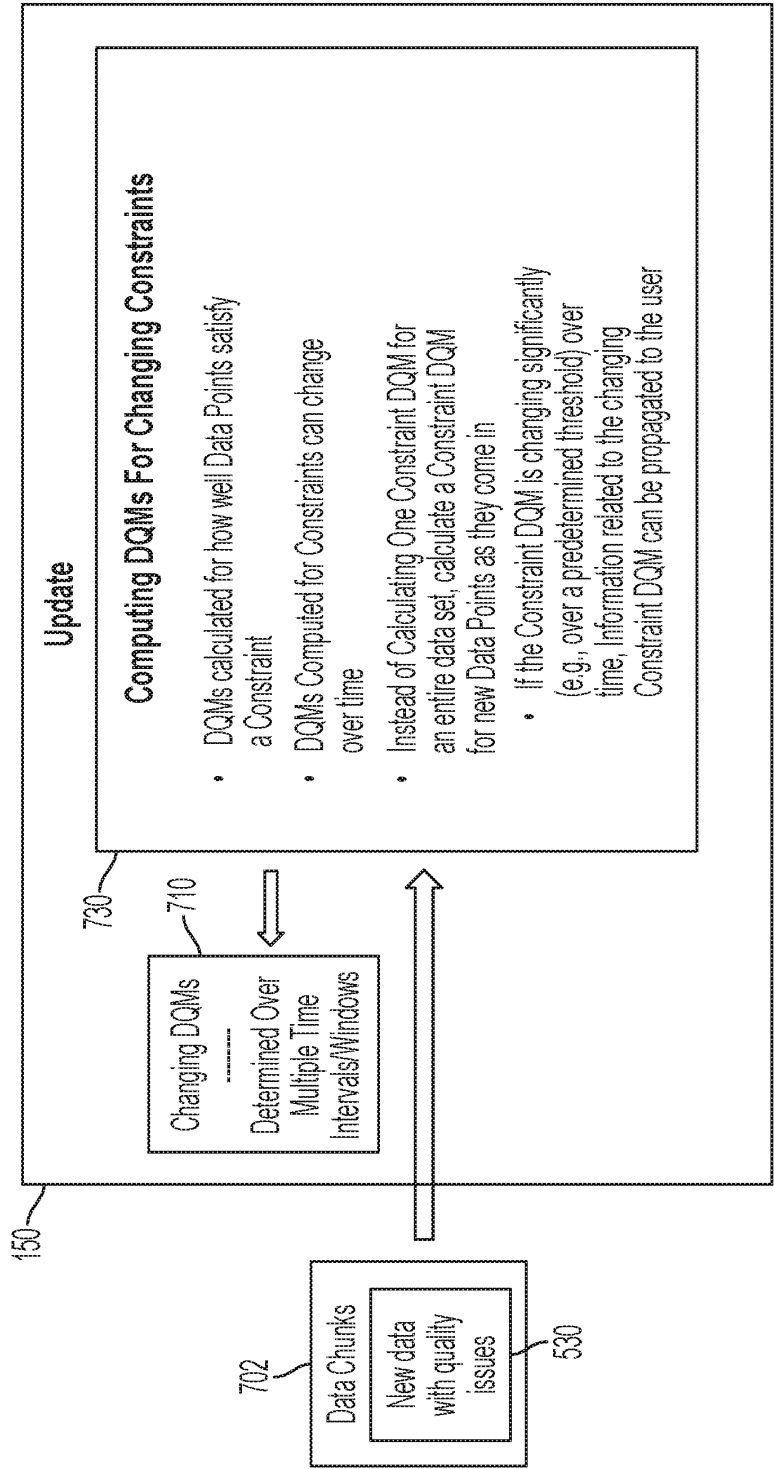


FIG. 9

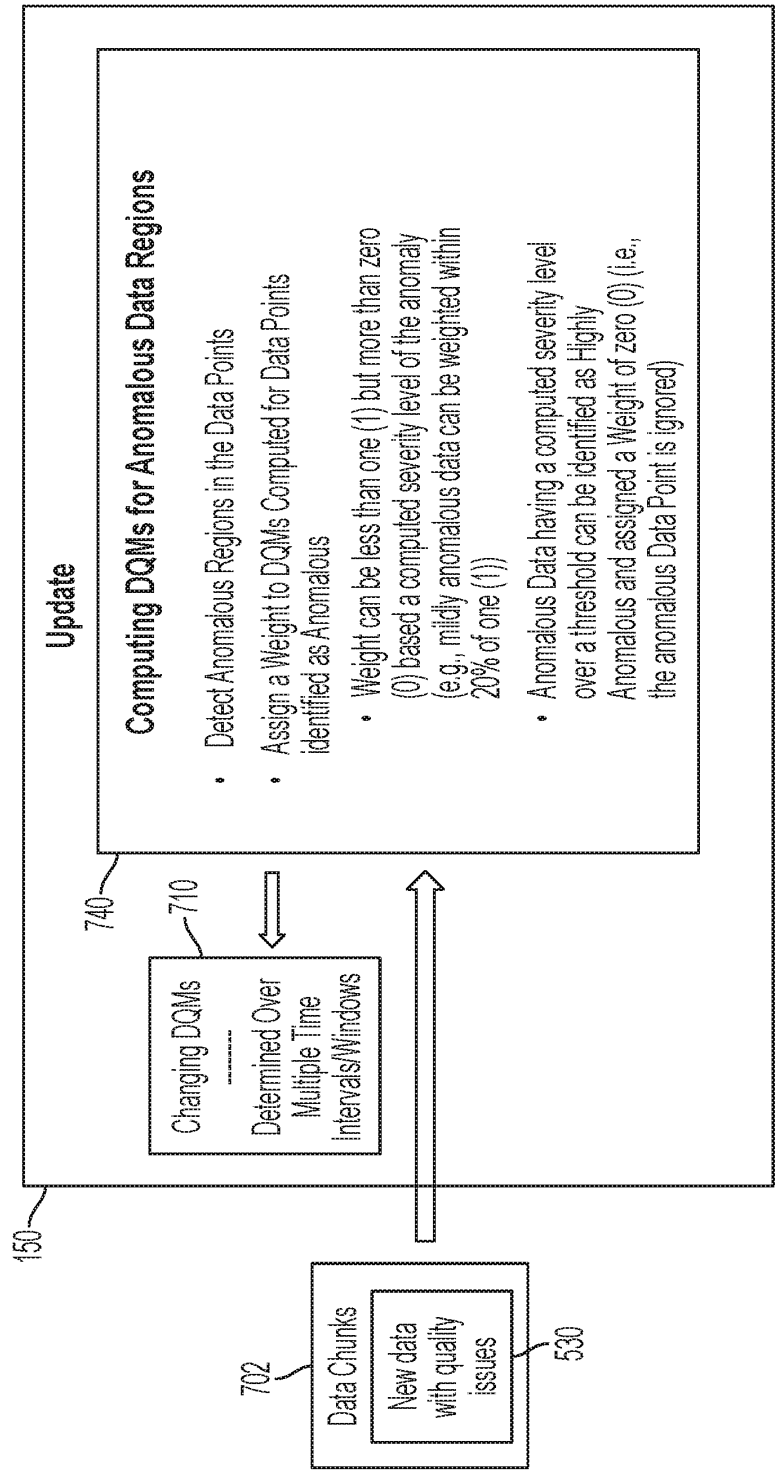


FIG. 10A

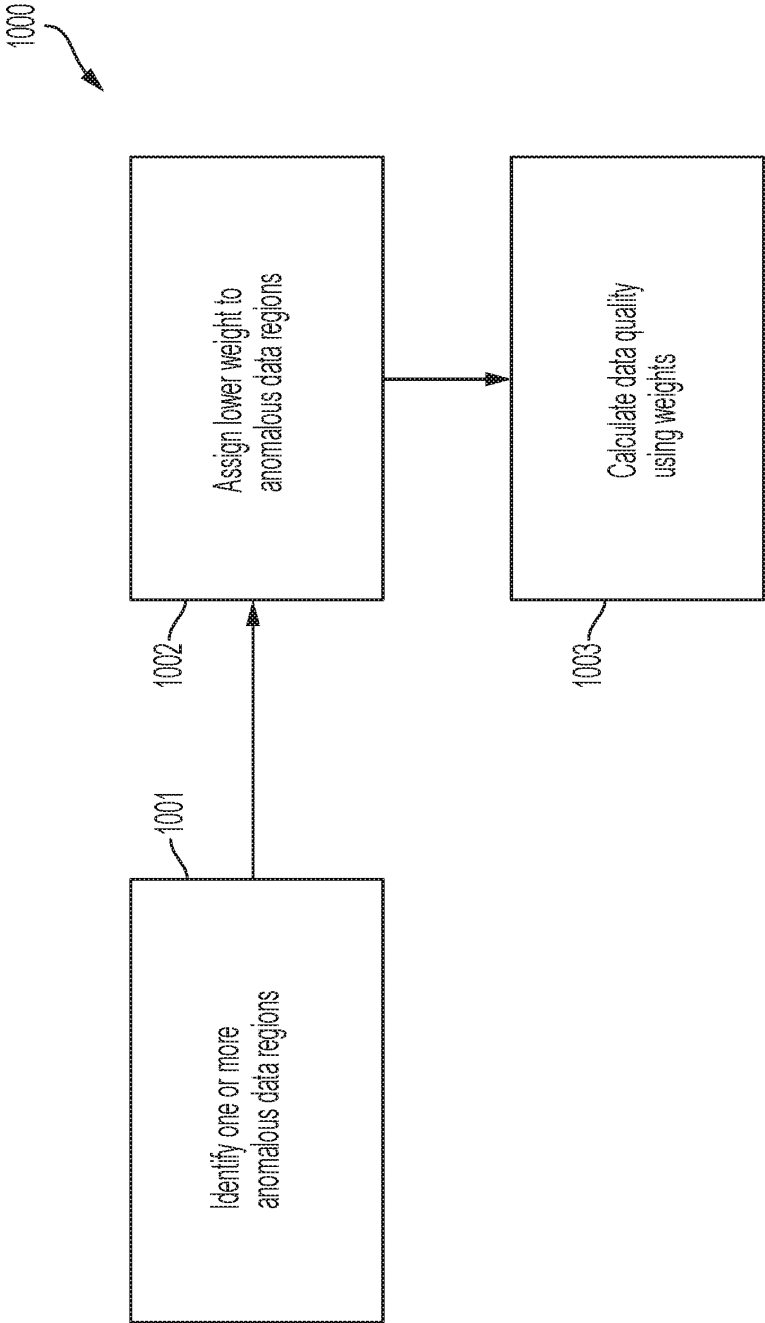


FIG. 10B

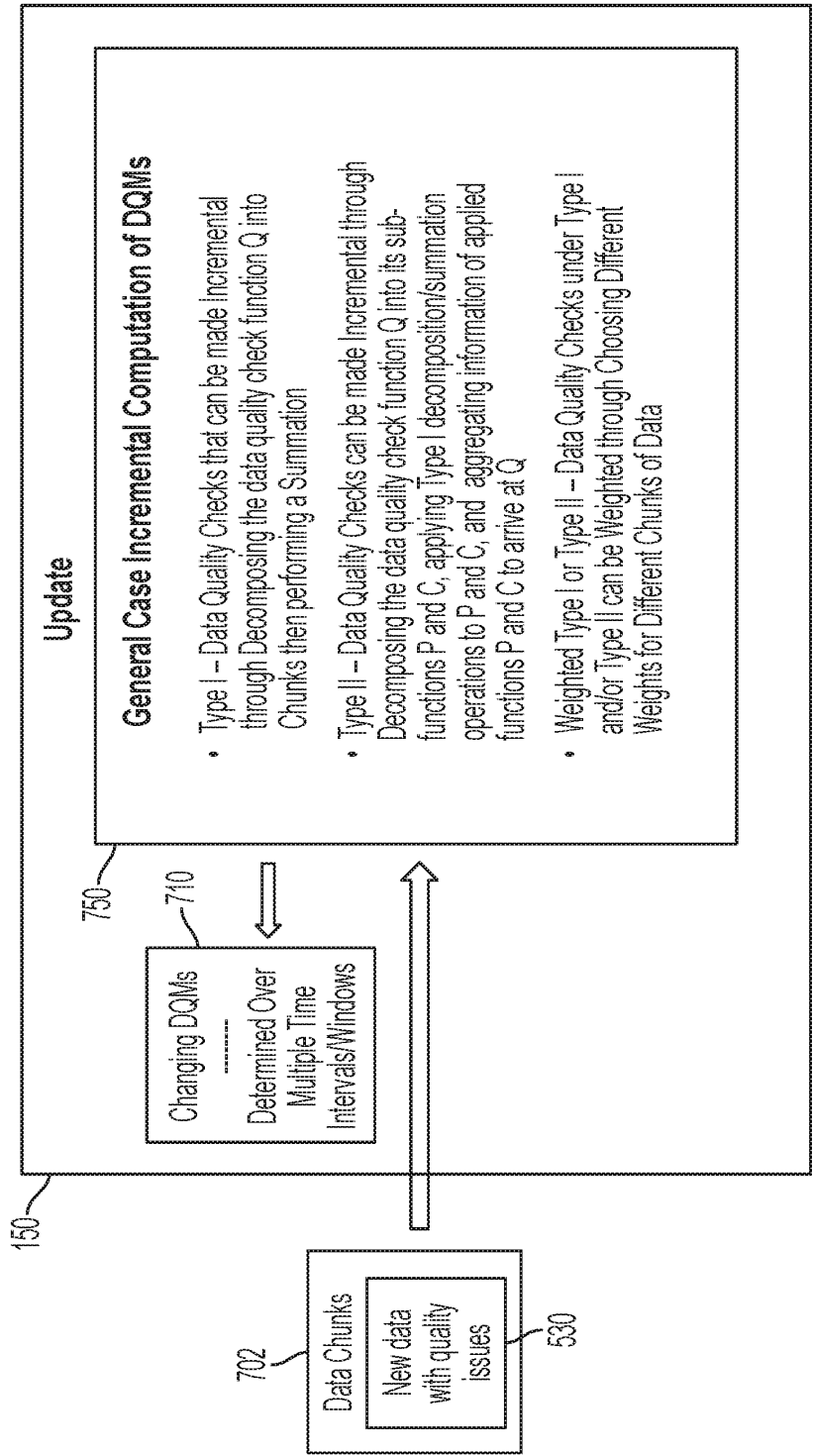
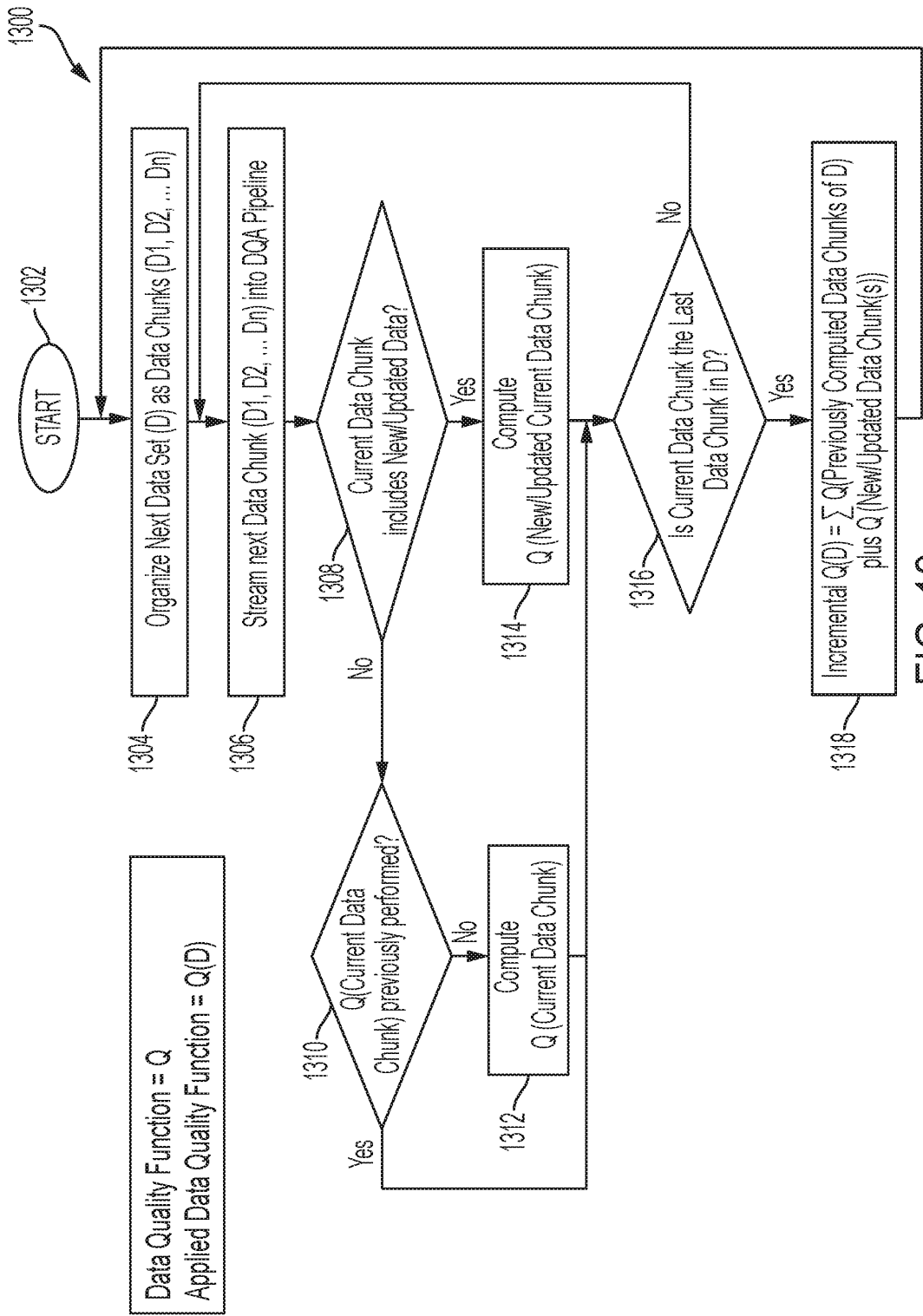


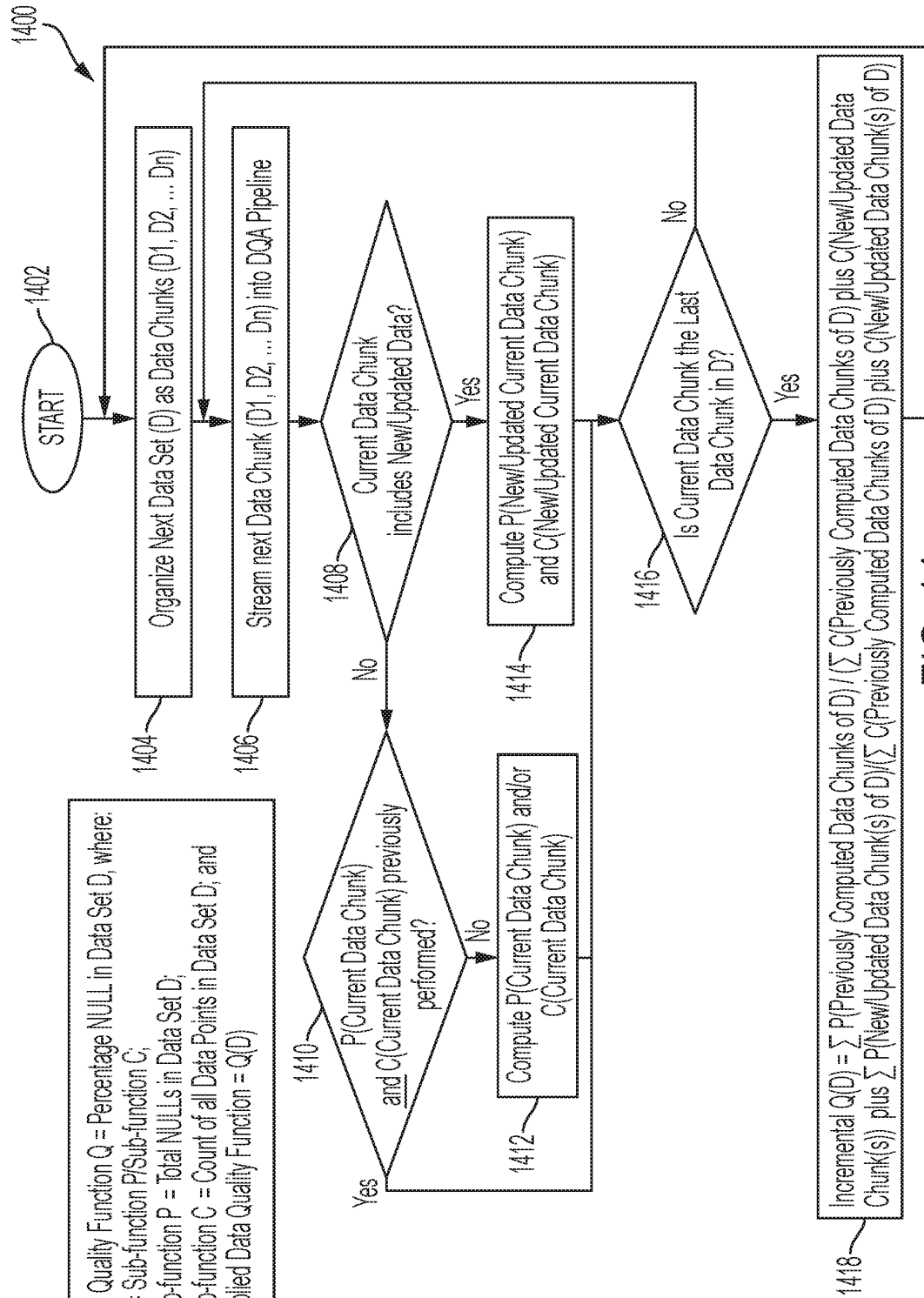
FIG. 11

General Case Incremental Computations of DQMs in DQA System 100

- Type I Data Quality Checks
 - check_duplicate_column_names
 - check_column_name
 - check_column_data_type
- Type II Data Quality Checks
 - check_na_columns
 - check_specific_element_column
 - check_infinity_column
 - check_zero_ratio_column
 - check_constant_columns
 - check_duplicate_values
 - check_most_occurring_values
 - check_low_variance_variables

FIG. 12





$$Q(D_W) = Q(\{D_1, D_2, \dots, D_n\}) = \sum_{i=1}^{i=n} Q(D_i)$$

Type I – Equation 1

$$Q(D_W \cup D_W) = \sum_{i=1}^{i=n} Q(D_i) + Q(D_{n+1}) = Q(D_W) + Q(D_{n+1})$$

Type I – Equation 2

$$Q(D) = Q(\{D_1, D_2, \dots, D_n\}) = \frac{\sum_{i=1}^{i=n} P(D_i)}{\sum_{i=1}^{i=n} C(D_i)}$$

Type II – Equation 3

$$Q(D_W \cup D_{n+1}) = \frac{\sum_{i=1}^{i=n} P(D_i) + P(D_{n+1})}{\sum_{i=1}^{i=n} C(D_i) + C(D_{n+1})} = \frac{P(D_W) + P(D_{n+1})}{C(D_W) + C(D_{n+1})}$$

Type II – Equation 4

FIG. 15

Assume that the weight for each Data Chunk is: $\{W_i\}$ for $i \in [1, 2, \dots, n]$

$D_W = \{D_1, D_2, \dots, D_n\}$, ordered over time:

$$Q(D_W) = \sum_{i=1}^{i=n} w_i * Q(D_i)$$

Type I - Equation 5

$D_W = \{D_1, D_2, \dots, D_n\}$, ordered over time:

$$Q(D_W) = \frac{\sum_{i=1}^{i=n} w_i * P(D_i)}{\sum_{i=1}^{i=n} w_i * C(D_i)}$$

Type II - Equation 6

FIG. 16

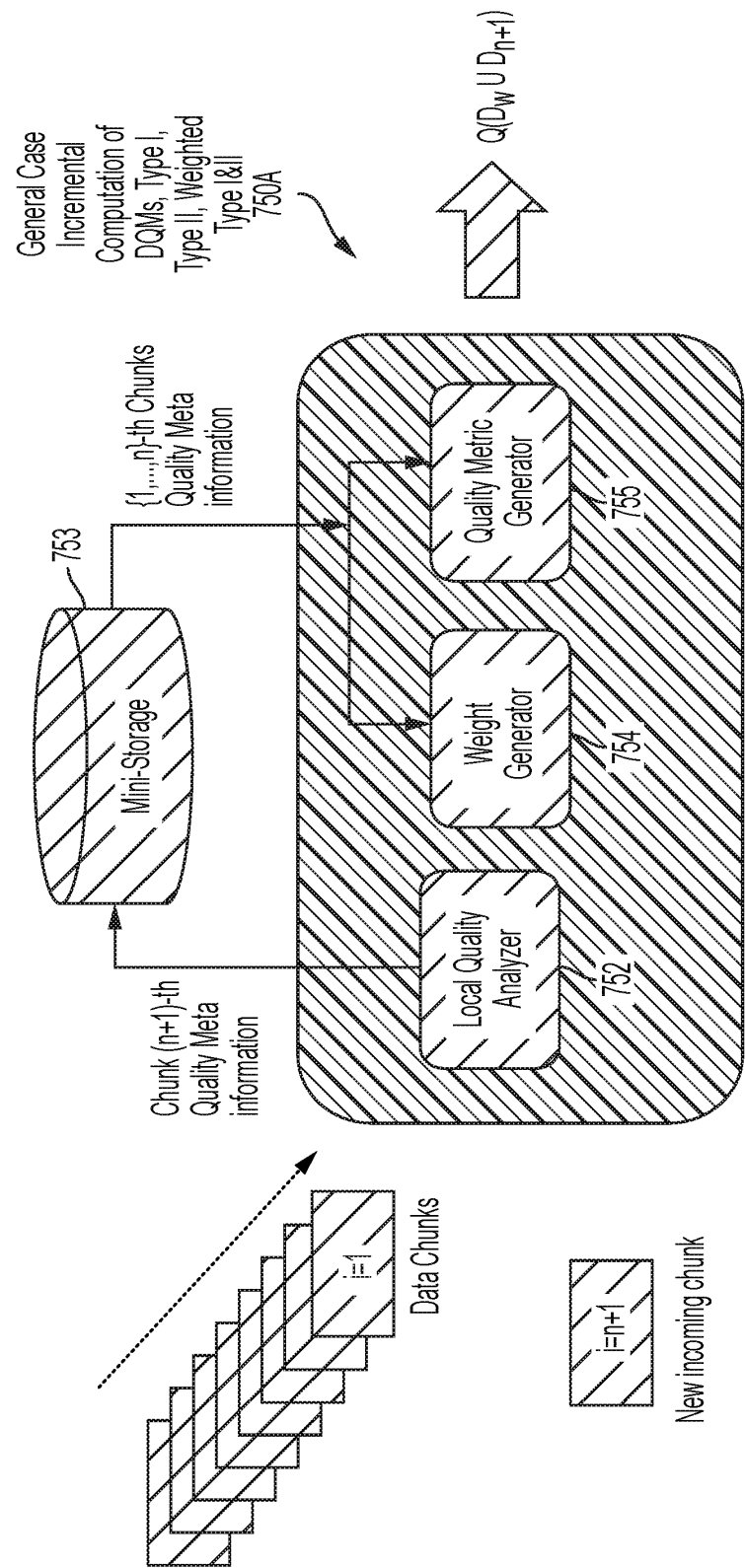


FIG. 17

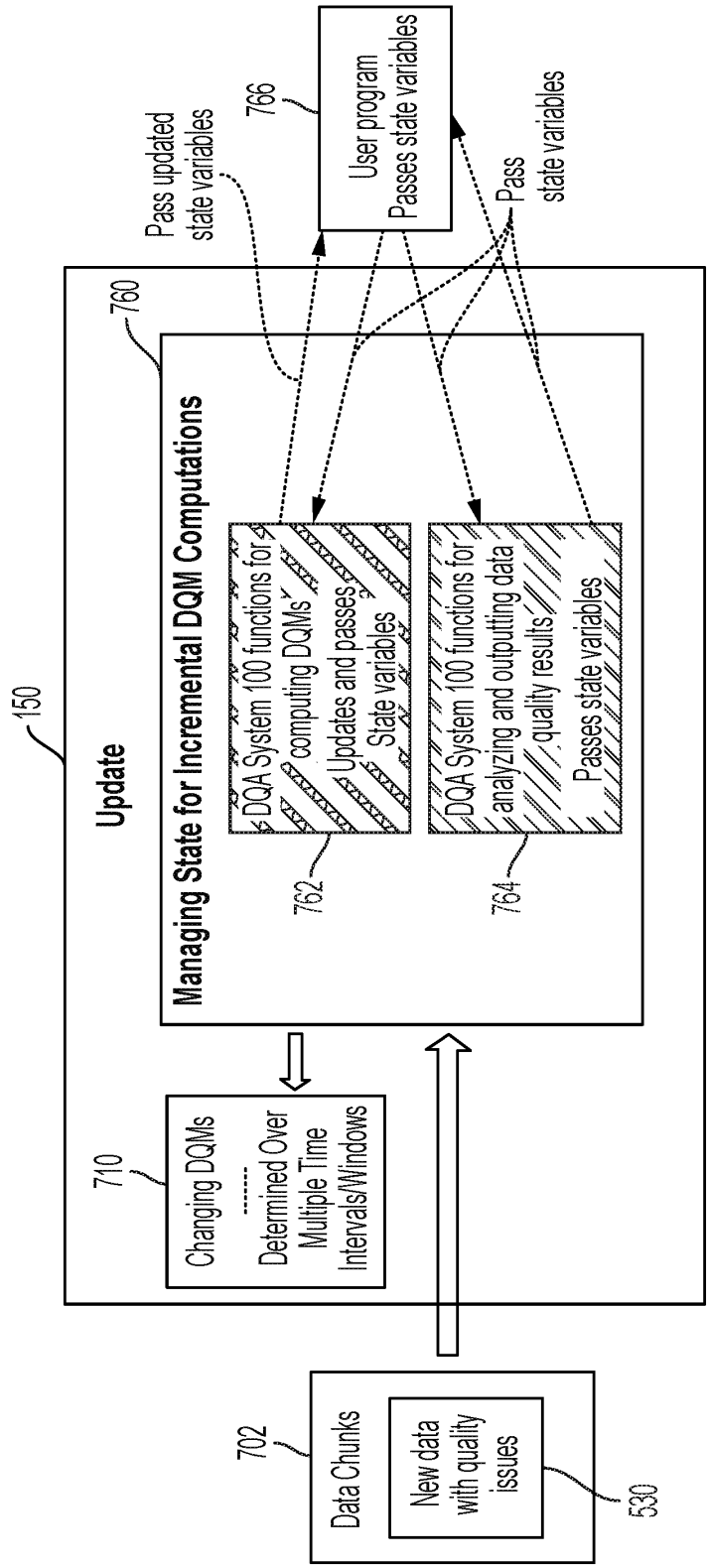


FIG. 18

Example: State Information for DQA System 100

- Data can be passed to the DQA System 100 as Python data-frames
- Data-frames include intervals (e.g., shown in FIG. 4) which are ranges of consecutive rows representing a subset of full data sets
- Incremental methods have a parameter, `interval_info_list`, where each element in the list contains information about the interval:
 - Row numbers corresponding to the interval
 - Other fields corresponding to data quality metrics for the interval:
 - `check_na_columns`: column counts of null values
 - `check_specific_element_column`: column proportions of missing values
 - `check_infinity_column`: column proportions of infinity values
 - `check_zero_ratio_column`: column proportions of zero values
 - Etc.
- `check_na_columns_incremental(df, row_offset, interval_info_list)`
 - `df` may only be a portion of the data set. If `row_offset > 0`, previous rows of the dataset were analyzed earlier
 - `row_offset` represents the true row number of the first row of `df`
 - `interval_info_list`: results from analyses of previous rows of the same data set
 - `check_na_columns_incremental` analyzes `df`, appends the results to `interval_info_list`, and returns the updated value of `interval_info_list`

FIG. 19

Example: Analyzing Data Quality Metrics Over Time Intervals

- analyze_data_by_interval(interval_info_list):
 - Analyzes data quality metrics across several intervals
 - Aggregate analysis across longer time intervals
 - Analysis for individual time intervals
 - Detect trends, anomalies across different time intervals
- check_na_columns_incremental gives proportions of NaNs over:
 - Entire data range
 - Only for most recent data
 - Changing proportion over time plotted as a graph
 - Using different weighting schemes (e.g., exponentially weighted moving average) to weight more recent data with higher weights

FIG. 20

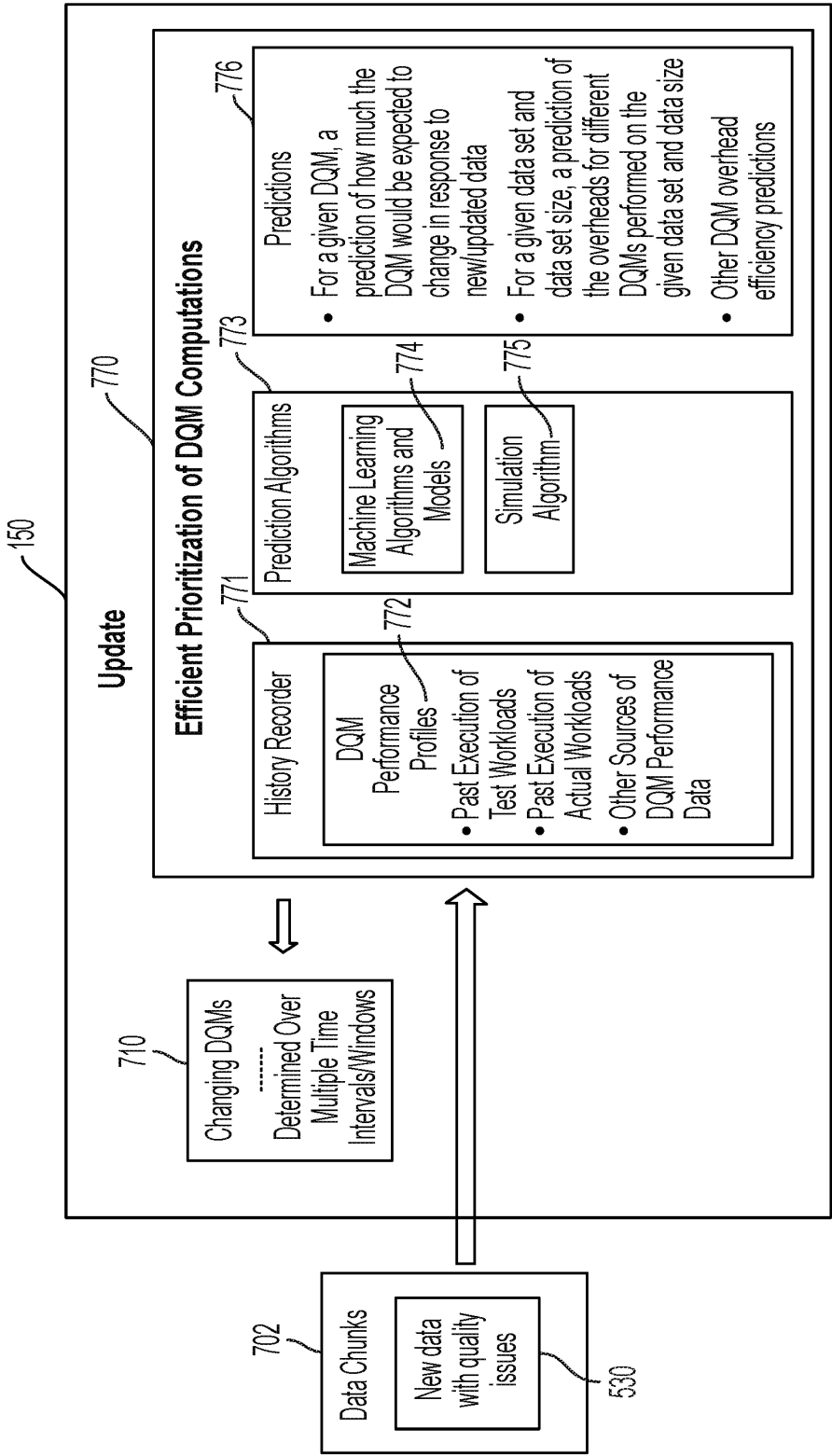


FIG. 21

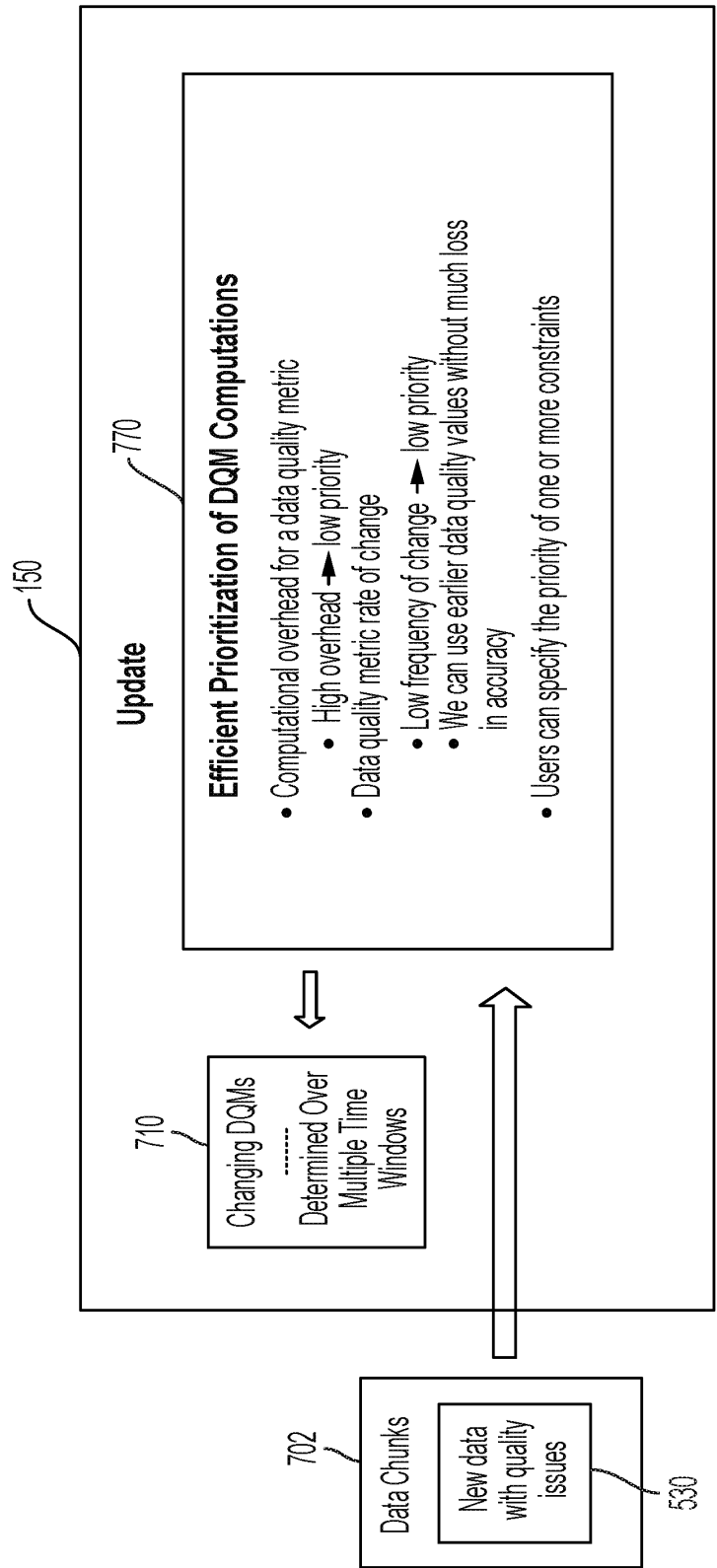


FIG. 22

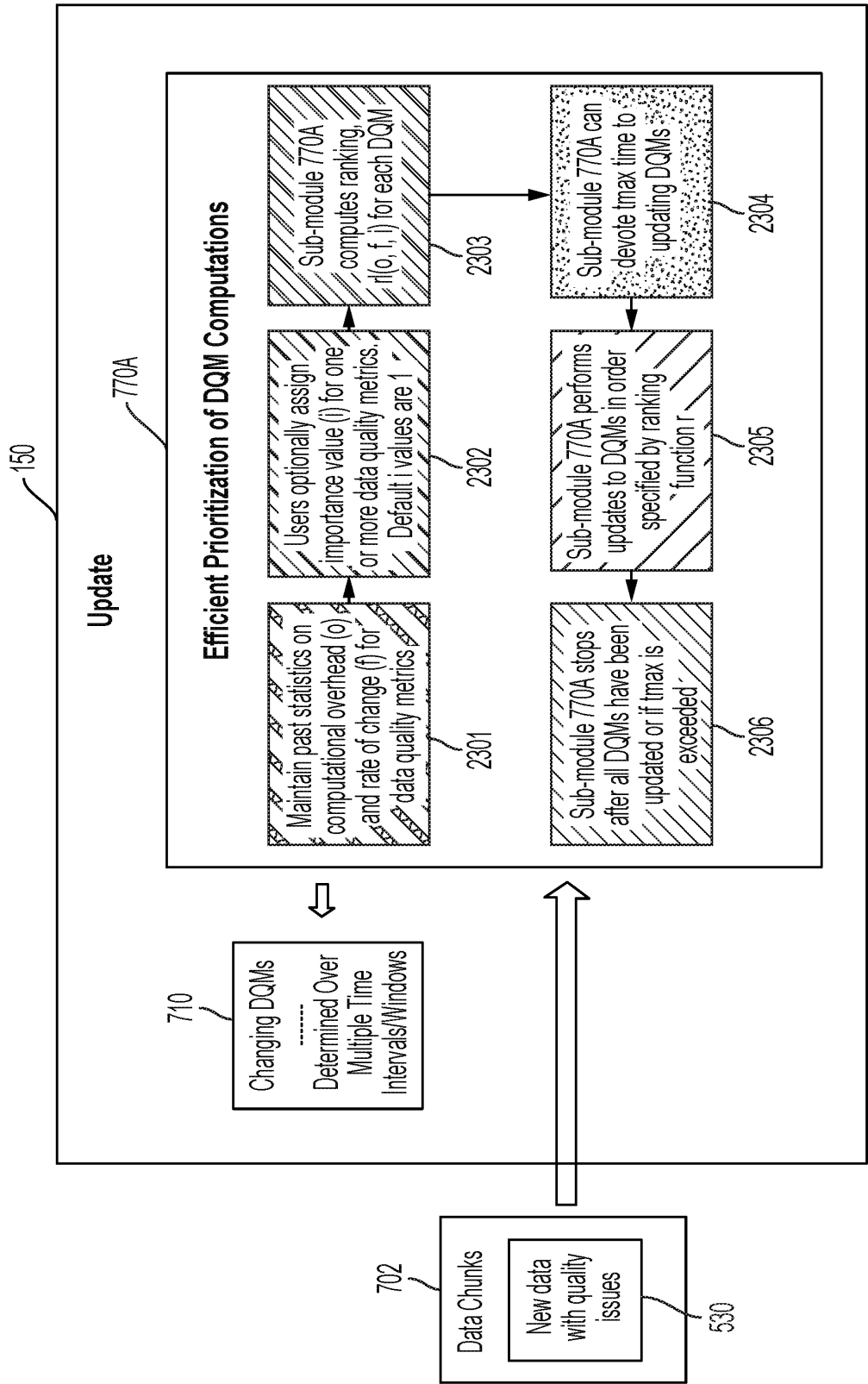


FIG. 23

Efficient Prioritization of DQM Computations 770A

- Several ranking functions can be used for r
- Example: $\text{rating} = a * o + b * f + c * i$
 - a is a negative constant, b , and c are positive constants
 - o : overhead for computing data quality constraint
 - f : frequency of change seen in past computations of data quality constraint
 - i : importance of data quality constraint (default value of 1 if not provided)
- Ranking function r orders constraints in decreasing order by ratings

FIG. 24

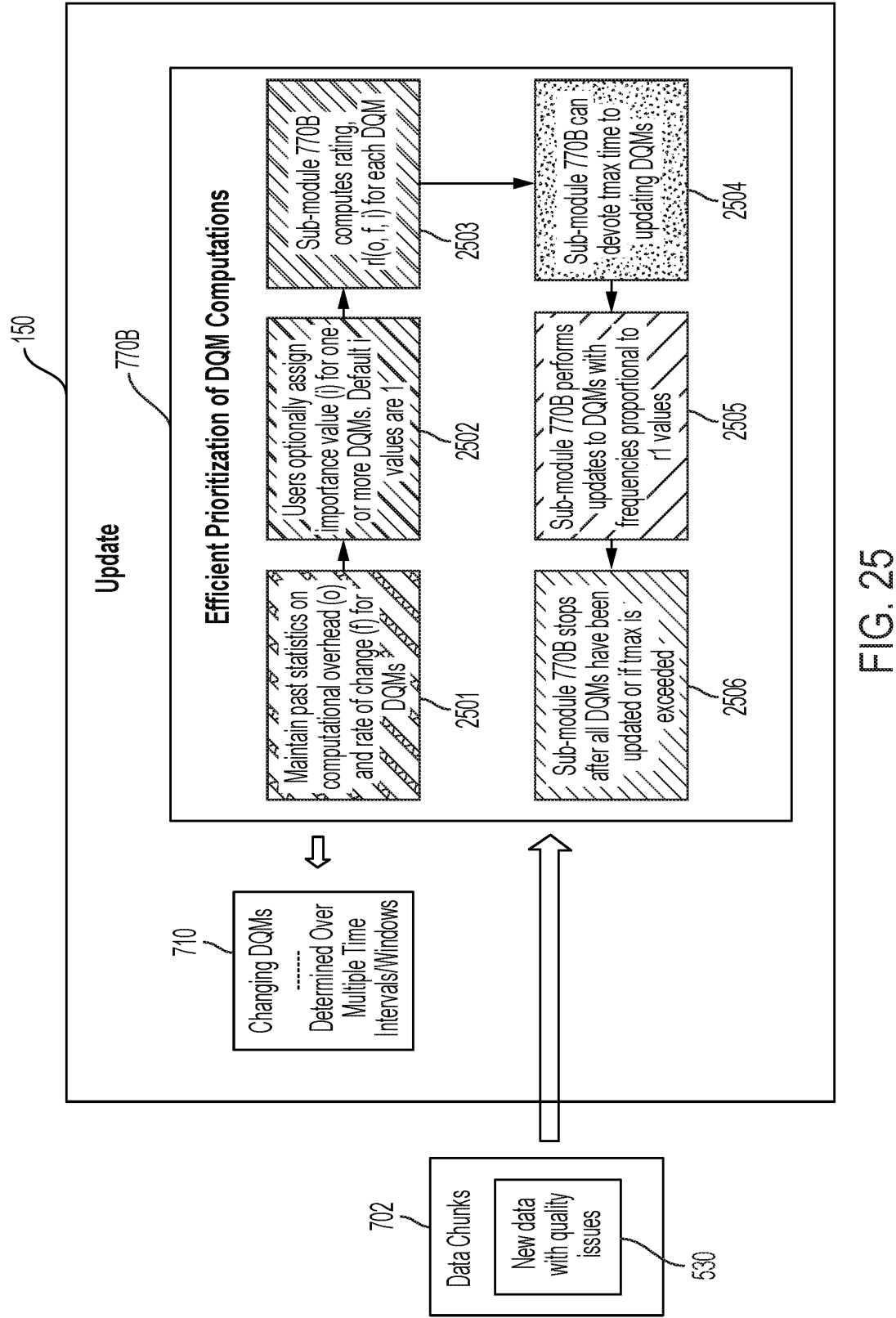


FIG. 25

Efficient Prioritization of DQM Computations 770B

- Several rating functions can be used for $r1$
- Example: $r1 = a * o + b * f + c * i$
 - a is a negative constant, b , and c are positive constants
 - o : overhead for computing data quality constraint
 - f : frequency of change seen in past computations of data quality constraint
 - i : importance of data quality constraint (default value of 1 if not provided)
- $r1$ values represent relative frequencies for which sub-module 770B should update a data quality metric
 - All $r1$ values should be positive
- In order to ensure that all data quality metrics are computed at least some of the time, even if t_{max} is always exceeded, ratio between highest and lowest values should not be too high

FIG. 26

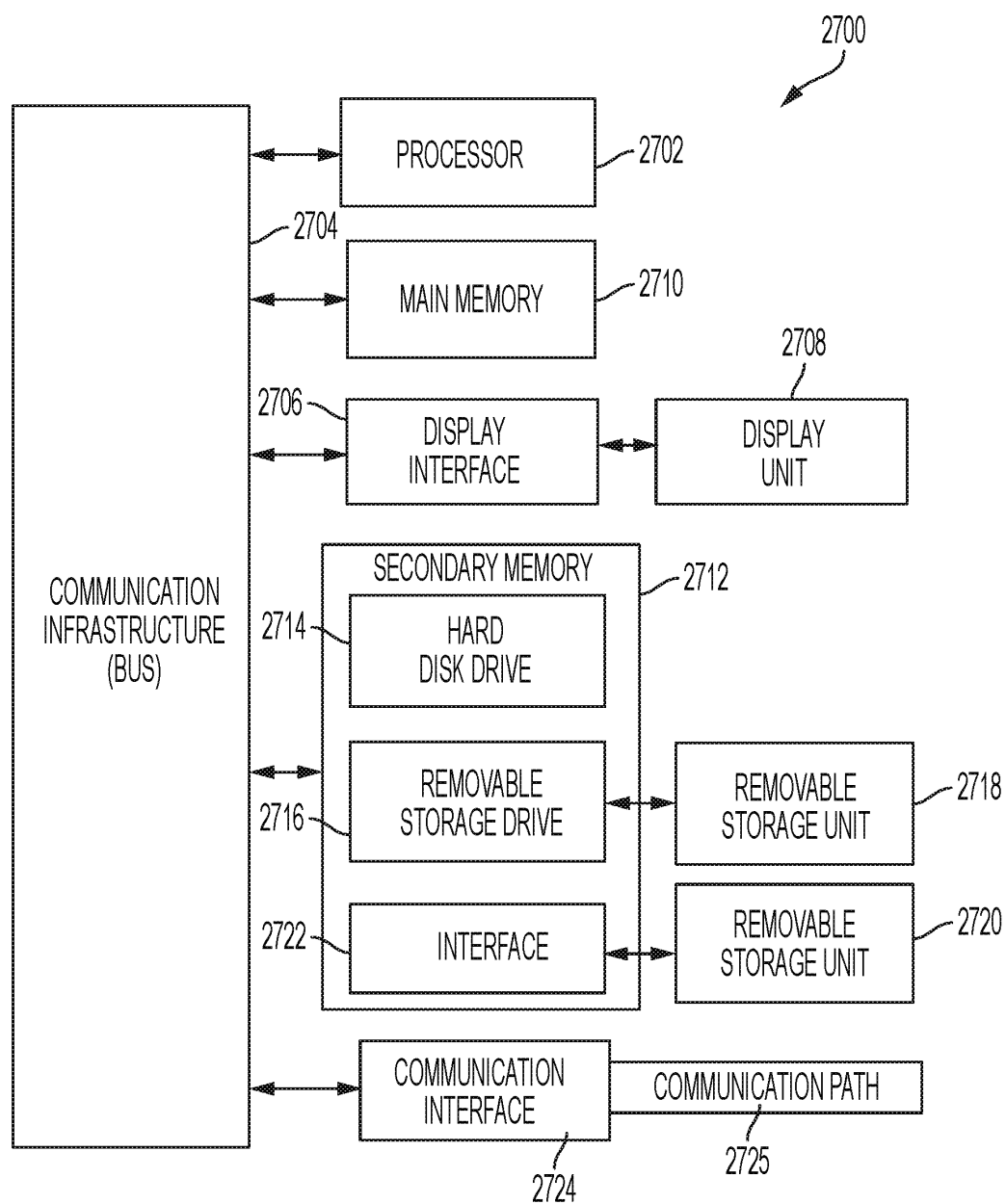


FIG. 27

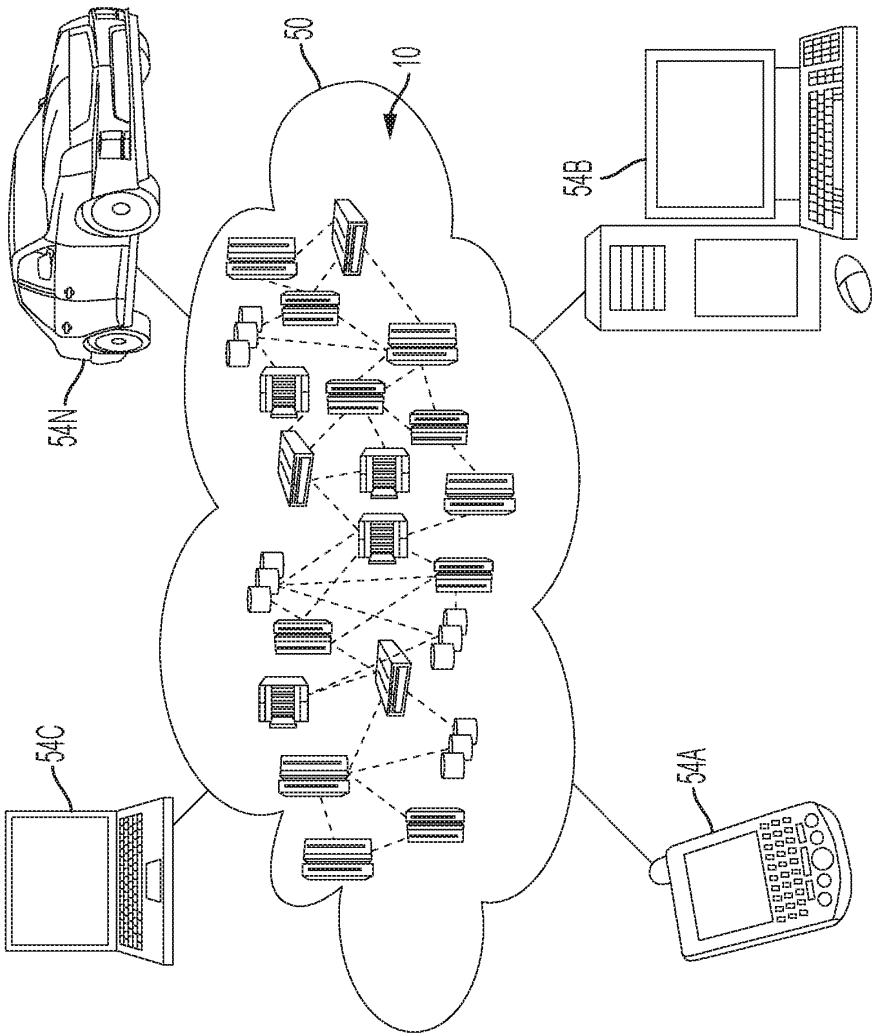


FIG. 28

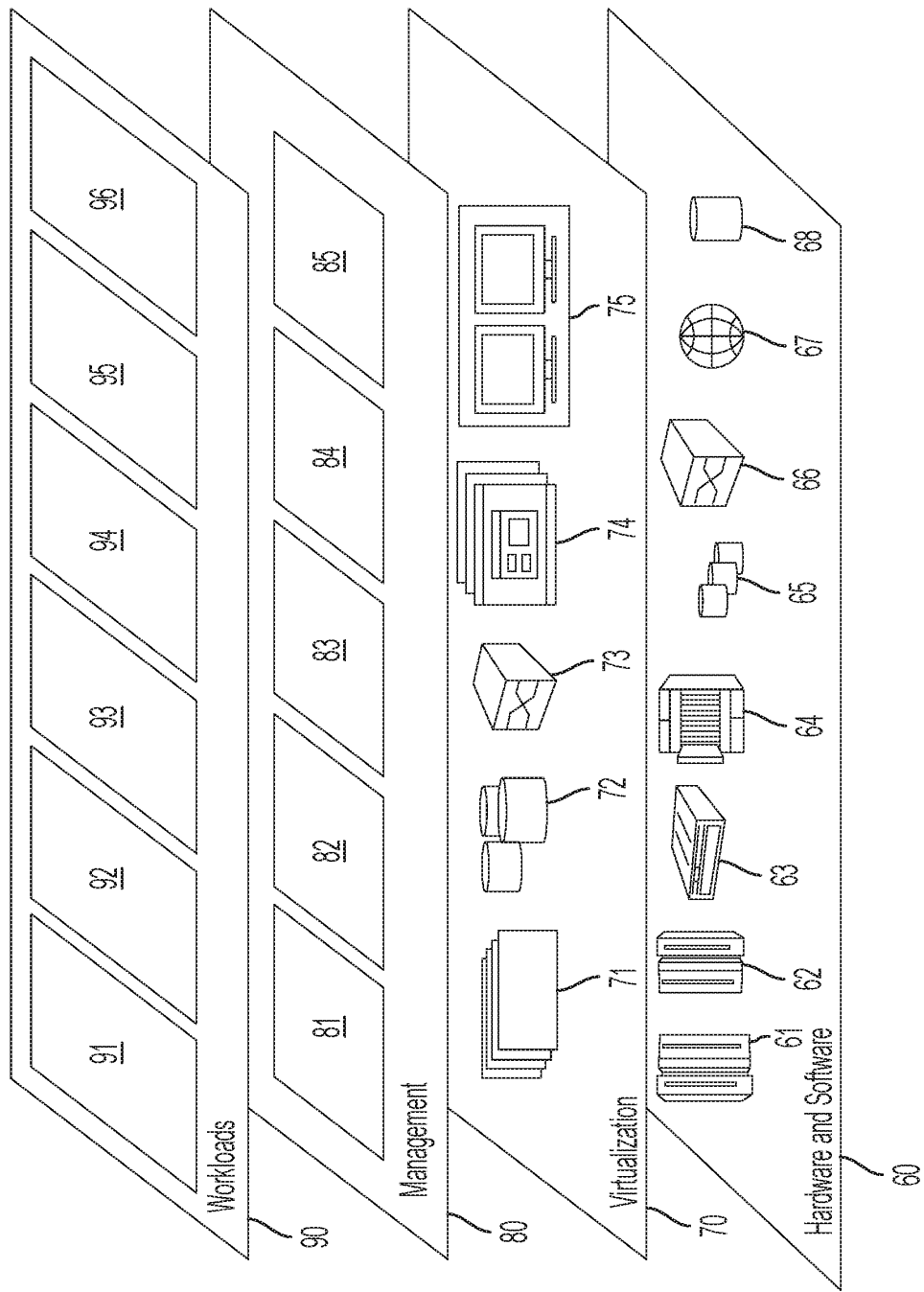


FIG. 29