

Part I

MapReduce patterns

Summarization Patterns

Summarization patterns

- Are used to implement applications that produce top-level/summarized view of the data
 - Numerical summarizations (Statistics)
 - Inverted index
 - Counting with counters

Summarization Patterns

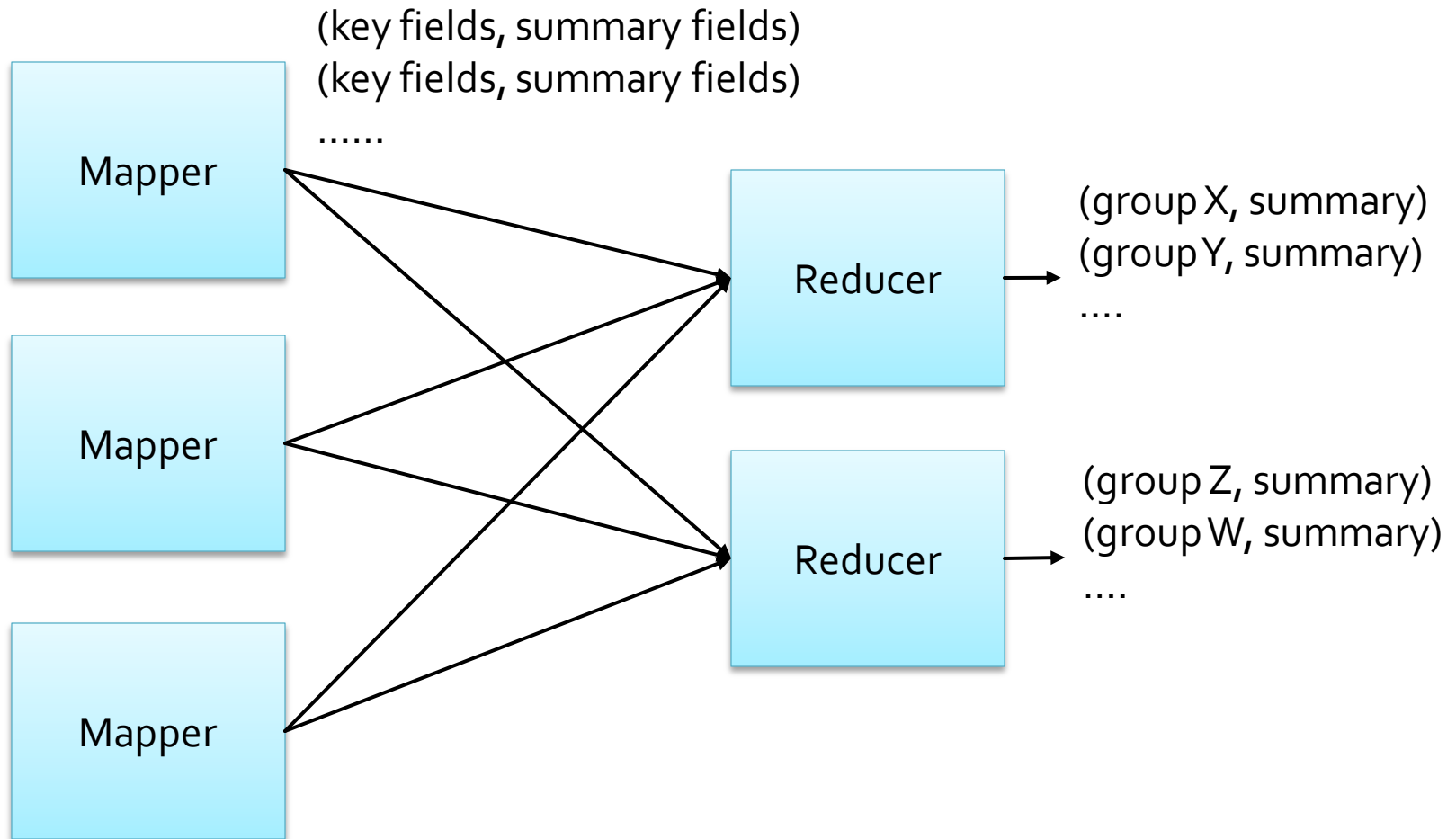
Numerical Summarizations

- Goal
 - Group records/objects by a key field(s) and calculate a numerical aggregate (average, max, min, standard deviation,..) per group
- Provide a top-level view of large input data sets
- Motivation
 - Few high-level statistics can be analyzed by domain experts to identify trends, anomalies, ...

Numerical Summarizations - structure

- Mappers
 - Output (key, value) pairs where
 - key is associated with the fields used to define groups
 - value is associated with the fields used to compute the aggregate statistics
- Reducers
 - Receive a set of numerical values for each “group-by” key and compute the final statistics for each “group”
- Combiners
 - If the computed statistic has specific properties (e.g., it is commutative and associative), combiners can be used to speed up performances

Numerical Summarizations - structure



Numerical Summarizations

- Known uses
 - Word count
 - Record count (per group)
 - Min/Max/Count (per group)
 - Average/Median/Standard deviation (per group)

Summarization Patterns

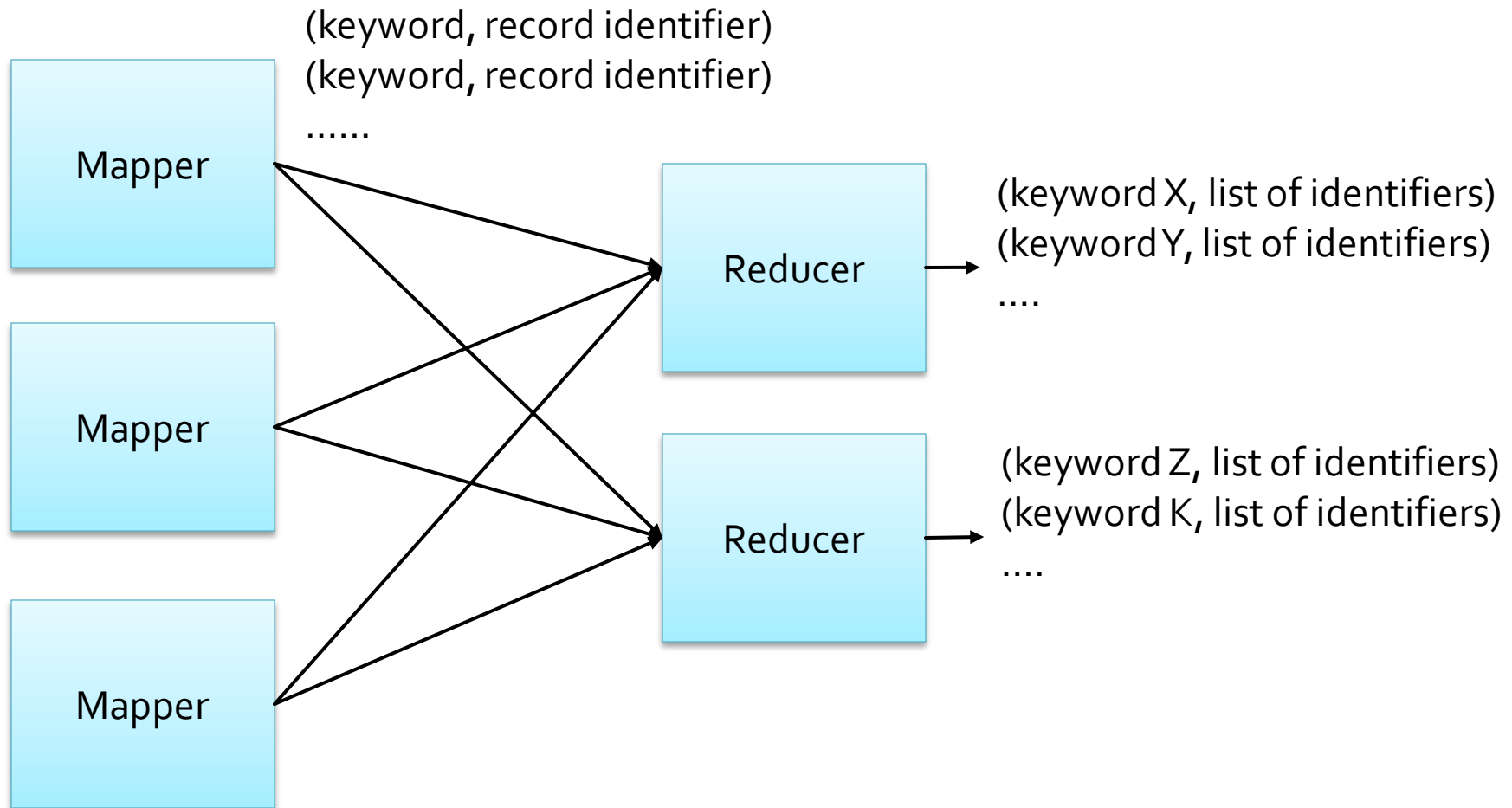
Inverted Index Summarizations

- Goal
 - Build an index from the input data to support faster searches or data enrichment
- Map terms to a list of identifiers
- Motivation
 - Improve search efficiency

Inverted Index Summarizations - structure

- Mappers
 - Output (key, value) pairs where
 - key is the set of fields to index (a keyword)
 - value is a unique identifier of the objects to associate with each "keyword"
- Reducers
 - Receive a set of identifiers for each keyword and simply concatenate them
- Combiners
 - Usually are not useful when using this pattern
 - Usually there are no values to aggregate

Inverted Index Summarizations - structure



Inverted Index Summarizations

- Most famous known use
 - Web search engine
 - Word – List of URLs (Inverted Index)

Summarization Patterns

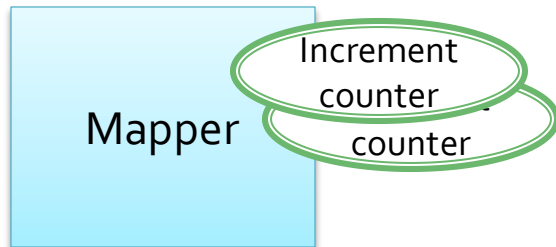
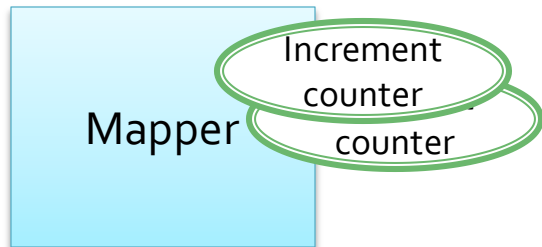
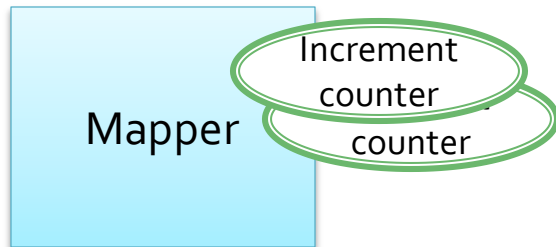
Counting with Counters

- Goal
 - Compute count summarizations of data sets
- Provide a top-level view of large data sets
- Motivation
 - Few high-level statistics can be analyzed by domain experts to identify trends, anomalies, ...

Counting with Counters - structure

- Mappers
 - Process each input record and increment a set of counters
- Map-only job
 - No reducers
 - No combiners
- The results are stored/printed by the Driver of the application

Counting with Counters - structure



Counting with Counters

- Known uses
 - Count number of records
 - Count a small number of unique instances
 - Summarizations

Filtering Patterns

Filtering Patterns

- Are used to select the subset of input records of interest
 - Filtering
 - Top K
 - Distinct

Filtering Patterns

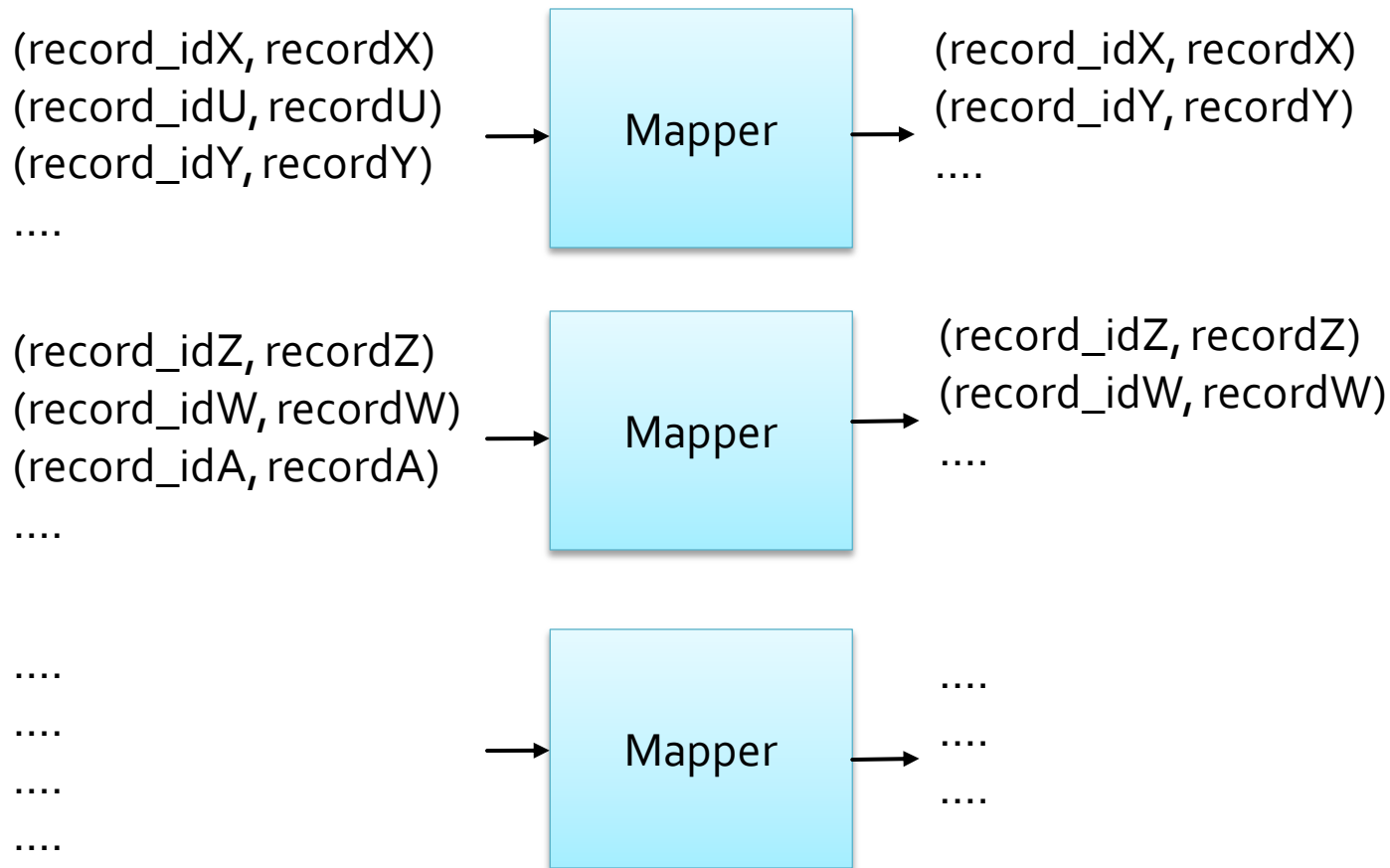
Filtering

- Goal
 - Filter out input records that are not of interest/keep only the ones that are of interest
- Focus the analysis of the records of interest
- Motivation
 - Depending on the goals of your application, frequently only a small subset of the input data is of interest for further analyses

Filtering - structure

- The input of the mapper is a set of records
 - Key = primary key
 - Value = record
- Mappers
 - Output one (key, value) pair for each record that satisfies the enforced filtering rule
 - Key is associated with the primary key of the record
 - Value is associated with the selected record
 - Reducers
 - The reducer is useless in this pattern
 - A map-only job is executed (number of reduce set to 0)

Filtering - structure



Filtering

- Known uses
 - Record filtering
 - Tracking events
 - Distributed grep
 - Data cleaning

Filtering Patterns

Top K

- Goal
 - Select a small set of top K records according to a ranking function
- Focus on the most important records of the input data set
- Motivation
 - Frequently the interesting records are those ranking first according to a ranking function
 - Most profitable items
 - Outliers

Top K - structure

- Mappers
 - Each mapper initializes an in-mapper (local) top k list
 - k is usually small (e.g., 10)
 - The current (local) top k-records of each mapper (i.e., instance of the mapper class) can be stored in main memory
 - Initialization performed in the setup method of the mapper
 - The map function updates the current in-mapper top k list

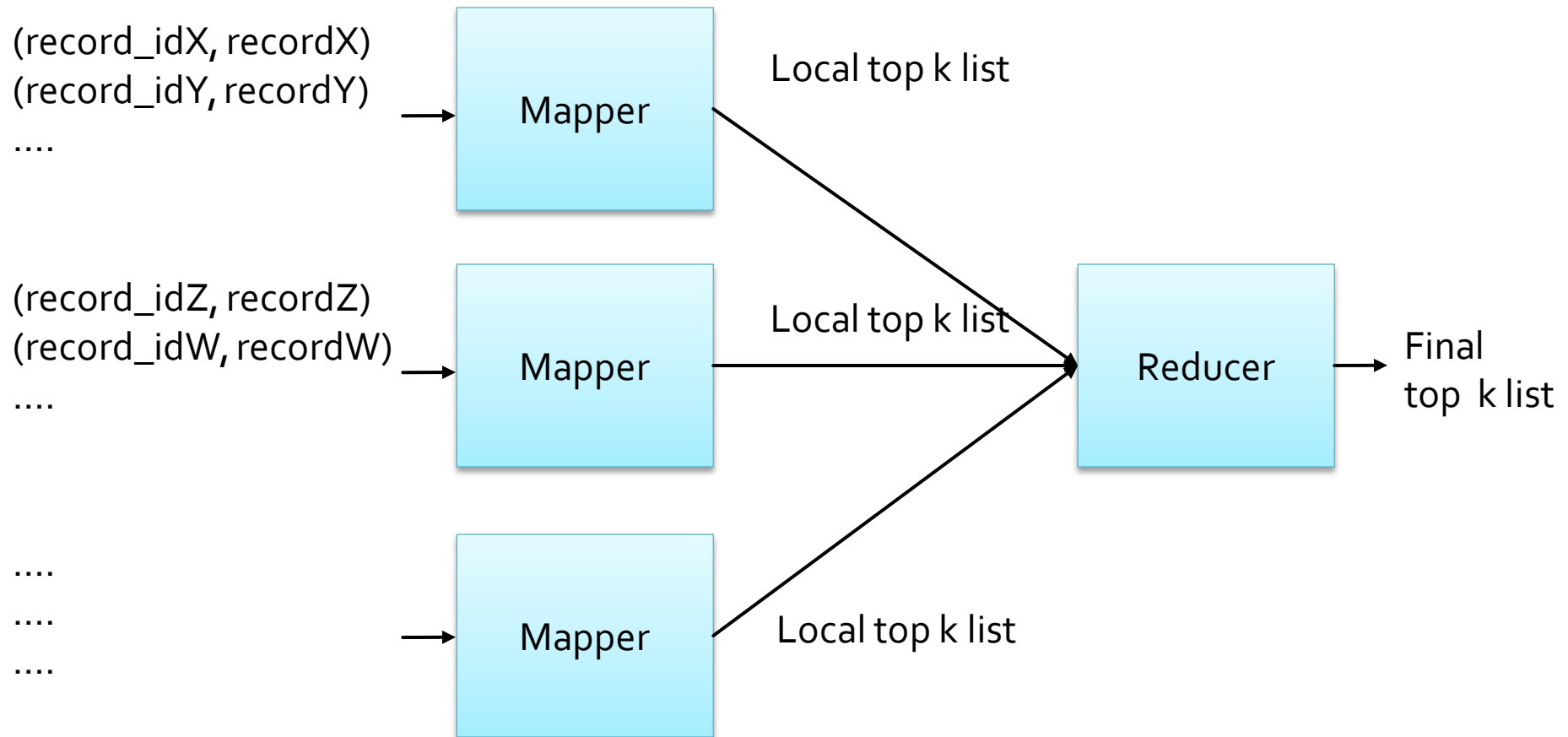
Top K - structure

- Mappers
 - The cleanup method emits the k (key, value) pairs associated with the in-mapper local top k records
 - Key is the “null key”
 - Value is a in-mapper top k record

Top K - structure

- Reducer
 - A single reducer must be instantiated (i.e., **one single instance** of the **reducer** class)
 - One single **global view** over the intermediate results emitted by the mappers to compute the final top k records
 - It computes the final top k list by merging the local lists emitted by the mappers
 - All input (key, value) pairs have the same key
 - Hence, the reduce method is called only once

Top K - structure



Top K

- Known uses
 - Outlier analysis (based on a ranking function)
 - Select interesting data (based on a ranking function)

Filtering Patterns

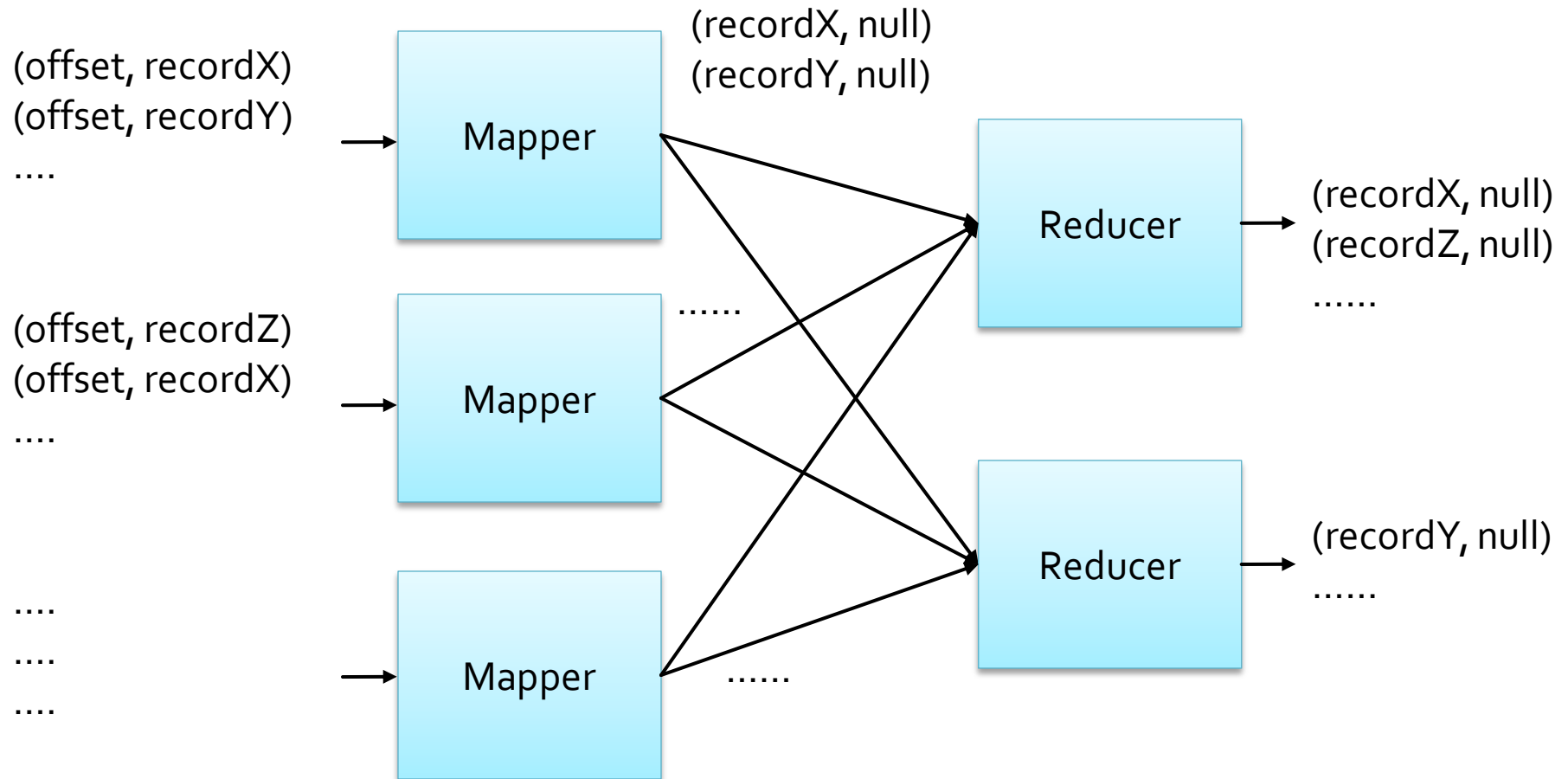
Distinct

- Goal
 - Find a unique set of values/records
- In some applications duplicate records are useless
- Motivation
 - Duplicates records are frequently useless

Distinct - structure

- Mappers
 - Emit one (key, value) pair for each input record
 - Key = input record
 - Value = null value
- Reducers
 - Emit one (key, value) pair for each input (key, list of values) pair
 - Key = input key, i.e., input record
 - Value = null value

Distinct - structure



Distinct

- Known uses
 - Duplicate data removal
 - Distinct value selection