**Unit 1.PME and Virtual MDM Overview**

**What this unit is about**

This unit describes the key concepts behind the InfoSphere MDM Virtual module, including terminology and the basics of matching. This unit is an overview of the topics we will be discussing throughout the course.

**What you should be able to do**

After completing this unit, you should be able to:

•Describe key concepts of Virtual MDM

•Understand Virtual MDM terminology

•Understand how the PME is used by the Virtual MDM

The Virtual module behaves as a registry-style MDM implementation. Matching and linking member records from any number of sources is the main goal of the InfoSphere MDM Virtual module.

The biggest strength of a registry-style implementation is that we can accept data from as many sources that will be supplying data to the implementation. The Virtual module does not change any of the source data, it accepts the data that has been passed to it.

If Virtual MDM find other similar members in the implementation, we can link them together and create a unified, golden view of the member(s) to give you a 360 degree, current view of the member information.

If there is any question that the possibility of matching members together could still be in question, Virtual MDM creates a “task” that will need to be reviewed to help make the final determination if they are the same or not.

With all of the member information in the Virtual modeul, we have the ability to meet your business goals of allowing you to search for members – or members who match and link and create a single golden view.

This information can be used to help your business in ways such as populating a data warehouse, or for business intelligence to make better decisions about your members.

Workbench is a graphic user interface that provides user management and configuration management tools for IBM InfoSphere MDM Virtual module. Simply put, it allows you to view and manage the configuration for Data or Relationships.

Using Workbench, a Virtual data model, algorithm, and thresholds can be easily adjusted to your requirements using a single toolset. Graphical analytics are available to correctly adjust the algorithms and thresholds to increase the accuracy and performance of a particular implementation. These features make it much easier for algorithms to be tuned for performance, and to improve the accuracy of matching based on analytics returned from Workbench.

**Basic functionality**

Workbench Configuration projects can be created and configured without a InfoSphere MDM instance or a data source. This is different from previous versions of configuration tools, like Identity hub Manager, where the engine and databases were written to directly. Instead, Workbench saves the configuration and uploads it to the engine allowing changes to be made “off line”.

Workbench allows users to perform many tasks that were once completed using scripts or multiple MDM software packages. Some of the key functionality includes:

**•**Creating, configuring, and editing member model dictionaries

**•**Creating, configuring, and editing algorithms

**•**Cleaning and de-duplicating data in the data extract

**•**Bucket analysis

**•**Threshold analysis

**•**Entity analysis

**Inspector** is a web-based interface used primarily by Data Stewards and Data Governors. It allows the user to do searching for members and entities. It also is the UI that Data Stewards use to resolve tasks by using drag and drop functionality. You can also use Inspector for Relationships and Hierarchy management. Data Governance can also be accomplished with Inspector when data manipulation is required.

**Pair Manager** is a stand-alone client that is used during the Threshold Analysis stage to allow end-users to look at a sample matched pair of members and determine if they are the same; not the same; or not enough information to determine same or not (basically determining that a tasks should be created for review). It is used to help in determining what the Algorithm thresholds should be set to. It allows the end user to look for and irregularities that may need to have the configuration adjusted to handle. Items such as False Positives.

On the next slides, we will start discussing some of the terminology that is used when discussion the InfoSphere MDM Virtual module.

This list will cover the main items that we will be using.

**Source:** A source is any system that contributes member attributes (records) to the InfoSphere MDM. Sources can add / update member data via any integration point such as the Inbound Message Broker, API, or periodic data feeds. A source will provide the InfoSphere MDM with 1 or more members. For Example, the Sources here include: CRM, Sales, Web, and Data Warehouse.

There are 2 different types of sources.

1. **Definitional source**: this is a source system that provide member records / attributes.

2. **Informational source**: a lookup or referral source that provides unique identifiers to a definitional source. For example, a Social Security Administration creates SSNs and the Department of Motor Vehicles create unique drivers license numbers.

**Entity:** A distinct person, organization, location, etc. that is represented by assigning the same Enterprise Identifier (EID) to one or more member records. An Entity is essentially a link between one or more member records.

In this example Deb Becker-Smith is the Entity (EID 456) that links to her three Member records below.

**Member:** A record that comes from a source which contains current and/or historical information\* Also referred to as a “Record”

A member is made up of 1 or more attributes.

In this example: member (record) 319883 is made up of attributes Name; Work Address; Home Phone; email.

**Attribute:** A demographic data element made up of one or more fields that identifies the traits of a member record

Drilling down into Debbie Beckersmith’s Member record, you can see her attributes: phone, name, DOB, gender, and addressAttributes can be comprised of one or more fields, for example:

**•**Dates are single-field attributes that are stored in string format

**•**Names are multi-field attributes (First-Name, Middle-Name, Last-Name, Title, etc)

**•**Other Attributes, such as Phone (whose Attribute Name is MEMPHONE), can contain multiple Attribute Types. (NEXT SLIDE)

**Attribute Type:** The database segment (or table) in which similar data elements are stored

This slide shows that we have 3 different segments (attribute type)

**•**MEMADDR is used to store address information about members

**•**MEMPHONE is used to store information about member phone numbers

**•**MEMNAME is used to store information about member names

The InfoSphere MDM has two methods for matching records:

1. **Deterministic Matching**: A rules-based process to determine an “exact match” between two records

**-**Works best for simple, easily-defined matches

**-**Skeptical form of matching – Yes or No -> The correlation between the two records must pass the test to be trusted.

**-**InfoSphere MDM uses deterministic matching for relationships and hierarchies

**•**e.g. 2 records with exact same address should be linked as one household

**•**e.g. Employee-to-Supervisor relationships also use deterministic matching

2. **Probabilistic Matching:** A process of using statistical analysis to determine the overall likelihood that two records match

**-**Preferred method for matching large data sets or when a large number of attributes are involved in the matching process

**-**How much Confidence, based contextually on your data, do we have that two records match

**-**Optimistic form of matching – Maybe! -> Looks for any level of correlation between the two records. So for the two mismatched records, the system would say “hey, look at that, you have almost nothing in common, but you both live on a Street and have the middle initial “g”. The score is slightly credited with that correlation, even though it might still be very negative overall.

**-**e.g. When you use a Search Engine to find something on the web, the results will return with an x% that the results match your search criteria.

The Algorithm is a series of processes (Standardization, Bucketing, and Comparison) used to probabilistically match records that results in a comparison score.

An algorithm uses the following steps

1. **Standardization** converts data into its simplest form for easier use during the matching process. In other words, standardization is used to clean up the data by reformatting it into consistent chunks.

2. **Bucketing** organizes records that share common values for faster search retrieval. Think of bucketing this way: is it easier to find a needle in a hay stack or in a jar labeled “needles”?

3. **Comparison** uses the probabilistic method to compare pairs of records and then assign a score based on the similarities and differences between the two records. In other words, the comparison function is a means for finding similarity or difference between two records and their attributes.

For Example:

Using the Phone Number (512) 634-5144, the standardization process removes the non-numeric data [ ()- ] and shaves the numbers down to the last 7 digits ( 6345144)

The Bucketing function (6345144) would be sorted in Ascending order (1344456) and placed in the “1344456” bucket

Comparison: phone numbers 6345144 and 6345414 have an edit distance of one (a simple transposition of the “14” versus a “41” which has a high correlation

So would likely score around 3.9 points

While we have only reviewed this process for Phone Numbers, it is important to note that during the Algorithm’s analysis of the data, it runs parallel analyses of each of the attributes in the records.

The first step of the algorithm is to standardize the data to be used by the algorithm.

This slide is an example of one of the existing standardization functions that is used against address data.

For all standardization functions, 2 things happen – 1st convert all the data to UPPER CASE and 2nd remove any punctuation to cleanse the data.

\*\* note that this is only happening to the data that will be used to create the derived data used by the algorithm – the source data stays intact \*\*

Then finally, using the standardized data – and depending on the standardization function you choose – parse the data elements into individual fields (tokens) that can be used for comparison.

The bottom of the slide is showing you how each of the 4 address provided would look once standardized.

The second step of the algorithm is to create the searchable buckets of the standardized data.

Look at the following five people. What are the things they have in common?

Create buckets based off of the records shared criteria, then note which people would be in the bucket (e.g., people with black hair).

The third step of the algorithm is to do the comparison of the standardized data.

There are several forms of comparison that exists within the engine / algorithm.

It is not strict like a deterministic algorithm where “If first name = JOHN and middle name = G and birth date = 10/23/1960

The probabilistic algorithm allows you to take into consideration things like:

**•**Exact match

**•**Starts with

**•**Edit distance – how many changes need to happen to make the 2 values the same

**•**Phonetics – Stacie vs Stacy

**•**Equivalency – Rob = Bob = Robert

**•**(date of birth) YEAR – does the year match between these to members?

You have the ability to choose from the list of comparison functions when configuring your algorithm to best meet the needs of your customer for this project.

Weights are used to define what the statistical frequency is of this piece of data for this attribute for the dataset.

When you are comparing members, you compare as much data as provided for the 2 members.

You can control how the comparison is done using the different comparison functions so specific weights are used depending on the comparison function. The weight generation process will create weights for those attributes that are determined to be used the most throughout the dataset.

Weights can be positive or negative values.

Here are examples of the 3 weight types that get created.

**Frequency-based Weights:** provide a score based on how often a value appears within the overall data population. Common values (like John) have a low score, rare values (like Chitsumungo) have a high score.

**Edit Distance Weights:** are a measure the similarity between two values. For example, “Gordon” vs. “Gorton” has a distance of 1 edit. Exact match has the highest score, but each edit lowers the score by a certain degree.

**Parameterized (PARM) Weights:** These weights control maximum caps on scores, extra credit points, and penalties for variance. For example, there is a maximum weight for Full Name that ensures that the name does not generate a disproportionate score.

When 2 members are being compared, each attribute comparison will produce a weight determined by which comparison function you have chosen to use for the attribute.

The comparison score is the accumulation of these weights that are determined by the algorithm for each attribute comparison that you have configured.

Then use the comparison score to determine where on the scale these 2 members compare.

**Link:** (v) The act of assigning two Member records to the same Entity either manually (using Inspector) or automatically (if the comparison score crosses the AL Threshold)

Going back to the Probabilistic Matching “maybe continuum,” let’s look at when Member records are automatically or manually assigned to an Entity.

**•**If the two Member records’ comparison score falls at or above the AL Threshold, the records will be automatically linked into the same entity.

**•**If the two Member records’ comparison score fall above the CR Threshold, but below the AL Threshold, a Task will be created.

**•**If the two Member records’ comparison score is below the CR Threshold, these 2 members are not similar at all and would be put into their own entity.

Entity Management is the process that runs in the background to do the member comparison and determine if 2 members should be in the same entity, create a task, or be in their own entity.

Entity Management can be part of the running engine process or as a stand-alone process.

There are 2 threshold that exist within the InfoSphere MDM – Autolink and Clerical Review.

The **Autolink** threshold is the upper limit / score where when 2 members are being compared and their comparison score is equal to or greater than the AL Threshold, these 2 members are considered to be the same and will be assigned the same entity Id (EID).

The **Clerical Review** threshold is the lower limit / score where when 2 members are being compared and their comparison score is less than the CR Threshold, these 2 members do not have enough in common that they are 2 different members. They would then be assigned their own entity ID (EID).

If the comparison score is equal to or greater than the CR Threshold but less than the AL Threshold, a task will be created for these members.

There is the potential where the algorithm is not smart enough or configured correctly to handle every possible scenario.

Certain situations could create a False Positive or a False Negative.

A **False Positive** occurs when 2 members are linked together when they are actually 2 different members.

This is usually the case in situations such as multiple births where same last name, same address, same home phone, etc..

A **False Negative** occurs when 2 members are not linked together when they are actually the same member.

This usually happens when there is thin data being provided by the sources and the algorithm does not have enough information to compare.

There are special comparison functions that exist to help in handling the False Positive since this can cause bigger issues than a False Negative.

This special comparison function is called the False Positive Filter (FPF).

Task: (n) A record, or set of records, that requires human intervention to make a decision about two joining records that fall between the CR & AL thresholds

1. Potential Overlay occurs when a record is updated with information that is radically different than the data that was already there

**-**Example: (customer support)Example: (married)

2. Potential Duplicate occurs when two records are in the same source and appear to represent the same person or organization.

**-**Example: (same person – same source)

**-**Example: (baby or twins – same source)

3. Potential Linkage occurs when two records in different sources and appear to represent the same person or organization.

**-**Example: (same person – 2 sources)

**-**Example: (baby or twins – 2 sources)

4. Review Identifier occurs when two records from the same source seem to be using the same identifier (SSN, Passport, Insurance, or Credit Card Number)

**-**Example: (identity theft)

**-**Example: (household credit card)

Here is the basic setup of our exercises for our course

**•**We will have data from our 2 (definitional) source.

**•**We will be creating a new algorthm for the PME that operates in the MDM Operation Server

**•**We will use a DB2 database that has been created for us.

Attribute definitions

The image that we will work with already has 7 attributes to store the data that is being provided by the (definitional) sources.

Attribute specific definitions also include Number of Active Attributes (Number Active), and Number of Historical attributes (Number Exists).

**•Number of Active Attributes (Number Active)** - Usually the most current attribute value is setup as the “Active” value for an attribute. In this case, the field would be set to 1. Should you have a case where you need more than one active value for the same attribute, increase the number active value.

**-**e.g. Marital status - At any given point in time, a person should only have one active status, such as Married.

**•Number of Historical attributes (Number Exists)** - Over a period of time, an attribute value may change. The number exists value determines how many historical or previous values along with the current active value(s) should be stored in the database.

**-**e.g. if number exists is set to 3 and number active is set to 1 for name and Mary Jones gets married to Jonathan Smith, her name entries may look like the following (oldest to most current):

**-**Mary Jonas - Status=Inactive

**-**Mary Jones - Status=Inactive

**-**Mary Smith - Status=Active

**•**If Mary gets married again, then the very first name or the oldest value would get purged.

**•**In the example above, Mary Jonas is removed and the resulting entries look like the following:

**-**Mary Jones - Status=Inactive

**-**Mary Smith - Status=Inactive

**-**Mary Johnson -Status=Active

Having completed this unit, you should be able to:

**•**Describe key concepts of Virtual MDM

**•**Understand Virtual MDM terminology