CHAPTER 4

ATTRIBUTE SELECTION AND TOKEN FORMATION

4.1 Attribute Selection

A data warehouse can have millions of records and hundreds of columns. The data cleaning process will be complex with this large amount of data in the data warehouse. For example, a dataset may contain 50 columns that describe characteristics of customers, but perhaps only 10 of those columns are used for duplicate detection and identification process. If the unneeded columns are taken while data cleaning process, more CPU and memory are required for the large amount of data. Reducing the dimensionality of the data reduces the time of the data cleaning process and allows algorithms to operate faster and more effectively in the further steps. Therefore time and effort are two important requirements to promptly and qualitatively select the attribute. Attribute selection is very important to reduce time and effort for further works such as record similarity and elimination process.

Table 4a shows sample records and attributes taken from customer dataset. The amount of records and attributes and their relativity is unknown to the users. Attribute selection is very important when comparing two records [KH, 97]. This step is the foundation step for all the remaining steps. The attribute itself may cause inconsistencies and redundancies, due to the use of different names to represent the same attribute or same name for different attributes.
Table 4a: Sample records and attributes

This flow diagram (Figure 4a) shows attribute selection procedure in a sequential way. First, the data set is identified for the data cleaning process. From this data set, attributes are analyzed by identifying types of the attribute, relationship between attributes, and properties of each attribute to select the appropriate attribute. Type of attribute is classified using data type, size or length of the attribute.
Threshold value is used to identify best attributes for the data cleaning process. The threshold value is measured by using three different criteria – High Threshold value, Data Quality and High Rank. High threshold value is calculated to identify high power attribute for the data cleaning process. Attributes are ranked based on the threshold value and data quality value. Finally, high rank attributes are selected for the next cleaning process to improve speed of the data cleaning process.

The user needs to identify the attributes that must be included in the analysis; relationship with the other attributes; types of data and the number of distinct field
values. Based on the above information the user needs to assign a ‘weight’ or ‘rank value’ for the selected attributes. In this research work, software agent is used to reduce the user interaction by identifying best attributes for the data cleaning process. Finally, the highest priority attributes are selected for the next process of data cleaning [JM, 06].

An attribute selection is a process that chooses best attributes according to a certain criterion. This attribute selection algorithm is used to increase the speed and improve the accuracy of the data cleaning process by removing redundant or irrelevant attribute from the data warehouse. There are three criteria used to identify relevant attributes for data cleaning process - Identifying key attributes, classifying attributes with high distinct value and low missing value and measurement types of the attributes.

![Figure 4b: Attribute selection using three main parameters](image)

Figure 4b shows the attribute selection with the above specified parameters. For each attribute, number of distinct value, missing value and type of the attribute is
calculated in data warehouse. Best attributes are selected based on these values for the data cleaning process.

a) Identifying key attributes

A key is an attribute or a set of attributes that uniquely identify a specific instance of the table. Every table in the data model must have a primary key whose values uniquely identify instances of the entity. The qualities of key attributes are:

- must have a non-null value for each instance of the entity
- the value must be unique for each instance of an entity
- the values must not change or become null during the life of each entity instance

This key attributes are directly selected for data cleaning process. The key may be primary key, candidate key, foreign key or composite key. Keys are the properties of the table in the data warehouse. These keys are stored in data warehouse.

b) Classifying distinct and missing values

A missing value is expressed and treated as a string of blanks. Missing character values are always same no matter whether it is expressed as one blank, or more than one blank. Obviously, missing character values are not the smallest strings. Distinct is used to retrieve number of rows that have unique values for each attribute. These two values are very important in data cleaning process. The accuracy of the result will be poor with low distinct value and high missing value. So these two values are required in the data cleaning process.
The distinct value is used to calculate an identification power of the attribute. The distinct value is very much important in an attribute selection. An identification power of the attribute \( (ipj) \) is used to evaluate the discriminating power of record attributes.

\[
\text{Identification power of the attribute } j = \frac{\text{Number of distinct equivalence classes on the total of record}}{\text{Total number of records}}
\]

The quality of data does not adequately characterize attributes candidate as matching key. For example, a record Citizen with fields Surname, Name, Address and Sex. Though the field Sex may have the greater quality value, it would not be appropriate as matching key because it can have only two values, i.e. Male and Female, and thus all records are simply divided into two sets, without having similar records close to each other. The identification power is not sufficient by itself to choose the best key for matching. If the values of an attribute are affected by many errors, the identification power may be high only because of such errors that originate many distinct classes.

c) Classifying types of attributes

The value of measurement types are also considered for the attribute selection. The data cleaning with numeric data will not be effective. For example, salary attribute is not effective to identify duplicate records because this attribute has many
similar values. There are four types of attributes: nominal, ordinal, interval and ratio. The different criteria are given for each attribute types. The categorical data is efficient for the data cleaning process. For example, address attribute is effective to identify duplicate because these attributes have many distinct values and they have enough information for duplicate data detection.

4.1.1 Attribute Selection Algorithm Analysis

An attribute selection algorithm works according to the specified constraints to select the attributes for the data cleaning process. The developed attribute selection algorithm is presented in Figure 4c. The attribute selection algorithm first selects the relation schema R including N attributes. Then it chooses the relation instance (table) r of the relation schema R. Finally selects the attributes Ai (A1,…..,AN) of the relation schema R including N attributes.

```
Input : N-Attributes, no. of tuples n, relation instance r
Output : S – Sub set of the attributes
Initialize : L - Temporary relation instance, x – Attribute set
Var : L - Temporary relation instance, x – Attribute set, F – Field set, i, j
begin
   I. Analyze attribute set X to select best attributes
   II. Calculate threshold value σ for each attribute
       for each attribute xi, i<0,1,2,…..N
       do
           i) Assign threshold value as 1 for key attribute, put into L
           ii) Calculate threshold σ with (σ : D ∩ M ∩ MT ∩ S)
end
```
a. Distinct(D) value of the attribute $x_i$ if tuple $\square_{i=1}^n t = t_i$

b. Missing(M) value of the attribute $x_i$ if tuple $\square_{i=1}^n t_i = \text{NULL}$

c. Measurement types(MT) of attribute (ordinal, nominal, interval, and ratio) $x_i$

Put into $L$.

end

III. Select attribute with high threshold value $\sigma_i$, then put into $L$.

IV. Ignore attribute with low threshold value

V. Calculate data quality $d_{q_j}$ of the selected attributes

for each attribute $x_i$, $i \in \{0,1,2,\ldots,N\}$

for each field $f_j$, $j \in \{0,1,2,\ldots,n\}$

compute

\[
\text{Completeness}_{j} = \frac{\sum_{i=0}^{N} \text{completeness}_{i,j}}{n}
\]

\[
\text{Accuracy}_{j} = \frac{\sum_{i=0}^{N} \text{accuracy}_{i,j}}{n}
\]

\[
\text{Consistency}_{j} = \frac{\sum_{i=0}^{N} \text{consistency}_{i,j}}{n}
\]

\[
d_{q_j} = \frac{\alpha \times \text{Completeness}_{j} + \beta \times \text{Accuracy}_{j} + \gamma \times \text{Consistency}_{j}}{n}
\]

end for

end for

VI. Rank attributes based on data quality and threshold value

VII. Select high rank attributes

End

*Figure 4c: Algorithm for attribute selection*
This attribute selection algorithm obtains the temporary relation schema $L$ with attribute name, type, missing value, distinct value, measurement type and threshold value. Then it reads the attribute $A_i$ from the relation instance $r$ of the relation schema $R$ and puts the name, type and size of the attribute $A_i$ to the temporary relation instance $L$. For each attribute, read the tuples (records) from the selected relation instance $r$ and find the count of missing target values and distinct target values of the attribute $A_i$ and calculate the percentage value. This percentage value of missing values and distinct values are stored in the temporary relation instance $L$.

Finally, the measurement type of the attribute $A_i$ is found and added to the temporary relation instance $L$ for each attribute $A_i$. The threshold values are calculated for every target attribute $A_i$ based on the missing values, distinct values and measurement type and put the threshold values for each attribute in the temporary relation instance $L$. Calculate data quality power for selected attributes $S$ from the temporary relation $L$ based on the threshold values for the next step of the data cleaning process.

The rules are defined according to the distinct value, missing value and types of data for the selection of attributes. The distinct value, missing value and types of data are used to form the rules to select attributes because these three parameters are very important in the data cleaning process. The rules are:

If (distinct value high) $\wedge$ (missing values low) $\wedge$ (type high) then $\sigma = 0.95$

If (distinct value high) $\wedge$ (missing values low) $\wedge$ (type low) then $\sigma = 0.85$
If (distinct value high) ^ (missing values high) ^ (type high) then $\sigma = 0.75$
If (distinct value high) ^ (missing values high) ^ (type low) then $\sigma = 0.65$
If (distinct value low) ^ (missing values low) ^ (type high) then $\sigma = 0.55$
If (distinct value low) ^ (missing values high) ^ (type high) then $\sigma = 0.45$
If (distinct value low) ^ (missing values low) ^ (type low) then $\sigma = 0.35$
If (distinct value low) ^ (missing values high) ^ (type low) then $\sigma = 0.25$

The distinct values and missing values are calculated for each attribute in the data set by using the total number of records. Based on the distinct value and missing value, the priority is given as low and high for the threshold calculation. The measurement type is assigned for each attribute based on the types of data. Here highest priority is assigned for categorical data than to other kind of data because the data cleaning process is easy and effective with categorical data.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Column Size</th>
<th>Data Type</th>
<th>Type Value</th>
<th>Uniques</th>
<th>Not Null</th>
<th>Threshold</th>
<th>Rank</th>
<th>Proceed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer ID</td>
<td>4</td>
<td>System Int32</td>
<td>50 %</td>
<td>100 %</td>
<td>100 %</td>
<td>0.8333</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Customer ID</td>
<td>4</td>
<td>System Int32</td>
<td>50 %</td>
<td>100 %</td>
<td>100 %</td>
<td>0.8333</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Customer Name</td>
<td>40</td>
<td>System String</td>
<td>81 %</td>
<td>81 %</td>
<td>100 %</td>
<td>0.8333</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Contact First Name</td>
<td>30</td>
<td>System String</td>
<td>81 %</td>
<td>81 %</td>
<td>100 %</td>
<td>0.8333</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>Contact Last Name</td>
<td>30</td>
<td>System String</td>
<td>81 %</td>
<td>81 %</td>
<td>100 %</td>
<td>0.8333</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Contact Title</td>
<td>5</td>
<td>System String</td>
<td>00 %</td>
<td>5 %</td>
<td>100 %</td>
<td>0.5157</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>Contact Position</td>
<td>30</td>
<td>System String</td>
<td>00 %</td>
<td>5 %</td>
<td>100 %</td>
<td>0.55</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Last Year Sales</td>
<td>8</td>
<td>System Decimal</td>
<td>80 %</td>
<td>76 %</td>
<td>100 %</td>
<td>0.0837</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Address 1</td>
<td>20</td>
<td>System String</td>
<td>75 %</td>
<td>27 %</td>
<td>47 %</td>
<td>0.53</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Address 2</td>
<td>20</td>
<td>System String</td>
<td>75 %</td>
<td>27 %</td>
<td>47 %</td>
<td>0.53</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>City</td>
<td>30</td>
<td>System String</td>
<td>80 %</td>
<td>27 %</td>
<td>100 %</td>
<td>0.6567</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Region</td>
<td>30</td>
<td>System String</td>
<td>80 %</td>
<td>27 %</td>
<td>100 %</td>
<td>0.6567</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Country</td>
<td>30</td>
<td>System String</td>
<td>80 %</td>
<td>17 %</td>
<td>100 %</td>
<td>0.59</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Postal Code</td>
<td>10</td>
<td>System String</td>
<td>80 %</td>
<td>82 %</td>
<td>100 %</td>
<td>0.8733</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table 4b: Attribute selection with sample data set
Table 4b shows the missing values, distinct values and type of data for each attribute of sample data set to calculate threshold value. Percentage of the distinct values, missing values and measurement types are calculated to find threshold value for each attributes. In this figure, high threshold value attributes are checked to select attributes for the next process of the cleaning process. Figure 4d shows the variance of the threshold for each attribute. The threshold values of attributes are selected between the ranges from 0.9 to 1. Here, the threshold values are calculated for each attribute based on the above three parameters. The threshold values are varied based on the experimental results.

Finally, the high quality attributes are selected for the further data cleaning process. The selected attributes threshold value ranges from 0.9 to 1. The attribute contact name, address, phone and postal code are selected for the next step of the cleaning process. This attribute selection approach demonstrates its efficiency and
effectiveness in dealing with higher dimensionality (thousands of attributes) in the data cleaning process. The attribute selection algorithm can eliminate both irrelevant and redundant attributes and is applicable to any type of data (nominal, numeric, etc.). Also, this attribute selection algorithm can handle different attribute types of data smoothly. The quality of the algorithm is confirmed by applying a set of rules. The main purpose for this attribute selection for data cleaning is to reduce time and to improve the speed for the further data cleaning process such as token formation, record similarity and elimination process in an efficient way.

The token based approach is applied on the selected attribute fields only. The similarity function with a long string input takes more time for the comparison process as well as, it requires multi-pass approach. For example, instead of comparing long string as “Department of Computer Applications, Karunya University” with “Dept of Comp Appl, KU”, token value is calculated for both strings as DCAKU and compared. The token based approach is developed to reduce the time for the comparison process and increase the speed of the data cleaning process.

4.1.2 Estimating the Quality of Attributes

One cannot select attributes without determining the quality of the attributes in some way relevant to the data cleaning process. The quality of an attribute should reflect the data cleaning process. There are two major approaches to estimate the quality of an attribute:
i. The quality of an attribute may be estimated by ignoring the other attributes [JKP, 94]. One goal of attribute selection is to remove all irrelevant features. Let $A$ be the full set of attributes, $A_i$ be a relevant attribute, and $S_i$ be the selected attributes $S_i = A - \{A_i\}$. An attribute $A_i$ is relevant if it satisfies the constraints which are specified to calculate threshold value. Otherwise, attribute $A_i$ is said to be irrelevant. If two attributes $A_i$ and $A_j$ in relation instance $r$ have the same values except for their names, then $A_i$ and $A_j$ are inconsistent attributes. This inconsistent attribute needs to be removed from the attribute selection.

ii. The quality of an attribute may be estimated by selecting the other attributes with the high threshold values ($\sigma$). Most approaches assign a quality measure directly to the attribute but some use indirect measures. For this, a threshold is defined first for the data cleaning process using its record values. The threshold values are calculated for every target attribute $A_i$ based on the missing values, distinct values and measurement type. The threshold value ($\sigma$) should be 0 to 1 ($0 \leq \sigma \leq 1$). The attribute $A_i$ is selected based on the high threshold values ie ($S_i = (\sigma) > 0.85$ [ZH, 07].

The quality of the data in the attribute is calculated based on the overall value of completeness, accuracy and consistency. For each attribute, $j$ is the mean related to the total number of values of such attribute.

\[
\text{Data Quality of the Attribute } j \text{ (d}_{jq}) = \frac{\alpha*\text{Completeness}_j + \beta*\text{Accuracy}_j + \pi*\text{Consistency}_j}{\alpha + \beta + \pi}
\]
The values $\alpha$, $\beta$ and $\pi$ is determined according to the results of the experiments on data matching.

4.2 Token Formation

A token is formed for each highest ranking attribute field. The following steps are carried out before forming the token. The steps are:

i) Remove unimportant tokens

The first step in the token formation is removing the unimportant characters before token formation, to get the smart or the best token for further data cleaning. The unimportant tokens consist of special characters, shortcut forms or ordinal forms, common or stop words, and title or salutation tokens. The common unimportant tokens are listed in Table 4c.

| a. Special characters | ``, ``, `<`, `>`, `-`, `%`, `+`, `(`, `)`, `*`, `-`, `$`, `!`, `|`, `\`, `@`, `:`, `;`, `=`, `?`, `{`, `}`, `~`, and etc |
| b. Title or Salutation | `Rev`, `Dr`, `Mr`, `Miss`, `Master`, `Madam`, `Sir`, `Chief`, `Ms`, `Mister`, `Shri`, `Drs`, `Dres instead of Dr.`., `Dr.`, `Mistress`, `Sis`, `Sri`, `Dear`, `Judge`, `Justice`, `Sister` |
| c. Ordinal forms | `st`, `nd`, `rd`, and `th` |
| d. Common abbreviations | `Pvt`, `Ltd`, `Co`, `Rd`, `St`, `Ave`, `Blk`, `Apt`, `Univ`, `Sch`, `Corp` and etc |
| e. Common words | `and`, `the`, `of`, `it`, `as`, `may`, `than`, `an`, `a`, `off`, `to`, `be`, `or`, `not`, `I`, `about`, `are`, `at`, `by`, `bom`, `de`, `en`, `for`, `from`, `how`, `in`, `is`, `la`, `on`, `that`, `this`, `was`, `what`, `when`, `where`, `who`, `will`, `with` and etc |

*Table 4c: Unimportant characters*
ii) **Expand abbreviations using Reference table**

The use of abbreviation creates problem in token formation. The expansion of abbreviation is important in the token formation. Some common abbreviations are listed in Table 4d. These abbreviations are stored in the log table or reference table. These tables are used as reference tables for the token formation.

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Shortcut</th>
<th>Full form</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Acc. a/c, A/C</td>
<td>account, account current</td>
</tr>
<tr>
<td>2</td>
<td>advt.</td>
<td>Advertisement</td>
</tr>
<tr>
<td>3</td>
<td>Apr.</td>
<td>April</td>
</tr>
<tr>
<td>4</td>
<td>Ave</td>
<td>Avenue</td>
</tr>
<tr>
<td>5</td>
<td>Co.</td>
<td>Company, country</td>
</tr>
<tr>
<td>6</td>
<td>Dept.</td>
<td>Department</td>
</tr>
<tr>
<td>7</td>
<td>Dep.</td>
<td>Departure</td>
</tr>
<tr>
<td>8</td>
<td>Est.</td>
<td>Established, estimated</td>
</tr>
<tr>
<td>9</td>
<td>Gov.</td>
<td>Government, governor</td>
</tr>
<tr>
<td>10</td>
<td>H.O</td>
<td>Head Office</td>
</tr>
<tr>
<td>11</td>
<td>Pvt</td>
<td>Private</td>
</tr>
<tr>
<td>12</td>
<td>Ltd</td>
<td>Limited</td>
</tr>
<tr>
<td>13</td>
<td>Rd</td>
<td>Road</td>
</tr>
<tr>
<td>14</td>
<td>Blk</td>
<td>Block</td>
</tr>
<tr>
<td>15</td>
<td>Apt</td>
<td>Apartment</td>
</tr>
</tbody>
</table>

*Table 4d: Reference Table with sample data*
iii) Formation of Tokens

Different token formation rules are followed for different kinds of data. The data may be numeric, alphanumerical or alphabetic. The rules are given in the algorithm.

a. Numeric Tokens: This numeric token formation rule is suitable for phone number, social security number, street number, apartment number, etc. First, it removes the unimportant characters and converts the character to numeric. Finally, groups the numbers to keep together as one token [ETO, 05].

b. Alphabetic Tokens: This alphabetic token formation rule is well suited for the names such as contact name, customer name, product name, book title, etc. First, it expands the abbreviations and removes the unimportant and stopping characters. Finally, it takes the first character from each word, sorts the selected characters and then groups together all of them into one token [ETO, 05].

c. Alphanumeric Tokens: This alphanumerical token rule is suited for address, product code etc. First, it splits alphanumerical into numeric and alphabetic, sorts out the divided token and then groups numbers and alphabets separately. Finally, the tokens in the field are grouped together to get token as one field [ETO, 05].
Input: Tables with dirty data, Reference table, Selected attributes
Output: LOG table with tokens
Var: i, j, m – attribute set, n – no. of records

begin
For attribute i = 1 to m
    for row j = 1 to n
        do
            i) remove unimportant characters
            ii) expand abbreviations using Reference table
            iii) if row(j) is numeric then
                    a. convert string into number
                    b. sort or arrange the number in order
                    c. form a token, then put into LOG table
                end if
            iv) if row(j) is alphanumeric then
                    a. separate numeric and alphanumeric
                    b. split alphanumeric into numeric and alphabetic
                    c. sort numeric and alphabetic separately
                    d. form a token, then put into LOG table
                end if
            v) if row(j) is alphabetic then
                    a. select the first character from each word
                    b. sort these letters in a specific order
                    c. string them together
                    d. if one word is present, take the first three character as token, then sort the characters
                    e. form a token, then put into LOG table
                end if
        end
    end
end

Figure 4e: Algorithm for Token Formation
Figure 4e shows an algorithm for token formation. In this algorithm, rules are specified to form the tokens. This algorithm works according to the type of the data. For example, if address attribute is selected, alphanumeric rule is used to form the tokens. The formed tokens are stored in LOG table.

Table 4e produces token key for the address field. The alphanumeric token rule is used in this table. First, it splits the alphanumeric into numeric and alphabetic and then it uses alphabetic rule. Finally, it combines together to get token key.

<table>
<thead>
<tr>
<th>Customer Credit ID</th>
<th>Address</th>
<th>Token key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7464 South Kingsway, Sterling Heights</td>
<td>7464SKSH</td>
</tr>
<tr>
<td>2</td>
<td>410 Eighth Avenue, DeKalb</td>
<td>410EAD</td>
</tr>
<tr>
<td>3</td>
<td>7429 Arbutus Boulevard, Blacklick</td>
<td>7429ABB</td>
</tr>
<tr>
<td>4</td>
<td>8287 Scott Road, Huntsville</td>
<td>8287SRH</td>
</tr>
<tr>
<td>5</td>
<td>480 Grant Way, San Diego</td>
<td>480GWSD</td>
</tr>
<tr>
<td>6</td>
<td>1984 Sydney Street, Austin</td>
<td>1984SSA</td>
</tr>
<tr>
<td>7</td>
<td>7655 Mayberry Crescent, Eden Prairie</td>
<td>7655MCE</td>
</tr>
<tr>
<td>8</td>
<td>413 Robson Lane, Winchester</td>
<td>413RLW</td>
</tr>
<tr>
<td>9</td>
<td>1842 St. Anne Place, Concord</td>
<td>1842APC</td>
</tr>
<tr>
<td>10</td>
<td>8404 Nelson Lane, Winchester</td>
<td>8404NLW</td>
</tr>
</tbody>
</table>

*Table 4e: Formation of Tokens for the address field*
iv) Maintaining LOG Table

The proposed token formation algorithm is used to form a token for the selected attributes. The formed tokens are stored in the LOG table. This LOG table is a temporary table to store tokens of the selected attribute field values. The comparison of records is taking place in the LOG Table to find duplicates. The sample LOG table with smart token is described in Table 4f.

<table>
<thead>
<tr>
<th>Customer Credit ID</th>
<th>Contact name key</th>
<th>Customer name key</th>
<th>Address key</th>
<th>Postal key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CC</td>
<td>CC</td>
<td>7464SKSH</td>
<td>48358</td>
</tr>
<tr>
<td>2</td>
<td>CM</td>
<td>PA</td>
<td>410EAD</td>
<td>60148</td>
</tr>
<tr>
<td>3</td>
<td>GJ</td>
<td>AABH</td>
<td>7429ABB</td>
<td>43005</td>
</tr>
<tr>
<td>4</td>
<td>AM</td>
<td>PC</td>
<td>8287SRH</td>
<td>35818</td>
</tr>
<tr>
<td>5</td>
<td>PR</td>
<td>ISW</td>
<td>480GWSD</td>
<td>92150</td>
</tr>
<tr>
<td>6</td>
<td>DH</td>
<td>FJR</td>
<td>1984SSA</td>
<td>78770</td>
</tr>
<tr>
<td>7</td>
<td>GW</td>
<td>HH</td>
<td>7655MCE</td>
<td>55327</td>
</tr>
<tr>
<td>8</td>
<td>BM</td>
<td>CCGS</td>
<td>413RLW</td>
<td>22618</td>
</tr>
<tr>
<td>9</td>
<td>PR</td>
<td>ACC</td>
<td>1842APC</td>
<td>01733</td>
</tr>
<tr>
<td>10</td>
<td>CC</td>
<td>BCT</td>
<td>8404NLW</td>
<td>22616</td>
</tr>
</tbody>
</table>

Table 4f: LOG Table with Smart Tokens

The token formation algorithm is used to form smart tokens for data cleaning and it is suitable for numeric, alphanumeric and alphabetic data. There are three different
rules described for the numeric, alphabetic, and alphanumeric tokens. The result of the
token based data cleaning is to remove duplicate data in an efficient way. Tokens are
formed for selected attribute field values using token formation algorithm. These formed
tokens are stored in the LOG Table 4f. The comparison of entire string is a costly
procedure than comparison of tokens. Therefore, the token formation is very important
to define best and smart token.

The idea behind this algorithm is to define smart tokens from fields of selected
multiple most important attributes by applying simple rules for defining numeric,
alphabetic, and alphanumeric tokens. Temporary table now consists of smart token
records, composed from field tokens of the records. These smart token records are sorted
out using block-token-key.

In this research work, block-token-key is generated by considering more
parameters to improve the quality of data by reducing the false-mismatches and true-
mismatches. The parameters for generating block-token-keys are:

i) High quality power attributes

ii) Sufficient information for duplicate detection

iii) High measurement type values

The result of this process is a sorted token table, which are used to compare
neighboring records for a match. Duplicates are easily detected from these tables. The
notion of “token records” was introduced for recording comparison. Existing algorithms
use only token keys extracted from records for either sorting or blocking (or both).
Block-token-key is explained in the following section.

4.3 Experimental Results

An attribute selection algorithm is implemented to select more important attributes which is having enough information for identifying duplicate records. Selected attributes are used for duplicate record detection. The selected attributes have enough information for duplicate data detection. In this chapter, the results are drawn below with different attribute values, numbers of duplicate records detection and token formation. Customer dataset and Student dataset are used to analyze the performance of attribute selection algorithm and token formation algorithm. Efficiency of duplicate detection and elimination largely depends on the selection of attributes. Token formation algorithm is used to form tokens for selected attribute field values to reduce the time for cleaning planning.
4.3.1 Attribute Selection with parameters

![Figure 4f: Attribute selection in Student Dataset](image)

Wrong selection of attribute affects the performance and accuracy of data cleaning process. Hence key fields are selected in such a way that fields should contain sufficient information to identify the duplication of the record. Figure 4f, 4g shows the attribute selection for data cleaning process. The threshold value is calculated using four
important criteria such as size of the field, missing values, distinct values and measurement type. The best attributes are selected based on the threshold value for the data cleaning process. The name attribute has the highest threshold value compared to other attributes and seven attributes are selected for the next process of data cleaning.

4.3.2 Attribute Vs Duplicates

Figure 4h: Attribute Vs Duplicate detected with varying window size in Student Dataset

Figure 4i: Attribute Vs Duplicate detected with varying window size in Customer Dataset
The identification of duplicates is mainly based on the selection of attributes and selection of window size. In the existing methods, fixed size sliding window is used to minimize the number of comparison. In this method, dynamic window size is used based on the similarities of field values. The best accuracy of the duplicate detection is obtained by using our dynamic method. Figure 4h, 4i shows how the number of duplicate records detected varies as the sliding window size changes and for dynamic size of window. The result of duplicate detection is varied based on the selection of window size and dynamic size. To test this phenomenon, results are taken by varying the attribute values for each execution setting window size between 10 and 50 and dynamic size.

4.3.3 Duplicates Vs No. of Attributes

![Figure 4j: Duplicate detected Vs No. of attribute selected in Student Dataset](image)
<table>
<thead>
<tr>
<th>No. of Columns</th>
<th>Key Columns Selected</th>
<th>No. of duplicate detected</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ADD_ADDRESS1</td>
<td>60988</td>
</tr>
<tr>
<td>2</td>
<td>ADD_ADDRESS1 ADD_NAME</td>
<td>58944</td>
</tr>
<tr>
<td>3</td>
<td>ADD_ADDRESS1 ADD_NAME ADD_PHONE1</td>
<td>50860</td>
</tr>
<tr>
<td>4</td>
<td>ADD_ADDRESS1 ADD_NAME ADD_PHONE1 ADD_CDATE</td>
<td>45128</td>
</tr>
<tr>
<td>5</td>
<td>ADD_ADDRESS1 ADD_NAME ADD_PHONE1 ADD_CDATE ADD_DEL</td>
<td>40156</td>
</tr>
<tr>
<td>6</td>
<td>ADD_ADDRESS1 ADD_NAME ADD_PHONE1 ADD_CDATE ADD_DEL ADD_PARENTTYPE</td>
<td>39160</td>
</tr>
<tr>
<td>7</td>
<td>ADD_ADDRESS1 ADD_NAME ADD_PHONE1 ADD_CDATE ADD_DEL ADD_PARENTTYPE ADD_PINCODE</td>
<td>33172</td>
</tr>
</tbody>
</table>

*Table 4g: Key columns and No. of duplicate detected in Student Dataset*

*Figure 4k: Duplicate detected Vs No. of attribute selected in Customer Dataset*
<table>
<thead>
<tr>
<th>No. of Columns</th>
<th>Key Columns Selected</th>
<th>No. of duplicate detected</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PHONE</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>PHONE FAX</td>
<td>30</td>
</tr>
<tr>
<td>3</td>
<td>PHONE FAX POSTAL_CODE</td>
<td>24</td>
</tr>
<tr>
<td>4</td>
<td>PHONE FAX POSTAL_CODE CUSTOMER_NAME</td>
<td>21</td>
</tr>
<tr>
<td>5</td>
<td>PHONE FAX POSTAL_CODE CUSTOMER_NAME E_MAIL</td>
<td>17</td>
</tr>
<tr>
<td>6</td>
<td>PHONE FAX POSTAL_CODE CUSTOMER_NAME E_MAIL CONTACT_LAST_NAME</td>
<td>17</td>
</tr>
<tr>
<td>7</td>
<td>PHONE FAX POSTAL_CODE CUSTOMER_NAME E_MAIL CONTACT_LAST_NAME CITY</td>
<td>13</td>
</tr>
<tr>
<td>8</td>
<td>PHONE FAX POSTAL_CODE CUSTOMER_NAME E_MAIL CONTACT_LAST_NAME CITY ADDRESS1</td>
<td>11</td>
</tr>
</tbody>
</table>

*Table 4h: Key columns and No. of duplicate detected in Customer Dataset*

Efficiency of the record matching algorithm mainly depends on the selection of the attributes. It has been observed from figure 4j that selection of key column influences the result greatly. Table 4c shows key field selection and approximate duplicate record detected for those selected key fields. The selection of combination of multiple attribute will be more useful to identify exact and inexact duplicates.
4.3.4 Time Vs Token Formation

![Graph showing Time Vs Token Formation](image)

**Figure 41: Time taken Vs token formation, attribute selection with different data size**

In this research work, token formation step takes very low time for token formation. Table loading and attribute selection from the data warehouse also takes few seconds to select best attribute for the data cleaning process (Shown in Figure 41). The time is varied based on the size of the data set.