# Data Mining - Query Language

## Introduction

The Data Mining Query Language was proposed by Han, Fu, Wang, et al for the DBMiner data mining system. The Data Mining Query Language is actually based on Structured Query Language (SQL). Data Mining Query Languages can be designed to support ad hoc and interactive data mining. This DMQL provides commands for specifying primitives. The DMQL can work with databases data warehouses as well. Data Mining Query Language can be used to define data mining tasks. Particularly we examine how to define data warehouse and data marts in Data Mining Query Language.

## Task-Relevant Data Specification Syntax

Here is the syntax of DMQL for specifying the task relevant data:

```
use database database_name,
or
use data warehouse data_warehouse_name
in relevance to att_or_dim_list
from relation(s)/cube(s) [where condition]
order by order_list
group by grouping_list
```

## Specifying Kind of Knowledge Syntax

Here we will discuss the syntax for Characterization, Discrimination, Association, Classification and Prediction.

### CHARACTERIZATION

The syntax for characterization is:

```
mine characteristics [as pattern_name]
  analyze  {measure(s) }
The analyze clause, specifies aggregate measures, such as count, sum, or count%.
For example:
Description describing customer purchasing habits.
mine characteristics as customerPurchasing
analyze count%
```

### DISCRIMINATION

The syntax for Discrimination is:

```
mine comparison [as {pattern_name]}
For {target_class } where  {t arget_condition }
{versus  {contrast_class_i }
where {contrast_condition_i}}
analyze  {measure(s) }
```

For Example, A user may define bigSpenders as customers who purchase items that costs $100 or more on average, and budgetSpenders as customers who purchase items at less than $100 on average. The mining of discriminant descriptions for customers from each of these categories can be specified in DMQL as:

```
mine comparison as purchaseGroups
```

```
for bigSpenders where avg(I.price) ≥$100
versus budgetSpenders where avg(I.price)< $100
analyze count
```

## ASSOCIATION

The syntax for Association is:

```
mine associations [ as {pattern_name} ]
{matching {metapattern} }
```

For Example:

```
mine associations as buyingHabits
matching P(X:customer,W) ^ Q(X,Y) ≥ buys(X,Z)
```

**Note:**Where, X is key of customer relation, P and Q are predicate variables and W,Y and Z are object variables.

## CLASSIFICATION

The syntax for Classification is:

```
mine classification [as pattern_name]
analyze classifying_attribute_or_dimension
```

For Example, To mine patterns classifying customer credit rating where the classes are determined by the attribute credit_rating, mine classification as classifyCustomerCreditRating

```
analyze credit_rating
```

## PREDICTION

The syntax for prediction is:

```
mine prediction [as pattern_name]
analyze prediction_attribute_or_dimension
{set {attribute_or_dimension_i= value_i}}
```

## CONCEPT HIERARCHY SPECIFICATION SYNTAX

To specify what concept hierarchies to use:

```
use hierarchy <hierarchy> for <attribute_or_dimension>
```

We use different syntax to define different type of hierarchies such as:

```
-schema hierarchies
define hierarchy time_hierarchy on date as [date,month quarter,year]
-
set-grouping hierarchies
define hierarchy age_hierarchy for age on customer as
level1: {young, middle_aged, senior} < level0: all
level2: {20, ..., 39} < level1: young
level3: {40, ..., 59} < level1: middle_aged
level4: {60, ..., 89} < level1: senior
-operation-derived hierarchies
```

```
define hierarchy age_hierarchy  for age  on customer  as
{age_category(1), ..., age_category(5)}
:= cluster(default, age, 5) < all(age)
-rule-based hierarchies
define hierarchy profit_margin_hierarchy  on item  as
level_1: low_profit_margin < level_0:  all
if (price - cost)< $50
   level_1:  medium-profit_margin < level_0:  all
if ((price - cost) > $50)  and ((price - cost) ≤ $250))
   level_1:  high_profit_margin < level_0:  all
```

## INTERESTINGNESS MEASURES SPECIFICATION SYNTAX

Interestingness measures and thresholds can be specified by the user with the statement:

```
with <interest_measure_name>  threshold = threshold_value
```

For Example:

```
with support threshold = 0.05
with confidence threshold = 0.7
```

## PATTERN PRESENTATION AND VISUALIZATION SPECIFICATION SYNTAX

We have syntax which allows users to specify the display of discovered patterns in one or more forms.

```
display as <result_form>
```

For Example :

```
display as table
```

# Full Specification of DMQL

As a market manager of a Company , you would like to characterize the buying habits of customers who purchase items priced at no less than $100, w.r.t customer's age, type of item purchased, & place in which item was made. You would like to know the percentage of customers having that characteristic. In particular, you are only interested in purchases made in Canada, & paid for with an American Express("AmEx") credit card. You would like to view the resulting descriptions in the form of a table.

```
use database AllElectronics_db
use hierarchy location_hierarchy for B.address
mine characteristics as customerPurchasing
analyze count%
in relevance to C.age,I.type,I.place_made
from customer C, item I, purchase P, items_sold S,  branch B
where I.item_ID = S.item_ID and P.cust_ID = C.cust_ID and
P.method_paid = "AmEx" and B.address = "Canada" and I.price ≥ 100
with noise threshold = 5%
display as table
```

# Data Mining Languages Standardization

Standardizing the Data Mining Languages will serve the following purposes:

- Systematic Development of Data Mining Solutions.
- Improve interoperability among multiple data mining systems and functions.
- Promote the education.
- Promote use of data mining systems in industry and society.

Source:

http://www.tutorialspoint.com/data_mining/dm_query_language.htm