API Language Specification

PENTANA AUDIT MK 12.0

Table of Contents

Introduction

Pentana Audit includes a new API system. API allows you to extract data stored within the Pentana Audit database using Pentana Audit SQL language (MKSQL).

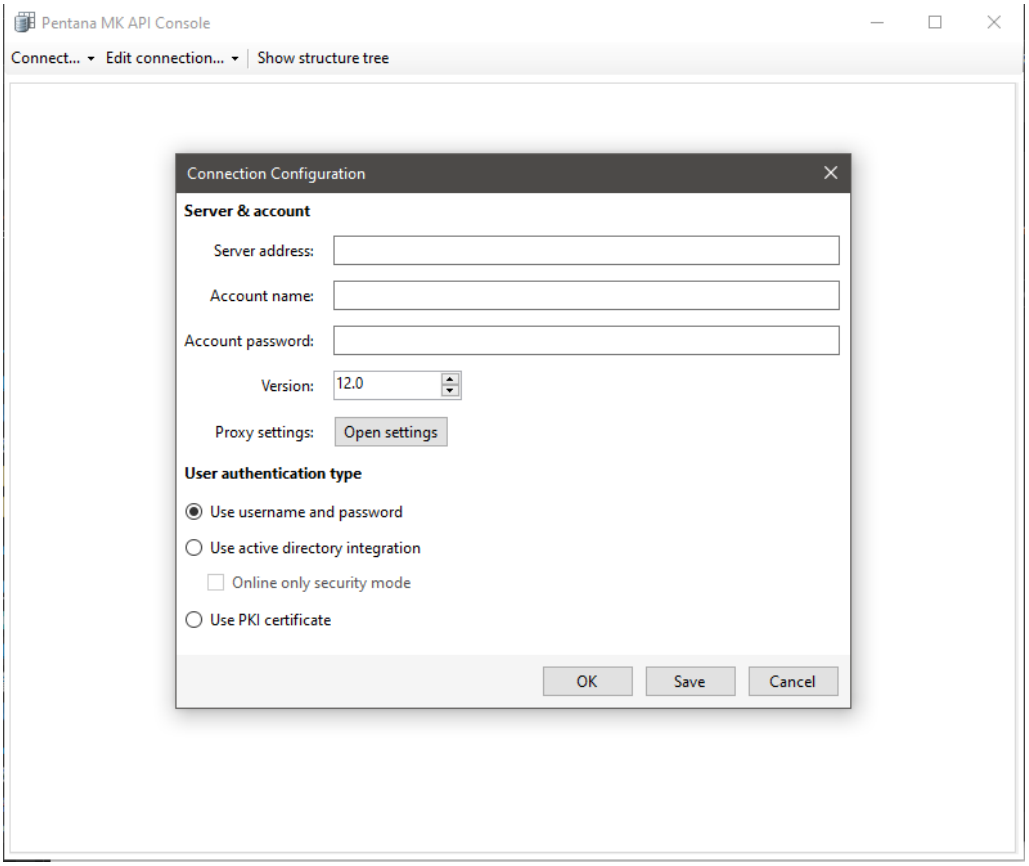
This document includes the language specification of MKSQL along with some example queries.

t : +44 1629 699 100 | e : info@ideagen.com | w : ideagen.com

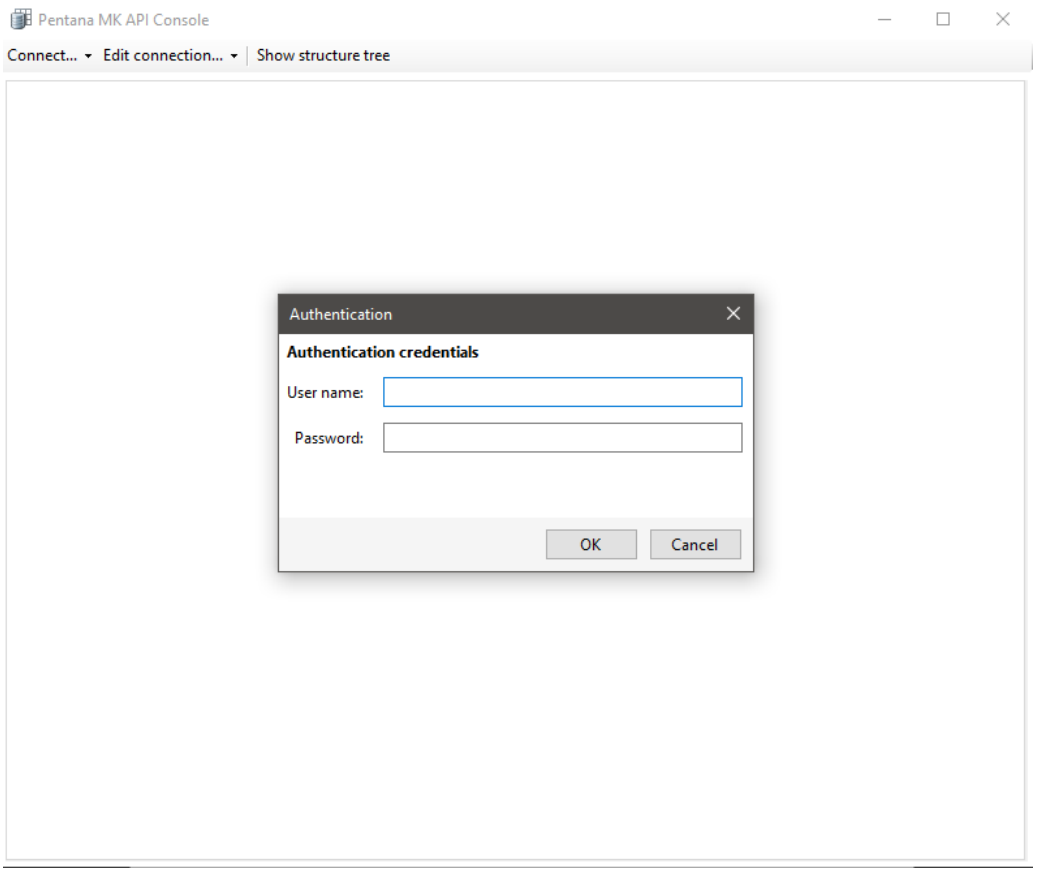
Pentana Audit API tool

Run **MKInsight.API.Console.exe** and select connect > new Connection.

Setup the connection as you would for the Pentana Audit client, click **Save** and add an appropriate name for the connection, e.g. Pentana Audit -Live database and click **OK**.



Enter the *username* and *password* for a configured Pentana Audit user or, you can use Active directory mode if your IT administrator configured this option when Pentana Audit was installed.



Pentana Audit API – Dynamic-link library (dll)

Please see the minimum prerequisite required before using this section:

- You will need to have knowledge of .Net and be familiar with C# and VB.Net. Otherwise please use Pentana Audit API Tool
- .Net 4.6.1
- Visual Studio or Alternative IDE

Create an empty C# project and add MKInsight.API.dll as reference

Standard Authentication

If your Pentana Audit system is not setup to use the standard authentication, skip the code below:

```
var session = Session.Create();
session.Config.ServerUri = new Uri("YOUR_WEB_SERVER_URL");
session.Config.AccountVersion = new Version("12.0");
session.Config.AccountName = "YOUR_ACCOUNT_NAME";
session.Config.AccountPassword = "YOUR_ACCOUNT_PASSWORD";
session.Config.UserAuthenticationType = Config.AuthenticationType.Credentials;
session.Config.AuthenticationUser = "YOUR_USERNAME";
session.Config.AuthenticationPassword = "YOUR_USER_PASSWORD";
session.Authenticate();
```

All strings start with YOUR_* they will need to be replaced with your connectivity settings.

Active Directory

If your Pentana Audit system is not setup to use Active directory, skip the code below:

```
var session = Session.Create();
session.Config.ServerUri = new Uri("YOUR_WEB_SERVER_URL");
session.Config.AccountVersion = new Version("12.0");
session.Config.AccountName = "YOUR_ACCOUNT_NAME";
session.Config.AccountPassword = "YOUR_ACCOUNT_PASSWORD";
session.Config.UserAuthenticationType = Config.AuthenticationType.Integrated;
session.Authenticate();
```

All strings start with YOUR_* they will need to be replaced with your connectivity settings.

Dual Factor Authentication

If your Pentana Audit system is not setup to use dual factor authentication, skip the code below:

Add the code below before `session.Authenticate();`

```
session.Config.AuthenticationPin = "YOUR_PIN_CODE";
```

If you would like to receive an email with your pin code. Add the below statement:

```
session.Config.RequestAuthenticationPin = true;
```

Proxy Settings

If your Pentana Audit system is not setup to use proxy settings, skip the code below:

The code below must be before `session.Authenticate();`

```
session.Config.ProxyAddress = new Uri("YOUR_PROXY_ADDRESS");
session.Config.ProxyLocalBypass = "YOUR_LOCALBY_PASS_YES_OR_NO";
```


an ideagen product
t : +44 1629 699 100 | e : info@ideagen.com | w : ideagen.com

You will need to set the session configuration for the ProxyAuthentication to one of the below statements depending on your current proxy system setup.

- `Config.ProxyAuthenticationType.Anonymous`
  If your proxy setup to use anonymous authentication.
- `Config.ProxyAuthenticationType.Integrated`
  If your proxy setup to use active directory integration authentication.
- `Config.ProxyAuthenticationType.Specified`
  If your proxy setup using username and password authentication you will need to add the below

```
session.Config.ProxyPassword = "YOUR_PASSWORD";
session.Config.ProxyUsername = "YOUR_USERNAME";
session.Config.ProxyDomain = "YOUR_DOMAIN";
```

- `Config.ProxyAuthenticationType.Certificate`
  Internal use only.
- `Config.ProxyAuthenticationType.Service`
  Internal use only.

You will need to set the session configuration for the ProxyHostType to one of the below statements depending on your current proxy system setup.

- `Config.ProxyHostType.Automatic`
- `Config.ProxyHostType.Manual`
- `Config.ProxyHostType.None`

All strings start with YOUR_* and they will need to be replaced with your proxy settings to Pentana Audit system.

Run MKSQL Query

The code to run MKSQL query must be placed after `session.Authenticate();`

```
System.Xml.Linq.XDocument doc = session.ExecuteQuery("Select Id From Audit");
```

Pentana Audit API will return *XDocument* which you can save to the xml document as below:

```
doc.Save(@"c:\temp\doc.xml");
```

For examples of MKSQL query system see section Pentana Audit SQL (MKSQL) & examples

Run MKSQL query with NamedParameters

The code to run MKSQL query must be placed after `session.Authenticate();`

NamedParameters is a collection of variables that can be passed along with MKSQL query and run by the API engine. See the below example:

```
NamedParameters nParams = new NamedParameters();
nParams.Set<int[]>("idList", new int[] { 1, 2, 3, 4, 5 });
nParams.Set<int>("version", 2);
nParams.Set<string>("name", "%test%");
nParams.Set<DateTime>("submitteddate", DateTime.UtcNow);

System.Xml.Linq.XDocument doc = session.ExecuteQuery("SELECT * FROM Audit WHERE Id IN(#idList) AND
Name LIKE #name OR SubmittedDate<#submitteddate OR Version=#version", nParams);
```

Pentana Audit API will return *XDocument* which you can save to the xml document as below:

```
doc.Save(@"c:\temp\doc.xml");
```

Download file attachment

The code to run MKSQL query must be placed after `session.Authenticate();`

To download one single file, you can use the method below:

```
session.DownloadBinaryData(new Guid("d709d788-eb92-4b59-b365-f9e767471482"),
@"c:\temp\testing.xlsx");
```

This query will download the file attachment from the Pentana Audit database to any location you specify.

The first parameter of DownloadBinaryData is the BinaryID of the file attachment within Pentana Audit database. BinaryID field is part of the Attachments->[] navigator.

To download multiple files, you can use the method below:

```
Guid fileBinary1 = new Guid("d709d788-eb92-4b59-b365-f9e767471482");
Guid fileBinary2 = new Guid("f5bb4d2d-2dfc-475d-8dfb-ac2f5da9dc72");

List<Guid> binaryIdList = new List<Guid>() { fileBinary1, fileBinary2 };

Dictionary<Guid, string> fileMapping = new Dictionary<Guid, string>();
fileMapping.Add(fileBinary1, "testing_file1.xlsx");
fileMapping.Add(fileBinary2, "testing_file2.docx");

session.DownloadBinaryData(binaryIdList, @"c:\temp\files", fileMapping);
```

The first parameter is a list of file GUIDs and the second one is the mapping for each file GUID to a filename.

The below query will give you an example of how to find out the BinaryID for an audit file with ID 1

```
SELECT Attachments->[*] FROM Audit WHERE Id=1
```

Download API Structure Documentation

The code to run MKSQL query must be placed after `session.Authenticate();`

```
System.Xml.Linq.XDocument doc = session.DownloadStructureDocumentation();
```

Pentana Audit API will return XDocument containing the entire API fields and navigators which you can save to the xml document as below:

```
doc.Save(@"c:\temp\StructureDocumentation.xml");
```

Advanced methods

- `session.Config.ConnectionTimeout`
  By default, this setting set to 20 seconds.
- `session.Config.ReceiveTimeout`
  By default, this setting set to 5 minutes.
- `session.Config.SendTimeout`
  By default, this setting set to 5 minutes.
- `session.Config.SpecificTempPath`

By default, the system will use user profile temp folder.

- `session.Config.RequireOnlineActiveDirectoryAuthentication`
  This method for use with Pentana Audit setup for online use only.

- `session.AuthenticateAsync`
  Asynchronous method of Authenticate

- `session.ExecuteQuery`
  Asynchronous method of Authenticate

- `session.DownloadStructureDocumentationAsync`
  Asynchronous method of DownloadStructureDocumentation

- `session.DownloadBinaryDataAsync`
  Asynchronous method of DownloadBinaryData

Internal use methods

- `session.Config.ServiceImpersonationId`
- `session.Config.ServicePassword`
- `session.Config.ServiceTokenType`
- `session.Config.AuthenticationCertificate`

Pentana Audit API tool database structure tree

The Pentana Audit database structure consists of:

- Entities
- Fields
- Meta data fields (the meta data fields start with the ! character)

  Some Entity meta data fields start with !1:FieldName.
  ! is for meta data;
  1 or 2 means the first meta data template for number 1, or 2 for the second template;
  FieldName will be replaced with the actual meta data reportable name.

- Entity navigators (Each navigator consists of columns, Meta data, Entity navigators)

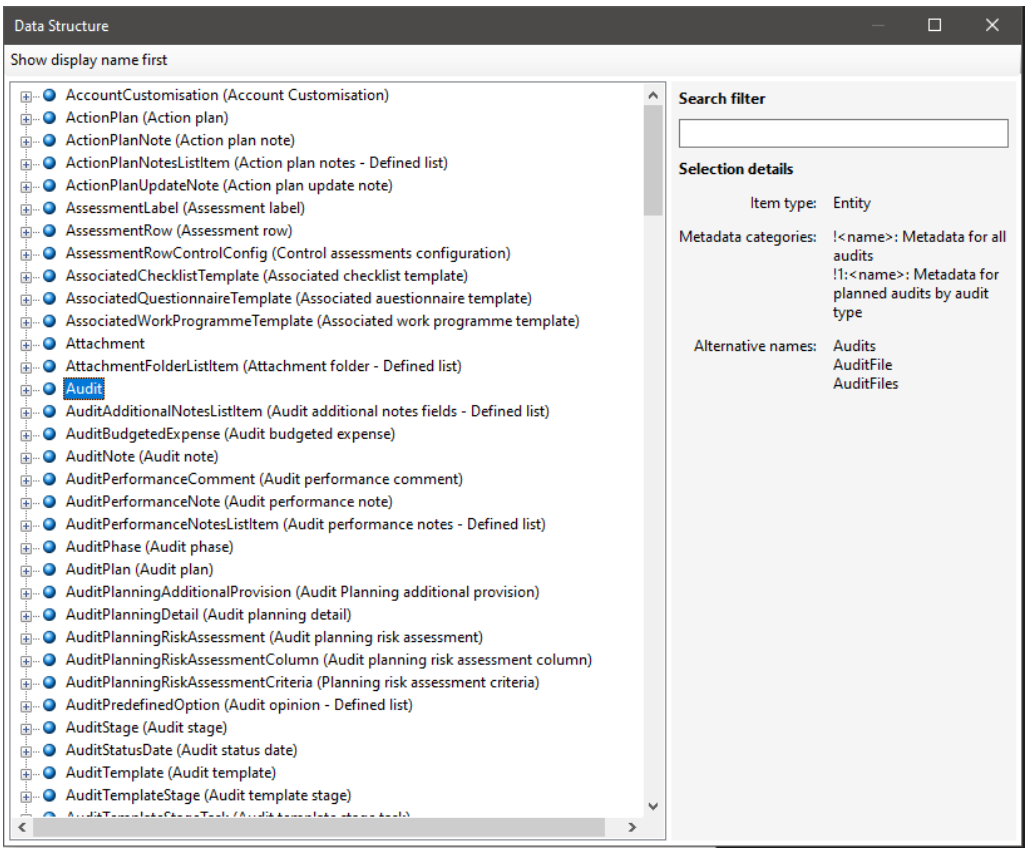
  MKSQL query language removes the need for TSQL join keywords and replaces them with the use of navigators.
  For example, to find the audit type name linked to an audit run the query below:

```
SELECT Id, Name, Type->[name] FROM Audit
```

  Type->[] is a navigator from the Audit entity.

You can drag and drop from the data structure to the query field (Text Box).

Click on "**Show structure tree"** to see the Pentana Audit database fields and navigator tree.



Pentana Audit SQL (MKSQL) & examples

MKSQL language is not TSQL, however there are similarities. Please see the examples below to help you understand how to use MKSQL language.

See Appendix 2 for the MKSQL technical specification.

Select statement

Once you have written the query press F5 or Execute query button.

Example 1:

```
SELECT Id, Name FROM Audit
```

Example 2:

```
SELECT Firstname, Surname FROM User
```

Example 3: (Use of TOP keyword)

```
SELECT TOP 10 Id, Name FROM Audit
```

Example 4: (User of ORDER keyword)

```
SELECT Id , Name FROM Audit ORDER BY Name ASC
SELECT Id , Name FROM Audit ORDER BY Name DESC
```

Example 5: (Use of AS keyword)

```
SELECT Id, Name, SubmittedDate AS SubmittedDate FROM Audit
```

Example 6: (Use MKSQL function – DATEADD)

See Appendix 1 for MKSQL functions

```
SELECT Id, DATEADD (mm, 1, SubmittedDate) AS NewSubmittedDate FROM Audit
SELECT Id , Name, GETDATE() AS TodayDate FROM Audit
```

Select statement using navigator

Example 7:

```
SELECT Id, Name, Type->[Name] FROM Audit
```

This query will return audit id, name and the audit type name.

Example 8:

```
SELECT Id, Checklists->[Name ORDER By Name DESC] FROM Audit
```

This query will return audit id and all checklist template names linked for each audit sorted by name DESC order.

Example 9:

```
SELECT Id, Checklists->[Name, Items->[Name]] FROM Audit
```

This query will return audit id's and all checklist template names and each checklist item for each audit.

Where statement

Example 10:

```
SELECT Id, Name FROM Audit WHERE ID >1
```

Example 11:

```
SELECT Id, Name FROM Audit WHERE Name IS NOT NULL
```

Example 12:

```
SELECT Id, Name FROM Audit WHERE Name IS NULL
```

Example 13:

```
SELECT Id, Name FROM Audit WHERE ID IN (1,2,3)
```

Example 14:

```
SELECT Id, Name FROM Audit WHERE Name LIKE '%audit%'
```

Example 15:

```
SELECT Id, Name FROM Audit WHERE (ID>10 AND Name IS NOT NULL) OR (ID IN
      (2,3,4))
```

Example 16:

```
SELECT Id, Name FROM Audit WHERE SubmittedDate Between DATEADD(m,-12
      ,GETDATE()) AND GETDATE()
```

This query will return all audit id's and names when the submitted date is between now and last year.

Where statement using navigator

Example 17:

```
SELECT Id, Name, Type->[Name] FROM Audit WHERE Type->[Name=' Compliance']
```

This query will return all audit id's, names and audit type names that contain the type called Compliance.

Example 18:

```
SELECT Id, Name, Type->[Name] FROM Audit WHERE Type->[]
```

This query will return all audit id's, names and audit type names that have type links.

Example 19:

```
SELECT Id, Name, Checklists->[Name, Items->[Name]] FROM Audit  WHERE
      Checklists->[Items->[Done=1]]
```

This query will return audit id, name, checklist template name and all checklist template items for each audit where all the items in the checklist have been ticked.

Example 20:

```
SELECT Id, Name, Type->[Name WHERE Name='Compliance'] FROM Audit WHERE
      Type->[Name IS NOT NULL]
```

This query will return the audit id, name and audit type name where the type name is called Compliance and the type must not be null.

Example 21: (Use of Aggregate Functions):

See Appendix 2 for MKSQL Aggregate Functions.

You can only use Aggregate Functions on the where side of the query.

```
SELECT Id, Name, Checklists->[Name, Items->[Name]] FROM Audit  WHERE
      Checklists->[items->[COUNT(* WHERE done=1)>3]]
```

This query will return audit id, name, checklist template name and all checklist template items for each audit where all checklist items have been ticked with a minimum of three items.

Example 22:

```
SELECT Id, Name, Score->[value] FROM Audit
      WHERE Score->[AVG(Value)>0.3]
```

This query will return all audit id's, names and audit score values if each audit score value average is above 0.3.

Advanced MKSQL Queries

Example 23:

```
SELECT Id, Name, Period, WorkProgrammes->[ApprovedBy->[DisplayName],
      Risks->[Name],
Controls->[Name], Tests->[Name]]
FROM Audit
WHERE WorkProgrammes->[COUNT(ID)>1]
```

This query will return audit id, name, period and all work programmes for each audit, with the approved by display name, risk name, control name and test name where there is at least one work programme for each audit.

Example 24:

```
SELECT ID,
      LockedItems->[EntryTime,
            LockedAudit->[ID],
            LockedChecklistTemplates->[ID]
      ],
      LockedReportTemplates->[ID],
      LockedTimesheets->[ID]
FROM User
```

This query will return the id and the user display name for locked work programmes, locked audits, locked work programme templates and locked timesheets.

Example 25:

```
SELECT Id, Name, Period, Version, StateName,
            AdditionalNotes->[*],
            PerformanceNotes->[* WHERE Notes IS NOT NULL],
            Users->[*, User->[ID, DisplayName], Role->[ID, Name]
            WHERE IsBusinessUser = 0],
            KeyDates->[Date],
            EntityUniverseLinks->[ParentBranch->[*]],
            RecordedTimeEntity->
                        [Time->[Date, Hours, User->[ID]],
                        AuditPhase->[ID],
                        Notes->[Date, Days, Expenses->[Date, Value, Code-
      >[ID]]]
 ]
```

```
FROM Audit
```

This query will return audit id, name, audit period, version, audit state, all additional notes, all performance notes where notes is not empty, all users apart from business users, key date dates, entity universe links, recorded time for each audit with notes details such as date, days and expenses date value and code id.

Example 26:

```
SELECT Id, Name, State,
Effectiveness->[LogDate, Comments WHERE LogDate <= GETDATE()]
FROM ManagedControl
WHERE state = %active
```

This query will return the id, name and state from the effectiveness log date and comments for each managed control where the log date is less than or equal to today's date and the state of the control is active.

Appendix 1 – MKSQL functions

MKSQL functions have a similar usage to that of TSQL functions. Supported functions are below:

Help about how to use each function can be found on MSDN.

- GETDATE()
- GETUTCDATE()
- DATEADD (*datepart, number, date)
- DATEDIFF (*datepart, startdate, enddate)
- LEFT (character_expression , integer_expression)
- LEN (character_expression)
- CHARINDEX (expressionToFind , expressionToSearch [, start_location])
- LTRIM (character_expression)
- REPLACE (string_expression, string_pattern , string_replacement)
- RTRIM (character_expression)
- SUBSTRING (expression ,start , length)

***datepart**
This is the part of the date to which an integer is added. The following table lists all valid datepart arguments. User-defined variable equivalents are not valid.

datepart	Abbreviations
year	yy, yyyy
quarter	qq, q
month	mm, m
dayofyear	dy, y
day	dd, d
week	wk, ww
hour	hh
minute	mi, n
second	ss, s

t : +44 1629 699 100 | e : info@ideagen.com | w : ideagen.com

Appendix 2 – MKSQL Aggregate Functions

MKSQL Aggregate Functions have a similar usage to that of TSQL Aggregate Functions. Supported Aggregate Functions are below:

Help about how to use each Aggregate Function can be found on MSDN.

- AVG
- SUM
- MIN
- MAX
- COUNT
- VAR
- VARP
- STDEV
- STDEVP
- ALL

t : +44 1629 699 100 | e : info@ideagen.com | w : ideagen.com

Appendix 3 – MKSQL specifications

```
grammar MKSql;

// rules for statements
select_statements:          GO* select_statement (GO+ select_statement?)* EOF;
select_statement:           SELECT (TOP top_count)? select_list FROM basic_name (WHERE search_condition)? (ORDER BY order_list)?;
security_condition:         search_condition EOF;

// basic rules for part identification
id:                         ID ('.' ID)?;
meta_variant_id:            INTEGER;
instance:                   INTEGER;
top_count:                  INTEGER;
basic_name:                 id;
meta_name:                  META_FIELD (meta_variant_id META_VARIANT_POSTFIX)? id;
all_field_names:            SELECT_ALL;
parameter_name:             PARAMETER ID;

// scalars
constant:                   STRING | NUMBER | INTEGER | ENUM;
constant_list:              constant (',' constant)*;
scalar:                     constant
                            | basic_name
                            | meta_name
                            | sql_function
                            | parameter_name
                            ;


// rules for select list elements
select_navigator:           (basic_name | meta_name) NAVIGATE '[' (TOP top_count)? select_list (WHERE search_condition)? (ORDER BY order_list)? ']';
select_list:                select_list_elem (',' select_list_elem)*;
select_list_elem:           all_field_names
                            | scalar (AS id)?
                            | select_navigator;


// rules for ordering
order_list_elem:            (basic_name | meta_name) dir=(ASC | DESC)?;
order_list:                 order_list_elem (',' order_list_elem)*;
```


t : +44 1629 699 100 | e : info@ideagen.com | w : ideagen.com

```
// rules for where search conditions
comparison_operator:        '=' | '>' | '<' | '<' '=' | '>' '=' | '<' '>';
aggregrate_func:            AVG | SUM | MIN | MAX | COUNT | VAR | VARP | STDEV | STDEVP | ALL;
search_condition:           search_condition_and (OR search_condition_and)*;
search_condition_and:       search_condition_not (AND search_condition_not)*;
search_condition_not:       NOT? predicate;
predicate:                  scalar comparison_operator scalar
                            | scalar NOT? BETWEEN scalar AND scalar
                            | scalar NOT? IN '(' (constant_list | parameter_name) ')'
                            | scalar NOT? LIKE scalar
                            | scalar IS NOT? NULL
                            | aggregrate_func '(' (scalar | all_field_names) (WHERE search_condition)? ')' comparison_operator cmpTo=scalar
                            | (basic_name | meta_name) NAVIGATE '[' search_condition? ']'
                            | basic_name NAVIGATE NULL
                            | '(' search_condition ')'
                            ;

// sql functions
sql_function:               fn=GETDATE '(' ')'
                            | fn=GETUTCDATE '(' ')'
                            | ('-')? fn=DATEDIFF '(' date_part ',' scalar ',' scalar ')'
                            | fn=DATEADD '(' date_part ',' scalar ',' scalar ')'
                            | fn=LEFT '(' scalar ',' scalar ')'
                            | ('-')? fn=LEN '(' scalar ')'
                            | ('-')? fn=CHARINDEX '(' scalar ',' scalar (',' scalar)? ')'
                            | fn=LTRIM '(' scalar ')'
                            | fn=REPLACE '(' scalar ',' scalar ',' scalar ')'
                            | fn=RTRIM '(' scalar ')'
                            | fn=SUBSTRING '(' scalar ',' scalar ',' scalar ')'
                            ;

date_part:                  'yy' | 'yyyy' | 'qq' | 'q' | 'mm' | 'm' | 'dy' | 'y' | 'dd' | 'd' | 'wk' | 'ww' | 'hh' | 'mi' | 'n' | 'ss' | 's';

// lexer
NAVIGATE:                   '->';
META_FIELD:                 '!';
PARAMETER:                  '#';
META_VARIANT_POSTFIX:       ':';
SELECT_ALL:                 '*';

ID:                         ([a-zA-Z] | '@' | FullWidthLetter) ([a-zA-Z_0-9] | FullWidthLetter)*;

STRING:                     '\'' (~'\'' | '\'\'')* '\'';
INTEGER:                    DEC_DIGIT+;
NUMBER:                     '-'? (DEC_DIGIT+ | DEC_DOT_DEC);
ENUM:                       '%' [a-zA-Z]+;
SPACE:                      [ \t\r\n]+    -> skip;
COMMENT:                    '/*' .*? '*/' -> channel(HIDDEN);
LINE_COMMENT:               '--' ~[\r\n]* -> channel(HIDDEN);

fragment DEC_DOT_DEC:       (DEC_DIGIT+ '.' DEC_DIGIT+ |  DEC_DIGIT+ '.' | '.' DEC_DIGIT+);
fragment DEC_DIGIT:         [0-9];
```


t : +44 1629 699 100 | e : info@ideagen.com | w : ideagen.com