199

200

Chapter 6

OBTAINING METADATA

6. OBTAINING METADATA

201

Learning Objectives

This chapter introduces **Obtaining Metadata** in **MySQL**. In this chapter, you will learn to:

- List the various available methods to access metadata
- The `INFORMATION_SCHEMA` database and its contents
- The `SHOW` statement
- The `DESCRIBE` statement
- The `mysqlshow` client program
- The differences between `SHOW` statements and `INFORMATION_SCHEMA` tables

6.1. Metadata Access Methods

202

Databases contain data, but information about the way databases are structured is metadata. This chapter discusses the various means by which MySQL provides access to metadata for database, tables, and other objects. It covers the following topics:

- Using the `INFORMATION_SCHEMA` database to access metadata
- Using `SHOW` and `DESCRIBE` statements to access metadata
- Using the `mysqlshow` program to access metadata

MySQL produces metadata for several aspects of database structure. To name a few, you can obtain names of databases and tables, information about columns and indexes in tables, or stored routine definitions.

One method by which MySQL makes metadata available is through a family of `SHOW` statements, each of which displays one kind of information. For example, `SHOW DATABASES` and `SHOW TABLES` return lists of database and table names, and `SHOW COLUMNS` produces information about definitions of columns in a table.

A client program, `mysqlshow`, acts as a command-line front end to a few of the `SHOW` statements. When invoked, it examines its arguments to determine what information to display, issues the appropriate `SHOW` statement, and displays the results that the statement returns.

`SHOW` and `mysqlshow` have been available since very early releases of MySQL. As of MySQL 5.0, metadata access is enhanced through the following addition:

- The `INFORMATION_SCHEMA` database is implemented. This provides better compliance with standard SQL because `INFORMATION_SCHEMA` is standard, not a MySQL-specific extension like `SHOW`.

203

6.2. The INFORMATION_SCHEMA Database/Schema

The **INFORMATION_SCHEMA** database is a central collection of data about the server and its databases. It is a "virtual database" in the sense that it is not stored on disk anywhere, yet it appears to contain tables just like any other database, and the contents of its tables can be accessed using **SELECT** like with any other tables. Furthermore, you can use **SELECT** to obtain information about **INFORMATION_SCHEMA** itself. For example, to list the names of its tables use the following statement:

204

```
mysql> SELECT   TABLE_NAME
    -> FROM     INFORMATION_SCHEMA.TABLES
    -> WHERE    TABLE_SCHEMA = 'information_schema'
    -> ORDER BY TABLE_NAME;
+---------------------------------------+
| TABLE_NAME                            |
+---------------------------------------+
| CHARACTER_SETS                        |
| COLLATIONS                            |
| COLLATION_CHARACTER_SET_APPLICABILITY |
| COLUMNS                               |
| COLUMN_PRIVILEGES                     |
| ENGINES                               |
| EVENTS                                |
| FILES                                 |
| GLOBAL_STATUS                         |
| GLOBAL_VARIABLES                      |
| KEY_COLUMN_USAGE                      |
| PARTITIONS                            |
| PLUGINS                               |
| PROCESSLIST                           |
| REFERENTIAL_CONSTRAINTS               |
| ROUTINES                              |
| SCHEMATA                              |
| SCHEMA_PRIVILEGES                     |
| SESSION_STATUS                        |
| SESSION_VARIABLES                     |
| STATISTICS                            |
| TABLES                                |
| TABLE_CONSTRAINTS                     |
| TABLE_PRIVILEGES                      |
| TRIGGERS                              |
| USER_PRIVILEGES                       |
| VIEWS                                 |
+---------------------------------------+
```

205

6.2.1. INFORMATION_SCHEMA Tables:

The tables shown in that list contain the following types of information:

- CHARACTER_SETS -- Information about available character sets
- COLLATIONS -- Information about collations for each character set
- COLLATION_CHARACTER_SET_APPLICABILITY -- Information about which character set applies to each collation
- COLUMNS -- Information about columns in tables
- COLUMN_PRIVILEGES -- Information about column privileges held by MySQL user accounts
- ENGINES -- Information about storage engines
- EVENTS -- Information about scheduled events
- FILES -- Information about the files in which MySQL NDB Disk Data tables are stored
- GLOBAL STATUS/VARIABLES -- Information about global server status variables
- KEY_COLUMN_USAGE -- Information about constraints on key columns
- PARTITIONS -- Information about table partitions
- PLUGINS -- Information about server plugins
- PROCESSLIST -- Information about which threads are running
- REFERENTIAL_CONSTRAINTS -- Information about foreign keys
- ROUTINES -- Information about stored procedures and functions
- SCHEMATA -- Information about databases
- SCHEMA_PRIVILEGES -- Information about database privileges held by MySQL user accounts
- SESSION STATUS/VARIABLES -- Information about server status variables
- STATISTICS -- Information about table indexes
- TABLES -- Information about tables in databases
- TABLE_CONSTRAINTS -- Information about constraints on tables
- TABLE_PRIVILEGES -- Information about table privileges held by MySQL user accounts
- TRIGGERS -- Information about triggers in databases
- USER_PRIVILEGES -- Information about global privileges held by MySQL user accounts
- VIEWS -- Information about views in databases

206

6.2.2. Displaying INFORMATION_SCHEMA Tables

To display the names of the columns in a given **INFORMATION_SCHEMA** table, use a statement of the following form, where the **TABLE_NAME** value specifies the table in which you are interested:

```
mysql> SELECT COLUMN_NAME
       FROM   INFORMATION_SCHEMA.COLUMNS
    -> WHERE  TABLE_SCHEMA = 'INFORMATION_SCHEMA'
    -> AND    TABLE_NAME = 'VIEWS';
+-----------------+
| COLUMN_NAME     |
+-----------------+
| TABLE_CATALOG   |
| TABLE_SCHEMA    |
| TABLE_NAME      |
| VIEW_DEFINITION |
| CHECK_OPTION    |
| IS_UPDATABLE    |
+-----------------+
```

Note: The names of the **INFORMATION_SCHEMA** database, its tables, and columns are not case sensitive.

When you retrieve metadata from **INFORMATION_SCHEMA** by using **SELECT** statements, you have the freedom to use any of the usual **SELECT** features:

- You can specify in the select list which columns to retrieve.
- You can restrict which rows to retrieve by specifying conditions in a **WHERE** clause.
- You can group the results with **GROUP BY** or sort them using **ORDER BY**.
- You can use joins, unions, and subqueries.
- You can retrieve the result of an **INFORMATION_SCHEMA** query into another table with **CREATE TABLE … SELECT** or **INSERT … SELECT**. This enables you to save the result and use it in other statements later.
- You can create views on top of **INFORMATION_SCHEMA** tables

207 The following examples demonstrate how to exploit various features of **SELECT** to pull out information in different ways from INFORMATION_SCHEMA:

- Display the storage engines used for the tables in the default database:

```
SELECT TABLE_NAME, ENGINE
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_SCHEMA = SCHEMA();
```

- Find all the tables that contain columns that have the **float** data type:

```
SELECT TABLE_SCHEMA, TABLE_NAME, COLUMN_NAME
FROM INFORMATION_SCHEMA.COLUMNS
WHERE DATA_TYPE = 'float';
```

- Display the default collation for each character set:

```
SELECT CHARACTER_SET_NAME, COLLATION_NAME
FROM INFORMATION_SCHEMA.COLLATIONS
WHERE IS_DEFAULT = 'Yes';
```

- Display the number of tables in each database:

```
SELECT TABLE_SCHEMA, COUNT(*)
FROM INFORMATION_SCHEMA.TABLES;
GROUP BY TABLE_SCHEMA;
```

208 The INFORMATION_SCHEMA is read-only. Its tables cannot be modified with statements such as INSERT, DELETE, or UPDATE. If you try, an error occurs:

```
mysql> DELETE FROM INFORMATION_SCHEMA.VIEWS;
ERROR 1288 (HY000): The target table VIEWS of the DELETE is not
updatable
```

The following are some examples of "real world" uses of INFORMATION_SCHEMA:

- Check data size of each database:

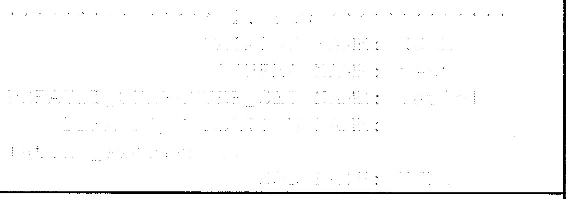
```
mysql> SELECT TABLE_SCHEMA, SUM(data_length+index_length)
    -> FROM INFORMATION_SCHEMA.TABLES GROUP BY TABLE_SCHEMA;
+--------------------+-------------------------------+
| table_schema       | sum(data_length+index_length) |
+--------------------+-------------------------------+
| information_schema |                          8192 |
| mysql              |                        446608 |
| world              |                        691507 |
+--------------------+-------------------------------+
```

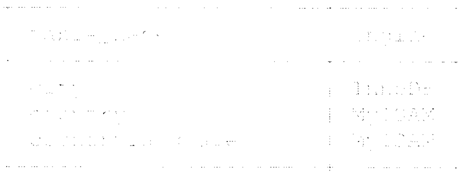
- Check user accounts and privileges (e.g., make sure every user has a password).
- Give a sum of each table.

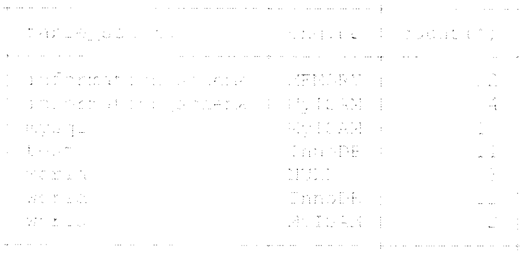
InLine Lab 5-A

In this exercise you will use the methods covered in this chapter for **Obtaining Metadata**. This will require a MySQL command line client and access to the mysql server and use of the **INFORMATION_SCHEMA** database.

ACTION (You Do)	COMPUTER RESPONSE / Comment

1. Use the **SELECT** command to obtain schemata information about the '**test**' schema:
`SELECT * FROM INFORMATION_SCHEMA.SCHEMATA`
`WHERE SCHEMA_NAME= 'test'\G`
	Returns a vertical chart of the **test** database specifications;
	

2. Change the database to the **INFORMATION_SCHEMA** database: `USE INFORMATION_SCHEMA;`	Shows that database has changed.
3. *Select* the table *name* and *engine* from the **tables** table for the **world** *schema*: `SELECT table_name, engine FROM tables WHERE table_schema='world';`	Lists all the tables in the **world** database, by name and engine; 

4. *Select* the table *name* and *engine* and *count* of *all* engines from the **tables** table and the **world** *schema* and *group by* schema and engine:
`SELECT table_schema, engine, count(*)`
`FROM tables`
`GROUP BY table_schema, engine;`
	Lists all the tables in the **world** database, by name and engine, with engine count;
	

ACTION (You Do)	COMPUTER RESPONSE / Comment

5. List the table name '**City**' and its' corresponding *data length*:
`SELECT table_name, data_length FROM tables`
`WHERE table_name='city';`
	Returns a row with the **City** table name and data length:
	[illegible]

6. Check how many **CHAR** and **VARCHAR** data types there are in the **world** database:
`SELECT DATA_TYPE, COUNT(*) FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_SCHEMA = 'world' AND DATA_TYPE IN ('CHAR', 'VARCHAR')`
`GROUP BY DATA_TYPE;`
How many are there for each type?
	Returns a row with the data type and count:
	[illegible]
	There are no **VARCHAR** data types in the **world** database, therefore it does not show up in the output..

6.3. Using SHOW and DESCRIBE

209

6.3.1. SHOW Statements

MySQL supports a set of many **SHOW** statements that each return one kind of metadata. This section describes a *few* of them:

[illegible]	[illegible]
SHOW DATABASES [LIKE '%']	Lists names of available databases (schemas or schemata)
SHOW CREATE TABLE <table>	Shows the CREATE TABLE statement that creates the given table or view.
SHOW [FULL] TABLES [LIKE '%']	Lists all tables available in current database. **FULL** will also show Table_Type: BASE TABLE or VIEW.
SHOW OPEN TABLES [LIKE '%']	Lists the non-TEMPORARY tables that are currently open in the table open cache.
SHOW [FULL] COLUMNS FROM <table>	Displays column structure information for a specified table.
SHOW INDEX FROM <table>	Returns table index information.
SHOW CHARACTER SET	Lists all available character sets.
SHOW COLLATION	Lists all available character sets, and with **LIKE** will indicate matching collations.
SHOW [FULL] PROCESSLIST	Displays which process threads are running.

210

SHOW DATABASES lists the names of the available databases:

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| test               |
| world              |
+--------------------+
```

SHOW TABLES lists the tables in the default database, or in the named database if a **FROM** clause is present:

```
mysql> SHOW TABLES;
+-----------------+
| Tables_in_world |
+-----------------+
| City            |
| Country         |
| CountryLanguage |
+-----------------+

mysql> SHOW TABLES FROM mysql;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| columns_priv              |
| db                        |
| func                      |
| help_category             |
| help_keyword              |
| help_relation             |
| help_topic                |
| host                      |
| proc                      |
| procs_priv                |
| tables_priv               |
| time_zone                 |
| time_zone_leap_second     |
| time_zone_name            |
| time_zone_transition      |
| time_zone_transition_type |
| user                      |
+---------------------------+
```

SHOW statements are available for metadata other than for databases, tables, and columns. For example, **SHOW CHARACTER SET** displays the available character sets along with their default collations, and **SHOW COLLATION** displays the collations for each character set:

```
mysql> SHOW CHARACTER SET;
+----------+-----------------------------+---------------------+--------+
| Charset  | Description                 | Default collation   | Maxlen |
+----------+-----------------------------+---------------------+--------+
| big5     | Big5 Traditional Chinese    | big5_chinese_ci     |      2 |
| dec8     | DEC West European           | dec8_swedish_ci     |      1 |
| cp850    | DOS West European           | cp850_general_ci    |      1 |
| hp8      | HP West European            | hp8_english_ci      |      1 |
| koi8r    | KOI8-R Relcom Russian       | koi8r_general_ci    |      1 |
| latin1   | ISO 8859-1 West European    | latin1_swedish_ci   |      1 |
| latin2   | ISO 8859-2 Central European | latin2_general_ci   |      1 |
...

mysql> SHOW COLLATION;
+----------------------+----------+-----+---------+----------+---------+
| Collation            | Charset  | Id  | Default | Compiled | Sortlen |
+----------------------+----------+-----+---------+----------+---------+
| big5_chinese_ci      | big5     |   1 | Yes     | Yes      |       1 |
| big5_bin             | big5     |  84 |         | Yes      |       1 |
| dec8_swedish_ci      | dec8     |   3 | Yes     |          |       0 |
| dec8_bin             | dec8     |  69 |         |          |       0 |
| cp850_general_ci     | cp850    |   4 | Yes     |          |       0 |
| cp850_bin            | cp850    |  80 |         |          |       0 |
| hp8_english_ci       | hp8      |   6 | Yes     |          |       0 |
...
```

212

6.3.2. DESCRIBE Statements

DESCRIBE, another metadata-display statement, is equivalent to SHOW COLUMNS. DESCRIBE can be abbreviated as DESC. The following statements display the same information:

```
DESCRIBE table_name;
DESC table_name;
SHOW COLUMNS FROM table_name;
```

However, whereas SHOW COLUMNS supports the optional FROM keyword, DESCRIBE does not.

DESCRIBE shows the column definitions for any INFORMATION_SCHEMA table:

```
mysql> DESCRIBE INFORMATION_SCHEMA.CHARACTER_SETS;
+----------------------+-------------+------+-----+---------+-------+
| Field                | Type        | Null | Key | Default | Extra |
+----------------------+-------------+------+-----+---------+-------+
| CHARACTER_SET_NAME   | varchar(64) | NO   |     |         |       |
| DEFAULT_COLLATE_NAME | varchar(64) | NO   |     |         |       |
| DESCRIPTION          | varchar(60) | NO   |     |         |       |
| MAXLEN               | bigint(3)   | NO   |     | 0       |       |
+----------------------+-------------+------+-----+---------+-------+
```

InLine Lab 6-B

In this exercise you will use the methods covered in this chapter for **SHOW/DESCRIBE**. This will require a MySQL command line client and access to the mysql server and use of the **world** database.

ACTION (You Do)	COMPUTER RESPONSE / Comment

1. Show *all* the databases available. And show all the tables in the **world** database:
`SHOW DATABASES;`
`SHOW TABLES FROM world;`
	Two different commands: 1-will return lists of all databases (including **INFORMATION_SCHEMA**):
	`+--------------------+`
	`| Database           |`
	`+--------------------+`
	`| information_schema |`
	`| mysql              |`
	`| world              |`
	`+--------------------+`
	`3 rows in set (0.54 sec)`
	and 2-all the tables in the **world** database:
	`+-----------------+`
	`| Tables_in_world |`
	`+-----------------+`
	`| City            |`
	`| Country         |`
	`| CountryLanguage |`
	`+-----------------+`
	`3 rows in set (0.05 sec)`

<table>
<tr><th>ACTION (You Do)</th><th>COMPUTER RESPONSE / Comment</th></tr>
<tr><td>2. Show detailed information about the columns in the <b>City</b> table. <b><i>Hint:</i></b> Use the \G terminator to get a more readable result:
<pre>SHOW FULL COLUMNS FROM City\G</pre></td><td>Results in a vertical list of several column details:
<pre>*************************** 1. row ***************************
     Field: ID
      Type: int(11)
 Collation: NULL
      Null: NO
       Key: PRI
   Default: NULL
     Extra: auto_increment
Privileges:
select,insert,update,references
   Comment:
*************************** 2. row ***************************
     Field: Name
      Type: char(35)
...</pre></td></tr>
<tr><td>3. Show information about the <i>indexes</i> in the <b>Country</b> database:
<pre>SHOW INDEX FROM Country\G</pre>
<b>Note:</b> <b><code>INFORMATION_SCHEMA.TABLE_CONSTRAINTS</code></b> and <b><code>INFORMATION_SCHEMA.STATISTICS</code></b> also contain index information.</td><td>Will show a detailed list of index information:
<pre>*************************** 1. row
****
        Table: Country
   Non_unique: 0
     Key name: PRIMARY
 Seq_in_index: 1
  Column_name: Code
    Collation: A
  Cardinality: 239
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE
      Comment:
1 row in set (0.00 sec)</pre></td></tr>
<tr><td>4. Show the structure of the <b>CountryLanguage</b> table:
<pre>DESCRIBE CountryLanguage;</pre></td><td>Shows in table form, the columns and attributes within the <b>CountryLanguage</b> table:
<pre>+-------------+---------------+ ...
| Field | Type |
+-------------+---------------+ ...
CountryCode	char(3)
Language	char(30)
IsOfficial	enum('T','F')
Percentage	float(4,1)
+-------------+---------------+ ...</pre></td></tr>
</table>

ACTION (You Do)	**COMPUTER RESPONSE / Comment**
5. Show any tables currently open in the table open cache: `SHOW OPEN TABLES;`	Will show a list of open tables, or an empty set if none are open..
6. List all the tables from the **INFORMATION_SCHEMA** database: `SHOW TABLES FROM INFORMATION_SCHEMA;`	Lists all tables: [illegible] . . .
7. List all *character sets* available: `mysql> SHOW CHARACTER SET;`	A full list of character sets on the current system will result.
8. List all *collations* available: `mysql> SHOW COLLATION;`	A full list of collations on the current system will result.

213 6.4. The mysqlshow Client

The **mysqlshow** client program produces information about the structure of your databases and tables. It provides a command-line interface to various forms of the **SHOW** statement that list the names of your databases, tables within a database, or information about table columns or indexes. The **mysqlshow** command has this syntax:

```
mysqlshow [options] [db_name [table_name [column_name]]]
```

The *options* part of the **mysqlshow** command may include any of the standard connection parameter options, such as **--host** or **--user**. You'll need to supply these options if the default connection parameters are not appropriate. **mysqlshow** also understands options specific to its own operation. Invoke **mysqlshow** with the **--help** option to see a complete list of its options.

214 The action performed by **mysqlshow** depends on the number of non-option arguments you provide:

- With no arguments, **mysqlshow** displays a result similar to that of **SHOW DATABASES**:

```
shell> mysqlshow
+--------------------+
|     Databases      |
+--------------------+
| information_schema |
| mysql              |
| world              |
+--------------------+
```

- With a single argument, **mysqlshow** interprets it as a database name and displays a result similar to that of **SHOW TABLES** for the database:

```
shell> mysqlshow world
Database: world
+-----------------+
|     Tables      |
+-----------------+
| City            |
| Country         |
| CountryLanguage |
+-----------------+
```

- With two arguments, **mysqlshow** interprets them as a database and table name and displays a result similar to that of **SHOW FULL COLUMNS** for the table. With three arguments, the output is the same as for two arguments except that **mysqlshow** takes the third argument as a column name and displays **SHOW FULL COLUMNS** output only for that column. The following commands are examples of this invocation syntax. (The output is not shown because it is too wide to fit the page.)

```
shell> mysqlshow world City
```

```
shell> mysqlshow world City CountryCode
```

When **mysqlshow** is used to display table structure, the **--keys** option may be given to display index structure as well. This information is similar to the output of **SHOW INDEX** (or **SHOW KEYS**) for the table.

If the final argument on the command line contains special characters, **mysqlshow** interprets the argument as a pattern and displays only the names that match the pattern. The special characters are '%' or '*' to match any sequence of characters, and '_' or '?' to match any single character. For example, the following command shows only those databases with a name that begins with '**w**':

```
shell> mysqlshow -u <user_name> -p "w%"
Enter Password: <password>
Wildcard: w%
+------------+
| Databases  |
+------------+
| world      |
+------------+
```

Note: The above example demonstrates the use of user and password parameters as part of command execution.

The pattern characters might be treated as special by your command interpreter. An argument that contains any such characters should be quoted, as shown in the preceding example. Alternatively, use a character that your command interpreter does not treat specially. For example, '*' can be used without quoting on Windows and '%' without quoting on Linux.

InLine Lab 6-C

In this exercise you will use the methods covered in this chapter for **SHOW/DESCRIBE**. This will require a MySQL command line client.

<table>
<tr><th>ACTION (You Do)</th><th>COMPUTER RESPONSE / Comment</th></tr>
<tr><td>

1. From the shell prompt, show the current databases:

```
shell> mysqlshow -uroot
-p
Enter password: ******
```

</td><td>

Shows all available `mysql` databases on the host:

```
+--------------------+
|     Databases      |
+--------------------+
| information_schema |
| cluster            |
| mysql              |
| test               |
| world              |
+--------------------+
```

</td></tr>
<tr><td>

2. Now add (to the above statement) to list the tables in the **world** database:

```
shell> mysqlshow world
-uroot -p
Enter password: ******
```

</td><td>

Shows list of all tables in the existing mysql database of **world**:

```
+-----------------+
|     Tables      |
+-----------------+
| city            |
| country         |
| countrylanguage |
+-----------------+
```

</td></tr>
<tr><td>

3. Now add (to the above statement) the specific table of **CountryLanguage**:

```
shell> mysqlshow world
CountryLanguage -uroot
-p
Enter password: ******
```

</td><td>

Shows the table structure of the **CountryLanguage** table;

```
+-------------+-----------------+-------------------+------+
-----+---------+-------+
---------------------------------+---------+
| Field       | Type            | Collation         | Null
| Key | Default | Extra |
  Privileges                      | Comment |
+-------------+-----------------+-------------------+------+
-----+---------+-------+
---------------------------------+---------+
| CountryCode | char(3)         | latin1_swedish_ci | NO   |
PRI |
  select,insert,update,references |
...
```

</td></tr>
</table>

215

Further Practice

In this exercise, you will use the information covered in this chapter to **obtain metadata** using t he **world** database.

1. Using **INFORMATION_SCHEMA**, list the number of rows contained in the tables of the **world** database.
2. Using the **STATISTICS** table, check if the **world** schema has any non_unique indexes.
3. How many different user accounts are there?
4. What is the total size (in bytes) of the **City** table, including indexes?

216 Chapter Summary

This chapter introduced **Obtaining Metadata** in **MySQL**. In this chapter, you learned to:

- List the various available methods to access metadata
- The INFORMATION_SCHEMA database and its contents
- The SHOW statement
- The DESCRIBE statement
- The mysqlshow client program
- The differences between SHOW statements and INFORMATION_SCHEMA tables