Improving Consistency, Performance, Compatibility, Ease of use

Alexander Barkov
Server Engineer
MariaDB plc

Plan for this session

Some tasks are released, some tasks are in stage branches.

Optimizer:

MDEV-32148 Inefficient WHERE timestamp_column=datetime_const_expr
MDEV-32203 Raise notes when an index cannot be used on data type mismatch

Unicode:

MDEV-27009 Add UCA-14.0.0 collations
MDEV-30577 Case folding for uca1400 collations is not up to date
MDEV-27266 Improve UCA collation performance for utf8mb3 and utf8mb4
MDEV-30164 System variable for default collations
MDEV-25829 Change default collation to utf8mb4_1400_ai_ci (on TODO)

Stored procedures / compatibility:

MDEV-32101 CREATE PACKAGE [BODY] for sql_mode=DEFAULT
MDEV-20034 sql_mode="oracle" does not support stored code returning … SYS_REFCURSOR
MDEV-12252 ROW data type for stored function return values
MDEV-31250 ROW variables do not get assigned from sub-selects

Optimizer

MDEV-32148 Inefficient WHERE timestamp_column=datetime_const_expr

Coding done, now in review (for 10.6)
https://github.com/MariaDB/server/tree/bb-10.6-bar-MDEV-32148

```
CREATE OR REPLACE TABLE t1 (timestamp_column TIMESTAMP);
INSERT INTO t1 ...;
SELECT * FROM t1
  WHERE timestamp_column='2001-01-01 10:20:30' /*DATETIME const expr*/;
```

Historically compared as DATETIME, because:

- DATETIME has a wider range than TIMESTAMP
- Two different TIMESTAMP values can have the same DATETIME value near the "fall back" DST change, as well as for leap seconds (5 leap seconds added since 2000)
- There are DATETIME gaps during the "spring forward" DST change. MariaDB adjusts values inside DST gaps to the start of the gap

Comparing as DATETIME takes all irregularities into account, but it is slow:

- the TIMESTAMP side gets converted to DATETIME per-row
- involves a slow localtime_r() system call.

MariaDB

MDEV-32148 Inefficient WHERE timestamp_column=datetime_const_expr

Coding done, now in review (for 10.6)
https://github.com/MariaDB/server/tree/bb-10.6-bar-MDEV-32148

```
CREATE OR REPLACE TABLE t1 (timestamp_column TIMESTAMP);
INSERT INTO t1 ...;
SELECT * FROM t1
  WHERE timestamp_column='2001-01-01 10:20:30' /*DATETIME const expr*/;
```

MDEV-32148 changes: TIMESTAMP can be compared to DATETIME as TIMESTAMP if:

- the DATETIME side is a constant
- the DATETIME side is inside a reliably wide monotone continuous period (without DST changes and leap seconds).

The DATETIME argument gets converted once to TIMESTAMP, so no data type conversion, like slow localtime_r() calls, happens per row.

Benchmarking on 1M rows: query time (in seconds) changed:

- from 0.253 to 0.189 for InnoDB (25% improvement)
- from 0.109 to 0.058 for MyISAM (47% improvement)
- from 0.106 to 0.052 for HEAP (51% improvement)

MariaDB

MDEV-32203 Raise notes when an index cannot be used on data type mismatch

Done, to be pushed soon (10.6) – coding was done mostly by Monty
https://github.com/MariaDB/server/tree/bb-10.6-mdev32203

New notifications if an index cannot be used by the range optimizer because of:

- data type mismatch
- collation mismatch for text string data types

Possible notification destination:

- Client session SHOW WARNINGS (as SQL Notes)
- Slow Log

New notifications are off by default. Run as DBA to enable:

```
-- Session SQL Notes
SET GLOBAL note_verbosity=all;
-- Slow Log
SET GLOBAL slow_query_log=1;
SET GLOBAL log_output='FILE';
SET GLOBAL long_query_time=0.1;
SET GLOBAL log_slow_verbosity=all;
```

MariaDB

MDEV-32203 Raise notes when an index cannot be used on data type mismatch

Done, to be pushed soon (10.6) – coding was done mostly by Monty
https://github.com/MariaDB/server/tree/bb-10.6-mdev32203

Run in a new user session:

```
CREATE OR REPLACE TABLE t1 (a VARCHAR(10), KEY(a));
INSERT INTO t1 SELECT seq FROM seq_0_to_1000;
SELECT a, SLEEP(1) FROM t1 WHERE a=10;
+------+
| a    |
+------+
| 10   |
+------+
1 row in set, 1 warning (1.006 sec)

SHOW WARNINGS;
+-------+------+-----------------------------------------------------------------------------------------------------+
| Level | Code | Message                                                                                             |
+-------+------+-----------------------------------------------------------------------------------------------------+
| Note  | 1105 | Cannot use key `a` part[0] for lookup: `test`.`t1`.`a` of type `varchar` = "10" of type `int` |
+-------+------+-----------------------------------------------------------------------------------------------------+
```

New notes also work in joins!

MariaDB

MDEV-32203 Raise notes when an index cannot be used on data type mismatch

Done, to be pushed soon (10.6) – coding was done mostly by Monty
https://github.com/MariaDB/server/tree/bb-10.6-mdev32203

Slow Log entry:

```
# Time: 230928 15:51:00
# User@Host: bar[bar] @ localhost []
# Thread_id: 11  Schema: test  QC_hit: No
# Query_time: 1.009794  Lock_time: 0.000081  Rows_sent: 1  Rows_examined: 1001
# Rows_affected: 0  Bytes_sent: 105
# Pages_accessed: 3  Pages_read: 0  Pages_updated: 0  Old_rows_read: 0
# Pages_read_time: 0.0000  Engine_time: 8.1950
# Full_scan: Yes  Full_join: No  Tmp_table: No  Tmp_table_on_disk: No
# Filesort: No  Filesort_on_disk: No  Merge_passes: 0  Priority_queue: No
#
# explain: id   select_type     table   type    possible_keys   key     key_len ref     rows    r_rows  filtered        r_filtered      Extra
# explain: 1    SIMPLE  t1      index   a       a       13      NULL    1001    1001.00 100.00  0.10    Using where; Using index
#
# Warnings
# Note 1105 Cannot use key `a` part[0] for lookup: `test`.`t1`.`a` of type `varchar` = "10" of type `int`
SET timestamp=1695901860;
SELECT a, SLEEP(1) FROM t1 WHERE a=10;
```

MariaDB

Unicode

MDEV-27009 Add UCA-14.0.0 collations

Released in 10.10.1 (22 Aug 2022)

Unicode Collation Algorithm version 14.0.0 based collations:

- Applicable to all Unicode character sets: utf8mb3, utf8mb4, utf16, utf32
- Have the _**uca1400**_ infix in their names (easy to recognize)
- One default (neutral) + 22 tailored collations (for various languages)
- 8 variants for every tailoring (language): **as/ai** * **cs/ci** * **pad/nopad**
- Primary, Secondary, Tertiary weight levels
- Built-in contractions, defined in the Default Unicode Collation Element Table (DUCET). MySQL-8.0 does not have built-in contractions in utf8mb4_0900_ai_ci

Easier to use:

- **New:** Can be used without the character set name prefix
- **New:** The character set prefix is detected automatically from the context
- **Compliance:** A step towards the SQL Standard: a collation can be applicable to multiple character sets

MariaDB

MDEV-27009 Add UCA-14.0.0 collations

Released in 10.10.1 (22 Aug 2022)

Getting a list of new collations:

```
MariaDB [test]> SHOW COLLATION LIKE '%uca1400%';
+--------------------------------+---------+------+---------+----------+---------+
| Collation                      | Charset | Id   | Default | Compiled | Sortlen |
+--------------------------------+---------+------+---------+----------+---------+
| uca1400_ai_ci                  | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_ai_cs                  | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_as_ci                  | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_as_cs                  | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_nopad_ai_ci            | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_nopad_ai_cs            | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_nopad_as_ci            | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_nopad_as_cs            | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_icelandic_ai_ci        | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_icelandic_ai_cs        | NULL    | NULL | NULL    | Yes      |       8 |
...
| uca1400_croatian_nopad_as_ci   | NULL    | NULL | NULL    | Yes      |       8 |
| uca1400_croatian_nopad_as_cs   | NULL    | NULL | NULL    | Yes      |       8 |
+--------------------------------+---------+------+---------+----------+---------+
184 rows in set (0.006 sec)
```

- There are no character set name prefixes in the column `**Collation**`
- NULL in columns: **Charset**, **Id**, **Default**

MariaDB

MDEV-27009 Add UCA-14.0.0 collations

Released in 10.10.1 (22 Aug 2022)

Getting a list of new collations from **COLLATION_CHARACTER_SET_APPLICABILITY**:

```
SELECT * FROM INFORMATION_SCHEMA.COLLATION_CHARACTER_SET_APPLICABILITY
  WHERE COLLATION_NAME LIKE '%uca1400_as%';
+----------------+--------------------+-----------------------+------+------------+
| COLLATION_NAME | CHARACTER_SET_NAME | FULL_COLLATION_NAME   | ID   | IS_DEFAULT |
+----------------+--------------------+-----------------------+------+------------+
| uca1400_as_ci  | utf8mb3            | utf8mb3_uca1400_as_ci | 2050 |            |
| uca1400_as_cs  | utf8mb3            | utf8mb3_uca1400_as_cs | 2051 |            |
| uca1400_as_ci  | ucs2               | ucs2_uca1400_as_ci    | 2562 |            |
| uca1400_as_cs  | ucs2               | ucs2_uca1400_as_cs    | 2563 |            |
| uca1400_as_ci  | utf8mb4            | utf8mb4_uca1400_as_ci | 2306 |            |
| uca1400_as_cs  | utf8mb4            | utf8mb4_uca1400_as_cs | 2307 |            |
| uca1400_as_ci  | utf16              | utf16_uca1400_as_ci   | 2818 |            |
| uca1400_as_cs  | utf16              | utf16_uca1400_as_cs   | 2819 |            |
| uca1400_as_ci  | utf32              | utf32_uca1400_as_ci   | 3074 |            |
| uca1400_as_cs  | utf32              | utf32_uca1400_as_cs   | 3075 |            |
+----------------+--------------------+-----------------------+------+------------+
10 rows in set (0.004 sec)
```

Old columns

- **COLLATION_NAME** – has no character set name prefixes for uca1400 collations
- **CHARACTER_SET_NAME** – no changes

New columns

- **FULL_COLLATION_NAME** - include character set name prefixes
- **ID**
- **IS_DEFAULT**

MariaDB

MDEV-27009 Add UCA-14.0.0 collations

Released in 10.10.1 (22 Aug 2022)

Easier to use: the short name (with no prefix) is applicable to multiple character sets:

```
CREATE OR REPLACE DATABASE db1;
CREATE TABLE db1.t1 (
  a TEXT CHARACTER SET utf8mb4 COLLATE uca1400_as_ci,
  b TEXT CHARACTER SET utf16 COLLATE uca1400_as_ci);
SHOW CREATE TABLE db1.t1;

+-------+-----------------------------------------------------------------
| Table | Create Table
+-------+-----------------------------------------------------------------
| t1    | CREATE TABLE `t1` (
  `a` text CHARACTER SET utf8mb4 COLLATE utf8mb4_uca1400_as_ci DEFAULT NULL,
  `b` text CHARACTER SET utf16 COLLATE utf16_uca1400_as_ci DEFAULT NULL
) ...
+-------+-----------------------------------------------------------------
```

MariaDB

MDEV-30577 Case folding for uca1400 collations is not up to date

Released in 10.10.4 (10 May 2023)

CaseFolding.txt from https://www.unicode.org/

- 413 new case folding entries were added between Unicode-5.2.0 and Unicode-14.0.0
- New entries did not work in old collations

```
CREATE OR REPLACE TABLE t1 (a VARCHAR(10) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_520_ci);
# Insert letters appeared in Unicode-6.1 (released in January 2012)
INSERT INTO t1 VALUES (_ucs2 0xA792) /* U+A792 LATIN CAPITAL LETTER C WITH BAR */;
INSERT INTO t1 VALUES (_ucs2 0xA793) /* U+A793 LATIN SMALL LETTER C WITH BAR */;

SELECT HEX(a), HEX(LOWER(a)), HEX(UPPER(a)), a, LOWER(a), UPPER(a) FROM t1;
+--------+--------------+--------------+------+----------+----------+
| HEX(a) | HEX(LOWER(a)) | HEX(UPPER(a)) | a    | LOWER(a) | UPPER(a) |
+--------+--------------+--------------+------+----------+----------+
| EA9E92 | EA9E92        | EA9E92        | Ȼ    | Ȼ        | Ȼ        | case folding is not up to date
| EA9E93 | EA9E93        | EA9E93        | ȼ    | ȼ        | ȼ        | case folding is not up to date
+--------+--------------+--------------+------+----------+----------+

ALTER TABLE t1 MODIFY a VARCHAR(10) CHARACTER SET utf8mb4 COLLATE uca1400_ai_ci;
SELECT HEX(a), HEX(LOWER(a)), HEX(UPPER(a)), a, LOWER(a), UPPER(a) FROM t1;
+--------+--------------+--------------+------+----------+----------+
| HEX(a) | HEX(LOWER(a)) | HEX(UPPER(a)) | a    | LOWER(a) | UPPER(a) |
+--------+--------------+--------------+------+----------+----------+
| EA9E92 | EA9E93        | EA9E92        | Ȼ    | ȼ        | Ȼ        | case folding is up to date
| EA9E93 | EA9E93        | EA9E92        | ȼ    | ȼ        | Ȼ        | case folding is up to date
+--------+--------------+--------------+------+----------+----------+
```

MariaDB

MDEV-27266 Improve UCA collation performance for utf8mb3 and utf8mb4

Released in 10.10.1 (22 Aug 2022)

Performance comparison script

```
SET NAMES utf8mb4 COLLATE uca1400_ai_ci;

-- Warning up
DO BENCHMARK(10000000,strcmp('xxxx','xxxx'));

-- Benchmarking
DO BENCHMARK(10000000,strcmp('aaaa','aaaa'));
DO BENCHMARK(10000000,strcmp('aaaaaaaa','aaaaaaaa'));
DO BENCHMARK(10000000,strcmp('яяяя','яяяя'));
DO BENCHMARK(10000000,strcmp('ssss','ßß'));
DO BENCHMARK(10000000,strcmp('ǎǎǎǎ','ǎǎǎǎ'));
DO BENCHMARK(10000000,strcmp('        ','        '));
```

Performance comparison results

		MySQL-8.0.38		MariaDB-10.10.4		
Str1	Str2	general_ci	0900_ai_ci	general_ci	uca1400_ai_ci	
aaaa	aaaa	0.213	0.335	0.101	**0.197**	mb1
aaaaaaaa	aaaaaaaa	0.352	0.540	0.114	**0.220**	mb1
яяяя	яяяя	0.297	0.399	0.195	**0.219**	mb2
ssss	ßß	----	0.332	----	**0.196**	mb1/mb2
ǎǎǎǎ	ǎǎǎǎ	0.421	**0.599**	0.274	0.710	mb3
		0.284	**0.476**	0.268	0.792	mb4

MySQL's **0900_ai_ci** vs MariaDB's **uca1400_ai_ci**

- MariaDB is much faster on **1-byte** and **2-byte** characters (very important!)
- MariaDB is slower on 3-byte and 4-byte characters (tolerable)

MariaDB

MDEV-30164 System variable for default collations

Released in 11.2.1 (21 Aug 2023)

Pre-MDEV-30164:

```
CREATE OR REPLACE TABLE t1 (a TEXT CHARACTER SET utf8mb4);
SHOW CREATE TABLE t1;
+-------+-------------------------------------------------------------
| Table | Create Table
+-------+-------------------------------------------------------------
|    t1 | CREATE TABLE `t1` (
  `a` text CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci DEFAULT NULL
) ...
+-------+-------------------------------------------------------------
```

Old behavior:

- Every character set has a hard coded default collation
- E.g. **utf8mb4_general_ci** is a hard coded default collation for **utf8mb4**
- The only way to use a non-default collation was an explicit COLLATE clause:

```
CREATE OR REPLACE TABLE t1 (a TEXT CHARACTER SET utf8mb4 COLLATE utf8mb4_german2_ci);
```

MDEV-30164 System variable for default collations

Released in 11.2.1 (21 Aug 2023)

A new system variable makes it easier: no annoying COLLATE clauses needed any more:

```
SET @@character_set_collations = 'cs1=col1, cs2=col2, cs3=col3';  /* A list of CHARSET=COLLATION pairs */
```

Example:

```
SET @@character_set_collations='utf8mb4=uca1400_german2_ai_ci,latin1=latin1_german2_ci';

CREATE OR REPLACE TABLE t1 (a TEXT CHARACTER SET utf8mb4, b TEXT CHARACTER SET latin1);

SHOW CREATE TABLE t1;
+-------+-----------------------------------------------------------------
| Table | Create Table
+-------+-----------------------------------------------------------------
|    t1 | CREATE TABLE `t1` (
  `a` text CHARACTER SET utf8mb4 COLLATE utf8mb4_uca1400_german2_ai_ci DEFAULT NULL,
  `b` text CHARACTER SET latin1 COLLATE latin1_german2_ci DEFAULT NULL
) ....
+-------+-----------------------------------------------------------------
```

Can be added to my.cnf

```
[server]
character-set-collations=utf8mb4=uca1400_german2_as_ci,latin1=latin1_german2_ci
```

MariaDB

MDEV-29446 Change SHOW CREATE TABLE to display default collations

Released in 10.3.37, 10.4.27, 10.5.18, 10.6.11, 10.8.6, 10.9.4, 10.10.2, 10.11.0 (26 Sep 2022)

Before MDEV-29446:

```
CREATE OR REPLACE TABLE t1 (a TEXT CHARACTER SET latin2, b TEXT CHARACTER SET big5);
SHOW CREATE TABLE t1;
+-------+------------------------------
| Table | Create Table
+-------+------------------------------
| t1    | CREATE TABLE `t1` (
  `a` text CHARACTER SET latin2,
  `b` text CHARACTER SET big5
) ENGINE=InnoDB DEFAULT CHARSET=latin1
+-------+------------------------------
```

After-MDEV-29446

```
+-------+-------------------------------------------------------
| Table | Create Table
+-------+-------------------------------------------------------
| t1    | CREATE TABLE `t1` (
  `a` text CHARACTER SET latin2 COLLATE latin2_general_ci DEFAULT NULL,
  `b` text CHARACTER SET big5 COLLATE big5_chinese_ci DEFAULT NULL
) ...
+-------+-------------------------------------------------------
```

The change allows to recreate tables reliably no matter what the current @@**character_set_collations** is.

MariaDB

Stored routines and Compatibility

MariaDB

MDEV-31250 ROW variables do not get assigned from subselects

Released in 10.4.31 (14 Aug 2023)

Before MDEV-29446:

```
CREATE OR REPLACE TABLE t1 (a INT, b TEXT);
INSERT INTO t1 VALUES (1,'b1');
DELIMITER $$
BEGIN NOT ATOMIC
  DECLARE r0 ROW TYPE OF t1;
  DECLARE r1 ROW TYPE OF t1;
  SELECT * INTO r0 FROM t1 WHERE a=1; -- This worked
  SET r1=(SELECT * FROM t1 WHERE a=1); -- This did not
  SELECT r0.a, r0.b, r1.a, r1.b;
END;
$$
DELIMITER ;
+------+------+------+------+
| r0.a | r0.b | r1.a | r1.b |
+------+------+------+------+
|    1 |   b1 | NULL | NULL | before the change
+------+------+------+------+
+------+------+------+------+
| r0.a | r0.b | r1.a | r1.b |
+------+------+------+------+
|    1 |   b1 |    1 |   b1 | after the change
+------+------+------+------+
```

MDEV-12252 ROW data type for stored function return values

Coding done, awaiting review (for 11.4) - https://github.com/MariaDB/server/tree/bb-11.3-bar-MDEV-12252

New data types allowed as a stored function return type

- **ROW(a INT, b VARCHAR(10))** – explicitly typed ROW, any sql_mode
- **ROW TYPE OF t1** – get a ROW structure from a table, sql_mode=DEFAULT
- **TYPE OF t1.col1** – get a scalar data type from a table column, sql_mode=DEFAULT
- **t1%ROWTYPE** – get a ROW structure from a table, sql_mode=ORACLE
- **t1.col1%TYPE** – get a scalar data type from a table column, sql_mode=ORACLE

```
CREATE OR REPLACE TABLE t1 (id INT PRIMARY KEY, first_name VARCHAR(128), last_name VARCHAR(128));
INSERT INTO t1 VALUES (1,'Edgar', 'Codd');

DELIMITER $$
CREATE OR REPLACE FUNCTION f1(pid INT)
  RETURNS ROW TYPE OF t1
  RETURN (SELECT * FROM t1 WHERE id=pid);

-- Calling the function
BEGIN NOT ATOMIC
  DECLARE r ROW TYPE OF t1 /*DEFAULT f1(1)*/;
  SET r= f1(1);
  SELECT r.id, r.first_name, r.last_name;
END;
$$
DELIMITER ;
+------+--------------+-------------+
| r.id | r.first_name | r.last_name |
+------+--------------+-------------+
|    1 |        Edgar |        Codd |
+------+--------------+-------------+
```

MariaDB

MDEV-20034 sql_mode="oracle" does not support ... SYS_REFCURSOR

Coding done, awaiting review (for 11.4) - https://github.com/MariaDB/server/commits/bb-11.3-bar-MDEV-20034

Now functions can return cursors:

```
SET sql_mode=DEFAULT;
CREATE OR REPLACE TABLE t1 (id INT, name VARCHAR(128));
INSERT INTO t1 VALUES (1,'John'),(2,'Paul');

DELIMITER /
CREATE OR REPLACE FUNCTION f1() RETURNS SYS_REFCURSOR
BEGIN
  DECLARE c SYS_REFCURSOR;
  OPEN c FOR SELECT * FROM t1;
  RETURN c;
END;
/
CREATE OR REPLACE PROCEDURE p1()
BEGIN
  DECLARE c SYS_REFCURSOR DEFAULT f1();
  DECLARE r ROW TYPE OF t1;
  DECLARE done INT DEFAULT FALSE;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
fetch_loop:
  LOOP
    FETCH c INTO r;
    IF done THEN
       LEAVE fetch_loop;
    END IF;
    SELECT r.id, r.name;
  END LOOP;
  CLOSE c;
END;
/
DELIMITER ;
```

sql_mode=ORACLE dialect:

```
SET sql_mode=ORACLE;
CREATE OR REPLACE TABLE t1 (id INT, name VARCHAR(128));
INSERT INTO t1 VALUES (1,'John'),(2,'Paul');

DELIMITER /
CREATE OR REPLACE FUNCTION f1 RETURN SYS_REFCURSOR AS
  c SYS_REFCURSOR;
BEGIN
  OPEN c FOR SELECT * FROM t1;
  RETURN c;
END;
/
CREATE OR REPLACE PROCEDURE p1 AS
  c SYS_REFCURSOR := f1();
  r t1%ROWTYPE;
BEGIN
  LOOP
    FETCH c INTO r;
    EXIT WHEN c%NOTFOUND;
    SELECT r.id, r.name;
  END LOOP;
  CLOSE c;
END;
/
DELIMITER ;
```

The output

```
CALL p1;

+------+--------+
| r.id | r.name |
+------+--------+
|    1 |   John |
+------+--------+
1 row in set (0.003 sec)

+------+--------+
| r.id | r.name |
+------+--------+
|    2 |   Paul |
+------+--------+
1 row in set (0.003 sec)
```

MariaDB

MDEV-32101 CREATE PACKAGE [BODY] for sql_mode=DEFAULT

Coding done, awaiting review (for 11.4) - https://github.com/MariaDB/server/commits/bb-11.3-bar-MDEV-32101

Defining tables:

```
CREATE OR REPLACE TABLE employee
(
  id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
  name VARCHAR(128),
  salary DECIMAL(9,2)
);

CREATE OR REPLACE TABLE employee_log
(
  ts TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  id INT NOT NULL,
  cmnt TEXT
);
```

Defining specifications for a new package

```
SET sql_mode=DEFAULT;

DELIMITER $$
CREATE OR REPLACE PACKAGE employee_tools
  FUNCTION getSalary(eid INT) RETURNS DECIMAL(10,2);
  PROCEDURE hireStd(ename VARCHAR(128));
  PROCEDURE raiseSalaryStd(eid INT);
END;
$$
DELIMITER ;
```

MariaDB

MDEV-32101 CREATE PACKAGE [BODY] for sql_mode=DEFAULT

Coding done, awaiting review (for 11.4) - https://github.com/MariaDB/server/commits/bb-11.3-bar-MDEV-32101

PACKAGE implementation:

```
DELIMITER $$
CREATE OR REPLACE PACKAGE BODY employee_tools
  -- package body variables
  DECLARE stdSalary DECIMAL (10,2) DEFAULT 5000;
  DECLARE stdRaiseAmount DECIMAL(10,2) DEFAULT 500;

  -- private routines
  PROCEDURE log(eid INT, ecmnt TEXT)
  BEGIN
    INSERT INTO employee_log (id, cmnt) VALUES (eid, ecmnt);
  END;

  -- public routines
  PROCEDURE hireStd(ename TEXT)
  BEGIN
    DECLARE eid INT;
    INSERT INTO employee (name, salary) VALUES (ename, stdSalary);
    SET eid= last_insert_id();
    CALL log(eid, CONCAT('hire ', ename, ' salary=', stdSalary));
  END;

  FUNCTION getSalary(eid INT) RETURNS DECIMAL(10,2)
  BEGIN
    DECLARE nSalary DECIMAL(10,2);
    SELECT salary INTO nSalary FROM employee WHERE id=eid;
    CALL log(eid, CONCAT('getSalary id=', eid, ' salary=', nSalary));
    RETURN nSalary;
  END;

  /* continued on the right bar */
```

PACKAGE implementation (continued):

```
  -- more public routines
  PROCEDURE raiseSalaryStd(eid INT)
  BEGIN
    UPDATE employee SET salary=salary+stdRaiseAmount WHERE id=eid;
    CALL log(eid, CONCAT('raiseSalaryStd id=', eid, ' RaiseAmount=', stdRaiseAmount));
  END;

  -- This code is executed when the current session
  -- calls any of the package routines for the first time
  CALL log(0, CONCAT('Session ', connection_id(), ' ', current_user, ' started'));
END;
$$
DELIMITER ;
```

Using the PACKAGE

```
CALL employee_tools.hireStd('John');
CALL employee_tools.raiseSalaryStd((SELECT id FROM employee WHERE name='John'));

SELECT * FROM employee_log;
+---------------------+----+---------------------------------------+
| ts                  | id | cmnt                                  |
+---------------------+----+---------------------------------------+
| 2023-09-28 15:03:58 |  0 | Session 5 bar@localhost started       |
| 2023-09-28 15:03:58 |  1 | hire John salary=5000.00              |
| 2023-09-28 15:04:01 |  1 | raiseSalaryStd id=1 RaiseAmount=500.00 |
+---------------------+----+---------------------------------------+
```

MariaDB


MariaDB
THANK YOU