MariaDB_10.11

Content

Datasets

```
CREATE TABLE authors (
 au_id CHAR(3) NOT NULL,
 au_fname VARCHAR(15) NOT NULL,
 au_lname VARCHAR(15) NOT NULL,
 phone VARCHAR(12) ,
 address VARCHAR(20) ,
 city VARCHAR(15) ,
 state CHAR(2) ,
 zip CHAR(5) ,
 CONSTRAINT authors_pk
 PRIMARY KEY (au_id),
 CONSTRAINT authors_unique1
 UNIQUE (au_fname, au_lname));
```

```
MariaDB [sparksql]> DESC authors;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| au_id    | char(3)     | NO   | PRI | NULL    |       |
| au_fname | varchar(15) | NO   | MUL | NULL    |       |
| au_lname | varchar(15) | NO   |     | NULL    |       |
| phone    | varchar(12) | YES  |     | NULL    |       |
| address  | varchar(20) | YES  |     | NULL    |       |
| city     | varchar(15) | YES  |     | NULL    |       |
| state    | char(2)     | YES  |     | NULL    |       |
| zip      | char(5)     | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
8 rows in set (0.006 sec)
```

```
INSERT INTO authors(au_id,au_fname,au_lname,phone,address,city,state,zip)
 values ('A07','Paddy',"O'Furniture",'941-925-0752','1442 Main St','Sarasota','FL','34236');
```

```
MariaDB [sparksql]> select * from authors;
+-------+----------+------------+--------------+--------------------+---------------+-------+-------+
| au_id | au_fname | au_lname   | phone        | address            | city          | state | zip   |
+-------+----------+------------+--------------+--------------------+---------------+-------+-------+
| A01   | Sarah    | Buchman    | 718-496-7223 | 75 West 205 St     | Bronx         | NY    | 10468 |
| A02   | Wendy    | Heydemark  | 303-986-7020 | 2922 Baseline Rd   | Boulder       | CO    | 80303 |
| A03   | Hallie   | Hull       | 415-549-4278 | 3800 Waldo Ave, #14F | San Francisco | CA    | 94123 |
| A04   | Klee     | Hull       | 415-549-4278 | 3800 Waldo Ave, #14F | San Francisco | CA    | 94123 |
| A05   | Christian | Kells     | 212-771-4680 | 114 Horatio St     | New York      | NY    | 10014 |
| A06   |          | Kellsey    | 650-836-7128 | 390 Serra Mall     | Palo Alto     | CA    | 94305 |
| A07   | Paddy    | O'Furniture | 941-925-0752 | 1442 Main St       | Sarasota      | FL    | 34236 |
+-------+----------+------------+--------------+--------------------+---------------+-------+-------+
7 rows in set (0.000 sec)
```

```
CREATE TABLE publishers (
  pub_id CHAR(3) NOT NULL,
```

```
 pub_name VARCHAR(20) NOT NULL
 city VARCHAR(15) NOT NULL,
 state CHAR(2) ,
 country VARCHAR(15) NOT NULL,
 CONSTRAINT publishers_pk PRIMARY KEY (pub_id));
```

```
MariaDB [sparksql]> desc publishers
    -> ;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| pub_id   | char(3)     | NO   | PRI | NULL    |       |
| pub_name | varchar(20) | NO   |     | NULL    |       |
| city     | varchar(15) | NO   |     | NULL    |       |
| state    | char(2)     | YES  |     | NULL    |       |
| country  | varchar(15) | NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
5 rows in set (0.006 sec)
```

```
INSERT INTO publishers (pub_id,pub_name,city,state,country)
 values ('P04','Tenterhooks Press','Berkeley','CA','USA');
```

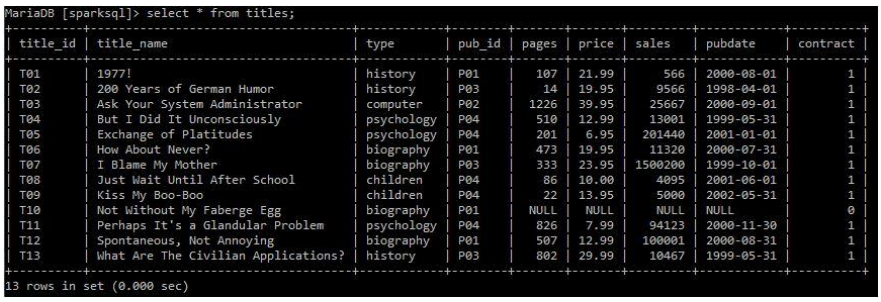
```
MariaDB [sparksql]> select * from publishers;
+--------+--------------------+--------------+-------+---------+
| pub_id | pub_name           | city         | state | country |
+--------+--------------------+--------------+-------+---------+
| P01    | Abatis Publishers  | New York     | NY    | USA     |
| P02    | Core Dump Books    | San Francisco| CA    | USA     |
| P03    | Schadenfreude Press| Hamburg      | NULL  | Germany |
| P04    | Tenterhooks Press  | Berkeley     | CA    | USA     |
+--------+--------------------+--------------+-------+---------+
4 rows in set (0.000 sec)
```

```
CREATE TABLE titles(
 title_id CHAR(3) NOT NULL,
 title_name VARCHAR(40) NOT NULL,
 type VARCHAR(10) ,
 pub_id CHAR(3) NOT NULL,
 pages INTEGER
 CHECK (pages > 0) ,
 price DECIMAL(5,2) ,
 sales INTEGER ,
 pubdate DATE ,
 contract SMALLINT NOT NULL,
 CONSTRAINT pk_titles PRIMARY KEY (title_id),
 CONSTRAINT title_id_chk
 CHECK (
  (SUBSTR(title_id, 1, 1) = 'T')
```

```
 AND (CAST(SUBSTR(title_id, 2, 2) AS SIGNED) BETWEEN 0 AND 99)),
CONSTRAINT price_chk CHECK (price >= 0.00 AND price < 100.00),
CONSTRAINT sales_chk CHECK (sales >= 0),
CONSTRAINT pubdate_chk CHECK (pubdate >= '1950-01-01'),
CONSTRAINT title_name_chk CHECK (title_name <> "AND contract >= 0));
```

```
MariaDB [sparksql]> desc titles;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| title_id   | char(3)      | NO   | PRI | NULL    |       |
| title_name | varchar(40)  | NO   |     | NULL    |       |
| type       | varchar(10)  | YES  |     | NULL    |       |
| pub_id     | char(3)      | NO   |     | NULL    |       |
| pages      | int(11)      | YES  |     | NULL    |       |
| price      | decimal(5,2) | YES  |     | NULL    |       |
| sales      | int(11)      | YES  |     | NULL    |       |
| pubdate    | date         | YES  |     | NULL    |       |
| contract   | smallint(6)  | NO   |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
9 rows in set (0.006 sec)
```

```
INSERT INTO titles (title_id,title_name,type,pub_id,pages,price,sales,pubdate,contract)
 values ('T13',"What Are The Civilian Applications?",'history','P03','802','29.99','10467','1999-
05-31','1');
```



```
MariaDB [sparksql]> select * from titles;
+----------+-----------------------------------+-----------+--------+-------+-------+---------+------------+----------+
| title_id | title_name                        | type      | pub_id | pages | price | sales   | pubdate    | contract |
+----------+-----------------------------------+-----------+--------+-------+-------+---------+------------+----------+
| T01      | 1977!                             | history   | P01    |   107 | 21.99 |     566 | 2000-08-01 |        1 |
| T02      | 200 Years of German Humor         | history   | P03    |    14 | 19.95 |    9566 | 1998-04-01 |        1 |
| T03      | Ask Your System Administrator     | computer  | P02    |  1226 | 39.95 |   25667 | 2000-09-01 |        1 |
| T04      | But I Did It Unconsciously        | psychology| P04    |   510 | 12.99 |   13001 | 1999-05-31 |        1 |
| T05      | Exchange of Platitudes            | psychology| P04    |   201 |  6.95 |  201440 | 2001-01-01 |        1 |
| T06      | How About Never?                  | biography | P01    |   473 | 19.95 |   11320 | 2000-07-31 |        1 |
| T07      | I Blame My Mother                 | biography | P03    |   333 | 23.95 | 1500200 | 1999-10-01 |        1 |
| T08      | Just Wait Until After School      | children  | P04    |    86 | 10.00 |    4095 | 2001-06-01 |        1 |
| T09      | Kiss My Boo-Boo                   | children  | P04    |    22 | 13.95 |    5000 | 2002-05-31 |        1 |
| T10      | Not Without My Faberge Egg        | biography | P01    |  NULL |  NULL |    NULL | NULL       |        0 |
| T11      | Perhaps It's a Glandular Problem  | psychology| P04    |   826 |  7.99 |   94123 | 2000-11-30 |        1 |
| T12      | Spontaneous, Not Annoying         | biography | P01    |   507 | 12.99 |  100001 | 2000-08-31 |        1 |
| T13      | What Are The Civilian Applications? | history | P03    |   802 | 29.99 |   10467 | 1999-05-31 |        1 |
+----------+-----------------------------------+-----------+--------+-------+-------+---------+------------+----------+
13 rows in set (0.000 sec)
```

```
CREATE TABLE title_authors (
 title_id CHAR(3) NOT NULL,
 au_id CHAR(3) NOT NULL,
 au_order SMALLINT NOT NULL,
 royalty_share DECIMAL(5,2) NOT NULL,
 CONSTRAINT title_authors_pk
 PRIMARY KEY (title_id, au_id),
 CONSTRAINT title_authors_fk1
 FOREIGN KEY (title_id)
 REFERENCES titles(title_id),
 CONSTRAINT title_authors_fk2
 FOREIGN KEY (au_id)
```

REFERENCES authors (au_id));

```
MariaDB [sparksql]> DESC title_authors;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| title_id     | char(3)      | NO   | PRI | NULL    |       |
| au_id        | char(3)      | NO   | PRI | NULL    |       |
| au_order     | smallint(6)  | NO   |     | NULL    |       |
| royalty_share | decimal(5,2) | NO   |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
4 rows in set (0.009 sec)
```

INSERT INTO title_authors (title_id,au_id,au_order,royalty_share)
values ('T13','A01',1,1.00);

```
MariaDB [sparksql]> SELECT * FROM title_authors;
+----------+-------+----------+---------------+
| title_id | au_id | au_order | royalty_share |
+----------+-------+----------+---------------+
| T01      | A01   |        1 |          1.00 |
| T02      | A01   |        1 |          1.00 |
| T03      | A05   |        1 |          1.00 |
| T04      | A03   |        1 |          0.60 |
| T04      | A04   |        2 |          0.40 |
| T05      | A04   |        1 |          1.00 |
| T06      | A02   |        1 |          1.00 |
| T07      | A02   |        1 |          0.50 |
| T07      | A04   |        2 |          0.50 |
| T08      | A06   |        1 |          1.00 |
| T09      | A06   |        1 |          1.00 |
| T10      | A02   |        1 |          1.00 |
| T11      | A03   |        2 |          0.30 |
| T11      | A04   |        3 |          0.30 |
| T11      | A06   |        1 |          0.40 |
| T12      | A02   |        1 |          1.00 |
| T13      | A01   |        1 |          1.00 |
+----------+-------+----------+---------------+
17 rows in set (0.001 sec)
```

CREATE TABLE royalties (
title_id CHAR(3) NOT NULL,
advance DECIMAL(9,2),
royalty_rate DECIMAL(5,2),
CONSTRAINT royalties_pk PRIMARY KEY (title_id),
CONSTRAINT royalties_title_id_fk
FOREIGN KEY (title_id)
REFERENCES titles(title_id));

```
MariaDB [sparksql]> DESC royalties;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| title_id    | char(3)     | NO   | PRI | NULL    |       |
| advance     | decimal(9,2)| YES  |     | NULL    |       |
| royalty_rate| decimal(5,2)| YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.007 sec)
```

INSERT INTO royalties (title_id,advance,royalty_rate)
 values ('T13',20000.00,0.06);

```
MariaDB [sparksql]> SELECT * FROM royalties;
+----------+------------+--------------+
| title_id | advance    | royalty_rate |
+----------+------------+--------------+
| T01      |   10000.00 |         0.05 |
| T02      |    1000.00 |         0.06 |
| T03      |   15000.00 |         0.07 |
| T04      |   20000.00 |         0.08 |
| T05      |  100000.00 |         0.09 |
| T06      |   20000.00 |         0.08 |
| T07      | 1000000.00 |         0.11 |
| T08      |       0.00 |         0.04 |
| T09      |       0.00 |         0.05 |
| T10      |       NULL |         NULL |
| T11      |  100000.00 |         0.07 |
| T12      |   50000.00 |         0.09 |
| T13      |   20000.00 |         0.06 |
+----------+------------+--------------+
13 rows in set (0.001 sec)
```

CREATE TABLE test_scores (
 name varchar(20), test varchar(20), score tinyint);

```
MariaDB [sparksql]> DESC test_scores;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| name  | varchar(20) | YES  |     | NULL    |       |
| test  | varchar(20) | YES  |     | NULL    |       |
| score | tinyint(4)  | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set (0.007 sec)
```

INSERT INTO test_scores VALUES
 ("Steve", "SQL", 75),

("Robert", "SQL", 43),
("Tracy", "SQL", 56),
("Tatiana", "SQL", 87),
("Steve", "Tuning", 83),
("Robert", "Tuning", 31),
("Tracy", "Tuning", 88),
("Tatiana", "Tuning", 83);

```
MariaDB [sparksql]> SELECT * FROM test_scores;
+---------+--------+-------+
| name    | test   | score |
+---------+--------+-------+
| Steve   | SQL    |    75 |
| Robert  | SQL    |    43 |
| Tracy   | SQL    |    56 |
| Tatiana | SQL    |    87 |
| Steve   | Tuning |    83 |
| Robert  | Tuning |    31 |
| Tracy   | Tuning |    88 |
| Tatiana | Tuning |    83 |
+---------+--------+-------+
8 rows in set (0.001 sec)
```

CREATE TABLE IF NOT EXISTS emp (
empno DECIMAL(4), ename VARCHAR(10), job VARCHAR(9),
mgr DECIMAL(4), hiredate CHAR(10), sal DECIMAL(7,2),
comm DECIMAL(7,2), deptno DECIMAL(2),
CONSTRAINT pk_emp PRIMARY KEY (empno));

```
MariaDB [sparksql]> DESC emp;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| empno    | decimal(4,0) | NO   | PRI | NULL    |       |
| ename    | varchar(10)  | YES  |     | NULL    |       |
| job      | varchar(9)   | YES  |     | NULL    |       |
| mgr      | decimal(4,0) | YES  |     | NULL    |       |
| hiredate | char(10)     | YES  |     | NULL    |       |
| sal      | decimal(7,2) | YES  |     | NULL    |       |
| comm     | decimal(7,2) | YES  |     | NULL    |       |
| deptno   | decimal(2,0) | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
8 rows in set (0.009 sec)
```

INSERT INTO emp VALUES
(7369, 'SMITH', 'CLERK', 7902, '1980-12-17', 800, NULL, 20),

```
(7499, 'ALLEN', 'SALESMAN', 7698, '1981-02-20', 1600, 300, 30),
(7521, 'WARD', 'SALESMAN', 7698, '1981-02-22', 1250, 500, 30),
(7566, 'JONES', 'MANAGER', 7839, '1981-04-02', 2975, NULL, 20),
(7654, 'MARTIN', 'SALESMAN', 7698, '1981-09-28', 1250, 1400,30),
(7698, 'BLAKE', 'MANAGER', 7839, '1981-05-01', 2850, NULL, 30),
(7782, 'CLARK', 'MANAGER', 7839, '1981-06-09', 2450, NULL, 10),
(7788, 'SCOTT', 'ANALYST', 7566, '1987-07-13', 3000, NULL, 20),
(7839, 'KING', 'PRESIDENT', NULL, '1981-11-17', 5000, NULL, 10),
(7844, 'TURNER', 'SALESMAN', 7698, '1981-09-08', 1500, 0, 30),
(7876, 'ADAMS', 'CLERK', 7788, '1987-07-13', 1100, NULL, 20),
(7900, 'JAMES', 'CLERK', 7698, '1981-12-03', 950, NULL, 30),
(7902, 'FORD', 'ANALYST', 7566, '1981-12-03', 3000, NULL, 20),
(7934, 'MILLER', 'CLERK', 7782, '1982-01-23', 1300, NULL, 10) ;
```

```
MariaDB [sparksql]> SELECT * FROM emp;
+-------+--------+-----------+------+------------+---------+---------+--------+
| empno | ename  | job       | mgr  | hiredate   | sal     | comm    | deptno |
+-------+--------+-----------+------+------------+---------+---------+--------+
|  7369 | SMITH  | CLERK     | 7902 | 1980-12-17 |  800.00 |    NULL |     20 |
|  7499 | ALLEN  | SALESMAN  | 7698 | 1981-02-20 | 1600.00 |  300.00 |     30 |
|  7521 | WARD   | SALESMAN  | 7698 | 1981-02-22 | 1250.00 |  500.00 |     30 |
|  7566 | JONES  | MANAGER   | 7839 | 1981-04-02 | 2975.00 |    NULL |     20 |
|  7654 | MARTIN | SALESMAN  | 7698 | 1981-09-28 | 1250.00 | 1400.00 |     30 |
|  7698 | BLAKE  | MANAGER   | 7839 | 1981-05-01 | 2850.00 |    NULL |     30 |
|  7782 | CLARK  | MANAGER   | 7839 | 1981-06-09 | 2450.00 |    NULL |     10 |
|  7788 | SCOTT  | ANALYST   | 7566 | 1987-07-13 | 3000.00 |    NULL |     20 |
|  7839 | KING   | PRESIDENT | NULL | 1981-11-17 | 5000.00 |    NULL |     10 |
|  7844 | TURNER | SALESMAN  | 7698 | 1981-09-08 | 1500.00 |    0.00 |     30 |
|  7876 | ADAMS  | CLERK     | 7788 | 1987-07-13 | 1100.00 |    NULL |     20 |
|  7900 | JAMES  | CLERK     | 7698 | 1981-12-03 |  950.00 |    NULL |     30 |
|  7902 | FORD   | ANALYST   | 7566 | 1981-12-03 | 3000.00 |    NULL |     20 |
|  7934 | MILLER | CLERK     | 7782 | 1982-01-23 | 1300.00 |    NULL |     10 |
+-------+--------+-----------+------+------------+---------+---------+--------+
14 rows in set (0.001 sec)
```

```
CREATE TABLE new_publishers (
 pub_id CHAR(3) NOT NULL,
 pub_name VARCHAR(20) NOT NULL,
 city VARCHAR(15) NOT NULL,
 state CHAR(2),
 country VARCHAR(15) NOT NULL,
 CONSTRAINT publishers_pk PRIMARY KEY (pub_id));
```

```
MariaDB [sparksql]> DESC new_publishers;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| pub_id   | char(3)     | NO   | PRI | NULL    |       |
| pub_name | varchar(20) | NO   |     | NULL    |       |
| city     | varchar(15) | NO   |     | NULL    |       |
| state    | char(2)     | YES  |     | NULL    |       |
| country  | varchar(15) | NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
5 rows in set (0.008 sec)
```

```
INSERT INTO new_publishers (pub_id,pub_name,city,state,country)
 values ('P11','Community Press','Los Angeles','CA','USA'),
 ('P12','Wiley and Sons','Pittsburgh','PA','USA'),
INSERT INTO new_publishers (pub_id,pub_name,city,country)
 VALUES ('P13','Verlang','Berlin','Germany'), ('P14','Springer','Berlin','Germany');
```

```
MariaDB [sparksql]> SELECT * FROM new_publishers;
+--------+----------------+-------------+-------+---------+
| pub_id | pub_name       | city        | state | country |
+--------+----------------+-------------+-------+---------+
| P11    | Community Press | Los Angeles | CA    | USA     |
| P12    | Wiley and Sons | Pittsburgh  | PA    | USA     |
| P13    | Verlang        | Berlin      | NULL  | Germany |
| P14    | Springer       | Berlin      | NULL  | Germany |
+--------+----------------+-------------+-------+---------+
4 rows in set (0.000 sec)
```

```
CREATE TABLE employee (
 emp_id CHAR(3) NOT NULL,
 emp_name VARCHAR(40) NOT NULL,
 boss_id CHAR(3),
 CONSTRAINT pk_employee PRIMARY KEY (emp_id),
 CONSTRAINT emp_id_chk
 CHECK ((SUBSTR(emp_id, 1, 1) = 'E') AND
 (CAST(SUBSTR(emp_id, 2, 2) AS SIGNED) BETWEEN 0 AND 99)),
 CONSTRAINT boss_id_chk
 CHECK ((SUBSTR(boss_id, 1, 1) = 'E') AND (CAST(SUBSTR(boss_id, 2, 2)
 AS SIGNED) BETWEEN 0 AND 99)));
```

```
MariaDB [sparksql]> DESC employee;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| emp_id   | char(3)     | NO   | PRI | NULL    |       |
| emp_name | varchar(40) | NO   |     | NULL    |       |
| boss_id  | char(3)     | NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
3 rows in set (0.010 sec)
```

```
INSERT INTO employee (emp_id,emp_name)
 VALUES ('E01','Lord Copper');
INSERT INTO employee (emp_id,emp_name,boss_id)
 VALUES ('E02','Jocelyn Hitchcock','E01'),
 ('E03','Mr. Salter','E01'),('E04','William Boot','E03'),('E05','Mr. Corker','E03');
```

```
MariaDB [sparksql]> SELECT * FROM employee;
+--------+-------------------+---------+
| emp_id | emp_name          | boss_id |
+--------+-------------------+---------+
| E01    | Lord Copper       | NULL    |
| E02    | Jocelyn Hitchcock | E01     |
| E03    | Mr. Salter        | E01     |
| E04    | William Boot      | E03     |
| E05    | Mr. Corker        | E03     |
+--------+-------------------+---------+
5 rows in set (0.000 sec)
```

Getting data from a dataset

SELECT y FROM

1. List the cities in which the authors live

```
SELECT city FROM authors;
```

```
MariaDB [sparksql]> SELECT city FROM authors;
+---------------+
| city          |
+---------------+
| Bronx         |
| Boulder       |
| San Francisco |
| San Francisco |
| New York      |
| Palo Alto     |
| Sarasota      |
+---------------+
7 rows in set (0.003 sec)
```

2. List the name, surname, city, and state of the authors.

SELECT au_fname, au_lname, city, state FROM authors;

```
MariaDB [sparksql]> SELECT au_fname, au_lname, city, state FROM authors;
+-----------+-------------+---------------+-------+
| au_fname  | au_lname    | city          | state |
+-----------+-------------+---------------+-------+
| Sarah     | Buchman     | Bronx         | NY    |
| Wendy     | Heydemark   | Boulder       | CO    |
| Hallie    | Hull        | San Francisco | CA    |
| Klee      | Hull        | San Francisco | CA    |
| Christian | Kells       | New York      | NY    |
|           | Kellsey     | Palo Alto     | CA    |
| Paddy     | O'Furniture | Sarasota      | FL    |
+-----------+-------------+---------------+-------+
7 rows in set (0.000 sec)
```

3. List city, state, country of publisher

SELECT city, state, country FROM publishers;

```
MariaDB [sparksql]> SELECT city, state, country FROM publishers;
+---------------+-------+---------+
| city          | state | country |
+---------------+-------+---------+
| New York      | NY    | USA     |
| San Francisco | CA    | USA     |
| Hamburg       | NULL  | Germany |
| Berkeley      | CA    | USA     |
+---------------+-------+---------+
4 rows in set (0.004 sec)
```

4. List all columns of the originator table.

SELECT * FROM authors;

```
MariaDB [sparksql]> SELECT * FROM authors;
+-------+----------+------------+--------------+---------------------+--------------+-------+-------+
| au_id | au_fname | au_lname   | phone        | address             | city         | state | zip   |
+-------+----------+------------+--------------+---------------------+--------------+-------+-------+
| A01   | Sarah    | Buchman    | 718-496-7223 | 75 West 205 St      | Bronx        | NY    | 10468 |
| A02   | Wendy    | Heydemark  | 303-986-7020 | 2922 Baseline Rd    | Boulder      | CO    | 80303 |
| A03   | Hallie   | Hull       | 415-549-4278 | 3800 Waldo Ave, #14F | San Francisco | CA    | 94123 |
| A04   | Klee     | Hull       | 415-549-4278 | 3800 Waldo Ave, #14F | San Francisco | CA    | 94123 |
| A05   | Christian | Kells     | 212-771-4680 | 114 Horatio St      | New York     | NY    | 10014 |
| A06   |          | Kellsey    | 650-836-7128 | 390 Serra Mall      | Palo Alto    | CA    | 94305 |
| A07   | Paddy    | O'Furniture | 941-925-0752 | 1442 Main St        | Sarasota     | FL    | 34236 |
+-------+----------+------------+--------------+---------------------+--------------+-------+-------+
7 rows in set (0.001 sec)
```

Creating aliases with AS

5. List authors' first name, last name, city, state, and postcode by changing the name of all columns except state.

```
SELECT au_fname AS "First name",
 au_lname AS "Last name",
 city AS "City",
 state,
 zip "Postal code"
 FROM authors;
```

```
MariaDB [sparksql]> SELECT au_fname AS "First name",
    -> au_lname AS "Last name",
    -> city AS "City",
    -> state,
    -> zip "Postal code"
    -> FROM authors;
+------------+-------------+---------------+-------+-------------+
| First name | Last name   | City          | state | Postal code |
+------------+-------------+---------------+-------+-------------+
| Sarah      | Buchman     | Bronx         | NY    | 10468       |
| Wendy      | Heydemark   | Boulder       | CO    | 80303       |
| Hallie     | Hull        | San Francisco | CA    | 94123       |
| Klee       | Hull        | San Francisco | CA    | 94123       |
| Christian  | Kells       | New York      | NY    | 10014       |
|            | Kellsey     | Palo Alto     | CA    | 94305       |
| Paddy      | O'Furniture | Sarasota      | FL    | 34236       |
+------------+-------------+---------------+-------+-------------+
7 rows in set (0.003 sec)
```

Elimination of duplicates with DISTINCT

6. List states in which the authors live

```
SELECT state FROM authors;
```

```
MariaDB [sparksql]> SELECT state FROM authors;
+-------+
| state |
+-------+
| NY    |
| CO    |
| CA    |
| CA    |
| NY    |
| CA    |
| FL    |
+-------+
7 rows in set (0.000 sec)
```

7. List states where authors live with no duplicates

SELECT DISTINCT state FROM authors;

```
MariaDB [sparksql]> SELECT DISTINCT state FROM authors;
+-------+
| state |
+-------+
| NY    |
| CO    |
| CA    |
| FL    |
+-------+
4 rows in set (0.006 sec)
```

8. List cities and states in which the authors live

SELECT city, state FROM authors;

```
MariaDB [sparksql]> SELECT city, state FROM authors;
+---------------+-------+
| city          | state |
+---------------+-------+
| Bronx         | NY    |
| Boulder       | CO    |
| San Francisco | CA    |
| San Francisco | CA    |
| New York      | NY    |
| Palo Alto     | CA    |
| Sarasota      | FL    |
+---------------+-------+
7 rows in set (0.001 sec)
```

9. List cities and states where authors live without duplicates.

```
SELECT DISTINCT city, state FROM authors;
```

```
MariaDB [sparksql]> SELECT DISTINCT city, state FROM authors;
+---------------+-------+
| city          | state |
+---------------+-------+
| Bronx         | NY    |
| Boulder       | CO    |
| San Francisco | CA    |
| New York      | NY    |
| Palo Alto     | CA    |
| Sarasota      | FL    |
+---------------+-------+
6 rows in set (0.002 sec)
```

Sorting by ORDER BY

10. List name, surname, city, and state of the authors in ascending order.

```
SELECT au_fname, au_lname, city, state
 FROM authors
 ORDER BY au_lname ASC;
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, city, state
    -> FROM authors
    -> ORDER BY au_lname ASC;
+-----------+-------------+---------------+-------+
| au_fname  | au_lname    | city          | state |
+-----------+-------------+---------------+-------+
| Sarah     | Buchman     | Bronx         | NY    |
| Wendy     | Heydemark   | Boulder       | CO    |
| Hallie    | Hull        | San Francisco | CA    |
| Klee      | Hull        | San Francisco | CA    |
| Christian | Kells       | New York      | NY    |
|           | Kellsey     | Palo Alto     | CA    |
| Paddy     | O'Furniture | Sarasota      | FL    |
+-----------+-------------+---------------+-------+
7 rows in set (0.003 sec)
```

11. List name, surname, city, and state of the authors in descending order of precedence.

```
SELECT au_fname, au_lname, city, state
 FROM authors
ORDER BY au_lname DESC;
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, city, state
    -> FROM authors
    -> ORDER BY au_lname DESC;
+-----------+-------------+---------------+-------+
| au_fname  | au_lname    | city          | state |
+-----------+-------------+---------------+-------+
| Paddy     | O'Furniture | Sarasota      | FL    |
|           | Kellsey     | Palo Alto     | CA    |
| Christian | Kells       | New York      | NY    |
| Hallie    | Hull        | San Francisco | CA    |
| Klee      | Hull        | San Francisco | CA    |
| Wendy     | Heydemark   | Boulder       | CO    |
| Sarah     | Buchman     | Bronx         | NY    |
+-----------+-------------+---------------+-------+
7 rows in set (0.004 sec)
```

12. List name, surname, city, and state of authors in ascending order by state and descending order by city.

```
SELECT au_fname, au_lname, city, state
 FROM authors
 ORDER BY state ASC, city DESC;
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, city, state
    -> FROM authors
    -> ORDER BY state ASC, city DESC;
+-----------+-------------+---------------+-------+
| au_fname  | au_lname    | city          | state |
+-----------+-------------+---------------+-------+
| Hallie    | Hull        | San Francisco | CA    |
| Klee      | Hull        | San Francisco | CA    |
|           | Kellsey     | Palo Alto     | CA    |
| Wendy     | Heydemark   | Boulder       | CO    |
| Paddy     | O'Furniture | Sarasota      | FL    |
| Christian | Kells       | New York      | NY    |
| Sarah     | Buchman     | Bronx         | NY    |
+-----------+-------------+---------------+-------+
7 rows in set (0.001 sec)
```

13. Sort by a column that does not appear in the table (zip)

```
SELECT city, state
 FROM authors
ORDER BY zip ASC;
```

```
MariaDB [sparksql]> SELECT city, state
    -> FROM authors
    -> ORDER BY zip ASC;
+---------------+-------+
| city          | state |
+---------------+-------+
| New York      | NY    |
| Bronx         | NY    |
| Sarasota      | FL    |
| Boulder       | CO    |
| San Francisco | CA    |
| San Francisco | CA    |
| Palo Alto     | CA    |
+---------------+-------+
7 rows in set (0.001 sec)
```

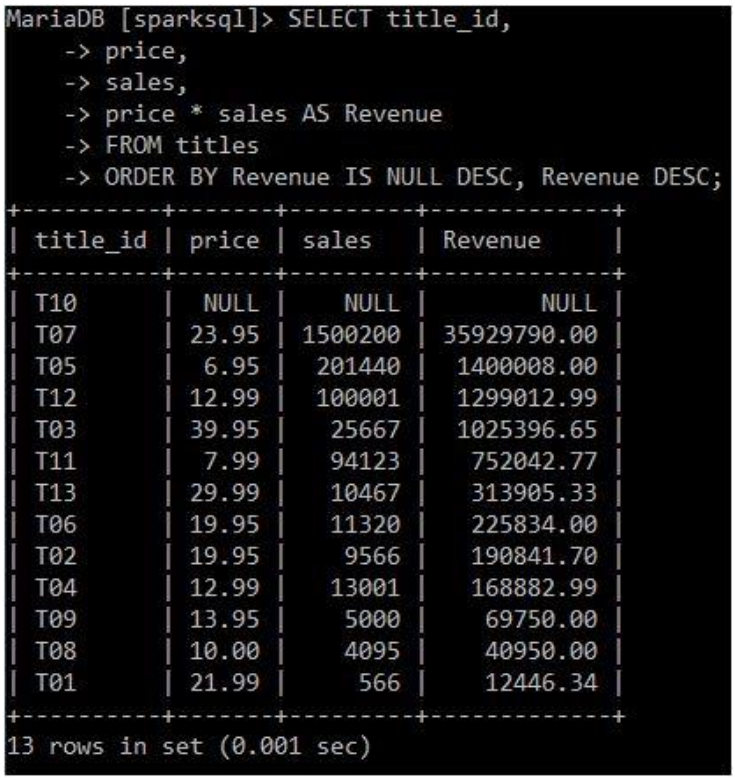
14. Sort by an expression associated with a column

```
SELECT title_id,
 price,
 sales,
 price * sales AS Revenue
 FROM titles
 ORDER BY Revenue DESC;
```

```
MariaDB [sparksql]> SELECT title_id,
    -> price,
    -> sales,
    -> price * sales AS Revenue
    -> FROM titles
    -> ORDER BY Revenue DESC;
+----------+-------+---------+-------------+
| title_id | price | sales   | Revenue     |
+----------+-------+---------+-------------+
| T07      | 23.95 | 1500200 | 35929790.00 |
| T05      |  6.95 |  201440 |  1400008.00 |
| T12      | 12.99 |  100001 |  1299012.99 |
| T03      | 39.95 |   25667 |  1025396.65 |
| T11      |  7.99 |   94123 |   752042.77 |
| T13      | 29.99 |   10467 |   313905.33 |
| T06      | 19.95 |   11320 |   225834.00 |
| T02      | 19.95 |    9566 |   190841.70 |
| T04      | 12.99 |   13001 |   168882.99 |
| T09      | 13.95 |    5000 |    69750.00 |
| T08      | 10.00 |    4095 |    40950.00 |
| T01      | 21.99 |     566 |    12446.34 |
| T10      |  NULL |    NULL |        NULL |
+----------+-------+---------+-------------+
13 rows in set (0.006 sec)
```

15. Sort by an expression associated with a column with leading nulls

```
SELECT title_id,
 price,
 sales,
 price * sales AS Revenue
 FROM titles
 ORDER BY Revenue IS NULL DESC, Revenue DESC;
```



```
MariaDB [sparksql]> SELECT title_id,
    -> price,
    -> sales,
    -> price * sales AS Revenue
    -> FROM titles
    -> ORDER BY Revenue IS NULL DESC, Revenue DESC;
+----------+-------+---------+-------------+
| title_id | price | sales   | Revenue     |
+----------+-------+---------+-------------+
| T10      |  NULL |    NULL |        NULL |
| T07      | 23.95 | 1500200 | 35929790.00 |
| T05      |  6.95 |  201440 |  1400008.00 |
| T12      | 12.99 |  100001 |  1299012.99 |
| T03      | 39.95 |   25667 |  1025396.65 |
| T11      |  7.99 |   94123 |   752042.77 |
| T13      | 29.99 |   10467 |   313905.33 |
| T06      | 19.95 |   11320 |   225834.00 |
| T02      | 19.95 |    9566 |   190841.70 |
| T04      | 12.99 |   13001 |   168882.99 |
| T09      | 13.95 |    5000 |    69750.00 |
| T08      | 10.00 |    4095 |    40950.00 |
| T01      | 21.99 |     566 |    12446.34 |
+----------+-------+---------+-------------+
13 rows in set (0.001 sec)
```

Row filtering with WHERE

16. List authors whose surname is not Hull

```
SELECT au_id, au_fname, au_lname
 FROM authors
 WHERE au_lname <> 'Hull';
```

```
MariaDB [sparksql]> SELECT au_id, au_fname, au_lname
    -> FROM authors
    -> WHERE au_lname <> 'Hull';
+-------+-----------+-------------+
| au_id | au_fname  | au_lname    |
+-------+-----------+-------------+
| A06   |           | Kellsey     |
| A05   | Christian | Kells       |
| A07   | Paddy     | O'Furniture |
| A01   | Sarah     | Buchman     |
| A02   | Wendy     | Heydemark   |
+-------+-----------+-------------+
5 rows in set (0.001 sec)
```

17. Listing books without a signed contract

```
SELECT title_name, contract
 FROM titles
 WHERE contract = 0;
```

```
MariaDB [sparksql]> SELECT title_name, contract
    -> FROM titles
    -> WHERE contract = 0;
+-----------------------------+----------+
| title_name                  | contract |
+-----------------------------+----------+
| Not Without My Faberge Egg |        0 |
+-----------------------------+----------+
1 row in set (0.002 sec)
```

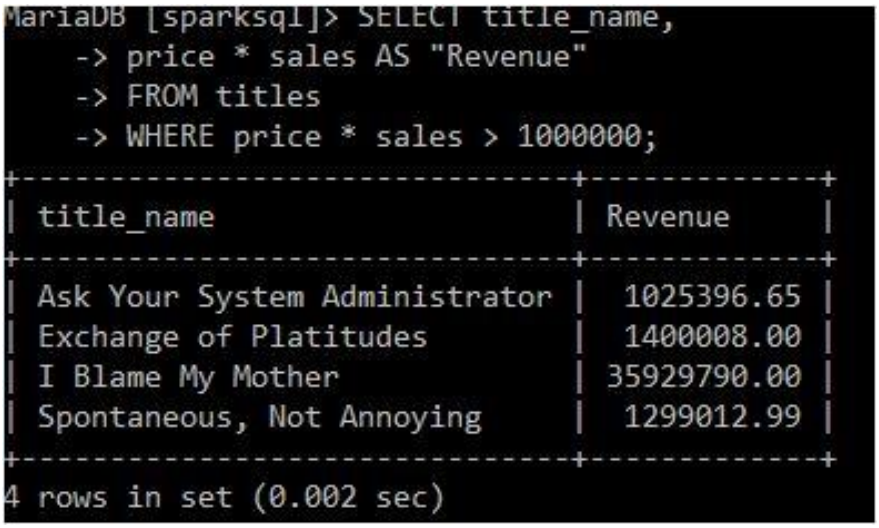
18. List titles published since 2001

```
SELECT title_name, pubdate
 FROM titles
 WHERE pubdate >= '2001-01-01';
```

```
MariaDB [sparksql]> SELECT title_name, pubdate
    -> FROM titles
    -> WHERE pubdate >= '2001-01-01';
+-----------------------------+------------+
| title_name                  | pubdate    |
+-----------------------------+------------+
| Exchange of Platitudes      | 2001-01-01 |
| Just Wait Until After School | 2001-06-01 |
| Kiss My Boo-Boo             | 2002-05-31 |
+-----------------------------+------------+
3 rows in set (0.002 sec)
```

19. List securities that have produced a profit in excess of 1,000,000.

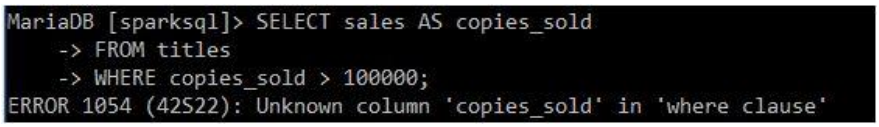
```
SELECT title_name,
 price * sales AS "Revenue"
 FROM titles
 WHERE price * sales > 1000000;
```



```
MariaDB [sparksql]> SELECT title_name,
    -> price * sales AS "Revenue"
    -> FROM titles
    -> WHERE price * sales > 1000000;
+--------------------------------+-------------+
| title_name                     | Revenue     |
+--------------------------------+-------------+
| Ask Your System Administrator  |  1025396.65 |
| Exchange of Platitudes         |  1400008.00 |
| I Blame My Mother              | 35929790.00 |
| Spontaneous, Not Annoying      |  1299012.99 |
+--------------------------------+-------------+
4 rows in set (0.002 sec)
```

20. Error: An alias is made in the SELECT and referenced in WHERE.

```
SELECT sales AS copies_sold
 FROM titles
 WHERE copies_sold > 100000;
```



```
MariaDB [sparksql]> SELECT sales AS copies_sold
    -> FROM titles
    -> WHERE copies_sold > 100000;
ERROR 1054 (42S22): Unknown column 'copies_sold' in 'where clause'
```

Combination and negation of conditions with AND, OR, and NOT

21. Combination and negation of conditions with AND, OR, and NOT

```
SELECT title_name, type, price
 FROM titles
 WHERE type = 'biography' AND price < 20;
```

```
MariaDB [sparksql]> SELECT title_name, type, price
    -> FROM titles
    -> WHERE type = 'biography' AND price < 20;
+--------------------------+-----------+-------+
| title_name               | type      | price |
+--------------------------+-----------+-------+
| How About Never?         | biography | 19.95 |
| Spontaneous, Not Annoying | biography | 12.99 |
+--------------------------+-----------+-------+
2 rows in set (0.002 sec)
```

22. List authors whose surname begins with a letter between H and Z and whose status is not in California.

```
SELECT au_fname, au_lname
 FROM authors
 WHERE au_lname >= 'H'
 AND au_lname <= 'Zz'
 AND state <> 'CA';
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname
    -> FROM authors
    -> WHERE au_lname >= 'H'
    -> AND au_lname <= 'Zz'
    -> AND state <> 'CA';
+-----------+-------------+
| au_fname  | au_lname    |
+-----------+-------------+
| Wendy     | Heydemark   |
| Christian | Kells       |
| Paddy     | O'Furniture |
+-----------+-------------+
3 rows in set (0.001 sec)
```

23. List authors who live in New York State, Colorado, or San Francisco City.

```
SELECT au_fname, au_lname, city, state
 FROM authors
 WHERE state in ('NY', 'CO') OR city='San Francisco';
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, city, state
    -> FROM authors
    -> WHERE state in ('NY', 'CO') OR city='San Francisco';
+-----------+-----------+---------------+-------+
| au_fname  | au_lname  | city          | state |
+-----------+-----------+---------------+-------+
| Sarah     | Buchman   | Bronx         | NY    |
| Wendy     | Heydemark | Boulder       | CO    |
| Hallie    | Hull      | San Francisco | CA    |
| Klee      | Hull      | San Francisco | CA    |
| Christian | Kells     | New York      | NY    |
+-----------+-----------+---------------+-------+
5 rows in set (0.003 sec)
```

24. List all publishers

```
SELECT pub_id, pub_name, state, country
 FROM publishers;
```

```
MariaDB [sparksql]> SELECT pub_id, pub_name, state, country
    -> FROM publishers;
+--------+----------------------+-------+---------+
| pub_id | pub_name             | state | country |
+--------+----------------------+-------+---------+
| P01    | Abatis Publishers    | NY    | USA     |
| P02    | Core Dump Books      | CA    | USA     |
| P03    | Schadenfreude Press  | NULL  | Germany |
| P04    | Tenterhooks Press    | CA    | USA     |
+--------+----------------------+-------+---------+
4 rows in set (0.003 sec)
```

25. List editors living or not living in California

```
SELECT pub_id, pub_name, state, country
 FROM publishers
 WHERE (state = 'CA')
 OR (state <> 'CA');
```

```
MariaDB [sparksql]> SELECT pub_id, pub_name, state, country
    -> FROM publishers
    -> WHERE (state = 'CA')
    -> OR (state <> 'CA');
+--------+-------------------+-------+---------+
| pub_id | pub_name          | state | country |
+--------+-------------------+-------+---------+
| P01    | Abatis Publishers | NY    | USA     |
| P02    | Core Dump Books   | CA    | USA     |
| P04    | Tenterhooks Press | CA    | USA     |
+--------+-------------------+-------+---------+
3 rows in set (0.001 sec)
```

26. List authors who do not live in California

```
SELECT au_fname, au_lname, state
 FROM authors
 WHERE state <> 'CA';
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, state
    -> FROM authors
    -> WHERE state <> 'CA';
+-----------+-------------+-------+
| au_fname  | au_lname    | state |
+-----------+-------------+-------+
| Sarah     | Buchman     | NY    |
| Wendy     | Heydemark   | CO    |
| Christian | Kells       | NY    |
| Paddy     | O'Furniture | FL    |
+-----------+-------------+-------+
4 rows in set (0.000 sec)
```

27. List books whose price is not less than 20 and which have sold more than 15,000 copies

```
SELECT title_name, sales, price
 FROM titles
 WHERE NOT (price < 20)
 AND (sales > 15000);
```

```
MariaDB [sparksql]> SELECT title_name, sales, price
    -> FROM titles
    -> WHERE NOT (price < 20)
    -> AND (sales > 15000);
+-------------------------------+---------+-------+
| title_name                    | sales   | price |
+-------------------------------+---------+-------+
| Ask Your System Administrator |   25667 | 39.95 |
| I Blame My Mother             | 1500200 | 23.95 |
+-------------------------------+---------+-------+
2 rows in set (0.001 sec)
```

28. List books 1) whose topic is history or 2) whose topic is biography and their price is under 20

```
SELECT title_id, type, price
 FROM titles
```

```
WHERE type = 'history'
OR type = 'biography'
AND price < 20;
```

```
MariaDB [sparksql]> SELECT title_id, type, price
    -> FROM titles
    -> WHERE type = 'history'
    -> OR type = 'biography'
    -> AND price < 20;
+----------+-----------+-------+
| title_id | type      | price |
+----------+-----------+-------+
| T01      | history   | 21.99 |
| T02      | history   | 19.95 |
| T06      | biography | 19.95 |
| T12      | biography | 12.99 |
| T13      | history   | 29.99 |
+----------+-----------+-------+
5 rows in set (0.000 sec)
```

29. List books 1) whose topic is history or biography AND 2) their price is under 20

```
SELECT title_id, type, price
 FROM titles
 WHERE (type = 'history'
 OR type = 'biography')
 AND price < 20;
```

```
MariaDB [sparksql]> SELECT title_id, type, price
    -> FROM titles
    -> WHERE (type = 'history'
    -> OR type = 'biography')
    -> AND price < 20;
+----------+-----------+-------+
| title_id | type      | price |
+----------+-----------+-------+
| T02      | history   | 19.95 |
| T06      | biography | 19.95 |
| T12      | biography | 12.99 |
+----------+-----------+-------+
3 rows in set (0.001 sec)
```

30. For debugging

```
SELECT type,
 type = 'history' AS "Hist?",
 type = 'biography' AS "Bio?",
```

```
price,
price < 20 AS "<20?"
FROM titles;
```

```
MariaDB [sparksql]> SELECT type,
    -> type = 'history' AS "Hist?",
    -> type = 'biography' AS "Bio?",
    -> price,
    -> price < 20 AS "<20?"
    -> FROM titles;
+------------+-------+------+-------+------+
| type       | Hist? | Bio? | price | <20? |
+------------+-------+------+-------+------+
| history    |     1 |    0 | 21.99 |    0 |
| history    |     1 |    0 | 19.95 |    1 |
| computer   |     0 |    0 | 39.95 |    0 |
| psychology |     0 |    0 | 12.99 |    1 |
| psychology |     0 |    0 |  6.95 |    1 |
| biography  |     0 |    1 | 19.95 |    1 |
| biography  |     0 |    1 | 23.95 |    0 |
| children   |     0 |    0 | 10.00 |    1 |
| children   |     0 |    0 | 13.95 |    1 |
| biography  |     0 |    1 |  NULL | NULL |
| psychology |     0 |    0 |  7.99 |    1 |
| biography  |     0 |    1 | 12.99 |    1 |
| history    |     1 |    0 | 29.99 |    0 |
+------------+-------+------+-------+------+
13 rows in set (0.001 sec)
```

Pattern matching using LIKE

31. List authors whose surname begins with Kel

```
SELECT au_fname, au_lname
 FROM authors
 WHERE au_lname LIKE 'Kel%';
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname
    -> FROM authors
    -> WHERE au_lname LIKE 'Kel%';
+-----------+----------+
| au_fname  | au_lname |
+-----------+----------+
|           | Kellsey  |
| Christian | Kells    |
+-----------+----------+
2 rows in set (0.004 sec)
```

32. List authors whose surnames have ll (ele, ele) in the third and fourth positions.

```
SELECT au_fname, au_lname
 FROM authors
 WHERE au_lname LIKE '__ll%';
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname
    -> FROM authors
    -> WHERE au_lname LIKE '__ll%';
+-----------+-----------+
| au_fname  | au_lname  |
+-----------+-----------+
|           | Kellsey   |
| Christian | Kells     |
| Hallie    | Hull      |
| Klee      | Hull      |
+-----------+-----------+
4 rows in set (0.001 sec)
```

33. List authors whose postcode starts with 94

```
SELECT au_fname, au_lname, city, state, zip
 FROM authors
 WHERE zip LIKE '94___';
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, city, state, zip
    -> FROM authors
    -> WHERE zip LIKE '94___';
+----------+----------+---------------+-------+-------+
| au_fname | au_lname | city          | state | zip   |
+----------+----------+---------------+-------+-------+
| Hallie   | Hull     | San Francisco | CA    | 94123 |
| Klee     | Hull     | San Francisco | CA    | 94123 |
|          | Kellsey  | Palo Alto     | CA    | 94305 |
+----------+----------+---------------+-------+-------+
3 rows in set (0.001 sec)
```

34. List authors whose phone does not begin with 212, 415, or 303.

```
SELECT au_fname, au_lname, phone
 FROM authors
 WHERE phone NOT LIKE '212-___-____'
 AND phone NOT LIKE '415-___-%'
 AND phone NOT LIKE '303-%';
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, phone
    -> FROM authors
    -> WHERE phone NOT LIKE '212-___-____'
    -> AND phone NOT LIKE '415-___-%'
    -> AND phone NOT LIKE '303-%';
+----------+-------------+--------------+
| au_fname | au_lname    | phone        |
+----------+-------------+--------------+
| Sarah    | Buchman     | 718-496-7223 |
|          | Kellsey     | 650-836-7128 |
| Paddy    | O'Furniture | 941-925-0752 |
+----------+-------------+--------------+
3 rows in set (0.000 sec)
```

35. List books whose title contains a %.

```
SELECT title_name
 FROM titles
 WHERE title_name LIKE '%!%%' ESCAPE '!';
```

```
MariaDB [sparksql]> SELECT title_name
    -> FROM titles
    -> WHERE title_name LIKE '%!%%' ESCAPE '!';
Empty set (0.001 sec)
```

Range filtering with BETWEEN

36. Print names of books containing MO independently of upper/lower case letters

```
SELECT au_fname, au_lname, zip FROM authors
 WHERE zip NOT BETWEEN '20000' AND '89999';
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, zip
    -> FROM authors
    -> WHERE zip NOT BETWEEN '20000' AND '89999';
+-----------+----------+-------+
| au_fname  | au_lname | zip   |
+-----------+----------+-------+
| Sarah     | Buchman  | 10468 |
| Hallie    | Hull     | 94123 |
| Klee      | Hull     | 94123 |
| Christian | Kells    | 10014 |
|           | Kellsey  | 94305 |
+-----------+----------+-------+
5 rows in set (0.001 sec)
```

Trimming characters with TRIM

37. List authors' surnames, removing any capital H at the beginning.

```
SELECT title_id, price
 FROM titles
 WHERE price BETWEEN 10 AND 19.95;
```

```
MariaDB [sparksql]> SELECT title_id, price
    -> FROM titles
    -> WHERE price BETWEEN 10 AND 19.95;
+----------+-------+
| title_id | price |
+----------+-------+
| T02      | 19.95 |
| T04      | 12.99 |
| T06      | 19.95 |
| T08      | 10.00 |
| T09      | 13.95 |
| T12      | 12.99 |
+----------+-------+
6 rows in set (0.001 sec)
```

38. From the titles table, print the title_id columns starting with T1 and having another character, ignoring possible leading and trailing spaces.

```
SELECT title_id, pubdate
 FROM titles
 WHERE pubdate BETWEEN '2000-01-01'
 AND '2000-12-31';
```

```
MariaDB [sparksql]> SELECT title_id, pubdate
    -> FROM titles
    -> WHERE pubdate BETWEEN '2000-01-01'
    -> AND '2000-12-31';
+----------+------------+
| title_id | pubdate    |
+----------+------------+
| T01      | 2000-08-01 |
| T03      | 2000-09-01 |
| T06      | 2000-07-31 |
| T11      | 2000-11-30 |
| T12      | 2000-08-31 |
+----------+------------+
5 rows in set (0.001 sec)
```

39. List books whose price is strictly above 10 and strictly below 19,95 less than 19.95

```
SELECT title_id, price
 FROM titles
 WHERE (price > 10)
 AND (price < 19.95);
```

```
MariaDB [sparksql]> SELECT title_id, price
    -> FROM titles
    -> WHERE (price > 10)
    -> AND (price < 19.95);
+----------+-------+
| title_id | price |
+----------+-------+
| T04      | 12.99 |
| T09      | 13.95 |
| T12      | 12.99 |
+----------+-------+
3 rows in set (0.001 sec)
```

List filtering with IN

40. List authors who do not live in any of these states: New York (NY), New Jersey (NJ), or California (CA). (NY), New Jersey (NJ), or California (CA).

```
SELECT au_fname, au_lname, state
 FROM authors
 WHERE state NOT IN ('NY', 'NJ', 'CA');
```

```
MariaDB [sparksql]> SELECT au_fname, au_lname, state
    -> FROM authors
    -> WHERE state NOT IN ('NY', 'NJ', 'CA');
+----------+-------------+-------+
| au_fname | au_lname    | state |
+----------+-------------+-------+
| Wendy    | Heydemark   | CO    |
| Paddy    | O'Furniture | FL    |
+----------+-------------+-------+
2 rows in set (0.000 sec)
```

41. List the title of the books together with their length for those books whose length is less than 30. The list has to be sorted by the length of the title.

```
SELECT title_id, advance
 FROM royalties
 WHERE advance IN (0.00, 1000.00, 5000.00);
```

```
MariaDB [sparksql]> SELECT title_id, advance
    -> FROM royalties
    -> WHERE advance IN
    -> (0.00, 1000.00, 5000.00);
+----------+---------+
| title_id | advance |
+----------+---------+
| T02      | 1000.00 |
| T08      |    0.00 |
| T09      |    0.00 |
+----------+---------+
3 rows in set (0.003 sec)
```

Find substrings with POSITION (LOCATE)

42. List books that were published on the first day of the year 2000, 2001, or 2002.

```
SELECT title_id, pubdate
 FROM titles
 WHERE pubdate IN
 ('2000-01-01', '2001-01-01', '2002-01-01');
```

```
MariaDB [sparksql]> SELECT title_id, pubdate
    -> FROM titles
    -> WHERE pubdate IN
    -> ('2000-01-01', '2001-01-01', '2002-01-01');
+----------+------------+
| title_id | pubdate    |
+----------+------------+
| T05      | 2001-01-01 |
+----------+------------+
1 row in set (0.001 sec)
```

43. List 1) the titles that contain the letter u in their first 10 positions and 2) the position of the letter u itself. The resulting table has to be sorted in descending order by the position of the letter u.

```
SELECT pub_id, city, state, country
 FROM publishers
 WHERE state = 'CA';
```

```
MariaDB [sparksql]> SELECT pub_id, city, state, country
    -> FROM publishers
    -> WHERE state = 'CA';
+--------+---------------+-------+---------+
| pub_id | city          | state | country |
+--------+---------------+-------+---------+
| P02    | San Francisco | CA    | USA     |
| P04    | Berkeley      | CA    | USA     |
+--------+---------------+-------+---------+
2 rows in set (0.001 sec)
```

44. List publishers who do not live in California

```
SELECT pub_id, city, state, country
FROM publishers
WHERE state <> 'CA';
```

```
MariaDB [sparksql]> SELECT pub_id, city, state, country
    -> FROM publishers
    -> WHERE state <> 'CA';
+--------+----------+-------+---------+
| pub_id | city     | state | country |
+--------+----------+-------+---------+
| P01    | New York | NY    | USA     |
+--------+----------+-------+---------+
1 row in set (0.001 sec)
```

45. Listar editores que no viven en California o cuyo estado es null

```
SELECT pub_id, city, state, country
FROM publishers
WHERE state <> 'CA';
```

```
MariaDB [sparksql]> SELECT pub_id, city, state, country
    -> FROM publishers
    -> WHERE state <> 'CA'
    -> OR state IS NULL;
+--------+----------+-------+---------+
| pub_id | city     | state | country |
+--------+----------+-------+---------+
| P01    | New York | NY    | USA     |
| P03    | Hamburg  | NULL  | Germany |
+--------+----------+-------+---------+
2 rows in set (0.001 sec)
```

46. List books whose subject is biography and their publication date is not null

```
SELECT title_id, type, pubdate
 FROM titles
 WHERE type = 'biography'
 AND pubdate IS NOT NULL;
```

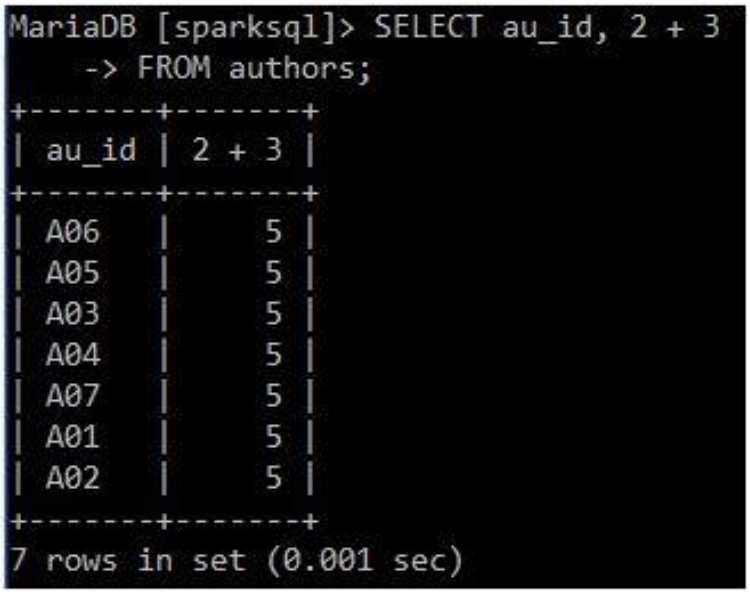
```
MariaDB [sparksql]> SELECT title_id, type, pubdate
    -> FROM titles
    -> WHERE type = 'biography'
    -> AND pubdate IS NOT NULL;
+----------+-----------+------------+
| title_id | type      | pubdate    |
+----------+-----------+------------+
| T06      | biography | 2000-07-31 |
| T07      | biography | 1999-10-01 |
| T12      | biography | 2000-08-31 |
+----------+-----------+------------+
3 rows in set (0.001 sec)
```

Operators and Functions

Creation of derived columns

47. Creation of an artificial derived columna

```
SELECT au_id, 2 + 3
 FROM authors;
```



```
MariaDB [sparksql]> SELECT au_id, 2 + 3
    -> FROM authors;
+-------+-------+
| au_id | 2 + 3 |
+-------+-------+
| A06   |     5 |
| A05   |     5 |
| A03   |     5 |
| A04   |     5 |
| A07   |     5 |
| A01   |     5 |
| A02   |     5 |
+-------+-------+
7 rows in set (0.001 sec)
```

48. List books with a 10% discount

```
SELECT title_id, price, 0.10 AS "Discount", price * (1 - 0.10) AS "New price"
FROM titles;
```

```
MariaDB [sparksql]> SELECT title_id,
    -> price,
    -> 0.10 AS "Discount",
    -> price * (1 - 0.10) AS "New price"
    -> FROM titles;
+----------+-------+----------+-----------+
| title_id | price | Discount | New price |
+----------+-------+----------+-----------+
| T01      | 21.99 |     0.10 |   19.7910 |
| T02      | 19.95 |     0.10 |   17.9550 |
| T03      | 39.95 |     0.10 |   35.9550 |
| T04      | 12.99 |     0.10 |   11.6910 |
| T05      |  6.95 |     0.10 |    6.2550 |
| T06      | 19.95 |     0.10 |   17.9550 |
| T07      | 23.95 |     0.10 |   21.5550 |
| T08      | 10.00 |     0.10 |    9.0000 |
| T09      | 13.95 |     0.10 |   12.5550 |
| T10      |  NULL |     0.10 |      NULL |
| T11      |  7.99 |     0.10 |    7.1910 |
| T12      | 12.99 |     0.10 |   11.6910 |
| T13      | 29.99 |     0.10 |   26.9910 |
+----------+-------+----------+-----------+
13 rows in set (0.003 sec)
```

49. List book advances with negative value.

```
SELECT title_id,
 advance AS "Advance"
 FROM royalties;
```

```
MariaDB [sparksql]> SELECT title_id,
    -> -advance AS "Advance"
    -> FROM royalties;
+----------+-------------+
| title_id | Advance     |
+----------+-------------+
| T01      |   -10000.00 |
| T02      |    -1000.00 |
| T03      |   -15000.00 |
| T04      |   -20000.00 |
| T05      |  -100000.00 |
| T06      |   -20000.00 |
| T07      | -1000000.00 |
| T08      |        0.00 |
| T09      |        0.00 |
| T10      |        NULL |
| T11      |  -100000.00 |
| T12      |   -50000.00 |
| T13      |   -20000.00 |
+----------+-------------+
13 rows in set (0.002 sec)
```

50. Listar libros de biografías junto con las ventas que han generado (venta = price*sales) en orden descendiente

```
SELECT title_id,
 price * sales AS Revenue
 FROM titles
 WHERE type = 'biography'
 ORDER BY Revenue DESC;
```

```
MariaDB [sparksql]> SELECT title_id,
    -> price * sales AS Revenue
    -> FROM titles
    -> WHERE type = 'biography'
    -> ORDER BY Revenue DESC;
+----------+-------------+
| title_id | Revenue     |
+----------+-------------+
| T07      | 35929790.00 |
| T12      |  1299012.99 |
| T06      |   225834.00 |
| T10      |        NULL |
+----------+-------------+
4 rows in set (0.001 sec)
```

51. List books with the number of pages divided by 10 as integers and as decimals

```
SELECT title_id,
 pages,
 pages DIV 10 AS "pages/10",
 pages/10.0 AS "pages/10.0"
 FROM titles;
```

```
MariaDB [sparksql]> SELECT title_id,
    -> pages,
    -> pages DIV 10 AS "pages/10",
    -> pages/10.0 AS "pages/10.0"
    -> FROM titles;
+----------+-------+----------+------------+
| title_id | pages | pages/10 | pages/10.0 |
+----------+-------+----------+------------+
| T01      |   107 |       10 |    10.7000 |
| T02      |    14 |        1 |     1.4000 |
| T03      |  1226 |      122 |   122.6000 |
| T04      |   510 |       51 |    51.0000 |
| T05      |   201 |       20 |    20.1000 |
| T06      |   473 |       47 |    47.3000 |
| T07      |   333 |       33 |    33.3000 |
| T08      |    86 |        8 |     8.6000 |
| T09      |    22 |        2 |     2.2000 |
| T10      |  NULL |     NULL |       NULL |
| T11      |   826 |       82 |    82.6000 |
| T12      |   507 |       50 |    50.7000 |
| T13      |   802 |       80 |    80.2000 |
+----------+-------+----------+------------+
13 rows in set (0.002 sec)
```

Order of evaluation

52. Example of precedence of arithmetic operators

```
SELECT 2 + 3 * 4 AS "2+3*4",
 (2 + 3) * 4 AS "(2+3)*4",
 6 / 2 * 3 AS "6/2*3",
 6 / (2 * 3) AS "6/(2*3)" ;
```

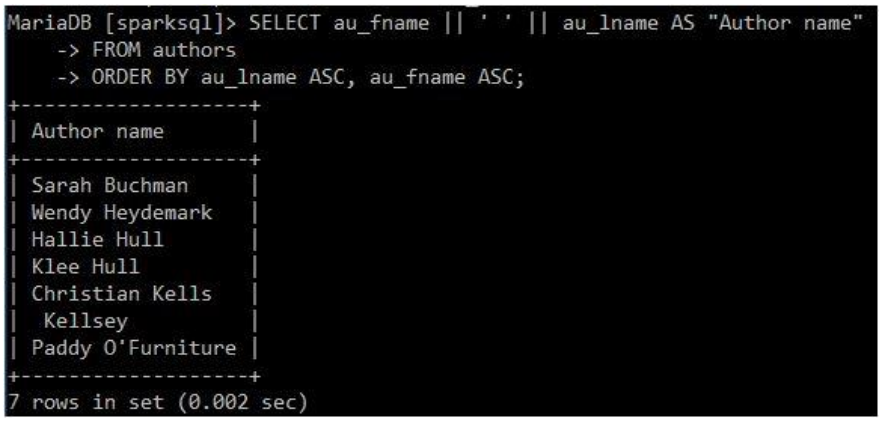
```
MariaDB [sparksql]> SELECT 2 + 3 * 4 AS "2+3*4",
    -> (2 + 3) * 4 AS "(2+3)*4",
    -> 6 / 2 * 3 AS "6/2*3",
    -> 6 / (2 * 3) AS "6/(2*3)" ;
+-------+---------+--------+---------+
| 2+3*4 | (2+3)*4 | 6/2*3  | 6/(2*3) |
+-------+---------+--------+---------+
|    14 |      20 | 9.0000 |  1.0000 |
+-------+---------+--------+---------+
1 row in set (0.002 sec)
```

Concatenation with ||

```
SET sql_mode := 'PIPES_AS_CONCAT';
```

53. List authors with first name and surname concatenated in a single column

```
SELECT au_fname || ' ' || au_lname AS "Author name"
 FROM authors
 ORDER BY au_lname ASC, au_fname ASC;
```



```
MariaDB [sparksql]> SELECT au_fname || ' ' || au_lname AS "Author name"
    -> FROM authors
    -> ORDER BY au_lname ASC, au_fname ASC;
+-------------------+
| Author name       |
+-------------------+
| Sarah Buchman     |
| Wendy Heydemark   |
| Hallie Hull       |
| Klee Hull         |
| Christian Kells   |
|  Kellsey          |
| Paddy O'Furniture |
+-------------------+
7 rows in set (0.002 sec)
```

54. List sales of biographies in descending order of sales

```
SELECT SALES
 || ' copies sold of title '
 || title_id
 AS 'Biography sales'
 FROM titles
 WHERE type = 'biography'
 AND sales IS NOT NULL
 ORDER BY sales DESC;
```

```
MariaDB [sparksql]> SELECT SALES
    -> || ' copies sold of title '
    -> || title_id
    -> AS 'Biography sales'
    -> FROM titles
    -> WHERE type = 'biography'
    -> AND sales IS NOT NULL
    -> ORDER BY sales DESC;
+----------------------------------+
| Biography sales                  |
+----------------------------------+
| 1500200 copies sold of title T07 |
| 100001 copies sold of title T12  |
| 11320 copies sold of title T06   |
+----------------------------------+
3 rows in set (0.003 sec)
```

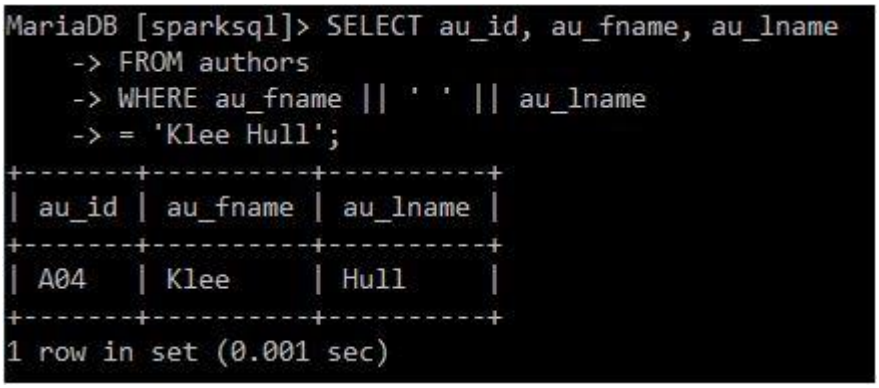
55. List biographies in descending order of publication date

```
SELECT 'Title '
 || title_name
 || ' published on '
 || pubdate
 AS "Biography sales"
 FROM titles
 WHERE type = 'biography' AND pubdate IS NOT NULL
 ORDER BY pubdate DESC;
```

```
MariaDB [sparksql]> SELECT 'Title '
    -> || title_name
    -> || ' published on '
    -> || pubdate
    -> AS "Biography sales"
    -> FROM titles
    -> WHERE type = 'biography' AND pubdate IS NOT NULL
    -> ORDER BY pubdate DESC;
+-----------------------------------------------------+
| Biography sales                                     |
+-----------------------------------------------------+
| Title Spontaneous, Not Annoying published on 2000-08-31 |
| Title How About Never? published on 2000-07-31      |
| Title I Blame My Mother published on 1999-10-01     |
+-----------------------------------------------------+
3 rows in set (0.001 sec)
```

56. List authors whose first and last name is Klee Hull

```
SELECT au_id, au_fname, au_lname
 FROM authors
 WHERE au_fname || ' ' || au_lname
 = 'Klee Hull';
```

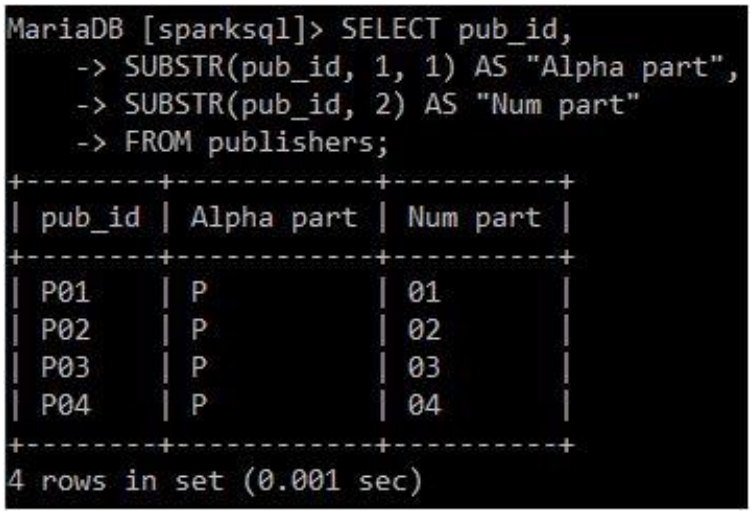


```
MariaDB [sparksql]> SELECT au_id, au_fname, au_lname
    -> FROM authors
    -> WHERE au_fname || ' ' || au_lname
    -> = 'Klee Hull';
+-------+----------+----------+
| au_id | au_fname | au_lname |
+-------+----------+----------+
| A04   | Klee     | Hull     |
+-------+----------+----------+
1 row in set (0.001 sec)
```

Substring extraction with Substring (Substr)

57. Split the primary key of the editors into alphabetical and numerical parts.

```
SELECT pub_id,
 SUBSTR(pub_id, 1, 1) AS "Alpha part",
 SUBSTR(pub_id, 2) AS "Num part"
 FROM publishers;
```

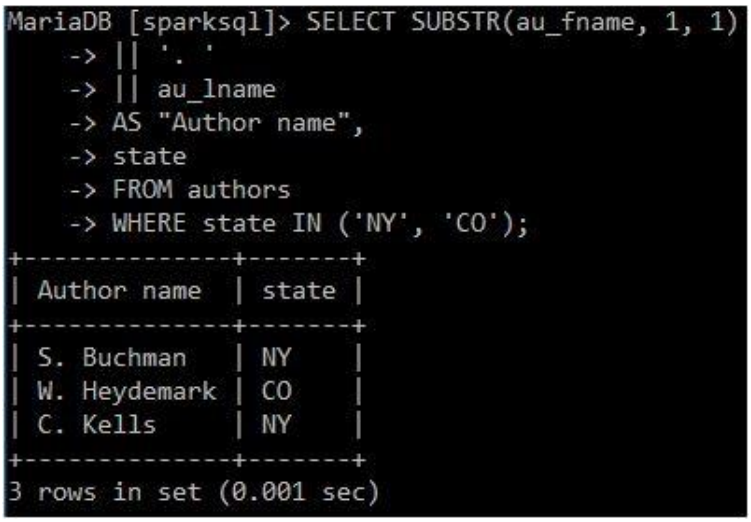


```
MariaDB [sparksql]> SELECT pub_id,
    -> SUBSTR(pub_id, 1, 1) AS "Alpha part",
    -> SUBSTR(pub_id, 2) AS "Num part"
    -> FROM publishers;
+--------+------------+----------+
| pub_id | Alpha part | Num part |
+--------+------------+----------+
| P01    | P          | 01       |
| P02    | P          | 02       |
| P03    | P          | 03       |
| P04    | P          | 04       |
+--------+------------+----------+
4 rows in set (0.001 sec)
```

58. List the initial letter of the first and last name of authors whose state is NY or

CO.

```
SELECT SUBSTR(au_fname, 1, 1)
 || '. '
 || au_lname
 AS "Author name",
 state
 FROM authors
 WHERE state IN ('NY', 'CO');
```



59. List authors whose phone number starts with 415

```
SELECT au_fname, au_lname, phone
 FROM authors
 WHERE SUBSTR(phone, 1, 3)='415';
```

LOWER and UPPER

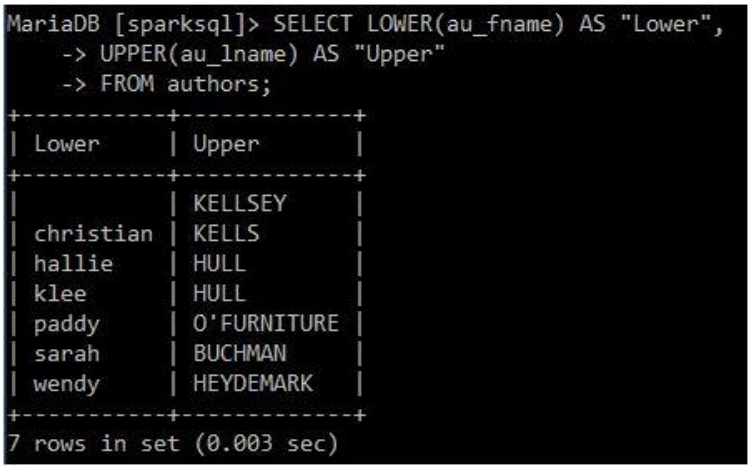
60. For each author, print first name in lower case and surname in upper case.

```
SELECT LOWER(au_fname) AS "Lower",
 UPPER(au_lname) AS "Upper"
 FROM authors;
```

```
MariaDB [sparksql]> SELECT LOWER(au_fname) AS "Lower",
    -> UPPER(au_lname) AS "Upper"
    -> FROM authors;
+-----------+-------------+
| Lower     | Upper       |
+-----------+-------------+
|           | KELLSEY     |
| christian | KELLS       |
| hallie    | HULL        |
| klee      | HULL        |
| paddy     | O'FURNITURE |
| sarah     | BUCHMAN     |
| wendy     | HEYDEMARK   |
+-----------+-------------+
7 rows in set (0.003 sec)
```

61. Imprimir nombre de libros que contengan MO independientemente de mayúculas y minúsculas

```
SELECT title_name
 FROM titles
 WHERE UPPER(title_name) LIKE '%MO%';
```



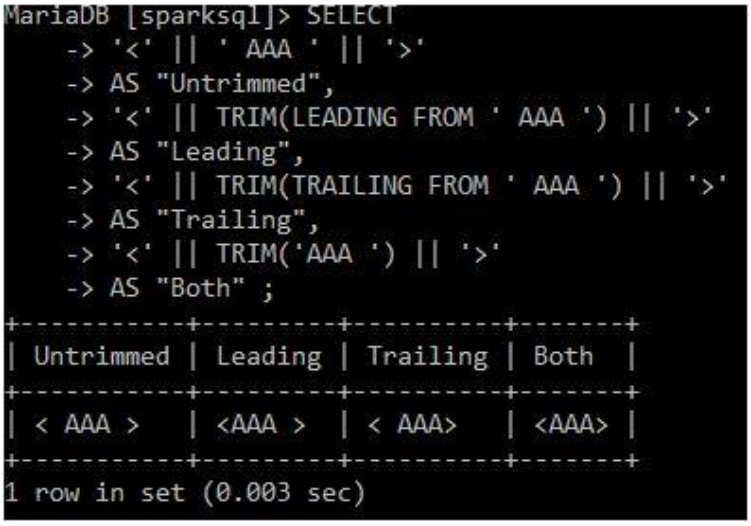
```
MariaDB [sparksql]> SELECT LOWER(au_fname) AS "Lower",
    -> UPPER(au_lname) AS "Upper"
    -> FROM authors;
+-----------+-------------+
| Lower     | Upper       |
+-----------+-------------+
|           | KELLSEY     |
| christian | KELLS       |
| hallie    | HULL        |
| klee      | HULL        |
| paddy     | O'FURNITURE |
| sarah     | BUCHMAN     |
| wendy     | HEYDEMARK   |
+-----------+-------------+
7 rows in set (0.003 sec)
```

Trimming characters with TRIM

62. Remove spaces from ' AAA ' at the beginning, at the end, at the beginning and

at the end.

```
SELECT
 '<' || ' AAA ' || '>'
 AS "Untrimmed",
 '<' || TRIM(LEADING FROM ' AAA ') || '>'
 AS "Leading",
 '<' || TRIM(TRAILING FROM ' AAA ') || '>'
 AS "Trailing",
 '<' || TRIM('AAA ') || '>'
 AS "Both" ;
```



63. List authors' surnames, removing any capital H at the beginning of the name.

at the beginning

```
SELECT au_lname,
 TRIM(LEADING 'H' FROM au_lname)
 AS "Trimmed name"
 FROM authors;
```

```
MariaDB [sparksql]> SELECT au_lname,
    -> TRIM(LEADING 'H' FROM au_lname)
    -> AS "Trimmed name"
    -> FROM authors;
+-------------+--------------+
| au_lname    | Trimmed name |
+-------------+--------------+
| Kellsey     | Kellsey      |
| Kells       | Kells        |
| Hull        | ull          |
| Hull        | ull          |
| O'Furniture | O'Furniture  |
| Buchman     | Buchman      |
| Heydemark   | eydemark     |
+-------------+--------------+
7 rows in set (0.002 sec)
```

64. From the titles table, print the title_id columns starting with T1 and have another character ignoring possible leading and trailing spaces

```
SELECT title_id
 FROM titles
 WHERE TRIM(title_id) LIKE 'T1_';
```

```
MariaDB [sparksql]> SELECT title_id
    -> FROM titles
    -> WHERE TRIM(title_id) LIKE 'T1_';
+----------+
| title_id |
+----------+
| T10      |
| T11      |
| T12      |
| T13      |
+----------+
4 rows in set (0.002 sec)
```

String length with CHARACTER_LENGTH (LENGTH)

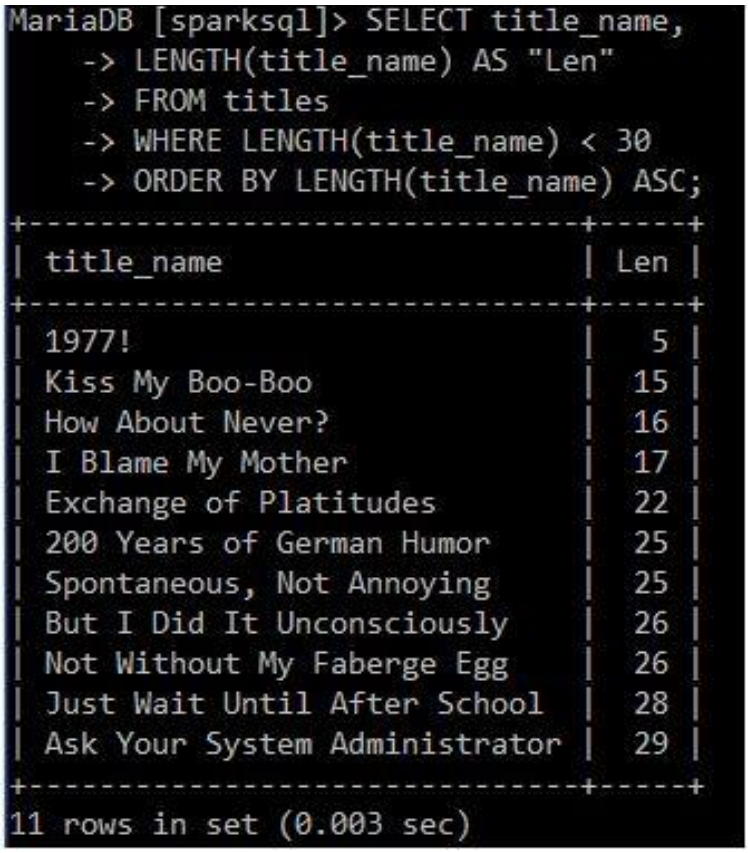
65. Print the name of the authors next to their length

```
SELECT au_fname, LENGTH(au_fname) AS "Len"
 FROM authors;
```

```
MariaDB [sparksql]> SELECT au_fname, LENGTH(au_fname) AS "Len"
    -> FROM authors;
+-----------+-----+
| au_fname  | Len |
+-----------+-----+
|           |   0 |
| Christian |   9 |
| Hallie    |   6 |
| Klee      |   4 |
| Paddy     |   5 |
| Sarah     |   5 |
| Wendy     |   5 |
+-----------+-----+
7 rows in set (0.003 sec)
```

66. List the title of the books together with their length for those books whose length is less than 30.whose length is less than 30. The list has to be sorted by the length of the title.

```
SELECT title_name,
 LENGTH(title_name) AS "Len"
 FROM titles
 WHERE LENGTH(title_name) < 30
 ORDER BY LENGTH(title_name) ASC;
```



```
MariaDB [sparksql]> SELECT title_name,
    -> LENGTH(title_name) AS "Len"
    -> FROM titles
    -> WHERE LENGTH(title_name) < 30
    -> ORDER BY LENGTH(title_name) ASC;
+-------------------------------+-----+
| title_name                    | Len |
+-------------------------------+-----+
| 1977!                         |   5 |
| Kiss My Boo-Boo               |  15 |
| How About Never?              |  16 |
| I Blame My Mother             |  17 |
| Exchange of Platitudes        |  22 |
| 200 Years of German Humor     |  25 |
| Spontaneous, Not Annoying     |  25 |
| But I Did It Unconsciously    |  26 |
| Not Without My Faberge Egg    |  26 |
| Just Wait Until After School  |  28 |
| Ask Your System Administrator |  29 |
+-------------------------------+-----+
11 rows in set (0.003 sec)
```

Finding substrings with POSITION (LOCATE)

67. List the position of the substring e in the author's first name and the position of the substring ma in the author's last name.

```
SELECT
 au_fname,
 LOCATE('e', au_fname) AS "Pos e",
 au_lname,
 LOCATE('ma', au_lname) AS "Pos ma"
 FROM authors;
```

```
MariaDB [sparksql]> SELECT
    -> au_fname,
    -> LOCATE('e', au_fname) AS "Pos e",
    -> au_lname,
    -> LOCATE('ma', au_lname) AS "Pos ma"
    -> FROM authors;
+-----------+-------+-------------+--------+
| au_fname  | Pos e | au_lname    | Pos ma |
+-----------+-------+-------------+--------+
|           |     0 | Kellsey     |      0 |
| Christian |     0 | Kells       |      0 |
| Hallie    |     6 | Hull        |      0 |
| Klee      |     3 | Hull        |      0 |
| Paddy     |     0 | O'Furniture |      0 |
| Sarah     |     0 | Buchman     |      5 |
| Wendy     |     2 | Heydemark   |      6 |
+-----------+-------+-------------+--------+
7 rows in set (0.002 sec)
```

68. List 1. the titles containing the letter u in their first 10 positions and 2) the position of the letter u itself. The resulting table has to be sorted in descending order by the position of the letter u.

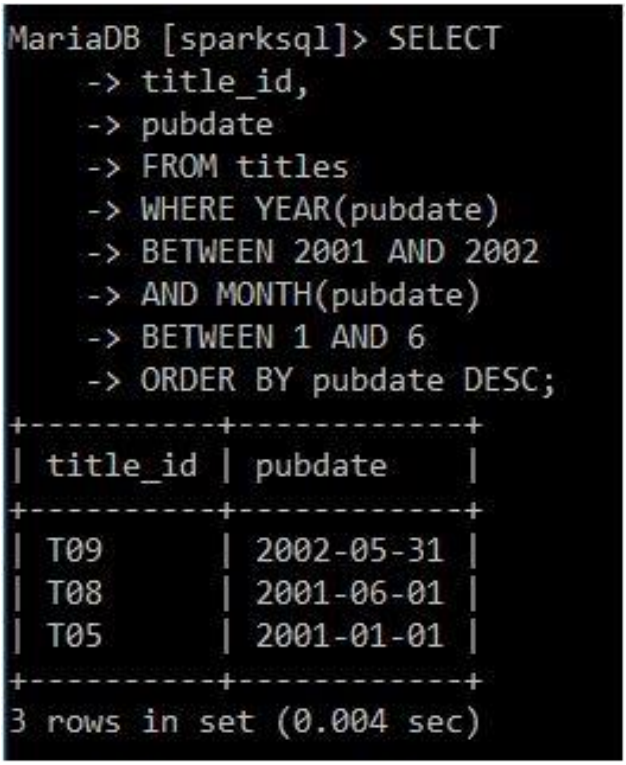
```
SELECT title_name,
 LOCATE('u', title_name) AS "Pos"
 FROM titles
 WHERE LOCATE('u', title_name)
 BETWEEN 1 AND 10
 ORDER BY LOCATE('u', title_name) DESC;
```

```
MariaDB [sparksql]> SELECT title_name,
    -> LOCATE('u', title_name) AS "Pos"
    -> FROM titles
    -> WHERE LOCATE('u', title_name)
    -> BETWEEN 1 AND 10
    -> ORDER BY LOCATE('u', title_name) DESC;
+--------------------------------+-----+
| title_name                     | Pos |
+--------------------------------+-----+
| Not Without My Faberge Egg     |  10 |
| Spontaneous, Not Annoying      |  10 |
| How About Never?               |   8 |
| Ask Your System Administrator  |   7 |
| But I Did It Unconsciously     |   2 |
| Just Wait Until After School   |   2 |
+--------------------------------+-----+
6 rows in set (0.001 sec)
```

Date Arithmetic

69. Print books that have been published in the first 6 months of the years 2001 or 2002. The output has to be sorted in descending order of date.

```
SELECT
 title_id,
 pubdate
 FROM titles
 WHERE YEAR(pubdate)
 BETWEEN 2001 AND 2002
 AND MONTH(pubdate)
 BETWEEN 1 AND 6
 ORDER BY pubdate DESC;
```



```
MariaDB [sparksql]> SELECT
    -> title_id,
    -> pubdate
    -> FROM titles
    -> WHERE YEAR(pubdate)
    -> BETWEEN 2001 AND 2002
    -> AND MONTH(pubdate)
    -> BETWEEN 1 AND 6
    -> ORDER BY pubdate DESC;
+----------+------------+
| title_id | pubdate    |
+----------+------------+
| T09      | 2002-05-31 |
| T08      | 2001-06-01 |
| T05      | 2001-01-01 |
+----------+------------+
3 rows in set (0.004 sec)
```

70. Obtaining current date and time

```
SELECT
 CURRENT_DATE AS "Date",
 CURRENT_TIME AS "Time",
 CURRENT_TIMESTAMP AS "Timestamp";
```

```
MariaDB [sparksql]> SELECT
    -> CURRENT_DATE AS "Date",
    -> CURRENT_TIME AS "Time",
    -> CURRENT_TIMESTAMP AS "Timestamp";
+------------+----------+---------------------+
| Date       | Time     | Timestamp           |
+------------+----------+---------------------+
| 2022-10-31 | 22:38:19 | 2022-10-31 22:38:19 |
+------------+----------+---------------------+
1 row in set (0.003 sec)
```

User information with USERS

71. Print the username logged in to the database

```
SELECT CURRENT_USER AS "User";
```

```
MariaDB [sparksql]> SELECT CURRENT_USER AS "User";
+----------------+
| User           |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.000 sec)
```

Type conversion with CAST

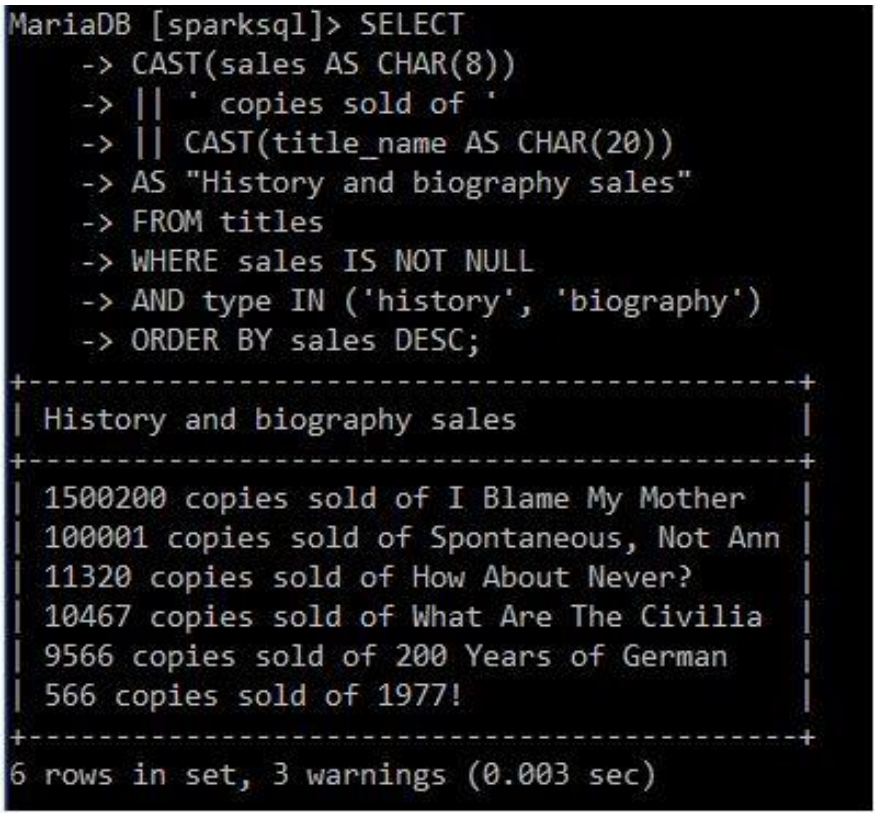
72. Convert price of books to INTEGER and CHAR(8)

```
SELECT
 price AS "price(DECIMAL)",
 CAST(price AS INTEGER) AS "price(INTEGER)",
 '<' || CAST(price AS CHAR(8)) || '>' AS "price(CHAR(8))"
 FROM titles;
```

```
MariaDB [sparksql]> SELECT
    -> price AS "price(DECIMAL)",
    -> CAST(price AS INTEGER) AS "price(INTEGER)",
    -> '<' || CAST(price AS CHAR(8)) || '>' AS "price(CHAR(8))"
    -> FROM titles;
+----------------+----------------+----------------+
| price(DECIMAL) | price(INTEGER) | price(CHAR(8)) |
+----------------+----------------+----------------+
|          21.99 |             22 | <21.99>        |
|          19.95 |             20 | <19.95>        |
|          39.95 |             40 | <39.95>        |
|          12.99 |             13 | <12.99>        |
|           6.95 |              7 | <6.95>         |
|          19.95 |             20 | <19.95>        |
|          23.95 |             24 | <23.95>        |
|          10.00 |             10 | <10.00>        |
|          13.95 |             14 | <13.95>        |
|           NULL |           NULL | NULL           |
|           7.99 |              8 | <7.99>         |
|          12.99 |             13 | <12.99>        |
|          29.99 |             30 | <29.99>        |
+----------------+----------------+----------------+
13 rows in set (0.003 sec)
```

73. In the titles table, convert the column sales to CHAR(8) and the column title_-.name column to CHAR(20) for use in a concatenation with characters.

```
SELECT
 CAST(sales AS CHAR(8))
 || ' copies sold of '
 || CAST(title_name AS CHAR(20))
 AS "History and biography sales"
 FROM titles
 WHERE sales IS NOT NULL
 AND type IN ('history', 'biography')
 ORDER BY sales DESC;
```



```
MariaDB [sparksql]> SELECT
    -> CAST(sales AS CHAR(8))
    -> || ' copies sold of '
    -> || CAST(title_name AS CHAR(20))
    -> AS "History and biography sales"
    -> FROM titles
    -> WHERE sales IS NOT NULL
    -> AND type IN ('history', 'biography')
    -> ORDER BY sales DESC;
+-----------------------------------------+
| History and biography sales             |
+-----------------------------------------+
| 1500200 copies sold of I Blame My Mother |
| 100001 copies sold of Spontaneous, Not Ann |
| 11320 copies sold of How About Never?   |
| 10467 copies sold of What Are The Civilia |
| 9566 copies sold of 200 Years of German |
| 566 copies sold of 1977!                |
+-----------------------------------------+
6 rows in set, 3 warnings (0.003 sec)
```

Evaluation of conditional values with CASE

74. Increase the price of books: history books by 10%, psychology books by 20%, and leave the rest unchanged.

```
SELECT
 title_id,
 type,
 price,
 CASE
 WHEN type = 'history' THEN price * 1.10
 WHEN type = 'psychology' THEN price * 1.20
 ELSE price
 END
 AS "New price"
 FROM titles
 WHERE price is NOT NULL
 ORDER BY type ASC, title_id ASC;
```

```
MariaDB [sparksql]> SELECT
    -> title_id,
    -> type,
    -> price,
    -> CASE
    -> WHEN type = 'history' THEN price * 1.10
    -> WHEN type = 'psychology' THEN price * 1.20
    -> ELSE price
    -> END
    -> AS "New price"
    -> FROM titles
    -> WHERE price is NOT NULL
    -> ORDER BY type ASC, title_id ASC;
+----------+------------+-------+-----------+
| title_id | type       | price | New price |
+----------+------------+-------+-----------+
| T06      | biography  | 19.95 |   19.9500 |
| T07      | biography  | 23.95 |   23.9500 |
| T12      | biography  | 12.99 |   12.9900 |
| T08      | children   | 10.00 |   10.0000 |
| T09      | children   | 13.95 |   13.9500 |
| T03      | computer   | 39.95 |   39.9500 |
| T01      | history    | 21.99 |   24.1890 |
| T02      | history    | 19.95 |   21.9450 |
| T13      | history    | 29.99 |   32.9890 |
| T04      | psychology | 12.99 |   15.5880 |
| T05      | psychology |  6.95 |    8.3400 |
| T11      | psychology |  7.99 |    9.5880 |
+----------+------------+-------+-----------+
12 rows in set (0.003 sec)
```

75. List the books with a sales category of type characters depending on the value of the sales column.

```
SELECT
 title_id,
 CASE
 WHEN sales IS NULL THEN 'Unknown'
 WHEN sales <= 1000 THEN 'Not more than 1,000'
 WHEN sales <= 10000 THEN 'Between 1,001 and 10,000'
 WHEN sales <= 100000 THEN 'Between 10,001 and 100,000'
 WHEN sales <= 1000000 THEN 'Between 100,001 and 1,000,000'
 ELSE 'Over 1,000,000'
 END
 AS "Sales category"
 FROM titles
```

```
ORDER BY sales ASC;
```

```
MariaDB [sparksql]> SELECT
    -> title_id,
    -> CASE
    -> WHEN sales IS NULL THEN 'Unknown'
    -> WHEN sales <= 1000 THEN 'Not more than 1,000'
    -> WHEN sales <= 10000 THEN 'Between 1,001 and 10,000'
    -> WHEN sales <= 100000 THEN 'Between 10,001 and 100,000'
    -> WHEN sales <= 1000000 THEN 'Between 100,001 and 1,000,000'
    -> ELSE 'Over 1,000,000'
    -> END
    -> AS "Sales category"
    -> FROM titles
    -> ORDER BY sales ASC;
+----------+-----------------------------+
| title_id | Sales category              |
+----------+-----------------------------+
| T10      | Unknown                     |
| T01      | Not more than 1,000         |
| T08      | Between 1,001 and 10,000    |
| T09      | Between 1,001 and 10,000    |
| T02      | Between 1,001 and 10,000    |
| T13      | Between 10,001 and 100,000  |
| T06      | Between 10,001 and 100,000  |
| T04      | Between 10,001 and 100,000  |
| T03      | Between 10,001 and 100,000  |
| T11      | Between 10,001 and 100,000  |
| T12      | Between 100,001 and 1,000,000 |
| T05      | Between 100,001 and 1,000,000 |
| T07      | Over 1,000,000              |
+----------+-----------------------------+
13 rows in set (0.005 sec)
```

Null testing with COALESCE

76. Lists the state column of the editors. If the value is null print N/A

```
SELECT
 pub_id,
 city,
 COALESCE(state, 'N/A') AS "state",
country
 FROM publishers;
```

```
MariaDB [sparksql]> SELECT
    -> pub_id,
    -> city,
    -> COALESCE(state, 'N/A') AS "state",
    -> country
    -> FROM publishers;
+--------+---------------+-------+---------+
| pub_id | city          | state | country |
+--------+---------------+-------+---------+
| P01    | New York      | NY    | USA     |
| P02    | San Francisco | CA    | USA     |
| P03    | Hamburg       | N/A   | Germany |
| P04    | Berkeley      | CA    | USA     |
+--------+---------------+-------+---------+
4 rows in set (0.005 sec)
```

Checking nulls with NULLIF

77. In this query if the value of the contract column of the titles table is 0 it is set to NULL

SELECT title_id, contract, NULLIF(contract, 0) AS "Null contract" FROM titles;

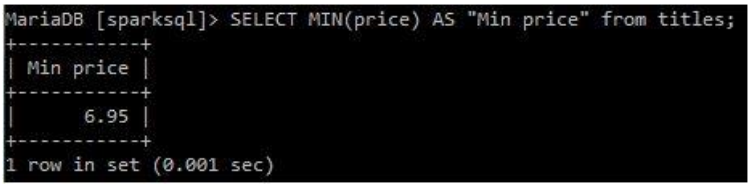
```
MariaDB [sparksql]> SELECT
    -> title_id,
    -> contract,
    -> NULLIF(contract, 0) AS "Null contract"
    -> FROM titles;
+----------+----------+---------------+
| title_id | contract | Null contract |
+----------+----------+---------------+
| T01      |        1 |             1 |
| T02      |        1 |             1 |
| T03      |        1 |             1 |
| T04      |        1 |             1 |
| T05      |        1 |             1 |
| T06      |        1 |             1 |
| T07      |        1 |             1 |
| T08      |        1 |             1 |
| T09      |        1 |             1 |
| T10      |        0 |          NULL |
| T11      |        1 |             1 |
| T12      |        1 |             1 |
| T13      |        1 |             1 |
+----------+----------+---------------+
13 rows in set (0.003 sec)
```

Summary and Aggregation of Data

Find minima with MIN

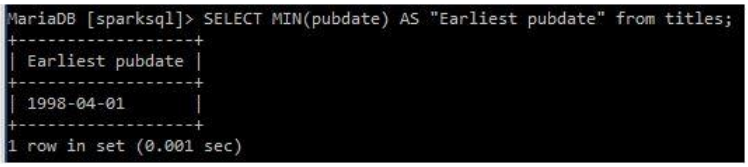
78. Print the lowest price of books

```
SELECT MIN(price) AS "Min price" from titles;
```



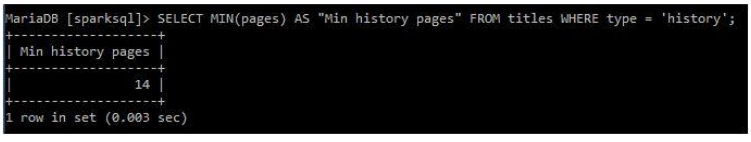
79. Print the earearliest date of publication of a book

```
SELECT MIN(pubdate) AS "Earliest pubdate" from titles;
```



80. From history books print the number of pages of the one with the lowest number

```
SELECT MIN(pages) AS "Min history pages" FROM titles WHERE type =
'history';
```



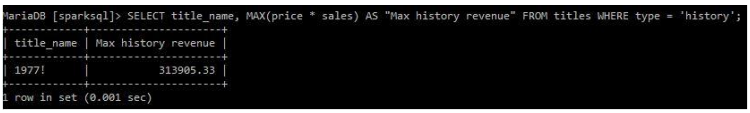
Find maximums with MAX

81. Print the lowest book price, the highest, and the difference range

```
SELECT MIN(price) AS "Min price", MAX(price) AS "Max price",MAX(price) -
MIN(price) AS "Range" FROM titles;
```

82. History books print the maximum price value multiplied by salts

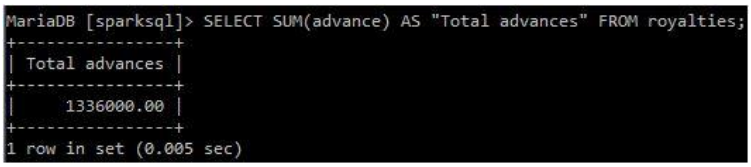
SELECT title_name, MAX(price * sales) AS "Max history revenue" FROM titles WHERE type = 'history';



Calculation of sums with SUM

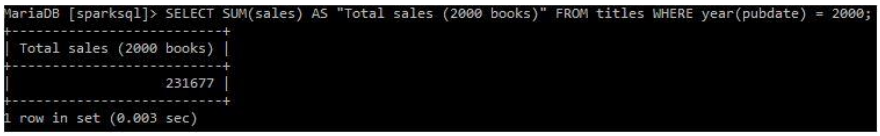
83. Print the sum of all advances paid to authors.

SELECT SUM(advance) AS "Total advances" FROM royalties;



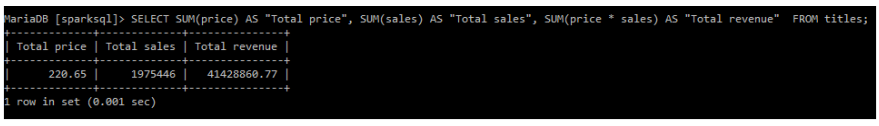
84. Print sales of all books published in 2000.

SELECT SUM(sales) AS "Total sales (2000 books)" FROM titles WHERE year(pubdate) = 2000;



85. Print the sum of 1) prices, 2) sales, and 3) multiplying prices by sales for all books.

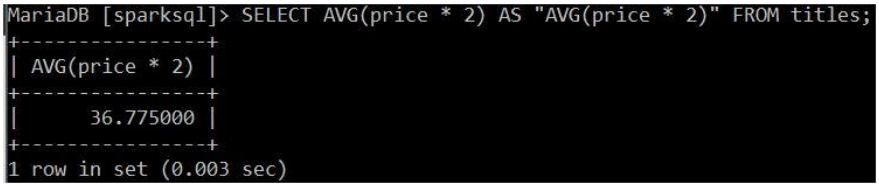
SELECT SUM(price) AS "Total price", SUM(sales) AS "Total sales", SUM(price * sales) AS "Total revenue" FROM titles;



Cálculo de medias con AVG

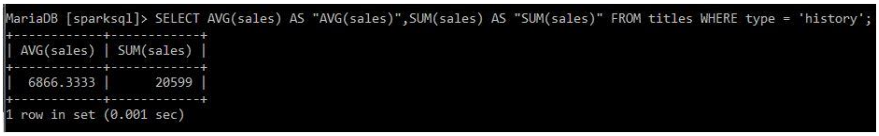
86. Print the average of the book prices multiplied by 2.

```
SELECT AVG(price * 2) AS "AVG(price * 2)" FROM titles;
```



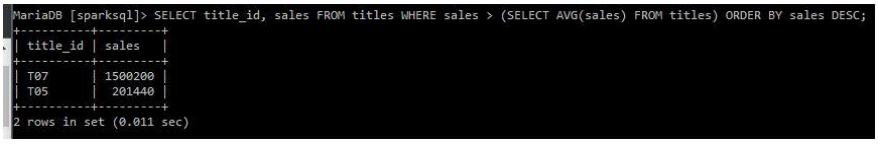
87. Print average and sum of history book sales.

```
SELECT AVG(sales) AS "AVG(sales)",SUM(sales) AS "SUM(sales)" FROM titles
WHERE type = 'history';
```



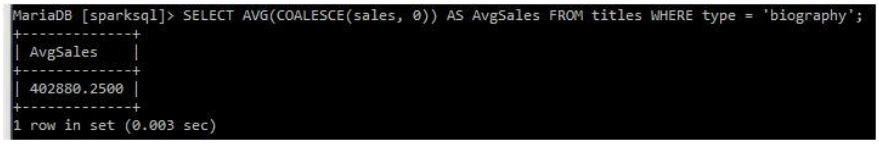
88. Print the books and their sales for all books whose sales are above average sales exceed the average.

```
SELECT title_id, sales FROM titles WHERE sales > (SELECT AVG(sales) FROM
titles) ORDER BY sales DESC;
```



89. Print the average sales of the biographies considering the nulls as 0.

```
SELECT AVG(COALESCE(sales, 0)) AS AvgSales FROM titles WHERE type =
'biography';
```

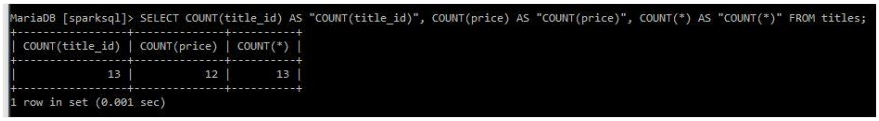


Calculation of number of rows with COUNT

90. Print the count of book titles, their price, and quantity.

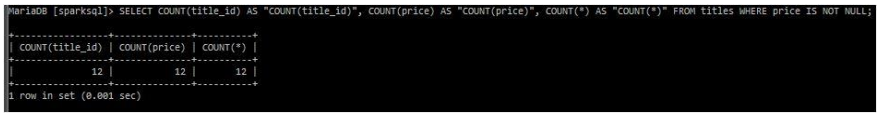
```
SELECT COUNT(title_id) AS "COUNT(title_id)", COUNT(price) AS
```

"COUNT(price)", COUNT(*) AS "COUNT(*)" FROM titles;



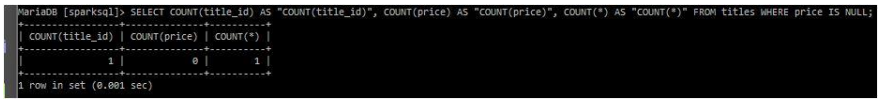
91. Print the count of the book titles, their price, and the quantity excluding books whose price is null.

SELECT COUNT(title_id) AS "COUNT(title_id)", COUNT(price) AS "COUNT(price)", COUNT(*) AS "COUNT(*)" FROM titles WHERE price IS NOT NULL;



92. Print the count of the book titles, their price, and the quantity for those books whose price is null.

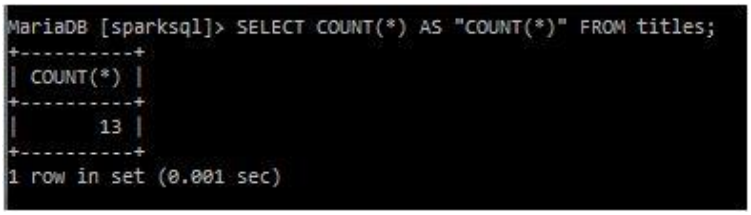
SELECT COUNT(title_id) AS "COUNT(title_id)", COUNT(price) AS "COUNT(price)", COUNT(*) AS "COUNT(*)" FROM titles WHERE price IS NULL;



Sum of distinct values with DISTINCT

93. Print the total number of books in the table titles.

SELECT COUNT(*) AS "COUNT(*)" FROM titles;



94. Consider the books that do not have NULL as a price and print 1) the number of them, 2) the sum of their prices, and 3) the average of their prices.

SELECT COUNT(price) AS "COUNT(price)",SUM(price) AS "SUM(price)", AVG(price) AS "AVG(price)" FROM titles;

```
MariaDB [sparksql]> SELECT COUNT(price) AS "COUNT(price)",SUM(price) AS "SUM(price)", AVG(price) AS "AVG(price)" FROM titles;
+--------------+------------+------------+
| COUNT(price) | SUM(price) | AVG(price) |
+--------------+------------+------------+
|           12 |     220.65 |  18.387500 |
+--------------+------------+------------+
1 row in set (0.001 sec)
```

95. Consider the books that do not have NULL as a price and print 1) the number of them, 2) the sum of their prices, and 3) the average of their prices.

SELECT COUNT(DISTINCT price) AS "COUNT(DISTINCT)", SUM(DISTINCT price) AS "SUM(DISTINCT)", AVG(DISTINCT price) AS "AVG(DISTINCT)" FROM titles;

```
MariaDB [sparksql]> SELECT COUNT(DISTINCT price) AS "COUNT(DISTINCT)", SUM(DISTINCT price) AS "SUM(DISTINCT)", AVG(DISTINCT price) AS "AVG(DISTINCT)" FROM
titles;
+-----------------+--------------+--------------+
| COUNT(DISTINCT) | SUM(DISTINCT) | AVG(DISTINCT) |
+-----------------+--------------+--------------+
|              10 |       187.71 |     18.771000 |
+-----------------+--------------+--------------+
1 row in set (0.007 sec)
```

96. For each author, print the number of books he/she wrote, including those in which he/she is a co-author.

SELECT au_id,COUNT(*) AS "num_books" FROM title_authors GROUP BY au_id;

```
MariaDB [sparksql]> SELECT au_id,COUNT(*) AS "num_books" FROM title_authors GROUP BY au_id;
+-------+-----------+
| au_id | num_books |
+-------+-----------+
| A01   |         3 |
| A02   |         4 |
| A03   |         2 |
| A04   |         4 |
| A05   |         1 |
| A06   |         3 |
+-------+-----------+
6 rows in set (0.007 sec)
```

97. Illustrates the difference between COUNT(state) and COUNT(*).

SELECT state, COUNT(state) AS "COUNT(state)", COUNT(*) AS "COUNT(*)" FROM publishers GROUP BY state;

```
MariaDB [sparksql]> SELECT state, COUNT(state) AS "COUNT(state)", COUNT(*) AS "COUNT(*)" FROM publishers GROUP BY state;
+-------+--------------+----------+
| state | COUNT(state) | COUNT(*) |
+-------+--------------+----------+
| NULL  |            0 |        1 |
| CA    |            2 |        2 |
| NY    |            1 |        1 |
+-------+--------------+----------+
3 rows in set (0.010 sec)
```

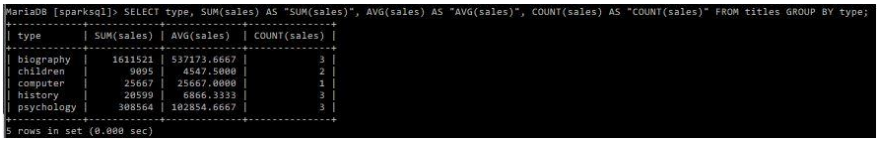
98. In order to have consistent mathematical results, it is necessary to use COUNT(sales) instead of COUNT(*).

SELECT type, SUM(sales)/COUNT(sales) AS "SUM/COUNT(sales)", SUM(sales)/COUNT(*) AS "SUM/COUNT(*)", AVG(sales) AS "AVG(sales)" FROM titles GROUP BY type;

```
MariaDB [sparksql]> SELECT type, SUM(sales)/COUNT(sales) AS "SUM/COUNT(sales)", SUM(sales)/COUNT(*) AS "SUM/COUNT(*)", AVG(sales) AS "AVG(sales)"
    -> FROM titles GROUP BY type;
+-----------+---------------------+---------------+---------------+
| type      | SUM/COUNT(sales)    | SUM/COUNT(*)  | AVG(sales)    |
+-----------+---------------------+---------------+---------------+
| biography |       537173.6667 |   402880.2500 |  537173.6667 |
| children  |         4547.5000 |     4547.5000 |    4547.5000 |
| computer  |        25667.0000 |    25667.0000 |   25667.0000 |
| history   |         6866.3333 |     6866.3333 |    6866.3333 |
| psychology|       102854.6667 |   102854.6667 |  102854.6667 |
+-----------+---------------------+---------------+---------------+
5 rows in set (0.003 sec)
```

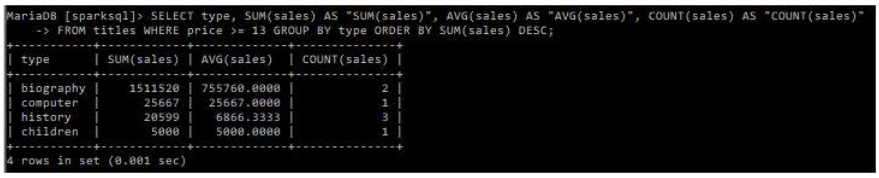
99. For each type of book, calculate statistics on total sales, average, and number of books.

```
SELECT type, SUM(sales) AS "SUM(sales)", AVG(sales) AS "AVG(sales)",
COUNT(sales) AS "COUNT(sales)" FROM titles GROUP BY type;
```



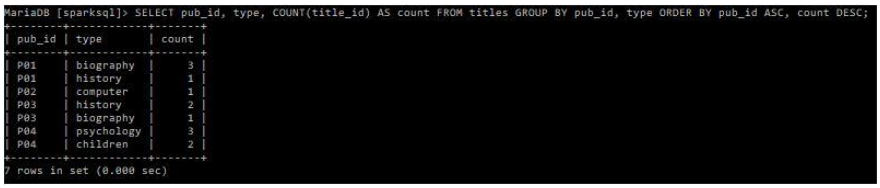
100. For each type of book, calculate the following statistics: (1) total sales, 2) average sales, and 3) number of books. The result has to be be sorted by total sales. In addition, only books with a price equal to or higher than only those books whose price is equal to or greater than 13. (use WHERE and ORDER BY).

```
SELECT type, SUM(sales) AS "SUM(sales)", AVG(sales) AS "AVG(sales)",
COUNT(sales) AS "COUNT(sales)"
FROM titles WHERE price >= 13 GROUP BY type ORDER BY SUM(sales) DESC;
```



101. For each publisher and type of book it lists the number of books sorted in ascending order by publisher and in descending order by number of books.

```
SELECT pub_id, type, COUNT(title_id) AS count FROM titles GROUP BY pub_id,
type ORDER BY pub_id ASC, count DESC;
```



102. Print the different types of books.

```
SELECT DISTINCT type FROM titles;
```

```
MariaDB [sparksql]> SELECT DISTINCT type FROM titles;
+------------+
| type       |
+------------+
| history    |
| computer   |
| psychology |
| biography  |
| children   |
+------------+
5 rows in set (0.003 sec)
```

103. Lists the average sales for each of the prices sorted by ascending price ascending.

SELECT price, AVG(sales) FROM titles WHERE price IS NOT NULL GROUP BY price ORDER BY price ASC;

```
MariaDB [sparksql]> SELECT price, AVG(sales) FROM titles WHERE price IS NOT NULL GROUP BY price ORDER BY price ASC;
+-------+-------------+
| price | AVG(sales)  |
+-------+-------------+
|  6.95 |  201440.0000 |
|  7.99 |   94123.0000 |
| 10.00 |    4095.0000 |
| 12.99 |   56501.0000 |
| 13.95 |    5000.0000 |
| 19.95 |   10443.0000 |
| 21.99 |     566.0000 |
| 23.95 | 1500200.0000 |
| 29.99 |   10467.0000 |
| 39.95 |   25667.0000 |
+-------+-------------+
10 rows in set (0.001 sec)
```

Group filtering with HAVING

104. List the number of books written per author as long as he/she has written more than 3 books.

SELECT au_id,COUNT(*) AS "num_books" FROM title_authors GROUP BY au_id HAVING COUNT(*) >= 3;

```
MariaDB [sparksql]> SELECT au_id,COUNT(*) AS "num_books" FROM title_authors GROUP BY au_id HAVING COUNT(*) >= 3;
+-------+-----------+
| au_id | num_books |
+-------+-----------+
| A01   |         3 |
| A02   |         4 |
| A04   |         4 |
| A06   |         3 |
+-------+-----------+
4 rows in set (0.103 sec)
```

105. For each type of book, it prints the number of books and the average profit (price*sales), but only for those types with an average sale of more than 1,000,000.

SELECT type, COUNT(price) AS "COUNT(price)", AVG(price * sales) AS "AVG revenue"
FROM titles GROUP BY type HAVING AVG(price * sales) > 1000000;

```
MariaDB [sparksql]> SELECT type, COUNT(price) AS "COUNT(price)", AVG(price * sales) AS "AVG revenue"
    -> FROM titles GROUP BY type HAVING AVG(price * sales) > 1000000;
+-----------+--------------+-----------------+
| type      | COUNT(price) | AVG revenue     |
+-----------+--------------+-----------------+
| biography |            3 | 12484878.996667 |
| computer  |            1 |  1025396.650000 |
+-----------+--------------+-----------------+
2 rows in set (0.033 sec)
```

106. For each publisher, list the number of books of each type, for publishers with more than one book per type.

SELECT pub_id, type, COUNT(*) AS count FROM titles
GROUP BY pub_id, type HAVING COUNT(*) > 1 ORDER BY pub_id ASC, count DESC;

```
MariaDB [sparksql]> SELECT pub_id, type, COUNT(*) AS count FROM titles
    -> GROUP BY pub_id, type HAVING COUNT(*) > 1 ORDER BY pub_id ASC, count DESC;
+--------+------------+-------+
| pub_id | type       | count |
+--------+------------+-------+
| P01    | biography  |     3 |
| P03    | history    |     2 |
| P04    | psychology |     3 |
| P04    | children   |     2 |
+--------+------------+-------+
4 rows in set (0.001 sec)
```

107. For books from publishers P03 and P04, list total sales and average price by type, for types with more than 10,000 total sales and less than 20 average.

SELECT type, SUM(sales) AS "SUM(sales)", AVG(price) AS "AVG(price)" FROM titles
WHERE pub_id IN ('P03', 'P04') GROUP BY type HAVING SUM(sales) > 10000 AND AVG(price) < 20;

```
MariaDB [sparksql]> SELECT type, SUM(sales) AS "SUM(sales)", AVG(price) AS "AVG(price)" FROM titles
    -> WHERE pub_id IN ('P03', 'P04') GROUP BY type HAVING SUM(sales) > 10000 AND AVG(price) < 20;
+------------+------------+------------+
| type       | SUM(sales) | AVG(price) |
+------------+------------+------------+
| psychology |     308564 |   9.310000 |
+------------+------------+------------+
1 row in set (0.005 sec)
```

Joins

Qualification of column names

108. Selecting authors who live in a city where a publisher also lives publisher.

SELECT au_id, authors.city FROM authors INNER JOIN publishers ON authors.city = publishers.city;

```
MariaDB [sparksql]> SELECT au_id, authors.city FROM authors INNER JOIN publishers ON authors.city = publishers.city;
+-------+---------------+
| au_id | city          |
+-------+---------------+
| A03   | San Francisco |
| A04   | San Francisco |
| A05   | New York      |
+-------+---------------+
3 rows in set (0.012 sec)
```

Creation of table aliases using AS

109. Select first and last names of authors who live in a city where at least one publisher also lives.

SELECT au_fname, au_lname, a.city FROM authors AS a INNER JOIN publishers p ON a.city = p.city;

```
MariaDB [sparksql]> SELECT au_fname, au_lname, a.city FROM authors AS a INNER JOIN publishers p ON a.city = p.city;
+-----------+----------+---------------+
| au_fname  | au_lname | city          |
+-----------+----------+---------------+
| Hallie    | Hull     | San Francisco |
| Klee      | Hull     | San Francisco |
| Christian | Kells    | New York      |
+-----------+----------+---------------+
3 rows in set (0.001 sec)
```

joins with JOIN or WHERE

110. Select first name (au_fname), last name (au_lname), and city (au_lname) for authors who live in a city in which at least one publisher also lives.

SELECT au_fname, au_lname, a.city FROM authors a INNER JOIN publishers p ON a.city = p.city;

```
MariaDB [sparksql]> SELECT au_fname, au_lname, a.city FROM authors a INNER JOIN publishers p ON a.city = p.city;
+-----------+----------+---------------+
| au_fname  | au_lname | city          |
+-----------+----------+---------------+
| Hallie    | Hull     | San Francisco |
| Klee      | Hull     | San Francisco |
| Christian | Kells    | New York      |
+-----------+----------+---------------+
3 rows in set (0.001 sec)
```

CROSS JOIN

111. Displays the selected columns of all possible combinations of rows of the authors and publishers tables.

SELECT au_id, pub_id, a.state AS "au_state", p.state AS "pub_state"
FROM authors a CROSS JOIN publishers p ORDER BY au_id, pub_id;

```
MariaDB [sparksql]> SELECT au_id, pub_id, a.state AS "au_state", p.state AS "pub_state"
    -> FROM authors a CROSS JOIN publishers p ORDER BY au_id, pub_id;
+-------+--------+----------+-----------+
| au_id | pub_id | au_state | pub_state |
+-------+--------+----------+-----------+
| A01   | P01    | NY       | NY        |
| A01   | P02    | NY       | CA        |
| A01   | P03    | NY       | NULL      |
| A01   | P04    | NY       | CA        |
| A02   | P01    | CO       | NY        |
| A02   | P02    | CO       | CA        |
| A02   | P03    | CO       | NULL      |
| A02   | P04    | CO       | CA        |
| A03   | P01    | CA       | NY        |
| A03   | P02    | CA       | CA        |
| A03   | P03    | CA       | NULL      |
| A03   | P04    | CA       | CA        |
| A04   | P01    | CA       | NY        |
| A04   | P02    | CA       | CA        |
| A04   | P03    | CA       | NULL      |
| A04   | P04    | CA       | CA        |
| A05   | P01    | NY       | NY        |
| A05   | P02    | NY       | CA        |
| A05   | P03    | NY       | NULL      |
| A05   | P04    | NY       | CA        |
| A06   | P01    | CA       | NY        |
| A06   | P02    | CA       | CA        |
| A06   | P03    | CA       | NULL      |
| A06   | P04    | CA       | CA        |
| A07   | P01    | FL       | NY        |
| A07   | P02    | FL       | CA        |
| A07   | P03    | FL       | NULL      |
| A07   | P04    | FL       | CA        |
+-------+--------+----------+-----------+
28 rows in set (0.001 sec)
```

NATURAL JOIN

112. For each book (titles table) prints the name (pub_name) of its publisher.

SELECT title_id, pub_id, pub_name FROM publishers NATURAL JOIN titles;

```
MariaDB [sparksql]> SELECT title_id, pub_id, pub_name FROM publishers NATURAL JOIN titles;
+----------+--------+--------------------+
| title_id | pub_id | pub_name           |
+----------+--------+--------------------+
| T01      | P01    | Abatis Publishers  |
| T02      | P03    | Schadenfreude Press |
| T03      | P02    | Core Dump Books    |
| T04      | P04    | Tenterhooks Press  |
| T05      | P04    | Tenterhooks Press  |
| T06      | P01    | Abatis Publishers  |
| T07      | P03    | Schadenfreude Press |
| T08      | P04    | Tenterhooks Press  |
| T09      | P04    | Tenterhooks Press  |
| T10      | P01    | Abatis Publishers  |
| T11      | P04    | Tenterhooks Press  |
| T12      | P01    | Abatis Publishers  |
| T13      | P03    | Schadenfreude Press |
+----------+--------+--------------------+
13 rows in set (0.003 sec)
```

113. For each book, list the name of the publisher (pub_name) and the advance (advance) as long as the advance is less than 20,000. Tables publishers, titles, and royalties.

SELECT title_id, pub_id, pub_name, advance FROM publishers
NATURAL JOIN titles NATURAL JOIN royalties WHERE advance < 20000;

```
MariaDB [sparksql]> SELECT title_id, pub_id, pub_name, advance FROM publishers
    -> NATURAL JOIN titles NATURAL JOIN royalties WHERE advance < 20000;
+----------+--------+--------------------+----------+
| title_id | pub_id | pub_name           | advance  |
+----------+--------+--------------------+----------+
| T01      | P01    | Abatis Publishers  | 10000.00 |
| T03      | P02    | Core Dump Books    | 15000.00 |
| T02      | P03    | Schadenfreude Press |  1000.00 |
| T08      | P04    | Tenterhooks Press  |     0.00 |
| T09      | P04    | Tenterhooks Press  |     0.00 |
+----------+--------+--------------------+----------+
5 rows in set (0.006 sec)
```

INNER JOIN

114. Join two tables (authors and title_authors) using the au_id column to list the title_id of the books each author wrote.

SELECT a.au_id, a.au_fname, a.au_lname, ta.title_id FROM authors a INNER JOIN title_authors ta ON a.au_id = ta.au_id ORDER BY a.au_id ASC, ta.title_id ASC;

```
MariaDB [sparksql]> SELECT a.au_id, a.au_fname, a.au_lname, ta.title_id
    -> FROM authors a INNER JOIN title_authors ta ON a.au_id = ta.au_id ORDER BY a.au_id ASC, ta.title_id ASC;
+-------+-----------+-----------+----------+
| au_id | au_fname  | au_lname  | title_id |
+-------+-----------+-----------+----------+
| A01   | Sarah     | Buchman   | T01      |
| A01   | Sarah     | Buchman   | T02      |
| A01   | Sarah     | Buchman   | T13      |
| A02   | Wendy     | Heydemark | T06      |
| A02   | Wendy     | Heydemark | T07      |
| A02   | Wendy     | Heydemark | T10      |
| A02   | Wendy     | Heydemark | T12      |
| A03   | Hallie    | Hull      | T04      |
| A03   | Hallie    | Hull      | T11      |
| A04   | Klee      | Hull      | T04      |
| A04   | Klee      | Hull      | T05      |
| A04   | Klee      | Hull      | T07      |
| A04   | Klee      | Hull      | T11      |
| A05   | Christian | Kells     | T03      |
| A06   |           | Kellsey   | T08      |
| A06   |           | Kellsey   | T09      |
| A06   |           | Kellsey   | T11      |
+-------+-----------+-----------+----------+
17 rows in set (0.003 sec)
```

115. Join three tables (authors, title_authors, and titles) using the column au_id and title_id to list the title_name of the books each author wrote.

SELECT a.au_id, a.au_fname, a.au_lname, t.title_name FROM authors a INNER JOIN title_authors ta ON a.au_id = ta.au_id INNER JOIN titles t ORDER BY a.au_id ASC, ta.title_id ASC;

```
MariaDB [sparksql]> SELECT a.au_id, a.au_fname, a.au_lname, t.title_name FROM authors a
    -> INNER JOIN title_authors ta ON a.au_id = ta.au_id
    -> INNER JOIN titles t
    -> ORDER BY a.au_id ASC, ta.title_id ASC;
+-------+----------+----------+-----------------------------------+
| au_id | au_fname | au_lname | title_name                        |
+-------+----------+----------+-----------------------------------+
| A01   | Sarah    | Buchman  | Kiss My Boo-Boo                   |
| A01   | Sarah    | Buchman  | 200 Years of German Humor         |
| A01   | Sarah    | Buchman  | Just Wait Until After School      |
| A01   | Sarah    | Buchman  | 1977!                             |
| A01   | Sarah    | Buchman  | I Blame My Mother                 |
| A01   | Sarah    | Buchman  | What Are The Civilian Applications? |
| A01   | Sarah    | Buchman  | How About Never?                  |
| A01   | Sarah    | Buchman  | Spontaneous, Not Annoying         |
| A01   | Sarah    | Buchman  | Exchange of Platitudes            |
| A01   | Sarah    | Buchman  | Perhaps It's a Glandular Problem  |
| A01   | Sarah    | Buchman  | But I Did It Unconsciously        |
| A01   | Sarah    | Buchman  | Not Without My Faberge Egg        |
| A01   | Sarah    | Buchman  | Ask Your System Administrator     |
| A01   | Sarah    | Buchman  | Exchange of Platitudes            |
| A01   | Sarah    | Buchman  | Perhaps It's a Glandular Problem  |
| A01   | Sarah    | Buchman  | But I Did It Unconsciously        |
| A01   | Sarah    | Buchman  | Not Without My Faberge Egg        |
| A01   | Sarah    | Buchman  | Ask Your System Administrator     |
| A01   | Sarah    | Buchman  | Kiss My Boo-Boo                   |
| A01   | Sarah    | Buchman  | 200 Years of German Humor         |
| A01   | Sarah    | Buchman  | Just Wait Until After School      |
| A01   | Sarah    | Buchman  | 1977!                             |
| A01   | Sarah    | Buchman  | I Blame My Mother                 |
| A01   | Sarah    | Buchman  | What Are The Civilian Applications? |
| A01   | Sarah    | Buchman  | How About Never?                  |
| A01   | Sarah    | Buchman  | Spontaneous, Not Annoying         |
| A01   | Sarah    | Buchman  | I Blame My Mother                 |
| A01   | Sarah    | Buchman  | What Are The Civilian Applications? |
```

116. Join two tables (titles and publishers) by means of the pub_id column to list 1) the title_id of the book, 2) the title_name of the book, 3) the pub_id of the publisher, and 4) the publisher's pub_name.

```
SELECT t.title_id, t.title_name, t.pub_id, p.pub_name FROM titles t
INNER JOIN publishers p ON p.pub_id = t.pub_id ORDER BY t.title_name ASC;
```

```
MariaDB [sparksql]> SELECT t.title_id, t.title_name, t.pub_id, p.pub_name FROM titles t
    -> INNER JOIN publishers p ON p.pub_id = t.pub_id ORDER BY t.title_name ASC;
+----------+--------------------------------------+--------+---------------------+
| title_id | title_name                           | pub_id | pub_name            |
+----------+--------------------------------------+--------+---------------------+
| T01      | 1977!                                | P01    | Abatis Publishers   |
| T02      | 200 Years of German Humor            | P03    | Schadenfreude Press |
| T03      | Ask Your System Administrator        | P02    | Core Dump Books     |
| T04      | But I Did It Unconsciously           | P04    | Tenterhooks Press   |
| T05      | Exchange of Platitudes               | P04    | Tenterhooks Press   |
| T06      | How About Never?                     | P01    | Abatis Publishers   |
| T07      | I Blame My Mother                    | P03    | Schadenfreude Press |
| T08      | Just Wait Until After School         | P04    | Tenterhooks Press   |
| T09      | Kiss My Boo-Boo                      | P04    | Tenterhooks Press   |
| T10      | Not Without My Faberge Egg           | P01    | Abatis Publishers   |
| T11      | Perhaps It's a Glandular Problem     | P04    | Tenterhooks Press   |
| T12      | Spontaneous, Not Annoying            | P01    | Abatis Publishers   |
| T13      | What Are The Civilian Applications?  | P03    | Schadenfreude Press |
+----------+--------------------------------------+--------+---------------------+
13 rows in set (0.001 sec)
```

117. Lists authors who live in the same city (city) and state (state) as a publisher.

```
SELECT a.au_id, a.au_fname, a.au_lname, a.city, a.state FROM authors a
INNER JOIN publishers p ON a.city = p.city AND a.state = p.state ORDER BY
a.au_id ASC;
```

```
MariaDB [sparksql]> SELECT a.au_id, a.au_fname, a.au_lname, a.city, a.state FROM authors a INNER JOIN publishers p
    -> ON a.city = p.city AND a.state = p.state ORDER BY a.au_id ASC;
+-------+----------+----------+---------------+-------+
| au_id | au_fname | au_lname | city          | state |
+-------+----------+----------+---------------+-------+
| A03   | Hallie   | Hull     | San Francisco | CA    |
| A04   | Klee     | Hull     | San Francisco | CA    |
| A05   | Christian | Kells    | New York      | NY    |
+-------+----------+----------+---------------+-------+
3 rows in set (0.001 sec)
```

118. Combines an inner join with WHERE conditions to list books published in the state of California (CA) or outside the North American countries (USA, Canada, Mexico).

```
SELECT t.title_id, t.title_name, p.state, p.country FROM titles t INNER JOIN
publishers p ON t.pub_id = p.pub_id WHERE p.state = 'CA' OR p.country NOT
IN ('USA', 'Canada', 'Mexico') ORDER BY t.title_id ASC;
```

```
MariaDB [sparksql]> SELECT t.title_id, t.title_name, p.state, p.country FROM titles t INNER JOIN publishers p
    -> ON t.pub_id = p.pub_id WHERE p.state = 'CA' OR p.country NOT IN ('USA', 'Canada', 'Mexico')
    -> ORDER BY t.title_id ASC;
+----------+--------------------------------------+-------+---------+
| title_id | title_name                           | state | country |
+----------+--------------------------------------+-------+---------+
| T02      | 200 Years of German Humor            | NULL  | Germany |
| T03      | Ask Your System Administrator        | CA    | USA     |
| T04      | But I Did It Unconsciously           | CA    | USA     |
| T05      | Exchange of Platitudes               | CA    | USA     |
| T07      | I Blame My Mother                    | NULL  | Germany |
| T08      | Just Wait Until After School         | CA    | USA     |
| T09      | Kiss My Boo-Boo                      | CA    | USA     |
| T11      | Perhaps It's a Glandular Problem     | CA    | USA     |
| T13      | What Are The Civilian Applications?  | NULL  | Germany |
+----------+--------------------------------------+-------+---------+
9 rows in set (0.003 sec)
```

119. Combines an inner join with the COUNT aggregate function and a GROUP BY clause to list the number of books an author wrote (or co-wrote).

```
SELECT a.au_id, COUNT(ta.title_id) AS "Num books" FROM authors a INNER
```

JOIN title_authors ta ON a.au_id = ta.au_id GROUP BY a.au_id ORDER BY a.au_id ASC;

```
MariaDB [sparksql]> SELECT a.au_id, COUNT(ta.title_id) AS "Num books" FROM authors a INNER JOIN title_authors ta ON a
.au_id = ta.au_id GROUP BY a.au_id ORDER BY a.au_id ASC;
+-------+-----------+
| au_id | Num books |
+-------+-----------+
| A01   |         3 |
| A02   |         4 |
| A03   |         2 |
| A04   |         4 |
| A05   |         1 |
| A06   |         3 |
+-------+-----------+
6 rows in set (0.001 sec)
```

120. Use WHERE conditions to list the advance paid for each biography. Royalty and title tables.

SELECT t.title_id, t.title_name, r.advance FROM royalties r INNER JOIN titles t ON r.title_id = t.title_id WHERE t.type = 'biography' AND r.advance IS NOT NULL ORDER BY r.advance DESC;

```
MariaDB [sparksql]> SELECT t.title_id, t.title_name, r.advance FROM royalties r INNER JOIN titles t ON r.title_id = t
.title_id WHERE t.type = 'biography' AND r.advance IS NOT NULL ORDER BY r.advance DESC;
+----------+--------------------------+------------+
| title_id | title_name               | advance    |
+----------+--------------------------+------------+
| T07      | I Blame My Mother        | 1000000.00 |
| T12      | Spontaneous, Not Annoying |   50000.00 |
| T06      | How About Never?         |   20000.00 |
+----------+--------------------------+------------+
3 rows in set (0.001 sec)
```

121. Use aggregate functions and the GROUP BY clause to list the amount and the advance paid for each type of book the advance paid for each type of book. Royalty and title tables.

SELECT t.type, COUNT(r.advance) AS "COUNT", SUM(r.advance) AS "advance" FROM royalties r INNER JOIN titles t ON r.title_id = t.title_id WHERE r.advance IS NOT NULL GROUP BY t.type ORDER BY t.type ASC;

```
MariaDB [sparksql]> SELECT t.type, COUNT(r.advance) AS "COUNT", SUM(r.advance) AS "advance"
    -> FROM royalties r INNER JOIN titles t ON r.title_id = t.title_id WHERE r.advance IS NOT NULL
    -> GROUP BY t.type ORDER BY t.type ASC;
+------------+-------+------------+
| type       | COUNT | advance    |
+------------+-------+------------+
| biography  |     3 | 1070000.00 |
| children   |     2 |       0.00 |
| computer   |     1 |   15000.00 |
| history    |     3 |   31000.00 |
| psychology |     3 |  220000.00 |
+------------+-------+------------+
5 rows in set (0.001 sec)
```

122. Use aggregate functions and the GROUP BY clause to list the amount and advance paid for each book type and publisher.Tables royalties and titles.

SELECT t.type, t.pub_id, COUNT(r.advance) AS "COUNT", SUM(r.advance) AS "advance" FROM royalties r INNER JOIN titles t ON r.title_id = t.title_id WHERE r.advance IS NOT NULL GROUP BY t.type, t.pub_id ORDER BY t.type ASC, t.pub_id ASC;

```
MariaDB [sparksql]> SELECT t.type, t.pub_id, COUNT(r.advance) AS "COUNT", SUM(r.advance) AS "advance"
    -> FROM royalties r INNER JOIN titles t ON r.title_id = t.title_id WHERE r.advance IS NOT NULL
    -> GROUP BY t.type, t.pub_id ORDER BY t.type ASC, t.pub_id ASC;
+------------+--------+-------+------------+
| type       | pub_id | COUNT | advance    |
+------------+--------+-------+------------+
| biography  | P01    |     2 |   70000.00 |
| biography  | P03    |     1 | 1000000.00 |
| children   | P04    |     2 |       0.00 |
| computer   | P02    |     1 |   15000.00 |
| history    | P01    |     1 |   10000.00 |
| history    | P03    |     2 |   21000.00 |
| psychology | P04    |     3 |  220000.00 |
+------------+--------+-------+------------+
7 rows in set (0.001 sec)
```

123. List title_id and number of co-authors of all books written by 2 or more authors by 2 or more authors.

SELECT ta.title_id, COUNT(ta.au_id) AS "authors" FROM authors a INNER JOIN title_authors ta ON a.au_id = ta.au_id GROUP BY ta.title_id HAVING COUNT(ta.au_id) > 1 ORDER BY ta.title_id ASC;

```
MariaDB [sparksql]> SELECT ta.title_id, COUNT(ta.au_id) AS "authors" FROM authors a INNER JOIN title_authors ta
    -> ON a.au_id = ta.au_id GROUP BY ta.title_id HAVING COUNT(ta.au_id) > 1 ORDER BY ta.title_id ASC;
+----------+---------+
| title_id | authors |
+----------+---------+
| T04      |       2 |
| T07      |       2 |
| T11      |       3 |
+----------+---------+
3 rows in set (0.001 sec)
```

124. List all books whose income (price * sales) exceeds at least 10 times the advance received.

SELECT t.title_id, t.title_name, r.advance, t.price * t.sales AS "Revenue" FROM titles t INNER JOIN royalties r ON t.price * t.sales > r.advance * 10 AND t.title_id = r.title_id ORDER BY t.price * t.sales DESC;

```
MariaDB [sparksql]> SELECT t.title_id, t.title_name, r.advance, t.price * t.sales AS "Revenue" FROM titles t
    -> INNER JOIN royalties r ON t.price * t.sales > r.advance * 10 AND t.title_id = r.title_id
    -> ORDER BY t.price * t.sales DESC;
+----------+--------------------------------------+------------+-------------+
| title_id | title_name                           | advance    | Revenue     |
+----------+--------------------------------------+------------+-------------+
| T07      | I Blame My Mother                    | 1000000.00 | 35929790.00 |
| T05      | Exchange of Platitudes               |  100000.00 |  1400008.00 |
| T12      | Spontaneous, Not Annoying            |   50000.00 |  1299012.99 |
| T03      | Ask Your System Administrator        |   15000.00 |  1025396.65 |
| T13      | What Are The Civilian Applications?  |   20000.00 |   313905.33 |
| T06      | How About Never?                     |   20000.00 |   225834.00 |
| T02      | 200 Years of German Humor            |    1000.00 |   190841.70 |
| T09      | Kiss My Boo-Boo                      |       0.00 |    69750.00 |
| T08      | Just Wait Until After School         |       0.00 |    40950.00 |
+----------+--------------------------------------+------------+-------------+
9 rows in set (0.001 sec)
```

OUTER JOIN

125. List authors living in cities where there is a publisher

SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a INNER JOIN publishers p ON a.city = p.city ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC;

```
MariaDB [sparksql]> SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a INNER JOIN publishers p
    -> ON a.city = p.city ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC;
+-----------+---------+------------------+
| au_fname  | au_lname | pub_name         |
+-----------+---------+------------------+
| Christian | Kells    | Abatis Publishers |
| Hallie    | Hull     | Core Dump Books   |
| Klee      | Hull     | Core Dump Books   |
+-----------+---------+------------------+
3 rows in set (0.001 sec)
```

126. Left outer join to include all authors in the result regardless of whether there is an editor living in the same city. whether or not there is an editor living in the same city.

SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a LEFT OUTER JOIN publishers p ON a.city = p.city ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC;

```
MariaDB [sparksql]> SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a LEFT OUTER JOIN publishers p
    -> ON a.city = p.city ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC;
+-----------+-------------+------------------+
| au_fname  | au_lname    | pub_name         |
+-----------+-------------+------------------+
| Sarah     | Buchman     | NULL             |
| Wendy     | Heydemark   | NULL             |
|           | Kellsey     | NULL             |
| Paddy     | O'Furniture | NULL             |
| Christian | Kells       | Abatis Publishers |
| Hallie    | Hull        | Core Dump Books   |
| Klee      | Hull        | Core Dump Books   |
+-----------+-------------+------------------+
7 rows in set (0.005 sec)
```

127. Right outer join to include all publishers in the result regardless of whether there is an author living in the same city.

SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a RIGHT OUTER JOIN publishers p
ON a.city = p.city ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC;

```
MariaDB [sparksql]> SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a RIGHT OUTER JOIN publishers p
    -> ON a.city = p.city ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC;
+-----------+---------+--------------------+
| au_fname  | au_lname | pub_name           |
+-----------+---------+--------------------+
| Christian | Kells    | Abatis Publishers   |
| Hallie    | Hull     | Core Dump Books     |
| Klee      | Hull     | Core Dump Books     |
| NULL      | NULL     | Schadenfreude Press |
| NULL      | NULL     | Tenterhooks Press   |
+-----------+---------+--------------------+
5 rows in set (0.001 sec)
```

128. Full outer join to include all authors and publishers in the result regardless of whether an author lives in the same city as a publisher.

FULL OUTER JOIN is not implemented in MariaDB.

129. Equivalent to full outer join

SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a LEFT OUTER JOIN publishers p ON a.city = p.city UNION DISTINCT SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a RIGHT OUTER JOIN publishers p ON a.city = p.city ORDER BY pub_name ASC, au_lname ASC, au_fname ASC;

```
MariaDB [sparksql]> SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a LEFT OUTER JOIN publishers p
    -> ON a.city = p.city UNION DISTINCT SELECT a.au_fname, a.au_lname, p.pub_name FROM authors a
    -> RIGHT OUTER JOIN publishers p ON a.city = p.city ORDER BY pub_name ASC, au_lname ASC, au_fname ASC ;
+-----------+-------------+--------------------+
| au_fname  | au_lname    | pub_name           |
+-----------+-------------+--------------------+
| Sarah     | Buchman     | NULL               |
| Wendy     | Heydemark   | NULL               |
|           | Kellsey     | NULL               |
| Paddy     | O'Furniture | NULL               |
| Christian | Kells       | Abatis Publishers  |
| Hallie    | Hull        | Core Dump Books    |
| Klee      | Hull        | Core Dump Books    |
| NULL      | NULL        | Schadenfreude Press|
| NULL      | NULL        | Tenterhooks Press  |
+-----------+-------------+--------------------+
9 rows in set (0.026 sec)
```

130. Lists the number of books written by each author including authors who have not written any books

SELECT a.au_id, COUNT(ta.title_id) AS "Num books" FROM authors a LEFT OUTER JOIN title_authors ta ON a.au_id = ta.au_id GROUP BY a.au_id ORDER BY a.au_id ASC;

```
MariaDB [sparksql]> SELECT a.au_id, COUNT(ta.title_id) AS "Num books" FROM authors a LEFT OUTER JOIN title_authors ta

    -> ON a.au_id = ta.au_id GROUP BY a.au_id ORDER BY a.au_id ASC;
+-------+-----------+
| au_id | Num books |
+-------+-----------+
| A01   |         3 |
| A02   |         4 |
| A03   |         2 |
| A04   |         4 |
| A05   |         1 |
| A06   |         3 |
| A07   |         0 |
+-------+-----------+
7 rows in set (0.016 sec)
```

131. List of authors who have not written any books

SELECT a.au_id, a.au_fname, a.au_lname FROM authors a LEFT OUTER JOIN title_authors ta ON a.au_id = ta.au_id WHERE ta.au_id IS NULL;

```
MariaDB [sparksql]> SELECT a.au_id, a.au_fname, a.au_lname FROM authors a LEFT OUTER JOIN title_authors ta
    -> ON a.au_id = ta.au_id WHERE ta.au_id IS NULL;
+-------+----------+-------------+
| au_id | au_fname | au_lname    |
+-------+----------+-------------+
| A07   | Paddy    | O'Furniture |
+-------+----------+-------------+
1 row in set (0.004 sec)
```

132. It combines an inner join and a left outer join to list all authors and any possible books they have written along with their sales.
But it shows sales only for those books that sold more than 100,000 copies.

SELECT a.au_id aid, a.au_fname fname, a.au_lname lname, COALESCE(tta.title_id, 'N/A') t_id, COALESCE(tta.title_name, 'N/A') name, COALESCE(tta.sales, 'N/A') sales FROM authors a LEFT OUTER JOIN (SELECT ta.au_id, t.title_id, t.title_name, t.sales FROM title_authors ta INNER JOIN titles t ON t.title_id = ta.title_id WHERE sales > 100000) tta ON a.au_id = tta.au_id ORDER BY a.au_id ASC, tta.title_id ASC;

```
MariaDB [sparksql]> SELECT a.au_id aid, a.au_fname fname, a.au_lname lname, COALESCE(tta.title_id, 'N/A') t_id,
    -> COALESCE(tta.title_name, 'N/A') name, COALESCE(tta.sales, 'N/A') sales
    -> FROM authors a LEFT OUTER JOIN (SELECT ta.au_id, t.title_id, t.title_name, t.sales FROM title_authors ta
    -> INNER JOIN titles t ON t.title_id = ta.title_id WHERE sales > 100000) tta ON a.au_id = tta.au_id
    -> ORDER BY a.au_id ASC, tta.title_id ASC;
+-----+-----------+-------------+------+--------------------------+---------+
| aid | fname     | lname       | t_id | name                     | sales   |
+-----+-----------+-------------+------+--------------------------+---------+
| A01 | Sarah     | Buchman     | N/A  | N/A                      | N/A     |
| A02 | Wendy     | Heydemark   | T07  | I Blame My Mother        | 1500200 |
| A02 | Wendy     | Heydemark   | T12  | Spontaneous, Not Annoying | 100001  |
| A03 | Hallie    | Hull        | N/A  | N/A                      | N/A     |
| A04 | Klee      | Hull        | T05  | Exchange of Platitudes   | 201440  |
| A04 | Klee      | Hull        | T07  | I Blame My Mother        | 1500200 |
| A05 | Christian | Kells       | N/A  | N/A                      | N/A     |
| A06 |           | Kellsey     | N/A  | N/A                      | N/A     |
| A07 | Paddy     | O'Furniture | N/A  | N/A                      | N/A     |
+-----+-----------+-------------+------+--------------------------+---------+
9 rows in set (0.010 sec)
```

SELECT ta.au_id, t.title_id, t.title_name, t.sales FROM title_authors ta INNER JOIN titles t ON t.title_id = ta.title_id WHERE sales > 100000;

```
MariaDB [sparksql]> SELECT ta.au_id, t.title_id, t.title_name, t.sales FROM title_authors ta INNER JOIN titles t
    -> ON t.title_id = ta.title_id WHERE sales > 100000;
+-------+----------+--------------------------+---------+
| au_id | title_id | title_name               | sales   |
+-------+----------+--------------------------+---------+
| A04   | T05      | Exchange of Platitudes   |  201440 |
| A02   | T07      | I Blame My Mother        | 1500200 |
| A04   | T07      | I Blame My Mother        | 1500200 |
| A02   | T12      | Spontaneous, Not Annoying |  100001 |
+-------+----------+--------------------------+---------+
4 rows in set (0.002 sec)
```

Self Join

133. List the name of each employee along with the name of his or her manager.

SELECT e1.emp_name AS "Employee name", e2.emp_name AS "Boss name" FROM employee e1 INNER JOIN employee e2 ON e1.boss_id = e2.emp_id;

```
MariaDB [sparksql]> SELECT e1.emp_name AS "Employee name", e2.emp_name AS "Boss name" FROM employee e1
    -> INNER JOIN employee e2 ON e1.boss_id = e2.emp_id;
+-------------------+-------------+
| Employee name     | Boss name   |
+-------------------+-------------+
| Jocelyn Hitchcock | Lord Copper |
| Mr. Salter        | Lord Copper |
| William Boot      | Mr. Salter  |
| Mr. Corker        | Mr. Salter  |
+-------------------+-------------+
4 rows in set (0.008 sec)
```

134. Select authors who live in the same state as the author A04

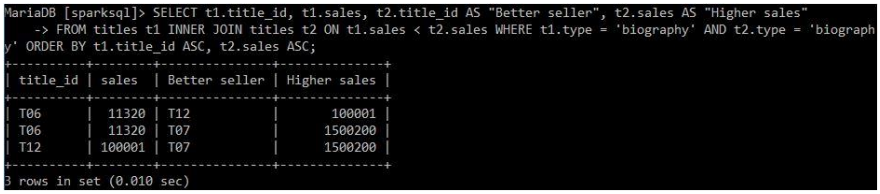
SELECT a1.au_id, a1.au_fname, a1.au_lname, a1.state FROM authors a1 INNER JOIN authors a2 ON a1.state = a2.state WHERE a2.au_id = 'A04';

```
MariaDB [sparksql]> SELECT a1.au_id, a1.au_fname, a1.au_lname, a1.state FROM authors a1 INNER JOIN authors a2
    -> ON a1.state = a2.state WHERE a2.au_id = 'A04';
+--------+----------+----------+-------+
| au_id | au_fname | au_lname | state |
+--------+----------+----------+-------+
| A03   | Hallie   | Hull     | CA    |
| A04   | Klee     | Hull     | CA    |
| A06   |          | Kellsey  | CA    |
+--------+----------+----------+-------+
3 rows in set (0.001 sec)
```

135. For each biography, it lists its id (title_id), its sales, and the other biographies that sold more than it. biographies that sold more than it.

SELECT t1.title_id, t1.sales, t2.title_id AS "Better seller", t2.sales AS "Higher

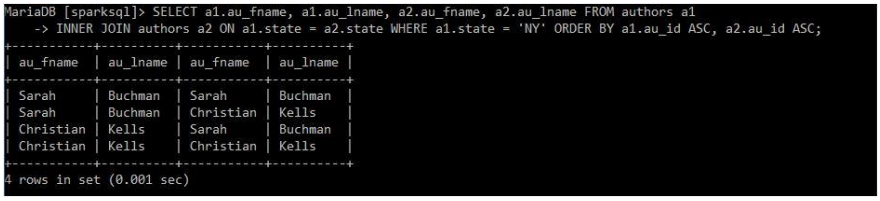
sales"
FROM titles t1 INNER JOIN titles t2 ON t1.sales < t2.sales WHERE t1.type = 'biography' AND t2.type = 'biography' ORDER BY t1.title_id ASC, t2.sales ASC;



```
MariaDB [sparksql]> SELECT t1.title_id, t1.sales, t2.title_id AS "Better seller", t2.sales AS "Higher sales"
    -> FROM titles t1 INNER JOIN titles t2 ON t1.sales < t2.sales WHERE t1.type = 'biography' AND t2.type = 'biograph
y' ORDER BY t1.title_id ASC, t2.sales ASC;
+----------+--------+---------------+-------------+
| title_id | sales  | Better seller | Higher sales |
+----------+--------+---------------+-------------+
| T06      |  11320 | T12           |      100001 |
| T06      |  11320 | T07           |     1500200 |
| T12      | 100001 | T07           |     1500200 |
+----------+--------+---------------+-------------+
3 rows in set (0.010 sec)
```

136. List the different pairs of authors who live in New York City.

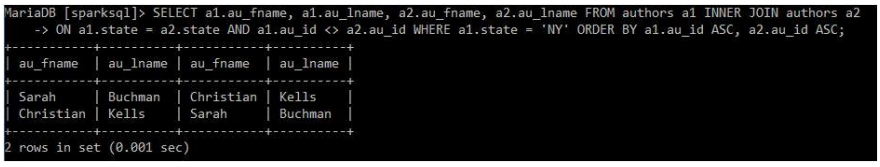
SELECT a1.au_fname, a1.au_lname, a2.au_fname, a2.au_lname FROM authors a1INNER JOIN authors a2 ON a1.state = a2.state WHERE a1.state = 'NY' ORDER BY a1.au_id ASC, a2.au_id ASC;



```
MariaDB [sparksql]> SELECT a1.au_fname, a1.au_lname, a2.au_fname, a2.au_lname FROM authors a1
    -> INNER JOIN authors a2 ON a1.state = a2.state WHERE a1.state = 'NY' ORDER BY a1.au_id ASC, a2.au_id ASC;
+-----------+----------+-----------+----------+
| au_fname  | au_lname | au_fname  | au_lname |
+-----------+----------+-----------+----------+
| Sarah     | Buchman  | Sarah     | Buchman  |
| Sarah     | Buchman  | Christian | Kells    |
| Christian | Kells    | Sarah     | Buchman  |
| Christian | Kells    | Christian | Kells    |
+-----------+----------+-----------+----------+
4 rows in set (0.001 sec)
```

137. Lists the different pairs of authors living in New York without redundancy.

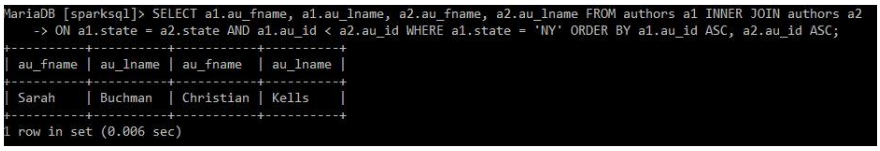
SELECT a1.au_fname, a1.au_lname, a2.au_fname, a2.au_lname FROM authors a1 INNER JOIN authors a2 ON a1.state = a2.state AND a1.au_id <> a2.au_id WHERE a1.state = 'NY' ORDER BY a1.au_id ASC, a2.au_id ASC;



```
MariaDB [sparksql]> SELECT a1.au_fname, a1.au_lname, a2.au_fname, a2.au_lname FROM authors a1 INNER JOIN authors a2
    -> ON a1.state = a2.state AND a1.au_id <> a2.au_id WHERE a1.state = 'NY' ORDER BY a1.au_id ASC, a2.au_id ASC;
+-----------+----------+-----------+----------+
| au_fname  | au_lname | au_fname  | au_lname |
+-----------+----------+-----------+----------+
| Sarah     | Buchman  | Christian | Kells    |
| Christian | Kells    | Sarah     | Buchman  |
+-----------+----------+-----------+----------+
2 rows in set (0.001 sec)
```

138. Lists the different pairs of authors living in New York without redundancies and without duplicate pairs.

SELECT a1.au_fname, a1.au_lname, a2.au_fname, a2.au_lname FROM authors a1 INNER JOIN authors a2
ON a1.state = a2.state AND a1.au_id < a2.au_id WHERE a1.state = 'NY' ORDER BY a1.au_id ASC, a2.au_id ASC;

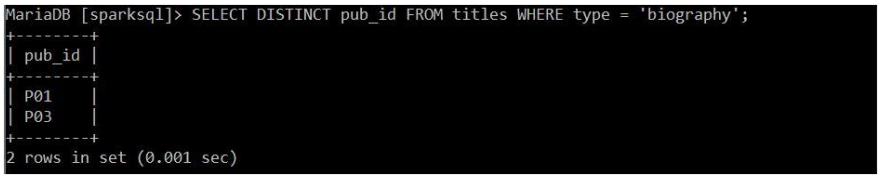


```
MariaDB [sparksql]> SELECT a1.au_fname, a1.au_lname, a2.au_fname, a2.au_lname FROM authors a1 INNER JOIN authors a2
    -> ON a1.state = a2.state AND a1.au_id < a2.au_id WHERE a1.state = 'NY' ORDER BY a1.au_id ASC, a2.au_id ASC;
+----------+----------+-----------+----------+
| au_fname | au_lname | au_fname  | au_lname |
+----------+----------+-----------+----------+
| Sarah    | Buchman  | Christian | Kells    |
+----------+----------+-----------+----------+
1 row in set (0.006 sec)
```

Sub-queries

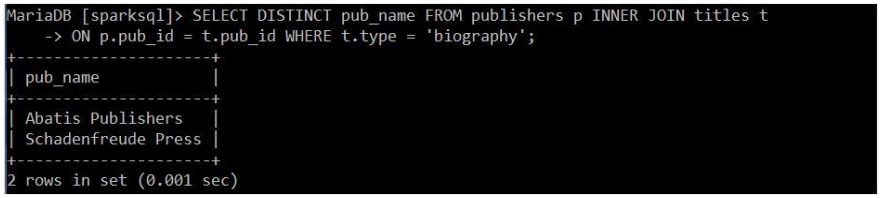
139. List the names of publishers that publish biographies (manual).

```
SELECT DISTINCT pub_id FROM titles WHERE type = 'biography';
```



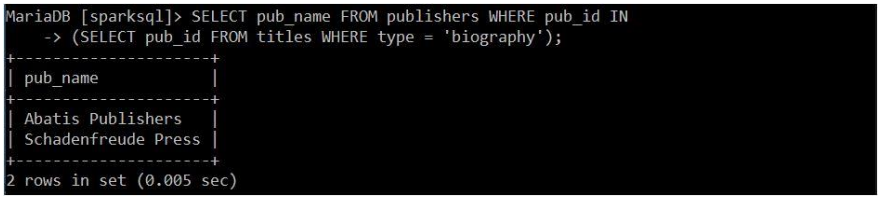
140. List the names of publishers who publish biographies (inner join).

```
SELECT DISTINCT pub_name FROM publishers p INNER JOIN titles t ON
p.pub_id = t.pub_id WHERE t.type = 'biography';
```



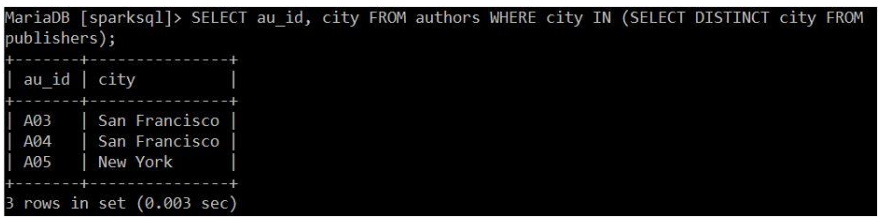
141. List the names of publishers who publish biographies (Sub-query).

```
SELECT pub_name FROM publishers WHERE pub_id IN
 (SELECT pub_id FROM titles WHERE type = 'biography');
```



142. Selecting authors who live in the same city as a publisher.

```
SELECT au_id, city FROM authors WHERE city IN (SELECT DISTINCT city FROM
publishers);
```



143. List authors who have not written any books.

SELECT au_id, au_fname, au_lname FROM authors WHERE au_id NOT IN (SELECT au_id FROM title_authors);

```
MariaDB [sparksql]> SELECT au_id, au_fname, au_lname FROM authors WHERE au_id NOT IN (SELECT
au_id FROM title_authors);
+-------+----------+-------------+
| au_id | au_fname | au_lname    |
+-------+----------+-------------+
| A07   | Paddy    | O'Furniture |
+-------+----------+-------------+
1 row in set (0.006 sec)
```

Simple sub-queries

144. List authors who live in the same state as the author with au_id = A04

SELECT au_id, au_fname, au_lname, state FROM authors WHERE state IN (SELECT state FROM authors WHERE au_id = 'A04');

```
MariaDB [sparksql]> SELECT au_id, au_fname, au_lname, state FROM authors WHERE state IN
    -> (SELECT state FROM authors WHERE au_id = 'A04');
+-------+----------+----------+-------+
| au_id | au_fname | au_lname | state |
+-------+----------+----------+-------+
| A03   | Hallie   | Hull     | CA    |
| A04   | Klee     | Hull     | CA    |
| A06   |          | Kellsey  | CA    |
+-------+----------+----------+-------+
3 rows in set (0.001 sec)
```

Correlated sub-queries

145. Lists books whose sales are equal to or exceed the average sales in their category.

SELECT t.title_id, t.type, t.sales FROM titles AS t WHERE sales >= (SELECT AVG(sales) FROM titles AS av WHERE av.type = t.type);

```
MariaDB [sparksql]> SELECT t.title_id, t.type, t.sales FROM titles AS t WHERE sales >=
    -> (SELECT AVG(sales) FROM titles AS av WHERE av.type = t.type);
+----------+------------+---------+
| title_id | type       | sales   |
+----------+------------+---------+
| T02      | history    |    9566 |
| T03      | computer   |   25667 |
| T05      | psychology |  201440 |
| T07      | biography  | 1500200 |
| T09      | children   |    5000 |
| T13      | history    |   10467 |
+----------+------------+---------+
6 rows in set (0.005 sec)
```

151. Changes the contract value to 0 for all rows in the titles table.

UPDATE titles SET contract = 0 ;

```
MariaDB [sparksql]> UPDATE titles SET contract = 0 ;
Query OK, 12 rows affected (0.026 sec)
Rows matched: 13  Changed: 12  Warnings: 0
```

152. Double the price of history books

```
UPDATE titles SET price = price * 2.0 WHERE type = 'history';
```

```
MariaDB [sparksql]> UPDATE titles SET price = price * 2.0 WHERE type = 'history';
Query OK, 3 rows affected (0.005 sec)
Rows matched: 3  Changed: 3  Warnings: 0
```

153. Update the type and pages columns for psychology books.

```
UPDATE titles SET type = 'self help', pages = NULL WHERE type =
'psychology';
```

```
MariaDB [sparksql]> UPDATE titles SET type = 'self help', pages = NULL WHERE type = 'psycholo
gy';
Query OK, 3 rows affected (0.009 sec)
Rows matched: 3  Changed: 3  Warnings: 0
```

154. Halve the sales of all books whose sales are above average.

```
UPDATE titles SET sales = sales * 0.5 WHERE sales > ( SELECT upper FROM (
SELECT AVG(sales) as upper FROM titles) as avgg );
```

```
MariaDB [sparksql]> UPDATE titles SET sales = sales * 0.5 WHERE
    -> sales > ( SELECT upper FROM ( SELECT AVG(sales) as upper FROM titles) as avgg );
Query OK, 2 rows affected (0.005 sec)
Rows matched: 2  Changed: 2  Warnings: 0
```

155. Change the publication date of all books by Sarah Buchman.

```
UPDATE titles SET pubdate = '2003-01-01' WHERE title_id IN (SELECT
title_id FROM title_authors WHERE au_id IN (SELECT au_id FROM authors
WHERE au_fname = 'Sarah' AND au_lname = 'Buchman'));
```

```
MariaDB [sparksql]> UPDATE titles SET pubdate = '2003-01-01' WHERE title_id IN (SELECT title_
id FROM title_authors WHERE au_id IN (SELECT au_id FROM authors WHERE au_fname = 'Sarah' AND
au_lname = 'Buchman'));
Query OK, 3 rows affected (0.008 sec)
Rows matched: 3  Changed: 3  Warnings: 0
```

156. Change the publisher of all Tenterhooks Press books to Abatis Publishers.

```
UPDATE titles SET pub_id = (SELECT pub_id FROM publishers
WHERE pub_name = 'Abatis Publishers') WHERE pub_id = (SELECT pub_id
FROM publishers WHERE pub_name = 'Tenterhooks Press');
```

```
MariaDB [sparksql]> UPDATE titles SET pub_id = (SELECT pub_id FROM publishers
    -> WHERE pub_name = 'Abatis Publishers') WHERE pub_id = (SELECT pub_id
    -> FROM publishers WHERE pub_name = 'Tenterhooks Press');
Query OK, 0 rows affected (0.008 sec)
Rows matched: 0  Changed: 0  Warnings: 0
```

Vistas

183. Create views with only some columns of the table authors.

```
CREATE VIEW au_names AS SELECT au_id, au_fname, au_lname FROM
authors;
```

```
MariaDB [sparksql]> CREATE VIEW au_names AS SELECT au_id, au_fname, au_lname FROM authors;
Query OK, 0 rows affected (0.019 sec)
```

184. Create a view with authors living in a city where there is at least one publisher.
at least one publisher.

```
CREATE VIEW cities (au_id, au_city, pub_id, pub_city) AS SELECT a.au_id,
a.city, p.pub_id, p.city FROM authors a INNER JOIN publishers p ON a.city =
p.city;
```

```
MariaDB [sparksql]> CREATE VIEW cities (au_id, au_city, pub_id, pub_city) AS SELECT a.au_id, a.city, p.pub_id, p.city
    -> FROM authors a INNER JOIN publishers p ON a.city = p.city;
Query OK, 0 rows affected (0.015 sec)
```

Analytical functions

185. Example table

```
CREATE TABLE test_scores (
 name varchar(20), test varchar(20), score tinyint );
```

```
MariaDB [sparksql]> DESC test_scores;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| name  | varchar(20) | YES  |     | NULL    |       |
| test  | varchar(20) | YES  |     | NULL    |       |
| score | tinyint(4)  | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set (0.007 sec)
```

```
INSERT INTO test_scores VALUES
 ("Steve", "SQL", 75),
 ("Robert", "SQL", 43),
 ("Tracy", "SQL", 56),
 ("Tatiana", "SQL", 87),
 ("Steve", "Tuning", 83),
 ("Robert", "Tuning", 31),
```

("Tracy", "Tuning", 88),
("Tatiana", "Tuning", 83);

```
MariaDB [sparksql]> SELECT * FROM test_scores;
+---------+--------+-------+
| name    | test   | score |
+---------+--------+-------+
| Steve   | SQL    |    75 |
| Robert  | SQL    |    43 |
| Tracy   | SQL    |    56 |
| Tatiana | SQL    |    87 |
| Steve   | Tuning |    83 |
| Robert  | Tuning |    31 |
| Tracy   | Tuning |    88 |
| Tatiana | Tuning |    83 |
+---------+--------+-------+
8 rows in set (0.001 sec)
```

186. Average per test

SELECT name, test, score, AVG(score) OVER (PARTITION BY test) AS avgtest FROM test_scores;

```
MariaDB [sparksql]> SELECT name, test, score, AVG(score) OVER (PARTITION BY test) AS avgtest FROM test_scores;
+---------+--------+-------+---------+
| name    | test   | score | avgtest |
+---------+--------+-------+---------+
| Tatiana | SQL    |    87 | 65.2500 |
| Steve   | SQL    |    75 | 65.2500 |
| Robert  | SQL    |    43 | 65.2500 |
| Tracy   | SQL    |    56 | 65.2500 |
| Tatiana | Tuning |    83 | 71.2500 |
| Steve   | Tuning |    83 | 71.2500 |
| Robert  | Tuning |    31 | 71.2500 |
| Tracy   | Tuning |    88 | 71.2500 |
+---------+--------+-------+---------+
8 rows in set (0.005 sec)
```

187. Average by name.

SELECT name, test, score, AVG(score) OVER (PARTITION BY name) AS avgname FROM test_scores;

```
MariaDB [sparksql]> SELECT name, test, score, AVG(score) OVER (PARTITION BY name) AS avgname FROM test_scores;
+---------+--------+-------+---------+
| name    | test   | score | avgname |
+---------+--------+-------+---------+
| Robert  | Tuning |    31 | 37.0000 |
| Robert  | SQL    |    43 | 37.0000 |
| Steve   | Tuning |    83 | 79.0000 |
| Steve   | SQL    |    75 | 79.0000 |
| Tatiana | Tuning |    83 | 85.0000 |
| Tatiana | SQL    |    87 | 85.0000 |
| Tracy   | Tuning |    88 | 72.0000 |
| Tracy   | SQL    |    56 | 72.0000 |
+---------+--------+-------+---------+
8 rows in set (0.001 sec)
```

188. Average by test and name

SELECT name, test, score, AVG(score) OVER (PARTITION BY test) AS avgtest, AVG(score) OVER (PARTITION BY name) AS avgname FROM test_scores;

```
MariaDB [sparksql]> SELECT name, test, score, AVG(score) OVER (PARTITION BY test) AS avgtest, AVG(score) OVER (PARTITION
BY name) AS avgname FROM test_scores;
+---------+--------+-------+---------+---------+
| name    | test   | score | avgtest | avgname |
+---------+--------+-------+---------+---------+
| Tatiana | SQL    |    87 | 65.2500 | 85.0000 |
| Steve   | SQL    |    75 | 65.2500 | 79.0000 |
| Robert  | SQL    |    43 | 65.2500 | 37.0000 |
| Tracy   | SQL    |    56 | 65.2500 | 72.0000 |
| Tatiana | Tuning |    83 | 71.2500 | 85.0000 |
| Steve   | Tuning |    83 | 71.2500 | 79.0000 |
| Robert  | Tuning |    31 | 71.2500 | 37.0000 |
| Tracy   | Tuning |    88 | 71.2500 | 72.0000 |
+---------+--------+-------+---------+---------+
8 rows in set (0.000 sec)
```

189. Order by score (row_number).

```
SELECT row_number() OVER (order BY score DESC) AS pos, name, test, score
FROM test_scores;
```

```
MariaDB [sparksql]> SELECT row_number() OVER (order BY score DESC) AS pos, name, test, score FROM test_scores;
+-----+---------+--------+-------+
| pos | name    | test   | score |
+-----+---------+--------+-------+
|   1 | Tracy   | Tuning |    88 |
|   2 | Tatiana | SQL    |    87 |
|   3 | Tatiana | Tuning |    83 |
|   4 | Steve   | Tuning |    83 |
|   5 | Steve   | SQL    |    75 |
|   6 | Tracy   | SQL    |    56 |
|   7 | Robert  | SQL    |    43 |
|   8 | Robert  | Tuning |    31 |
+-----+---------+--------+-------+
8 rows in set (0.003 sec)
```

190. Order by rank

```
SELECT rank() OVER (order BY score DESC) AS pos, name, test, score FROM
test_scores;
```

```
MariaDB [sparksql]> SELECT rank() OVER (order BY score DESC) AS pos, name, test, score FROM test_scores;
+-----+---------+--------+-------+
| pos | name    | test   | score |
+-----+---------+--------+-------+
|   1 | Tracy   | Tuning |    88 |
|   2 | Tatiana | SQL    |    87 |
|   3 | Tatiana | Tuning |    83 |
|   3 | Steve   | Tuning |    83 |
|   5 | Steve   | SQL    |    75 |
|   6 | Tracy   | SQL    |    56 |
|   7 | Robert  | SQL    |    43 |
|   8 | Robert  | Tuning |    31 |
+-----+---------+--------+-------+
8 rows in set (0.001 sec)
```

191. Order by score (dense_rank).

```
SELECT dense_rank() OVER (order BY score DESC) AS pos, name, test, score
FROM test_scores;
```

```
MariaDB [sparksql]> SELECT dense_rank() OVER (order BY score DESC) AS pos, name, test, score FROM test_scores;
+-----+---------+--------+-------+
| pos | name    | test   | score |
+-----+---------+--------+-------+
|   1 | Tracy   | Tuning |    88 |
|   2 | Tatiana | SQL    |    87 |
|   3 | Tatiana | Tuning |    83 |
|   3 | Steve   | Tuning |    83 |
|   4 | Steve   | SQL    |    75 |
|   5 | Tracy   | SQL    |    56 |
|   6 | Robert  | SQL    |    43 |
|   7 | Robert  | Tuning |    31 |
+-----+---------+--------+-------+
8 rows in set (0.000 sec)
```

192. Order by test and score (row_number).

```
SELECT row_number() OVER (partition by test order BY score DESC) AS pos,
name, test, score FROM test_scores;
```

```
MariaDB [sparksql]> SELECT row_number() OVER (partition by test order BY score DESC) AS pos,
    -> name, test, score FROM test_scores;
+-----+---------+--------+-------+
| pos | name    | test   | score |
+-----+---------+--------+-------+
|   1 | Tatiana | SQL    |    87 |
|   2 | Steve   | SQL    |    75 |
|   3 | Tracy   | SQL    |    56 |
|   4 | Robert  | SQL    |    43 |
|   1 | Tracy   | Tuning |    88 |
|   2 | Tatiana | Tuning |    83 |
|   3 | Steve   | Tuning |    83 |
|   4 | Robert  | Tuning |    31 |
+-----+---------+--------+-------+
8 rows in set (0.001 sec)
```

193. Order by test and rank.

SELECT rank() OVER (partition by test order BY score DESC) AS pos, name, test, score FROM test_scores;

```
MariaDB [sparksql]> SELECT rank() OVER (partition by test order BY score DESC) AS pos,
    -> name, test, score FROM test_scores;
+-----+---------+--------+-------+
| pos | name    | test   | score |
+-----+---------+--------+-------+
|   1 | Tatiana | SQL    |    87 |
|   2 | Steve   | SQL    |    75 |
|   3 | Tracy   | SQL    |    56 |
|   4 | Robert  | SQL    |    43 |
|   1 | Tracy   | Tuning |    88 |
|   2 | Tatiana | Tuning |    83 |
|   2 | Steve   | Tuning |    83 |
|   4 | Robert  | Tuning |    31 |
+-----+---------+--------+-------+
8 rows in set (0.011 sec)
```

194. Order by test and score (dense_rank).

SELECT dense_rank() OVER (partition by test order BY score DESC) AS pos, name, test, score FROM test_scores;

```
MariaDB [sparksql]> SELECT dense_rank() OVER (partition by test order BY score DESC) AS pos,
    -> name, test, score FROM test_scores;
+-----+---------+--------+-------+
| pos | name    | test   | score |
+-----+---------+--------+-------+
|   1 | Tatiana | SQL    |    87 |
|   2 | Steve   | SQL    |    75 |
|   3 | Tracy   | SQL    |    56 |
|   4 | Robert  | SQL    |    43 |
|   1 | Tracy   | Tuning |    88 |
|   2 | Tatiana | Tuning |    83 |
|   2 | Steve   | Tuning |    83 |
|   3 | Robert  | Tuning |    31 |
+-----+---------+--------+-------+
```

195. Compare score with the next (lead).

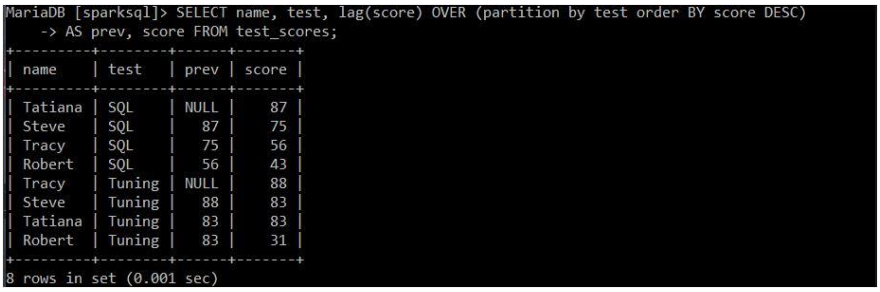
SELECT name, test, score, lead(score) OVER (partition by test order BY score DESC) AS next FROM test_scores;

```
MariaDB [sparksql]> SELECT name, test, score, lead(score) OVER (partition by test order BY score DESC)

    -> AS next FROM test_scores;
+---------+--------+-------+------+
| name    | test   | score | next |
+---------+--------+-------+------+
| Tatiana | SQL    |    87 |   75 |
| Steve   | SQL    |    75 |   56 |
| Tracy   | SQL    |    56 |   43 |
| Robert  | SQL    |    43 | NULL |
| Tracy   | Tuning |    88 |   83 |
| Steve   | Tuning |    83 |   83 |
| Tatiana | Tuning |    83 |   31 |
| Robert  | Tuning |    31 | NULL |
+---------+--------+-------+------+
8 rows in set (0.001 sec)
```

196. Compare score with previous (lag).

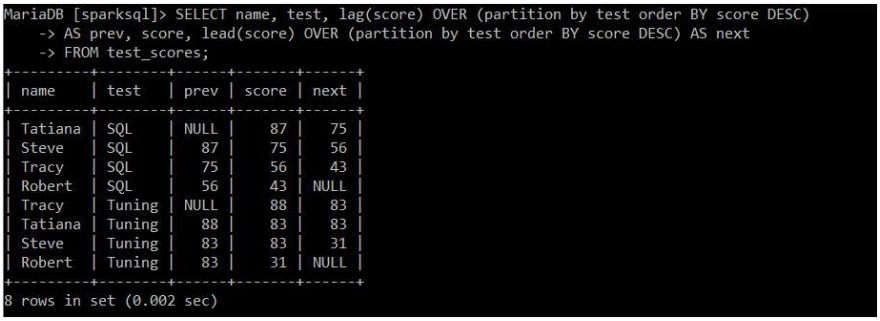
```
SELECT name, test, lag(score) OVER (partition by test order BY score DESC)
AS prev, score FROM test_scores;
```



```
MariaDB [sparksql]> SELECT name, test, lag(score) OVER (partition by test order BY score DESC)
    -> AS prev, score FROM test_scores;
+---------+--------+------+-------+
| name    | test   | prev | score |
+---------+--------+------+-------+
| Tatiana | SQL    | NULL |    87 |
| Steve   | SQL    |   87 |    75 |
| Tracy   | SQL    |   75 |    56 |
| Robert  | SQL    |   56 |    43 |
| Tracy   | Tuning | NULL |    88 |
| Steve   | Tuning |   88 |    83 |
| Tatiana | Tuning |   83 |    83 |
| Robert  | Tuning |   83 |    31 |
+---------+--------+------+-------+
8 rows in set (0.001 sec)
```

197. Compare score with previous score (lag) and next score (lead).

```
SELECT name, test, lag(score) OVER (partition by test order BY score DESC)
AS prev, score, lead(score) OVER (partition by test order BY score DESC) AS
next FROM test_scores;
```



```
MariaDB [sparksql]> SELECT name, test, lag(score) OVER (partition by test order BY score DESC)
    -> AS prev, score, lead(score) OVER (partition by test order BY score DESC) AS next
    -> FROM test_scores;
+---------+--------+------+-------+------+
| name    | test   | prev | score | next |
+---------+--------+------+-------+------+
| Tatiana | SQL    | NULL |    87 |   75 |
| Steve   | SQL    |   87 |    75 |   56 |
| Tracy   | SQL    |   75 |    56 |   43 |
| Robert  | SQL    |   56 |    43 | NULL |
| Tracy   | Tuning | NULL |    88 |   83 |
| Tatiana | Tuning |   88 |    83 |   83 |
| Steve   | Tuning |   83 |    83 |   31 |
| Robert  | Tuning |   83 |    31 | NULL |
+---------+--------+------+-------+------+
8 rows in set (0.002 sec)
```

198. Table example.

```
CREATE TABLE IF NOT EXISTS emp (
 empno DECIMAL(4), ename VARCHAR(10), job VARCHAR(9),
 mgr DECIMAL(4), hiredate CHAR(10), sal DECIMAL(7,2),
 comm DECIMAL(7,2), deptno DECIMAL(2),
 CONSTRAINT pk_emp PRIMARY KEY (empno));
```

```
MariaDB [sparksql]> DESC emp;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| empno    | decimal(4,0) | NO   | PRI | NULL    |       |
| ename    | varchar(10)  | YES  |     | NULL    |       |
| job      | varchar(9)   | YES  |     | NULL    |       |
| mgr      | decimal(4,0) | YES  |     | NULL    |       |
| hiredate | char(10)     | YES  |     | NULL    |       |
| sal      | decimal(7,2) | YES  |     | NULL    |       |
| comm     | decimal(7,2) | YES  |     | NULL    |       |
| deptno   | decimal(2,0) | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
8 rows in set (0.009 sec)
```

INSERT INTO emp VALUES
(7369, 'SMITH', 'CLERK', 7902, '1980-12-17', 800, NULL, 20),
(7499, 'ALLEN', 'SALESMAN', 7698, '1981-02-20', 1600, 300, 30),
(7521, 'WARD', 'SALESMAN', 7698, '1981-02-22', 1250, 500, 30),
(7566, 'JONES', 'MANAGER', 7839, '1981-04-02', 2975, NULL, 20),
(7654, 'MARTIN', 'SALESMAN', 7698, '1981-09-28', 1250, 1400,30),
(7698, 'BLAKE', 'MANAGER', 7839, '1981-05-01', 2850, NULL, 30),
(7782, 'CLARK', 'MANAGER', 7839, '1981-06-09', 2450, NULL, 10),
(7788, 'SCOTT', 'ANALYST', 7566, '1987-07-13', 3000, NULL, 20),
(7839, 'KING', 'PRESIDENT', NULL, '1981-11-17', 5000, NULL, 10),
(7844, 'TURNER', 'SALESMAN', 7698, '1981-09-08', 1500, 0, 30),
(7876, 'ADAMS', 'CLERK', 7788, '1987-07-13', 1100, NULL, 20),
(7900, 'JAMES', 'CLERK', 7698, '1981-12-03', 950, NULL, 30),
(7902, 'FORD', 'ANALYST', 7566, '1981-12-03', 3000, NULL, 20),
(7934, 'MILLER', 'CLERK', 7782, '1982-01-23', 1300, NULL, 10) ;

```
MariaDB [sparksql]> SELECT * FROM emp;
+-------+--------+-----------+------+------------+---------+---------+--------+
| empno | ename  | job       | mgr  | hiredate   | sal     | comm    | deptno |
+-------+--------+-----------+------+------------+---------+---------+--------+
|  7369 | SMITH  | CLERK     | 7902 | 1980-12-17 |  800.00 |    NULL |     20 |
|  7499 | ALLEN  | SALESMAN  | 7698 | 1981-02-20 | 1600.00 |  300.00 |     30 |
|  7521 | WARD   | SALESMAN  | 7698 | 1981-02-22 | 1250.00 |  500.00 |     30 |
|  7566 | JONES  | MANAGER   | 7839 | 1981-04-02 | 2975.00 |    NULL |     20 |
|  7654 | MARTIN | SALESMAN  | 7698 | 1981-09-28 | 1250.00 | 1400.00 |     30 |
|  7698 | BLAKE  | MANAGER   | 7839 | 1981-05-01 | 2850.00 |    NULL |     30 |
|  7782 | CLARK  | MANAGER   | 7839 | 1981-06-09 | 2450.00 |    NULL |     10 |
|  7788 | SCOTT  | ANALYST   | 7566 | 1987-07-13 | 3000.00 |    NULL |     20 |
|  7839 | KING   | PRESIDENT | NULL | 1981-11-17 | 5000.00 |    NULL |     10 |
|  7844 | TURNER | SALESMAN  | 7698 | 1981-09-08 | 1500.00 |    0.00 |     30 |
|  7876 | ADAMS  | CLERK     | 7788 | 1987-07-13 | 1100.00 |    NULL |     20 |
|  7900 | JAMES  | CLERK     | 7698 | 1981-12-03 |  950.00 |    NULL |     30 |
|  7902 | FORD   | ANALYST   | 7566 | 1981-12-03 | 3000.00 |    NULL |     20 |
|  7934 | MILLER | CLERK     | 7782 | 1982-01-23 | 1300.00 |    NULL |     10 |
+-------+--------+-----------+------+------------+---------+---------+--------+
14 rows in set (0.001 sec)
```

199. Salary of an employee together with the average salary of his or her department.

SELECT empno, ename, deptno, sal, AVG(sal) OVER (PARTITION BY deptno) AS avg_dept_sal FROM emp;

```
MariaDB [sparksql]> SELECT empno, ename, deptno, sal, AVG(sal) OVER (PARTITION BY deptno) AS avg_dept_
sal FROM emp;
+-------+--------+--------+---------+--------------+
| empno | ename  | deptno | sal     | avg_dept_sal |
+-------+--------+--------+---------+--------------+
|  7934 | MILLER |     10 | 1300.00 |  2916.666667 |
|  7782 | CLARK  |     10 | 2450.00 |  2916.666667 |
|  7839 | KING   |     10 | 5000.00 |  2916.666667 |
|  7788 | SCOTT  |     20 | 3000.00 |  2175.000000 |
|  7369 | SMITH  |     20 |  800.00 |  2175.000000 |
|  7876 | ADAMS  |     20 | 1100.00 |  2175.000000 |
|  7566 | JONES  |     20 | 2975.00 |  2175.000000 |
|  7902 | FORD   |     20 | 3000.00 |  2175.000000 |
|  7698 | BLAKE  |     30 | 2850.00 |  1566.666667 |
|  7499 | ALLEN  |     30 | 1600.00 |  1566.666667 |
|  7844 | TURNER |     30 | 1500.00 |  1566.666667 |
|  7521 | WARD   |     30 | 1250.00 |  1566.666667 |
|  7900 | JAMES  |     30 |  950.00 |  1566.666667 |
|  7654 | MARTIN |     30 | 1250.00 |  1566.666667 |
+-------+--------+--------+---------+--------------+
14 rows in set (0.024 sec)
```

200. Salary of an employee together with the average salary of all employees.

SELECT empno, ename, deptno, sal, AVG(sal) OVER () AS avg_dept_sal FROM emp;

```
MariaDB [sparksql]> SELECT empno, ename, deptno, sal, AVG(sal) OVER () AS avg_dept_sal FROM emp;
+-------+--------+--------+---------+--------------+
| empno | ename  | deptno | sal     | avg_dept_sal |
+-------+--------+--------+---------+--------------+
|  7369 | SMITH  |     20 |  800.00 |  2073.214286 |
|  7900 | JAMES  |     30 |  950.00 |  2073.214286 |
|  7876 | ADAMS  |     20 | 1100.00 |  2073.214286 |
|  7521 | WARD   |     30 | 1250.00 |  2073.214286 |
|  7654 | MARTIN |     30 | 1250.00 |  2073.214286 |
|  7934 | MILLER |     10 | 1300.00 |  2073.214286 |
|  7844 | TURNER |     30 | 1500.00 |  2073.214286 |
|  7499 | ALLEN  |     30 | 1600.00 |  2073.214286 |
|  7782 | CLARK  |     10 | 2450.00 |  2073.214286 |
|  7698 | BLAKE  |     30 | 2850.00 |  2073.214286 |
|  7566 | JONES  |     20 | 2975.00 |  2073.214286 |
|  7788 | SCOTT  |     20 | 3000.00 |  2073.214286 |
|  7902 | FORD   |     20 | 3000.00 |  2073.214286 |
|  7839 | KING   |     10 | 5000.00 |  2073.214286 |
+-------+--------+--------+---------+--------------+
14 rows in set (0.004 sec)
```

201. Salary of an employee together with the department's minimum wage.

SELECT empno, ename, deptno, sal, MIN(sal) OVER (PARTITION BY deptno order by sal) AS min_dept_sal FROM emp ;

```
MariaDB [sparksql]> SELECT empno, ename, deptno, sal, AVG(sal) OVER () AS avg_dept_sal FROM emp;
+-------+--------+--------+---------+--------------+
| empno | ename  | deptno | sal     | avg_dept_sal |
+-------+--------+--------+---------+--------------+
|  7369 | SMITH  |     20 |  800.00 |  2073.214286 |
|  7900 | JAMES  |     30 |  950.00 |  2073.214286 |
|  7876 | ADAMS  |     20 | 1100.00 |  2073.214286 |
|  7521 | WARD   |     30 | 1250.00 |  2073.214286 |
|  7654 | MARTIN |     30 | 1250.00 |  2073.214286 |
|  7934 | MILLER |     10 | 1300.00 |  2073.214286 |
|  7844 | TURNER |     30 | 1500.00 |  2073.214286 |
|  7499 | ALLEN  |     30 | 1600.00 |  2073.214286 |
|  7782 | CLARK  |     10 | 2450.00 |  2073.214286 |
|  7698 | BLAKE  |     30 | 2850.00 |  2073.214286 |
|  7566 | JONES  |     20 | 2975.00 |  2073.214286 |
|  7788 | SCOTT  |     20 | 3000.00 |  2073.214286 |
|  7902 | FORD   |     20 | 3000.00 |  2073.214286 |
|  7839 | KING   |     10 | 5000.00 |  2073.214286 |
+-------+--------+--------+---------+--------------+
14 rows in set (0.004 sec)
```

202. Salary of an employee together with the difference between his or her salary and the department's minimum wage.

SELECT empno, ename, deptno, sal, sal – MIN(sal) OVER (PARTITION BY deptno order by sal) AS diff_min FROM emp ;

```
MariaDB [sparksql]> SELECT empno, ename, deptno, sal, sal - MIN(sal) OVER (PARTITION BY deptno order by
 sal) AS diff_min FROM emp ;
+-------+--------+--------+---------+----------+
| empno | ename  | deptno | sal     | diff_min |
+-------+--------+--------+---------+----------+
|  7934 | MILLER |     10 | 1300.00 |     0.00 |
|  7782 | CLARK  |     10 | 2450.00 |  1150.00 |
|  7839 | KING   |     10 | 5000.00 |  3700.00 |
|  7369 | SMITH  |     20 |  800.00 |     0.00 |
|  7876 | ADAMS  |     20 | 1100.00 |   300.00 |
|  7566 | JONES  |     20 | 2975.00 |  2175.00 |
|  7902 | FORD   |     20 | 3000.00 |  2200.00 |
|  7788 | SCOTT  |     20 | 3000.00 |  2200.00 |
|  7900 | JAMES  |     30 |  950.00 |     0.00 |
|  7654 | MARTIN |     30 | 1250.00 |   300.00 |
|  7521 | WARD   |     30 | 1250.00 |   300.00 |
|  7844 | TURNER |     30 | 1500.00 |   550.00 |
|  7499 | ALLEN  |     30 | 1600.00 |   650.00 |
|  7698 | BLAKE  |     30 | 2850.00 |  1900.00 |
+-------+--------+--------+---------+----------+
14 rows in set (0.002 sec)
```

203. Salary of an employee together with the highest salary of all employees hired up to that day.

SELECT empno, hiredate, ename, sal, MAX(sal) OVER (order by hiredate) AS max FROM emp ;

```
MariaDB [sparksql]> SELECT empno, hiredate, ename, sal, MAX(sal) OVER (order by hiredate) AS max
    -> FROM emp ;
+-------+------------+--------+---------+---------+
| empno | hiredate   | ename  | sal     | max     |
+-------+------------+--------+---------+---------+
|  7369 | 1980-12-17 | SMITH  |  800.00 |  800.00 |
|  7499 | 1981-02-20 | ALLEN  | 1600.00 | 1600.00 |
|  7521 | 1981-02-22 | WARD   | 1250.00 | 1600.00 |
|  7566 | 1981-04-02 | JONES  | 2975.00 | 2975.00 |
|  7698 | 1981-05-01 | BLAKE  | 2850.00 | 2975.00 |
|  7782 | 1981-06-09 | CLARK  | 2450.00 | 2975.00 |
|  7844 | 1981-09-08 | TURNER | 1500.00 | 2975.00 |
|  7654 | 1981-09-28 | MARTIN | 1250.00 | 2975.00 |
|  7839 | 1981-11-17 | KING   | 5000.00 | 5000.00 |
|  7900 | 1981-12-03 | JAMES  |  950.00 | 5000.00 |
|  7902 | 1981-12-03 | FORD   | 3000.00 | 5000.00 |
|  7934 | 1982-01-23 | MILLER | 1300.00 | 5000.00 |
|  7876 | 1987-07-13 | ADAMS  | 1100.00 | 5000.00 |
|  7788 | 1987-07-13 | SCOTT  | 3000.00 | 5000.00 |
+-------+------------+--------+---------+---------+
14 rows in set (0.001 sec)
```

204. Salary of an employee together with the average between the salary 1) of the employee hired before, 2) of the employee himself/herself, and 3) of the employee hired after.

SELECT empno, hiredate, ename, sal, AVG(sal) OVER (order by hiredate rows between 1 preceding and 1 following) AS avg FROM emp ;

```
MariaDB [sparksql]> SELECT empno, hiredate, ename, sal, AVG(sal) OVER (order by hiredate rows between
 preceding and 1 following ) AS avg FROM emp ;
+-------+------------+--------+---------+-------------+
| empno | hiredate   | ename  | sal     | avg         |
+-------+------------+--------+---------+-------------+
|  7369 | 1980-12-17 | SMITH  |  800.00 | 1200.000000 |
|  7499 | 1981-02-20 | ALLEN  | 1600.00 | 1216.666667 |
|  7521 | 1981-02-22 | WARD   | 1250.00 | 1941.666667 |
|  7566 | 1981-04-02 | JONES  | 2975.00 | 2358.333333 |
|  7698 | 1981-05-01 | BLAKE  | 2850.00 | 2758.333333 |
|  7782 | 1981-06-09 | CLARK  | 2450.00 | 2266.666667 |
|  7844 | 1981-09-08 | TURNER | 1500.00 | 1733.333333 |
|  7654 | 1981-09-28 | MARTIN | 1250.00 | 2583.333333 |
|  7839 | 1981-11-17 | KING   | 5000.00 | 2400.000000 |
|  7900 | 1981-12-03 | JAMES  |  950.00 | 2983.333333 |
|  7902 | 1981-12-03 | FORD   | 3000.00 | 1750.000000 |
|  7934 | 1982-01-23 | MILLER | 1300.00 | 1800.000000 |
|  7876 | 1987-07-13 | ADAMS  | 1100.00 | 1800.000000 |
|  7788 | 1987-07-13 | SCOTT  | 3000.00 | 2050.000000 |
+-------+------------+--------+---------+-------------+
14 rows in set (0.002 sec)
```

205. Salary of an employee together with the average between the salary 1) of the employees hired on the last date that there was hiring prior to the employee's 2)of the employee himself/herself and the employees hired on the same day, and 3) of the employees hired on the next date of hire after the employee's date of hire.

CREATE VIEW IF NOT EXISTS ranked as SELECT dense_rank() OVER(ORDER BY hiredate) rank, empno, hiredate, ename, sal FROM emp ;

SELECT * FROM ranked ;

```
MariaDB [sparksql]> CREATE VIEW IF NOT EXISTS ranked as SELECT dense_rank() OVER( ORDER BY hiredate )
 rank, empno, hiredate, ename, sal FROM emp ;
Query OK, 0 rows affected (0.017 sec)

MariaDB [sparksql]> SELECT * FROM ranked ;
+------+-------+------------+--------+---------+
| rank | empno | hiredate   | ename  | sal     |
+------+-------+------------+--------+---------+
|    1 |  7369 | 1980-12-17 | SMITH  |  800.00 |
|    2 |  7499 | 1981-02-20 | ALLEN  | 1600.00 |
|    3 |  7521 | 1981-02-22 | WARD   | 1250.00 |
|    4 |  7566 | 1981-04-02 | JONES  | 2975.00 |
|    5 |  7698 | 1981-05-01 | BLAKE  | 2850.00 |
|    6 |  7782 | 1981-06-09 | CLARK  | 2450.00 |
|    7 |  7844 | 1981-09-08 | TURNER | 1500.00 |
|    8 |  7654 | 1981-09-28 | MARTIN | 1250.00 |
|    9 |  7839 | 1981-11-17 | KING   | 5000.00 |
|   10 |  7900 | 1981-12-03 | JAMES  |  950.00 |
|   10 |  7902 | 1981-12-03 | FORD   | 3000.00 |
|   11 |  7934 | 1982-01-23 | MILLER | 1300.00 |
|   12 |  7788 | 1987-07-13 | SCOTT  | 3000.00 |
|   12 |  7876 | 1987-07-13 | ADAMS  | 1100.00 |
+------+-------+------------+--------+---------+
14 rows in set (0.010 sec)
```

SELECT empno, hiredate, ename, sal, AVG(sal) OVER (order by rank range between 1 preceding and 1 following) AS avg FROM ranked ;

```
MariaDB [sparksql]> SELECT empno, hiredate, ename, sal, AVG(sal) OVER (order by rank range between 1
preceding and 1 following ) AS avg FROM ranked ;
+-------+------------+--------+---------+-------------+
| empno | hiredate   | ename  | sal     | avg         |
+-------+------------+--------+---------+-------------+
|  7369 | 1980-12-17 | SMITH  |  800.00 | 1200.000000 |
|  7499 | 1981-02-20 | ALLEN  | 1600.00 | 1216.666667 |
|  7521 | 1981-02-22 | WARD   | 1250.00 | 1941.666667 |
|  7566 | 1981-04-02 | JONES  | 2975.00 | 2358.333333 |
|  7698 | 1981-05-01 | BLAKE  | 2850.00 | 2758.333333 |
|  7782 | 1981-06-09 | CLARK  | 2450.00 | 2266.666667 |
|  7844 | 1981-09-08 | TURNER | 1500.00 | 1733.333333 |
|  7654 | 1981-09-28 | MARTIN | 1250.00 | 2583.333333 |
|  7839 | 1981-11-17 | KING   | 5000.00 | 2550.000000 |
|  7900 | 1981-12-03 | JAMES  |  950.00 | 2562.500000 |
|  7902 | 1981-12-03 | FORD   | 3000.00 | 2562.500000 |
|  7934 | 1982-01-23 | MILLER | 1300.00 | 1870.000000 |
|  7788 | 1987-07-13 | SCOTT  | 3000.00 | 1800.000000 |
|  7876 | 1987-07-13 | ADAMS  | 1100.00 | 1800.000000 |
+-------+------------+--------+---------+-------------+
14 rows in set (0.002 sec)
```

206. For each employee, average wages of employees hired before, employee's wages, average wages of employees hired after.

SELECT empno, hiredate, ename, avg(sal) OVER (order by hiredate rows between unbounded preceding and 1 preceding) AS avgBefore, sal, avg(sal) OVER (order by hiredate rows between 1 following and unbounded following) AS avgAfter FROM emp ;

```
MariaDB [sparksql]> SELECT empno, hiredate, ename, avg(sal) OVER (order by hiredate rows between
    -> unbounded preceding and 1 preceding ) AS avgBefore, sal, avg(sal) OVER (order by hiredate rows
 between 1 following and unbounded following ) AS avgAfter FROM emp ;
+-------+------------+--------+-------------+---------+-------------+
| empno | hiredate   | ename  | avgBefore   | sal     | avgAfter    |
+-------+------------+--------+-------------+---------+-------------+
|  7369 | 1980-12-17 | SMITH  |        NULL |  800.00 | 2171.153846 |
|  7499 | 1981-02-20 | ALLEN  |  800.000000 | 1600.00 | 2218.750000 |
|  7521 | 1981-02-22 | WARD   | 1200.000000 | 1250.00 | 2306.818182 |
|  7566 | 1981-04-02 | JONES  | 1216.666667 | 2975.00 | 2240.000000 |
|  7698 | 1981-05-01 | BLAKE  | 1656.250000 | 2850.00 | 2172.222222 |
|  7782 | 1981-06-09 | CLARK  | 1895.000000 | 2450.00 | 2137.500000 |
|  7844 | 1981-09-08 | TURNER | 1987.500000 | 1500.00 | 2228.571429 |
|  7654 | 1981-09-28 | MARTIN | 1917.857143 | 1250.00 | 2391.666667 |
|  7839 | 1981-11-17 | KING   | 1834.375000 | 5000.00 | 1870.000000 |
|  7902 | 1981-12-03 | FORD   | 2186.111111 | 3000.00 | 1587.500000 |
|  7900 | 1981-12-03 | JAMES  | 2267.500000 |  950.00 | 1800.000000 |
|  7934 | 1982-01-23 | MILLER | 2147.727273 | 1300.00 | 2050.000000 |
|  7788 | 1987-07-13 | SCOTT  | 2077.083333 | 3000.00 | 1100.000000 |
|  7876 | 1987-07-13 | ADAMS  | 2148.076923 | 1100.00 |        NULL |
+-------+------------+--------+-------------+---------+-------------+
14 rows in set (0.001 sec)
```