Prepared Statements

With PDO & PHP

Chris Shumake

Created: 4/8/15

Presented: 4/14/15

MYSQL WORLDWIDE CONFERENCE & EXPO

PERCONA LIVE

Who Am I?

- DBA at Bronto Software, Inc
- In the MySQL world for seven years
- PHP shop for same amount of time
- Seen both good-enough db-wrappers and *no* db-wrappers

Who is Bronto?

- Marketing automation for commerce
- 1300+ customers
- 200+ employees
- Headquarters in Durham, NC
- Offices in London, Sydney, NY and LA
- #1 email marketing provider to the Internet Retailer top 1000

INTERNET RETAILER TOP 1000
#1
EMAIL MARKETING PROVIDER

What is Bronto in Database Terms?

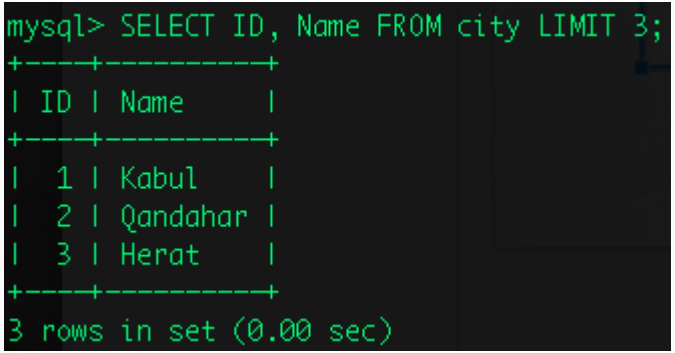
- Highly transactional environment
- 50+ TB of unique data in MySQL
- Percona Server 5.5.32
- Reasonable standards in schema
- Reasonably good-enough dbwrapper
- SQL Injection Free for more than ten years

Outline

- SQL Statements vs Prepared Statements
- Life of a SQL Statement
- Benefits & Drawbacks
- Coding Basics
- Code Considerations
- Operational Considerations

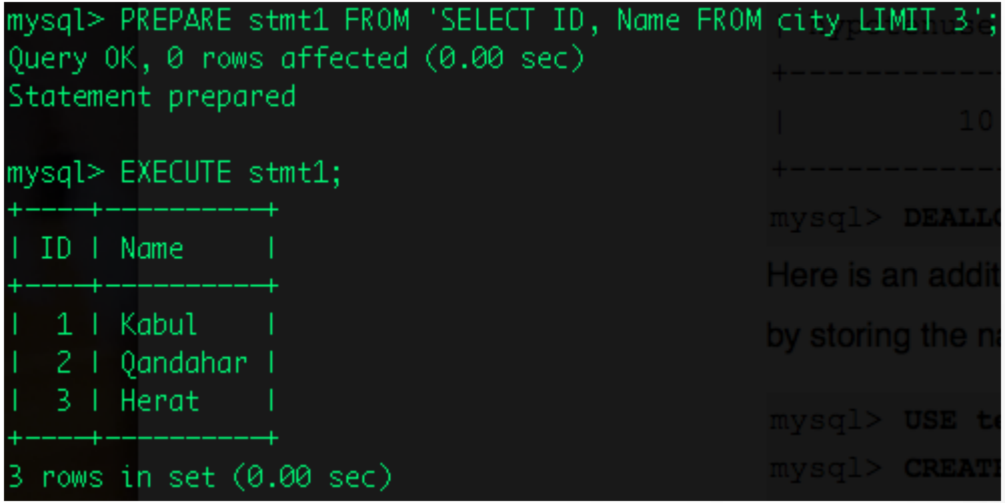
SQL Statements vs Prepared Statements

SQL Statements



```
mysql> SELECT ID, Name FROM city LIMIT 3;
+----+-----------+
| ID | Name      |
+----+-----------+
|  1 | Kabul     |
|  2 | Qandahar  |
|  3 | Herat     |
+----+-----------+
3 rows in set (0.00 sec)
```

Prepared Statements

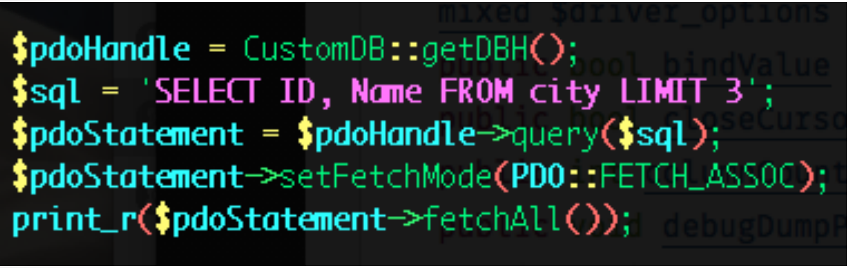


```
mysql> PREPARE stmt1 FROM 'SELECT ID, Name FROM city LIMIT 3';
Query OK, 0 rows affected (0.00 sec)
Statement prepared

mysql> EXECUTE stmt1;
+----+-----------+
| ID | Name      |
+----+-----------+
|  1 | Kabul     |
|  2 | Qandahar  |
|  3 | Herat     |
+----+-----------+
3 rows in set (0.00 sec)
```

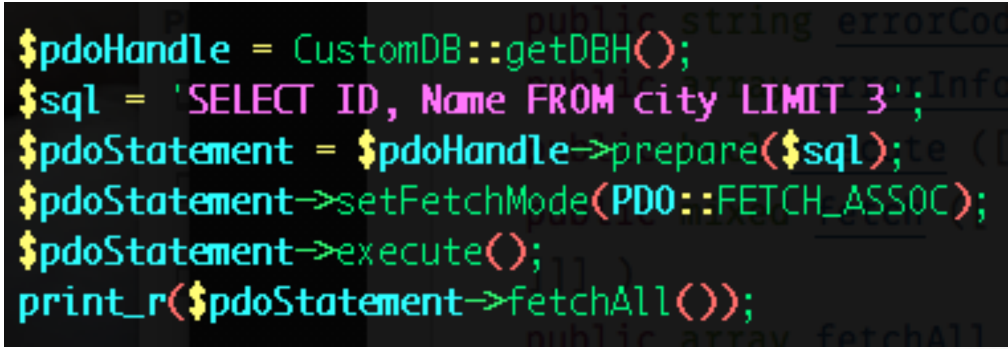
SQL Statements vs Prepared Statements

SQL Statements



```
$pdoHandle = CustomDB::getDBH();
$sql = 'SELECT ID, Name FROM city LIMIT 3';
$pdoStatement = $pdoHandle->query($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
print_r($pdoStatement->fetchAll());
```

Prepared Statements

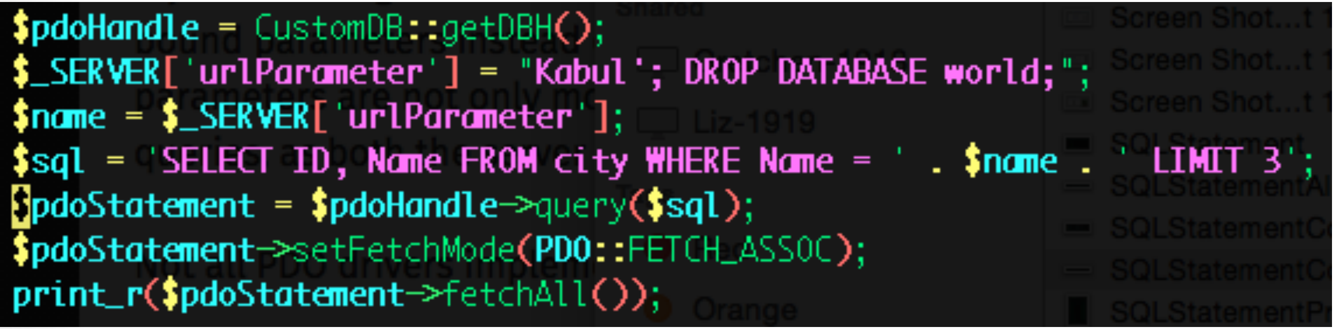


```
$pdoHandle = CustomDB::getDBH();
$sql = 'SELECT ID, Name FROM city LIMIT 3';
$pdoStatement = $pdoHandle->prepare($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
$pdoStatement->execute();
print_r($pdoStatement->fetchAll());
```

Note the different treatment PDO uses: “query” vs “prepare”

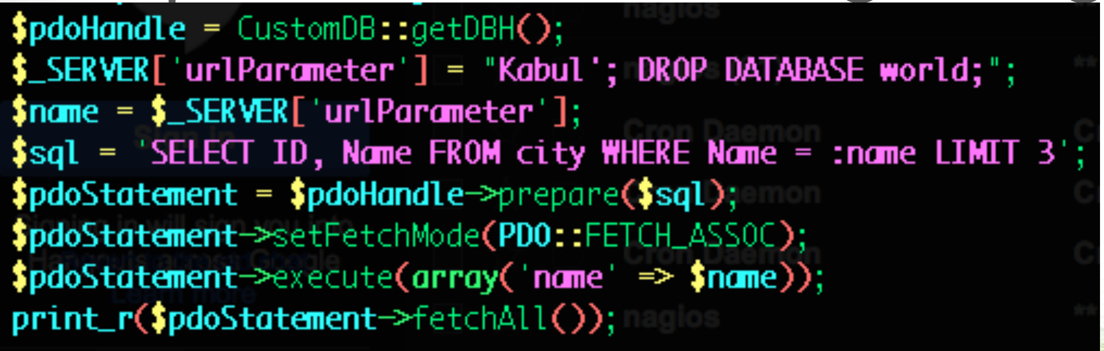
SQL Statements vs Prepared Statements

SQL Statements, with Injection



```
$pdoHandle = CustomDB::getDBH();
$_SERVER['urlParameter'] = "Kabul'; DROP DATABASE world;";
$name = $_SERVER['urlParameter'];
$sql = 'SELECT ID, Name FROM city WHERE Name = ' . $name . ' LIMIT 3';
$pdoStatement = $pdoHandle->query($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
print_r($pdoStatement->fetchAll());
```

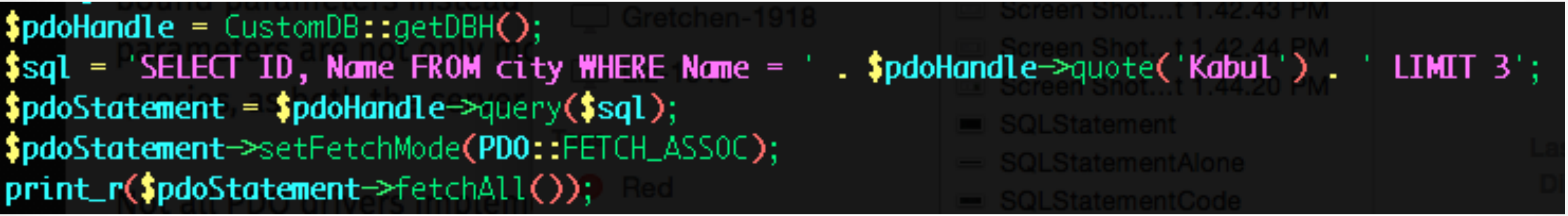
Prepared Statements, ignoring Injection



```
$pdoHandle = CustomDB::getDBH();
$_SERVER['urlParameter'] = "Kabul'; DROP DATABASE world;";
$name = $_SERVER['urlParameter'];
$sql = 'SELECT ID, Name FROM city WHERE Name = :name LIMIT 3';
$pdoStatement = $pdoHandle->prepare($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
$pdoStatement->execute(array('name' => $name));
print_r($pdoStatement->fetchAll());
```

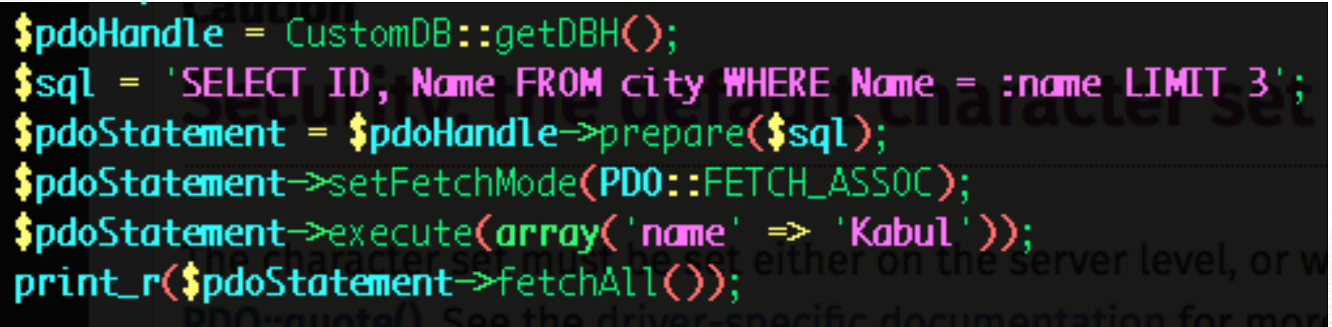
SQL Statements vs Prepared Statements

SQL Statements, with Parameters



```
$pdoHandle = CustomDB::getDBH();
$sql = 'SELECT ID, Name FROM city WHERE Name = ' . $pdoHandle->quote('Kabul') . ' LIMIT 3';
$pdoStatement = $pdoHandle->query($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
print_r($pdoStatement->fetchAll());
```

Prepared Statements, with Parameters



```
$pdoHandle = CustomDB::getDBH();
$sql = 'SELECT ID, Name FROM city WHERE Name = :name LIMIT 3';
$pdoStatement = $pdoHandle->prepare($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
$pdoStatement->execute(array('name' => 'Kabul'));
print_r($pdoStatement->fetchAll());
```

SQL Statements

- Example using the World Database:
  SELECT ID, Name FROM city LIMIT 3;
- Full Solution each statement
- Statement per Resultset

SQL Statements: Full Solution per statement

Every statement hits each step:

- **Parsing**: interpretation of text into syntax
- **Resolution**: matching to columns/tables
- **Optimization**: finding best path to answer
- **Execution**: read data, return resultset

(From Guilhem Bichot - http://bit.ly/1GLtzt4)

SQL Statements: Parsing

Parsing: interpretation of text into syntax

SELECT ID, Name FROM city LIMIT 3;

- Throws errors on syntax problems
- I can’t imagine that it’s expensive compared to the rest of the process, but I have no data on this assumption.

SQL Statements: Resolution

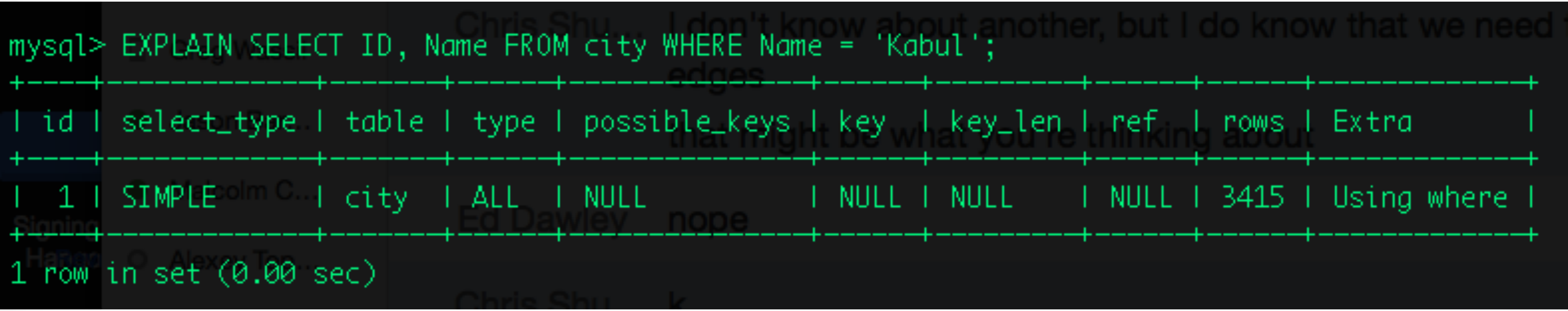
Resolution: matching to columns/tables

```
SELECT ID, Name FROM city LIMIT 3;
```

- Throws errors on resolution failures or ambiguities (ex: two columns w/ same name)
- Can be expensive when subqueries, joins, and table aliasing, etc, combine to generate a larger, complicated query

SQL Statements: Optimization

Optimization: finding best path to answer SELECT ID, Name FROM city WHERE Name = 'Kabul';

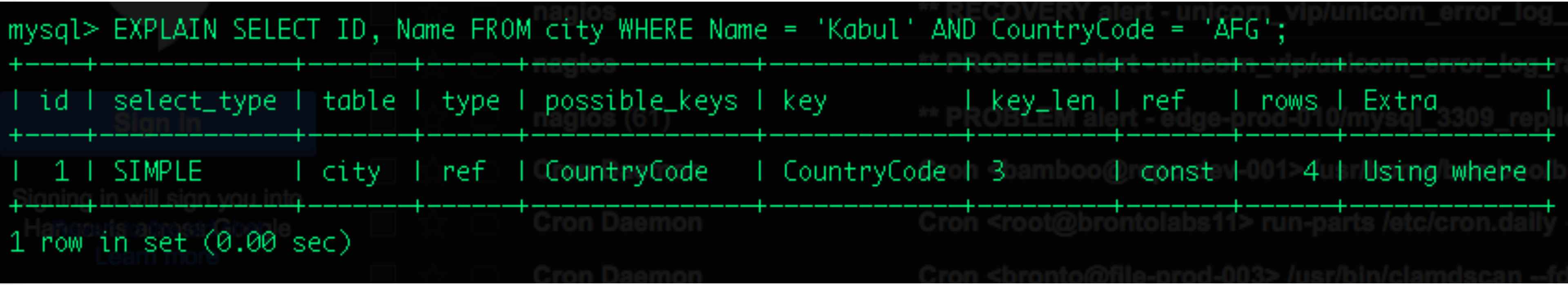


```
mysql> EXPLAIN SELECT ID, Name FROM city WHERE Name = 'Kabul';
+----+-------------+-------+------+---------------+------+---------+------+------+-------------+
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra       |
+----+-------------+-------+------+---------------+------+---------+------+------+-------------+
|  1 | SIMPLE      | city  | ALL  | NULL          | NULL | NULL    | NULL | 3415 | Using where |
+----+-------------+-------+------+---------------+------+---------+------+------+-------------+
1 row in set (0.00 sec)
```

Lack of index, creates full table scan

SQL Statements: Optimization

Optimization: finding best path to answer SELECT ID, Name FROM city WHERE Name = 'Kabul' AND CountryCode = 'AFG';

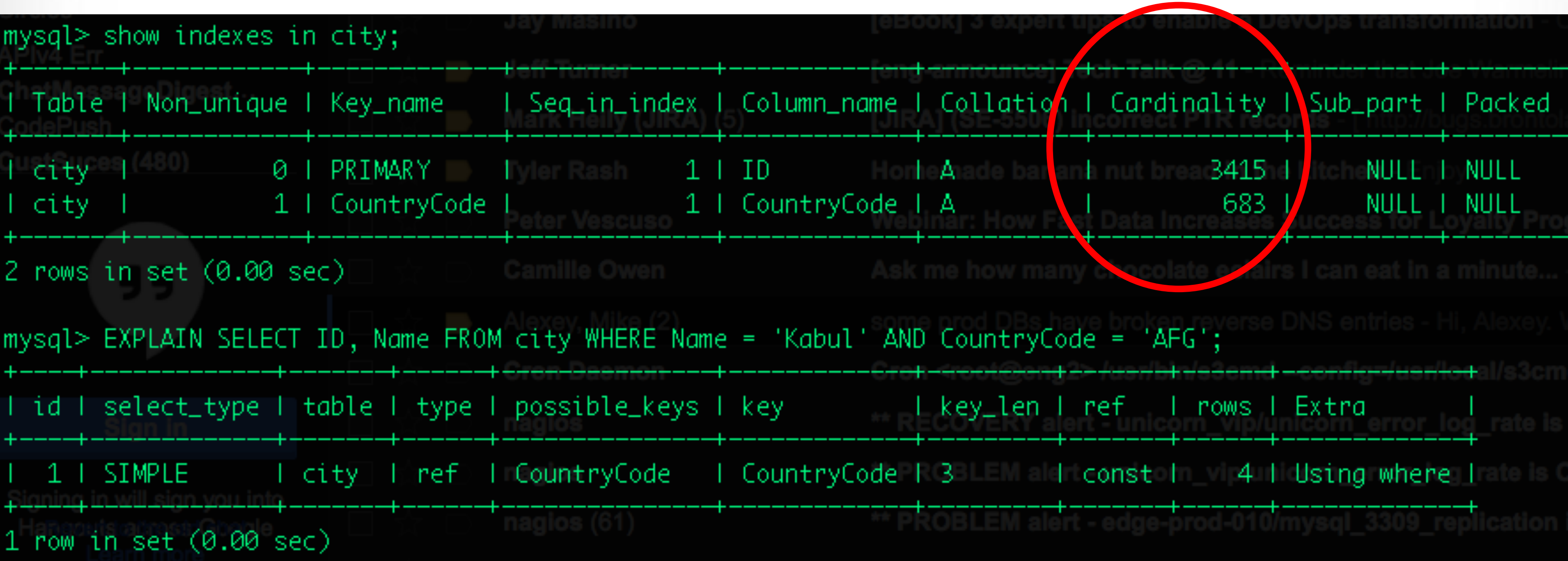


```
mysql> EXPLAIN SELECT ID, Name FROM city WHERE Name = 'Kabul' AND CountryCode = 'AFG';
+----+-------------+-------+------+---------------+-------------+---------+-------+------+-------------+
| id | select_type | table | type | possible_keys | key         | key_len | ref   | rows | Extra       |
+----+-------------+-------+------+---------------+-------------+---------+-------+------+-------------+
|  1 | SIMPLE      | city  | ref  | CountryCode   | CountryCode | 3       | const |    4 | Using where |
+----+-------------+-------+------+---------------+-------------+---------+-------+------+-------------+
1 row in set (0.00 sec)
```

With index, rows is based on Cardinality

SQL Statements: Optimization

Determine least work based on Cardinality

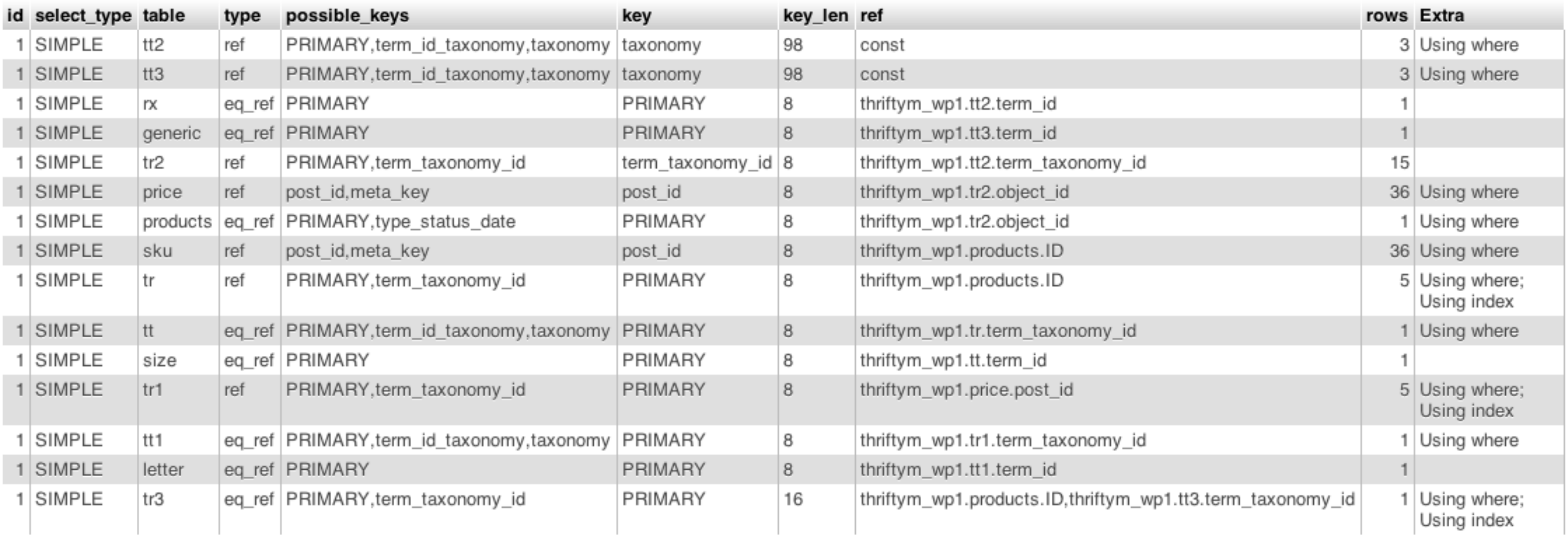


```
mysql> show indexes in city;
+-------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------
| Table | Non_unique | Key_name    | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed
+-------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------
| city  |          0 | PRIMARY     |            1 | ID          | A         |        3415 |     NULL | NULL
| city  |          1 | CountryCode |            1 | CountryCode | A         |         683 |     NULL | NULL
+-------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------
2 rows in set (0.00 sec)

mysql> EXPLAIN SELECT ID, Name FROM city WHERE Name = 'Kabul' AND CountryCode = 'AFG';
+----+-------------+-------+------+---------------+-------------+---------+-------+------+-------------+
| id | select_type | table | type | possible_keys | key         | key_len | ref   | rows | Extra       |
+----+-------------+-------+------+---------------+-------------+---------+-------+------+-------------+
|  1 | SIMPLE      | city  | ref  | CountryCode   | CountryCode | 3       | const |    4 | Using where |
+----+-------------+-------+------+---------------+-------------+---------+-------+------+-------------+
1 row in set (0.00 sec)
```

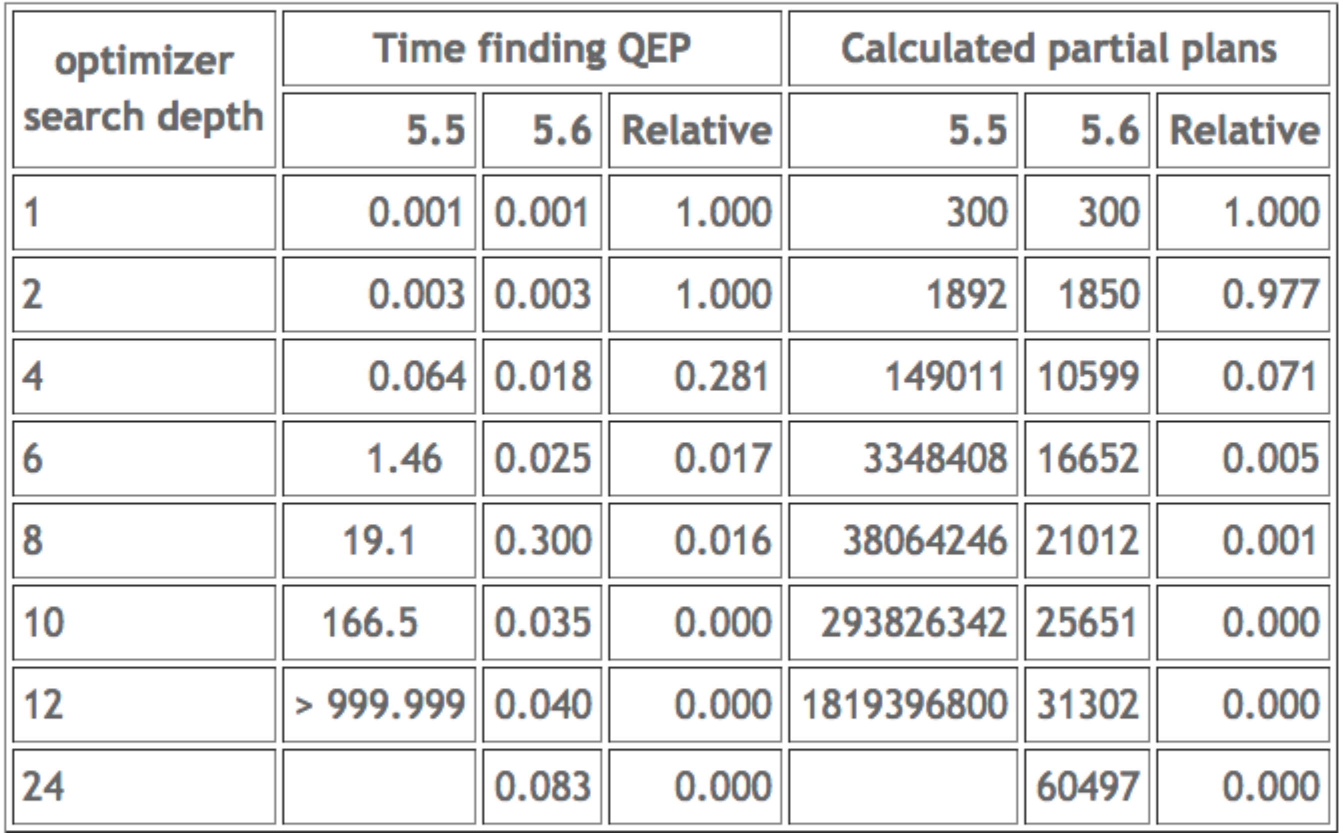
SQL Statements: Optimization

It can get complicated and expensive



id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	tt2	ref	PRIMARY,term_id_taxonomy,taxonomy	taxonomy	98	const	3	Using where
1	SIMPLE	tt3	ref	PRIMARY,term_id_taxonomy,taxonomy	taxonomy	98	const	3	Using where
1	SIMPLE	rx	eq_ref	PRIMARY	PRIMARY	8	thriftym_wp1.tt2.term_id	1	
1	SIMPLE	generic	eq_ref	PRIMARY	PRIMARY	8	thriftym_wp1.tt3.term_id	1	
1	SIMPLE	tr2	ref	PRIMARY,term_taxonomy_id	term_taxonomy_id	8	thriftym_wp1.tt2.term_taxonomy_id	15	
1	SIMPLE	price	ref	post_id,meta_key	post_id	8	thriftym_wp1.tr2.object_id	36	Using where
1	SIMPLE	products	eq_ref	PRIMARY,type_status_date	PRIMARY	8	thriftym_wp1.tr2.object_id	1	Using where
1	SIMPLE	sku	ref	post_id,meta_key	post_id	8	thriftym_wp1.products.ID	36	Using where
1	SIMPLE	tr	ref	PRIMARY,term_taxonomy_id	PRIMARY	8	thriftym_wp1.products.ID	5	Using where; Using index
1	SIMPLE	tt	eq_ref	PRIMARY,term_id_taxonomy,taxonomy	PRIMARY	8	thriftym_wp1.tr.term_taxonomy_id	1	Using where
1	SIMPLE	size	eq_ref	PRIMARY	PRIMARY	8	thriftym_wp1.tt.term_id	1	
1	SIMPLE	tr1	ref	PRIMARY,term_taxonomy_id	PRIMARY	8	thriftym_wp1.price.post_id	5	Using where; Using index
1	SIMPLE	tt1	eq_ref	PRIMARY,term_id_taxonomy,taxonomy	PRIMARY	8	thriftym_wp1.tr1.term_taxonomy_id	1	Using where
1	SIMPLE	letter	eq_ref	PRIMARY	PRIMARY	8	thriftym_wp1.tt1.term_id	1	
1	SIMPLE	tr3	eq_ref	PRIMARY,term_taxonomy_id	PRIMARY	16	thriftym_wp1.products.ID,thriftym_wp1.tt3.term_taxonomy_id	1	Using where; Using index

SQL Statements: Optimization



optimizer search depth	Time finding QEP			Calculated partial plans		
	5.5	5.6	Relative	5.5	5.6	Relative
1	0.001	0.001	1.000	300	300	1.000
2	0.003	0.003	1.000	1892	1850	0.977
4	0.064	0.018	0.281	149011	10599	0.071
6	1.46	0.025	0.017	3348408	16652	0.005
8	19.1	0.300	0.016	38064246	21012	0.001
10	166.5	0.035	0.000	293826342	25651	0.000
12	> 999.999	0.040	0.000	1819396800	31302	0.000
24		0.083	0.000		60497	0.000

Seriously complicated and expensive

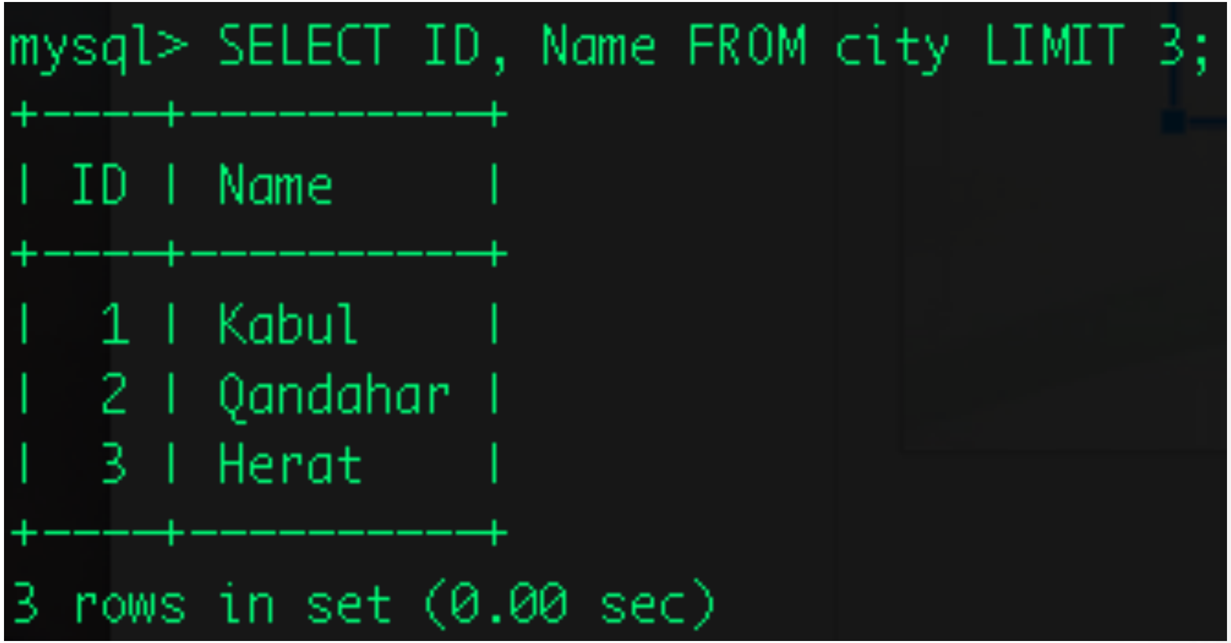
(image source: Jorgen Loland)

SQL Statements: Execution

Always happens

“executing”

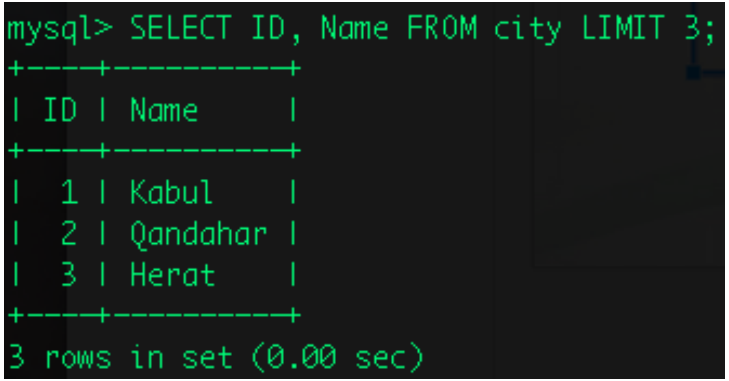
“Sending data”



```
mysql> SELECT ID, Name FROM city LIMIT 3;
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```

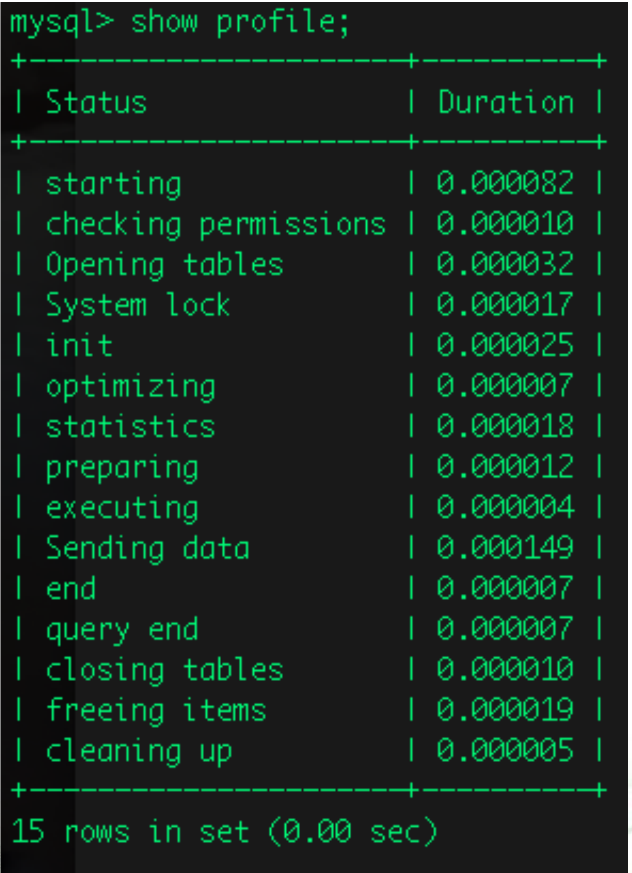
SQL Statements: Example using World

- SQL Statement:

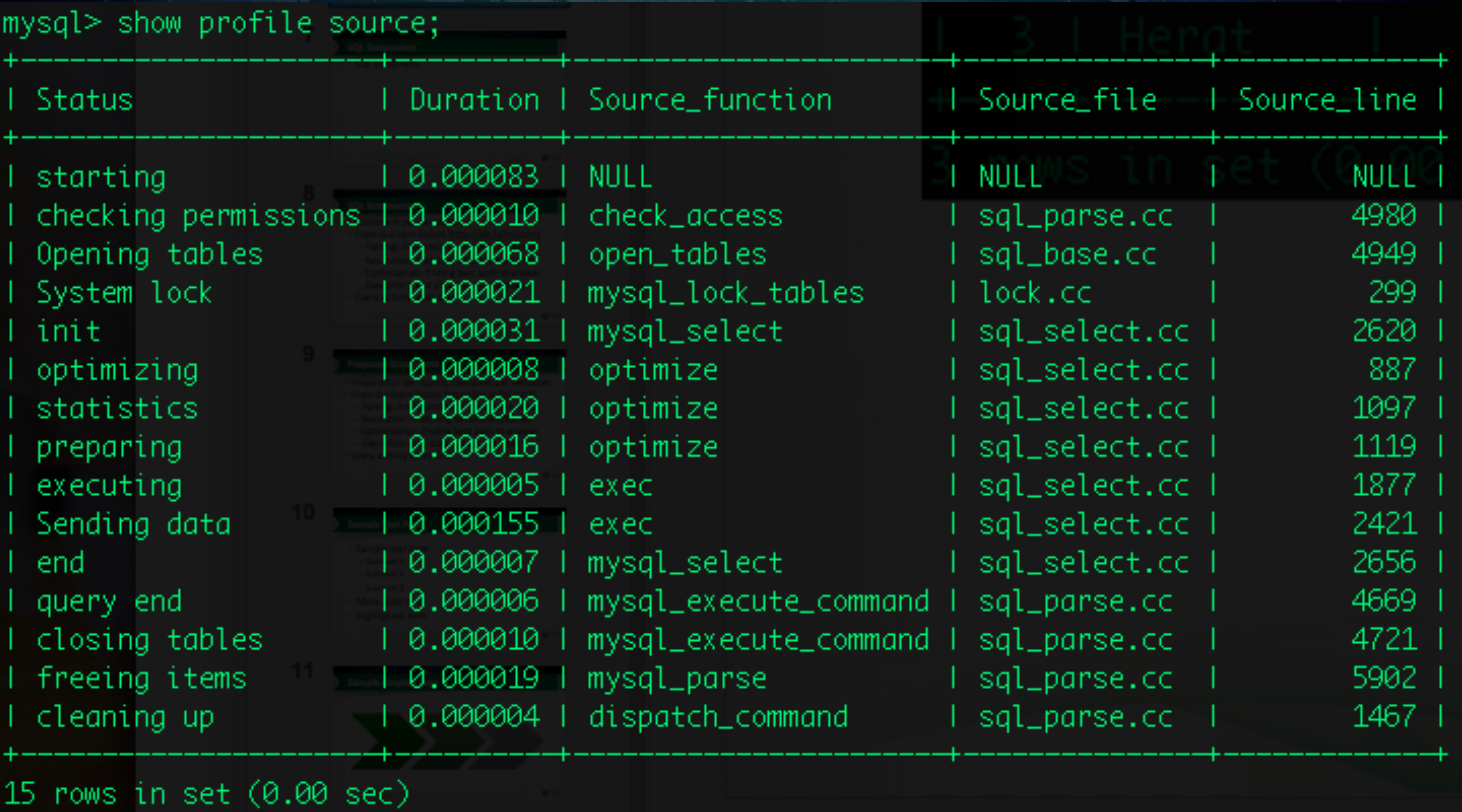


```
mysql> SELECT ID, Name FROM city LIMIT 3;
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```

Parsing, Resolution, Optimizing, Execution



```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000082 |
| checking permissions | 0.000010 |
| Opening tables       | 0.000032 |
| System lock          | 0.000017 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000018 |
| preparing            | 0.000012 |
| executing            | 0.000004 |
| Sending data         | 0.000149 |
| end                  | 0.000007 |
| query end            | 0.000007 |
| closing tables       | 0.000010 |
| freeing items        | 0.000019 |
| cleaning up          | 0.000005 |
+----------------------+----------+
15 rows in set (0.00 sec)
```



```
mysql> show profile source;
+----------------------+----------+-----------------------+---------------+-------------+
| Status               | Duration | Source_function       | Source_file   | Source_line |
+----------------------+----------+-----------------------+---------------+-------------+
| starting             | 0.000083 | NULL                  | NULL          |        NULL |
| checking permissions | 0.000010 | check_access          | sql_parse.cc  |        4980 |
| Opening tables       | 0.000068 | open_tables           | sql_base.cc   |        4949 |
| System lock          | 0.000021 | mysql_lock_tables     | lock.cc       |         299 |
| init                 | 0.000031 | mysql_select          | sql_select.cc |        2620 |
| optimizing           | 0.000008 | optimize              | sql_select.cc |         887 |
| statistics           | 0.000020 | optimize              | sql_select.cc |        1097 |
| preparing            | 0.000016 | optimize              | sql_select.cc |        1119 |
| executing            | 0.000005 | exec                  | sql_select.cc |        1877 |
| Sending data         | 0.000155 | exec                  | sql_select.cc |        2421 |
| end                  | 0.000007 | mysql_select          | sql_select.cc |        2656 |
| query end            | 0.000006 | mysql_execute_command | sql_parse.cc  |        4669 |
| closing tables       | 0.000010 | mysql_execute_command | sql_parse.cc  |        4721 |
| freeing items        | 0.000019 | mysql_parse           | sql_parse.cc  |        5902 |
| cleaning up          | 0.000004 | dispatch_command      | sql_parse.cc  |        1467 |
+----------------------+----------+-----------------------+---------------+-------------+
15 rows in set (0.00 sec)
```

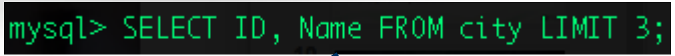
SQL Statements: Full Solution per statement

Every statement hits each step:

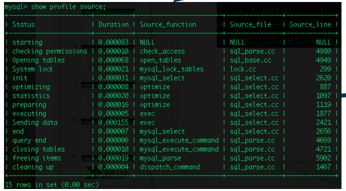
- **Parsing**: interpretation of text into syntax
- **Resolution**: matching to columns/tables
- **Optimization**: finding best path to answer
- **Execution**: read data, return resultset

(From Guilhem Bichot - http://bit.ly/1GLtzt4)

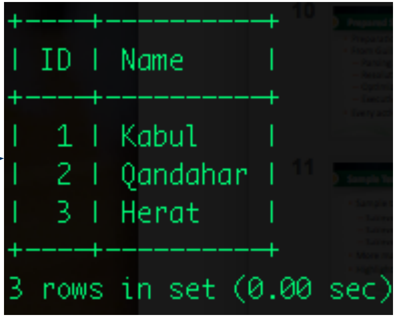
SQL Statements: Statement per Resultset



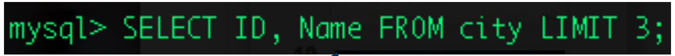
```
mysql> SELECT ID, Name FROM city LIMIT 3;
```



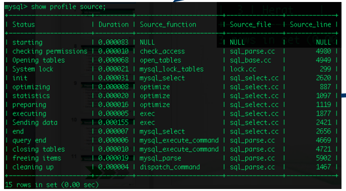
```
mysql> show profile source;
+----------------------+----------+-----------------------+---------------+-------------+
| Status               | Duration | Source_function       | Source_file   | Source_line |
+----------------------+----------+-----------------------+---------------+-------------+
| starting             | 0.000083 | NULL                  | NULL          |        NULL |
| checking permissions | 0.000010 | check_access          | sql_parse.cc  |        4980 |
| Opening tables       | 0.000068 | open_tables           | sql_base.cc   |        4949 |
| System lock          | 0.000021 | mysql_lock_tables     | lock.cc       |         299 |
| init                 | 0.000031 | mysql_select          | sql_select.cc |        2620 |
| optimizing           | 0.000008 | optimize              | sql_select.cc |         887 |
| statistics           | 0.000020 | optimize              | sql_select.cc |        1097 |
| preparing            | 0.000016 | optimize              | sql_select.cc |        1119 |
| executing            | 0.000005 | exec                  | sql_select.cc |        1877 |
| Sending data         | 0.000155 | exec                  | sql_select.cc |        2421 |
| end                  | 0.000007 | mysql_select          | sql_select.cc |        2656 |
| query end            | 0.000006 | mysql_execute_command | sql_parse.cc  |        4669 |
| closing tables       | 0.000010 | mysql_execute_command | sql_parse.cc  |        4721 |
| freeing items        | 0.000019 | mysql_parse           | sql_parse.cc  |        5902 |
| cleaning up          | 0.000004 | dispatch_command      | sql_parse.cc  |        1467 |
+----------------------+----------+-----------------------+---------------+-------------+
15 rows in set (0.00 sec)
```



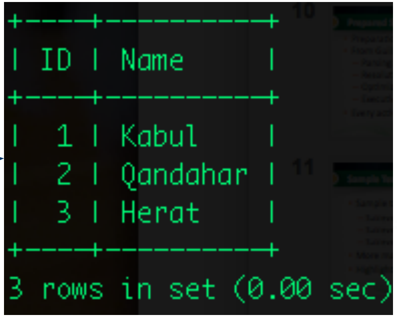
```
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```



```
mysql> SELECT ID, Name FROM city LIMIT 3;
```

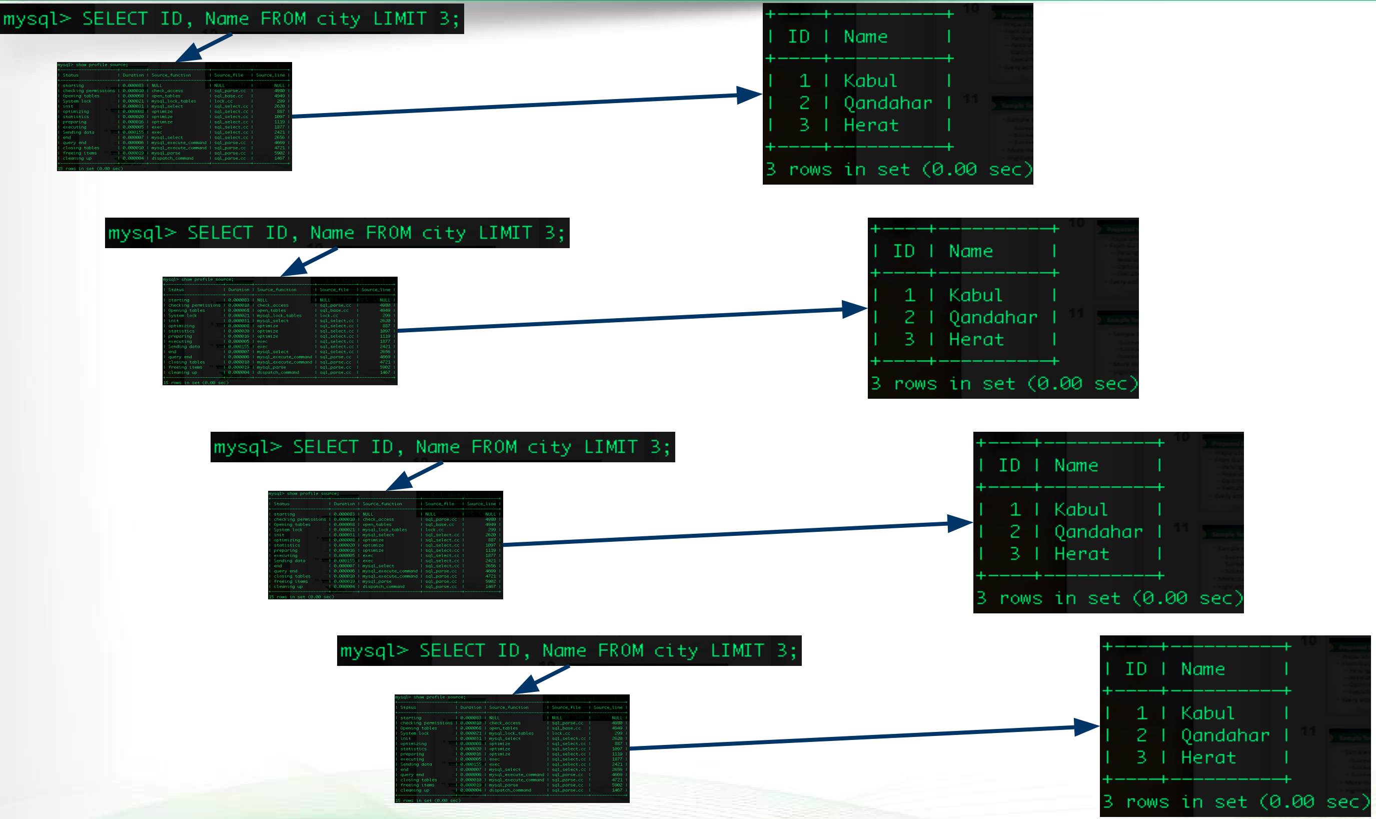


```
mysql> show profile source;
+----------------------+----------+-----------------------+---------------+-------------+
| Status               | Duration | Source_function       | Source_file   | Source_line |
+----------------------+----------+-----------------------+---------------+-------------+
| starting             | 0.000083 | NULL                  | NULL          |        NULL |
| checking permissions | 0.000010 | check_access          | sql_parse.cc  |        4980 |
| Opening tables       | 0.000068 | open_tables           | sql_base.cc   |        4949 |
| System lock          | 0.000021 | mysql_lock_tables     | lock.cc       |         299 |
| init                 | 0.000031 | mysql_select          | sql_select.cc |        2620 |
| optimizing           | 0.000008 | optimize              | sql_select.cc |         887 |
| statistics           | 0.000020 | optimize              | sql_select.cc |        1097 |
| preparing            | 0.000016 | optimize              | sql_select.cc |        1119 |
| executing            | 0.000005 | exec                  | sql_select.cc |        1877 |
| Sending data         | 0.000155 | exec                  | sql_select.cc |        2421 |
| end                  | 0.000007 | mysql_select          | sql_select.cc |        2656 |
| query end            | 0.000006 | mysql_execute_command | sql_parse.cc  |        4669 |
| closing tables       | 0.000010 | mysql_execute_command | sql_parse.cc  |        4721 |
| freeing items        | 0.000019 | mysql_parse           | sql_parse.cc  |        5902 |
| cleaning up          | 0.000004 | dispatch_command      | sql_parse.cc  |        1467 |
+----------------------+----------+-----------------------+---------------+-------------+
15 rows in set (0.00 sec)
```



```
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```

Full Solution to keep going



```
mysql> SELECT ID, Name FROM city LIMIT 3;
```

```
mysql> show profile source;
+----------------------+----------+-----------------------+---------------+-------------+
| Status               | Duration | Source_function       | Source_file   | Source_line |
+----------------------+----------+-----------------------+---------------+-------------+
| starting             | 0.000083 | NULL                  | NULL          |        NULL |
| checking permissions | 0.000010 | check_access          | sql_parse.cc  |        4980 |
| Opening tables       | 0.000068 | open_tables           | sql_base.cc   |        4949 |
| System lock          | 0.000021 | mysql_lock_tables     | lock.cc       |         299 |
| init                 | 0.000031 | mysql_select          | sql_select.cc |        2620 |
| optimizing           | 0.000008 | optimize              | sql_select.cc |         887 |
| statistics           | 0.000020 | optimize              | sql_select.cc |        1097 |
| preparing            | 0.000016 | optimize              | sql_select.cc |        1119 |
| executing            | 0.000005 | exec                  | sql_select.cc |        1877 |
| Sending data         | 0.000155 | exec                  | sql_select.cc |        2421 |
| end                  | 0.000007 | mysql_select          | sql_select.cc |        2656 |
| query end            | 0.000006 | mysql_execute_command | sql_parse.cc  |        4669 |
| closing tables       | 0.000010 | mysql_execute_command | sql_parse.cc  |        4721 |
| freeing items        | 0.000019 | mysql_parse           | sql_parse.cc  |        5902 |
| cleaning up          | 0.000004 | dispatch_command      | sql_parse.cc  |        1467 |
+----------------------+----------+-----------------------+---------------+-------------+
15 rows in set (0.00 sec)
```

```
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```

mysql> SELECT ID, Name FROM city LIMIT 3;

mysql> show profile source; (15 rows in set (0.00 sec))

| ID | Name | — 1 Kabul, 2 Qandahar, 3 Herat — 3 rows in set (0.00 sec)

mysql> SELECT ID, Name FROM city LIMIT 3;

mysql> show profile source; (15 rows in set (0.00 sec))

| ID | Name | — 1 Kabul, 2 Qandahar, 3 Herat — 3 rows in set (0.00 sec)

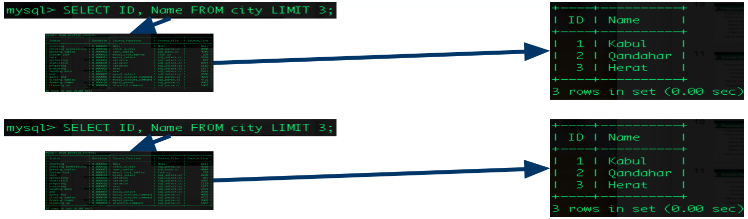
mysql> SELECT ID, Name FROM city LIMIT 3;

mysql> show profile source; (15 rows in set (0.00 sec))

| ID | Name | — 1 Kabul, 2 Qandahar, 3 Herat — 3 rows in set (0.00 sec)

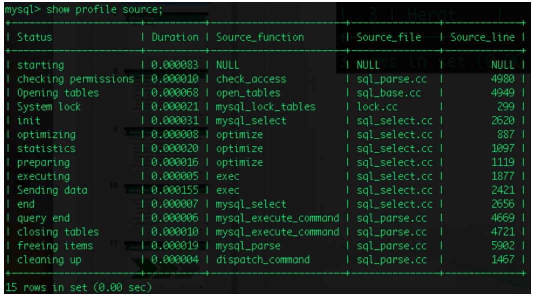
SQL Statements

- Statement per Resultset



```
mysql> SELECT ID, Name FROM city LIMIT 3;
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)

mysql> SELECT ID, Name FROM city LIMIT 3;
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```



```
mysql> show profile source;
+----------------------+----------+-----------------------+---------------+-------------+
| Status               | Duration | Source_function       | Source_file   | Source_line |
+----------------------+----------+-----------------------+---------------+-------------+
| starting             | 0.000083 | NULL                  | NULL          |        NULL |
| checking permissions | 0.000010 | check_access          | sql_parse.cc  |        4980 |
| Opening tables       | 0.000068 | open_tables           | sql_base.cc   |        4949 |
| System lock          | 0.000021 | mysql_lock_tables     | lock.cc       |         299 |
| init                 | 0.000031 | mysql_select          | sql_select.cc |        2620 |
| optimizing           | 0.000008 | optimize              | sql_select.cc |         887 |
| statistics           | 0.000020 | optimize              | sql_select.cc |        1097 |
| preparing            | 0.000016 | optimize              | sql_select.cc |        1119 |
| executing            | 0.000005 | exec                  | sql_select.cc |        1877 |
| Sending data         | 0.000155 | exec                  | sql_select.cc |        2421 |
| end                  | 0.000007 | mysql_select          | sql_select.cc |        2656 |
| query end            | 0.000006 | mysql_execute_command | sql_parse.cc  |        4669 |
| closing tables       | 0.000010 | mysql_execute_command | sql_parse.cc  |        4721 |
| freeing items        | 0.000019 | mysql_parse           | sql_parse.cc  |        5902 |
| cleaning up          | 0.000004 | dispatch_command      | sql_parse.cc  |        1467 |
+----------------------+----------+-----------------------+---------------+-------------+
15 rows in set (0.00 sec)
```

- Full Solution per statement

- Every activity costs CPU

Prepared Statements

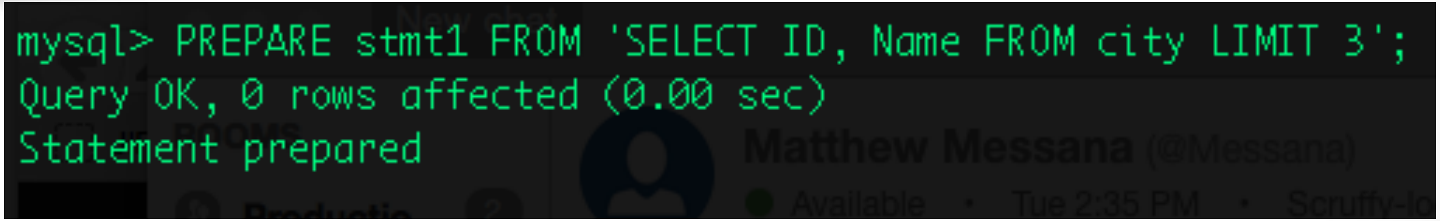
- Preparation per Query, Execution per Resultset
- From Guilhem Bichot (http://bit.ly/1GLtzt4)
  - **Parsing**: interpretation of text into syntax
  - **Resolution**: matching to columns/tables
  - **Optimization**: finding best path to answer
  - **Execution**: read data, return resultset
- Every activity costs CPU

Prepared Statements

- Example using the World Database:
  SELECT ID, Name FROM city LIMIT 3;
- Partial Solution at Preparation
  - Parsing, Resolution, some Optimization
- Partial Solution at Execution
  - Remaining Optimization, Execution
- Execution per Resultset

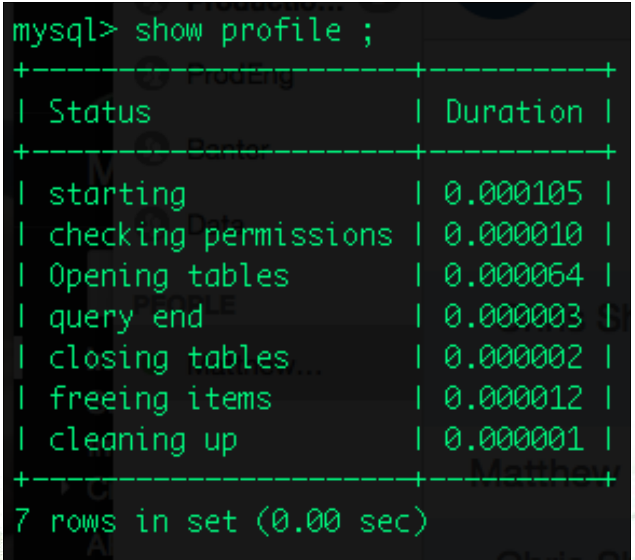
Prep Stmts: Partial Solution at Preparation

- Prep Stmt:



```
mysql> PREPARE stmt1 FROM 'SELECT ID, Name FROM city LIMIT 3';
Query OK, 0 rows affected (0.00 sec)
Statement prepared
```

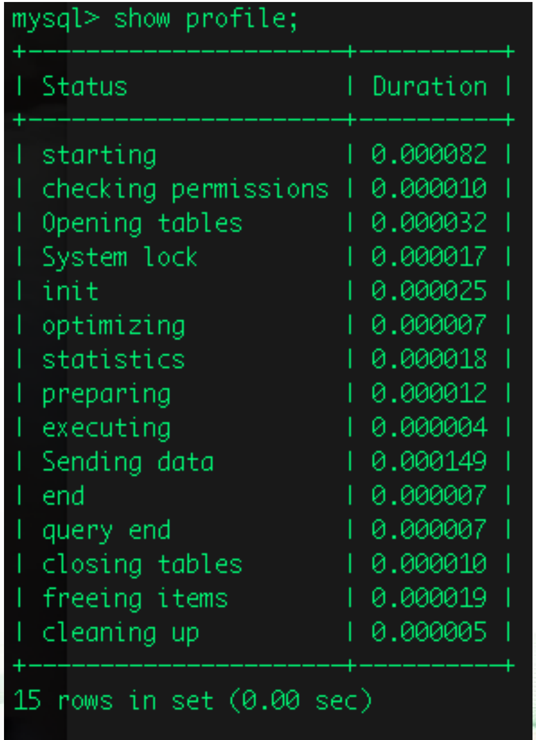
Parsing, Resolution, Some Optimizing



```
mysql> show profile ;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000105 |
| checking permissions | 0.000010 |
| Opening tables       | 0.000064 |
| query end            | 0.000003 |
| closing tables       | 0.000002 |
| freeing items        | 0.000012 |
| cleaning up          | 0.000001 |
+----------------------+----------+
7 rows in set (0.00 sec)
```

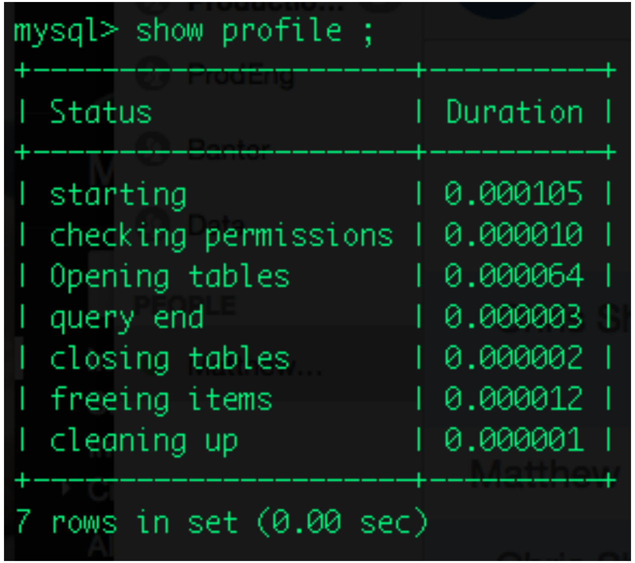
Remember the SQL Statement Profile?

SQL Statement



```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000082 |
| checking permissions | 0.000010 |
| Opening tables       | 0.000032 |
| System lock          | 0.000017 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000018 |
| preparing            | 0.000012 |
| executing            | 0.000004 |
| Sending data         | 0.000149 |
| end                  | 0.000007 |
| query end            | 0.000007 |
| closing tables       | 0.000010 |
| freeing items        | 0.000019 |
| cleaning up          | 0.000005 |
+----------------------+----------+
15 rows in set (0.00 sec)
```

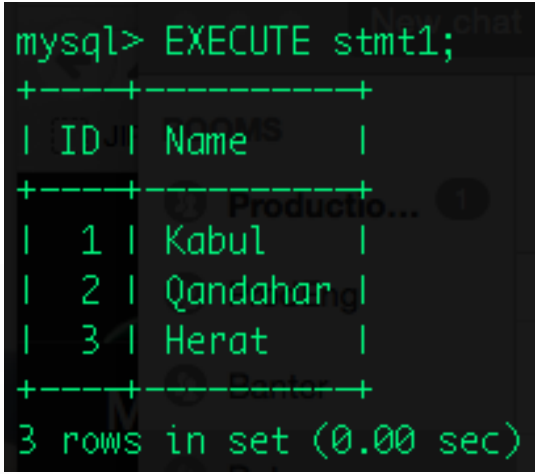
Prepared Statement



```
mysql> show profile ;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000105 |
| checking permissions | 0.000010 |
| Opening tables       | 0.000064 |
| query end            | 0.000003 |
| closing tables       | 0.000002 |
| freeing items        | 0.000012 |
| cleaning up          | 0.000001 |
+----------------------+----------+
7 rows in set (0.00 sec)
```

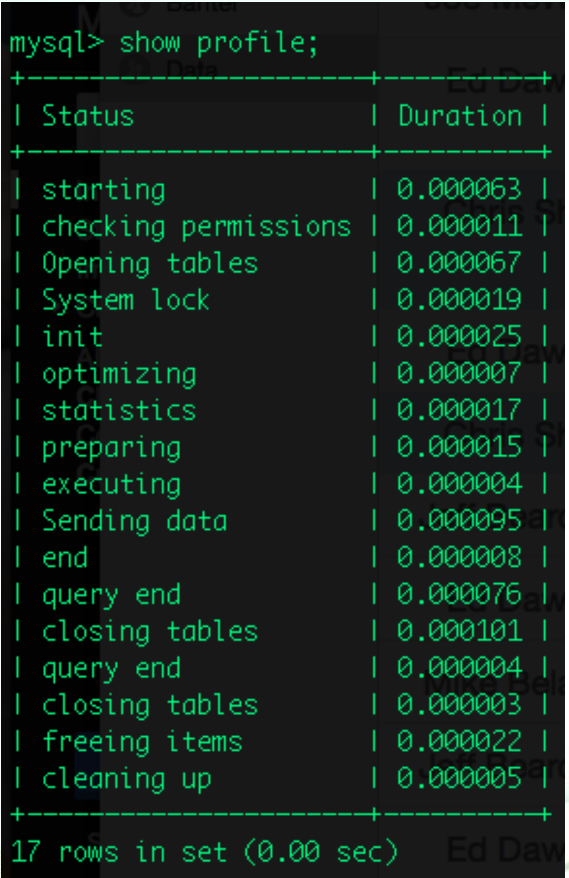
Prep Stmts: Remainder Solution at Execution

- Prep Stmt:



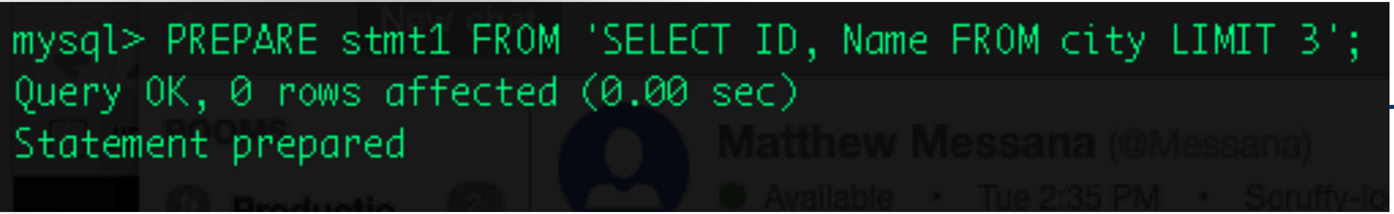
```
mysql> EXECUTE stmt1;
+----+-----------+
| ID | Name      |
+----+-----------+
|  1 | Kabul     |
|  2 | Qandahar  |
|  3 | Herat     |
+----+-----------+
3 rows in set (0.00 sec)
```

Remaining Optimization, Execution

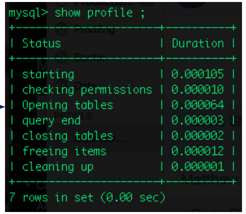


```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000063 |
| checking permissions | 0.000011 |
| Opening tables       | 0.000067 |
| System lock          | 0.000019 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000017 |
| preparing            | 0.000015 |
| executing            | 0.000004 |
| Sending data         | 0.000095 |
| end                  | 0.000008 |
| query end            | 0.000076 |
| closing tables       | 0.000101 |
| query end            | 0.000004 |
| closing tables       | 0.000003 |
| freeing items        | 0.000022 |
| cleaning up          | 0.000005 |
+----------------------+----------+
17 rows in set (0.00 sec)
```

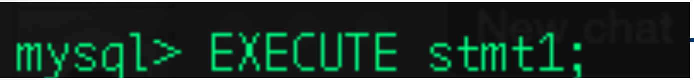
Prep Stmts: Prep per Stmt & Exec per Resultset



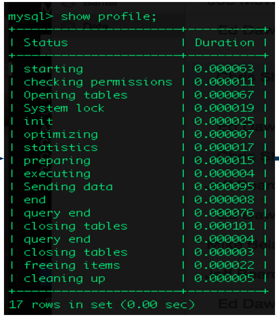
```
mysql> PREPARE stmt1 FROM 'SELECT ID, Name FROM city LIMIT 3';
Query OK, 0 rows affected (0.00 sec)
Statement prepared
```



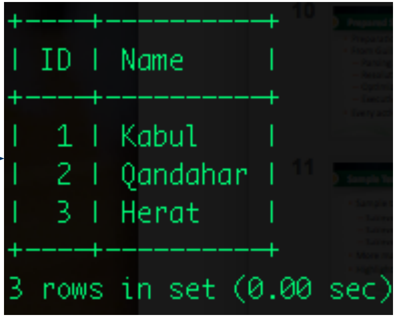
```
mysql> show profile ;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000105 |
| checking permissions | 0.000010 |
| Opening tables       | 0.000064 |
| query end            | 0.000003 |
| closing tables       | 0.000002 |
| freeing items        | 0.000012 |
| cleaning up          | 0.000001 |
+----------------------+----------+
7 rows in set (0.00 sec)
```



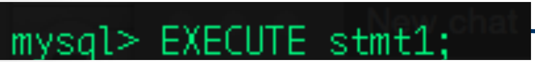
```
mysql> EXECUTE stmt1;
```



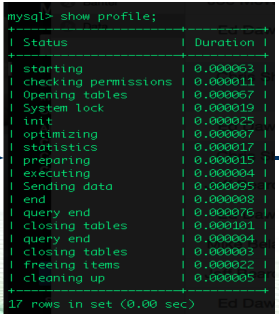
```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000063 |
| checking permissions | 0.000011 |
| Opening tables       | 0.000067 |
| System lock          | 0.000019 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000017 |
| preparing            | 0.000015 |
| executing            | 0.000004 |
| Sending data         | 0.000095 |
| end                  | 0.000008 |
| query end            | 0.000076 |
| closing tables       | 0.000101 |
| query end            | 0.000004 |
| closing tables       | 0.000003 |
| freeing items        | 0.000022 |
| cleaning up          | 0.000005 |
+----------------------+----------+
17 rows in set (0.00 sec)
```



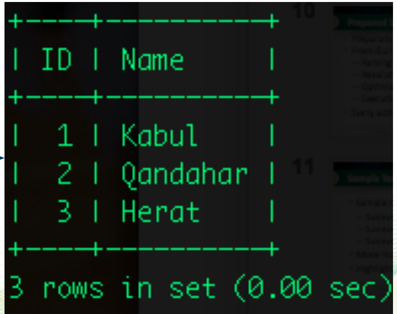
```
+----+-----------+
| ID | Name      |
+----+-----------+
|  1 | Kabul     |
|  2 | Qandahar  |
|  3 | Herat     |
+----+-----------+
3 rows in set (0.00 sec)
```



```
mysql> EXECUTE stmt1;
```

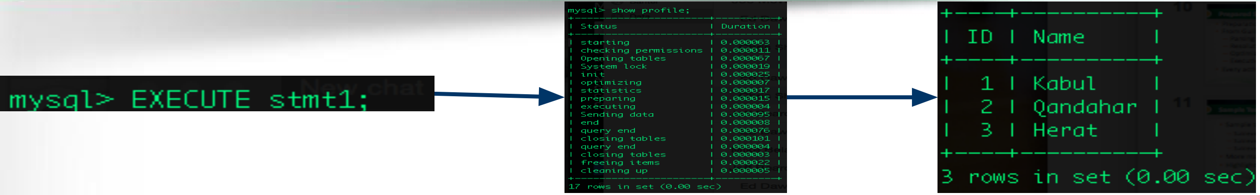


```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000063 |
| checking permissions | 0.000011 |
| Opening tables       | 0.000067 |
| System lock          | 0.000019 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000017 |
| preparing            | 0.000015 |
| executing            | 0.000004 |
| Sending data         | 0.000095 |
| end                  | 0.000008 |
| query end            | 0.000076 |
| closing tables       | 0.000101 |
| query end            | 0.000004 |
| closing tables       | 0.000003 |
| freeing items        | 0.000022 |
| cleaning up          | 0.000005 |
+----------------------+----------+
17 rows in set (0.00 sec)
```



```
+----+-----------+
| ID | Name      |
+----+-----------+
|  1 | Kabul     |
|  2 | Qandahar  |
|  3 | Herat     |
+----+-----------+
3 rows in set (0.00 sec)
```

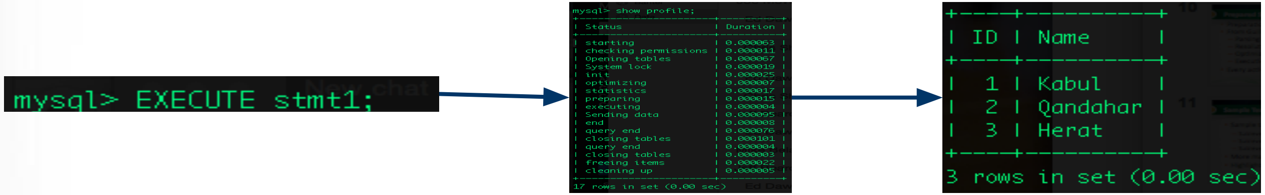
Remainder Solution just keeps going



```
mysql> EXECUTE stmt1;
```

```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000063 |
| checking permissions | 0.000011 |
| Opening tables       | 0.000067 |
| System lock          | 0.000019 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000017 |
| preparing            | 0.000015 |
| executing            | 0.000004 |
| Sending data         | 0.000095 |
| end                  | 0.000008 |
| query end            | 0.000076 |
| closing tables       | 0.000101 |
| query end            | 0.000004 |
| closing tables       | 0.000003 |
| freeing items        | 0.000022 |
| cleaning up          | 0.000005 |
+----------------------+----------+
17 rows in set (0.00 sec)
```

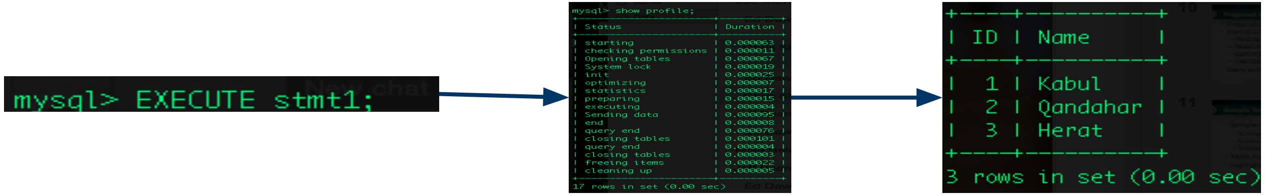
```
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```



```
mysql> EXECUTE stmt1;
```

```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000063 |
| checking permissions | 0.000011 |
| Opening tables       | 0.000067 |
| System lock          | 0.000019 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000017 |
| preparing            | 0.000015 |
| executing            | 0.000004 |
| Sending data         | 0.000095 |
| end                  | 0.000008 |
| query end            | 0.000076 |
| closing tables       | 0.000101 |
| query end            | 0.000004 |
| closing tables       | 0.000003 |
| freeing items        | 0.000022 |
| cleaning up          | 0.000005 |
+----------------------+----------+
17 rows in set (0.00 sec)
```

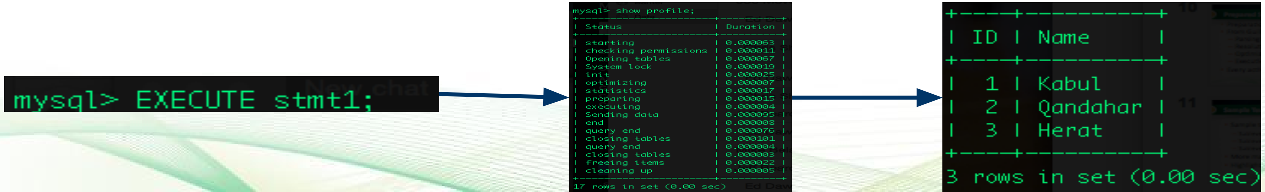
```
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```



```
mysql> EXECUTE stmt1;
```

```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000063 |
| checking permissions | 0.000011 |
| Opening tables       | 0.000067 |
| System lock          | 0.000019 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000017 |
| preparing            | 0.000015 |
| executing            | 0.000004 |
| Sending data         | 0.000095 |
| end                  | 0.000008 |
| query end            | 0.000076 |
| closing tables       | 0.000101 |
| query end            | 0.000004 |
| closing tables       | 0.000003 |
| freeing items        | 0.000022 |
| cleaning up          | 0.000005 |
+----------------------+----------+
17 rows in set (0.00 sec)
```

```
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```

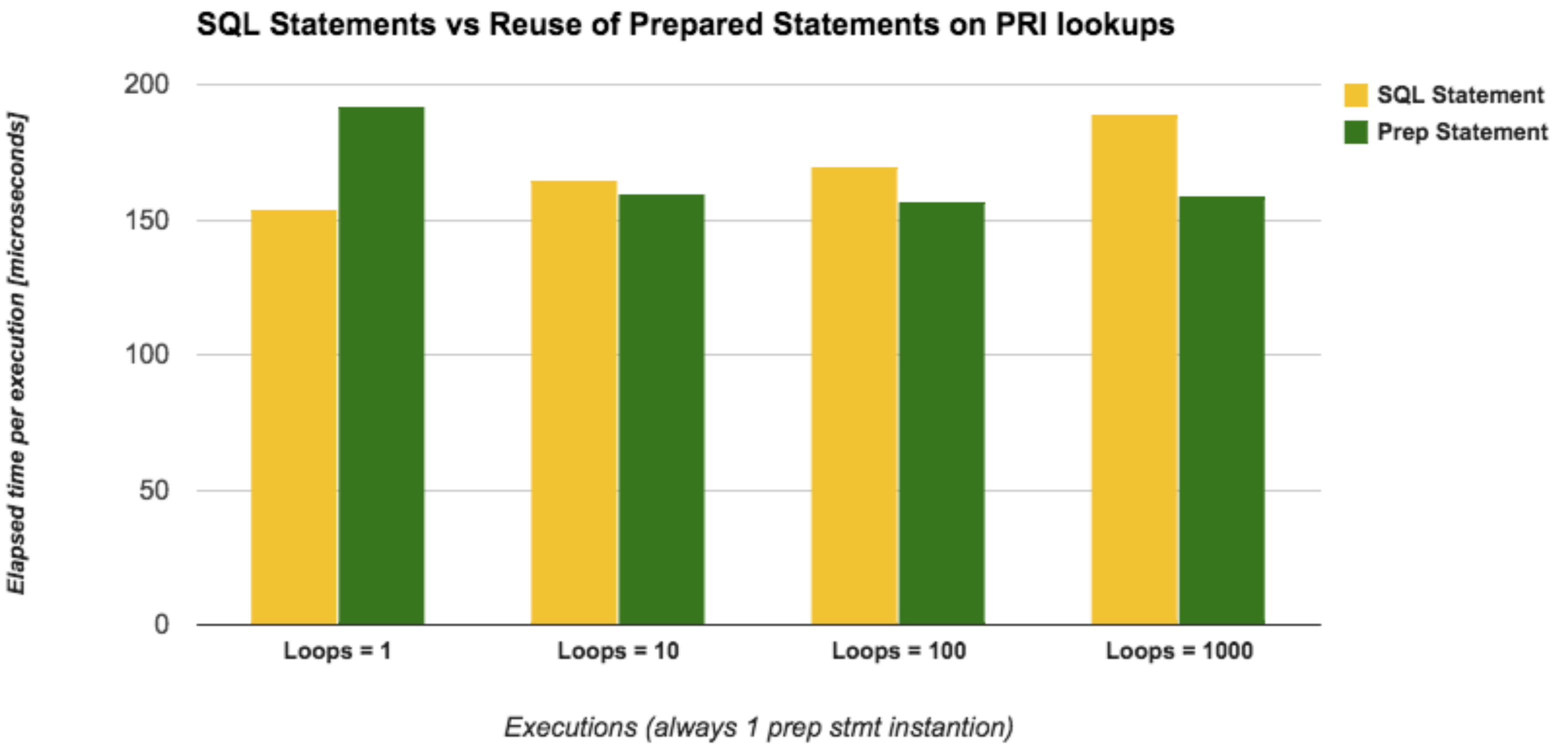


```
mysql> EXECUTE stmt1;
```

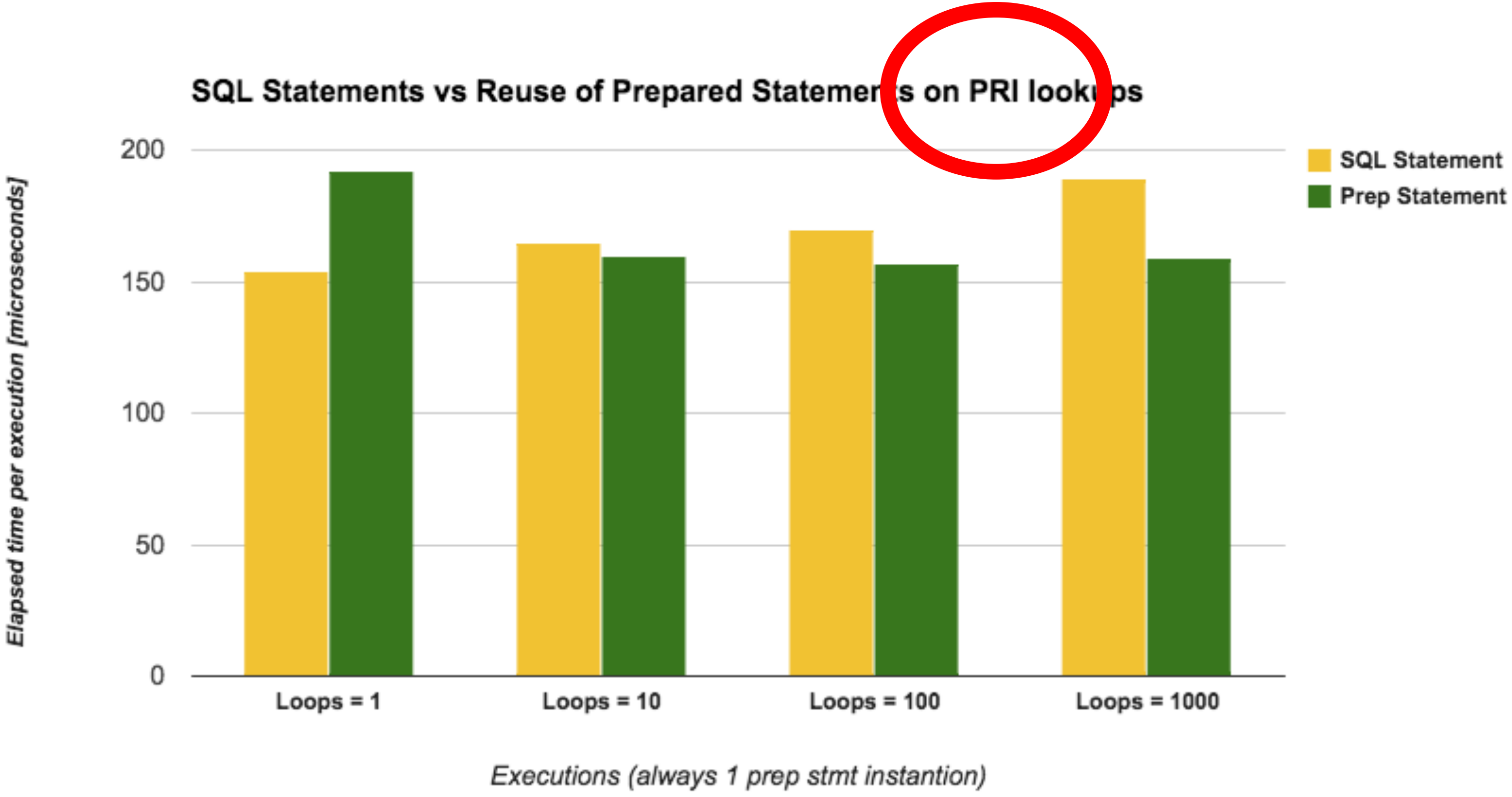
```
mysql> show profile;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| starting             | 0.000063 |
| checking permissions | 0.000011 |
| Opening tables       | 0.000067 |
| System lock          | 0.000019 |
| init                 | 0.000025 |
| optimizing           | 0.000007 |
| statistics           | 0.000017 |
| preparing            | 0.000015 |
| executing            | 0.000004 |
| Sending data         | 0.000095 |
| end                  | 0.000008 |
| query end            | 0.000076 |
| closing tables       | 0.000101 |
| query end            | 0.000004 |
| closing tables       | 0.000003 |
| freeing items        | 0.000022 |
| cleaning up          | 0.000005 |
+----------------------+----------+
17 rows in set (0.00 sec)
```

```
+----+----------+
| ID | Name     |
+----+----------+
|  1 | Kabul    |
|  2 | Qandahar |
|  3 | Herat    |
+----+----------+
3 rows in set (0.00 sec)
```

Speed Comparisons



Speed Comparisons



Review: SQL Statements vs Prepared Statements

SQL Statements

- Simplest
- Solution per Resultset
- Stateless
- Risk of Injection

Prepared Statements

- Partial Solution at Preparation
- Remainder Solution at Execution
- Execution per Resultset
- Less stateless
- No risk of Injection

Benefits & Drawbacks

SQL Statements

- Self-contained, prepare-free
- Long queries full of quoting can grow less readable
- Fastest option for one-offs
- Risk of Injection

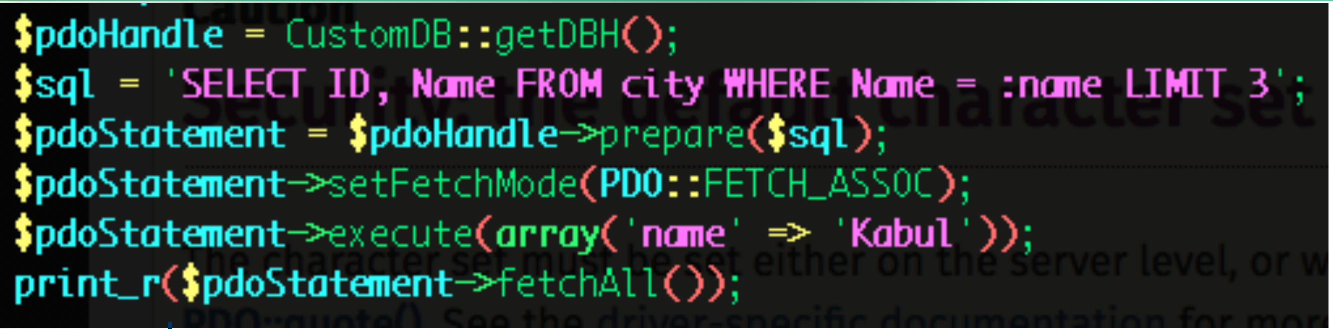
Prepared Statements

- 2-step, handle management
- Fastest option for re-use
- No visibility into memory consumption until 5.7
- Pool for PrepStmt Handles
- No risk of Injection

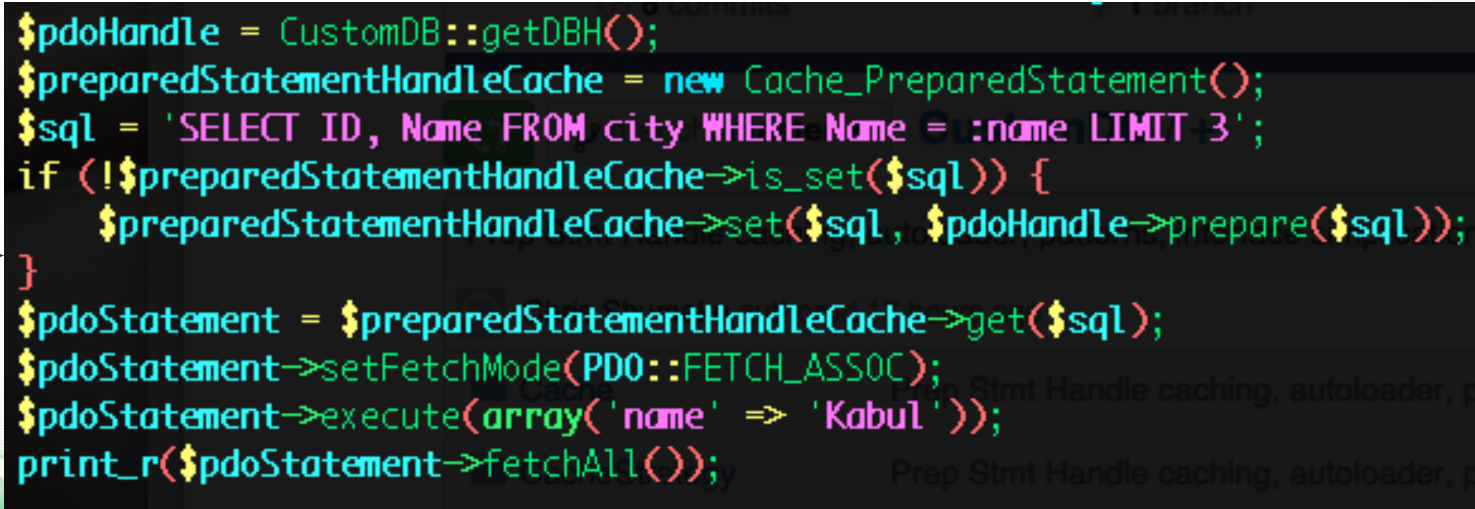
Coding Basics

How to deal with each issue Prepared Statements will cause

Coding Basics: Re-use Statement

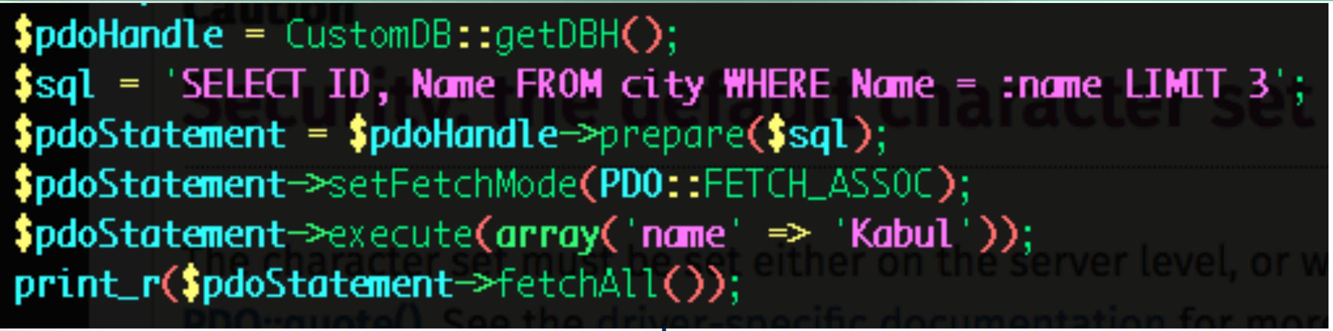


```
$pdoHandle = CustomDB::getDBH();
$sql = 'SELECT ID, Name FROM city WHERE Name = :name LIMIT 3';
$pdoStatement = $pdoHandle->prepare($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
$pdoStatement->execute(array('name' => 'Kabul'));
print_r($pdoStatement->fetchAll());
```



```
$pdoHandle = CustomDB::getDBH();
$preparedStatementHandleCache = new Cache_PreparedStatement();
$sql = 'SELECT ID, Name FROM city WHERE Name = :name LIMIT 3';
if (!$preparedStatementHandleCache->is_set($sql)) {
    $preparedStatementHandleCache->set($sql, $pdoHandle->prepare($sql));
}
$pdoStatement = $preparedStatementHandleCache->get($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
$pdoStatement->execute(array('name' => 'Kabul'));
print_r($pdoStatement->fetchAll());
```

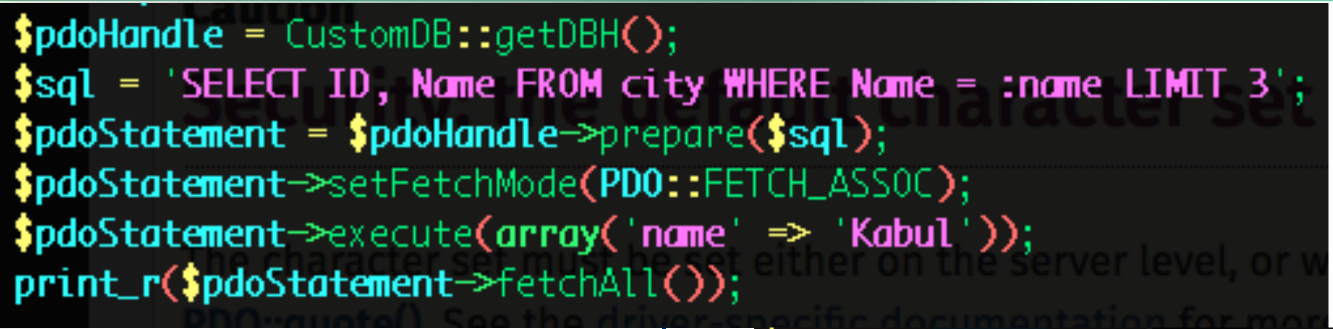
Coding Basics: Error Handling



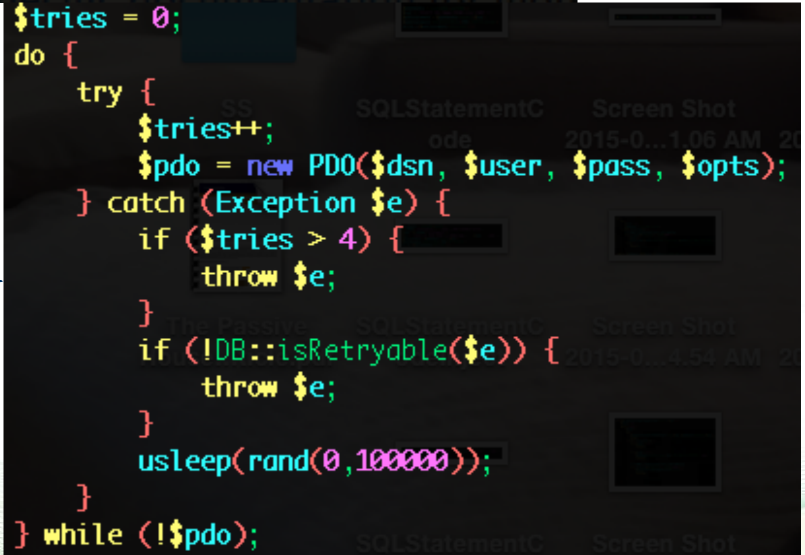
```
$pdoHandle = CustomDB::getDBH();
$sql = 'SELECT ID, Name FROM city WHERE Name = :name LIMIT 3';
$pdoStatement = $pdoHandle->prepare($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
$pdoStatement->execute(array('name' => 'Kabul'));
print_r($pdoStatement->fetchAll());
```

- isRetryError: Max Conns, Max PrepStmts
- isReconnectError: Connections, permissions
- Unrecoverable: Invalid syntax

Coding Basics: Error Handling - Max Conns

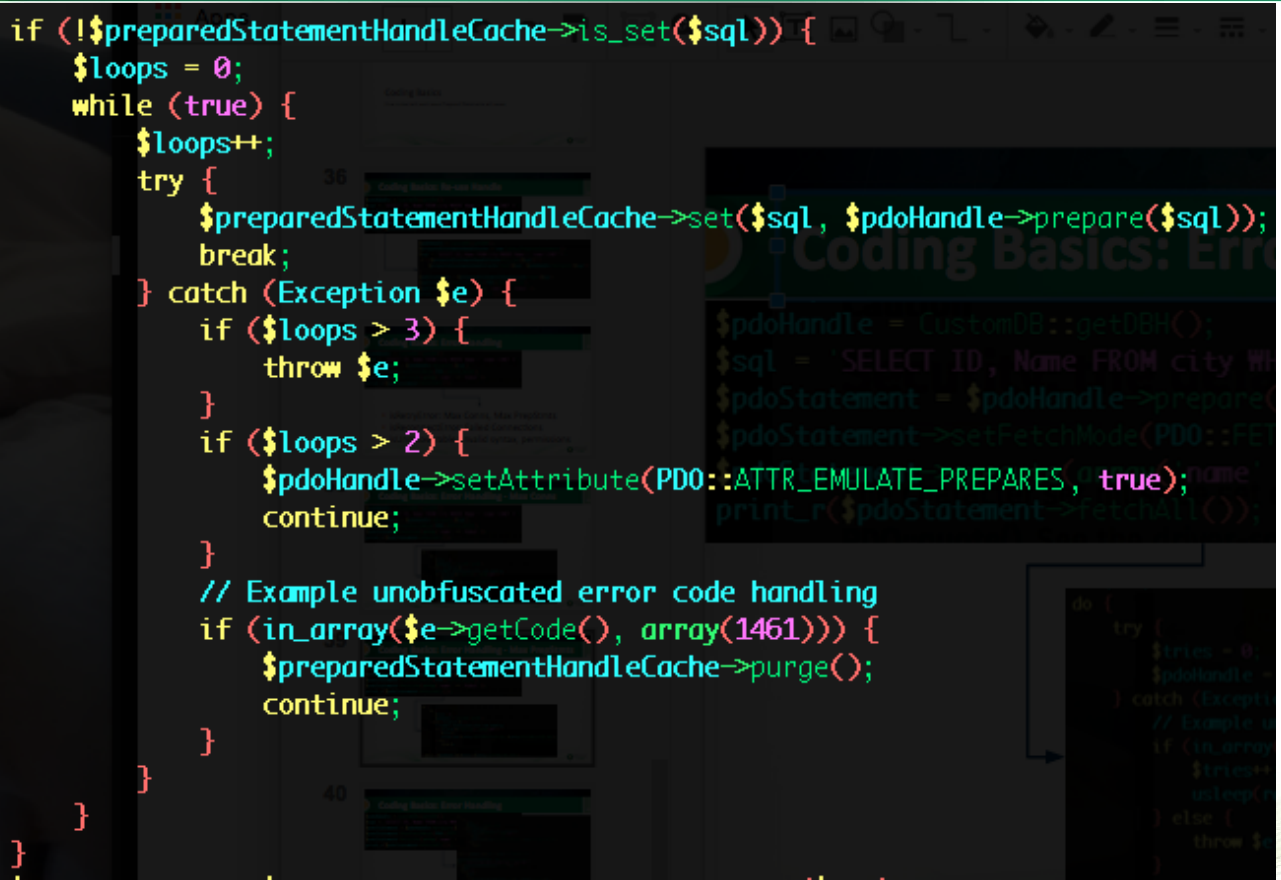


```php
$pdoHandle = CustomDB::getDBH();
$sql = 'SELECT ID, Name FROM city WHERE Name = :name LIMIT 3';
$pdoStatement = $pdoHandle->prepare($sql);
$pdoStatement->setFetchMode(PDO::FETCH_ASSOC);
$pdoStatement->execute(array('name' => 'Kabul'));
print_r($pdoStatement->fetchAll());
```



```php
$tries = 0;
do {
    try {
        $tries++;
        $pdo = new PDO($dsn, $user, $pass, $opts);
    } catch (Exception $e) {
        if ($tries > 4) {
            throw $e;
        }
        if (!DB::isRetryable($e)) {
            throw $e;
        }
        usleep(rand(0,100000));
    }
} while (!$pdo);
```

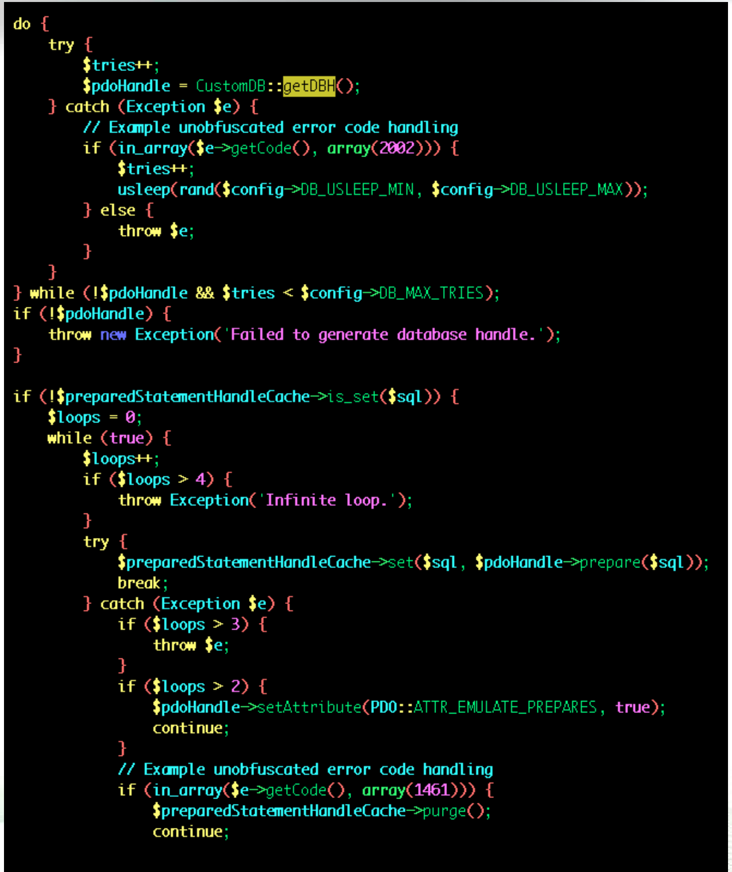
Coding Basics: Error Handling - Max PrepStmts



```
if (!$preparedStatementHandleCache->is_set($sql)) {
    $loops = 0;
    while (true) {
        $loops++;
        try {
            $preparedStatementHandleCache->set($sql, $pdoHandle->prepare($sql));
            break;
        } catch (Exception $e) {
            if ($loops > 3) {
                throw $e;
            }
            if ($loops > 2) {
                $pdoHandle->setAttribute(PDO::ATTR_EMULATE_PREPARES, true);
                continue;
            }
            // Example unobfuscated error code handling
            if (in_array($e->getCode(), array(1461))) {
                $preparedStatementHandleCache->purge();
                continue;
            }
        }
    }
}
```

PERCONA LIVE

Coding Basics: Error Handling - Reconnects

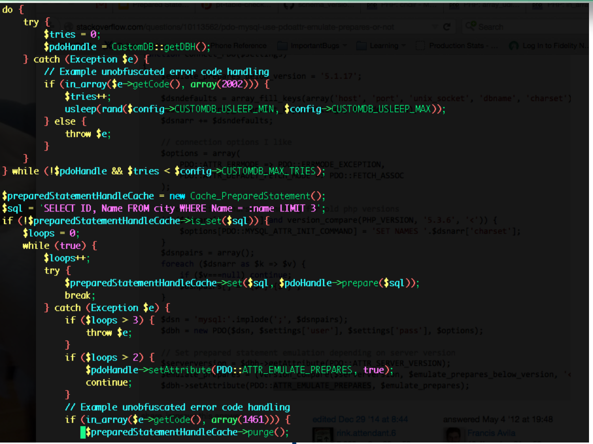


```
do {
    try {
        $tries++;
        $pdoHandle = CustomDB::getDBH();
    } catch (Exception $e) {
        // Example unobfuscated error code handling
        if (in_array($e->getCode(), array(2002))) {
            $tries++;
            usleep(rand($config->DB_USLEEP_MIN, $config->DB_USLEEP_MAX));
        } else {
            throw $e;
        }
    }
} while (!$pdoHandle && $tries < $config->DB_MAX_TRIES);
if (!$pdoHandle) {
    throw new Exception('Failed to generate database handle.');
}

if (!$preparedStatementHandleCache->is_set($sql)) {
    $loops = 0;
    while (true) {
        $loops++;
        if ($loops > 4) {
            throw Exception('Infinite loop.');
        }
        try {
            $preparedStatementHandleCache->set($sql, $pdoHandle->prepare($sql));
            break;
        } catch (Exception $e) {
            if ($loops > 3) {
                throw $e;
            }
            if ($loops > 2) {
                $pdoHandle->setAttribute(PDO::ATTR_EMULATE_PREPARES, true);
                continue;
            }
            // Example unobfuscated error code handling
            if (in_array($e->getCode(), array(1461))) {
                $preparedStatementHandleCache->purge();
                continue;
```

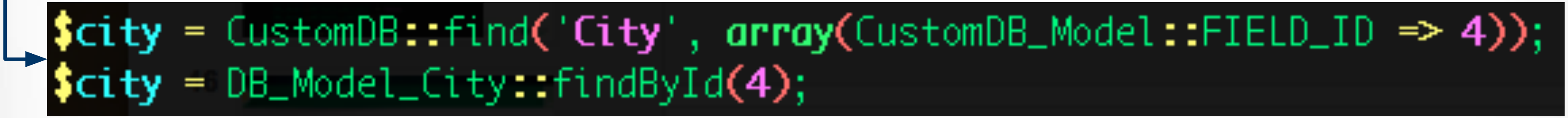
- More loops
  - Repeated code
  - Repeated errors

Coding Basics: DB Wrapper



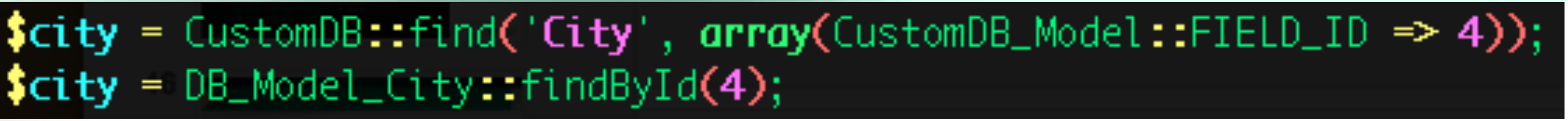
```
do {
    try {
        $tries = 0;
        $pdoHandle = CustomDB::getDBH();
    } catch (Exception $e) {
        // Example unobfuscated error code handling
        if (in_array($e->getCode(), array(2002))) {
            $tries++;
            usleep(rand($config->CUSTOMDB_USLEEP_MIN, $config->CUSTOMDB_USLEEP_MAX));
        } else {
            throw $e;
        }
    }
} while (!$pdoHandle && $tries < $config->CUSTOMDB_MAX_TRIES);

$preparedStatementHandleCache = new Cache_PreparedStatement();
$sql = 'SELECT ID, Name FROM city WHERE Name = :name LIMIT 3';
if (!$preparedStatementHandleCache->is_set($sql)) {
    $loops = 0;
    while (true) {
        $loops++;
        try {
            $preparedStatementHandleCache->set($sql, $pdoHandle->prepare($sql));
            break;
        } catch (Exception $e) {
            if ($loops > 3) {
                throw $e;
            }
            if ($loops > 2) {
                $pdoHandle->setAttribute(PDO::ATTR_EMULATE_PREPARES, true);
                continue;
            }
            // Example unobfuscated error code handling
            if (in_array($e->getCode(), array(1461))) {
                $preparedStatementHandleCache->purge();
```



```
$city = CustomDB::find('City', array(CustomDB_Model::FIELD_ID => 4));
$city = DB_Model_City::findById(4);
```

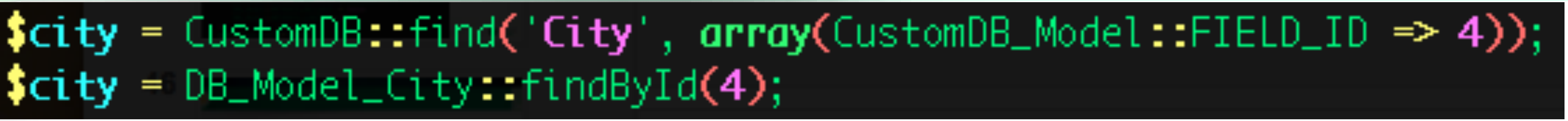
Coding Basics: Good Enough DB Wrappers



```
$city = CustomDB::find('City', array(CustomDB_Model::FIELD_ID => 4));
$city = DB_Model_City::findById(4);
```

- Its job: object instantiation & property setting
- Error handling: connections go away
- Insight: logging, stats, exceptions
- Caching: expiration, bypassable, layers
- Fully bypassable w/SQL, avoids ORM behavior
- Avoids handling code 1317, allows maint mode

Coding Basics: DB Wrapper Points



```
$city = CustomDB::find('City', array(CustomDB_Model::FIELD_ID => 4));
$city = DB_Model_City::findById(4);
```

- Error Handling:
  - Reconnect Errors: ensure prep stmt caches flushed
- Caching:
  - Use the SQL and database identifier for an index hash for prepared statements
  - Use the same, plus the values, for results caching

Code Considerations

How to deal with each issue Prepared Statements will cause

Issues to resolve in Code

- Additional state:
  - Creating Prepared Statements
  - Managing Prepared Statement Handles
  - Finding Prepared Statement Handles for Use
  - Closing Prepared Statement Handles on Issue
- Ripping out all Injection Prevention Code

Error Codes related to Prepared Statements

- 1243: Unknown Prepared Statement Handle
  - Deallocate & re-prepare
- 1390: Too Many Parameters
  - Limit parameter count, potentially encode a splitter for any IN clause
- 1420: Exec called on Stmt w/ Open Cursor
  - Close cursor & re-execute

Error Codes related to Prepared Statements

- 1444: Recursion with Stored Procedure
  - Do not use recursion w/ Prep Stmt in SPs
- 1461: Too Many Prepared Statements
  - Flush Prep Stmt Cache & re-prepare
  - Eventually, activate Emulation
- 1615: Statement needs to be Re-Prepared
  - Re-prepare

Good code prevents all errors vs SQL Stmts

- If the server-side prepared statement pool is full, emulation can prevent client-side errors
- If the database is down, prepared statements won’t hurt you. Or help you.

Operational Considerations

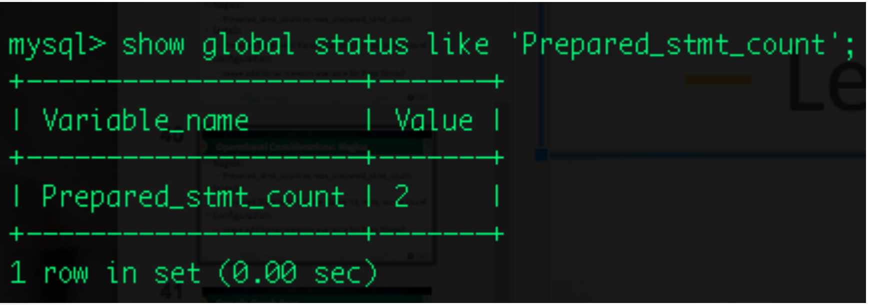
How to spot each issue Prepared Statements will cause

Operational Considerations

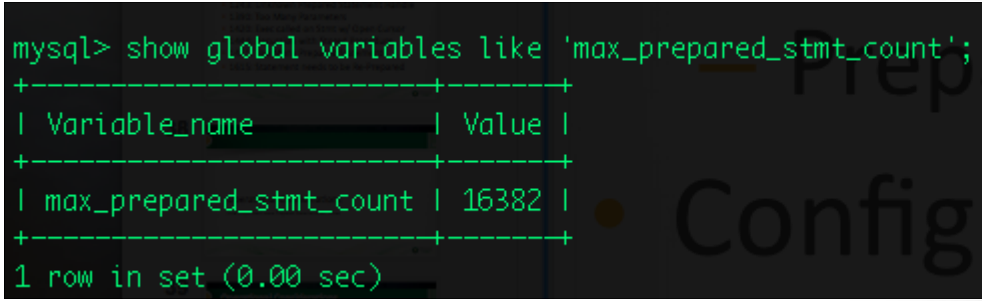
- Nagios:
  - Prepared_stmt_count vs max_prepared_stmt_count
- Stats:
  - Prepared Statement Handle Cache hit, miss, purge count
- Configuration:
  - Leave additional memory available for Prep Stmts? Maybe.

Operational Considerations: Nagios

Prepared_stmt_count vs max_prepared_stmt_count



```
mysql> show global status like 'Prepared_stmt_count';
+---------------------+-------+
| Variable_name       | Value |
+---------------------+-------+
| Prepared_stmt_count | 2     |
+---------------------+-------+
1 row in set (0.00 sec)
```



```
mysql> show global variables like 'max_prepared_stmt_count';
+-------------------------+-------+
| Variable_name           | Value |
+-------------------------+-------+
| max_prepared_stmt_count | 16382 |
+-------------------------+-------+
1 row in set (0.00 sec)
```

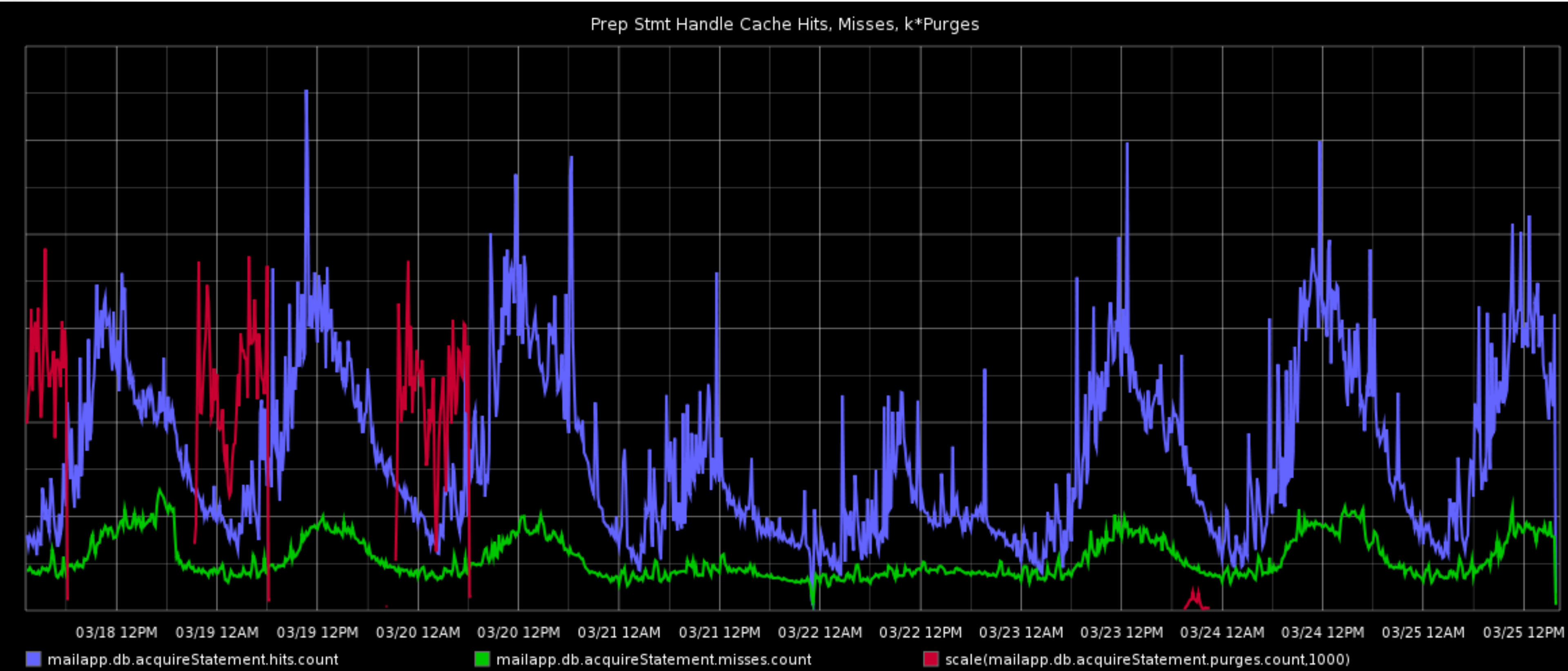
Monitor it like max_connections

Operational Considerations: Nagios

Monitor it like max_connections?

Yes. Still, monitor it.

Operational Considerations: StatsD



Operational Considerations: Configuration

- Prepared Statement Handle Cache
  - Strategy: LRU - if dropping handles is consistent
  - Strategy: Queue - if cache is large enough
  - Cache Size: 200 at Bronto leads to a Queue
- Requires a Good Enough database wrapper

Operational Considerations: Parameter Types

- Named Parameters
  - Associative arrays in code increase readability
  - Values do appear in processlist
- Ordered Parameters
  - Can still use associative array
  - Values do not appear in processlist

Addressing Notes

- “Prepared Statement allocation is specific to a session until the end of a session or deallocation.” <- Will read this aloud (read: unconfirmed memory leak if Stored Proc killed w/open Prep Statement Handle)
- Large lists of Named Parameters are not any more expensive than large lists of Ordered Parameters

Wrap-up

- Prepared Statements are Injection-immune
- Prepared Statements want a handle cache
- Prepared Statements have resolvable failure scenarios with reasonable code
- A good-enough database wrapper is needed

Thanks

Questions?

Walkthrough:
github.com/cvshumake/pLiveDB

Basic implementation for some examples:
github.com/cvshumake/CustomDB

Slides available:
shumake.mobi/slides/PerconaLive2015