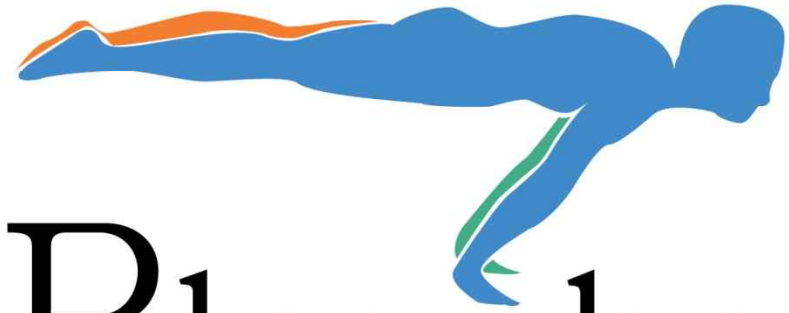


Planche

for MySQL

정 주원

정 주원

NIPA Global Open Frontier Lab

Korea Sencha User Group

https://www.facebook.com/jungjuwon

jjwcom@nate.com

https://github.com/jeongjuwon/planche

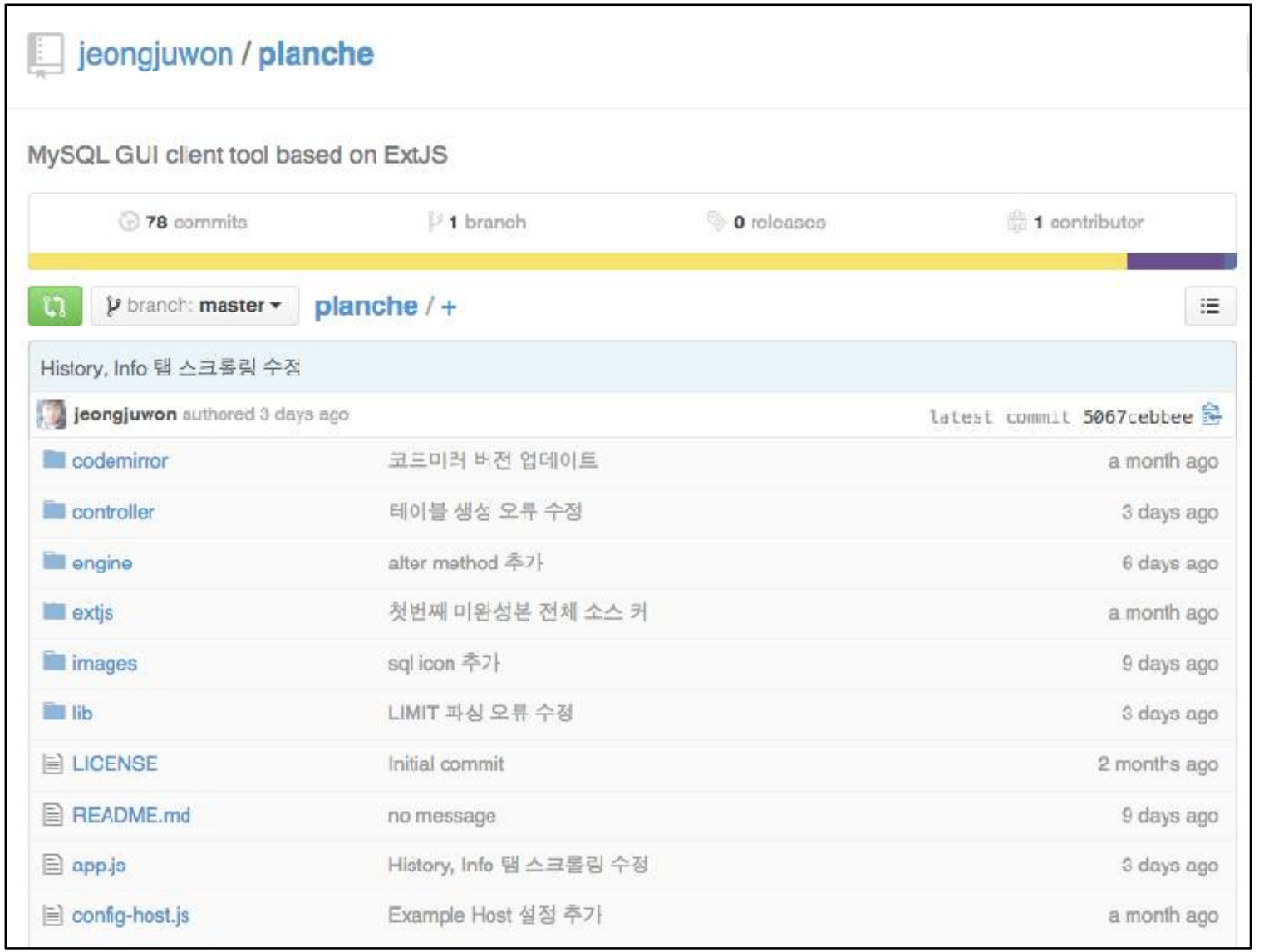
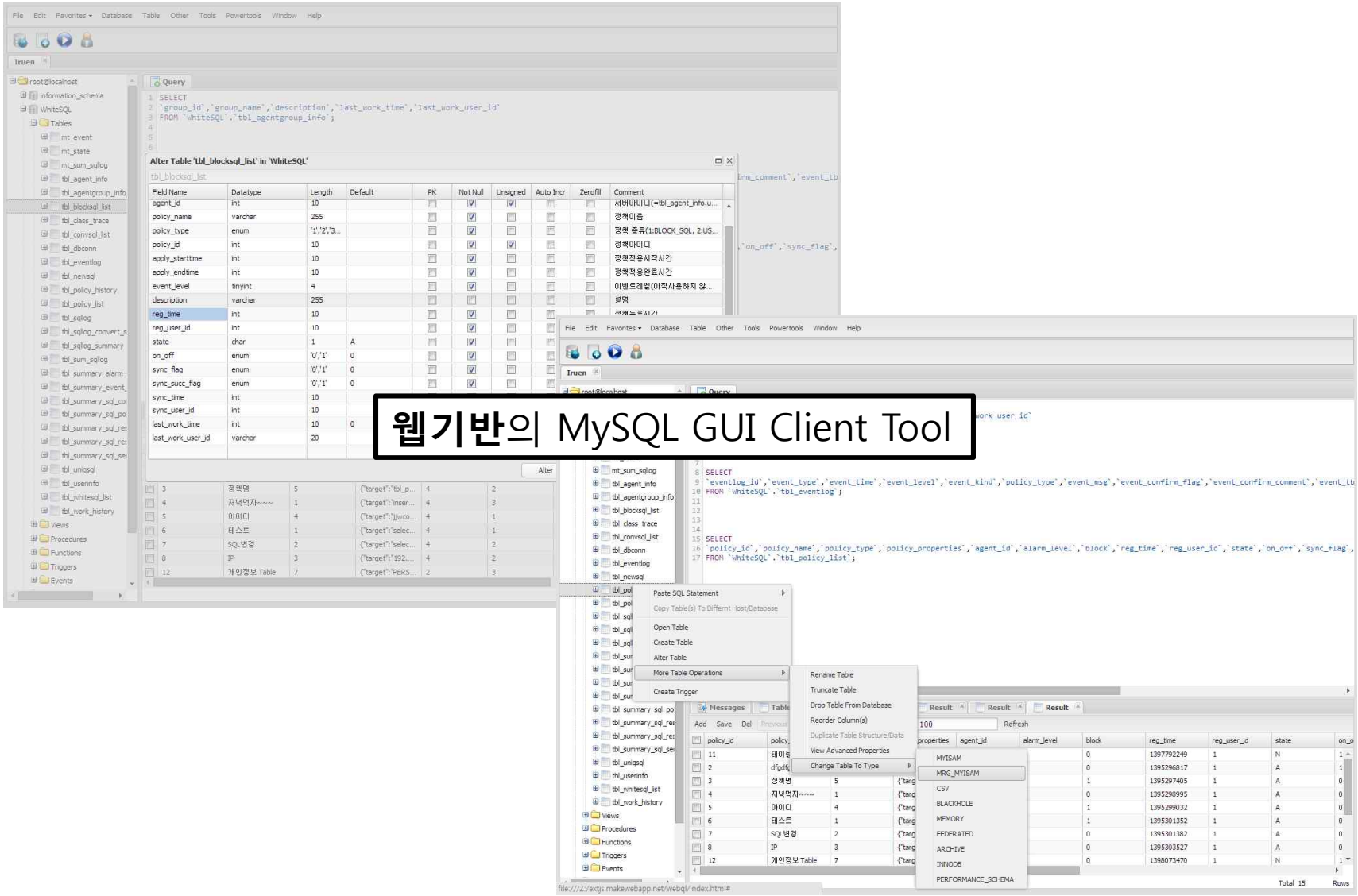


Planche for MySQL



FRAMEWORK

Planche

Sencha Ext JS

http://www.sencha.com/products/extjs/

Why Ext JS?

(완벽한)컴포넌트 기반

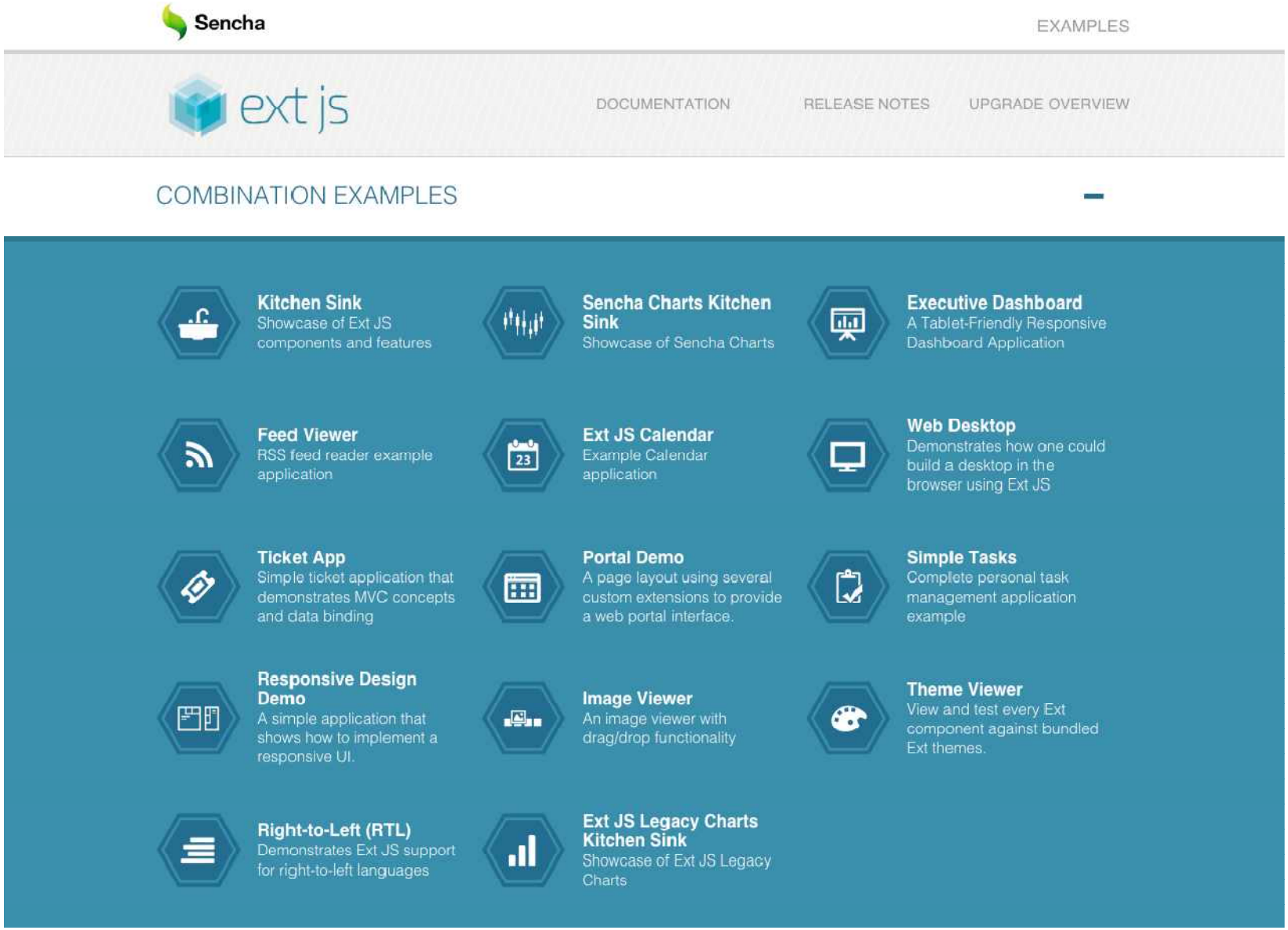
마크업(X)

데스크탑 APP(같은..)

http://youtu.be/QmQzCi6MqU4

Planche

http://dev.sencha.com/ext/5.0.1/examples/index.html



Code Mirror

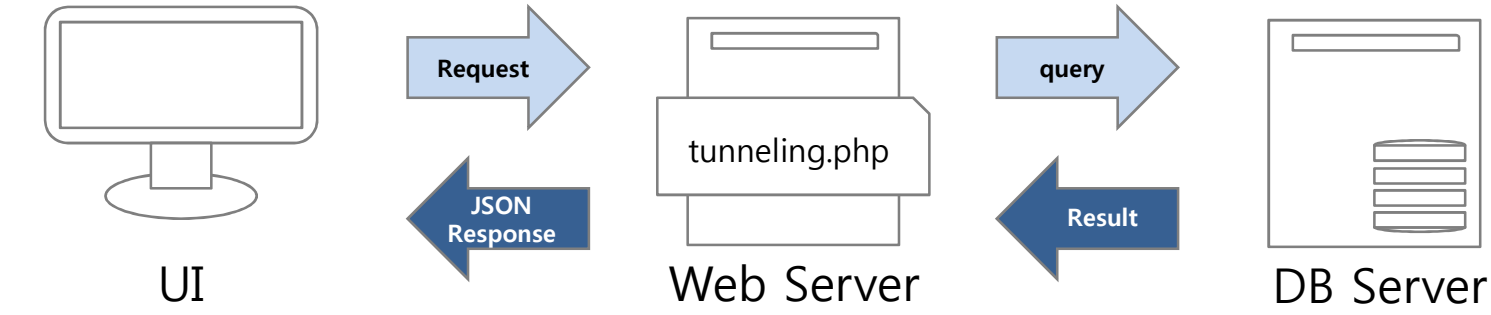
```
// The bindings defined specifically in the Sublime Text mode
var bindings = {
  "Alt-Left": "goSubwordLeft",
  "Alt-Right": "goSubwordRight",
  "Ctrl-Up": "scrollLineUp",
  "Ctrl-Down": "scrollLineDown",
  "Shift-Ctrl-L": "splitSelectionByLine",
  "Shift-Tab": "indentLess",
  "Esc": "singleSelectionTop",
  "Ctrl-L": "selectLine",
  "Shift-Ctrl-K": "deleteLine",
  "Ctrl-Enter": "insertLineAfter",
  "Shift-Ctrl-Enter": "insertLineBefore",
  "Ctrl-D": "selectNextOccurrence",
  "Shift-Ctrl-Space": "selectScope",
  "Shift-Ctrl-M": "selectBetweenBrackets",
  "Ctrl-M": "goToBracket",
  "Shift-Ctrl-Up": "swapLineUp",
```

- Javascript기반의 Text Editor
- **Planche에서 Query Editor로 쓰임**
- 60개 이상의 다양한 언어 편집 지원
- 다양한 API, PLUGIN 지원

CONCEPT

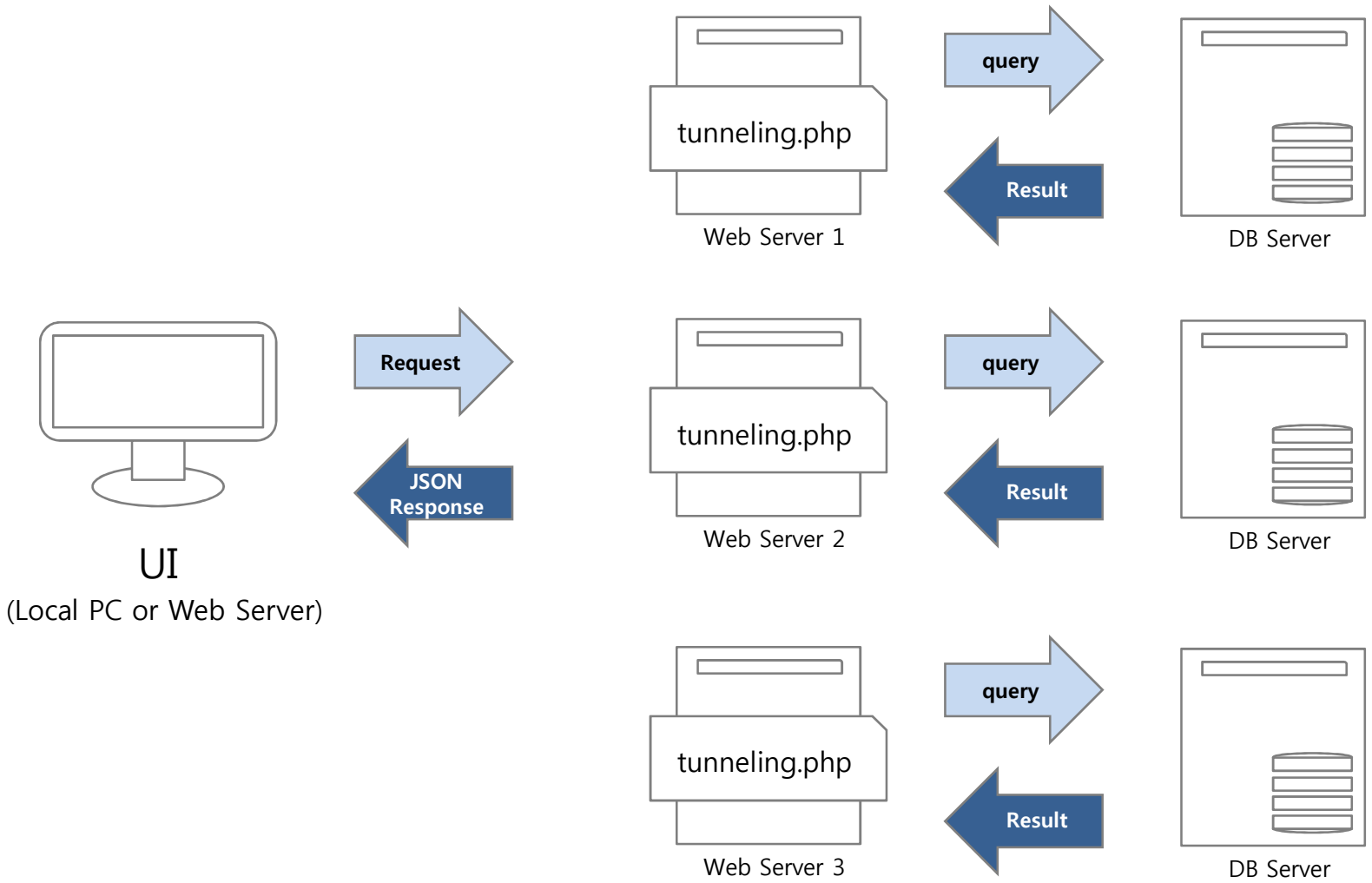
Concept

- HTTP 터널링



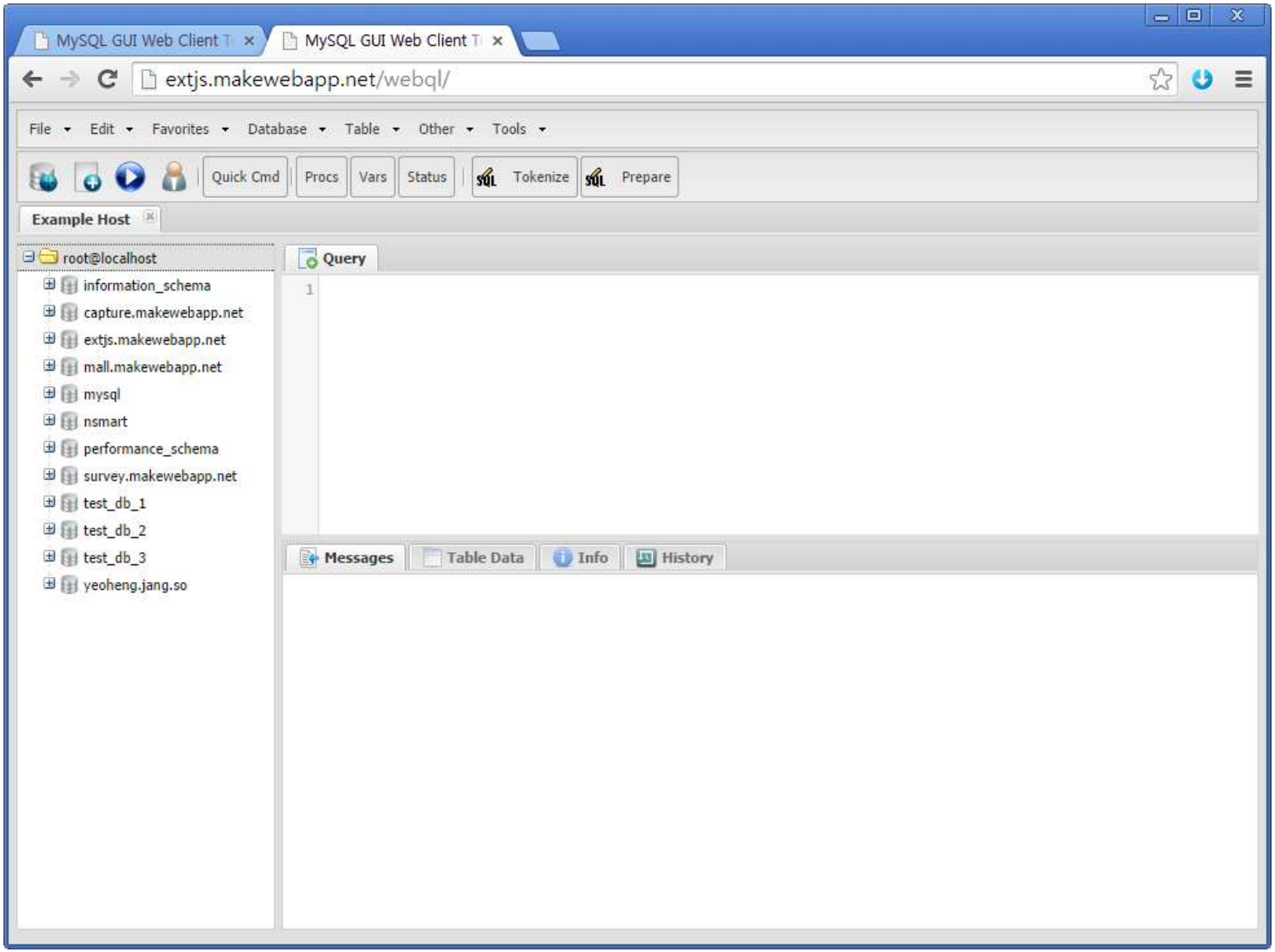
Concept

- HTTP 터널링



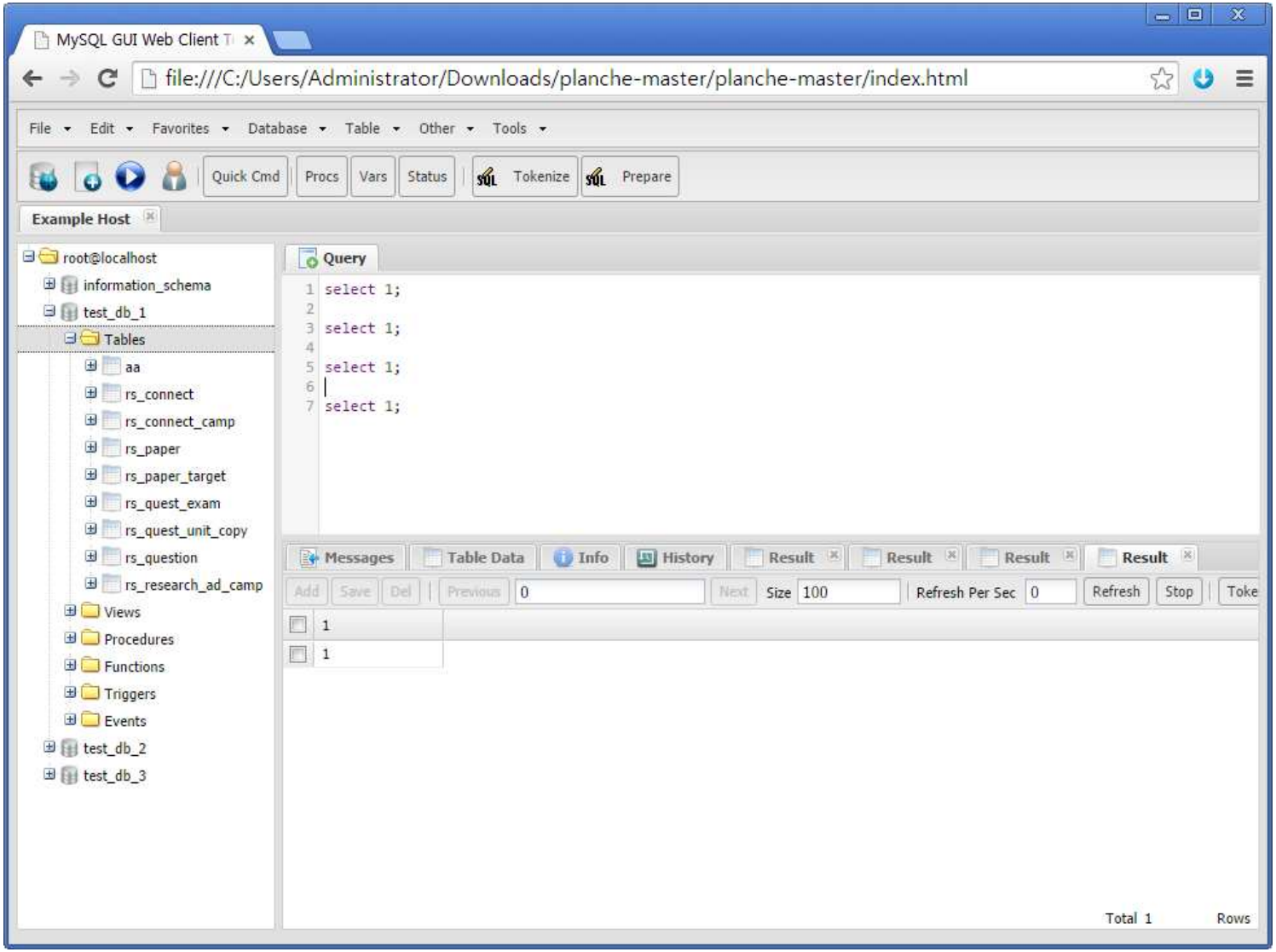
Concept

- 기존 처럼 웹서버에 설치 후 URL 입력



Concept

- PC에 다운 받은 상태에서 바로 실행



Concept

- AJAX를 사용하지 않음

Concept

- JSONP 사용

Concept을 위한 구현 방법

```
Ext.namespace('Planche');

Ext.application({
    appFolder   : '.',
    name        : 'Planche',
    history     : [],
    launch      : function() {

        //#         ./lib/Window.js
        //#         편리하지만 내부적으로 AJAX 호출
        Ext.create('Planche.lib.Window', {
            width : 800,
            height : 500
        });

        //#         AJAX 호출
        Ext.Ajax.request({
            url: 'tunneling.php',
            params: {
                host    : 'host',
                user    : 'user',
                pass    : 'pass',
                db      : 'db',
                query   : 'query'
            },
            success: function(response){

                //todo
            }
        });
    }
});
```

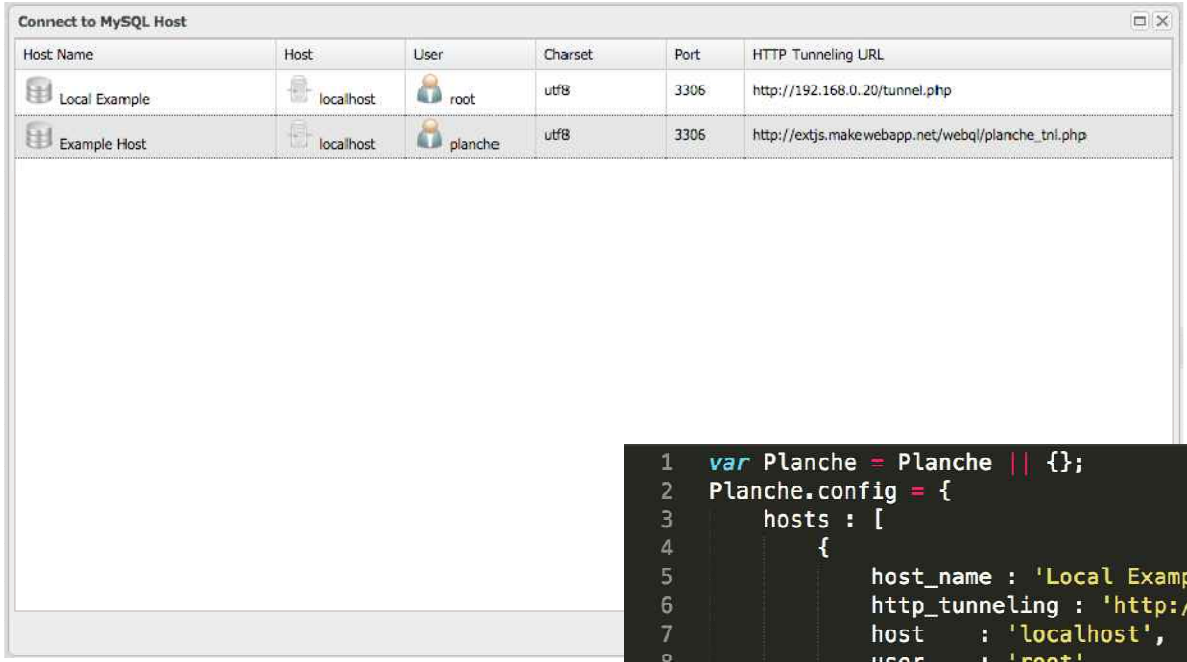
JSONP

```
Ext.namespace('Planche');

Ext.application({
    appFolder   : '.',
    name        : 'Planche',
    history     : [],
    launch      : function() {

        //#         ./lib/Window.js
        //#         Ext.require 이용
        Ext.require('Planche.lib.Window', function(){

            Ext.create('Planche.lib.Window', {
                width : 800,
                height : 500
            });

        }, this);

        //#         JSONP 호출
        Ext.data.JsonP.request({
            url: 'tunneling.php',
            params: {
                host    : 'host',
                user    : 'user',
                pass    : 'pass',
                db      : 'db',
                query   : 'query'
            },
            callbackKey : 'callback',
            success: function(response) {

                //todo
            }
        });
    }
});
```

CONNECTION

Connection

호스트 설정




```
var Planche = Planche || {};
Planche.config = {
    hosts : [
        {
            host_name : 'Local Example',
            http_tunneling : 'http://192.168.0.20/tunnel.php',
            host    : 'localhost',
            user    : 'root',
            pass    : 'password',
            charset : 'utf8',
            port    : 3306
        },
        {
            host_name : 'Example Host',
            http_tunneling : 'http://extjs.makewebapp.net/webql/planche_tnl.php',
            host    : 'localhost',
            user    : 'planche',
            pass    : 'planche',
            charset : 'utf8',
            port    : 3306
        }
    ]
}
```

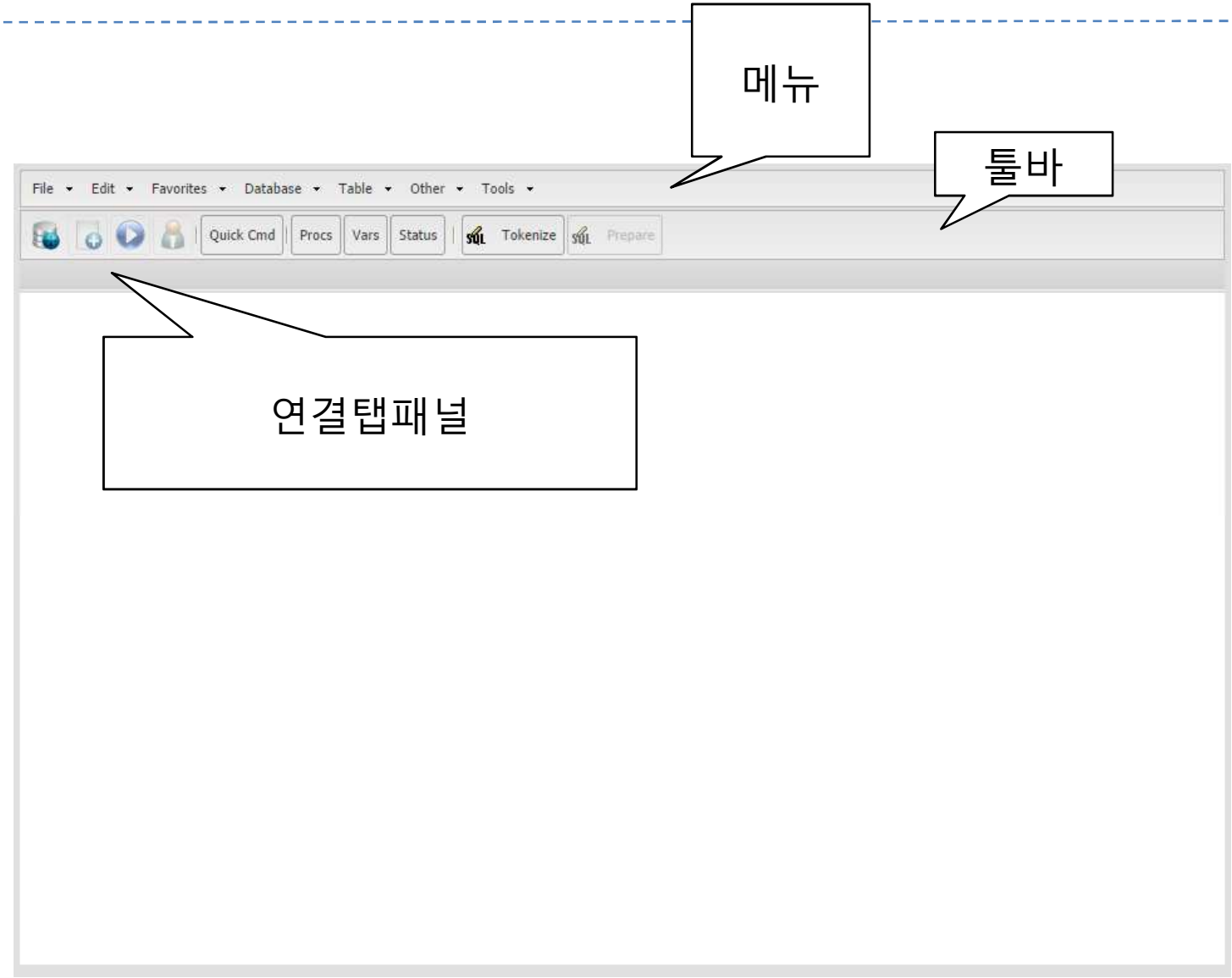
LAYOUT

레이아웃

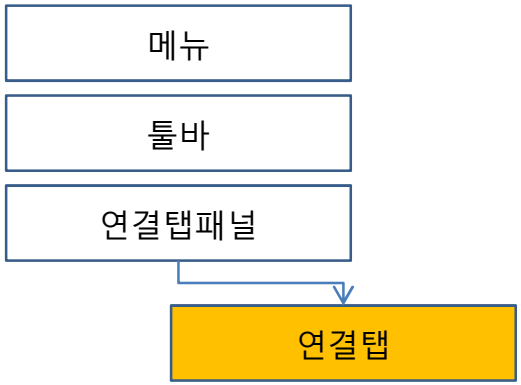
메뉴

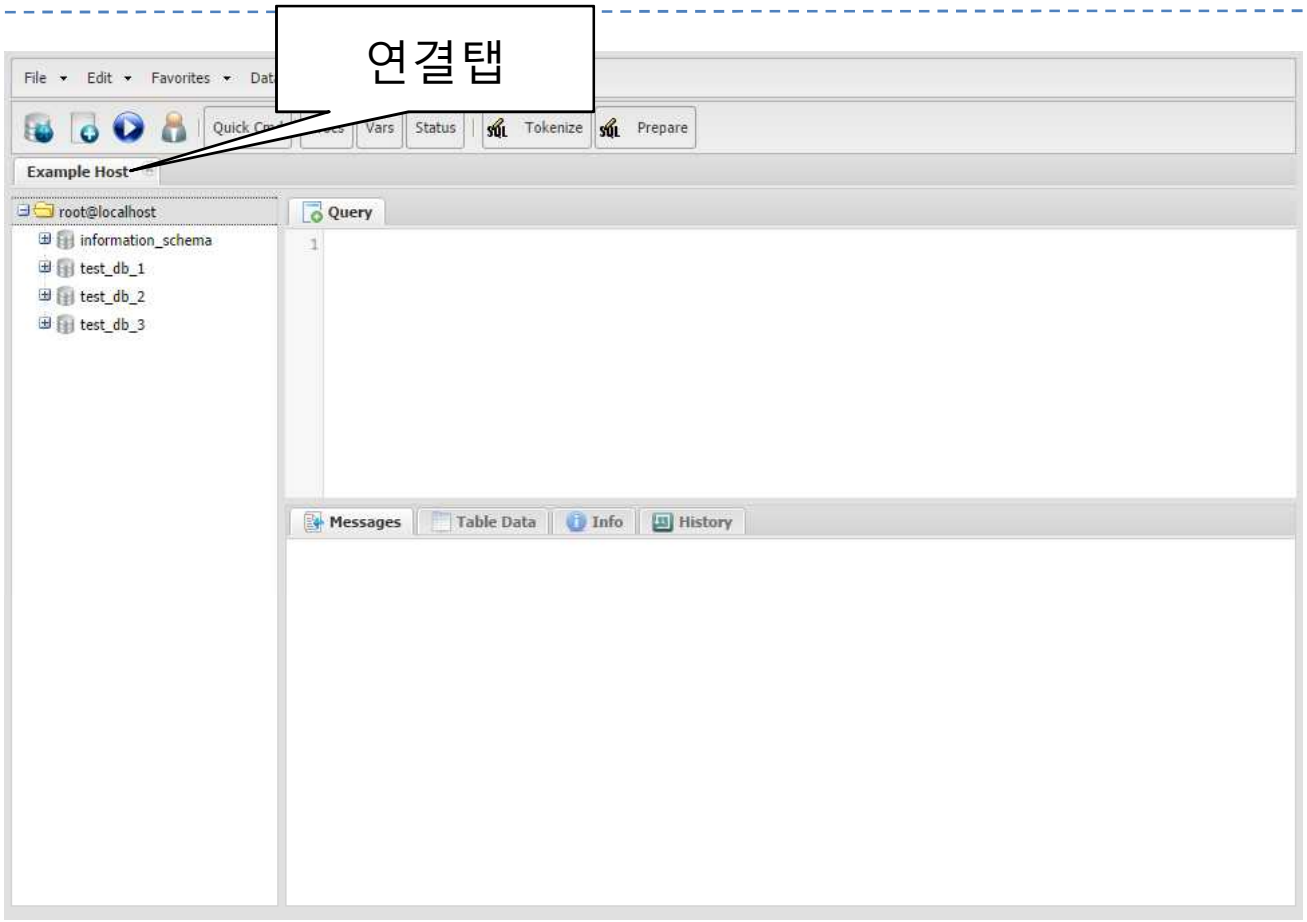
툴바

연결탭패널

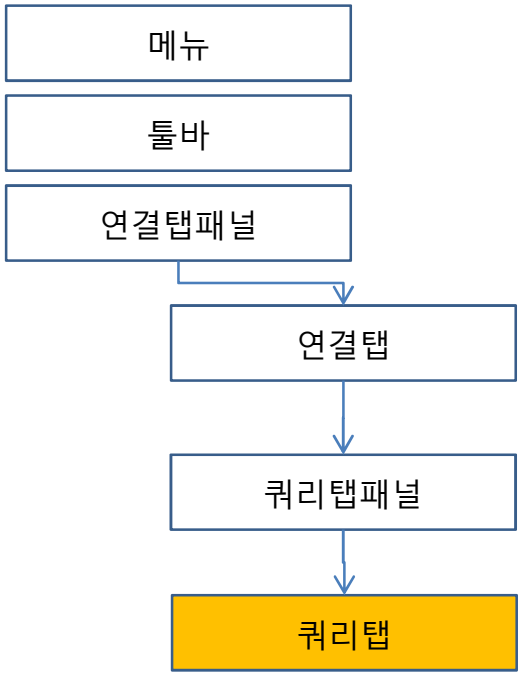


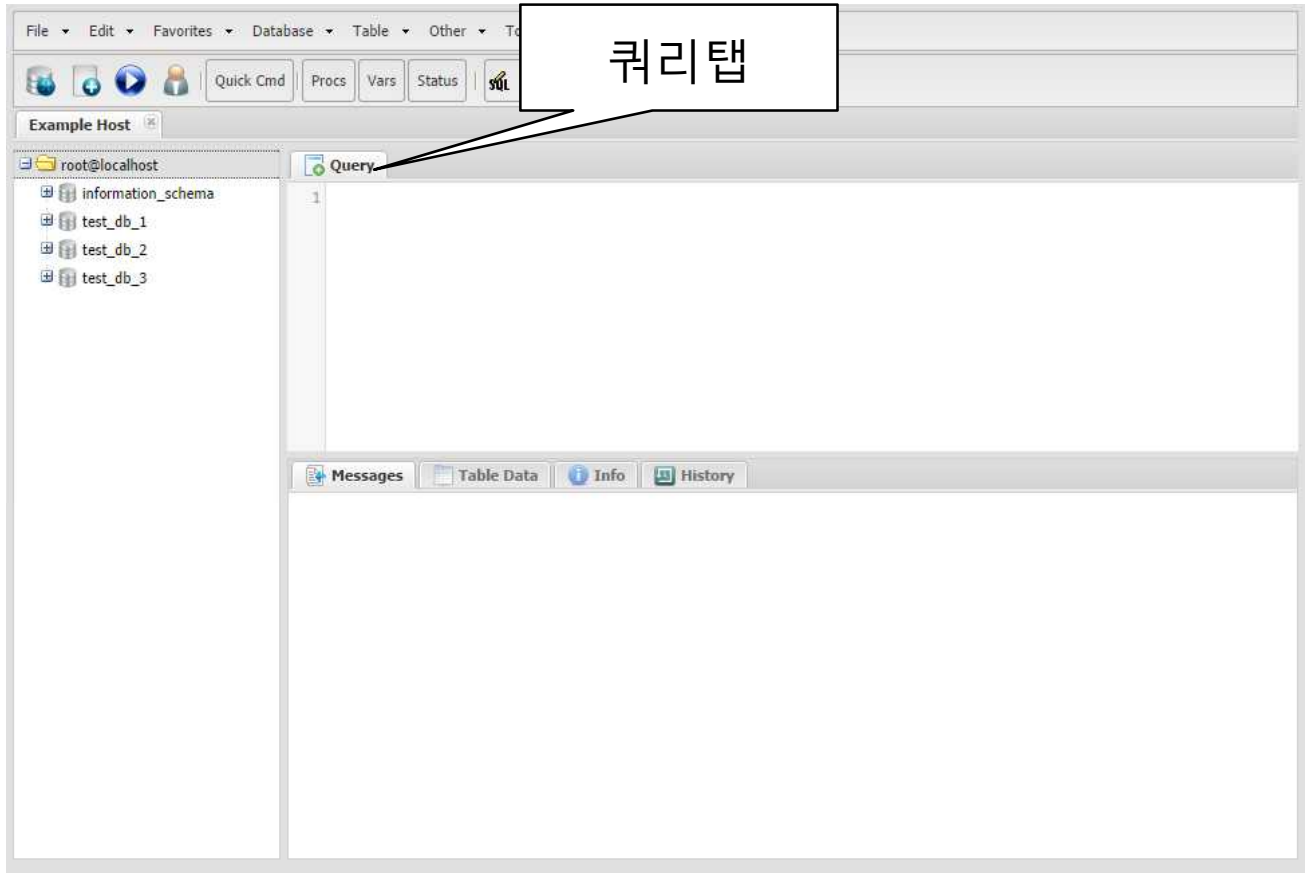
레이아웃


메뉴
툴바
연결탭패널
연결탭

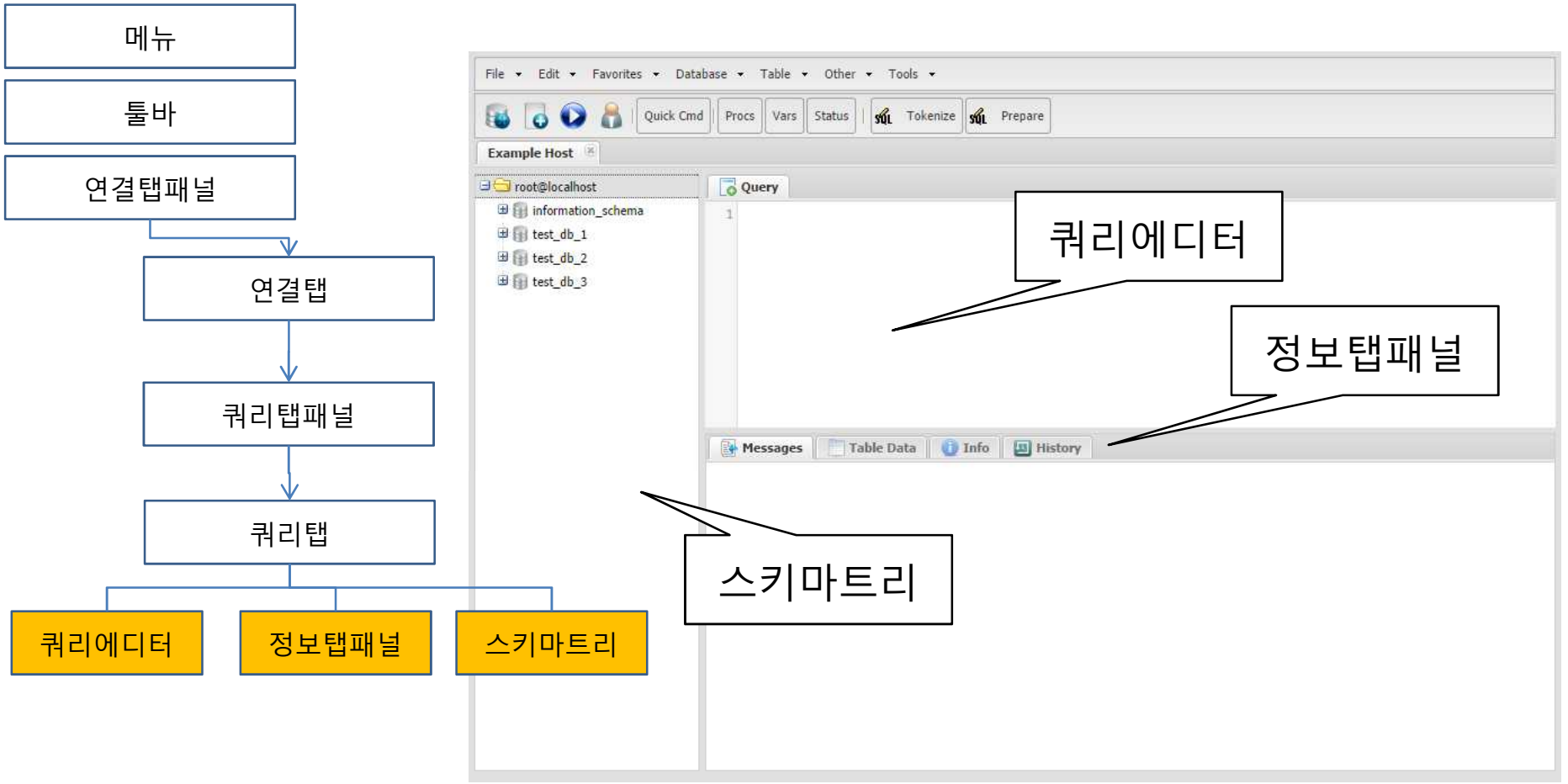

연결탭
File
Edit
Favorites
Vars
Status
Tokenize
Prepare
Example Host
root@localhost
information_schema
test_db_1
test_db_2
test_db_3
Query
1
Messages
Table Data
Info
History

레이아웃

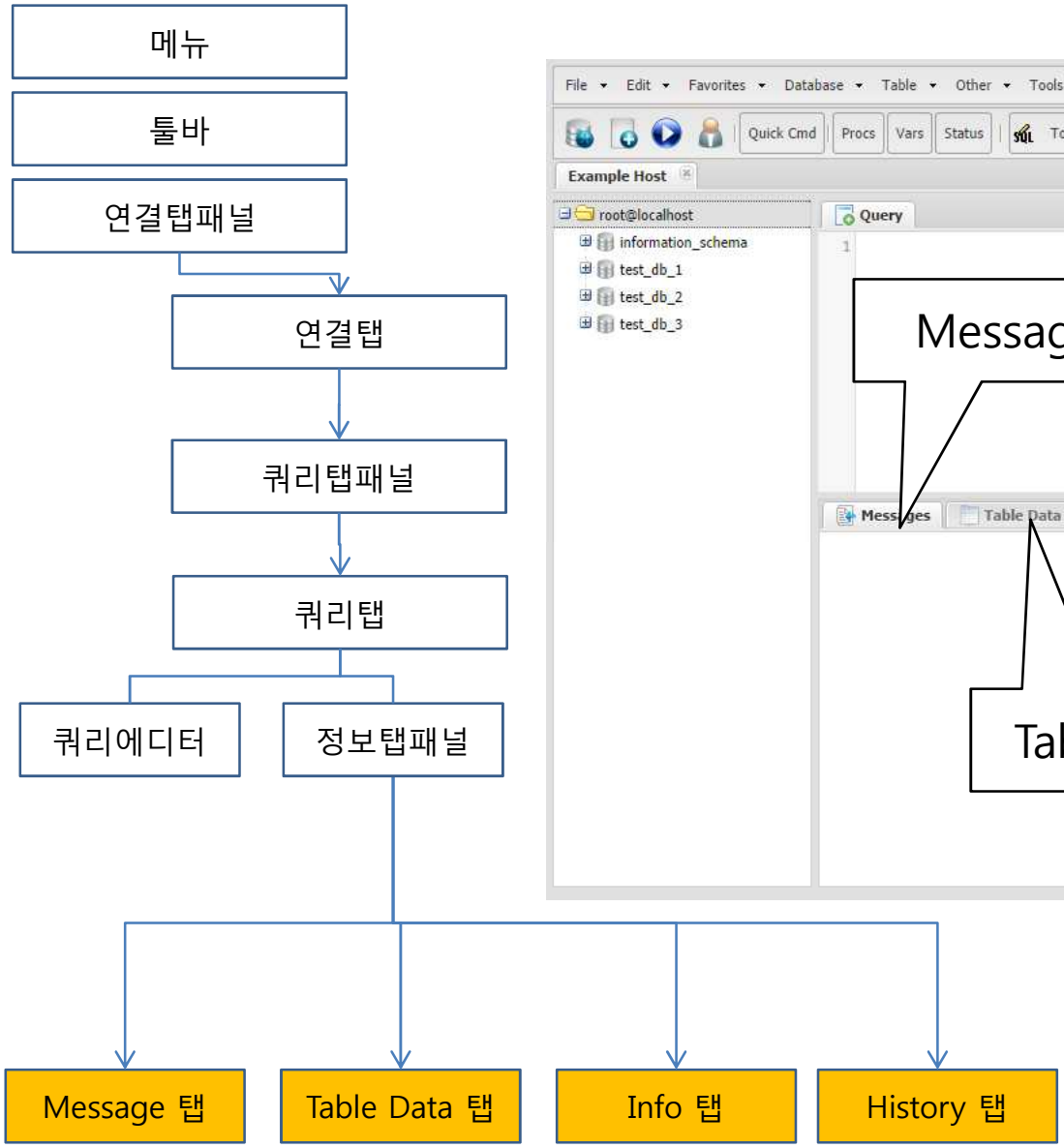

메뉴
툴바
연결탭패널
연결탭
쿼리탭패널
쿼리탭

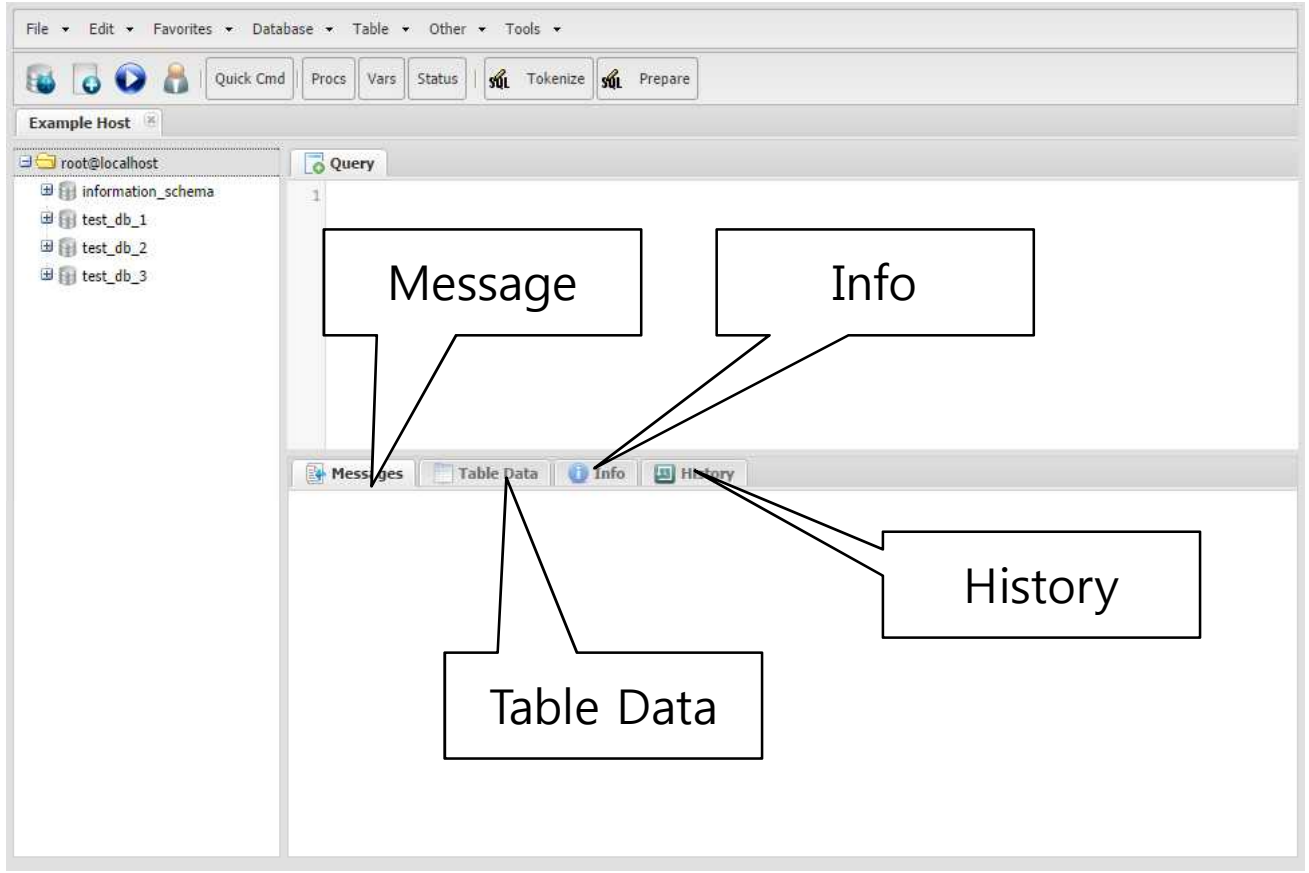

쿼리탭
File
Edit
Favorites
Database
Table
Other
Quick Cmd
Procs
Vars
Status
Example Host
root@localhost
information_schema
test_db_1
test_db_2
test_db_3
Query
Messages
Table Data
Info
History

레이아웃

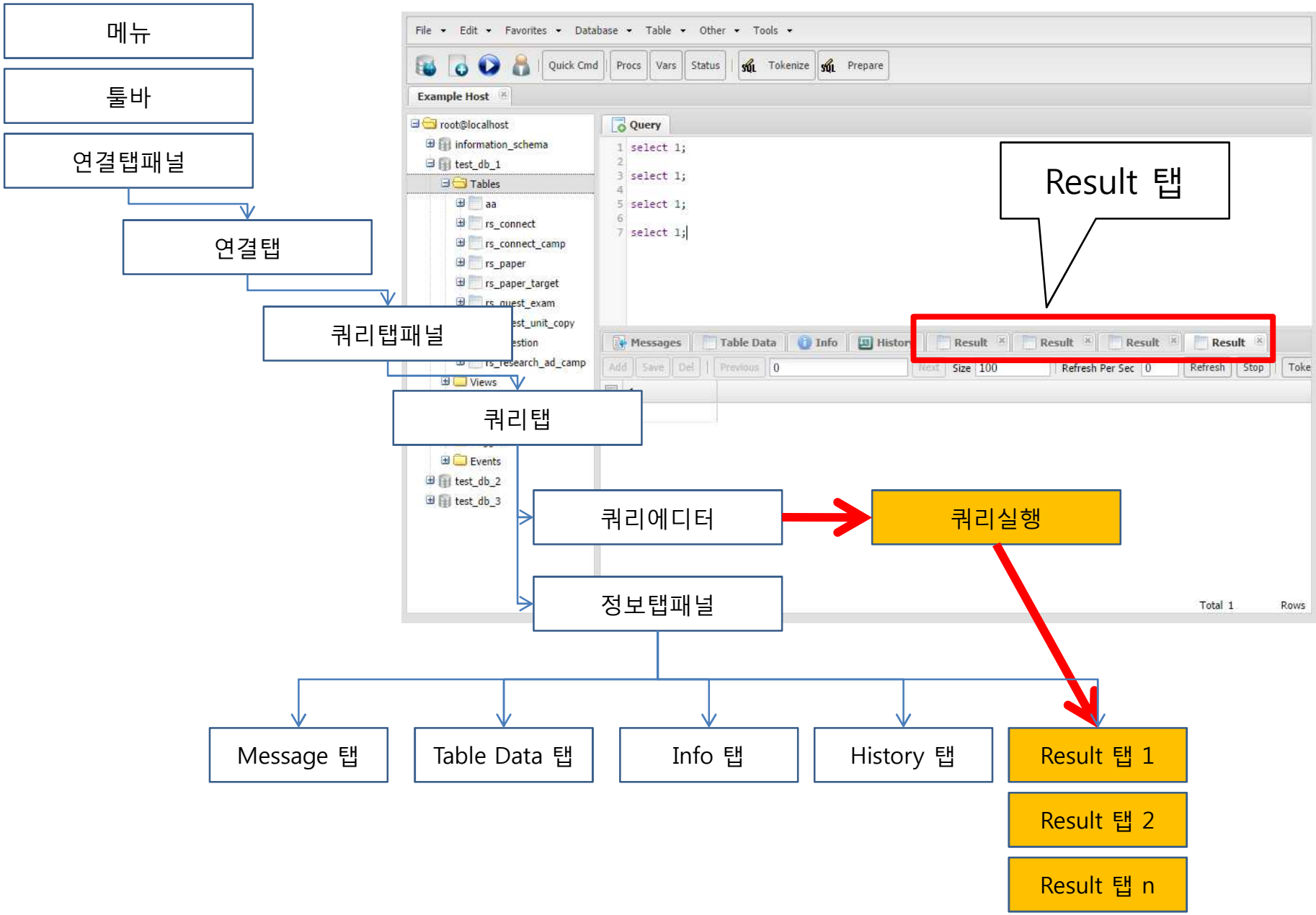

메뉴
툴바
연결탭패널
연결탭
쿼리탭패널
쿼리탭
쿼리에디터
정보탭패널
스키마트리
File
Edit
Favorites
Database
Table
Other
Tools
Quick Cmd
Procs
Vars
Status
Tokenize
Prepare
Example Host
root@localhost
information_schema
test_db_1
test_db_2
test_db_3
Query
Messages
Table Data
Info
History
쿼리에디터
정보탭패널
스키마트리

레이아웃


메뉴
툴바
연결탭패널
연결탭
쿼리탭패널
쿼리탭
쿼리에디터
정보탭패널
Message 탭
Table Data 탭
Info 탭
History 탭

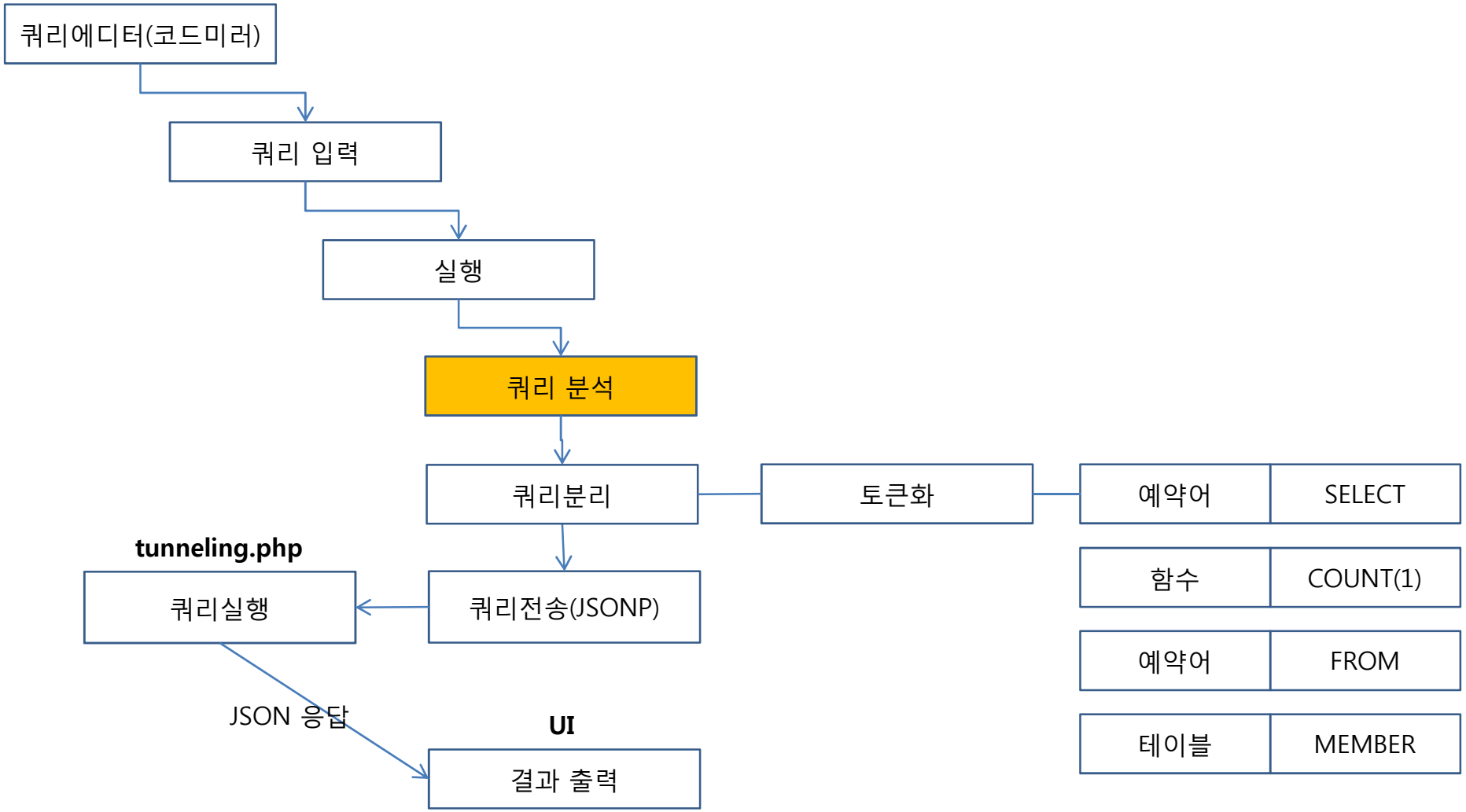

File
Edit
Favorites
Database
Table
Other
Tools
Quick Cmd
Procs
Vars
Status
Tokenize
Prepare
Example Host
root@localhost
information_schema
test_db_1
test_db_2
test_db_3
Query
Message
Info
Table Data
History

레이아웃


메뉴
툴바
연결탭패널
연결탭
쿼리탭패널
쿼리탭
쿼리에디터
정보탭패널
쿼리실행
Result 탭
Message 탭
Table Data 탭
Info 탭
History 탭
Result 탭 1
Result 탭 2
Result 탭 n
File
Edit
Favorites
Database
Table
Other
Tools
Quick Cmd
Procs
Vars
Status
Tokenize
Prepare
Example Host
root@localhost
information_schema
test_db_1
Tables
aa
rs_connect
rs_connect_camp
rs_paper
rs_paper_target
Views
Events
test_db_2
test_db_3
Query
select 1;
Messages
Table Data
Info
Result
Total 1
Rows

쿼리 실행

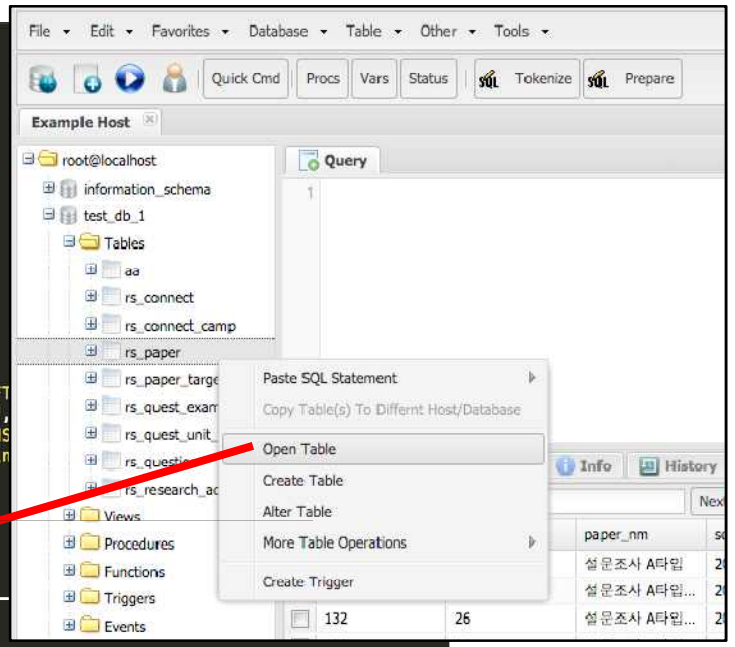
쿼리 실행 구조


쿼리에디터(코드미러)
쿼리 입력
실행
쿼리 분석
쿼리분리
토큰화
예약어
SELECT
함수
COUNT(1)
예약어
FROM
테이블
MEMBER
tunneling.php
쿼리실행
쿼리전송(JSONP)
JSON 응답
UI
결과 출력

UI에 연결된 쿼리 처리 방법

Ext.String.format을 이용한 쿼리 치환

```
Ext.define('Planche.engine.MySQL', function(){

    var queries = {
        SELECT_USER : 'SELECT * FROM `mysql`.`user`',
        SHOW_FULL_FIELDS : 'SHOW FULL FIELDS FROM `{0}`.`{1}`',
        SHOW_ADVANCED_PROPERTIES : 'SHOW TABLE STATUS FROM `{0}` LIKE "{1}"',
        SHOW_DATABASE : 'show databases',
        OPEN_TABLE : 'SELECT * FROM `{0}`.`{1}`',
        SHOW_PROCEDURES : 'SHOW PROCEDURE STATUS WHERE DB = "{0}"',
        SHOW_FUNCTIONS : 'SHOW FUNCTION STATUS WHERE DB = "{0}"',
        SHOW_TRIGGERS : 'SHOW TRIGGERS FROM `{0}`',
        SHOW_VARIABLES : 'SHOW VARIABLES',
        SHOW_STATUS : 'SHOW STATUS',
        SHOW_GLOBAL_STATUS : 'SHOW GLOBAL STATUS',
        SHOW_SESSION_STATUS : 'SHOW SESSION STATUS'
    };

    var joins           = ['NATURAL JOIN', 'INNER JOIN', 'LFET INNER JOIN', 'RIGHT INNER JOIN', 'LEFT
    var functions       = ['ABS', 'ACOS', 'ADDDATE', 'ADDTIME', 'AES_DECRYPT', 'AES_ENCRYPT', 'AREA',
    var reserved_words  = ['ACCESSIBLE', 'ADD', 'ALL', 'ALTER', 'ANALYZE', 'AND', 'AS', 'ASC', 'ASENS
    var data_types      = ['tinyint', 'int', 'varchar', 'float', 'double', 'timestamp', 'bit', 'bigin

    var regexpLimit     = "LIMIT\\s+[0-9]+((\\s+?,|,)?(\\s+)?[0-9]+)";

    return {
        singleton: true,
        constructor: function(config) {

            this.callParent(arguments);
        },
        getQuery : function(query){

            var args = Ext.Array.slice(arguments, 1);
            args.unshift(queries[query]);
            return Ext.String.format.apply(this, args);
        },
        getDataTypes : function(){

            return data_types;
        },
        getDataTypesToJSON : function(){

            var json = [];
            Ext.Array.each(data_types, function(type, idx){

                json.push([type, type]);
            })

            return json;
```




```
openTable : function(node){

    var tab = this.getActiveTableDataTab();
        db = this.getParentNode(node),

        //파서 생성
        parser = Ext.create('Planche.lib.QueryParser', this.getEngine()),

        //선택된 MySQL 엔진에서 쿼리 가져옴
        query = this.getEngine().getQuery('OPEN_TABLE', db, node.data.text),

        //쿼리 파싱
        queries = parser.parse(query);


        query = queries[0];

    this.tunneling({
        db : db,
        query : query.getSQL(),
```

실행되는 케이스

▶ 커서가 위치해 있는 곳에서 바로 실행

Query

```
1  select 1;
2
3  select 2;
4
5  select 3;
6
7  select 4;
8
9  select 5;
10
```

▶ 선택되어 있는 쿼리 실행

Query

```
1  select 1;
2
3  select 2;
4
5  select 3;
6
7  select 4;
8
9  select 5;
10
```

▶ 다중 선택(하나는 잘못 선택) 쿼리 실행

Query

```
1  select 1;
2
3  select 2;
4
5  select 3;
6
7  select 4;
8
9  select 5;
10
```

Messages | Table Data | Info | History | Result

ect 2

▶ You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'ect 2' at line 1

Delimiter

예) 프로시져 생성 쿼리 – delimiter에 의한 쿼리구분자 변경 (; => $$)

```
1  DELIMITER $$
2
3  USE `test_db_1`$$
4
5  DROP PROCEDURE IF EXISTS `proc3`$$
6
7  CREATE DEFINER=`websql`@`localhost` PROCEDURE `proc3`()
8  BEGIN
9
10   SELECT`uniq_id`,`camp_id`,`freq`,`uid`
11   FROM `test_db_1`.`rs_connect_camp`;
12
13 END$$
14
15 DELIMITER;
16
17 SELECT 1;
18
```

Delimiter

변경된 쿼리 구분자를 파악하여 제대로 완성된 쿼리만 보내야 한다.

```
1  DELIMITER $$
2
3  USE `test_db_1`$$
4
5  DROP PROCEDURE IF EXISTS `proc3`$$
6
7  CREATE DEFINER=`websql`@`localhost` PROCEDURE `proc3`()
8  BEGIN
9
10   SELECT`uniq_id`,`camp_id`,`freq`,`uid`
11   FROM `test_db_1`.`rs_connect_camp`;
12
13 END$$
14
15 DELIMITER;
16
17 SELECT 1;
18
```

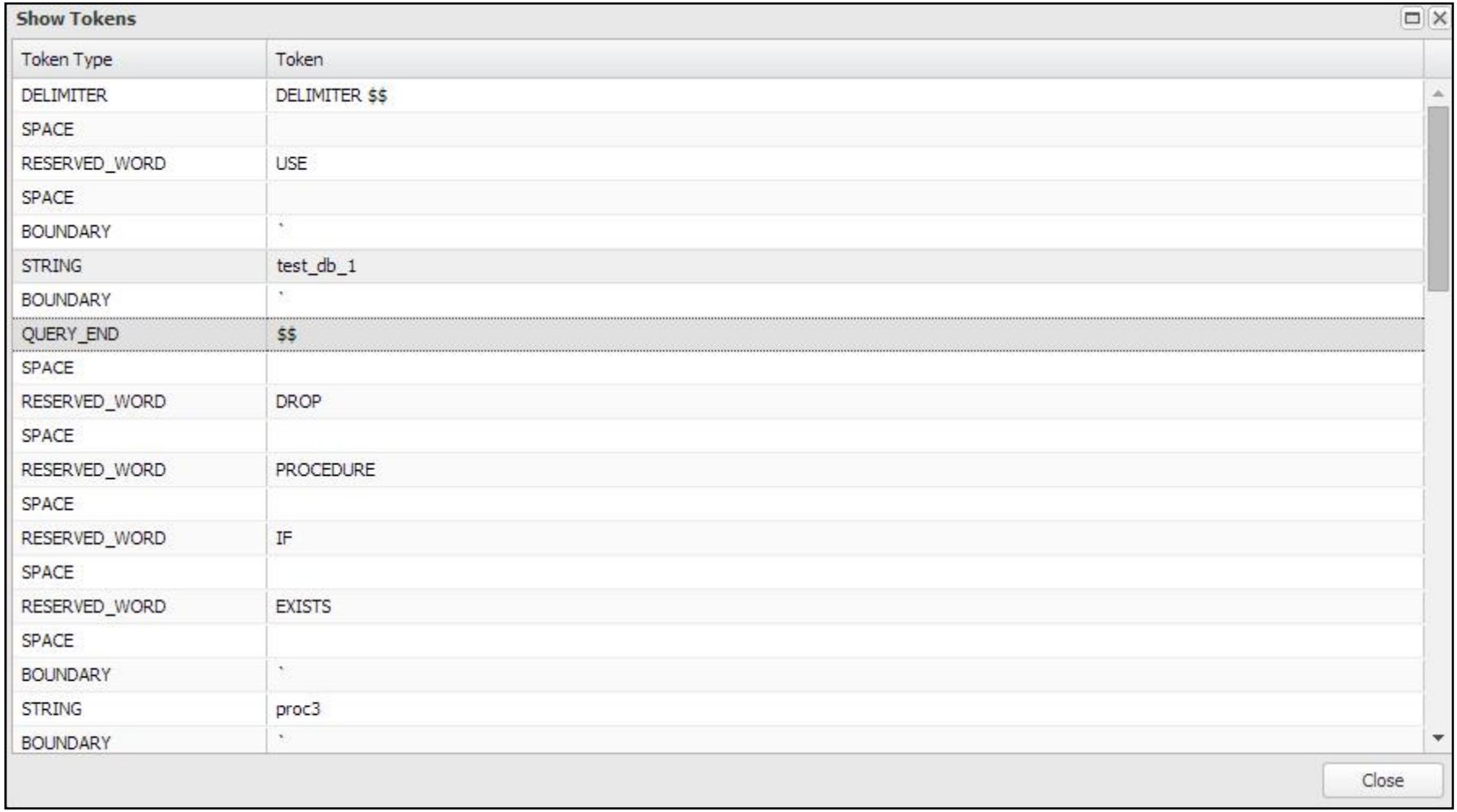
Concept

쿼리배열에 저장된 쿼리를 하나씩 꺼내며 재귀한다.

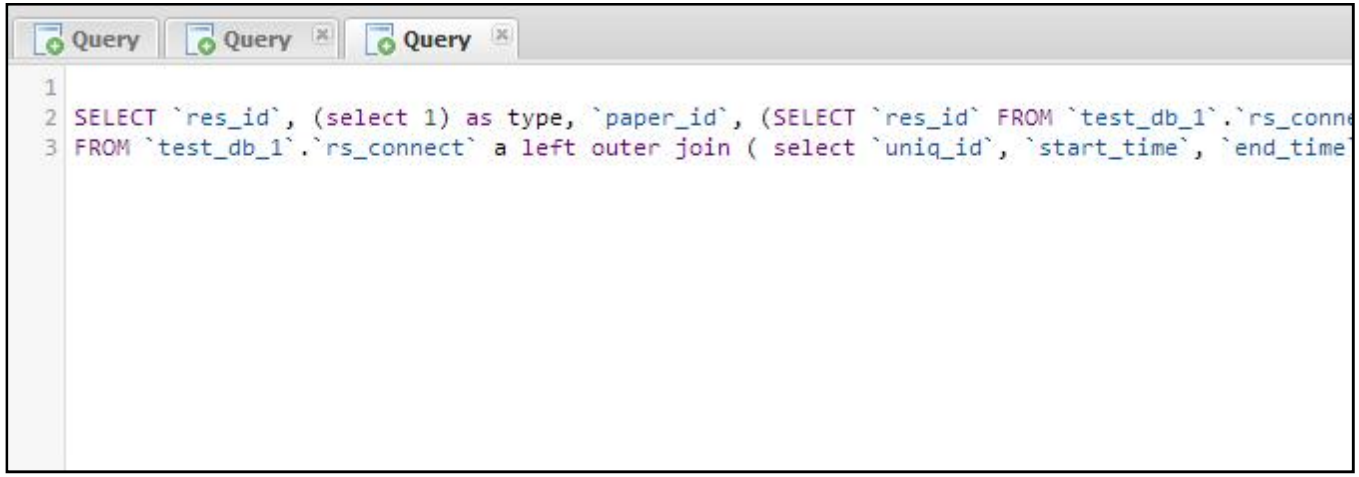
```
executeQuery : function(){

    var queries = this.getParsedQuery();

    var tunneling;
    var messages = [];
    (tunneling = Ext.Function.bind(function(){

        var query = queries.shift();

        if(query) {

            this.tunneling({
                db : db,
                query : query.getSQL(),
                success : function(config, response){

                    if(response.is_result_query == true){

                        var grid = this.initQueryResult({
                            icon    : 'images/icon_table.png',
                            closable : true,
                            title : 'Result'
                        }, db, query, response),
                    }
                    else {

                        var msg = response.affected_rows+' row(s) affected<br/><br/>';
                        msg += 'Execution Time : '+response.exec_time+'<br/>';
                        msg += 'Transfer Time  : '+response.transfer_time+'<br/>';
                        msg += 'Total Time     : '+response.total_time;
                        messages.push(query.getSQL()+'<br/><br/>'+msg);
                    }
                    this.getActiveMainTab().setLoading(false);
                    tunneling();
                },
                failure : function(config, response){

                    messages.push(query.getSQL()+'<span class=\'query_err\'>▶ '+response.message+'</span>');
                    this.getActiveMainTab().setLoading(false);
                    tunneling();
                }
            })
        }
        else {

            this.afterExecuteQuery(messages);
        }
```

Tokenizing

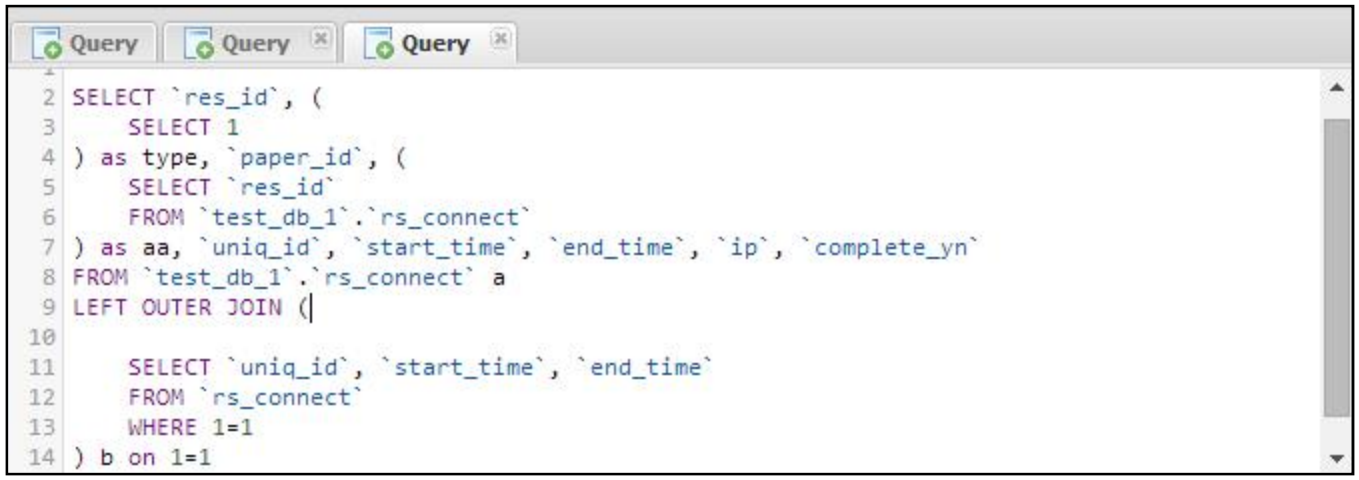
쿼리 분석전 실행할 문자열(쿼리)을 토큰화하여 토큰별 정보를 구축한다.



(정렬 전)

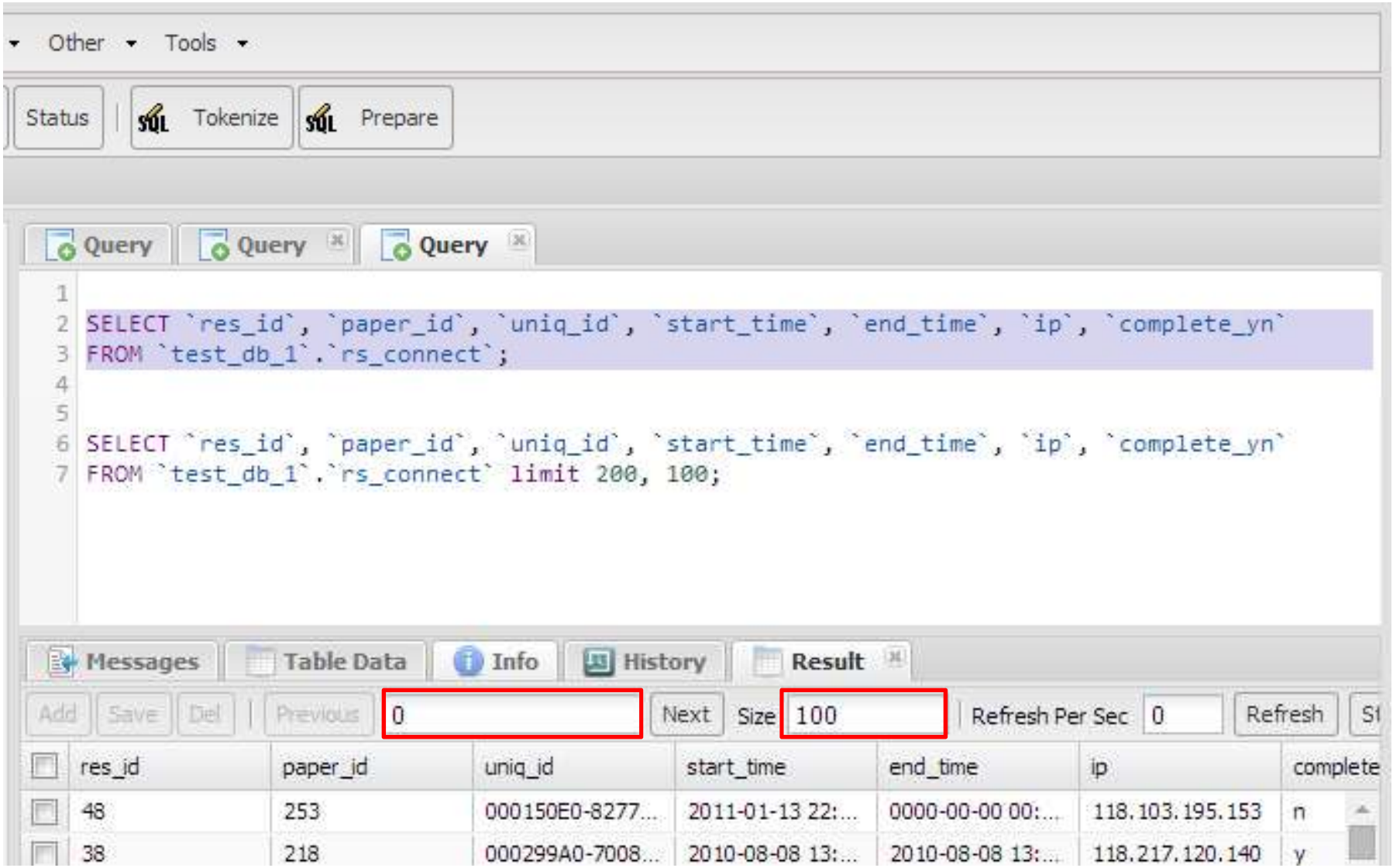


(정렬 후) 구축된 토큰별 정보로 적당한 곳에서 들여쓰기와 개행을 해준다.



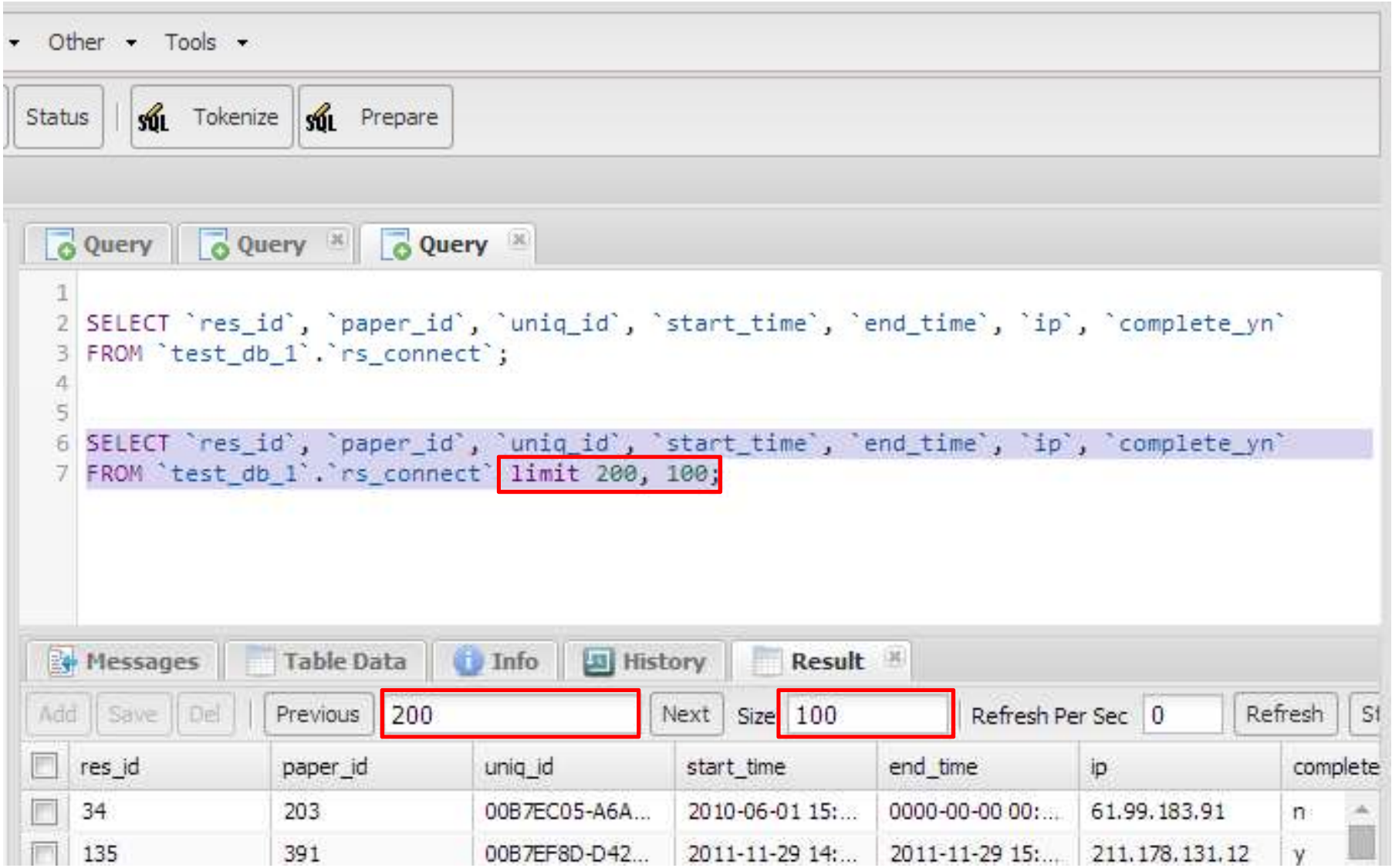
LIMIT

LIMIT 절이 없는 경우



쿼리 파싱 – LIMIT 절 추가

LIMIT 절이 있는 경우



Demo

http://github.com/jeongjuwon/planche

Q & A

https://www.facebook.com/groups/korea.sencha/