Scalable Data Services with mongoDB

High Performance
High Availability

for...

Managers

Architects

Developers (Web)

Admins

Data ...

http://www.flickr.com/photos/annarbor/4350618884/in/set-72157623414542180/

Data Services ...

services data for...

Dynamic Web Sites

Mobile Applications

data services are driven by...

Databases (on)

Remote Servers

mongo ...

mongo what?

```
{
  “type“:  “db“,
  “name“:  “mongo“
}
```

mongo key features?

```
{
  “tables“:  false,
  “sql“:  false,
  “documents“:  true,
  “json“:  true,
  “buzzword_bingo_compatible“: true
}
```

more mongo features?

```
{
  “implemented_in“:  “C++“,
  “has_replication“:  true,
  “has_sharding“:  true,
  “stores_files“:  true,
  “commercial_support“:  true
}
```

still more mongo features?

```
{
  “documentation“:  “excellent“,
  “speed“:  “blazing_fast“,
  “level_to_start_with“:  “low“,
  “stores_files“:  true,
  “location_search“:  true
}
```

mongo suited for?

```
{
  “reads“:  “a great many“
  “writes“:  “just a few“
}
```

mongo not suited for?

```
{
  “accuracy“:  “must have“,
  “consistency“:  “must have“,
  “internal_references“: “many“
}
```

language access?

```
[
  “php“,
  “ruby“,
  “java“,
  “python“
]
```

more language access?

```
[
  “perl“,
  “javascript“,
  “scala“,
  “erlang“
]
```

still more language access?

“tl;dr“

license?

```
{
  “core“:  “AGPL“
  “drivers“:  “Apache“
}
```

No SQL ...

tables?

PERSONS:
id, firstname, lastname
1, "Christian", "Hartmann"

ADDRESSES:
street, zip, city, person_id
"Katzlerstraße 9", 10829, "Berlin", 1

URLS:
url, person_id
“http://hartmann-it-design.de“, 1
“mailto:christian@hartmann-it-design.de", 1

documents!

```
{
  "firstname": "Christian",  "lastname": "Hartmann",
  "postal": {
    "street": "Katzlerstraße 9",   "zip": 10829, "city": "Berlin"
  },
  "urls": [
    "http://hartmann-it-design.de",
    "mailto:christian@hartmann-it-design.de",
  ],
  "rev": 1.0
}
```

storing files ...

storing files in database?

yes!

small files

and large files

and even huge files

and even across multiple servers

“GridFS“

GridFS?

```
{
  “name“:  “GridFS“,
  “type“:  “collection“
}
```

```
> (no)    mount --type gridfs
```

Scalability …

need for scalability

mass of data

high performance

scalability factors

amount of data

performance

replication (HA)

server 1:	A-X
server 2:	A-X
server 3:	A-X

automatic failover

automatic desaster recovering

with or without master

multi data center

sharding (HP)

server 1:	A-F
server 2:	H-K
server 3:	L-P
server 4:	Q-T
server 5:	U-X

for (SQL) Developers …

SQL?

no!

sorry

no sql here

sql mappings...

ALTER TABLE users ADD ...

oops .. n/a (there is no alter table)

more sql mappings...

```
SELECT * FROM users WHERE age=33
```

```
db.users.find({age:33})
```

still more sql mappings...

```
SELECT order_id FROM orders o,
order_line_items li WHERE
li.order_id=o.order_id AND li.sku=12345
```

```
db.orders.find({"items.sku":12345},{_id:1})
```

syntactic shugar ...

```
db.users.find( { name: /^[JY]oe|^[JY]ö/i } )
```

more syntactic shugar ...

```
db.users.find( {age: {'$exists':
false}} ).count()
```

can't stop with syntactic shugar

...

```
db.users.find( { homes : { $size: 2 } } )
```

still more - it's annoying i know ...

```
db.meta.find( { name: { $notin :
[ “foo“,“bar“ ] }} )
```

map & reduce?

available

no need for (if distributed storage reasoned) for Admins ...

replication (master slave)

server 1:   A-X   (master)

server 2:   A-X   (slave)

[/etc/mongodb.conf]

```
master = true
# or
slave = true
source = master.example.com
```

replication (replica set)

server 1:   (primary)

server 2:   (secondary)

server 3:   (recovering)

replSet = my_first_repl_set

replication (replica set) cont...

```
> var config = {
    _id : “my_first_repl_set“,
    members : [
        {_id : 0, host : “server1“ },
        {_id : 1, host : “server2“ },
        {_id : 1, host : “server3“ }
    ]
}
> rs.initiate(config)
```

sharding (components)

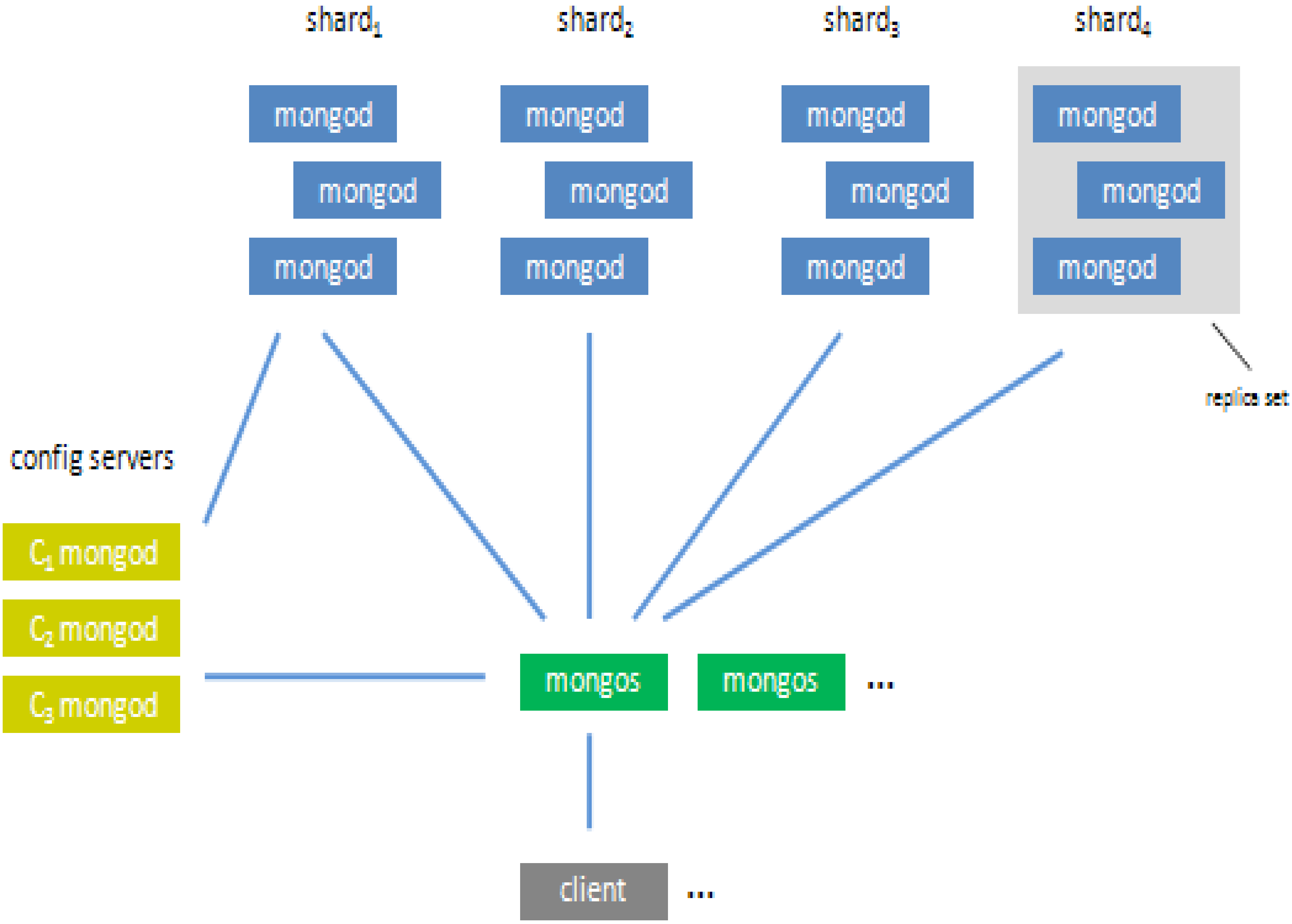
shards (preferable replicated)

config servers (preferable replicated)

routing servers

sharding key[s]

chunks


shard$_1$
shard$_2$
shard$_3$
shard$_4$
mongod
mongod
mongod
mongod
mongod
mongod
mongod
mongod
mongod
mongod
mongod
mongod
replica set
config servers
C$_1$ mongod
C$_2$ mongod
C$_3$ mongod
mongos
mongos
...
client
...

sharding (simple config)

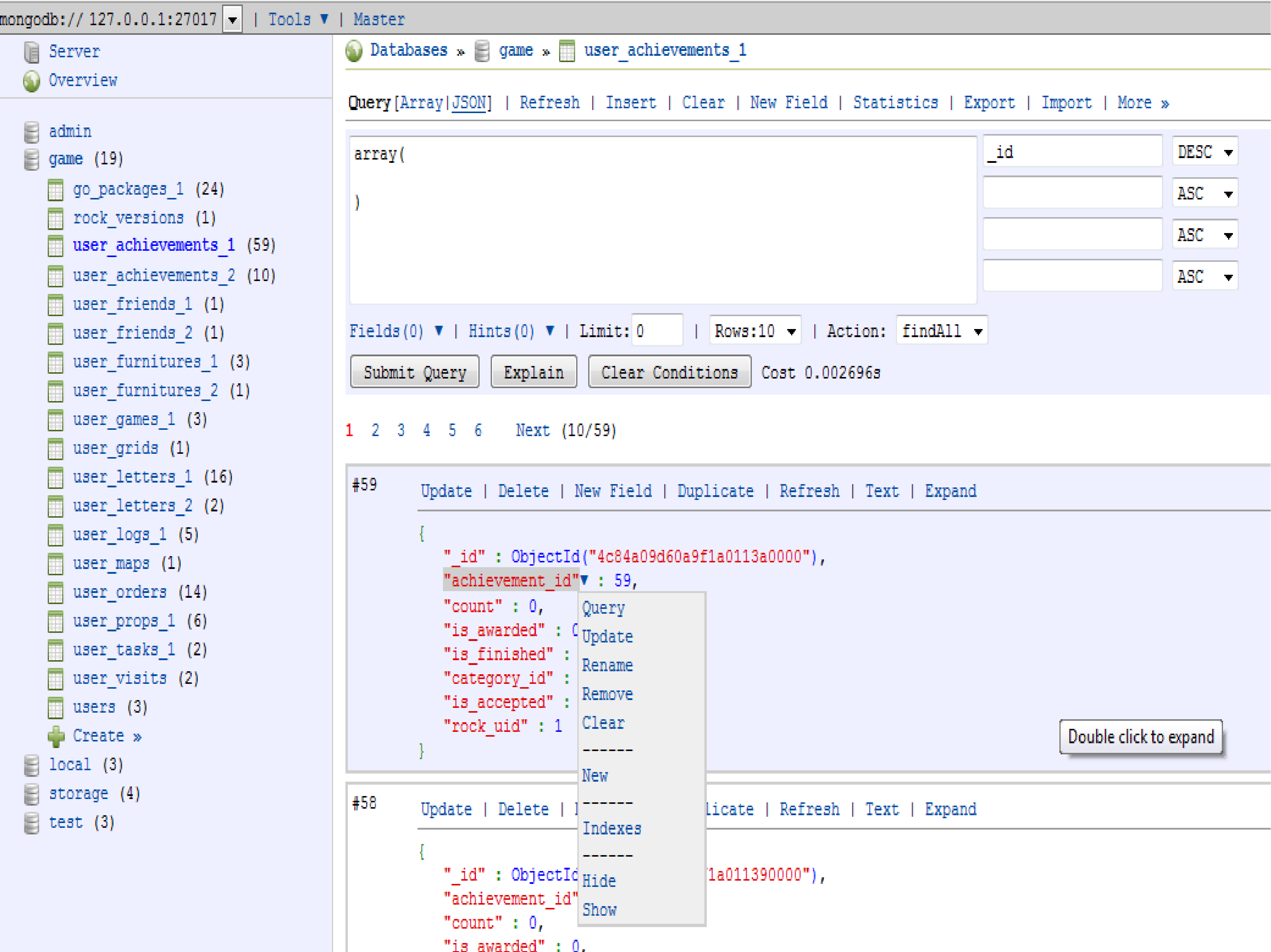
```
> db.runCommand( { “enablesharding“ :
"test" } )


> db.runCommand( {
   “shardcollection“ : "test.users",
   “key“ : { “name“ : 1 }
   } )
```

sharding (chunk config)

```
{
  "ns" : "test.users",
  "min" : { "name" : "A" },
  "max" : { "name" : "F" },
  "shard" : "shard1"
}
```

GUI ? ...


mongodb:// 127.0.0.1:27017
| Tools ▼ | Master
Server
Overview
Databases » game » user_achievements_1
Query [Array|JSON] | Refresh | Insert | Clear | New Field | Statistics | Export | Import | More »
Fields(0) ▼ | Hints(0) ▼ | Limit:
| Rows:10 | Action: findAll
Submit Query
Explain
Clear Conditions
1 2 3 4 5 6 Next (10/59)
Update | Delete | New Field | Duplicate | Refresh | Text | Expand
Query
Update
Rename
Remove
Clear

New

Indexes

Hide
Show
Double click to expand
Update | Delete | I
licate | Refresh | Text | Expand
Create »

admin
game (19)
go_packages_1 (24)
rock_versions (1)
user_achievements_1 (59)
user_achievements_2 (10)
user_friends_1 (1)
user_friends_2 (1)
user_furnitures_1 (3)
user_furnitures_2 (1)
user_games_1 (3)
user_grids (1)
user_letters_1 (16)
user_letters_2 (2)
user_logs_1 (5)
user_maps (1)
user_orders (14)
user_props_1 (6)
user_tasks_1 (2)
user_visits (2)
users (3)
local (3)
storage (4)
test (3)

```
array(

)
```

_id	DESC
	ASC
	ASC
	ASC

Limit: 0

Cost 0.002696s

#59

```
{
   "_id" : ObjectId("4c84a09d60a9f1a0113a0000"),
   "achievement_id" : 59,
   "count" : 0,
   "is_awarded" : 0
   "is_finished" :
   "category_id" :
   "is_accepted" :
   "rock_uid" : 1
}
```

#58

```
{
   "_id" : ObjectIc                 1a011390000"),
   "achievement_id"
   "count" : 0,
   "is_awarded" : 0,
```