目前系统结构：

Filters: whoisFilter errorFilter

参数处理：提取用户角色、接收的返回值类型、查询类型、查询参数、参数解码/idn转换

查询逻辑：

验证参数合法性

查主表（递归）

对于每个result：

根据角色取有权限的字段

处理不同类型的字段：

array字段

自定义字段

join字段：

根据查询类型重新设置字段名

join递归查询

普通字段，根据format设置结果字段名称

问题：

1. 所有类型的查询逻辑都在一起，掺杂不同类型、字段的特殊处理，使用长函数递归，修改维护困难
2. 查询参数处理、类型处理、查询逻辑、视图处理都在一起，维护困难
3. 关联查询sql使用：

多sql查询：a, a\_b+b,b\_c,c

每次查询拼sql

1. 人工执行测试，容易遗漏
2. 动态字段处理

**Filters**

EncodeFilter

RateLimitsFilter: decline rate limits. Todo：Local map -> remote cache

ErrorFilter: handle wrong query type

**RequestParser**

Parse request to:

queryType:domain/entity/nameserver/ip/as/event/ds/key

queryString

isFuzzyQuery

format

Validate queryString by query type

**QueryEngine**

**SearchQuery Executor**

*Domain core data: from dnr domain and rir domain*

*Entity core data: from dnr entity and rir entity*

*Ns core data: from nameserver*

*TODO:*

*IndexUpdater*

*Solr data shading: seprate core*

*Solr failover: solr cloud*

**CacheQuery Executor**

Query entity from cache (except ip/as), cache init after app startup

Cache value format: json – serialized from map of entity and all joined entities.

Cache key and value content:

domain:handle:{domain handle} : get domain by handle

domain:ldhName:{domain ldhName} : get domain by name

entity:handle:{entity ldhName} : get all kinds of entity by handle

registrar:name:{registrar name} : get registrar by name

variants:handle:{ variants handle} : get variants by handle

secureDns:handle:{securedns handle} : get secureDns by handle

dnskey:handle:{dnskey handle} : get dnskey by handle

ds:handle:{ds handle} : get ds by handle

notice:handle:{notice handle} : get notice by handle

remark:handle:{notice handle} : get remark by handle

**ip and as** *: range query, query from db*

**error** *: now is query from db by error code -> cache in local memory*

*TODO：*

*Redis data shading: twemproxy or redis cluster*

*Redis failover: keepalive or redis cluster*

**DbQuery Executor**

Query from db.

BaseDao members:

String : getResultKey() - result map key

List<String> : getFields()

Map<String,Object>:query() :

Query results

Handle column of every result:

preHandle()

handle() – handleJoinedEntity()

postHandle()

Sql query use preparedStatment.

**PermissionController**

Remove unAuthed objects and columns from full result.

QueryPermission:

List<String> getAuthedColumns(role)

**ResponseWriter**

Write response according to accept format.

**Test**

Integration test use python

TODO: continued test

**TODO：**

**Dynamic Column Manager**