Описание спецификации СМЭВ QL

Содержание

[1. Цели разработки спецификации СМЭВ-QL 3](#_Toc147135975)

[1.1 Эффект для потребителя и МЦ 3](#_Toc147135976)

[1.2 Эффект для поставщика 3](#_Toc147135977)

[1.3 Эффект для СМЭВ и Минцифры 3](#_Toc147135978)

[1.4 Концепция 3](#_Toc147135979)

[1.5 Соглашения 3](#_Toc147135980)

[1.6 Ограничения 4](#_Toc147135981)

[2. Язык и синтаксис 4](#_Toc147135982)

[2.1 Моделирование 4](#_Toc147135983)

[2.2 Запросы и ответы 4](#_Toc147135984)

[3. Типизация 4](#_Toc147135985)

[3.1 Типы данных в модели и приведение типов 4](#_Toc147135986)

[4. Валидация 5](#_Toc147135987)

[4.1. Схема данных 5](#_Toc147135988)

[4.2. Запросы 6](#_Toc147135989)

[5. Моделирование данных 6](#_Toc147135990)

[5.1. Описание и наименование ресурсов 6](#_Toc147135991)

[5.2. Состав ресурса 7](#_Toc147135992)

[5.3 Пример описания базовой модели 12](#_Toc147135993)

[6. Запросы 13](#_Toc147135994)

[6.1. Типы запросов 13](#_Toc147135995)

[7. Исполнение запросов 17](#_Toc147135996)

[7.1 Обработка модели 17](#_Toc147135997)

[7.2 Этапы обработки запроса 17](#_Toc147135998)

[7.3 Извлечение данных их источников 18](#_Toc147135999)

[8. Ответы 19](#_Toc147136000)

[8.1 Ошибки 20](#_Toc147136001)

# 1. Цели разработки спецификации СМЭВ QL

Основной целью разработки спецификации СМЭВ QL является обеспечение стандартизированного обмена данными в СМЭВ с использованием документных моделей витрин данных. Реализация протокола СМЭВ QL преследует достижение эффектов для участников и оператора СМЭВ.

## 1.1 Эффект для потребителя и МЦ

– Высокая скорость предоставления ответа за счет:

– сокращения количества запросов;

– параллелизма извлечения данных;

– строгого ограничения протокола;

– Простота составления запроса к Витринам данных

– Получение готового объекта для использования

– Гарантия актуальности данных на определенный момент времени

## 1.2 Эффект для поставщика

– Защита витрины от неоптимальных запросов

– Повышение скорости вычисления за счет максимальной оптимизации запроса

– Возможность не изменяя запросы:

– оптимизировать структуру хранения;

– заменять движок СУБД;

– расширять состав атрибутов объектов хранения;

– Контроль потребителей собственных данных

– Контроль целей потребления собственных данных

## 1.3 Эффект для СМЭВ и Минцифры

– Возможность управлять доступом к объектам и атрибутам;

– Высокая связанность моделей данных Витрин данных НСУД;

– Сокращение объема ответов;

– Повышение скорости развития услуг ЕПГУ, расширения модели НСУД;

– Сокращение количества передаваемых запросов.

## 1.4 Концепция

СМЭВ QL представляет из себя семантический протокол, реализованный на открытых сериализаторах. Транспортным протоколом данных для СМЭВ QL является HTTP. Сериализация запросов, моделей и ответов происходит либо в YAML, либо в JSON, в зависимости от требований протокола.

## 1.5 Соглашения

Использование протокола определяет ряд основных соглашений для его пользователей, которых стоит придерживаться в случае неоднозначности трактования требований протокола или их реализаций:

1. Объекты, разработанные с использованием СМЭВ QL должны легко читаться человеком.

2. Именование моделей и атрибутов должно максимально отражать их содержание.

3. Стиль\_разделения\_имен – under\_score.

4. «Примеры превосходят описание» – в спецификации при неоднозначном трактовании требований необходимо ориентироваться на приведенные примеры, как фактическую истину.

5. Спецификация должна реализовываться в концепции минимальной достаточности – при возможности удалить часть текста спецификации, без потери качества восприятия, необходимо это сделать.

## 1.6 Ограничения

Спецификация не накладывает ограничений на состав данных при соблюдении правил моделирования и приведения типов.

# 2. Язык и синтаксис

## 2.1 Моделирование

Для моделирвоания документного слоя данных в спецификации выбран язык разметки YAML.

## 2.2 Запросы и ответы

Для написания запросов, а также в качестве сериализатора ответов, спецификация определяет использование JSON.

# 3. Типизация

Фактические типы данных наследуют типы данных JSON (включая NULL):

– string;

– number;

– object;

– array;

– boolean;

– null.

## 3.1 Типы данных в модели и приведение типов

В описании модели допускается указание фактического типа данных атрибута ресурса вторым элементом массива type. Указание является опциональным, по умолчанию подразумевается неограниченный STRING. Пример из описания модели:

fields:  
 id:  
 name: Идентификатор записи  
 type:  
 - number  
 - SHORT  
 length: 20  
 nullable: not NULL  
 key: PRIMARY

В качестве второго уточняющего типа следует применять типы НСУД:

– STRING;

– DOUBLE;

– FLOAT;

– BOOLEAN;

– BYTE (не поддерживается на витрине);

– BINARY;

– BIG\_DECIMAL (не поддерживается на витрине);

– LONG;

– INTEGER;

– SHORT;

– DATE;

– TIME;

– TIMESTAMP.

# 4. Валидация

## 4.1. Схема данных

Гарантируется синтаксическая правильность и соответствие JSON-схеме. Схема данных, не прошедшая валидацию, не должна быть зарегистрирована. Валидация должна быть выполнена на этапе регистрации схемы данных.

Для описания ограничений данных можно применять шаблоны из регулярных выражений.

### 4.1.1. Атрибуты модели

Каждый ресурс схемы должен в описании иметь разделы, представленные в Таблице 1.

Таблица 1 - Атрибуты модели

| Наименование поля | Обязательность | Назначение |
| --- | --- | --- |
| name | да | Наименование |
| description | да | Описание ресурса |
| fields | да | Описание полей |

### 4.1.2. Атрибуты ресурса

Каждый ресурс схемы должен иметь указание на обязательность, как представлено в Таблице 2.

Таблица 2 - Обязательность полей

| Наименование поля | Обязательность | Назначение |
| --- | --- | --- |
| id | да | Идентификатор записи |
| created\_at | да | Время создания записи |
| updated\_at | да | Время последнего изменения записи |

### 4.1.3. Ссылочная целостность

При регистрации ресурсов должна проверяться правильность и достаточность указанных ссылок в разделах connections и restrictions

## 4.2. Запросы

### 4.2.1. Запрос данных

Проверяется синтаксическая правильность запроса и гарантирует отсутствие ошибок в контексте заданной схемы данных. Запросы, не прошедшие валидацию, не могут быть выполненными системой. Валидация выполняется непосредственно перед выполнением запроса.

Описание структуры запроса приведено в разделе [6. Запросы](#_6._Запросы)

Запрос содержит описание ресурсов и связей. Запрос проверяется соответствие схеме данных.

#### 4.2.1.1. Обязательность полей и блоков

Таблица 2 - Обязательность полей и блоков

| Наименование поля | Обязательность | Назначение |
| --- | --- | --- |
| для указания ресурса |  |  |
| conditions | нет | Условия запроса |
| attributes | да | Запрашиваемые атрибуты ресурса |
| технический блок (credentials) |  |  |
| system | да | Описание потребителя |
| - mnemonic | да | UUID потребителя |
| - instance\_id | нет | UUID источника |
| - user\_id | нет | UUID пользователя |
| request | да | Параметры запроса |
| - id | да | Идентификатор запроса |
| - sub\_id | нет | Идентификатор подзапроса |
| - purpose\_id | да | Идентификатор цели запроса |
| - audit | нет | Параметры аудита |
| signature | нет | Электронная подпись |
| - digest | да\* | Хеш |
| - signature | да\* | Подпись |

# 5. Моделирование данных

Язык описания моделей СМЭВ QL — YAML. Во избежание разночтений из двух допустимых расширений файлов использоваться должен только model.yaml. Основные причины выбора языка разметки:

1. Легко читается человеком;

2. Легкий и быстрый парсинг;

3. Не имеет дополнительных синтаксических усложнений;

4. Визуально подходит для отображения свойств объектов модели;

5. Удачно реализовано наследование;

## 5.1. Описание и наименование ресурсов

Ресурсы описываются массивом блоков, имя блока — название ресурса в единственном числе на английском (либо транслите), пример приведен в Таблице 4.

Таблица 4 – Описание и наименование ресурсов

| таблица | ресурс |
| --- | --- |
| vehicles | vehicle |
| people | person |

Допустимо применять названия, как представлено в Таблице 5.

Таблица 5 – Пример допустимых названий

| таблица | таблица |
| --- | --- |
| samolety | samolet |

## 5.2. Состав ресурса

### 5.2.1. Блоки

Обязательные блоки ресурса:

1. name — имя на русском языке.

2. fields — массив объектов, описывающих атрибуты ресурса.

3. sources — источник данных модели.

Опциональные блоки ресурса:

1. description — краткое описание ресурса на русском языке.

2. connections — блок описания связей с другими моделями.

3. restrictions — ограничения связей и полей модели.

### 5.2.2. Блок name

В блоке name указывается наименование ресурса на русском языке.

### 5.2.3. Блок description

В блоке description указывается краткое описание ресурса модели.

### 5.2.4 Блок fields

В блоке fields перечисляются поля описываемой модели, их типы, ограничения и адреса. Рекомендуемые поля ресурса: (*&default\_fields*)

1. id

2. created\_at

3. updated\_at

Каждый атрибут должен быть описан исходя из своего содержания:

fields:   
 <<: \*default\_fields  
 id:   
 <<: \*default\_field  
 name: идентификатор записи  
 type: # тип поля  
 - number # тип JSON  
 - bigint # логический тип  
 length: 20 # длина поля, по-умолчанию 0 — не ограничена  
 nullable: not NULL # признак NULL  
 key: PRIMARY # тип ключа PRIMARY/INDEX/UNIQUE/NONE

Существуют пресеты типов по-умолчанию: Базовый *&default\_field*

default\_field: &string  
 name: строка  
 type:   
 - string  
 - STRING   
 length: 0  
 nullable: NULL   
 key: NULL  
 source: &default\_source

Строка *&ds*

ds: &string  
 name: строка  
 type:   
 - string  
 - STRING   
 length: 0  
 nullable: NULL   
 key: NULL

Целое число *&dn*

dn: &number  
 name: число  
 type:   
 - number  
 - bigint   
 length: 0  
 nullable: NULL   
 key: NULL

Пример использования пресетов:

fields:   
 <<: \*default\_fields  
 year:   
 <<: \*dn  
 name: Год рождения   
 last\_name:   
 <<: \*ds  
 name: Фамилия  
 nullable: not NULL

#### 5.2.4.1 Специальное извлечение extract

В редких случаях возможно явное переопределение способа извлечения конкретного поля из источника. Для этого необходимо использовать атрибут extract в котором явно указать способ извлечения и источник:

fields:   
 <<: \*default\_fields  
 last\_name:   
 <<: \*ds  
 extract:   
 source: \*default\_source  
 request: 'SELECT HashBytes('MD5', last\_name) from users'

В этом примере поле с фамилией будет всегда извелкаться преобразуясь в хэш, в каком бы запросе оно не участвовало.

#### 5.2.4.2 Специальное преобразование transform

Для извлечения некоторых полей может потребоваться обязательное предварительное преобразование перед запросом. Для этого необходимо использовать специальный атрибут transform:

fields:  
 <<: \*default\_fields  
 last\_name:   
 <<: \*ds  
 transform:  
 before: HashBytes('MD5', last\_name)  
 after: ^.{0,8}

В подблоке before указывается преобразование *запроса* ДО извлечения данных (в примере — запрашиваем хэш от значения), в блоке after регулярное выражение над результатом ПОСЛЕ извлечения (в примере — возвращаем только первые 8 символов).

#### 5.2.4.3 Ограничения на извлечения атрибутов guard

Для ограничения возможности получения некоторых атрибутов без предварительно предоставления их же (или дополнительных) значения извлекающим необходимо определить атрибуты-ограничители в блоке guard.

Пример:

fields:  
 <<: \*default\_fields  
 first\_name: \*ds  
 last\_name:  
 <<: \*ds  
 guard: [last\_name]  
 snils:  
 <<: \*ds  
 guard: [last\_name first\_name snils]

В примере извлечение first\_name не ограничивается. Для получения last\_name фамилия должна быть передана в блоке conditions, а для получения snils в conditions должны присутствовать фамилия, имя и сам СНИЛС.

### 5.2.5. Блок connections

В блоке connections указываются связанные ресурсы описываемой модели. Поддерживаемые блоки связи:

has\_many — владет множественными ресурсами.

belongs\_to — принадлежит ресурсу.

В блоках связей перечисляются ресурсы в единственном числе. Для каждого блока могут быть переопределены ключи по-умолчанию:

connections:  
 has\_many:  
 - connected\_resource\_name:  
 primary\_key: id  
 foreign\_key: connected\_resource\_name\_id  
 belongs\_to:  
 - connected\_resource\_name:  
 primary\_key: connected\_resource\_name\_id  
 foreign\_key: id

### 5.2.6. Блок conditions

В блоке conditions указываются дополнительные ограничения для атрибутов модели.

# Блок conditions описывает ограничения и разрешения на использование условий поиска  
conditions:  
 allowed: [id, name] # если заполнено, то поиск разрешен только по этим полям и полям с ключами  
 denied: [snils] # если заполнено, то поиск запрещен по этим ключам  
 always: # наличие условий в блоке always должно ко всем запросам ресурса добавлять эти условия, если указаны в запросе, то перетирать  
 - region: ["=", "77"]  
 - blocked: ["=", true]

### 5.2.7. Блок sources

Блок source должен описывать источники данных модели. Один из источников должен обязательно быть выбран «по-умолчанию», название такого источника default\_field.

Состав источника данных REST (Prostore):

sources:  
 prostore\_source:  
 type: rest  
 version: 1.0  
 adapter: prostore  
 protocol: http  
 host: localhost  
 port: 9090  
 path: api/v1/datamarts/query?format=json  
 template: '{ "query": "%{request}" }'  
 payload-path: result  
 headers:  
 - accept: application/json  
 - content-type: application/json

Состав источника данных СМЭВ QL:

sources:  
 smevql\_source:  
 type: rest  
 version: 1.0  
 adapter: smevql  
 protocol: http  
 host: localhost  
 port: 8989  
 path: data  
 headers:  
 - content-type: application/json  
 threads-count: 10  
 connection-timeout: 0

Состав источника данных DB (Postgres):

sources:  
 default\_source: &default\_source  
 host: localhost  
 port: 5432  
 driver: pg  
 schema: datamart  
 username: postgres  
 password: passw0rd  
 table: self  
 field: self

Предпочтительно использовать переменные окружения для конфигурирования источников данных.

Поля table и field по-умолчанию содержат значение self, оно должно восприниматься, как равенство названия ресурса и таблицы, а также названия атрибута модели и атрибута таблицы. Данные значения могут быть перезаписаны для каждого атрибута, либо заменены специальным запросом extract.

#### 5.2.7.1 Условия выбора источников

В блоке c указанием источника в модели допускается указание условий его выбора через блок conditions. Блок conditions содержит массив условий применения источника на основании значений полученных в запросе атрибутов:

extract:  
 source:  
 - name: prostore1  
 table: misdm.slots  
 conditions:  
 # попадание в промежуток  
 - range:  
 field: age  
 from: 0  
 to: 2  
 - eq:  
 field: color  
 not: "blue"  
 - name: prostore2  
 table: misdm.39slots  
 conditions:  
 # ограничения по (не)равенству  
 - eq:  
 field: resource\_id  
 is: 1  
 - name: prostore3  
 table: misdm.39slots  
 conditions:  
 # ограничения по наличию в источнике  
 - eq:  
 field: snils  
 extract:  
 source: redis  
 table: default\_table  
 key: resource\_hashed\_id  
 algorithm: md5  
 # select count(\*) > 0 from offices.offices where resource\_hashed\_id = ?  
 # параметр: md5(snils)  
 is: true  
 - name: prostore\_default  
 table: misdm.39slots  
 conditions:  
 - fallback: true

Все условия внутри источника действуют через логическое И (AND). Если запрос подходит под два и более источников, то данные извлекаются и объединяются из всех подходящих. Возможные алгоритмы шифрования: \* MD5 \* SHA1 \* SHA256 \* CRC32 ##

## 5.3 Пример описания базовой модели

resources:

- base\_model: &base\_model

name: Название ресурса

description: Описание ресурса

fields: &ref\_0

- id:

- name: Идентификатор записи

- type:

- number

- bigint

- length: 20

- nullable: not NULL

- key: PRIMARY

- created\_at:

- name: Время создания записи

- type:

- string

- timestamp

- length: 10

- updated\_at:

- name: Время последнего изменения записи

- type:

- string

- timestamp

- length: 10

connections: &ref\_1

has\_many:

- connected\_resource:

primary\_key: id

foreign\_key: connected\_resource\_id

belongs\_to:

- connected\_resource:

primary\_key: connected\_resource\_id

foreign\_key: id

restrictions: &ref\_2

personal\_data:

- code

- single\_name\_2:

<< : \*base\_model

name: Ресурс 2

description: Описание ресурса 2

fields: \*ref\_0

connections: \*ref\_1

restrictions: \*ref\_2

# 6. Запросы

## 6.1. Типы запросов

Сервер, поддерживающий работу по спецификации СЭМВ-QL должен предоставлять три адреса для трех типов запросов:

1. http://smev-ql-server:5811/data/ – адрес запросов данных.

2. http://smev-ql-server:5811/model/ – адрес запросов модели и связей.

3. http://smev-ql-server:5811/spec/ – адрес запросов спецификации конкретного инстанса СМЭВ QL сервера

### 6.2. Запросы данных

Запрос данных передается в теле POST-запроса JSON-объектом, следующей структуры:

– query

– resource\_name

– conditions

– attributes

– connected\_resources

– credentials

– system

– request

– signature

Пример запроса данных:

{  
 "query":{  
 "resource\_name":{  
 "conditions":{  
 "id":10,  
 "value":">5"  
 },  
 "attributes":[  
 "id",  
 "name",  
 "value"  
 ],  
 "connected\_resource\_name":{  
 "conditions":{  
   
 },  
 "attributes":[  
 "id",  
 "name"  
 ]  
 }  
 }  
 },  
 "credentials":{  
 "system":{  
 "mnemonic":"<guid>",  
 "instance\_id":"<guid>",  
 "user\_id":"<guid>"  
 },  
 "request":{  
 "id":"<guid>",  
 "sub\_id":"<guid>",  
 "name":"request\_name",  
 "purpose\_id":"<guid>",  
 "audit":false,  
 "audit\_id":"<guid>",  
 "audit\_token":"<token>"  
 },  
 "signature":{  
 "digest":"",  
 "signature":""  
 }  
 }  
}

#### 6.2.1 Условия фильтрации (И/AND)

Объединение условий происходит по логическому И (AND) Поддерживаются операции сравнения:

= (по умолчанию)

>

>=

<

<=

in

Условия сравнения применимы к численным типам, датам, временам и таймштампам. Варианты определения условий фильтрации:

По равенству (простое равенство)

{  
 "query": {  
 "office": {  
 "conditions": {  
 "phone": "(347) 246-53-00"  
 }  
 }  
 }  
}

На основе сравнения, краткая запись (Сравнение краткое)

{  
 "query": {  
 "office": {  
 "conditions": {  
 "area": [">","130"]  
 }  
 }  
 }  
}

На основе сравнения, полная запись (Сравнение полное)

{  
 "query": {  
 "office": {  
 "conditions": {  
 "area": {  
 "op": ">",  
 "value": "130"  
 }  
 }  
 }  
 }  
}

Комплексное условие

{  
 "query": {  
 "office": {  
 "conditions": {  
 "area": [">","130"],  
 "floor": ["in", [1, 2]]  
 }  
 }  
 }  
}

#### 6.2.2 Условия фильтрации (ИЛИ/OR)

В conditions СМЭВ QL запроса поддерживается возможность указывать логический ИЛИ через зарезервированное слово or. В блок or необходимо передать массив объектов, содержащих условия в свою очередь объединённые логическим И (AND). Все условия, находящиеся на одном уровне с or группируются через логическое И (AND), как при обычном СМЭВ QL Запросе.

Пример:

{  
 "query": {  
 "vehicle": {  
 "conditions": {  
 "lastname": "Иванов",  
 "middlename": "Иван",  
 "birthdate": "2021-11-29 00:00:00",  
 "or": [  
 {  
 "vin": "WDS123456789"  
 },  
 {  
 "vin2": "WDS123456789",  
 "model": "mercedes"  
 }  
 ],  
 "fetch": {  
 "order": [  
 [  
 "id",  
 "ASC"  
 ],  
 [  
 "number",  
 "DESC"  
 ]  
 ],  
 "page": [  
 2,  
 10  
 ]  
 }  
 }  
 }  
 }  
}

#### 6.2.4 Сортировка и пагинация

В блоке conditions опционально можно добавить блок fetch, в котором указывать условия сортировки и выбора страниц.

Пример блока:

{  
 "query": {  
 "vehicle": {  
 "conditions": {  
 "lastname": "Иванов",  
 "middlename": "Иван",  
 "birthdate": "2021-11-29 00:00:00",  
 "fetch": {  
 "order": [  
 [  
 "id",  
 "ASC"  
 ],  
 [  
 "number",  
 "DESC"  
 ]  
 ],  
 "page": [  
 2,  
 10  
 ]  
 }  
 }  
 }  
 }  
}

Если порядок сортировки не указан, то применяется ASC. Если не указаны страницы, то по умолчанию всегда устанавливается первая страница с установленным лимитом.

### 6.3. Запросы модели

Запрос схемы СМЭВ QL сервера выполняется методом GET без дополнительных параметров. Ответ возвращается в формате JSON, без указания источников.

### 6.4. Запросы спецификации

Запрос спецификации выполняется методом GET без дополнительных параметров. Ответ метода спецификации:

{  
 "spec":{  
 "server":{  
 "type":"СМЭВ QL Сервер",  
 "version":"mvp0",  
 "env":"production"  
 },  
 "protocol":{  
 "type":"СМЭВ QL",  
 "version":"0.1"  
 }  
 }  
}

# 7. Исполнение запросов

## 7.1 Обработка модели

Рекомендации при реализации СМЭВ QL сервера. При запуске СМЭВ QL сервера модель данных должна валидироваться на предмет формального соответствия спецификации и сериализации. После инициализации необходимо проверить доступность источников, перечисленных в блоке sources. Целесообразно держать дерево модели в оперативной памяти весь период работы СМЭВ QL сервера.

## 7.2 Этапы обработки запроса

Этапы обработки запроса СМЭВ QL могут отличаться в зависимости от конкретной реализации. Рекомендуемая последовательность шагов исполнения:

1. Валидация запроса на соответствие протоколу.

2. Валидация запроса на соответствие модели данных.

3. Валидация подписи запроса (при наличии).

4. Валидация блока credentials в случае наличия ограничений конкретной реализации.

5. Построение плана запросов согласно схеме.

6. Определение последовательности и параллелизма плана.

7. Выполнение плана.

8. Сериализация ответа.

9. Подпись ответа (при наличии).

## 7.3 Извлечение данных их источников

В текущей реализации спецификации подразумевается два способа извлечения данных сервером:

1. SQL-выражение.

2. REST-запрос.

3. СМЭВ QL запрос

### 7.3.1 SQL-выражения

Базовая логика извлечения атрибутов данных из СУБД, поддерживающих SQL должна строится по следующим шагам:

1. Атрибуты, относящиеся к одной таблице должны быть объединены в один *простой* SQL запрос с условиями из блока conditions:

– Для атрибутов, имеющих специальные трансформации before, подставляются именно они в данный запрос.

2. Если запрашивается вложенные объекты, то параллельным запросом должны быть извлечены ключи.

3. Связанные объекты после получения ключей извлекаются также параллельно с аналогичным объединением атрибутов в простой SQL-запрос. Помимо условий из conditions необходимо использовать фильтр IN (*[?]*), где *[?]* — массив связующих ключей.

4. Если для атрибута определен специфический запрос извлечения, то он запускается параллельно с наложенными условиями из блока conditions.

5. При наличии в блоке transform подблока after, после извлечения необходимо выполнить преобразование над результатом до сериализации всего ответа.

### 7.3.2 REST-запросы

В текущей версии спецификации REST-источники должны строго соответствовать структуре модели (если не переопределены в блоке extract). Ответы должны быть сериализованы в JSON, авторизация — base.

При извлечении вложенных ресурсов СМЭВ QL сервер должен в параллельном режиме выбирать данные для каждого родительского объекта.

### 7.3.2 СМЭВ QL запросы

При реализации запросов к источнику СМЭВ QL должна выполняться подстановка значений полей в блоке credentials определенным образом:

"credentials": {  
 "system": {  
 "mnemonic": "mnemonic\_smev\_ql\_server",  
 "instance\_id": "smev\_ql\_server\_instance",  
 "user\_id": "original\_user\_id"  
 },  
 "request": {  
 "id": "original\_request\_id",  
 "sub\_id": "original\_sub\_request\_id",  
 "name": "request\_name",  
 "purpose\_id": "dataRequestId", // идентификатор подзапроса СМЭВ QL  
 "audit": true, // для цеполчек необходимо включать аудит  
 "audit\_id": "purpose\_id",  
 "audit\_token": "purpose\_id" // текст, название поля аудита  
 },  
 "signature": {  
 "digest": null,  
 "signature": null  
 }  
 }

# 8. Ответы

Ответы СМЭВ QL сервера представляют из себя объект JSON, схема которого определяется составом запроса.

Например для запроса:

{  
 "query":{  
 "people":{  
 "conditions":{  
 "age":"<35"  
 },  
 "attributes":[  
 "name",  
 "phone",  
 "vsu\_code"  
 ],  
 "military\_office":{  
 "attributes":[  
 "address"  
 ]  
 }  
 }  
 },  
 "credentials":{  
   
 }  
}

Ответ будет следующим:

{  
 "response":{  
 "people":[  
 {  
 "name":"Иван",  
 "phone":"+79011001010",  
 "vsu\_code":"1025",  
 "military\_office":[  
 {  
 "address":"г.Москва, ул.Угрешко"  
 }  
 ]  
 },  
 {  
 "name":"Пётр",  
 "phone":"+79022002020",  
 "vsu\_code":"1026",  
 "military\_office":[  
 {  
 "address":"г.Москва, Хилков переулок"  
 }  
 ]  
 }  
 ]  
 },  
 "credentials":{  
   
 }  
}

## 8.1 Ошибки

Ошибки выводятся в блоке response:

{  
 "response": {  
 "errors": [  
 {  
 "error": "Запрещен вывод атрибутов без переданного guard",  
 "code": "401"  
 }  
 ]  
 },  
 "credentials": {  
   
 }  
}

### 8.1.1 Базовые ошибки СМЭВ QL

**1ХХ Ошибки разбора запроса**

– 101 Запрос не должен называться ‘errors’ — Неправильное название запроса ‘errors’

**2ХХ Ошибки модели**

– 201 Неизвестный атрибут — У ресурса не найден атрибут с соотв. именем

– 202 Неизвестный ресурс — Ресурс с соотв. именем не зарегистрирован в модели данных

– 203 Неизвестная связь — Не найдена связь между двумя ресурсами

– 204 Неправильная связь — Размеры ключей не соответствуют для соединения одного ресурса с другим

**3ХХ Ошибки источников**

– 301 Неизвестный источник — Не найдено описание источника данных с соотв. именем

– 302 Неизвестный адаптер — Не найден адаптер с соотв. именем

**4ХХ Ошибки доступов и ограничений**

– 401 Запрещен вывод атрибутов без переданного guard — Среди запрашиваемых атрибутов есть атрибут с невыполненными ограничениями в блоке guard (не переданы в запросе атрибуты из guard)

– 402 Недостаточно атрибутов для выбора источника — Среди атрибутов фильтрации нет атрибутов, необходимых для выбора источника

– 403 Запрещенные атрибуты для поиска — Среди атрибутов фильтрации есть атрибуты, указанные в блоке denied модели

– 404 Атрибуты для поиска не разрешены — Среди атрибутов фильтрации есть атрибуты, которые не указаны в разрешающем блоке allowed модели

– 405 Попытка переопределения фиксированных условий поиска — Среди атрибутов фильтрации есть атрибуты, которые пытаются переопределить фиксированные ограничения поиска в блоке always модели

**9ХХ Прочие ошибки**

– 901 Непредвиденная ошибка — Непредвиденная ошибка