# Язык Hash Query

Этот язык представляет собой простой декларативный LINQ (Language Integrated Queries) или SQL подобный язык. Целью создания языка было желание предоставить единый, унифицированный интерфейс для работы с криптографическими хэшами строк и файлов с некоторыми дополнительными плюшками вроде восстановления строк по хэшам, поиск файлов по хэшам и пр.

## Основы синтаксиса

Все задания на работу с хэшами в языке представляют собой предложения разделенные точкой с запятой «**;**». Они могут передаваться компилятору или через командную строку (опция **-q**) или быть записанными в простом текстовом файле, который потом скармливается компилятору (опция **-f**). В файле может быть произвольное количество запросов, но не более 10 тыс. (по соображениям производительности).

Пробельные символы в запросах (переводы строк, табуляторы и пр.) не учитываются, за исключением тех, что находятся собственно в данных (строках в кавычках или апострофах).

### Общий вид запросов

Все запросы имеют вид:

**for** … **do** …;

или

**let** var **=** ‘value’;

После for и do идут инструкции специфичные для каждой из категорий запроса (см. далее).

Точка с запятой в конце запроса обязательна, даже в случае единичного запроса.

В инструкциях могут встречаться строки определяемые пользователем (пути к файлам, каталогам, значения хэшей, собственно строки для вычисления по ним хэша). Строка должна быть как в апострофах (одиночных кавычках), так и в обычных кавычках (двойных кавычках). В запросах передаваемых через командную строку должны использоваться апострофы (понятно почему), в запросах из файлов можно использовать любые кавычки.

В инструкциях после **for**, но до **do** могут встречаться подвыражения **let** и **where** не являющиеся обязательными.

Инструкции, начинающиеся с **let,** определяют переменные, которые могут быть использованы в запросах.

### Подвыражение let

Подвыражение **let** не является обязательным и может быть опущено. Также, могут быть опущены отдельные части выражения.

Подвыражение **let** содержит список инструкций с разделителем запятой идущих после ключевого слова **let**, порядок инструкций не имеет значения. Инструкция имеет следующий вид:

<переменная>.<имя атрибута> = <значение атрибута>

Переменная — это любая последовательность латинских букв и цифр, но начинаться она должна **обязательно** с буквы. Переменная должна быть задано до **let**.

Имя атрибута является одним из зарезервированных имен.

Значением атрибута является либо строка (в апострофах или кавычках), либо целое число.

Пример:

let s.md5 = ‘202CB962AC59075B964B07152D234B70’, s.min = 3

### Подвыражение where

Подвыражение **where** используется для задания условий, например для фильтрации файлов по некоторым признакам.

**where** cостоит из выражений вида:

<переменная>.<имя атрибута> <условный оператор> <значение>

идущих после ключевого слова **where** и соединенных операторами конъюнкции (логическое И) — **and** и дизъюнкции (логическое ИЛИ) — **or**, или другими словами после ключевого слова **where** должно быть булево выражение.

Переменная — это любая последовательность латинских букв и цифр, но начинаться она должна **обязательно** с буквы. Переменная должна быть задано до **where**.

Условный оператор — это один из следующий операторов:

* **==** (равно)
* **!=** (не равно)
* **>** (больше)
* **<** (меньше)
* **>=** (больше или равно)
* **<=** (меньше или равно)
* **~** (соответствует регулярному выражению)
* **!~** (не соответствует регулярному выражению)

Для группировки частей выражения (изменения порядка выполнения условных операторов) могут быть использованы круглые скобки.

Пример:

where (f.md5 == ‘202CB962AC59075B964B07152D234B70’ and f.limit == 100) or (f.offset == 10 and f.md4 == ‘C58CDA49F00748A3BC0FCFA511D516CB’)

### Категории запросов

Все запросы можно разделить на 5 категорий:

1. Определение переменных
2. Запросы на вычисление хэша строки
3. Запросы на восстановление строки по её хэшу
4. Запросы на вычисление хэша отдельных файлов
5. Запросы на вычисление хэша файлов в заданном каталоге

#### Определение переменных

Данные запросы нужны для определения переменных, ссылаться на которые можно в других запросах. Это нужно, например, для устранения дублирования одних и тех же данных, в случае если их по смыслу необходимо использовать в разных местах.

Запросы имеют следующий вид:

**let** <переменная> = '<строка>';

На определенную в этом запросе переменную, точнее данные, определяемые этой переменной, можно ссылаться в других запросах.

#### Запросы на вычисление хэша строки

Запросы имеют следующий вид:

**for** **string** '<строка>' **do** <хэш>;

Хэш может принимать одно из следующих значений (регистр имеет значение):

* md5
* md4
* sha1
* sha256
* sha384
* sha512
* whirlpool
* crc32

#### Запросы на восстановление строки по её хэшу

Общий вид запроса такой:

**for string** <переменная> **from hash** '<значение хэша>' [**let** …] **do crack** <хэш>**;**

Здесь хэш является одним из поддерживаемых типов хэшей (см. предыдущий раздел), значение хэша.

Подвыражение **let** используется для управления параметрами восстановления строки, а именно:

1. словарь (атрибут **dict**)
2. минимальная длина (атрибут **min**)
3. максимальная длина (атрибут **max**)

#### Запросы на вычисление хэша отдельных файлов

Общий вид запроса:

**for file** f **from** '<путь к файлу>' [**let** …] **do** <хэш|**validate**>;

Такие запросы предназначены либо для вычисления хэша отдельных файлов, либо для валидации файлов по хэшу, что определяется видом инструкции **do**.

Подвыражение **let** используется для определения той части файла, по которой производится валидация или для которой вычисляется хэш. Если опущено, то используется весь файл целиком. Атрибуты могут быть следующие:

1. Хэш (атрибут **md5**|**md4**|**sha1**|**sha256**|**sha384**|**sha512**|**whirlpool**|**crc32**) — имеет смысл только при валидации файла.
2. Смещение относительно начала файла в байтах (атрибут **offset**)
3. Размер данных для которых вычисляется хэш в байтах (атрибут **limit**)

Хэш может принимать одно из следующих значений (регистр имеет значение):

* md5
* md4
* sha1
* sha256
* sha384
* sha512
* whirlpool
* crc32

#### Запросы на вычисление хэша файлов в заданном каталоге

Общий вид запроса:

**for file** f **from dir** '<каталог>' [**where** ...] **do** <хэш|**find**> [**withsubs**];

Такие запросы предназначены либо для вычисления хэша файлов в каталоге(ах) — после **do** идет хэш, либо для поиска файлов (по хэшам) — после **do** идет ключевое слово **find**.

Подвыражение where используется для фильтрации файлов. Допустимые атрибуты:

1. Имя файла (атрибут **name**)
2. Полный путь к файлу включая имя (атрибут **path**)
3. Размер файла в байтах (атрибут **size**)
4. Хэш (атрибут **md5**|**md4**|**sha1**|**sha256**|**sha384**|**sha512**|**whirlpool**|**crc32**) — имеет смысл только при поиске файла.
5. Смещение относительно начала файла в байтах (атрибут **offset**)
6. Размер данных для которых вычисляется хэш в байтах (атрибут **limit**)

Опция **withsubs** задействует вложенные каталоги (по умолчанию выключена).

Для фильтрации файлов по имени и/или пути могут быть использованы Perl совместимые регулярные выражения (PCRE). Для фильтрации файлов по регулярным выражениям используются условные операторы **~** и **!~**.