GQLcheatsheet

sones

GQL

The sones Graph Query Language (GQL) allows you to create, modify, administrate and query a graph (database).

CREATE TYPE

Create a user-defined datatype

```
CREATE [ABSTRACT] TYPE Identifier [EXTENDS Identifier] [ATTRIBUTES(AttrIdentifier
[=DefaultValue])] [BACKWARDEDGES(BackwardEdge [,BackwardEdge])] [UNIQUE(Identifier
[,Identifier])] [MANDATORY(Identifier [,Identifier])] [INDICES(IndexOptOnCreateType)]
```

CREATE TYPES

Create user-defined datatypes having circular dependencies

```
CREATE TYPES [[ABSTRACT] bulk_type_list_member] [,[ABSTRACT]bulk_type_list_member]]
```

ALTER TYPE

Manipulates a user-defined datatype

```
ALTER TYPE Identifier ADD ATTRIBUTES(attr_definition [,attr_definition])|
ALTER TYPE Identifier DROP ATTRIBUTES (Identifier [,Identifier]) |
ALTER TYPE Identifier ADD BACKWARDEDGES (BackwardEdge [,BackwardEdge]) |
ALTER TYPE Identifier DROP BACKWARDEDGES (Identifier [,Identifier]) |
ALTER TYPE Identifier RENAME ATTRIBUTE Identifier TO Identifier |
ALTER TYPE Identifier RENAME TO Identifier |
ALTER TYPE Identifier RENAME BACKWARDEDGE Identifier TO Identifier |
ALTER TYPE Identifier UNIQUE (Identifier [,Identifier]) |
ALTER TYPE Identifier DROP UNIQUE
```

DROP TYPE

Deletes a user-defined datatype

```
DROP TYPE Identifier
```

TRUNCATE

Removes all records from a user-defined datatype

```
TRUNCATE Identifier
```

CREATE INDEX

Creates an index on a user-defined type

```
CREATE INDEX identifier [EDITION identifier] ON identifier (index_attribute_list)
[INDEXTYPE Identifier]
```

REBUILD INDICES

rebuilds all or specific index of a user-defined datatype

```
REBUILD INDICES
REBUILD INDICES TypeIdentifier [,TypeIdentifier]
```

DROP INDEX

Deletes an index of a user-defined datatype

```
FROM Identifier DROP INDEX IndexIdentifier [EDITION EditionIdentifier]
```

INSERT

Insert an object into a graph

```
INSERT INTO Identifier VALUES (AttrAssign [,AttrAssign]) [WHERE where_condition]

AttrAssign = [idNode=Value]|[idNode={REF(..),REFUUID(..),SETOF(..),LISTOF(..),SETOFUUIDS
(..)}
```

INSERTORUPDATE

Inserts or Updates an object into a graph

```
INSERTORUPDATE Identifier VALUES (AttrAssign [,AttrAssign]) [WHERE where_condition]
```

INSERTORREPLACE

Inserts or Replaces an object into a graph

```
INSERTORREPLACE Identifier VALUES (AttrAssign [,AttrAssign]) [WHERE where_condition]
```

UPDATE

Updates an object within a graph

```
UPDATE Identifier SET (update_attr_element [,update_attr_element]) [WHERE
where_condition]
```

REPLACE

Replaces an object within a graph

```
REPLACE Identifier VALUES (AttrAssign [,AttrAssign]) WHERE where_condition
```

DELETE

Deletes data from an object with a graph

```
FROM Identifier DELETE [id] [WHERE where_condition]
```

BEGIN TRANSACTION

Starts a new transaction

```
BEGIN [DISTRIBUTED] | [LONG-RUNNING] | [DISTRIBUTED LONG-RUNNING] TRANSACTION [ISOLATION
= {Full | Read | Write}] [Name=transaction_name] [TIMESTAMPE=transaction_timestamp]
```

COMMIT TRANSACTION

Commits a currently running transaction

```
COMMIT TRANSACTION [Name=transaction_name] [ASYNC]
```

ROLLBACK TRANSACTION

Revoke all changes of a transaction (rollback)

```
ROLLBACK TRANSACTION [Name=transaction_name] [ASYNC]
```

BUILT-IN Datatypes (case sensitive)

```
Integer, UnsignedInteger, Double, String, Boolean, DateTime, BackwardEdge, List<Primitive-
Datatype>, Set<Userdefined-Datatype>
```

Aggregates

```
COUNT | AVG | MIN | MAX | SUM
```

Functions

```
CONCAT | TOUPPER | MAXWEIGHT | SUBSTRING | TOP | CURRENTDATE
```

Comments

```
Comments within lines start with /* and end with */
```

DESCRIBE

Get information on items within the database

```
DESCRIBE INDICES | DESCRIBE INDEX Identifier [edition] |
DESCRIBE EDGES | DESCRIBE EDGE Identifier |
DESCRIBE FUNCTIONS | DESCRIBE FUNCTION Identifier |
DESCRIBE AGGREGATES | DESCRIBE AGGREGATE Identifier |
DESCRIBE SETTINGS [ON TYPE TypeIdentifier | ON ATTRIBUTE attr_definition | ON DB |  ON
SESSION] | DESCRIBE SETTING Identifier [ON TYPE TypeIdentifier | ON ATTRIBUTE attr_definition
| ON DB | ON SESSION] |
DESCRIBE TYPES | DESCRIBE TYPE TypeIdentifier
```

SETTING

Set, get or remove a setting (in a session or global)

```
SETTING DB SettingOperation (= SET ("SettingName"=value [,"SettingName"=value]) | GET
("SettingName" [,"SettingName"]) | REMOVE ("SettingName" [,"SettingName"])
SETTING SESSION SettingOperation | SETTING TYPE Identifier SettingOperation |
SETTING ATTRIBUTE Identifier SettingOperation
SettingName = (DEPTH, SETREADONLY, TYPE, UUID, SHOWREVISION, SELECTTIMEOUT)
```

DUMP

Create a complete (query) dump in different formats

```
DUMP [{GDDL|GDML|ALL}] [AS {GQL,CSV}]
```

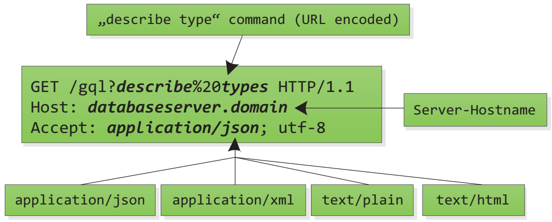
SELECT

Retrieve objects or their data from a graph

```
FROM Identifier alias [,Identifier Alias] SELECT * | selection_list
[WHERE where_condition] [GROUP BY group_expr] [ORDER BY order_expr] [OFFSET offset_expr]
[LIMIT  limit_expr] [DEPTH  depth_expr]
```

REST

The REST API can be used to run GQL queries. The query is encapsulated into the URL (URL encoded) and data then is returned in the specified format (xml / json / text). Basic authentication is mandatory.



```
GET /gql?describe%20types HTTP/1.1
Host: databaseserver.domain
Accept: application/json; utf-8
```

Accessing the log:

```
GET /Logfile
```

Login and Access

sones graphDB can be accessed through a number of different interfaces like SOAP WebService, REST - even a WebShell running inside every browser is available.

WebShell

sones graphDB WebShell is a unix-like shell in a web browser allowing the user to interact with an instance of a sones graphDB. It is based on well-known technologies and libraries like HTML, JavaScript, JQuery and our REST API. Authentication is mandatory.

Anyone can gain access to a personalized instance by logging on to http://www.sones.com.

useful commands:

command	parameters	description
help	n/a	displays help text
clear	n/a	clears the screen
format	xml I json I text	by default results are displays as xml. Output format can be switched to JSON or plain-text

sample xml output:

```
<sones version="1.0">
  <GraphDB version="1.0">
    <queryresult version="1.0">
      <query>FROM UserProfile u SELECT *</query>
      <result>Successful</result>
      <duration resolution="ms">1</duration>
      <warnings />
      <errors />
      <results>
        <DBObject>
          <attribute name="Username" type="String">User1</attribute>
          <attribute name="Age" type="Int64">25</attribute>
        </DBObject>
      </results>
    </queryresult>
  </GraphDB>
</sones>
```

sample json output:

```
{
  "query": "FROM User u SELECT *",
  "result": "Successful",
  "warnings": [],
  "errors": [],
  "results":
  [{
   "Username": "User1",
   "Age": "25"
   }],
  "Duration": ["1","ms"]
}
```

based on GQL 1.1 specification — comments to info@sones.com are welcome — Last Update: June 28th, 2010

GQLcheatsheet examples

Examples

CREATE TYPE

```
CREATE TYPE
    UserProfile ATTRIBUTES (String Username, Integer Age, SET<UserProfile> Friends)
    BACKWARDEDGES (UserProfile.Friends FriendOf)
    UNIQUE (Username)
    MANDATORY (Username)
CREATE TYPE
    SecondUserProfile EXTENDS UserProfile ATTRIBUTES (SET<UserProfile> Enemies)
CREATE TYPE
    ThirdUserProfile EXTENDS UserProfile ATTRIBUTES (LIST<String> Hobbies)
```

CREATE TYPES

```
CREATE TYPES
    Tag ATTRIBUTES (String Name) BACKWARDEDGES (Website.Tags TaggedSite) INDICES (Name)
    Website ATTRIBUTES (String Name, String URL, SET<Tag> Tags) INDICES (Name)
```

ALTER TYPE

```
ALTER TYPE
    UserProfile ADD ATTRIBUTES (SET<UserProfile> Enemies, UserProfile Father)
ALTER TYPE
    UserProfile ADD BACKWARDEDGES (UserProfile.Enemies EnemyOf)
ALTER TYPE
    UserProfile ADD ATTRIBUTES (SET<WEIGHTED(Double, DEFAULT = 1.5, SORTED = DESC)
    <UserProfile>> WeightedFriends )
```

DROP TYPE

```
DROP TYPE SecondUserProfile
```

DROP INDEX

```
FROM UserProfile DROP INDEX IDX_Name
```

CREATE INDEX

```
CREATE INDEX
    IDX_Name ON UserProfile(Name) INDEXTYPE Hashtable
```

TRUNCATE

```
TRUNCATE UserProfile
```

INSERTORUPDATE

```
INSERTORUPDATE
    UserProfile VALUES (Name = "User1", Age = 40) WHERE Name = "User1"
INSERTORUPDATE
    UserProfile VALUES (Name = "User5") WHERE Name = "User5"
```

INSERTORREPLACE

```
INSERTORREPLACE
    UserProfile VALUES (Name = "User1-new", Age = 50) WHERE Name = "User1"
```

INSERT Insert an object into a graph

```
INSERT INTO
    UserProfile VALUES (Name = "User1", Age = 25)
INSERT INTO
    UserProfile VALUES (Name = "User2", Age = 22, Friends = SETOF(Name = "User1"))
INSERT INTO
    UserProfile VALUES (Name = "User3" Father = REF(Name = "User1"), Enemies = SETOF(Name = "User2"))
INSERT INTO
    UserProfile VALUES (Name = "User4", WeightedFriends = SETOF(Name = "User1":(1.2)))
INSERT INTO
    ThirdUserProfile VALUES (Name = "User5", Hobbies = LISTOF("Books","Music"))
INSERT INTO
    UserProfile VALUES (Name = "UndefinedAttributeUser", Weight = 70.1) /* Undefined Attribute */
```

SELECT

```
FROM UserProfile U SELECT U.Friends.Friends.Name WHERE U.Name = "User1"
FROM UserProfile U SELECT U.Name WHERE U.Friends.Friends.Name = "User1"
FROM UserProfile U SELECT U.Friends.Count()
```

DESCRIBE

```
DESCRIBE TYPES
DESCRIBE TYPE UserProfile
DESCRIBE SETTING Depth ON TYPE UserProfile
DESCRIBE SETTING Depth ON ATTRIBUTE UserProfile.Friends
DESCRIBE SETTINGS ON DB
DESCRIBE INDEX Tag.Name
DESCRIBE FUNCTION CONCAT
```

SETTING

```
SETTING TYPE User SET („DEPTH“=2)
```

UPDATE

```
UPDATE
    UserProfile SET (Friends += SETOF(Name="User2", Name="User4")
    WHERE Name = "User3"
```

REPLACE

```
REPLACE
    UserProfile VALUES (Name = "User1", Age = 50)
    WHERE Name= "User1-new"
```

DELETE

```
FROM UserProfile DELETE WHERE Name = "User1"
```

Functions

```
FROM UserProfile U
    SELECT U.Name.CONCAT("-> Age:", U.Age) AS "Name"
FROM UserProfile U
    SELECT U.Name, CURRENTDATE()
FROM UserProfile U
    SELECT U.WeightedFriends.MAXWEIGHT()
FROM UserProfile U
    SELECT U.Name.SUBSTRING(2,3)
FROM UserProfile U
    SELECT U.Friends.Path(S.Name = "User1", 10, 10, "true", "false")
    WHERE U.Name = "User2" DEPTH 1
```

Aggregates

```
FROM UserProfile U
    SELECT AVG(U.Age)
FROM UserProfile U
    SELECT AVG(U.Age), U.Name
    GROUP BY U.Name
    OFFSET 10
    LIMIT 10
```

based on GQL 1.1 specification — comments to info@sones.com are welcome — Last Update: June 28th, 2010 © sones GmbH