NosDB SQL Query Language Cheat Sheet

This cheat sheet provides references to commonly used SQL queries to manage and access data in NosDB as simple JSON documents.

Example Group JSON Documents

```json
{
    "id": "Group0001",
    "name": "Peterson",
    "members": [
        {
            "firstName": "Jacques",
        },
        {
            "firstName": "Markus",
        },
        {
            "firstName": "Samantha",
            "gender": "female",
            "toys": [
                {
                    "givenName": "Barbie"
                }
            ],
        },
        {
            "id": "Group0002",
            "name": "Peterson",
        }
    ],
    "creationDate": "20150103T120000",
    "isRegistered": true,
    "location": {
        "type": "Point",
        "coordinates": [33.9, -4.8]
    },
    "address": {
        "state": "PA",
        "county": "New Castle",
        "city": "Massachusetts"
    }
}
```

SQL SELECT Query

```sql
-- Get all Group(s) which has id equals to 'Group0001'.
SELECT * FROM Groups WHERE id = 'Group0001';
```

SQL SELECT Query + JSON

```sql
-- Get items(s) by JSON.
SELECT * FROM Groups WHERE address = {
    "state": "CL",
    "county": "Colorado",
    "city": "Colorado"
};
```

SQL UDF in .NET

```sql
-- UDF definition:
function (input, pattern): false
{
    return Regex.IsMatch(input, pattern);
}
```

SQL Insert + JSON Child

```sql
-- Inserting a document which has a JSON document as a child.
INSERT INTO Groups ('id', lastName, members)
VALUES ('Group0001', 'Peterson', {
    "firstName": "Jacques",
    "gender": "female",
    "toys": [
        {
            "givenName": "Barbie"
        }
    ],
    "name": "Peterson",
    ...
});
```

SQL Insert + JSON Array

```sql
-- Inserting a JSON document which has JSON documents in a JSON array.
INSERT INTO "Groups" (id, lastName, members)
VALUES ('Group0001', 'Peterson', {
    "firstName": "Jacques",
    "gender": "female",
    "toys": {
        "givenName": "Barbie"
    },
});
```

SQL Insert Query

```sql
-- Inserting a simple document into collection "Groups".
INSERT INTO Groups ('id', 'lastName')
VALUES ('Group0001', "Peterson")
```

Built-in Functions (Case insensitive)

<table>
<thead>
<tr>
<th>Mathematics</th>
<th>avg, sum, round</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>lcase, ucase, len, mid, format</td>
</tr>
<tr>
<td>Date Time</td>
<td>Now</td>
</tr>
<tr>
<td>Type Indifferent</td>
<td>Max, Min, Count, First, Last</td>
</tr>
<tr>
<td>Custom Methods</td>
<td>Define UDF in .NET</td>
</tr>
</tbody>
</table>

Operators

<table>
<thead>
<tr>
<th>Arithmetic</th>
<th>+, -, *, /, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical</td>
<td>AND, OR, NOT</td>
</tr>
<tr>
<td>Comparison</td>
<td>=, !=, &gt;, &gt;=, &lt;, &lt;=, &lt;&gt;</td>
</tr>
<tr>
<td>String</td>
<td>+ (concatenate)</td>
</tr>
</tbody>
</table>

SQL Update Query

```sql
-- Adding/updating "passports"=null in all documents
UPDATE Groups SET "Passports"=null
```
### Data Update Query + Filter

**-- Deleting the key “Passports” where ID Exists.**

**UPDATE** Groups **SET** (DELETE "Passports")
WHERE id EXISTS

2 affected documents. “Passports” deleted from both.

### SQL UPDATE Addition in an Array

**-- Adding a pet in childrens’ pets if the family is registered.**

**UPDATE** Groups **SET**
children.pets **ADD**
{
  "givenname": "Diesel"
}
WHERE isRegistered = false

1 affected document.

### Update Options

**EQUALS (=)**
Replaces the value of a key if exists, else adds the key-value pair in the document.

**ADD**
Adds the value(s) given at the end of the array existing against an attribute.

**INSERT**
Adds the value(s) at the end of array if they do not exist in the array against an attribute.

**REMOVE**
Removes the given value(s) from the array against an attribute.

**REPLACE**
Replaces given values by their pairs given in the array existing against an attribute.

**RENAME TO**
Renames a given attribute in the document to the specified name.

**DELETE**
Deletes the specified attribute from the document if exists.

### SQL Delete + Filter

Deleting the registered group.

**DELETE FROM** "Families" WHERE isRegistered = false

1 affected document.

### Sample SELECT Queries

#### Comparison (range) operators

**SELECT** *
**FROM** Groups
**WHERE** children.grade >= 5

#### Logical operators

**SELECT** *
**FROM** Groups
**WHERE** children.grade >= 5 **AND** isRegistered = true

#### ORDER BY keyword

**SELECT** id, address.city
**FROM** Groups
**ORDER BY** address.city

#### IN keyword

**SELECT** *
**FROM** Groups
**WHERE** address.state IN
('NY', 'WA', 'CA', 'PA', 'OH', 'OR', 'MI', 'WI')

#### Constant Evaluation

**SELECT** 1+2 **AS** NumberThree
**FROM** Groups

#### Parameterized SQL

**SELECT** *
**FROM** Groups
**WHERE** lastName = @lastName **AND** address.state = @addressState

#### String Built-in functions

**SELECT** Families.id, address.city
**FROM** Groups
**WHERE** STRLEN(Families.id) == 5

#### Exists Keyword

**SELECT** *
**FROM** Groups
**WHERE** grade EXIST

#### Array Projection

**SELECT** (location.coordinates) **SLICE** [0,1]
**FROM** Groups
**WHERE** grade EXIST

#### Delimited Identifiers

**SELECT** *
**FROM** $Groups$
**WHERE** "grade" EXIST

#### Embedded Attribute with indexer

**SELECT** *
**FROM** Groups
**WHERE** members.toys[0].givenname = 'Lisa'

### Sample DML Queries

#### Parameterized Insert

**INSERT INTO** Groups ("Name","Collection")
**VALUES** (@name, @array)

#### Inserting Custom DateTime

**INSERT INTO** Groups ("Name","DateTime")
**VALUES** ('Josh', DateTime('12/30/2011'))

#### Replacing items in an array

**UPDATE** Groups
**SET** (Collection REPLACE {1=3, 4=5, 7=10})

#### Adding items in an array

**UPDATE** Groups
**SET** (Collection ADD {id:1})

#### Removing items from an array

**UPDATE** Groups
**SET** (Collection REMOVE {id:1})

#### Renaming an Attribute

**UPDATE** Groups
**SET** (RENAME Collection TO 'Items')