Database

Methods

- Get
- BatchGet
- Put
- BatchPut
- Delete
- BatchDelete
- Update
- BatchUpdate
- Query

Documentation

Get

`cloudSDK.database.get(Table, Key)`

Params

1. Table / Collection Name **STRING**
2. Key **OBJECT**

Example

```
const item = await cloudSDK.database.get("Connect_Products", {account:
"tester", id: 5});
```

Response

Database Item **OBJECT**

BatchGet

`cloudSDK.database.batchGet(Table, Keys)`

Params

1. Table / Collection Name **STRING**
2. Keys **ARRAY** of **OBJECTS**

Example

```
const item = await cloudSDK.database.batchGet("Connect_Products", [
{account: "tester", id: 5},
{account: "tester2", id: 1},
{account: "tester2", id: 3}
]);
```

Response

Database Items **ARRAY**

Put

`cloudSDK.database.put(Table, Item, Expected)`

Params

1. Table / Collection Name **STRING**
2. Item **OBJECT**
3. Expected (optional) **OBJECT**

The item must include the primary key of the table / collection

Expected

- operator (`EQ | NE | IN | LE | LT | GE | GT | BETWEEN | NOT_NULL | NULL | CONTAINS | NOT_CONTAINS | BEGINS_WITH`)
- value **STRING** **NUMBER** **BOOLEAN**
- exists **BOOLEAN**

Exists checks if the item is existing in the database or not. If `exists = false`, the put operation will only proceed if the item doesn't already exist in the database.

You can use `exists` only

```
{
  stock: {
    operator: "GT", // Greater Than
    value: 3 // if operator BETWEEN: [2, 10],
    Exists: true
  }
}
```

Example

```
const item = await cloudSDK.database.put(
  "Connect_Products",
  {account: "tester", id: 5, name: "DJI Mavic Pro", description: "..."
[...]},
  {stock: {Exists: false}} // Only create if stock doesn't exists
  );
```

Response

Database creation response **OBJECT** (unfiltered platform response)

BatchPut

`cloudSDK.database.batchPut(Table, Items)`

Params

1. Table / Collection Name **STRING**
2. Items **ARRAY** of **OBJECTS**

> The items must include the primary key of the table / collection

> `Expected` is not available in batchPut

Example

```
const item = await cloudSDK.database.put(
  "Connect_Products",
  [
  {account: "tester", id: 5, name: "DJI Mavic Pro", description: "..."
[...]},
  {account: "tester", id: 6, name: "DJI Mavic Zoom", description: "..."
[...]},
  {account: "tester", id: 7, name: "DJI Mavic Air", description: "..."
[...]}
  ]
  );
```

Response

Database creation response **OBJECT** (unfiltered platform response)

Delete

`cloudSDK.database.delete(Table, Key)`

Params

1. Table / Collection Name **STRING**
2. Key **OBJECT**

Example

```
await cloudSDK.database.delete("Connect_Products", {account: "tester",
id: 5});
```

Response

Database deletion response **OBJECT** (unfiltered platform response)

BatchDelete

`cloudSDK.database.batchDelete(Table, Keys)`

Params

1. Table / Collection Name **STRING**
2. Keys **ARRAY** of **OBJECTS**

Example

```
await cloudSDK.database.delete("Connect_Products", [
{account: "tester", id: 5},
{account: "tester", id: 6},
{account: "tester", id: 7}
]);
```

Response

Database deletion response **OBJECT** (unfiltered platform response)

Update

`cloudSDK.database.update(TableName, Key, UpdateAttributes, Expected)`

Params

1. Table / Collection Name **STRING**
2. Key **OBJECT**
3. UpdateAttributes **OBJECT**
4. Expected (optional) **OBJECT**

UpdateAttributes

You don’t need to update the whole item but single keys

Basic

```
{
  stock: 5,
  // optional keys
}
```

Extended

Params

- action (`ADD | SET | DELETE | REMOVE`) STRING
- value (no required if delete) STRING OBJECT ARRAY NUMBER BOOLEAN

```
{
  stock: {
    action: "ADD",
    value: 3
  }
}
```

Expected

- operator (`EQ | NE | IN | LE | LT | GE | GT | BETWEEN | NOT_NULL | NULL | CONTAINS | NOT_CONTAINS | BEGINS_WITH`)
- value STRING NUMBER BOOLEAN

```
{
  stock: {
    operator: "GT", // Greater Than
    value: 3 // if operator BETWEEN: [2, 10]
  }
}
```

Example

```
await cloudSDK.database.update(
  "Connect_Products",
  // Key
  {
    account: "tester",
    id: 5
  },
  // UpdateAttributes
  {
    name: "New Name",
    stock: {
      action: "ADD",
      value: 3
    }
  },
  // Expected
  {
    stock: {
      operator: "GT", // Greater Than
      value: 3 // if operator BETWEEN: [2, 10]
    }
  }
)
```

Response

Database update response **OBJECT**

BatchUpdate

`cloudSDK.database.batchUpdate(TableName, Items)`

Params

1. Table / Collection Name **STRING**
2. Items **ARRAY** of **OBJECTS**

Params of Items

1. Table / Collection Name **STRING**
2. Key **OBJECT**
3. UpdateAttributes **OBJECT**
4. Expected (optional) **OBJECT**

Key key

```
{account: "tester", id: 5}
```

UpdateAttributes `updateAttributes`

> You don't need to update the whole item but single keys

Basic

```
{
  stock: 5,
  // optional keys
}
```

Extended

Params

- action (`ADD | PUT | DELETE`) STRING
- value (no required if delete) STRING OBJECT ARRAY NUMBER BOOLEAN

```
{
  stock: {
    action: "ADD",
    value: 3
  }
}
```

Expected `expected`

- operator (`EQ | NE | IN | LE | LT | GE | GT | BETWEEN | NOT_NULL | NULL | CONTAINS | NOT_CONTAINS | BEGINS_WITH`)
- value STRING NUMBER BOOLEAN

```
{
  stock: {
    operator: "GT", // Greater Than
    value: 3 // if operator BETWEEN: [2, 10]
  }
}
```

Example

```
await cloudSDK.database.batchUpdate(
  "Connect_Products",
  [
  // First update item
  {
    key: {
      account: "tester",
      id: 5
    },
    updateAttributes: {
      name: "New Name",
      stock: {
        action: "ADD",
        value: 3
      }
    },
    expected: {
      stock: {
        operator: "GT", // Greater Than
        value: 3 // if operator BETWEEN: [2, 10]
      }
    }
  },
  // Second update item
  {
    key: {
      account: "tester",
      id: 6
    },
    updateAttributes: {
      name: "New Name",
      stock: {
        action: "ADD",
        value: 6
      }
    },
    expected: {
      stock: {
        operator: "GT", // Greater Than
        value: 4 // if operator BETWEEN: [2, 10]
      }
    }
  }
  ]
)
```

Response

Database update response **OBJECT**

Query

```
cloudSDK.database.query(Table, PrimaryKey, Settings)
```

Params

1. Table / Collection Name **STRING**
2. PrimaryKey **OBJECT**
3. Settings (optional) **OBJECT**

PrimaryKey

Basic key

```
{
  account: "tester",
  // optional keys
}
```

Extended key

Operators:

```
EQ | NE | IN | LE | LT | GE | GT | BETWEEN | NOT_NULL | NULL | CONTAINS | NOT_CONTAINS | BEGINS_WITH
```

```
{
  account: "tester",
  id: {
    operator: "GT", // Greater Than
    value: 3 // if operator BETWEEN: [2, 10]
  }
}
```

Settings

- index **STRING**
- limit **NUMBER**
- startKey **OBJECT**
- forward **BOOLEAN**
- attributes **ARRAY**
- filter **OBJECT**

Example

```
const items = await cloudSDK.database.query(
  "Connect_Products",
  {account: "tester"}, // PrimaryKey
  // Settings
  {
    index: "sku-index",
    limit: 15,
    startKey: {account: "tester", id: 10}, // Will start at this item
    forward: false,
    attributes: ["account", "id", "name", "stock"]
  }
);
```

Response

Database Items **ARRAY**