Influx-DB Wrapper

Changes applied to stslgov2

stsl.go API() list

v1	v2	v2 Description
NewTimeSeriesClientData()	NewTimeSeriesClientData()	Constructor for type TimeSeriesClientData which is used to store data: connection to timeseriesDB, and information of DB to be used.
CreateTimeSeriesConnection()	CreateTimeSeriesConnection()	Creates a connection to TimeSeriesDB.
CreateTimeSeriesDB()	CreateTimeSeriesDB()	Creates the DB of name specified during the constructor of TimeSeriesClientData and set it default(infinite) retention policy.
CreateTimeSeriesDBWithRetentionPolicy()	CreateTimeSeriesDBWithRetentionPolicy()	Creates the DB of name specified during the constructor of TimeSeriesClientData and set it custom retention policy.
DeleteTimeSeriesDB()	DeleteTimeSeriesDB()	Deletes the DB of name specified during the constructor of TimeSeriesClientData.
-	UpdateTimeSeriesDBRetentionPolicy()	Updates the DB of name specified during the constructor of TimeSeriesClientData with custom retention policy.
DropMeasurement()	DropMeasurement()	Deletes the measurement specified as an arguement.
CreateRetentionPolicy()	***deprecated*** **- ref: 2. Not managing *retention policies* seperately, *but* mapping to *bucket***	
UpdateRetentionPolicy()		
DeleteRetentionPolicy()		
Set()	Set()	Mimics the traditional set operation of key-value pair. Inserts key-value pair into fieldset of TimeSeriesDB.
Get()	Get()	Mimics the traditional get operation of key-value pair. Gets the latest by time value of given key.
Query()	Query()	Generic query API for querying the TimeSeriesDB. Return type is QueryTableResult structure of TimeSeriesDB.api GO library. In influxDBv2, generic query should be used with flux.
WritePoint()	WritePoint()	Generic write API to write a set of tags & fields to mentioned measurement/table in TimeSeriesDB.
InsertJson()	InsertJson()	Use to insert JSON object in mentioned measurement/table.
InsertJsonArray()	InsertJsonArray()	Use to insert JSON array as individual rows in mentioned measurement/table. To be used only when top level JSON has array and not when array is nested inside one existing JSON. Eg. Not to be used for UeMetrics with multiple neighbor cells.
Flatten()	Flatten()	Generic API to flatten JSON data. This will handle nested JSON as well and split it into individual columns.

Details

1. Using '*bucket*' (Database+Retention Policy) units as TimeSeriesDb

stslgov1 : timeSeriesDB = "Database"

slsgov1

```
type TimeSeriesClientData struct {
        Iclient            TimeSeriesDataGoClient // Connection to TimeSeriesDB
        timeSeriesDbName   string                 // TimeSeries DB to be used for this XAPP
        timeSeriesUserName string                 // Username for accessing the TimeSeries DB
        timeSeriesPassword string                 // Password for accessing the TimeSeries DB
}
```

stslgov1 : connection{"hostname", "port"}, auth{"user", "password"}

slsgov1

```
func (timeserData *TimeSeriesClientData) CreateTimeSeriesConnection() (err error) {
        // TimeSeriesDB specific intialization
        hostname := os.Getenv("TIMESERIESDB_SERVICE_HOST")
        if hostname == "" {
                hostname = "127.0.0.1"
        }
        port := os.Getenv("TIMESERIESDB_SERVICE_PORT_HTTP")
        if port == "" {
                port = "8080"
        }
```

stslgov2 : timeSeriesDB = "Bucket"(databases and retention policies are mapped to buckets [link])

slsgov2

```
type TimeSeriesClientData struct {
        iClient           influxdb2.Client // Connection to TimeSeriesDB
        timeSeriesOrgName string
        timeSeriesDb      TimeSeriesDb // TimeSeries DB to be used for this XAPP
}

type TimeSeriesDb struct {
        Name            string
        RetentionPolicy string
}
```

stslgov2 : connection{"host"(hostname:port)}, auth{"token"}

slsgov2

```
func (timeserData *TimeSeriesClient) CreateTimeSeriesConnection(host, token string) (err error) {
        log.Info().Msgf("Establishing connection with TimeSeriesDB host: %v\n", host)
        (*timeserData).iClient = influxdb2.NewClient(host, token)
```

- Using 'bucket' (Database+Retention Policy) units as TimeSeriesDb
- User credential values(UserName, Password) used for database authentication are replaced by token use.

2. Not managing *retention policies* seperately, *but* mapping to *bucket*

- Thus, Create/Update/DeleteRetentionPolicy() are deleted.
- UpdateTimeSeriesDBRetentionPolicy() is newly implemented for updating TimeSeriesDb in use.

3. The concept of organization is updated, but it is not used.

- An *organization*, a workspace for a group of users is newly updated in influxDBv2.
- User can switch *organization* to use by setting $TIMESERIESDB_SERVICE_ORG_NAME.

4. Batch writing code is removed

stslgov1 : *NewBatchPoint() ... Write(batchPoint) ...*

stslgov2 : *WritePoint()*

- In influxDBv2 service, *WritePoint()* implements batch writing logic.

5. Using influxDBv2 go client providing APIs

stslgov1 : Query(NewQuery("CREATE DATABASE ...")) (based of InfluxQL)

slsgov1

```
func (timeserData *TimeSeriesClientData) CreateTimeSeriesDB() (err error) {
        q := timesrclient.NewQuery(fmt.Sprintf("CREATE DATABASE %v", (*timeserData).timeSeriesDbName), "", "")

func (timeserData *TimeSeriesClientData) DeleteTimeSeriesDB() (err error) {
        q := timesrclient.NewQuery(fmt.Sprintf("DROP DATABASE %v", (*timeserData).timeSeriesDbName), "", "")

func (timeserData *TimeSeriesClientData) DropMeasurement(measurement string) (err error) {
        q := timesrclient.NewQuery(fmt.Sprintf("DELETE FROM %v", measurement), (*timeserData).timeSeriesDbName,
"")

func (timeserData *TimeSeriesClientData) Get(measurement, key string) (result interface{}, err error) {
        queryStr := fmt.Sprintf("SELECT %v FROM %v ORDER BY time DESC LIMIT 1", key, measurement)
```

stslgov2 : bucketsAPI.CreateBucket()

slsgov2

```
func (timeserData *TimeSeriesClient) CreateTimeSeriesDB() (err error) {
        _, err = bucketsAPI.CreateBucketWithName(context.Background(), org, bucketName, domain.RetentionRule{
                EverySeconds: durationInt64,
        })

func (timeserData *TimeSeriesClientData) DeleteTimeSeriesDB() (err error) {
        err = bucketsAPI.DeleteBucket(context.Background(), bucket)

func (timeserData *TimeSeriesClientData) DropMeasurement(measurement string) (err error) {
        err = deleteAPI.DeleteWithName(ctx, orgName, bucketName, startTime, stopTime, predicate)

func (timeserData *TimeSeriesClient) Get(measurement, key string) (result interface{}, err error) {
        fluxQueryStr := fmt.Sprintf(`
        from(bucket: "%s")
    |> range(start: -%s)
    |> filter(fn: (r) => r._measurement == "%s" and r._field == "%s")
        `, timeserData.timeSeriesDb.Name, timeserData.timeSeriesDb.RetentionPolicy, measurement, key)
```

- In stslgov1, most of the service methods are based on DB query.
- In stslgov2, various types of APIs are provided.

6. Flux based generic query

stslgov1 : "SELECT FROM WHERE" (based of InfluxQL)

stslgov2 : "from(bucket: ...) |> range |> filter ..." (based of Flux)

slsgov2

```
func (timeserData *TimeSeriesClient) CreateTimeSeriesDB() (err error) {
        _, err = bucketsAPI.CreateBucketWithName(context.Background(), org, bucketName, domain.RetentionRule{
                EverySeconds: durationInt64,
        })

func (timeserData *TimeSeriesClientData) DeleteTimeSeriesDB() (err error) {
        err = bucketsAPI.DeleteBucket(context.Background(), bucket)

func (timeserData *TimeSeriesClientData) DropMeasurement(measurement string) (err error) {
        err = deleteAPI.DeleteWithName(ctx, orgName, bucketName, startTime, stopTime, predicate)

func (timeserData *TimeSeriesClient) Get(measurement, key string) (result interface{}, err error) {
        fluxQueryStr := fmt.Sprintf(`
        from(bucket: "%s")
    |> range(start: -%s)
    |> filter(fn: (r) => r._measurement == "%s" and r._field == "%s")
        `, timeserData.timeSeriesDb.Name, timeserData.timeSeriesDb.RetentionPolicy, measurement, key)
```

https://docs.influxdata.com/influxdb/v2.2/reference/syntax/flux/flux-vs-influxql/#influxql-and-flux-parity

ENV VAR

TIMESERIESDB_SERVICE_HOST : influxdb service host (e.g. http://localhost:8086)

TIMESERIESDB_SERVICE_TOKEN : influxdb service token for auth. (User can get admin's API token from WebUI in first. login > Data > API Tokens > admin's token)

TIMESERIESDB_SERVICE_ORG_NAME : influxdb service org name to use.

stslgov1