



ITRS Insights v1.4

Data Streams and Schema User Guide v1.0

Copyright 2016. ITRS Group Ltd. All rights reserved.

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of those agreements. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or any means electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of ITRS Group Ltd.

ITRS Group Ltd

6th Floor, The Bonhill Building, 15 Bonhill Street,

London, EC2A 4DN, UK

t: +44 (0)20 7638 6700

f: +44 (0)20 7256 5760

CONTENTS

CHAPTER 1

 ITRS Insights Data Streams & Schema User Guide 5

 Introduction 6

Schema? 9

GLOSSARY

INDEX

This page intentionally left blank.

CHAPTER 1

ITRS INSIGHTS DATA STREAMS & SCHEMA USER GUIDE

INTRODUCTION

Data **streams** are the lifeblood of Insights - they channel the data you want to analyse into Insights' mathematical brain, Valo.

Stream: append-only data (messages or events) coming from one or more external **Contributors**

All that you need to do before you can start running queries and searches is tell Insights what the data is and how it's structured. This is done through a simple **schema**.

Schema: a document of understanding that defines what data a stream contains - its structure and shape - and in what format.

Each stream has a **schema** that defines what information is expected, and in what format. Multiple streams can be organized into **Collections**.

As with everything in Insights, we've designed the process of bringing data in to be as simple as possible, regardless of whether it's streaming or historical. To this end, there are really only three major inputs required from you:

1. Choose a **repository** type
2. Prepare your data and create a **schema**
3. Import the data **stream** to Insights

Once you've completed these steps, you can either bring in additional streams or just get to work analysing.

This user guide will talk you through all three steps.

1. Choose a Repository

The **semi-structured repository (SSR)** is based on Lucene which provides very powerful text search capabilities. Even though Lucene is geared towards indexing text, it also has very good index support for numerical data. However, if the data contains purely numerical fields the TSR might be a better fit as a repository for this kind of data.

The **Time Series Repository (TSR)** is a custom built data repository designed to handle numerical time series oriented data streams. By time series we mean a series of data points each of which have an associated time stamp.

Before you import a data stream into Insights, and assuming you want to store your data, you must choose a type of repository to store it in.

Insights stores data in two different repositories:

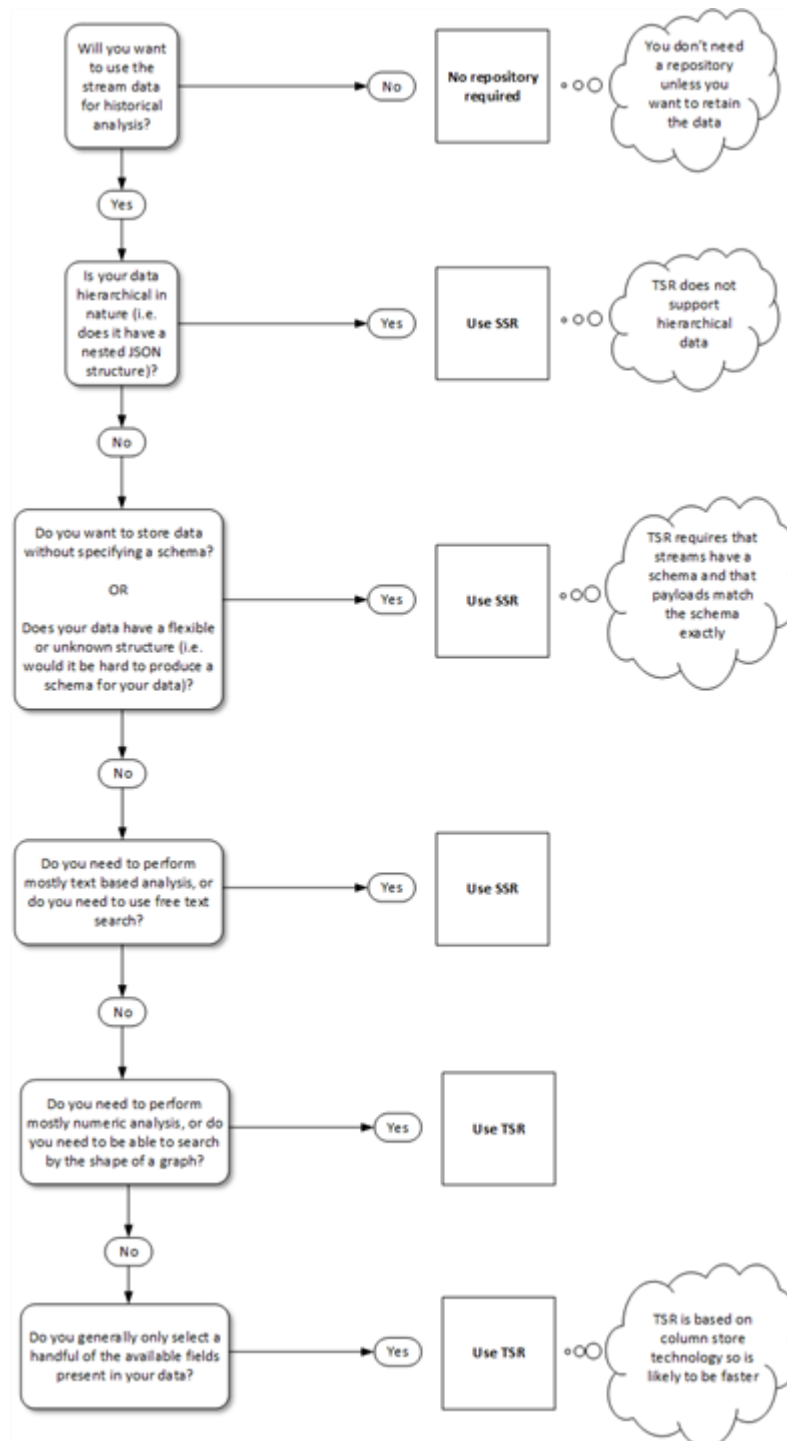
1. The **Semi-Structured Repository (SSR)**
2. The **Time Series Repository (TSR)**

Note: streaming data that is not stored does not require a repository at all.

Which is best for you is a decision largely driven by what you want to do with your data. Historical analysis? Textual analysis?

Check out our tech reference here, [Repositories - TSR & SSR](#) to fully understand what the differences are, or skip the technical bit and go through our flowchart to help you decide.

Flow Diagram for Choosing a Repository



Text Version

Historical analysis?

Will you need to perform historical analysis on the stream's data?

No -> i.e. you only need to perform real-time analysis and have no need to retain the data - then you don't need to use a repository at all.

Yes -> Continue reading

Hierarchical data?

Is your data hierarchical in nature (i.e. does it have a nested JSON structure)?

Yes -> you need to use the **SSR**, because the TSR doesn't support hierarchical data.

Schema?

Does your data have a flexible or unknown structure (i.e. would it be hard to produce a schema for your data)?

or

Do you want to begin storing data without specifying a schema?

Yes -> you need to use the **SSR**, because the TSR requires that streams have a schema and that payloads match the schema exactly

Textual analysis?

Do you need to perform mostly text based analysis, or do you need to use free text search?

Yes -> you should use the **SSR**.

Numeric analysis?

Do you need to perform mostly numeric analysis, or do you need to be able to search by the shape of a graph?

Yes -> you should use the **TSR**.

Queries Only?

Do you generally only select a handful of the available fields present in your data?

Yes -> you should use the **TSR** as it is based on column store technology so is likely to be

2. Prepare your DATA & Create a SCHEMA

Schema: a document of understanding that defines what data a stream contains - its structure and shape - and in what format.

One challenge when you have a number of different contributors is to ensure that everyone is speaking the same language and meaning the same thing, otherwise comparisons and analytics will have limited value.

To do this, we use a document of understanding (a **schema**) that defines what data the stream contains in what format.

For instance, we may want the stream to provide information as data points (also known as payloads) about the time of day, the type of appliance, and the current consumption.

An Insights schema contains three elements:

1. A user defined schema `version`
2. A `config` section
3. A `topDef` section that defines the data schema

This is the basic Insights schema:

```
{
  "version": "1.0.0",
  "config": {
    "key": []
  },
  "topDef": {}
}
```

topDef

The topDef section can contain a simple or a complex type.

A simple type is defined as:

```
"topDef": { "type" : "double" }
```

A complex type is defined as:

```
"topDef": {  
  "type" : "record",  
  "properties" : {  
    "host" : { "type" : "string" }  
  }  
}
```

Example Schema

For an example the first version of a schema which defines a schema for a stream of data coming from our smart houses – with 4 elements in each payload, the date, the house number, the energy consumed in the last 12 hours and the quality of the data – may look like this:

```
{  
  "schema": {  
    "version": "1.0.0",  
    "config": {},  
    "topDef": {  
      "type": "record",  
      "properties": {  
        "date": { "type": "date", "optional":  
"false" },  
        "energyConsumed": { "type": "double",  
"optional": "false" },  
        "houseNumber": { "type": "byte",  
"optional": "true" },  
        "quality": { "type": "boolean",  
"optional": "true" }  
      }  
    }  
  }  
}
```

Note: Valid field identifiers must:

1. Start with a character
2. Not contain spaces
3. Only contain alpha numeric characters and underscores

Supported Schema Types & Examples

byte	8 bit integer	200
short	16 bit integer	1003
int	32 bit integer	100345
long	64 bit integer	100345000
double	Double precision float	23.5
date	A date field in ISO8601 Format	2015-01-24
time	A time field in ISO8601 Format	13:04:16
datetime	A datetime field in ISO8601 Format	2015-01-24T13:04:16
duration	A duration field	12 seconds
string	A string field	This is a log message
boolean	A Boolean field	true
geoCircle	A geographic circle field	S33d 51m 23s E151d 12m 53s 0.0001
geoPoint	A geographic point field	S33d 51m 23.943s E151d 12m 53.8812s
geoRect-angle	A geographic rectangle field	S33d 51m 24s E151d 12m 53s S33d 51m 23s E151d 12m 53.5s
ip	An IP4/IP6 field	192.10.0.120
uuid	A uuid field	4957aac1-c11d-44bf-9d79-ca23dd213cf6
contributor	A Contributor ID represented as a string (a Contributor takes the form: <server- >/contributors/<tenant>/<contributorType>/<contributorId>)	c4d29de9-d7a3-4040-91d2-e3683cbac05a
email	An email address field	john.-doe@domain.com
uri	A URI	www.google.com
tags	A tags field	
record	A record / object field (a record represents a complex type. Anything that has a document-like structure (think JSON) can be represented as a record.)	
array	An array field	

Example Schema

```
{
  "type" : "record"
  "properties" : {
    "contributor" : { "type" : "contributor",
"definition": "probes" },
    "timestamp" : { "type" : "datetime" },
    "host" : {
      "type" : "record",
      "properties" : {
        "name" : { "type" : "string" },
        "ip" : { "type" : "ip" },
      }
    },
    "os" : {
      "type" : "record",
      "properties" : {
        "name" : { "type" : "string" },
        "version" : { "type" : "string" }
      }
    }
  }
}
```

Example Data

```
{
  "contributor" : "df625733-6fd8-465b-ac60-76fd23a52b4a",
  "timestamp" : "2015-03-12 14:30:07",
  "host" : {
    "name" : "ABC123",
    "ip" : "192.10.0.120",
    "mac" : "00-B0-D0-86-BB-F7"
  },
  "os" : {
    "name" : "Windows",
    "version" : "Vista"
  }
}
```

Typed Array

```
{
  "type" : "sequence",
  "underlying" : {
    "type" : "..."
  }
}
```

Example Schema

```
{
  "type" : "sequence",
  "underlying" : {
    "type" : "double"
  }
}
```

```
}
```



Data

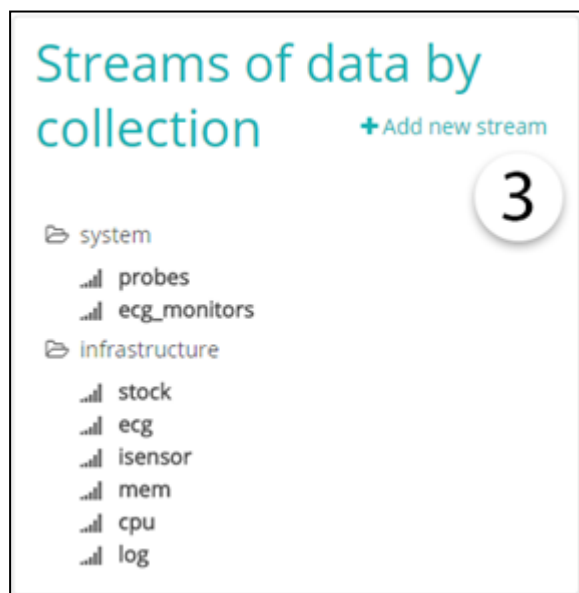
```
[30.4, 30.8, 20.0]
```

3. Import Data Stream

To import a data stream, you have to create your schema, add it to the Insights Stream Editor, then select the repository type for the data to be stored in (None is selected by default, and this is used for streaming data).

How to Create a Schema

1. Run **ITRS Insights**
2. Click **Home**  to select the **Home Page tab**
3. Hover your mouse cursor over the **Streams of Data by Collection Panel**, then click **+Add new stream** to open a **New Stream tab**
4. Enter a **Stream Name**
5. Select a **Collection** from the **drop-down dialog**, or click inside the text field and type the name of the **Collection** that you want to create
6. Type or paste in the **schema** for your data stream. Use the YAML or JSON buttons to switch between display format
7. Choose a repository format in the **Save Stream In panel**
8. Click **Create Schema** 



NEW STREAM

Export Session

New Stream

SchemaTagging

Stream Name

Stream Collection

4

My New Stream

5

Test Collection

Stream Schema

6

```
{
  "topDef": {
    "type": "record",
    "properties": {
      "date": { "type": "date", "optional": "false" },
      "maxTemp": { "type": "double", "optional": "false" },
      "daysOfAccumulation": { "type": "byte", "optional": "true" },
      "airQuality": { "type": "boolean", "optional": "true" }
    }
  }
}
```

YAMLJSON

7

SAVE STREAM IN

SSR

TSR

NONE

8

CREATE SCHEMA

This page intentionally blank.

GLOSSARY

C

Collector

New term for a lightweight monitoring agent, which is deployed on every managed Node. See Netprobe.

Contributor

An external source that provides one or more data streams to Valo. Contributors can be grouped together by common attributes to create Domains.

D

Dashboard

Dashboards provide an aggregated view of Data Visualizations

Domain

Two or more Contributors grouped together by a common attribute. Domains help to organise and group different data sources in intuitive ways.

G

Gateway

A network node that provides access to another network that uses different protocols and enables transmitted data to use its routing paths.

I

Insights

A streaming big data analytics platform that simplifies the complexity involved in analysing vast amounts of data at speed. Insights combines big data storage with a real time computation engine and in-built machine learning and algorithms.

ITRS Geneos

A real-time monitoring tool for managing increasingly complex and interconnected IT estates. Built for financial services and trading organisations, it

collects a multitude of data relating to the performance of the servers, infrastructure, connectivity and applications, analyses it to detect anything untoward, and presents it in relevant, intuitive visualisations to help diagnose and fix issues quickly.

N

Netprobe

Lightweight monitoring agent, which is deployed on every managed Node

Notebook

A place in which to store a query or search, or a pipeline of queries or searches.

S

Schema

Document of understanding that defines what data a stream contains - its structure and shape - and in what format.

SSR

The semi-structured repository (SSR) is based on Lucene which provides very powerful text search capabilities. Even though Lucene is geared towards indexing text, it also has very good index support for numerical data. However, if the data contains purely numerical fields the TSR might be a better fit as a repository for this kind of data.

Stream

A stream is made up of data (messages or events) coming from one or more external contributors. Each stream has a schema that defines what information, in what format is expected. Streams are append only.

T

Tenant

The highest level of data grouping. Within a company, it may be different departments such as Sales or Engineering. If a government is collecting data from its cities, the tenant could be the city name.

Transformational Pipeline

Each query instruction takes an input and produces an output via some form of transformation.

TSR

The Time Series Repository (TSR) is a custom built data repository designed to handle numerical time series oriented data streams. By time series we mean a series of data points each of which has an associated time stamp.

V

Valo

Real-time analytics on data streams

INDEX

S

Schema 11

This page intentionally blank.