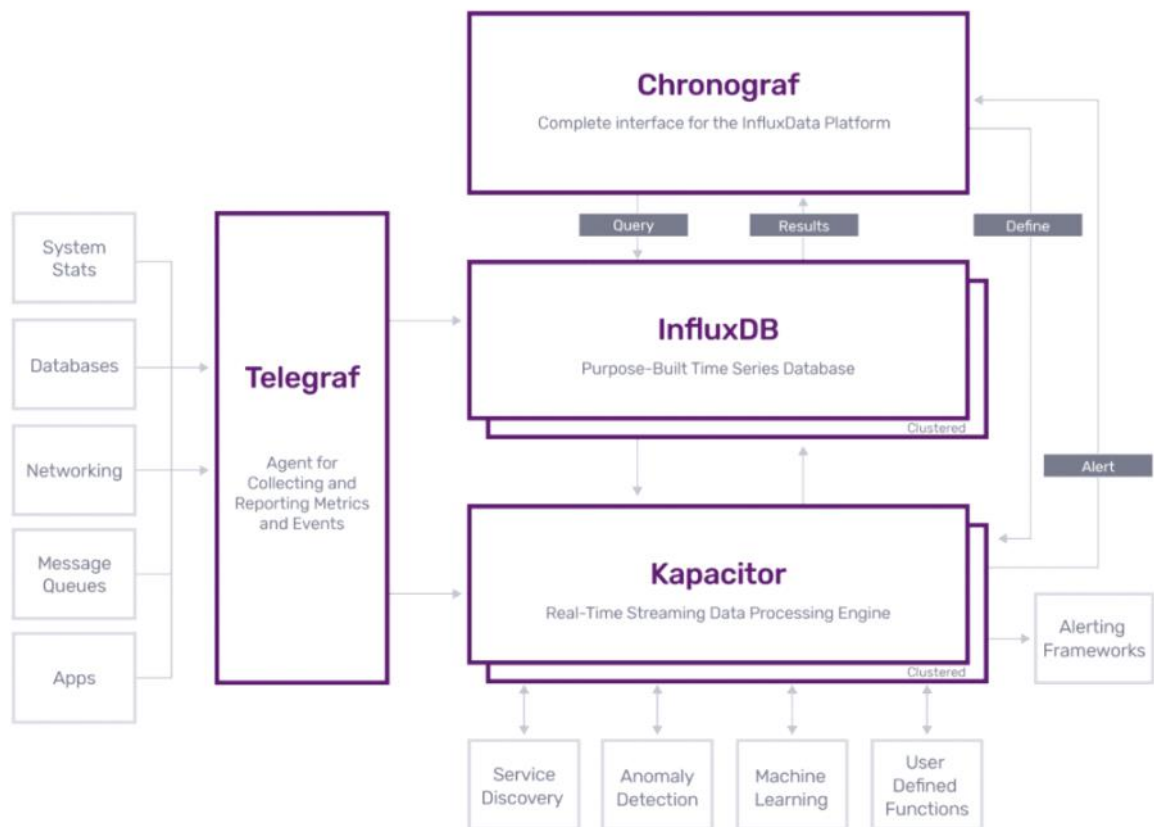


1 - Introduzione

domenica 14 febbraio 2021 03:46

L'applicazione **edge-influxdata-stack** consente di portare direttamente su Edge Device i servizi di database dello stack di **InfluxData**, dedicato espressamente ai dati come **serie temporali**. Questo insieme di servizi è chiamato "**TICK Stack**", dai nomi dei 4 servizi utilizzati:



- **Telegraf**: è un server agente basato su molteplici plugin per la raccolta e la segnalazione di metriche. (In questa applicazione non considereremo il servizio Telegraf poiché non è necessario per il nostro scopo, ma può essere integrato dall'utente semplicemente aggiungendolo nel file docker-compose insieme agli altri servizi)
- **InfluxDB**: è un database di serie temporali costruito per gestire alti carichi di scrittura e query. InfluxDB è un datastore personalizzato ad alte prestazioni scritto specificamente per dati cronometrabili, e particolarmente utile per casi d'uso come il monitoraggio IoT e l'analisi in tempo reale. InfluxDB offre anche un linguaggio di query simile a SQL per interagire con i dati
- **Chronograf**: è l'interfaccia utente amministrativa e il motore di visualizzazione dello stack. Rende facile l'impostazione e la manutenzione del monitoraggio e degli avvisi per l'infrastruttura. È semplice da usare e include modelli e librerie che ti permettono di costruire rapidamente dashboard con visualizzazioni in tempo reale dei tuoi dati e di creare facilmente regole di allarme e automazione
- **Kapacitor**: è un motore di elaborazione dati nativo. Può elaborare sia i dati in streaming che quelli in batch da InfluxDB. Permette di inserire una logica personalizzata o funzioni definite dall'utente per elaborare gli avvisi con soglie dinamiche, abbinare le metriche ai modelli, calcolare le anomalie statistiche ed eseguire azioni specifiche basate su questi avvisi

Per maggior informazioni su TICK Stack consulta la pagina ufficiale InfluxData (<https://www.influxdata.com/time-series-platform/>).

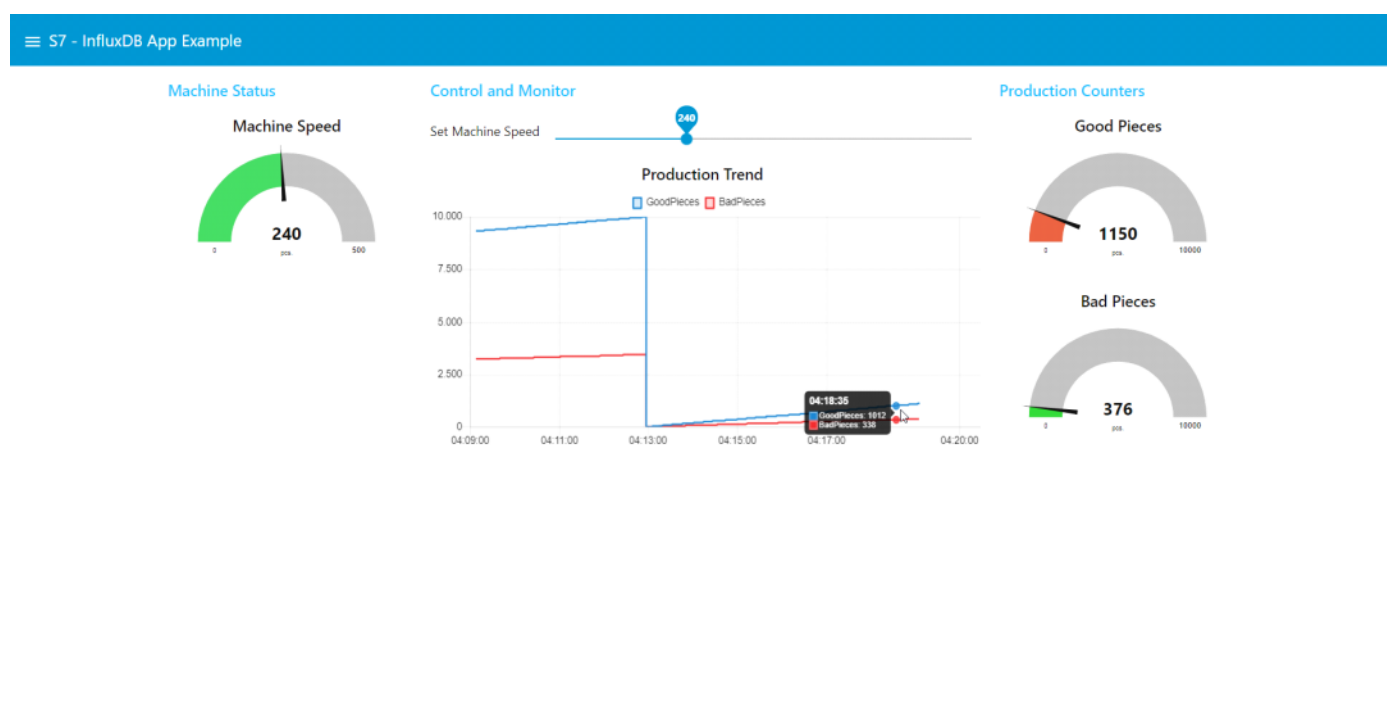
Prima di iniziare

Questa guida descrive come utilizzare e installare l'App **edge-influxdata-stack**.

Verificare i **requisiti** necessari alla pagina [2 –Requisiti App](#) prima di procedere all'installazione. I dettagli per la procedura di **installazione** sono disponibili alla pagina [3 – Installazione App](#).

Per tutti i **dettagli** sui servizi InfluxData utilizzati nell'applicazione e per i tutti **riferimenti** online al loro utilizzo consultare il capitolo [4 - Utilizzo App](#).

L'applicazione viene fornita con un **esempio applicativo** alla pagina [5 - Esempio Applicativo](#), che mostra come poter gestire il flusso di dati con il database InfluxDB tramite l'applicazione SIMATIC Flow Creator (Node-RED) ed il nodo dedicato **node-red-contrib-influxdb**. Inoltre sarà mostrato come visualizzare i dati letti tramite le funzionalità della dashboard di Chronograf e tramite la dashboard di SIMATIC Flow Creator. Per maggiori informazioni sull'installazione del nodo dedicato a InfluxDB consulta il capitolo [A - Installazione nodo NodeRED InfluxDB](#).



Al capitolo [6 - Costruzione App](#) è indicato nel dettaglio come questa applicazione è stata costruita utilizzando l'ambiente Docker.

2 - Requisiti App

Friday, February 5, 2021 10:48 PM

Requisiti Hardware

- L' applicazione **edge-influxdata-stack** è compatibile solo con dispositivi **SIEMENS** dotati di funzionalità **Industrial Edge** abilitata.

Requisiti Software

- L' applicazione **edge-influxdata-stack** necessita di **1800 MB** di **RAM** per il funzionamento, così divisa tra i vari microservizi:

Nome servizio	Limite Memoria
<i>edge-influxdb</i>	1200 MB
<i>edge-chronograf</i>	400 MB
<i>edge-kapacitor</i>	200 MB

Nota: Questi limiti sono stati fissati per un volume di dati di media intensità nel database InfluxDB e per garantire un utilizzo medio costante delle dashboard di Chronograf, ma possono essere modificati in base alle proprie esigenze agendo sul file docker-compose e quindi sulla configurazione dell'app nel software Edge App Publisher, creando una versione personalizzata di questa applicazione.

3 - Installazione App

enerdì 5 febbraio 2021 21:47

L'applicazione **edge-influxdata-stack** viene fornita con il pacchetto app precostruito **edge-influxdata-stack_x.x.x.app** (in base alla versione fornita x.x.x) che può essere installato specificamente sugli Edge Device utilizzando **SIMATIC Edge**.

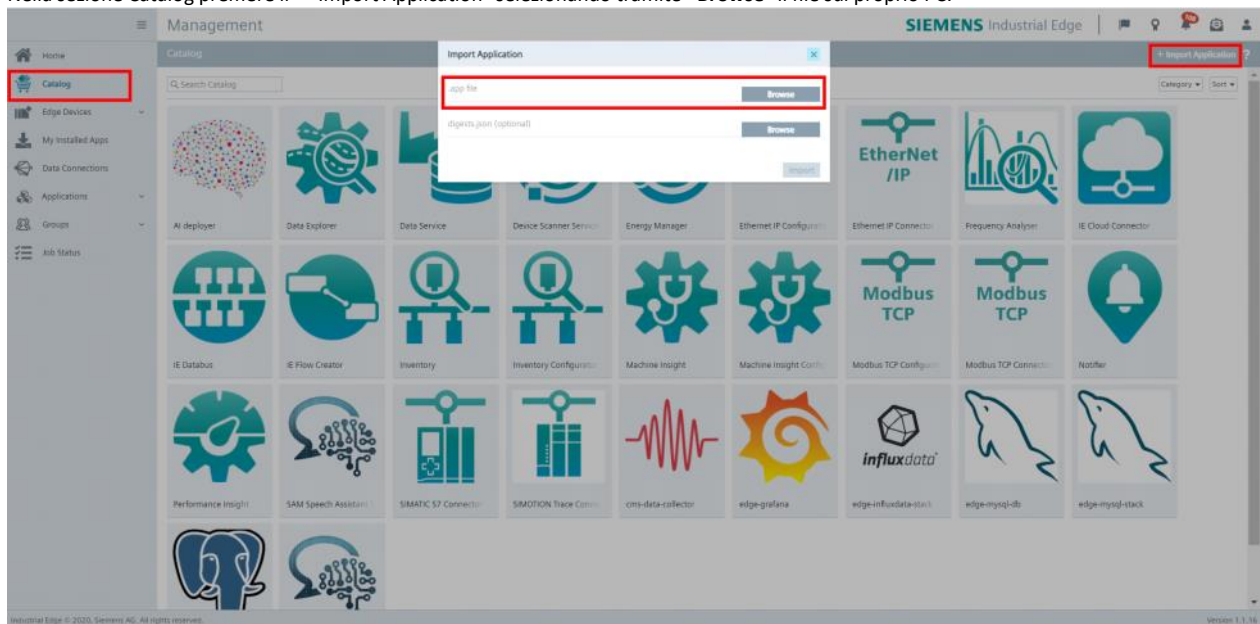
Prerequisiti

- Un **Edge Device** correttamente configurato e con onboarding su un sistema Industrial Edge Management (**IEM**) accessibile e con i req
- Se si utilizza Industrial Edge App Publisher per modificare/pubblicare l'applicazione sarà necessaria la connessione ad un **Docker Engine** per il software **Industrial Edge App Publisher**.

Caricamento App su IEM

A - Import file app in IEM Portal

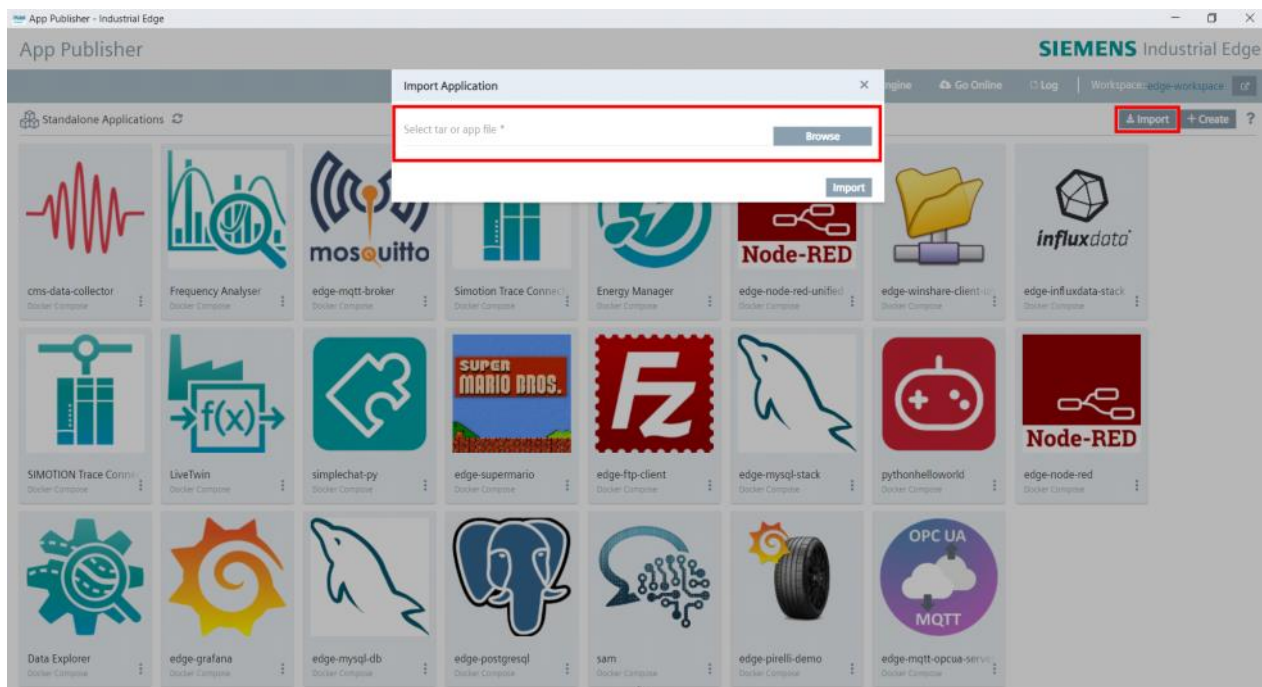
1. Copiare il file scaricato di **edge-influxdata-stack_x.x.x.app** (in base alla versione x.x.x) sul proprio PC.
2. Aprire la pagina web **IEM Portal** del sistema **Industrial Edge Management** che controlla gli Edge Device desiderati.
3. Nella sezione Catalog premere il "+ Import Application" selezionando tramite **"Browse"** il file sul proprio PC.



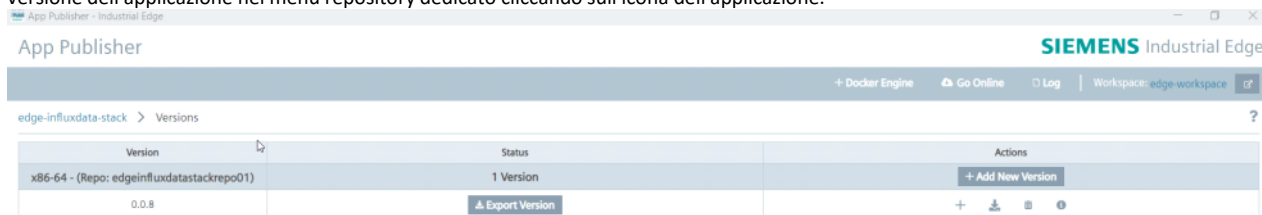
4. Attendi che l'app sia importata. E' possibile verificare lo stato tramite il tasto **"Tasks"** in alto a destra.

B - Import file app in Edge App Publisher

1. Copiare il file scaricato di **edge-influxdata-stack_x.x.x.app** (in base alla versione x.x.x) sul proprio PC.
2. Aprire il software **Industrial Edge App Publisher**
3. Tramite il tasto **Import** aprire il menù per importare il file app fornito e selezionare il file sul proprio PC tramite il tasto **Browse**.



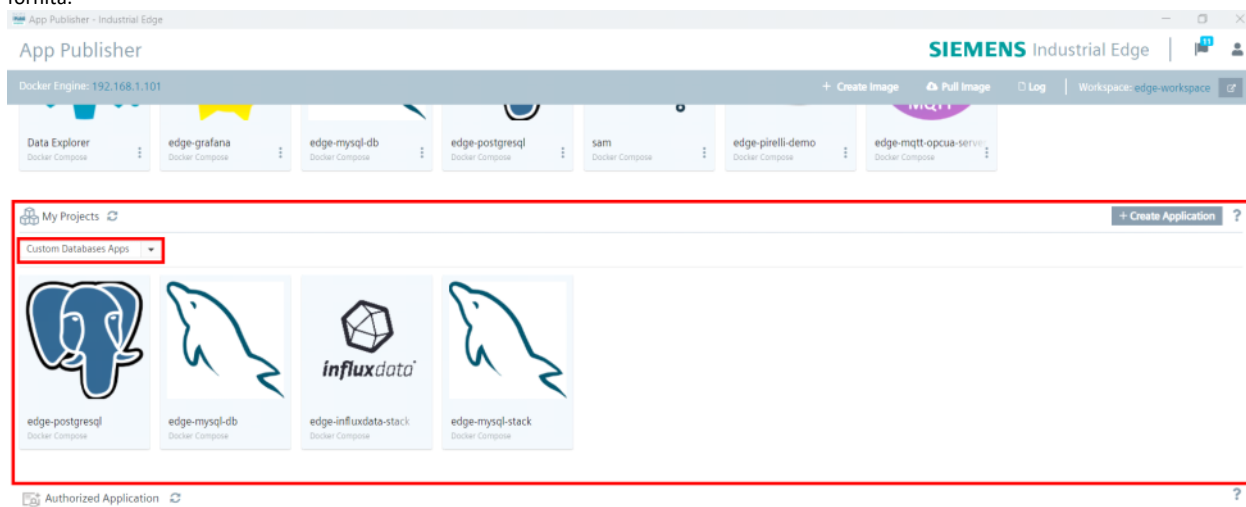
4. Attendere il completamento dell'importazione dell'applicazione.
5. Al termine l'applicazione sarà visibile nella sezione Standalone Applications del software Industrial Edge App Publisher. E' possibile anche creare una nuova versione dell'applicazione nel menù repository dedicato cliccando sull'icona dell'applicazione.



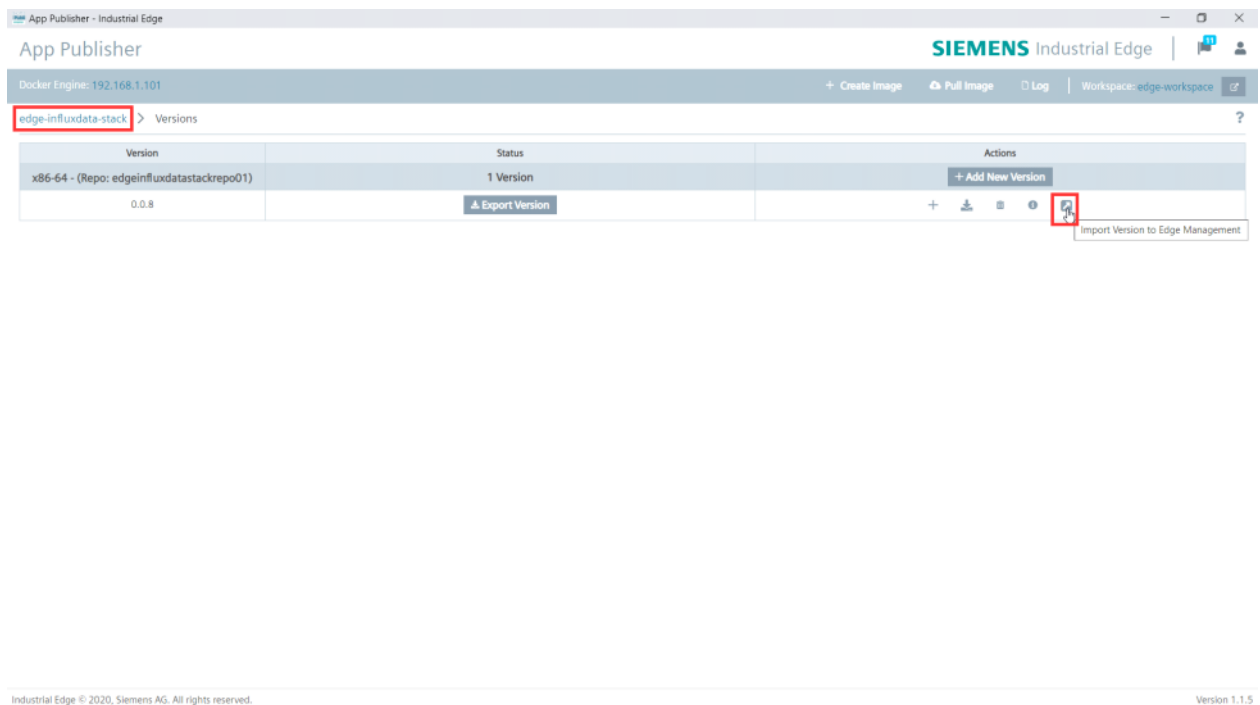
6. Connettersi al proprio IEM tramite il tasto "Go Online", inserendo l'indirizzo <https> del proprio IEM e premendo su **Connect**. Verrà richiesto il **login** da parte dell'utente.



7. Una volta connessi, appariranno i Progetti e le Applicazioni attualmente su IEM. Scegliere o creare un nuovo progetto dove poter inserire l'applicazione fornita.



8. Aprire il menù della repository dell'app edge-influxdata-stack premendo nella sezione Standalone Application l'icona dell'applicazione.
9. Premere il tasto "Import Version to Edge Management" per importare l'applicazione su IEM.



- Selezionare un progetto su cui caricare l'applicazione e premere "Create" per eseguire il task di import della applicazione su IEM. Attendere il completamento del task. Lo stato del task è visibile nella sezione Tasks.

Import Version

Select projects
Custom Databases Apps

Select Category
Other

Website
www.siemens.it

☐ Use Edge Core Auth Service (optional)
Do not use Edge Core Auth Service.

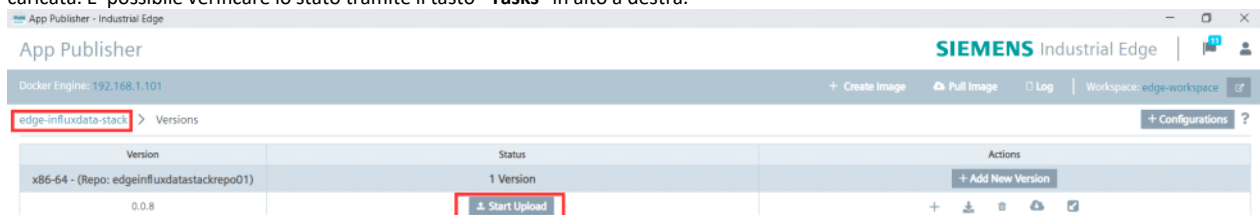
☐ External Configurator
No external configurator.

Version
Major: 0, Minor: 0, Maintenance: 8

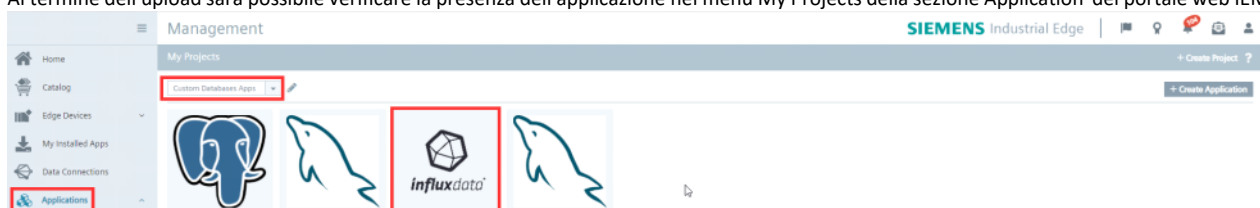
Release Notes (optional)
- Endpoint from internal Apps -> edge-influxdb:8086
- Endpoint from external -> [core-name]:38086
- Chronograf Web UI reachable at fledge-chronograf

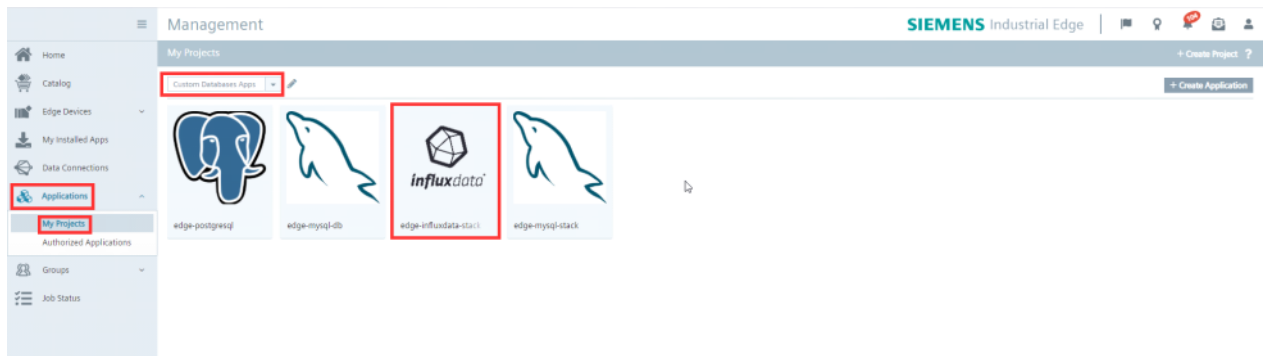
Create

- Recarsi ora nella sezione My Projects del software Edge App Publisher, selezionare il progetto in cui è localizzata l'applicazione edge-influxdata-stack e entrare nel menù della repository.
- Premere Start Upload per iniziare il trasferimento della versione desiderata verso il catalogo applicazioni locale dell'istanza IEM. Attendi che l'app sia caricata. E' possibile verificare lo stato tramite il tasto "Tasks" in alto a destra.

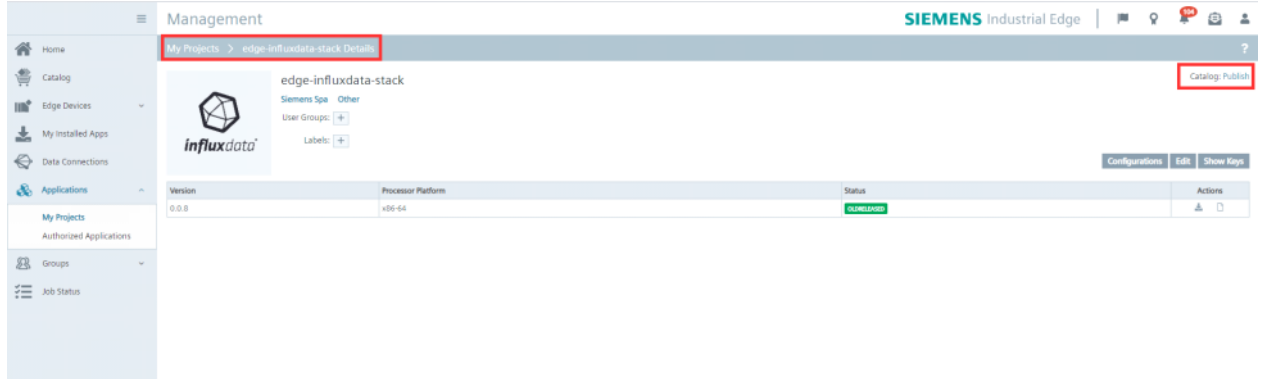


- Al termine dell'upload sarà possibile verificare la presenza dell'applicazione nel menù My Projects della sezione Application del portale web IEM

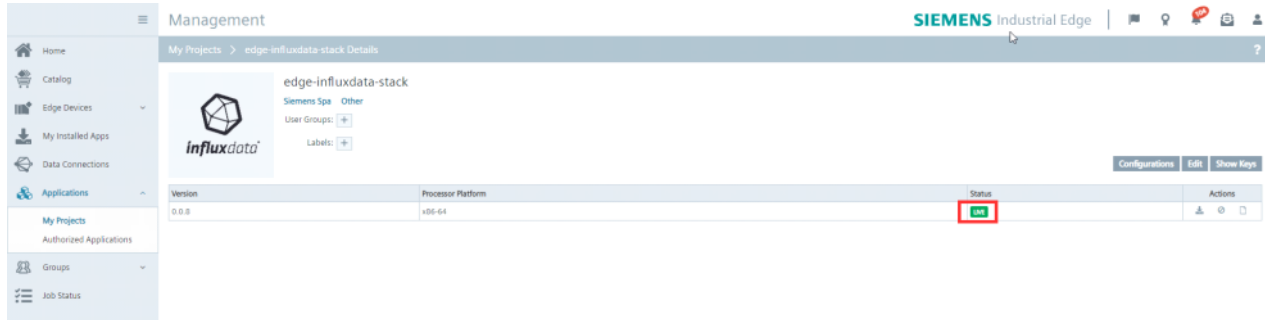




- Premere il tasto "Publish" in alto a destra per scegliere la versione da rendere pubblica nel Catalog IEM e pubblicare l'applicazione.



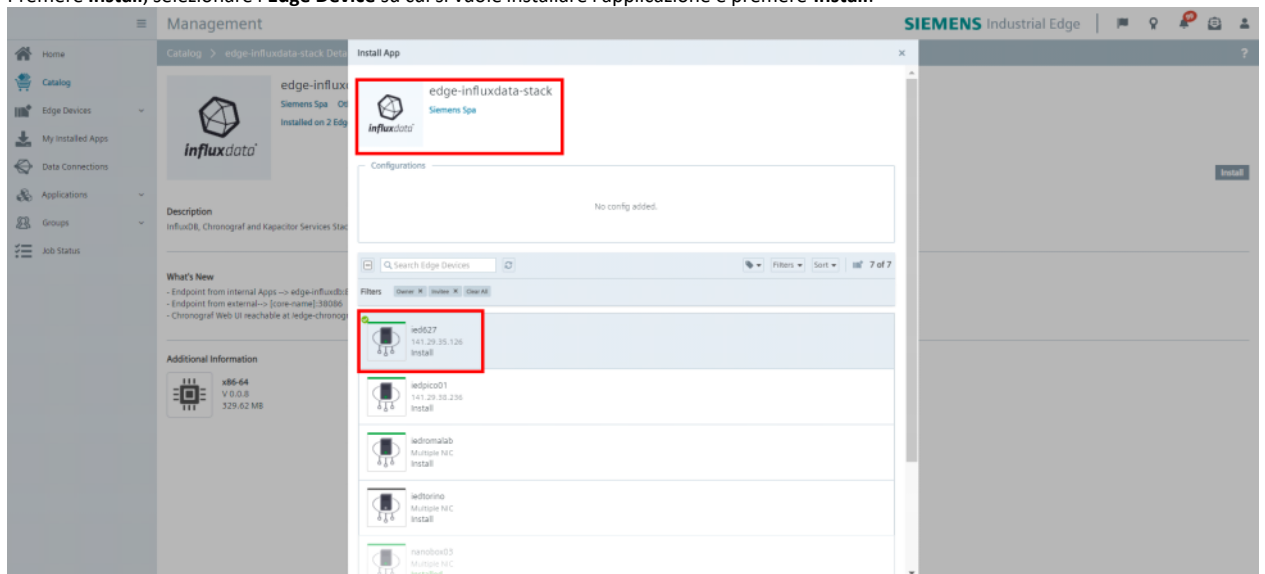
- Verificare che la versione desiderata dell'applicazione sia ora nello stato "Live" e che nella sezione Catalog dello IEM Port al sia visibile l'applicazione creata con la versione corretta.



Installazione App su Edge Device

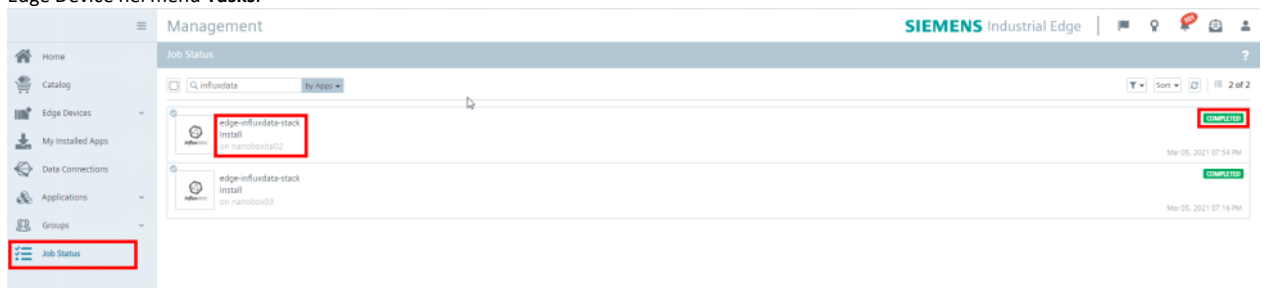
Una volta importata l'app all'interno del proprio **IEM Catalog** seguendo il paragrafo precedente, sarà possibile selezionarla per l'installazione su uno degli Edge Device gestiti.

- Dalla sezione **Catalog** selezionare l'app **edge-influxdata-stack** per aprire la finestra dedicata.
- Premere **Install**, selezionare l'**Edge Device** su cui si vuole installare l'applicazione e premere **Install**.





3. Verrà creato un **task** sul Edge Device per l'installazione. Attendi che l'app sia installata. E' possibile verificare lo stato all'interno della sezione **Job Status** o sul Edge Device nel menù **Tasks**.



4. Ora è possibile usare l'applicazione dall'Edge Device configurato.

4 - Utilizzo App

lunedì 1 marzo 2021 20:35

Come specificato nel capitolo 1 - Introduzione, l'applicazione **edge-influxdata-stack** è basata sullo stack di applicazioni InfluxDB, Chronograf e Kapacitor.

Di seguito sono presentati nel dettaglio i diversi servizi che compongono l'applicazione e le loro diverse modalità di utilizzo.

InfluxDB

InfluxDB è un **database di serie temporali** progettato per gestire elevati carichi di scrittura e di query. È un componente del **TICK Stack** (<https://www.influxdata.com/time-series-platform/>). InfluxDB è pensato per essere utilizzato come "backing store" per qualsiasi caso d'uso che coinvolga **grandi quantità di dati con timestamp**, compreso il monitoraggio DevOps, le metriche delle applicazioni, i dati dei sensori IoT e l'analisi in tempo reale.

- Per maggiori dettagli riguardo ai database di serie temporali consultare il link "*What is a timeseries database?*" (<https://www.influxdata.com/time-series-database/>).
- La **documentazione completa** di InfluxDB è disponibile al seguente link "*InfluxDB Documentation*" - (<https://docs.influxdata.com/influxdb/v1.8/>).

Ecco alcune delle **caratteristiche** che InfluxDB supporta attualmente e che lo rendono un'ottima scelta per lavorare con i dati delle serie temporali.

- Data storage personalizzato ad alte prestazioni scritto appositamente per i dati delle serie temporali. Il motore TSM permette un'alta velocità di caricamento e la compressione dei dati.
- Scritto interamente in Go. Si compila in un singolo binario senza dipendenze esterne.
- API HTTP di scrittura e interrogazione semplici e performanti.
- Supporto dei plugin per altri protocolli di ingestione dei dati come Graphite, collectd e OpenTSDB.
- Espressivo linguaggio di interrogazione SQL-like fatto su misura per interrogare facilmente i dati aggregati.
- I tag permettono alle serie di essere indicizzate per query veloci ed efficienti.
- Le politiche di conservazione scadono automaticamente e in modo efficiente i dati obsoleti.
- Le query continue calcolano automaticamente i dati aggregati per rendere le query frequenti più efficienti.

InfluxDB mette a disposizione due modalità di **interrogazione** del database, con due **linguaggi di Query** chiamati **FLUX** e **InfluxQL**. L'applicazione **edge-influxdata-stack** e l'**esempio applicativo** al capitolo [5 - Esempio Applicativo](#) utilizzano il linguaggio di Query InfluxQL, che è molto simile ai linguaggi query utilizzati nei classici database SQL.

- Per analogie e differenze tra un classico database SQL e il database InfluxDB per serie temporali consultare il seguente link "*Compare InfluxDB to SQL databases*" - (<https://docs.influxdata.com/influxdb/v1.8/concepts/crosswalk/>).
- La guida completa del linguaggio di query InfluxQL è consultabile al link "*Influx Query Language (InfluxQL)*" - (https://docs.influxdata.com/influxdb/v1.8/query_language/).

Chronograf

Chronograf è l'applicazione **grafica web** di InfluxData. Chronograf consente di **amministrare** gli altri componenti del TICK Stack per **visualizzare** i dati raccolti e quelli di monitoraggio utilizzando query personalizzate e creare facilmente **regole** di allarme e automazione del flusso dei dati.

- Per maggiori informazioni sul servizio Chronograf consulta il link ufficiale "*What is Chronograf?*" - (<https://www.influxdata.com/time-series-platform/chronograf/>).
- La guida introduttiva all'utilizzo di Chronograf è disponibile al link "*Getting Started With Chronograf*" - (<https://docs.influxdata.com/chronograf/v1.8/introduction/getting-started/>).

Ecco alcune delle **caratteristiche** principali del servizio Chronograf:

- Monitora i dati dell'applicazione con dashboard pre-create da Chronograf
- Crea dashboard personalizzate complete di vari tipi di grafici e variabili
- Indaga i dati con l'esploratore di dati e i modelli di query di Chronograf
- Gestione degli avvisi insieme a Kapacitor, il framework di elaborazione dati di InfluxData per la creazione di avvisi, l'esecuzione di lavori ETL e il rilevamento di anomalie nei tuoi dati.
- Genera avvisi di soglia sui tuoi dati, visualizza tutti gli avvisi attivi su un dashboard e invia avvisi ad altri servizi di gestione eventi tra cui *Slack*, *Telegram* e altri.
- Crea e cancella i database e implementa politiche di conservazione (*Retention Policies*)
- Visualizza le query in esecuzione e impedisce alle query inefficienti di sovraccaricare il tuo sistema
- Crea, cancella e assegna permessi agli utenti

Sono disponibili diverse **guide** su come utilizzare Chronograf per implementare le funzionalità descritte sopra al seguente link "*Guides for Chronograf*" - (<https://docs.influxdata.com/chronograf/v1.8/guides/>)

Kapacitor

Kapacitor è il **framework** del TICK Stack per l'**elaborazione** dei dati che rende facile la creazione di avvisi, l'esecuzione di lavori **ETL** e il rilevamento di anomalie.

- Per maggiori informazioni sul servizio Kapacitor consulta il link ufficiale "Why use Kapacitor?" - (<https://www.influxdata.com/time-series-platform/kapacitor/>)
- La documentazione ufficiale di Kapacitor è disponibile al link "*Kapacitor 1.5 documentation*" - (<https://docs.influxdata.com/kapacitor/v1.5/>)

Ecco alcune delle **caratteristiche** che Kapacitor supporta attualmente e che lo rendono un'ottima scelta per l'elaborazione dei dati:

- Elabora sia dati in streaming che dati in batch.
- Interroga i dati da InfluxDB in modo schedato
- Eseguire qualsiasi trasformazione attualmente possibile in InfluxQL.
- Memorizza i dati trasformati in InfluxDB.
- Aggiunge funzioni personalizzate definite dall'utente per rilevare le anomalie.
- Integrabile con altri servizi come HipChat, OpsGenie, Alerta, Sensu, PagerDuty, Slack e altri.

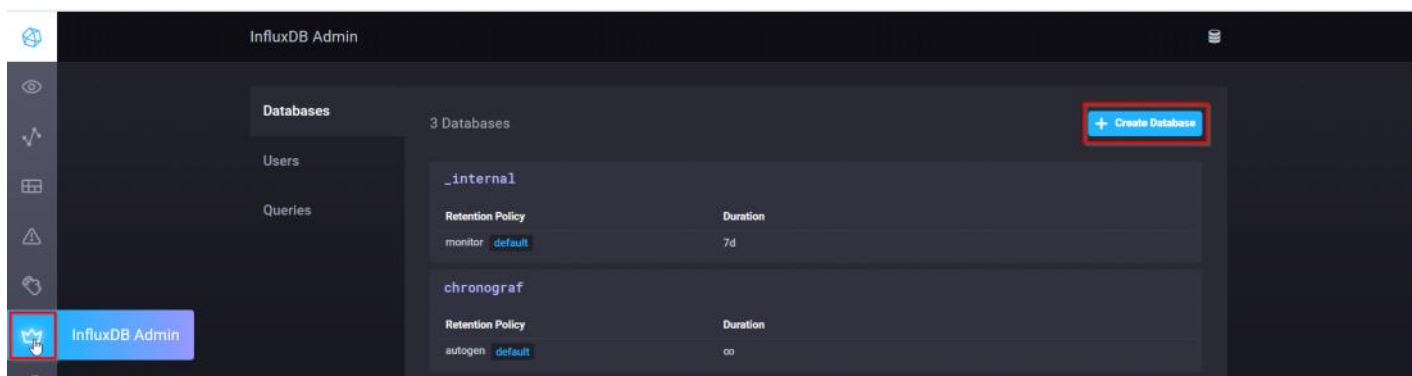
Sono disponibili diversi tutorial e guide all'utilizzo di Kapacitor ai link ufficiali "*Kapacitor Guides*" - (<https://docs.influxdata.com/kapacitor/v1.5/guides/>) e "*Working with Kapacitor*" - (<https://docs.influxdata.com/kapacitor/v1.5/working/>).

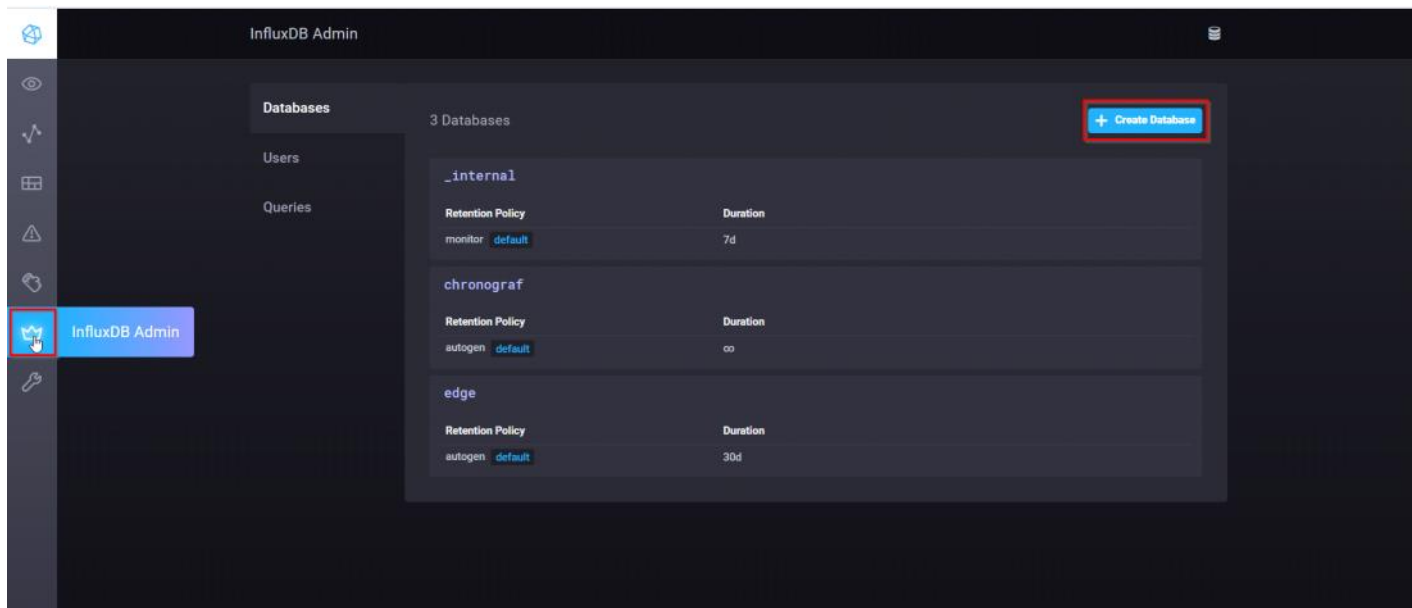
Di seguito sono presentate alcune funzionalità del TICK Stack:

Creare un nuovo database con Chronograf

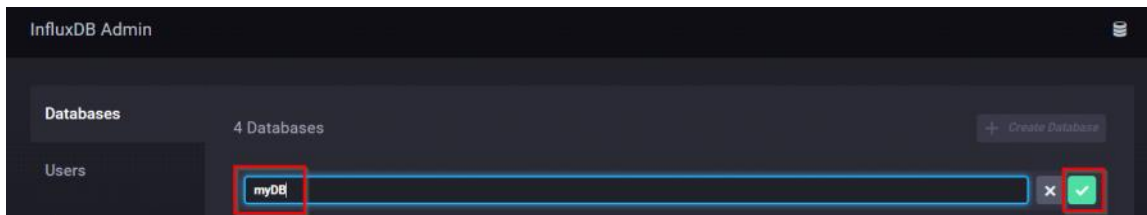
Per creare un nuovo database all'interno di InfluxDB è possibile utilizzare il servizio Chronograf, seguendo la seguente procedura:

1. Connettersi all'interfaccia Chronograf utilizzando l' URL [https://\[device-ip-address\]/edge-chronograf/](https://[device-ip-address]/edge-chronograf/)
2. Nella sezione "**InfluxDB Admin**" premere il tasto "**Create Database**"





3. Inserire il nome del nuovo database da aggiungere e premere il tasto "Spunta"



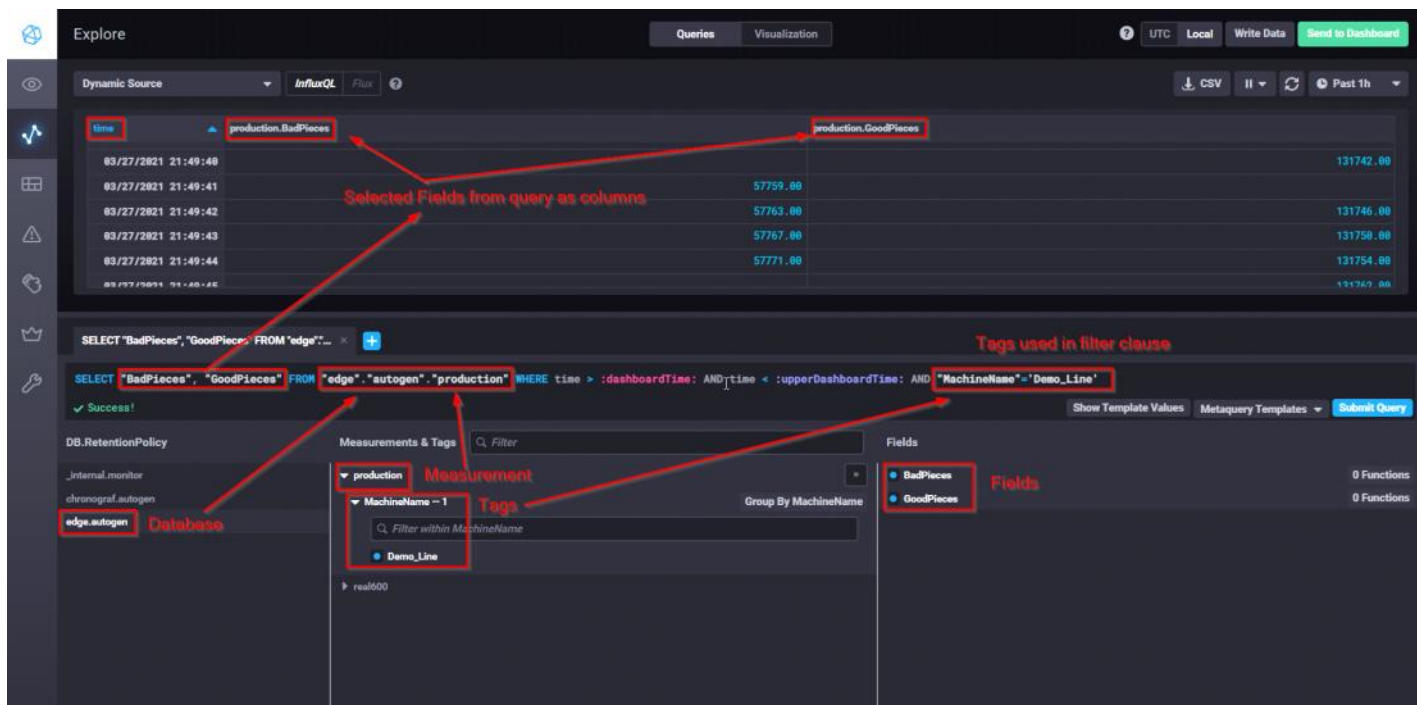
4. Il nuovo database viene creato correttamente.

Leggere dati dal database con Chronograf

Per poter consultare velocemente i dati raccolti o poter testare delle query risulta utile utilizzare Chronograf e la sua funzionalità "Explore". Qui sarà possibile utilizzare i linguaggi di query InfluxDB tra cui InfluxQL per poter interrogare il database e visualizzare i dati in diverse modalità grafiche in modo guidato.

Di seguito sono presentati i passi per poter effettuare una query con il menù guidato all'interno della sezione "Explore":

1. Connettersi all'interfaccia Chronograf utilizzando l' URL [https://\[device-ip-address\]/edge-chronograf/](https://[device-ip-address]/edge-chronograf/)
2. Nella sezione "**Explore**" selezionare nella colonna "**DB.RetentionPolicy**" il nome del database e della relativa retention policy da interrogare
3. Selezionare poi nella colonna "**Measurement & Tags**" la **misura** interessata ed eventualmente quali campi **Tags** saranno utilizzati nel filtro della query
4. Selezionare quindi i campi **Fields** nella colonna "**Fields**" che si vogliono leggere ed applicare eventualmente le funzioni di **aggregazione** predefinite.
5. Apparirà una query nel riquadro **editor**, che eventualmente è possibile editare al fine di modificare il risultato.



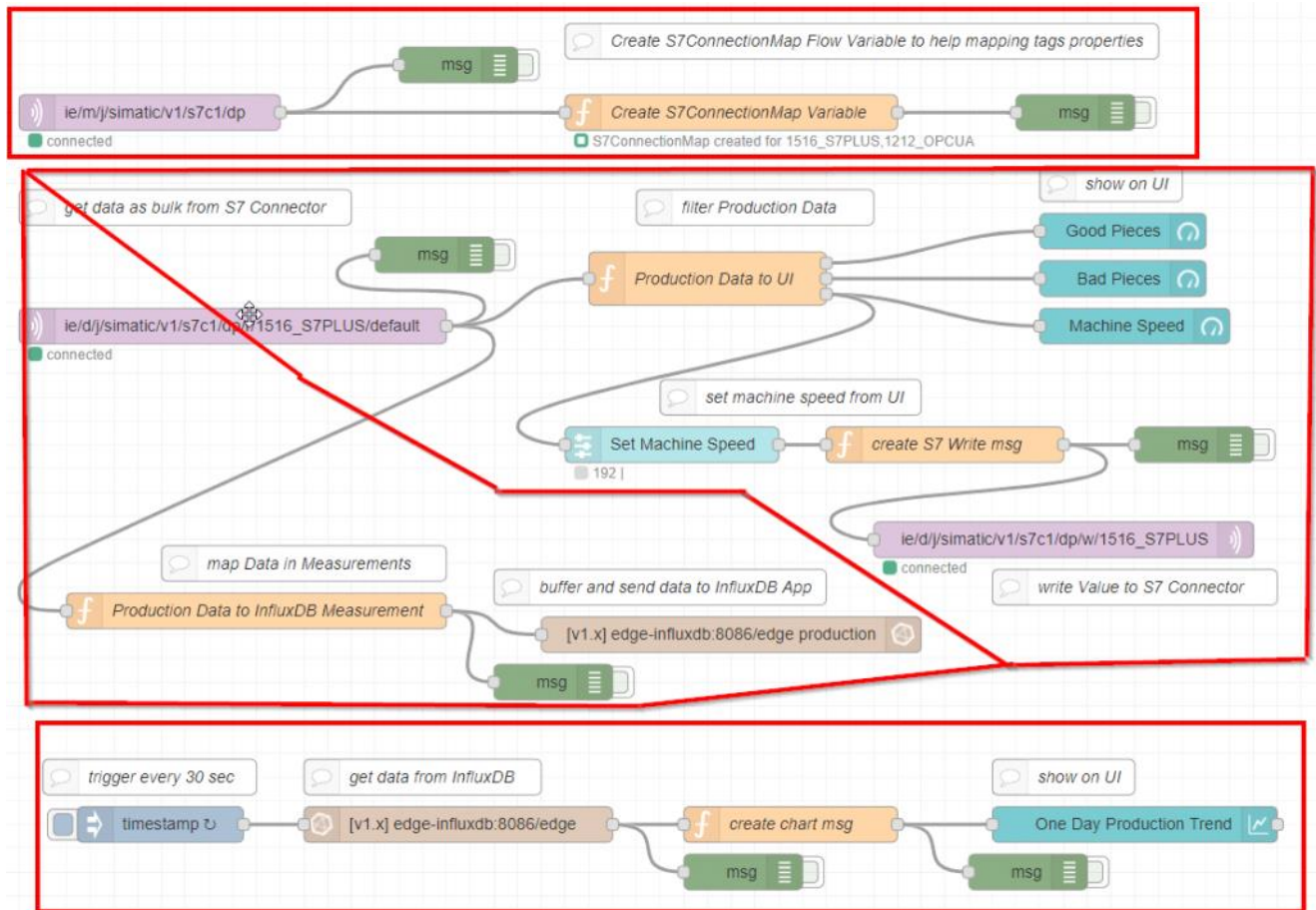
6. Tramite il **selettore temporale** in alto a destra è possibile impostare il range temporale di richiesta dei dati
7. Tramite il selettore centrale "**Visualization**" è possibile modificare il tipo di visualizzazione dei dati

5 - Esempio Applicativo

domenica 14 febbraio 2021 03:46

L'utilizzo di **database** consente di salvare dati a medio-lungo termine e di poter agire a posteriori sui dati raccolti per attività come la visualizzazione di dati storici o l'analisi di dati storici per ottenere informazioni importanti.

In questo esempio applicativo sarà possibile utilizzare il database **InfluxDB** per implementare una raccolta dati efficace e molteplici modalità di visualizzazione di queste informazioni.



Scopo del applicazione

Utilizzando le funzionalità offerte dall'applicazione fornita e dalle applicazioni elencate nel paragrafo seguente "Prerequisiti", sarà possibile implementare diverse funzionalità:

- Configurazione scambio dati con SIMATIC S7 Connector App e IE Databus App
- Scrittura dati provenienti da IE Databus con protocollo MQTT in database InfluxDB dedicato
- Dashboard per Visualizzazione risultati con Chronograf
- Lettura dati da database InfluxDB dedicato con filtro temporale e filtro condizionale
- Visualizzazione risultati con SIMATIC Flow Creator

Prerequisiti

- Per consentire la comunicazione con una sorgente dati S7, un **PLC** della famiglia **SIMATIC** (S7-300, S7-1200, S7-1500,...).

- L'applicazione **SIMATIC S7 Connector** deve essere **installata** e **configurata** sull'Edge Device utilizzato.
- L'applicazione **SIMATIC IE Databus** deve essere **installata** e **configurata** sull'Edge Device utilizzato.
- L'applicazione **SIMATIC Flow Creator** deve essere **installata** sull'Edge Device utilizzato.
- Il nodo **node-red-contrib-influxdb** deve essere installato nella libreria nodi di SIMATIC Flow Creator. Per maggiori dettagli seguire il capitolo Installazione nodo NodeRED InfluxDB.
- Il file **Flow_AppExample_S7toInfluxDB.json** deve essere importato all'interno dell'applicazione SIMATIC Flow Creator tramite la funzionalità di Import dal menù dedicato.
- Il file **Production_Chronograf.json** contiene la dashboard per Chronograf, importabile dall'applicazione
- L'applicazione **edge-influxdata-stack** deve essere **installata** sull'Edge Device utilizzato. Per maggiori dettagli seguire il capitolo [3 - Installazione App](#).
- L'applicazione **edge-influxdata-stack** viene fornita con un database pre-caricato nominato **"edge"**. Questo database deve necessariamente esistere per permettere al flusso NodeRED creato di funzionare correttamente.

Configurazione scambio dati con SIMATIC S7 Connector App e IE Databus App

Per poter utilizzare il database **InfluxDB** per il salvataggio di informazioni, occorre prima effettuare uno **scambio dati** con una sorgente dati in grado di generare ciclicamente nuovi valori.

In questo esempio applicativo viene considerato l'utilizzo di una sorgente dati **PLC SIMATIC S7**, configurata all'interno dell'applicazione **SIMATIC S7 Connector** inserendo le proprietà necessarie alla comunicazione e la lista delle variabili da monitorare.

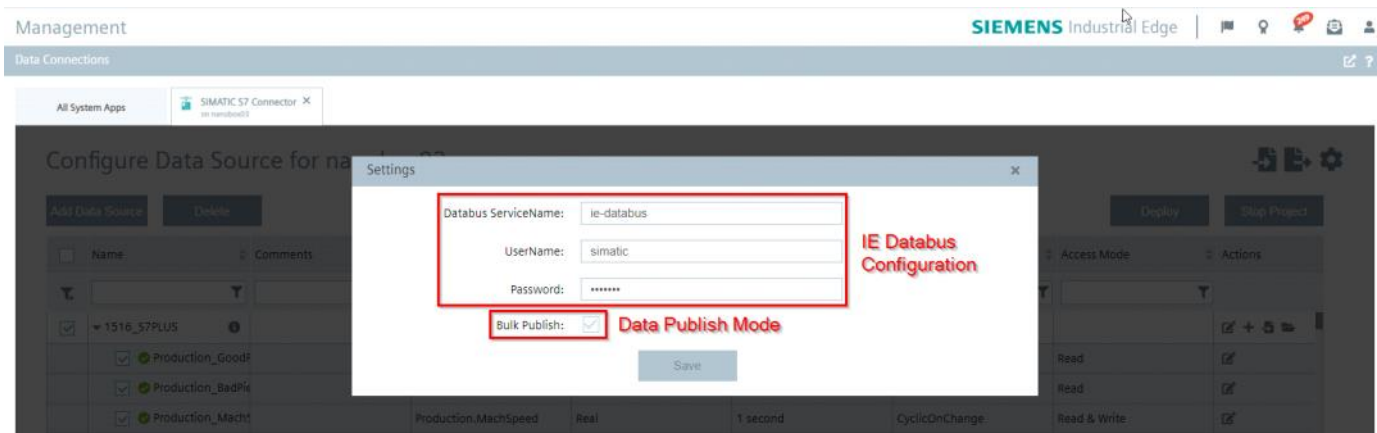
All'interno dell'applicazione S7 Connector, in questo caso, una CPU S7-1500 viene configurata come Datasource con protocollo **S7+** e con modalità **"Bulk Publish"** di pubblicazione dei dati:

La modalità Bulk Publish può essere impostata tramite il tasto **"Settings"**



presente nell'interfaccia di

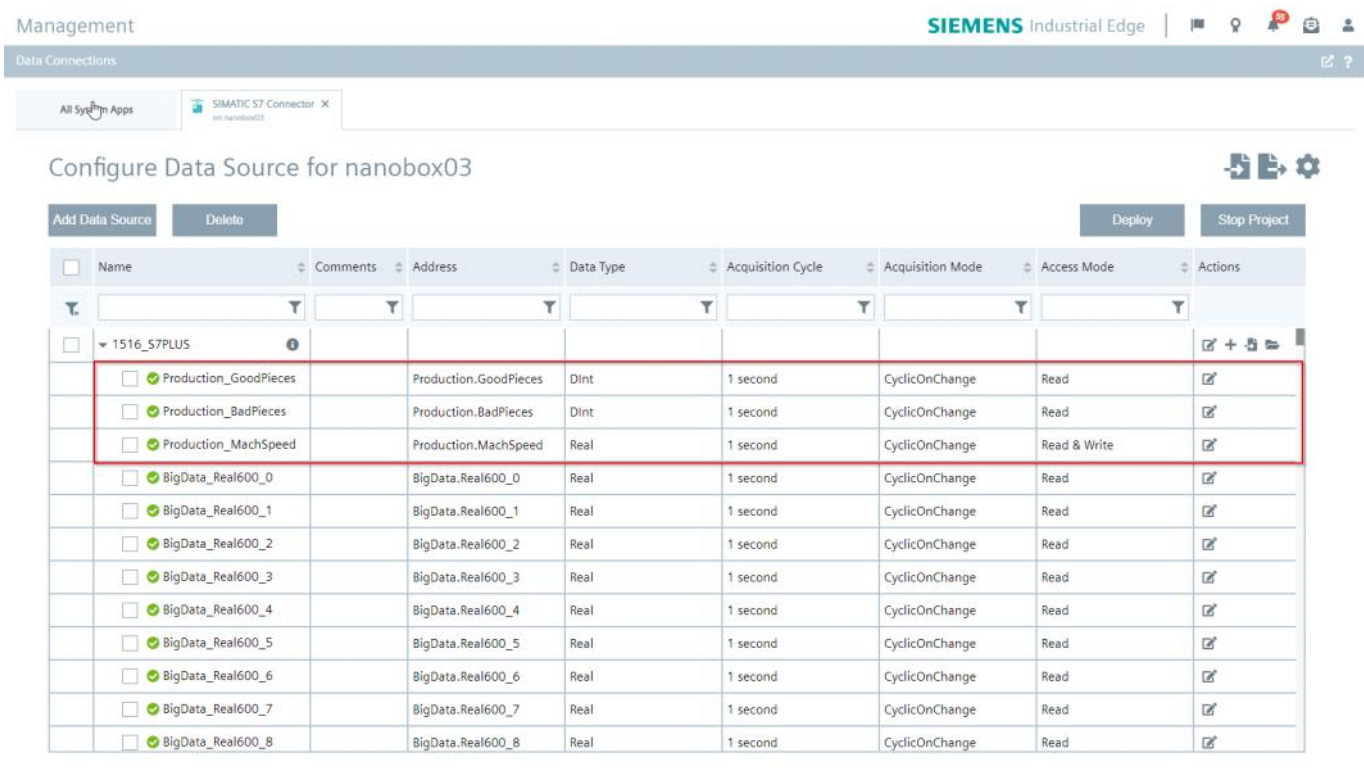
configurazione di S7 Connector, insieme all'utente e alla password utilizzati per connettersi al broker MQTT dell'applicazione **SIMATIC IE Databus** (in questo esempio useremo utente **"simatic"** e password **"simatic"**):



Per lo scopo verranno considerate 3 **variabili**, schematizzate nella tabella seguente:

Id Datapoint	Descrizione	S7+ Address	Type	Access Mode
<i>Production_GoodPies</i>	Contatore Pezzi Prodotti	Production.GoodPies	Dint	Read
<i>Production_BadPies</i>	Contatore Pezzi Scartati	Production.BadPies	Dint	Read
<i>Production_MachSpeed</i>	Velocità di produzione in pezzi/min.	Production.MachSpeed	Real	Read&Write

Di seguito è visibile il risultato della configurazione della sorgente dati con S7 Connector all'interno della sezione "**Data Connections**" del sistema **Industrial Edge Management**:



Nella modalità "**Bulk Publish**", quando S7 Connector effettua una lettura dei dati, viene utilizzato un solo topic MQTT dove sono pubblicate tutte le variabili che hanno subito un cambiamento di valore nel tempo di ciclo configurato (modalità "*CyclicOnChange*").

I dati letti dalle variabili configurate saranno quindi disponibili attraverso l'applicazione **SIMATIC IE Databus** utilizzando il topic MQTT dedicato alla datasource configurata (in questo esempio "*1516_S7PLUS*"), che emetterà ciclicamente un messaggio JSON contenente anche la proprietà "**vals**", un

array contenente tutte le proprietà delle variabili lette.

Di seguito un esempio di **messaggio JSON** ottenuto da S7 Connector tramite MQTT in fase di lettura delle variabili:

```
{
  "topic": "ie/d/j/simatic/v1/s7c1/dp/r/1516_S7PLUS/default",
  "payload": {
    "seq": 84631,
    "vals": [
      {
        "id": "101",
        "qc": 3,
        "ts": "2021-03-10T22:23:04.146Z",
        "val": 80
      },
      {
        "id": "102",
        "qc": 3,
        "ts": "2021-03-10T22:23:04.146Z",
        "val": 20
      },
      {
        "id": "103",
        "qc": 3,
        "ts": "2021-03-10T22:23:04.146Z",
        "val": 120.5
      }
    ]
  }
}
```

Di seguito vengono analizzate le tipiche **proprietà** del messaggio ottenuto da S7 Connector tramite MQTT in fase di lettura delle variabili, con riferimento al messaggio di esempio sopra indicato:

Proprietà	Descrizione	Valore Esempio
<i>topic</i>	Indica il topic mqtt di provenienza del messaggio ricevuto	ie/d/j/simatic/v1/s7c1/dp/r/1516_S7PLUS/default
<i>payload</i>	E' il corpo del messaggio. Contiene tutte le proprietà popolate da S7 Connector	{ "seq": 84631, "vals": [...] }
<i>seq</i>	Numero progressivo della sequenza di lettura. Ogni nuova lettura incrementa questo numero di 1.	84631
<i>vals</i>	Array che contiene tutte le variabili lette in un ciclo e le loro proprietà	[{ "id": "103", "qc": 3, "ts": "2021-03-10T22:23:04.146Z", "val": 120.5 },]
<i>id</i>	Id corrispondente al nome della variabile configurato all'interno dell'app S7 Connector	Production_MachSpeed
<i>qc</i>	Quality Code della lettura	3

<i>ts</i>	Timestamp in formato ISO86901 (yyyy-MM-ddThh:MM:ss)	"2021-03-10T22:23:04.146Z"
<i>val</i>	Valore della variabile letta	120.5

Per maggiori informazioni consultare i manuali dedicati:

- SIMATIC S7 Connector Operating Manual - <https://support.industry.siemens.com/cs/us/en/view/109783783>
- SIMATIC IE Databus Operating Manual - <https://support.industry.siemens.com/cs/us/en/view/109783784>
- Edge Management Operation Manual - <https://support.industry.siemens.com/cs/us/en/view/109793845>

Creazione ConnectionMap per Mapping Nome Tag - Id Tag

Come visto sopra, i messaggi ricevuti sul topic **data** contengono una proprietà **"id"**, un numero univoco assegnato da S7Connector Configurator ad ogni Tag configurata. La corrispondenza tra "id" e "nome" delle varie tag è visibile all'interno del messaggio MQTT chiamato **"metadata"**, che S7 Connector Configurator invia ad ogni client MQTT connesso al topic **ie/m/j/simatic/v1/s7c1/dp** e che viene aggiornato ogni qualvolta la configurazione delle Data Source in S7 Connector viene modificata.

Di seguito un esempio di messaggio MQTT "metadata" ricevuto:

```
{
  "topic": "ie/m/j/simatic/v1/s7c1/dp",
  "payload": {
    "seq": 1,
    "connections": [
      {
        "name": "1516_S7PLUS",
        "type": "S7+",
        "dataPoints": [
          {
            "name": "default",
            "topic": "ie/d/j/simatic/v1/s7c1/dp/r/1516_S7PLUS/default",
            "publishType": "bulk",
            "dataPointDefinitions": [
              {
                "name": "Production_GoodPieces",
                "id": "101",
                "dataType": "DInt"
              },
              {
                "name": "Production_BadPieces",
                "id": "102",
                "dataType": "DInt"
              },
              {
                "name": "Production_MachSpeed",
                "id": "103",
                "dataType": "Real"
              },
              {
                "name": "Production_RejectRatio",
                "id": "104",
                "dataType": "Real"
              }
            ]
          }
        ]
      }
    ]
  }
}
```

```

    }
  ]
}
],
},
"qos": 1,
"retain": true
}

```

Per poter risalire al "name" della Tag ricevuta a partire dalla proprietà "id" sarà necessario creare un collegamento tra queste due proprietà ottenute sul topic Metadata da S7 Connector. All'interno del flusso di SIMATIC Flow Creator dell'esempio applicativo fornito è presente una funzione che crea una variabile globale chiamata "**S7ConnectionMap**" che contiene diverse Mappe di corrispondenza di tipo **Map** (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Map?retiredLocale=it) che consentono di ottenere il nome di una tag a partire dall'id e viceversa. Di seguito un esempio di come utilizzare la variabile "**S7ConnectionMap**" in un nodo "function" di Flow Creator:

```

// get S7Map variable
let S7ConnectionMap = flow.get("S7ConnectionMap");

// find index of PLC connection
let connectionIndex = S7ConnectionMap.nameList.indexOf("1516_S7PLUS");

// use the PLC index to get the right PLC Name -> ID Map
let nameIDMap = S7ConnectionMap.nameIDMaps[connectionIndex];

// use the PLC index to get the right PLC ID -> Name Map
let IDnameMap = S7ConnectionMap.IDnameMaps[connectionIndex];

// USE MAPS
// get Tag ID from Tag Name
let tagId = nameIDMap.get("Production_MachSpeed")
//
// output >> 103

// get Tag Name from Tag ID
let tagName = IDnameMap.get(103)
//
// output >> "Production_MachSpeed"

```

Questa variabile globale viene creata a partire dal messaggio MQTT ricevuto dal nodo "**MQTT In**" (1) sul topic **ie/m/j/simatic/v1/s7c1/dp** con la funzione del nodo "**Function**" (2) seguente:

```

// Create an object that contains each S7 Connector connection property
// with different Map Objects to create correspondence between Tags IDs,
Names and Types.

// initialize the connections Mapping Object
let S7ConnectionMap = {
  "nameList":[],      // array of available S7 Connections names. Order
is the same in Map objects below.
  "typeList":[],      // array of available S7 Connections types. Order
is the same of nameList..
  "nameIDMaps":[],    // array of Tags Names-IDs object. Order is the
same of nameList.
  "IDnameMaps":[],    // array of Tags IDs-Names Map object. Order is
the same of nameList.
  "IDTypeMaps":[]     // array of Tags IDs-Type Map object. Order is the
same of nameList.
}

```

```

// Check Payload
let m = msg.payload;
if (m.seq == undefined) {
  // update global maps
  flow.set("S7ConnectionMap", null);
  // update function node status
  node.status({fill:"red",shape:"ring",text:"S7Map cannot be created"});

  return null;
}

// Iterate through connections
for (let i = 0; i < m.connections.length; i++)
{
  let connection = m.connections[i];
  // push connection name and type in global object
  S7ConnectionMap.nameList.push(connection.name);
  S7ConnectionMap.typeList.push(connection.type);
  // init maps
  let nameIDMap = new Map();
  let IDNameMap = new Map();
  let IDTypeMap = new Map();

  // Iterate through dataPoints
  let dataPoints = connection.dataPoints;
  for (let j = 0; j < dataPoints.length; j++)
  {
    let dataPoint = dataPoints[j];
    // Iterate through dataPointDefinitions
    let dataPointDefinitions = dataPoint.dataPointDefinitions;
    for (let k = 0; k < dataPointDefinitions.length; k++)
    {
      let dataPointDefinition = dataPointDefinitions[k];
      // push in maps the datapoint property
      nameIDMap.set(dataPointDefinition.name,
dataPointDefinition.id);
      IDNameMap.set(dataPointDefinition.id,
dataPointDefinition.name);
      IDTypeMap.set(dataPointDefinition.id,
dataPointDefinition.dataType);
    }
  }
  // push mappings in global object
  S7ConnectionMap.nameIDMaps.push(nameIDMap);
  S7ConnectionMap.IDnameMaps.push(IDNameMap);
  S7ConnectionMap.IDTypeMaps.push(IDTypeMap);
}

// update global maps
flow.set("S7ConnectionMap", S7ConnectionMap);

// set S7Map as output payload
msg.payload = S7ConnectionMap;

// update function node status
node.status({fill:"green",shape:"ring",text:"S7ConnectionMap created for "
+ S7ConnectionMap.nameList.join()});

return msg;

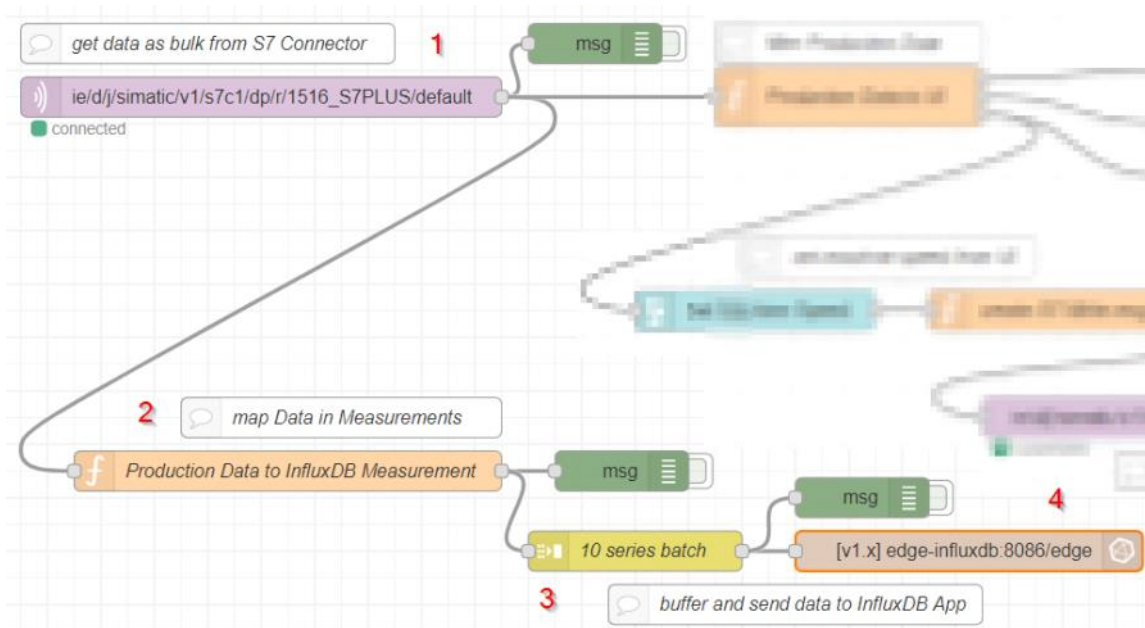
```

Nelle sezioni seguenti la variabile "**S7ConnectionMap**" verrà utilizzata per interpretare la lettura e la scrittura dei dati con la sorgente PLC.

Scrittura dati provenienti da IE Databus con protocollo MQTT in database InfluxDB

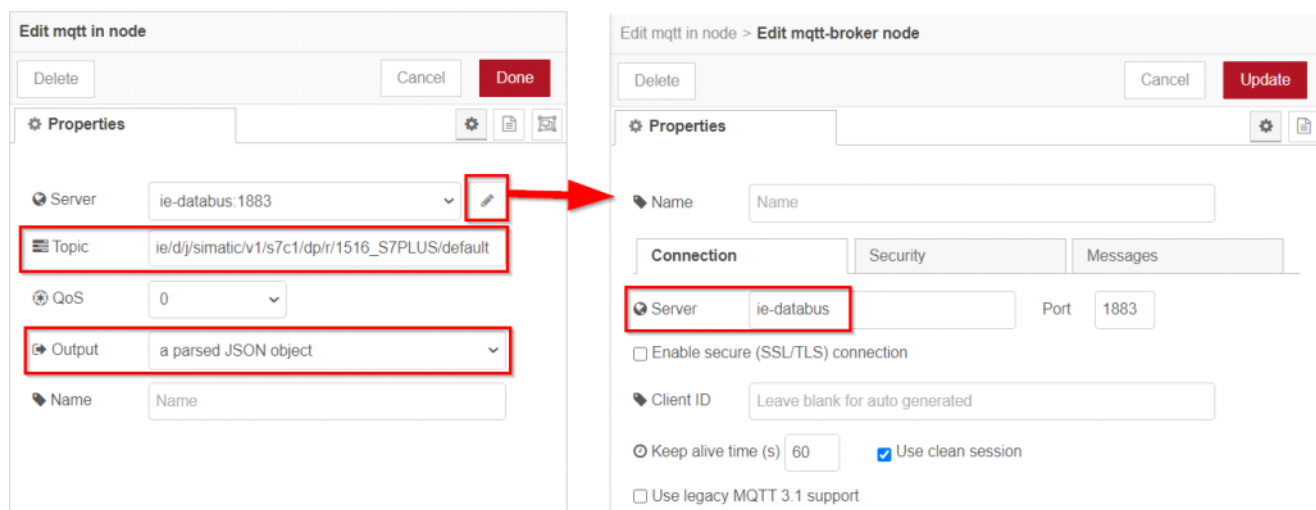
Una volta configurate le variabili da scambiare con la sorgente dati PLC tramite l'applicazione **S7 Connector**, è possibile utilizzare l'applicazione **SIMATIC Flow Creator** per raccogliere i dati letti, processarli ed inviarli al database InfluxDB pre-configurato dell'applicazione **edge-influxdata-stack**.

All'interno del flusso di SIMATIC Flow Creator dell'esempio applicativo fornito, la lettura dei dati provenienti da S7 Connector viene effettuata tramite nodo "**MQTT In**" (1), che successivamente invia il contenuto del messaggio ad un nodo "**Function**" (2) programmato per processare i dati e renderli compatibili con la standard richiesto dal nodo "**InfluxDB Out**" (4) che si occuperà di inviare la richiesta al database InfluxDB denominato "edge". Per un'ottimizzazione del carico di scrittura, un nodo "join"(3) crea un batch di 10 serie di dati prima dell'invio al database InfluxDB. Di seguito sono indicati i dettagli del flusso NodeRED coinvolti nella funzione di scrittura dei dati su database InfluxDB:



1. Ricezione messaggio bulk da MQTT

Attraverso il nodo "MQTT In", Flow Creator si connette al Broker MQTT IE Databus, utilizzando il topic dedicato al PLC configurato per lo scopo, denominato in questo esempio come "1516_S7PLUS":



Qui, ad ogni ciclo di lettura, riceveremo messaggi da MQTT IE Databus, che contiene tutti i dati letti dal PLC S7-1500 configurato all'interno dell'applicazione S7 Connector. Il nome del PLC configurato sarà parte del topic da sottoscrivere.

Di seguito un esempio di output per il nodo "MQTT In":

```
{
  "topic": "ie/d/j/simatic/v1/s7c1/dp/r/1516_S7PLUS/default",
  "payload": {
    "seq": 84631,
    "vals": [
      {
        "id": "101",
        "qc": 3,
        "ts": "2021-03-10T22:23:04.146Z",
        "val": 80
      },
      {
        "id": "102",
        "qc": 3,
        "ts": "2021-03-10T22:23:04.146Z",
        "val": 20
      },
      {
        "id": "103",
        "qc": 3,
        "ts": "2021-03-10T22:23:04.146Z",
        "val": 120.5
      }
    ]
  }
}
```

Per la connessione al broker MQTT **SIMATIC IE Databus** è necessaria la configurazione di un utente abilitato allo scambio dati. In questo caso è stato configurato utilizzando i seguenti parametri:

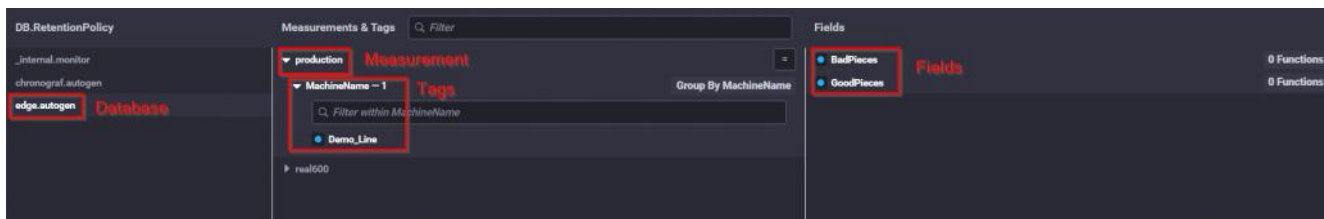
IE Databus Address	IE Databus Port	IE Databus User	IE Databus Password
ie-databus	1883	simatic	simatic

2. Pre-processamento Dati per InfluxDB

Prima di poter salvare i dati ricevuti nel database Influx DB sarà necessario **formattare** il messaggio ricevuto dal nodo "MQTT In" attraverso un nodo "function", secondo le specifiche richieste dal nodo "influx-out" dedicato.

Il nodo "**influx-out**" consente di inviare in diverse modalità nuovi dati all'interno del database InfluxDB, come specificato nella documentazione dedicata del nodo node-red-contrib-influxdb (<https://flows.nodered.org/node/node-red-contrib-influxdb>).

In questo caso creeremo un "**Measurement**" InfluxDB chiamato "**production**" in cui inseriremo diversi "**Field**" InfluxDB corrispondenti ad ognuna delle variabili lette da S7 Connector ed una "**Tag**" InfluxDB, un metadato che indica la macchina di provenienza di ogni serie di "Field" InfluxDB:



Per maggiori dettagli consulta il capitolo 4 - Utilizzo App.

Considerando le variabili utilizzate in questo esempio applicativo, per poter inserire i dati letti nel database InfluxDB con il modello presentato sopra, il nodo "influx-out" prevede che la proprietà `msg.payload` in ingresso al nodo abbia la struttura seguente:

```
[
  [{
    Production_BadPieces: 2,
    Production_GoodPieces: 10,
    time: 1616881844000000
  },
  {
    MachineName:"test_machine"
  }],
  [{
    Production_GoodPieces: 14,
    time: 1616881845000000
  },
  {
    machine:"test_machine"
  }]
]
```

Invieremo un batch di 10 serie di dati sotto forma di array, in cui ogni elemento è un array composto da due elementi: il primo è un oggetto contenente le proprietà "Field" nella forma "FieldKey:FieldValue" e il timestamp di ogni serie, il secondo è un oggetto contenente le proprietà "Tag" nella forma "TagKey:TagValue".

Il nodo "**function**" consente di processare un messaggio in Input tramite uno script in linguaggio Javascript e creare uno o più messaggi di uscita.

Con la funzione seguente trasformeremo la proprietà `msg.payload` ricevuta da IE Databus con il nodo "MQTT In", che contiene tutti i dati letti dal PLC S7-1500 configurato nell'applicazione S7 Connector, in un messaggio corrispondente allo standard richiesto dal nodo "influx-out" come sopra indicato:

```
// create out message
let outMsg = {"payload": null};

// Insert here the ordered output tag list
let selectedTags = ["Production_GoodPieces",
  "Production_BadPieces",
  "Production_MachSpeed"];

// get S7Map variable
let S7ConnectionMap = flow.get("S7ConnectionMap");

// find index of 1516_S7PLUS connection
let connectionIndex = S7ConnectionMap.nameList.indexOf("1516_S7PLUS");
// use the index to get the right map
let nameIDMap = S7ConnectionMap.nameIDMaps[connectionIndex];

// get IDs of selected Tags
let selectedIDs = selectedTags.map(name => nameIDMap.get(name));
```

```

// create an empty series with null timestamp
let outSeries = [{"time": null}, {"machine": "test_machine"}]

// iterate through readed tags
for (let i=0; i < msg.payload.vals.length; i++){

    // iterate through selected ids
    for(let j=0; j < selectedIDs.length; j++){

        // search for tagselectedTags[j]
        if(msg.payload.vals[i].id == selectedIDs[j])
        {
            outSeries[0].time = new
Date(Date.parse(msg.payload.vals[i].ts)).getTime();
            outSeries[0][selectedTags[j]] =
Number(msg.payload.vals[i].val.toFixed(2));

            // stop on first match, ids are uniques
            break;
        }
    }
}

// if time was changed means that some ids was finded
if (outSeries[0].time)
{
    outMsg.payload = outSeries;
    return outMsg;
}

```

Questa funzione invierà in uscita dal nodo "function" un array contenente uno o più "field" in base ai dati ricevuti in ingresso. La proprietà "id" di ogni variabile letta viene confrontata con l'id delle variabili selezionate recuperate tramite la variabile globale **"S7ConnectionMap"** e in caso di match la funzione prosegue. Il timestamp (epoch) in ms viene ricavato a partire dalla proprietà "ts" di ogni variabile letta. Il valore della variabile identificato dalla proprietà "val" viene convertito a numero ed eventualmente approssimato a 3 cifre decimali. Viene aggiunta la "Tag" InfluxDB denominata "machine" come metadato per ogni serie di dati.

Il nodo "join" successivo si occuperà di creare l'array di serie che identifica il batch di dati da inserire successivamente nel database InfluxDB con la misura "production".

3. Buffer serie di dati in Batch

Il database InfluxDB è ottimizzato per le serie temporali e può gestire carichi di query più elevate, ma è buona abitudine cercare di ottimizzare il carico di scrittura raccogliendo qualche dato prima di scriverlo nel database.

Questa funzione di buffer dei dati può essere semplicemente creata in Flow Creator utilizzando un nodo "join", che si occupa di unire una serie di messaggi in un unico messaggio secondo diverse regole e configurazioni.

In questo esempio applicativo, il nodo "join" utilizzato crea un batch di 5 serie di dati sotto forma di array. Una volta che il contatore dei messaggi raggiunge il limite identificato dalla proprietà "count" configurata, il nodo "join" procederà all'invio del batch dati al database InfluxDB.

Edit join node

Delete Cancel Done

Properties

Mode: manual

Combine each: msg. payload

to create: an Array

Send the message:

- After a number of message parts: 5
- After a timeout following the first message: seconds
- After a message with the msg.complete property set

Name: 5 series batch

4. Invio Dati a InfluxDB

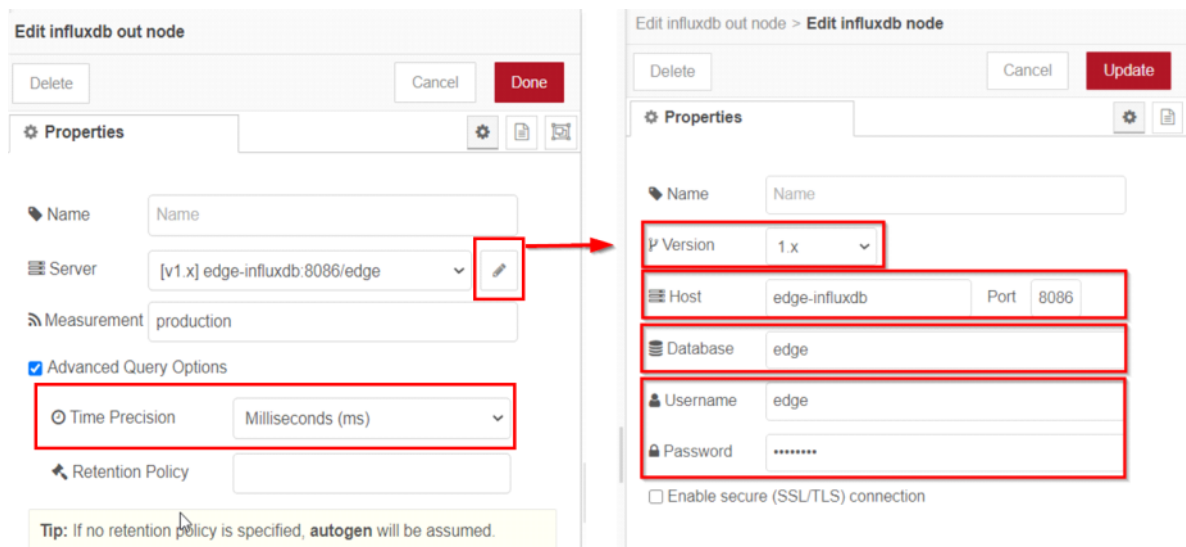
Per la comunicazione con il database InfluxDB è possibile sfruttare le InfluxDB API esposte da InfluxDB per lo scambio dati con il database. Queste InfluxDB API sono alla base del nodo di Node-RED node-red-contrib-influxdb, creato per scrivere e interrogare i dati da un database di serie temporali InfluxDB.

Questi nodi supportano sia i database InfluxDB 1.x che InfluxDB 2.0 in modo configurabile. Per verificare le opzioni fornite dalle diverse versioni consultare documentazione dedicata del nodo **node-red-contrib-influxdb** (<https://flows.nodered.org/node/node-red-contrib-influxdb>).

L'applicazione edge-influxdata-stack fornita comprende il servizio container "**edge-influxdb**" basato sulla versione 1.8 di InfluxDB ed espone un database pre-definito denominato "**edge**" con 'utente e la password pre-configurate "**edge**" / "**edge**". Questa applicazione può essere contattata da altre applicazioni presenti sia sull'Edge Device utilizzato sia da reti esterne connesse alle porte fisiche dell'Edge Device grazie alla funzionalità di Port forwarding offerta dal servizio Docker.

Questi parametri possono essere inseriti direttamente nella configurazione del parametro "**Server**" dei nodi della libreria node-red-contrib-influxdb:

Service Name	Host name	Internal Port	External Host	External Port	Database Name	User	Password
edge-influxdb	edge-influxdb	8086	[ied-ip-address] oppure [ied-dns-name]	38086	edge	edge	edge



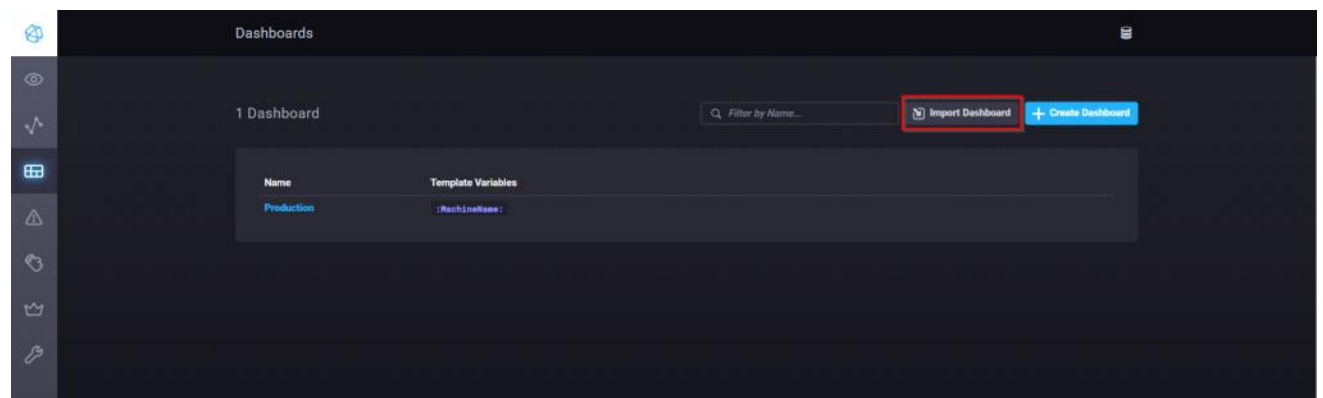
Tramite la checkbox "Advanced Query Options" è possibile pre-definire anche la precisione del tempo inserito nelle serie di dati. Come definito nel paragrafo "Pre-processingo Dati per InfluxDB", in questo esempio applicativo viene calcolato il tempo (epoch) in ms e l'opzione "Time Precision" viene quindi impostata su "milliseconds (ms)".

Dashboard per Visualizzazione risultati con Chronograf

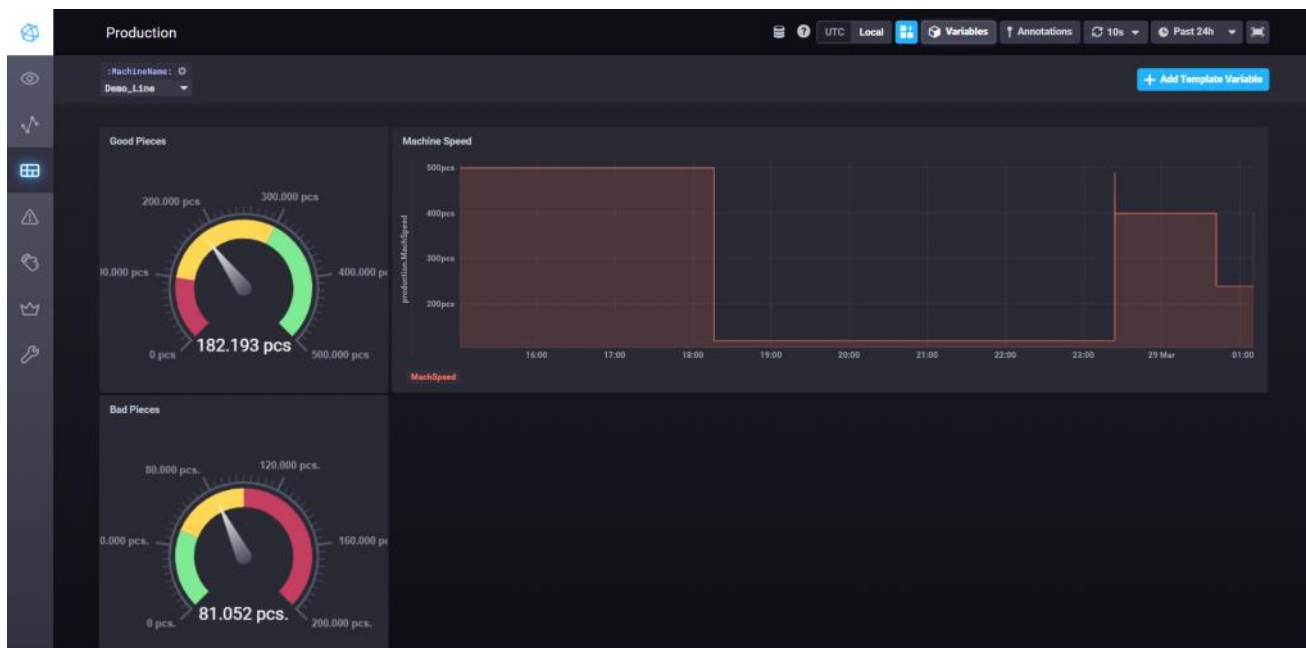
L'applicazione edge-influxdata-stack viene fornita con il servizio "edge-chronograf", l'interfaccia utente amministrativa e il motore di visualizzazione dello stack. Grazie all'applicazione Chronograf, tra le altre cose, è possibile sia esplorare i dati presenti nei vari database sia realizzare Web Dashboard con oggetti grafici in modo rapido.

In questo esempio applicativo viene fornita la dashboard "**Production**" che può essere direttamente importata come file in formato JSON. Per importare la dashboard:

1. Connettersi all'interfaccia Chronograf utilizzando l'URL [https://\[device-ip-address\]/edge-chronograf/](https://[device-ip-address]/edge-chronograf/)
2. Nella sezione "**Dashboards**" premere il tasto "**Import Dashboard**"




3. E scegliere il file **Production_Chronograf.json** fornito
4. Al completamento la dashboard "Production" sarà visibile



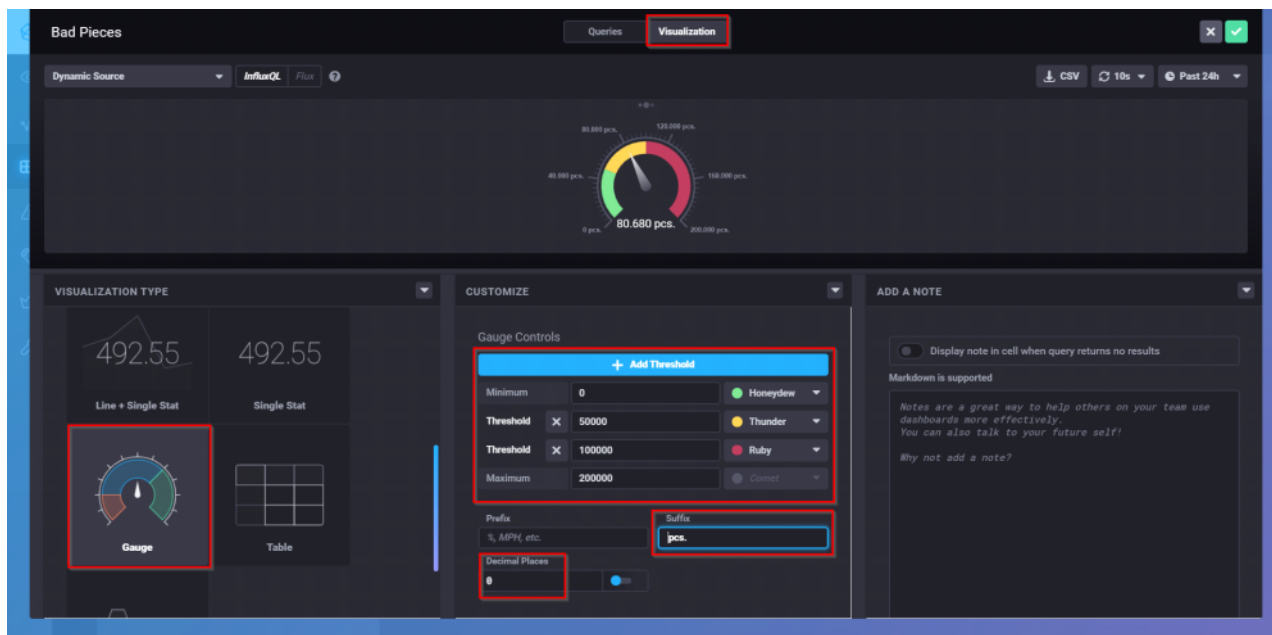
Creare un nuovo oggetto grafico su Dashboard Chronograf

Le dashboard sono personalizzabili con nuovi oggetti grafici e query, ed è possibile crearne molteplici.

Per poter creare un nuovo oggetto grafico all'interno della dashboard :

1. premere il tasto  "Add a cell to Dashboard"
2. Selezionare prima il database di riferimento (ad es. "edge"), poi la measurement desiderata (ad es. "production") e successivamente tutti i "fields" e le "tags" desiderate. Appairà automaticamente una query standard per la lettura dei dati selezionati, che è possibile personalizzare in base allo scopo.

3. Nel menù "Visualization" è possibile scegliere il tipo di oggetto da visualizzare e alcune impostazioni grafiche come soglie, colori e altro:



4. Premere il tasto di conferma in alto a destra per inserire il nuovo oggetto all'interno della Dashboard.

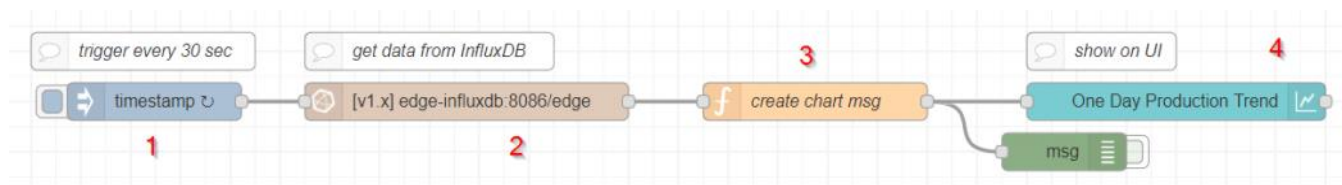
Lettura dati da database InfluxDB dedicato con filtro temporale e filtro condizionale

Oltre all'inserimento di nuovi dati all'interno di un "measurement" del database InfluxDB, potrebbe essere necessario interrogare il database per processare e richiedere dati storici, ad esempio per visualizzazione, analisi o reportistica.

La libreria di nodi node-red-contrib-influxdb offre la possibilità di effettuare query personalizzate per interrogare una o più misure in un database InfluxDB tramite il nodo "influxdb-in". La query è specificata nella configurazione del nodo o tramite la proprietà msg.query in ingresso. Il risultato della query viene restituito nel messaggio in output dal nodo con tramite la proprietà msg.payload.

All'interno del flusso di SIMATIC Flow Creator dell'esempio applicativo fornito, la lettura dei dati dal database InfluxDB viene azionata da un messaggio di trigger dal nodo "inject" (1) ogni 30 secondi, dopo il quale il nodo "influxdb-in" (2) invia la query configurata al database InfluxDB. Una volta ottenuta la risposta, questa viene processata e formattata da un nodo "function" (3) per la visualizzazione di un grafico su Dashboard tramite il nodo "ui_chart" (4).

Di seguito sono indicati i dettagli del flusso NodeRED coinvolti:



In questo esempio applicativo utilizzeremo il nodo "influxdb-in" connesso al servizio database "edge-influxdb" e una query dedicata per leggere i dati salvati "Production_GoodPieces" e "Production_BadPieces" nelle ultime 24 ore e aggregarli in modo da ridurre il numero di campioni da visualizzare.

Edit influxdb in node

Delete Cancel Done

Properties

Name

Server [v1.x] edge-influxdb:8086/edge

☐ Raw Output ☐ Advanced Query Options

</> Query

```

1 SELECT ROUND(MEAN("Production_GoodPieces")) AS "GoodPiecesSub",
2     ROUND(MEAN("Production_BadPieces")) AS "BadPiecesSub"
3 FROM "edge"."autogen"."production"
4 WHERE time > now() - 1d AND time < now()
5 GROUP BY time(1m),* FILL(null)

```

La sintassi utilizzata per la query è InfluxQL. Per maggiori informazioni sul linguaggio di query InfluxQL visitare la documentazione ufficiale al link https://docs.influxdata.com/influxdb/v1.8/query_language/.

```

SELECT ROUND(MEAN("Production_GoodPieces")) AS "GoodPiecesSub",
        ROUND(MEAN("Production_BadPieces")) AS "BadPiecesSub"
FROM "edge"."autogen"."production"
WHERE time > now() - 1d AND time < now()
GROUP BY time(1m),* FILL(null)

```

La query utilizzata andrà a selezionare dai dati delle ultime 24 ore la media dei valori "Production_GoodPieces" e "Production_BadPieces" per ogni minuto. Questo consente di sottocampionare i dati grezzi, raccolti ogni secondo, con un fattore 1/60. La risposta del database viene inviata quindi in uscita al nodo "influxdb-in" con la proprietà msg.payload. Questa sarà un array contenente le diverse serie temporali di "Field" e "Tags" ed i loro valori. Di seguito un estratto di esempio del messaggio ricevuto in risposta alla query:

```

{
  "payload":[
    {
      "time":"2021-03-27T20:47:00.000Z",
      "GoodPiecesSub":131096,
      "BadPiecesSub":57514,
      "machine":"test_machine"
    },
    {
      "time":"2021-03-27T20:48:00.000Z",
      "GoodPiecesSub":131327,
      "BadPiecesSub":57597,
      "machine":"test_machine"
    },
    ...
    ...
    {
      "time":"2021-03-28T20:47:00.000Z",
      "GoodPiecesSub":131802,
      "BadPiecesSub":59069,
      "machine":"test_machine"
    },
    {

```

```

    "time": "2021-03-28T20:48:00.000Z",
    "GoodPiecesSub": 131886,
    "BadPiecesSub": 59106,
    "machine": "test_machine"
  },
  ],
  "topic": ""
}

```

A questo punto sarà possibile processare il contenuto della proprietà `msg.payload` formattandolo per la visualizzazione su un grafico a linee.

Dashboard per Visualizzazione risultati con SIMATIC Flow Creator

Per poter visualizzare i dati ottenuti dalla query precedentemente descritta è possibile utilizzare la funzionalità di Web Dashboard di SIMATIC Flow Creator con i nodi della libreria "node-red-dashboard", con una serie di nodi dedicati a diversi oggetti grafici come manometri, campi di testo, grafici e tanto altro.

Per maggiori informazioni sulla libreria "node-red-dashboard" visitare la documentazione ufficiale <https://flows.nodered.org/node/node-red-dashboard>.



Il nodo "charts-ui" consente di visualizzare grafici di diverso tipo (linee, barre, torta) sulla Web Dashboard di SIMATIC Flow Creator, sia inviando in real-time nuovi dati sia inviando l'intera struttura dati.

In questo esempio applicativo, a partire dai dati ricevuti dal nodo "influxdb-out" come risultato della query, verrà creata la struttura dati corretta per la visualizzazione di un grafico a linee contenente i dati delle ultime 24 ore. Utilizzando il nodo "charts-ui" è possibile definire i vari parametri di configurazione come la Tab ("S7 - InfluxDB App Example") e il Gruppo ("Control and Monitor") Dashboard di appartenenza:

Prima di procedere alla visualizzazione sarà però necessario formattare correttamente i dati secondo lo standard del nodo "chart-ui", utilizzando in questo caso un nodo "function" con una funzione dedicata al processamento dell'array di serie temporali ricevuto.

Per maggiori informazioni su come formattare correttamente una o più serie di dati per la visualizzazione tramite nodo "chart-ui" visitare la documentazione ufficiale del nodo node-red-dashboard dedicata specificatamente al nodo <https://github.com/node-red/node-red-dashboard/blob/master/Charts.md>

Di seguito la funzione utilizzata per la formattazione delle serie temporali ricevute:

```
// define influxdb fields names
let outFields = ["GoodPiecesSub", "BadPiecesSub"]

// create base out message
let outMsg = {payload:[{}]};

// create chart properties
outMsg.payload[0].series = [outFields[0], outFields[1]];
outMsg.payload[0].data = [[],[]];
outMsg.payload[0].labels = [""];

// scan input data series and create chart data series
for (let i = 0; i < msg.payload.length; i++)
{
    let ts = new Date(msg.payload[i].time).getTime();
    // if value of selected fields exist, push to data array together with
    timestamp
    if (msg.payload[i][outFields[0]] !== null)
    {
        outMsg.payload[0].data[0].push({"x": ts, "y": msg.payload[i]
[outFields[0]]});
    }
    if (msg.payload[i][outFields[1]] !== null)
    {
        outMsg.payload[0].data[1].push({"x": ts, "y": msg.payload[i]
[outFields[1]]});
    }
}
```

```

[outFields[1]]});
    }
}

return outMsg;

```

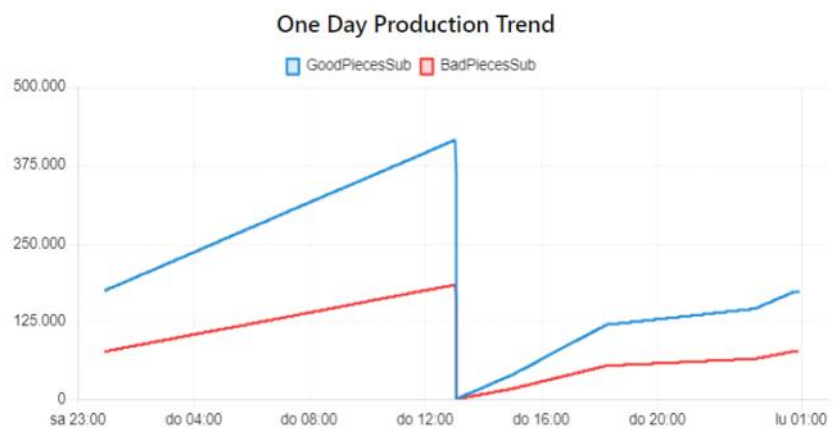
Questa funzione crea la struttura dati per un grafico a linee con due serie temporali denominate "GoodPiecesSub" e "BadPiecesSub", contenente diversi campioni ed il relativo timestamp. Di seguito un estratto di esempio del messaggio in uscita dal nodo "function":

```

{
  "payload": [
    {
      "series": ["GoodPiecesSub", "BadPiecesSub"],
      "data": [
        [
          {
            "x": 1616884320000,
            "y": 166138
          },
          ...
          ...
          {
            "x": 1616944260000,
            "y": 92754
          }
        ],
        [
          {
            "x": 1616884320000,
            "y": 72844
          },
          ...
          ...
          {
            "x": 1616944260000,
            "y": 41251
          }
        ]
      ],
      "labels": [""]
    }
  ]
}

```

I dati formattati sono visualizzabili direttamente come due serie su grafico a linee collegandosi alla web dashboard di SIMATIC Flow Creator utilizzando il link [https://\[device-ip-address\]/ui/](https://[device-ip-address]/ui/):



6 - Costruzione App

lunedì 15 febbraio 2021 00:25

L'applicazione è composta da diversi microservizi, tutti basati sul "TICK stack" fornito da InfluxData. Per maggior informazioni su TICK Stack consulta la pagina ufficiale InfluxData

<https://www.influxdata.com/time-series-platform/>.

I microservizi utilizzati in questa versione sono:

- **InfluxDB**
- **Kapacitor**
- **Chronograf**

L'applicazione edge generata verrà dotata anche di un "*service name*" che consentirà di raggiungere la web dashboard di Chronograf. Per maggiori informazioni seguire il paragrafo **Configurazione Reverse Proxy per Chronograf** in questo capitolo.

Di seguito sono elencati i file principali per la costruzione dell'app:

[docker-compose.yml](#)

Le immagini base che compongono questa app sono state costruite usando lo strumento **docker-compose** (<https://docs.docker.com/compose/>) con il comando **docker-compose up -d --build** sul seguente file **docker-compose.yml**:

```
version: "2.4"
services:
  edge-influxdb:
    build: ./influxdb
    image: edge-influxdb:0.0.1
    container_name: edge-influxdb
    restart: always
    ports:
      - 38086:8086
      - 38088:8088
    environment:
      - INFLUXDB_DB=edge
      - INFLUXDB_USER=edge
      - INFLUXDB_USER_PASSWORD=edge
      - INFLUXDB_HTTP_FLUX_ENABLED=false
    volumes:
      - edge-influxdb-volume:/var/lib/influxdb
    mem_limit: 1200mb
    networks:
      - proxy-redirect
  edge-kapacitor:
    image: kapacitor:1.5.9-alpine
    container_name: edge-kapacitor
    restart: always
    environment:
      - KAPACITOR_INFLUXDB_0_URLS_0=http://edge-influxdb:8086
      - KAPACITOR_MQTT_0_ENABLED=true
      - KAPACITOR_MQTT_0_NAME=ie-databus-kapacitor
      - KAPACITOR_MQTT_0_URL=tcp://ie-databus:1883
      - KAPACITOR_MQTT_0_USERNAME=edge
      - KAPACITOR_MQTT_0_PASSWORD=edge
    links:
```

```

    - "edge-influxdb"
  depends_on:
    - "edge-influxdb"
  volumes:
    - edge-kapacitor-volume:/var/lib/kapacitor
  mem_limit: 400mb
  networks:
    - proxy-redirect
edge-chronograf:
  image: chronograf:1.9-alpine
  container_name: edge-chronograf
  restart: always
  ports:
    - 38888:8888
  environment:
    - BASE_PATH=/edge-chronograf
    - KAPACITOR_URL=http://edge-kapacitor:9092
    - INFLUXDB_URL=http://edge-influxdb:8086
    - INFLUXDB_USER=edge
    - INFLUXDB_PASS=edge
  links:
    - "edge-influxdb"
    - "edge-kapacitor"
  depends_on:
    - "edge-influxdb"
    - "edge-kapacitor"
  volumes:
    - edge-chronograf-volume:/var/lib/chronograf
  mem_limit: 200mb
  networks:
    - proxy-redirect
volumes:
  edge-influxdb-volume:
    name: edge-influxdb-volume
  edge-kapacitor-volume:
    name: edge-kapacitor-volume
  edge-chronograf-volume:
    name: edge-chronograf-volume
networks:
  proxy-redirect:
    external: true
    name: proxy-redirect

```

Il file docker-compose sopra:

- crea e porta in esecuzione i servizi **edge-influxdb**, **edge-chronograf** e **edge-kapacitor**:
- **edge-influxdb**:
 - costruisce l'immagine Docker personalizzata **edge-influxdb:0.0.1** usando il file **influxdb/Dockerfile**.
 - Mappa le porte 8086 e 8088 necessarie all'utilizzo del database rispettivamente sulle porte 38086 e 38088.
 - Crea il database predefinito "edge" e l'utente "edge" con password "edge" per il login al database InfluxDB. In questo caso viene mantenuto disabilitato il linguaggio query FLUX tramite la variabile d'ambiente `INFLUXDB_HTTP_FLUX_ENABLED`.
 - Mappa la cartella `/var/lib/influxdb` all'interno del container `edge-influxdb` al volume `edge-influxdb-volume` nel sistema host. Qui saranno salvati tutti i dati relativi ai database InfluxDB.
- **edge-chronograf**
 - Usa l'immagine base `chronograf:1.8-alpine`
 - Mappa la porta 8888 che consente l'accesso alla web dashboard di Chronograf sulla

- porta 38888.
- Imposta le connessioni di default verso i microservizi edge-influxdb e edge-kapacitor tramite le variabili d'ambiente dedicate (KAPACITOR_URL, INFLUXDB_URL, INFLUXDB_USER, INFLUXDB_PASS) e predisporre un prefisso per l'url della web dashboard tramite la variabile BASE_PATH. Questa verrà utilizzata per impostare la funzionalità di Reverse Proxy nella configurazione con Edge App Publisher.
- Mappa la cartella /var/lib/chronograf all'interno del container edge-chronograf al volume edge-chronograf -volume nel sistema host. Qui saranno salvati tutte le customizzazioni fatte a chronograf .
- Attende che i microservizi edge-influxdb e edge-kapacitor siano in esecuzione tramite il campo depends_on
- **edge-kapacitor**
 - Usa l'immagine base kapacitor:1.5-alpine
 - Imposta la connessione di default verso il microservizio edge-influxdb tramite la variabile d'ambiente dedicata KAPACITOR_INFLUXDB_0_URLS_0
 - Imposta la connessione al Broker MQTT **ie-databus** offerto dai dispositivi Industrial Edge tramite le variabili d'ambiente dedicate KAPACITOR_MQTT_0_NAME, KAPACITOR_MQTT_0_URL, KAPACITOR_MQTT_0_USER, KAPACITOR_MQTT_0_PASSWORD, come indicato qui (<https://docs.influxdata.com/kapacitor/v1.6/administration/configuration/#kapacitor-configuration-file-location>)
 - Mappa la cartella /var/lib/kapacitor all'interno del container edge-kapacitor al volume edge-kapacitor -volume nel sistema host. Qui saranno salvati tutte le customizzazioni fatte a kapacitor .
 - Attende che il microservizio edge-influxdb sia in esecuzione tramite il campo depends_on

Influxdb/Dockerfile

Al fine di personalizzare l'immagine Docker di base per il microservizio edge-influxdb di questa App, è stato utilizzato il seguente **Dockerfile**:

```
#from base image
FROM influxdb:1.8-alpine

COPY ./influxdb.conf /etc/influxdb/influxdb.conf
```

- Dal file docker-compose precedente, Docker avvierà il processo di costruzione del microservizio edge-influxdb dall'immagine base influxdb:1.8-alpine con il sistema Linux Alpine e InfluxDB preinstallati.
- Viene copiato il file influxdb.conf che contiene la configurazione modificata di InfluxDB nel percorso di default /etc/influxdb/influxdb.conf. I parametri modificati rispetto alla configurazione di default sono:

Section	Property	Value	Default Value
<i>data</i>	<i>index-version</i>	"tsi1"	"inmem"
<i>monitor</i>	<i>store-interval</i>	"12s"	"10s"
<i>subscriber</i>	<i>write-concurrency</i>	50	40

Per i dettagli sulle motivazioni dell'utilizzo del modo "tsi1" visitare il seguente link <https://www.influxdata.com/blog/how-to-overcome-memory-usage-challenges-with-the-time-series-index/>

Importare l'app in Edge App Publisher

Per importare l'app nel software **SIMATIC Edge App Publisher**, è possibile utilizzare il file `edge-influxdata-stack_x.x.x.app` caricandolo con il tasto **"Import"** all'interno del proprio elenco di applicazioni o in alternativa è possibile costruire le immagini in un ambiente Docker utilizzando il file `docker-compose.yml` descritto sopra con il comando `docker-compose up -d --build`.

Importando il file `docker-compose.yml` descritto sopra nell'Edge App Publisher verranno applicate alcune modifiche per rendere l'app compatibile con l'ambiente SIMATIC Edge:

- Il parametro **build** viene eliminato poiché l'immagine è già stata costruita.

Configurazione Reverse Proxy per Chronograf

L'applicazione `edge-influxdata-stack` utilizza la funzionalità reverse proxy offerta dalla piattaforma SIMATIC Edge che consente di mappare un nome servizio ad un endpoint URL di un applicazione, sostituendo il percorso `http(s)://[device-ip-address]:[my-app-port]/my-app-path/` con `http(s)://[device-ip-address]/my-reverse-proxy-app-path/`.

In questa applicazione esponiamo l'interfaccia grafica del servizio "edge-chronograf" per l'amministrazione e l'esplorazione del database InfluxDB. Questo servizio, come specificato nei file specificati sopra, sarebbe raggiungibile utilizzando la porta 38888 sulla rete esterna dell'Edge Device, che viene a sua volta "connessa" alla porta 8888 del servizio "edge-chronograf" grazie al runtime Docker.

L'utilizzo del reverse proxy con nome servizio "edge-chronograf" consentirà la seguente mappatura:

[http://\[device-ip-address\]:38888](http://[device-ip-address]:38888) → [https://\[device-ip-address\]/edge-chronograf](https://[device-ip-address]/edge-chronograf)

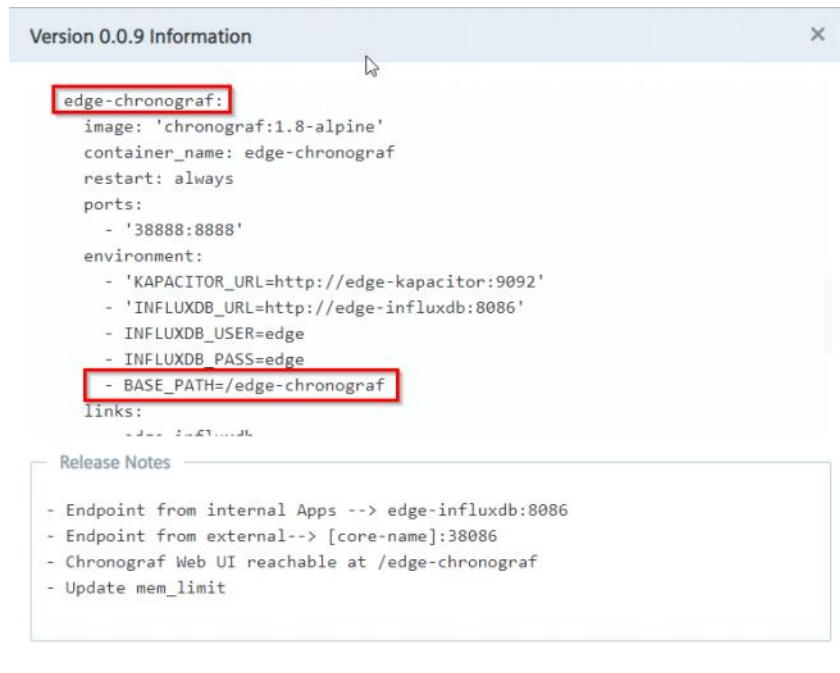
Per impostare la funzionalità di Reverse Proxy, all'interno del software Industrial Edge App Publisher è stata configurata la sezione "Network" sul servizio "edge-chronograf" con i seguenti parametri:

The screenshot shows the 'Network' configuration window. The 'Network Mode' is set to 'Reverse Proxy'. The 'Container Port' is empty. The 'HTTPS' dropdown is set to 'HTTPS'. The 'Service Name' is empty. The 'Custom Header' is empty. The 'Rewrite Target' is empty. A list of rewrite rules is shown, with one rule selected: 'name: edge-chronograf/ protocol: HTTP port: '8888' headers: " rewriteTarget: /edge-chronograf'. The 'Ports' section is empty.

Docker Service	Container Port	Protocol	Service Name	Rewrite Target
----------------	----------------	----------	--------------	----------------

<i>edge-chronograf</i>	8888	HTTP	edge-chronograf/	/edge-chronograf
------------------------	------	------	------------------	------------------

Chronograf prevede la possibilità di configurare il path da utilizzare come Reverse Proxy utilizzando la variabile d'ambiente **BASE_PATH**. Questa variabile d'ambiente è stata impostata direttamente all'interno del docker-compose file, come specificato anche nei precedenti capitoli.



Per maggiori informazioni sulla variabile d'ambiente **BASE_PATH** e sulle altre variabili d'ambiente utilizzate da Chronograf consultare la documentazione ufficiale al link <https://docs.influxdata.com/chronograf/v1.8/administration/config-options/#--basepath---p>.

A - Installazione nodo NodeRED InfluxDB

domenica 28 febbraio 2021 03:52

! N.B. - L'installazione di nodi aggiuntivi nell'applicazione SIMATIC Flow Creator (basato su Node-RED) è consentito, ma questi nodi non sono ufficialmente supportati da Siemens.

1. Aprire la pagina Editor dell'applicazione SIMATIC Flow Creator tramite click sull'icona presente sull'Edge Device utilizzato o utilizzando l'URL [core name]/sfc-root/
2. Dal menù principale navigare nel sottomenù "Manage Palette"
3. Recarsi nella sezione "Install"
4. Cercare il nodo con nome " node-red-contrib-influxdb" e premere il tasto "Install" per iniziare l'installazione
5. Al termine dell'installazione, chiudere il sottomenù tramite il tasto "Close".
6. Se richiesto dal nodo installato, potrebbe essere necessario un riavvio dell'applicazione SIMATIC Flow Creator

Per maggiori informazioni consultare la documentazione ufficiale del nodo [node-red-contrib-influxdb](#).