

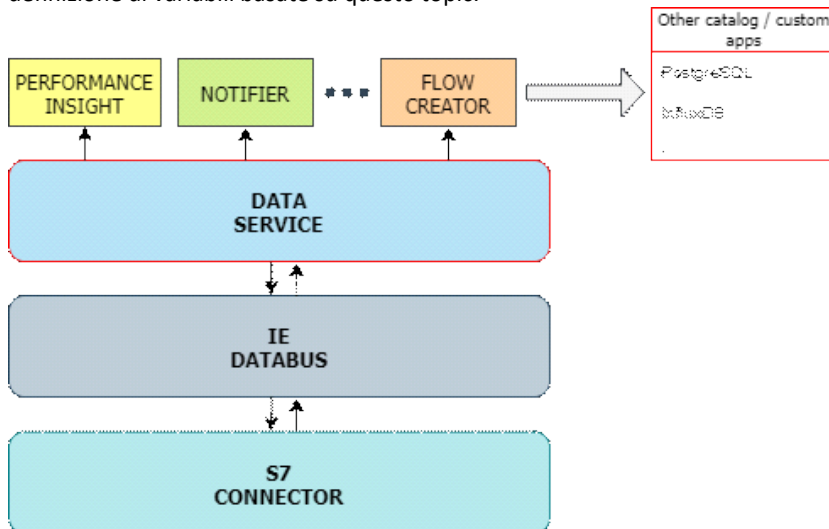
# 1 - Introduzione

giovedì 14 ottobre 2021 13:25

Data Service è un'applicazione Siemens per Industrial Edge dedicata al raggruppamento e all'archiviazione, per un certo periodo di tempo, di dati e serie temporali provenienti da dispositivi di campo o di automazione.

L'applicazione permette il collegamento di app a catalogo, quali ad esempio Performance Insight o Notifier, con IE Databus (MQTT Broker) o con Unified Comfort Panel (Open Pipe).

A tal fine, l'IE Databus riceve i dati direttamente dai dispositivi di automazione tramite opportuni connettori, quali ad esempio S7 Connector. All'interno dell'applicazione Data Service, il topic relativo ai metadati viene letto dall'IE Databus, permettendo così la definizione di variabili basate su questo topic.



## Prima di iniziare

Questa guida riporta un **esempio applicativo**, descritto in dettaglio al paragrafo [5 - Esempio Applicativo](#), che mostra come poter gestire un flusso di dati con l'applicazione Data Service tramite SIMATIC Flow Creator (Node-RED).

In particolare, nell'esempio applicativo verrà mostrato come integrare le tag lette da un PLC all'interno dell'applicazione Data Service e, successivamente, come estrarre i dati dall'app Data Service tramite SIMATIC Flow Creator, con lo scopo di utilizzarli in altre applicazioni. Verrà inoltre mostrato come poter visualizzare i dati letti tramite dashboard di SIMATIC Flow Creator, ed eventualmente esportarli in un file CSV.

## 2 - Esempio Applicativo

giovedì 14 ottobre 2021 13:26

### Prerequisiti

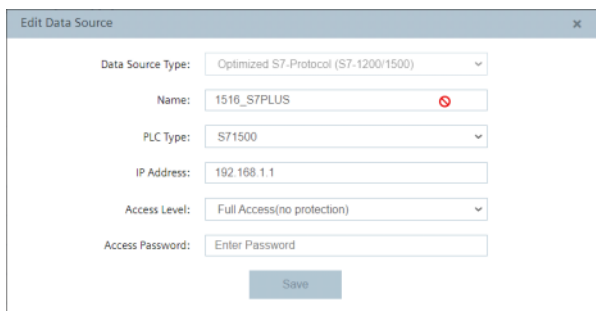
- Un **Edge Device** correttamente configurato e con onboarding effettuato su un sistema **Industrial Edge Management (IEM)** accessibile.
- Per consentire la comunicazione con una sorgente dati S7, un **PLC** della famiglia **SIMATIC** (S7-300, S7-1200, S7-1500,...).
- L'applicazione **SIMATIC S7 Connector** 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** e **configurata** sull'Edge Device utilizzato.
- Il file **Flow\_AppExample\_DataService.json** deve essere importato all'interno dell'applicazione SIMATIC Flow Creator tramite la funzionalità **Import** nel menù dedicato.
- L'applicazione **Data Service** deve essere **installata** sull'Edge Device utilizzato.

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

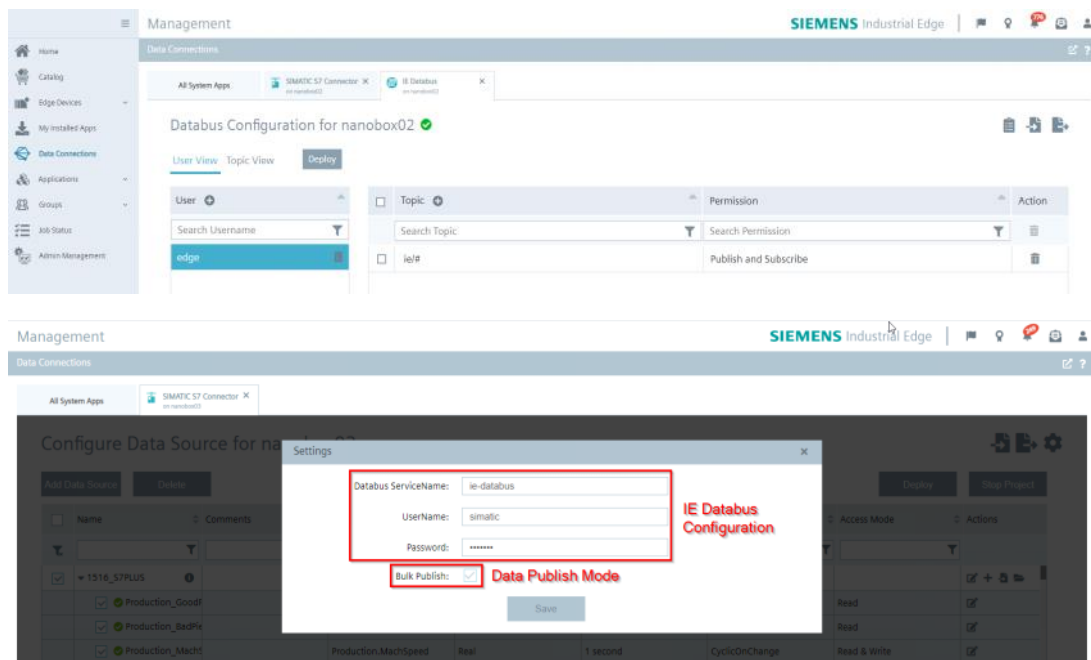
Per poter archiviare dati all'interno dell'applicazione **Data Service**, occorre innanzitutto effettuare uno **scambio dati** con una sorgente in grado di generare ciclicamente nuovi valori.

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

All'interno dell'applicazione S7 Connector, in questo caso, una CPU S7-1500 è stata 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 *edge* e password *edge*):

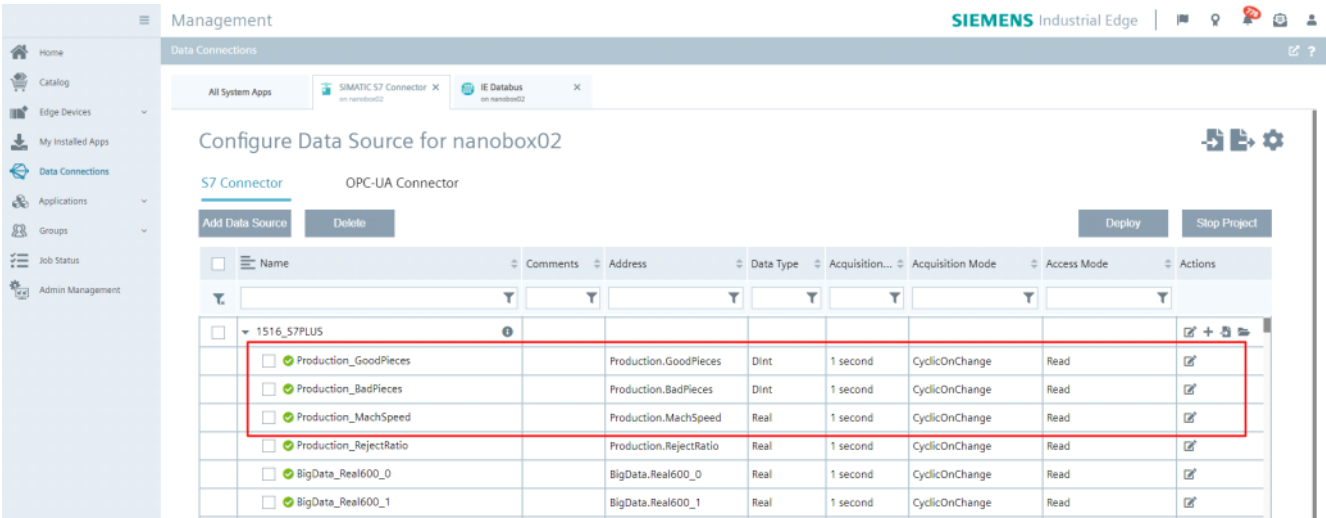


Verranno considerate **3 variabili**, schematizzate nella tabella seguente:

| Id Datapoint                 | Descrizione                          | S7+ Address           | Type | Access Mode |
|------------------------------|--------------------------------------|-----------------------|------|-------------|
| <i>Production_GoodPieces</i> | Contatore Pezzi Prodotti             | Production.GoodPieces | Dint | Read        |
| <i>Production_BadPieces</i>  | Contatore Pezzi Scartati             | Production.BadPieces  | 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 sorgente dati configurata (in questo esempio "1516\_S7PLUS"), che emetterà ciclicamente un messaggio JSON contenente la proprietà **vals**, ovvero un array con 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
      }
    ]
  }
}
```

Nella tabella seguente è specificato il significato delle tipiche **proprietà** del messaggio ottenuto da S7 Connector tramite MQTT in fase di lettura delle variabili, con riferimento al messaggio di esempio sopra riportato:

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

|            |  |                            |
|------------|--|----------------------------|
| <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 sulle applicazioni S7 Connector, IE Databus e sulla lettura variabili tramite protocollo MQTT, 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>

## Configurazione variabili e Aspect con Data Service

Le funzionalità principali dell'applicazione Data Service sono:

- Creazione e configurazione di **Asset**.
- Creazione di **Aspect** e variabili per l'analisi dei dati.
- Collegamento di sorgenti dati ad **Aspect** e variabili.

Dopo aver configurato il topic e le tag di interesse all'interno delle applicazioni IE Databus e S7 Connector, è necessario quindi configurare le stesse all'interno dell'applicazione Data Service.

Questa operazione può essere effettuata accedendo al Data Service e aggiungendo le variabili di interesse all'interno di un **Asset**.

Un **Asset** è una **rappresentazione digitale di una macchina o di un sistema di automazione, con una o più sorgenti dati** (PLC). I dati che descrivono la sorgente vengono quindi raccolti all'interno dell'Asset, per poi essere messi a disposizione per elaborazioni successive.

L'applicazione Data Service prevede la presenza di un **Asset pre-configurato**, chiamato **edge**.

Volendo differenziare tra loro le variabili acquisite, è possibile creare nuovi **Child Asset**, selezionando il tasto **Add child asset**. Tutti i **Child Asset** creati avranno come nodo padre l'Asset **edge** pre-configurato.

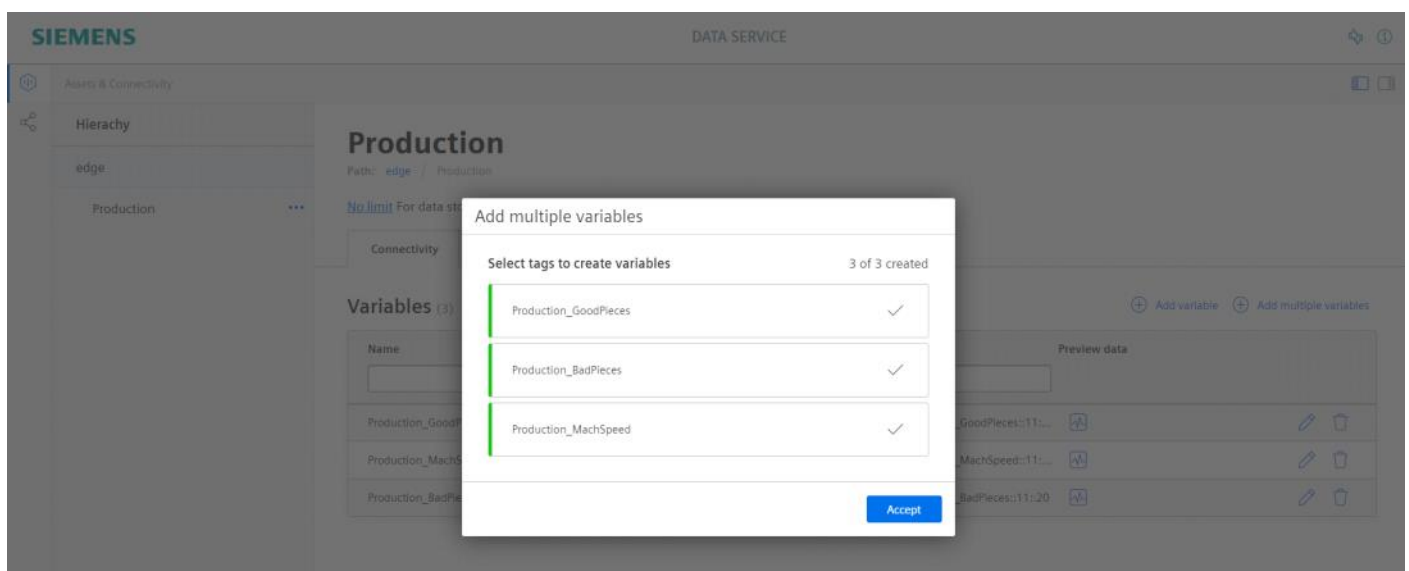
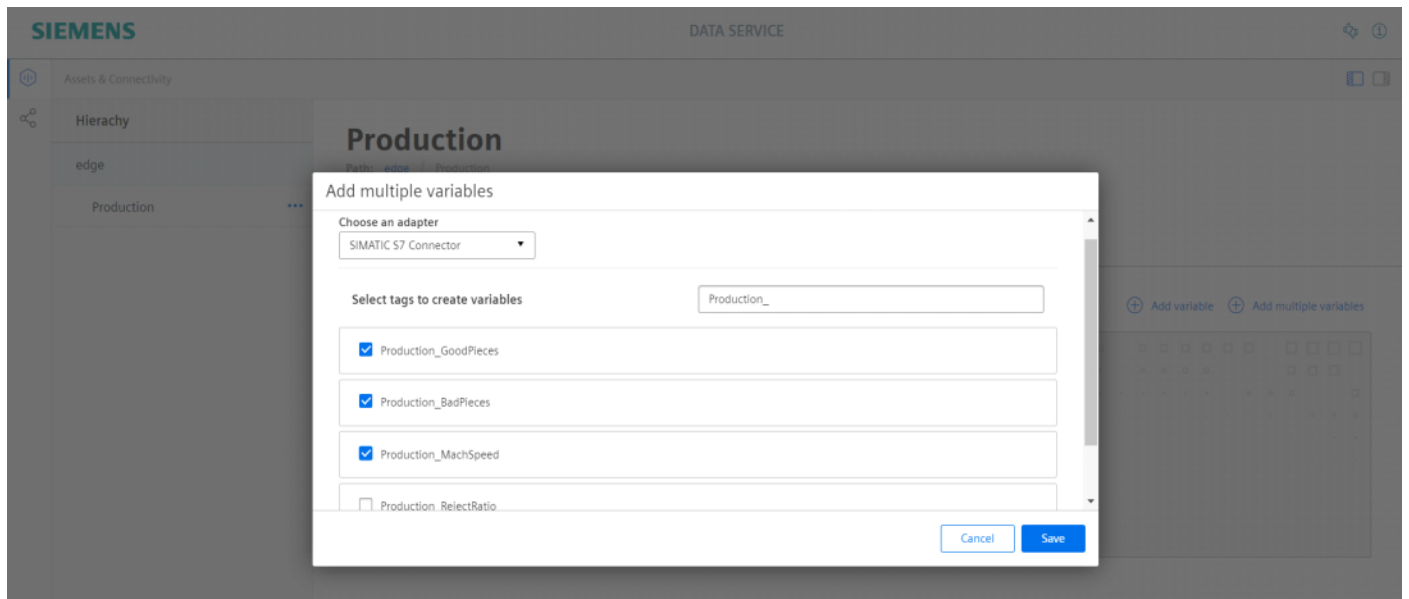
In questo esempio applicativo, le variabili di interesse verranno caricate all'interno del **Child Asset** di nome **Production**, di seguito configurato.

The screenshot shows the Siemens Data Service interface. On the left, a sidebar titled 'Assets & Connectivity' shows a hierarchy with 'edge' selected. The main area displays the 'edge' asset with a path of 'edge' and data stored for 30 days. There are buttons for 'Connectivity' and 'Aspects'. Below this, the 'Variables' section shows a table with two variables: 'SinWave\_1' and 'SinWave\_2', both of type 'Float' and linked to a specific topic. The table has columns for Name, Type, Topic, and Preview data.

The screenshot shows the Siemens Data Service interface with the 'Add asset' dialog box open. The dialog box has a 'Name' field with the value 'Production' and an 'Optional description field'. There are 'Cancel' and 'Add asset' buttons. The background shows the 'edge' asset configuration with the 'Variables' section visible.

Dopo aver creato il nuovo **Child Asset**, è possibile aggiungere una o più variabili, utilizzando i tasti **Add variable** o **Add multiple variables**.

Tramite la funzionalità **Add multiple variables**, in particolare, l'applicazione permette di specificare il connettore all'interno del quale sono state configurate la sorgente dati e le tag di interesse (in questo esempio applicativo SIMATIC S7 Connector). Una volta selezionato il tipo di connettore, la finestra mostrerà le tag configurate all'interno del connettore, e sarà quindi possibile selezionare le tag da importare.



Come evidenziato nel paragrafo precedente, le tag considerate in questo esempio applicativo e che andranno quindi aggiunte all'interno del **Child Asset Production**, sono le seguenti:

|                              |                                      |
|------------------------------|--------------------------------------|
| <i>Production_GoodPieces</i> | Contatore Pezzi Prodotti             |
| <i>Production_BadPieces</i>  | Contatore Pezzi Scartati             |
| <i>Production_MachSpeed</i>  | Velocità di produzione in pezzi/min. |

Per ognuna delle variabili aggiunte, l'applicazione specifica il tipo di dato e il topic su cui i data points verranno acquisiti e, se necessario, permette la modifica delle proprietà associate alle tag stesse.

**SIEMENS** DATA SERVICE

Assets & Connectivity

Hierarchy

- edge
  - Production

Path: edge / Production




No limit For data storage set

Connectivity Aspects

### Production

Variables (3)

+ Add variable + Add multiple variables

| Name                  | Type  | Topic                                    | Preview data  |
|-----------------------|-------|--|---|
| Production_GoodPieces | Int32 | 1516_S7PLUS:Production_GoodPieces:11:... |  |
| Production_MachSpeed  | Float | 1516_S7PLUS:Production_MachSpeed:11:...  |  |
| Production_BadPieces  | Int32 | 1516_S7PLUS:Production_BadPieces:11:20   |  |

**SIEMENS** DATA SERVICE

Assets & Connectivity

Hierarchy

- edge
  - Production

Path: edge / Production

No limit For data storage set

Connectivity Aspects

### Production

Variables (3)

+ Add variable + Add multiple variables

Preview data

GoodPieces:11:...

MachSpeed:11:...

BadPieces:11:20

Choose an adapter

SIMATIC S7 Connector

Choose a tag

1516\_S7PLUS:Production\_GoodPieces

Advanced 1516\_S7PLUS:Production\_GoodPieces:11:21 Production\_GoodPieces Int32

Tag

1516\_S7PLUS:Production\_GoodPieces:11:21

This tag will be used in tag definition.

Name \*

Production\_GoodPieces

This is the name of the tag that is shown later in the applications.

Data type \*

Int32

This is the data type of the variable.

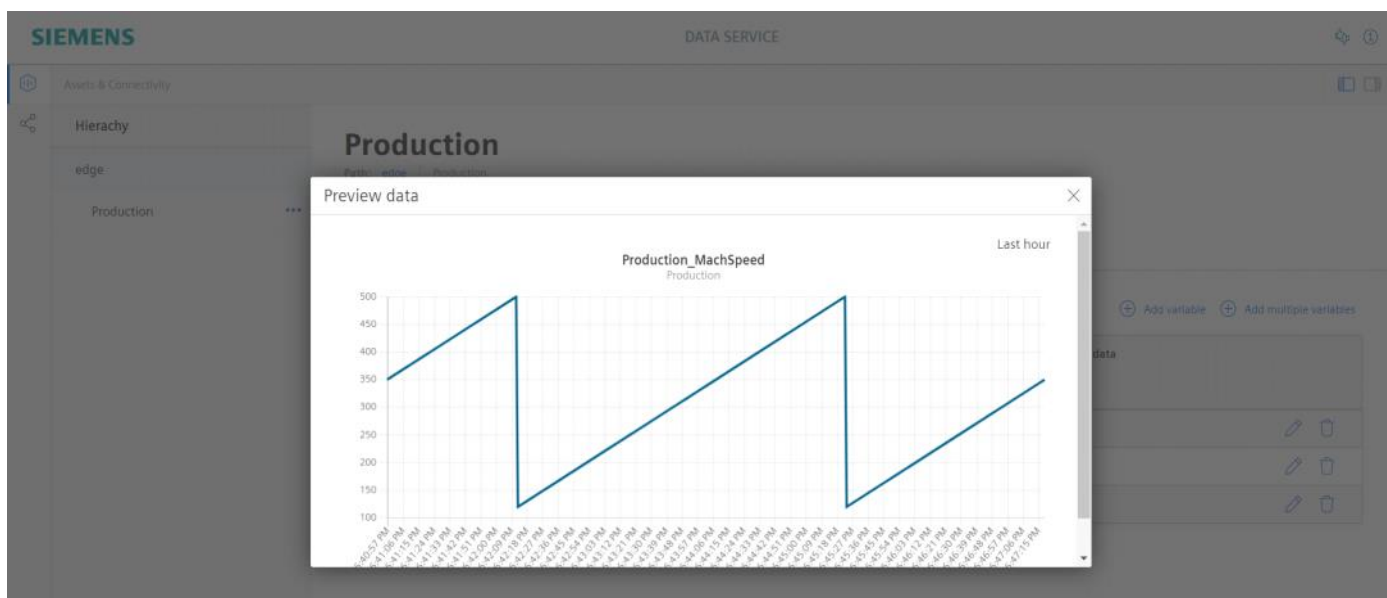
Unit

Unit

This is the unit of the variable that is shown later in the applications.

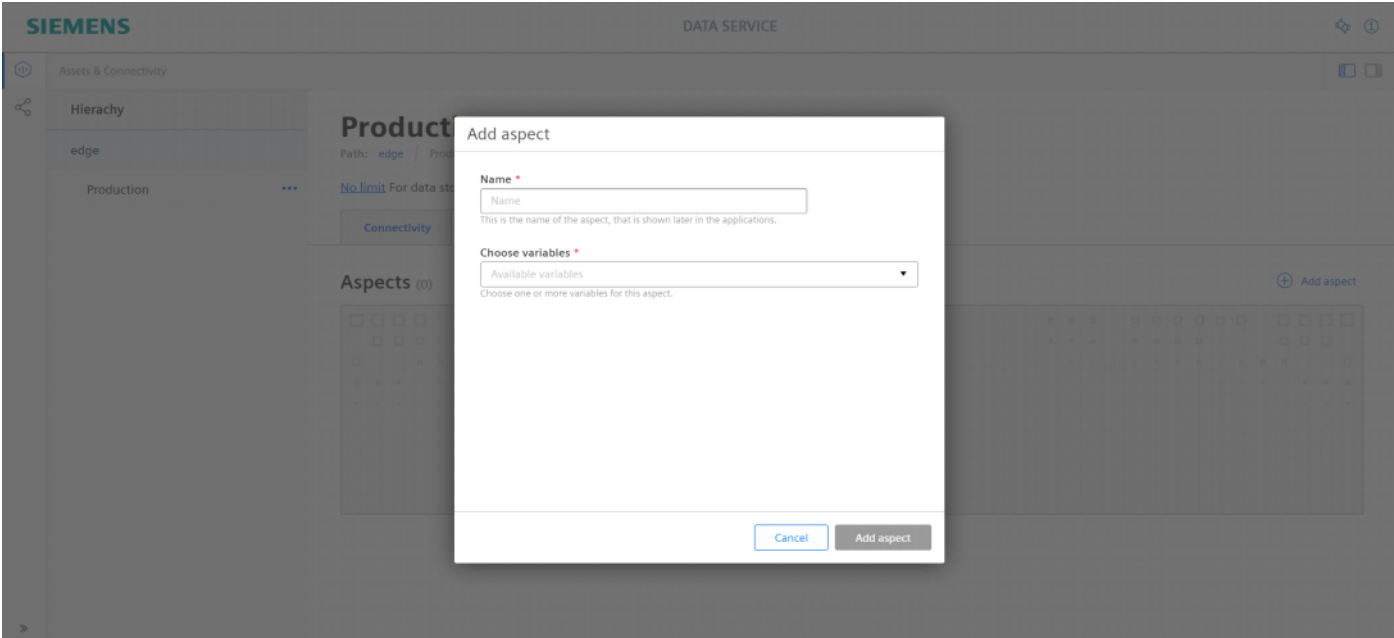
Cancel Edit variable

Cliccando sul tasto **Preview data** è invece possibile visualizzare l'andamento nell'ultima ora della variabile selezionata, come mostrato nell'immagine sottostante per la tag **Production\_MachSpeed**.

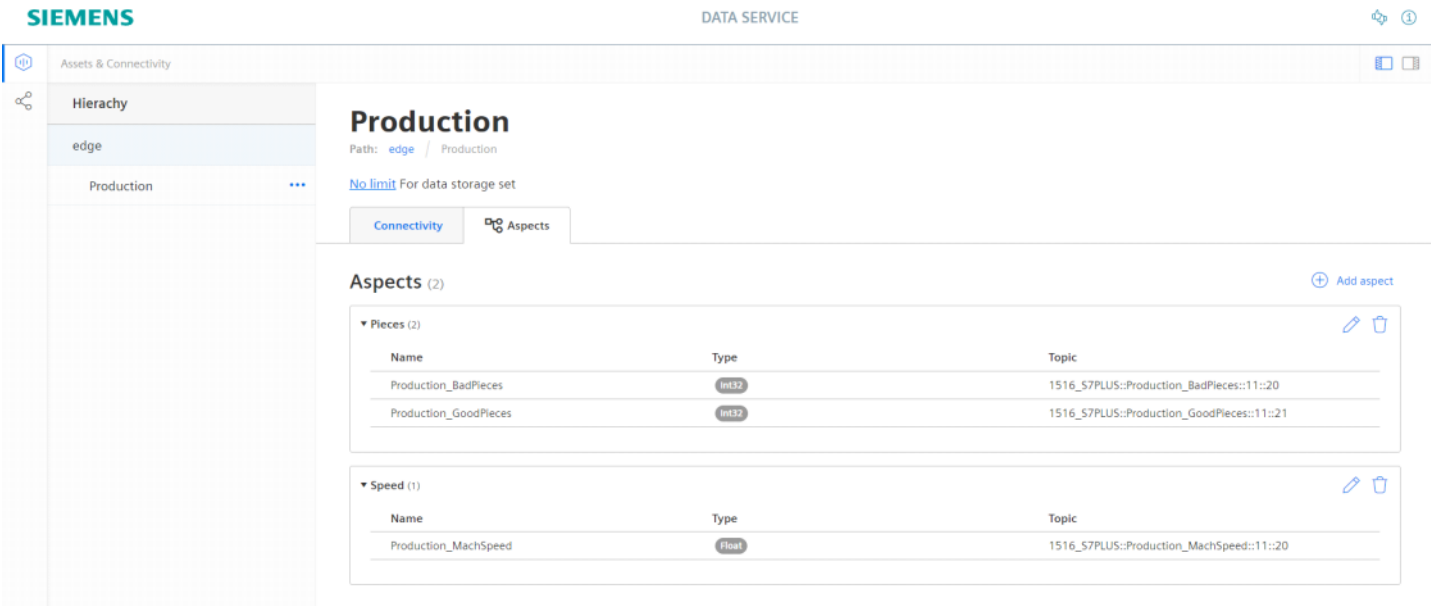
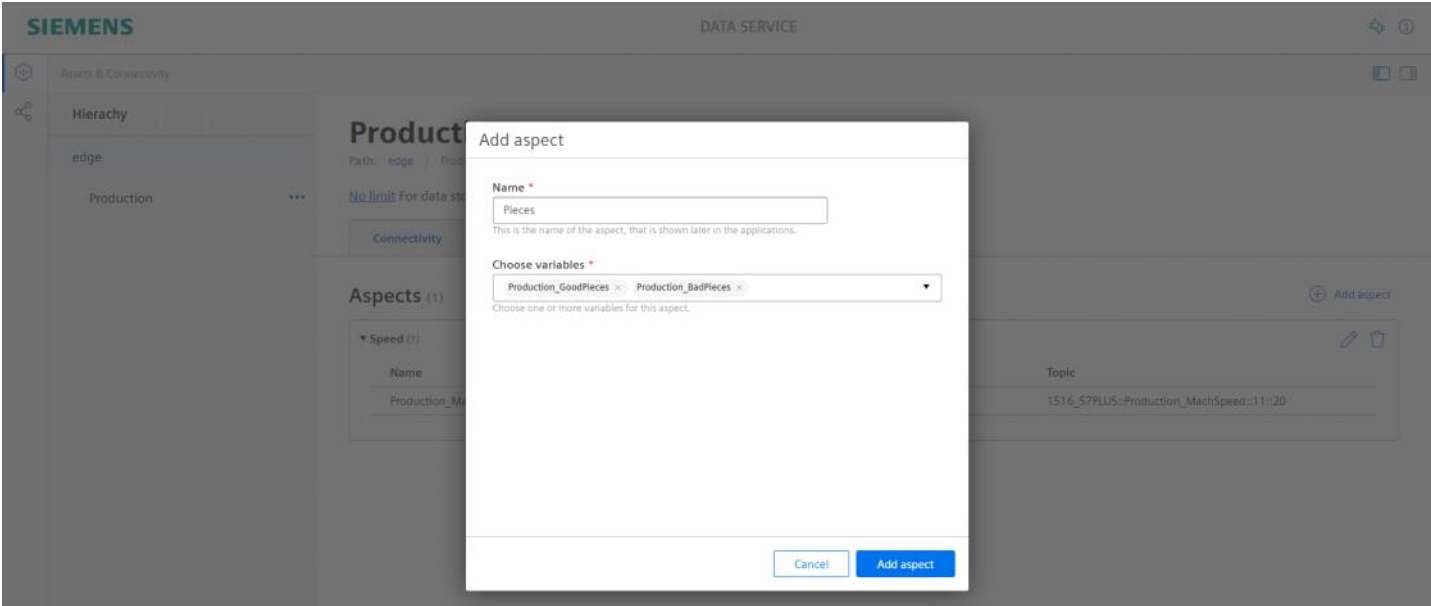


L'applicazione Data Service, oltre alla creazione di **Asset**, permette di creare dei raggruppamenti di variabili chiamati **Aspect**. Un **Aspect** non è altro che un **meccanismo di modellazioni dei dati**, che permette di **raggruppare i data points acquisiti in sottogruppi sulla base della loro assegnazione logica**. Ad esempio, se consideriamo una macchina con **Aspect** "Consumo di energia", questo conterrà tutti i data points di variabili quali "Potenza", "Tensione", ecc. Per poter creare un nuovo **Aspect**, è necessario navigare all'interno della tab **Aspects**, e cliccare su **Add aspect**.

A questo punto, l'applicazione richiederà di definire il nome del nuovo **Aspect**, e tutte le variabili ad esso associate.



Nel caso specifico di questo esempio applicativo, sono stati creati due **Aspect** differenti: **Pieces** per raccogliere i dati relativi al numero di pezzi conformi (**Production\_GoodPieces**) e non conformi (**Production\_BadPieces**) prodotti, e **Speed**, per raccogliere invece la velocità della macchina (**Production\_MachSpeed**).





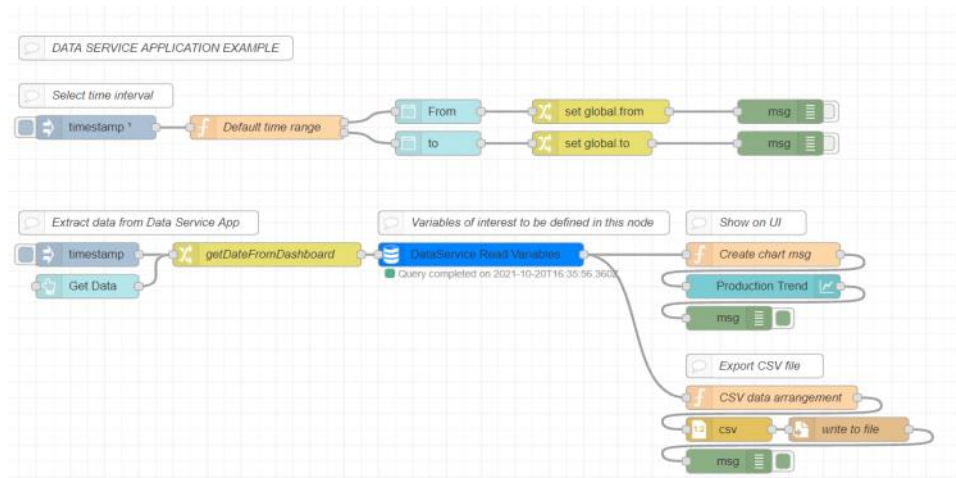
Per maggiori informazioni sull'utilizzo e le funzionalità dell'applicazione Data Service, consultare la documentazione dedicata: <https://support.industry.siemens.com/cs/mdm/109781417?c=148911920651&lc=en-US>

## Estrazione di tag da Data Service tramite SIMATIC Flow Creator

Utilizzando le API esposte dall'applicazione Data Service, questo paragrafo mostra come sfruttare un **nodo sub-flow in SIMATIC Flow Creator per leggere i dati dall'applicazione Data Service in base ai nomi delle tag**.

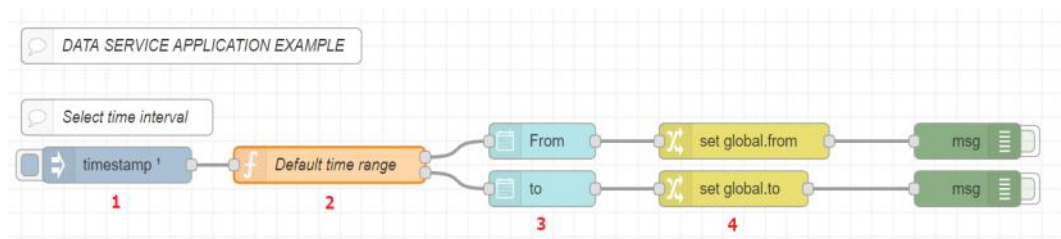
Nel flusso presentato, in particolare, sono disponibili le seguenti funzionalità:

- Selezione dell'intervallo di tempo di estrazione dei dati.
- Estrazione dei dati dal database dell'applicazione Data Service tramite API.
- Visualizzazione dei dati tramite Web dashboard SIMATIC Flow Creator.
- Salvataggio dei dati in un file CSV.

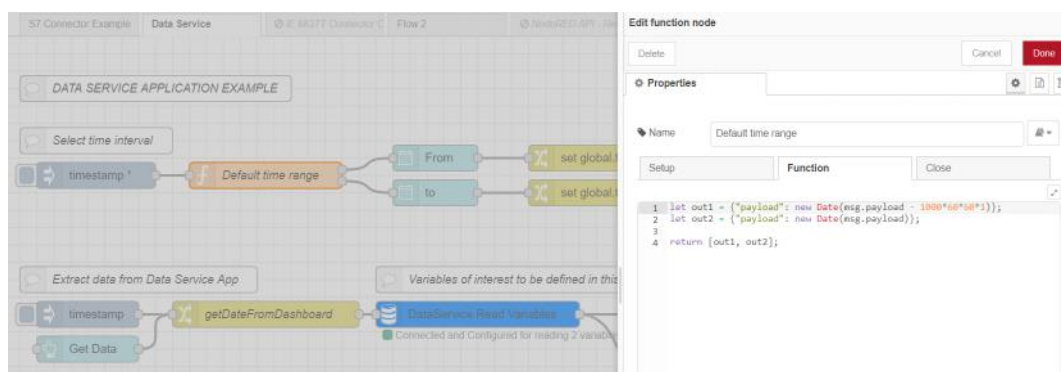


### Selezione dell'intervallo di tempo di estrazione dei dati

Il primo flusso nell'esempio applicativo permette di **impostare un intervallo temporale**, che sarà preso come riferimento per le successive estrazioni dati. A seconda degli intervalli di tempo definiti nel nodo *function* (2) di seguito, il flusso imposterà due variabili globali, **global.from** e **global.to**.



Dal nodo *inject* (1), il nodo *function* (2) prende la data e ora corrente impostandola come valore della variabile globale **global.to**. Per quanto riguarda invece la variabile **global.from**, questa viene impostata sottraendo 1 ora dalla data e ora corrente. In questo modo, l'estrazione dei dati prenderà in considerazione solo i dati dell'ultima ora.

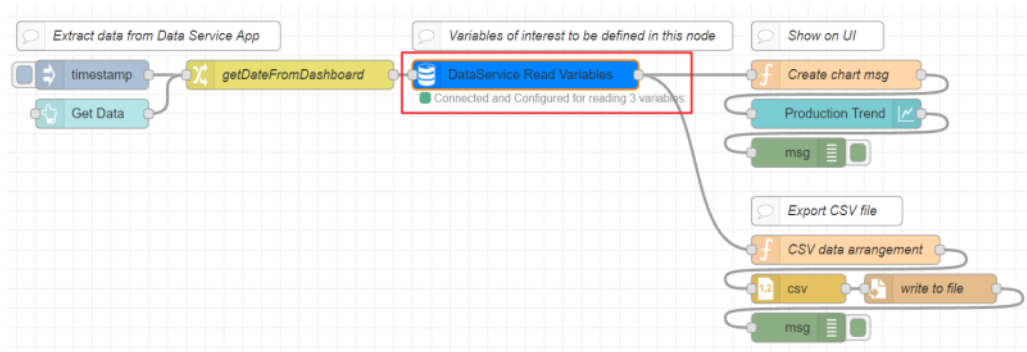


Se necessario, è possibile modificare l'intervallo temporale in modo tale da estrarre un volume maggiore di dati dall'applicazione Data Service.

### Estrazione dati da Data Service tramite API

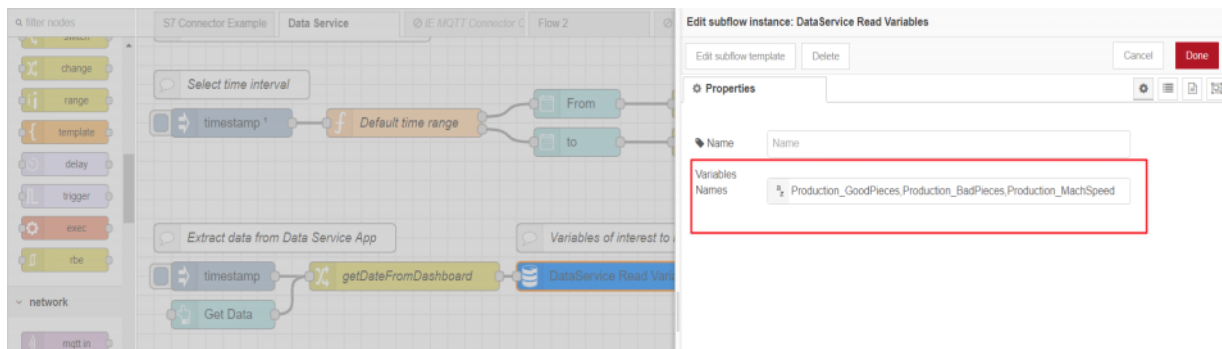
Nel flusso principale, il nodo **DataService Read Variables** prende come input l'intervallo di tempo impostato al paragrafo precedente, insieme all'elenco delle variabili esplicitate nella configurazione del nodo stesso.





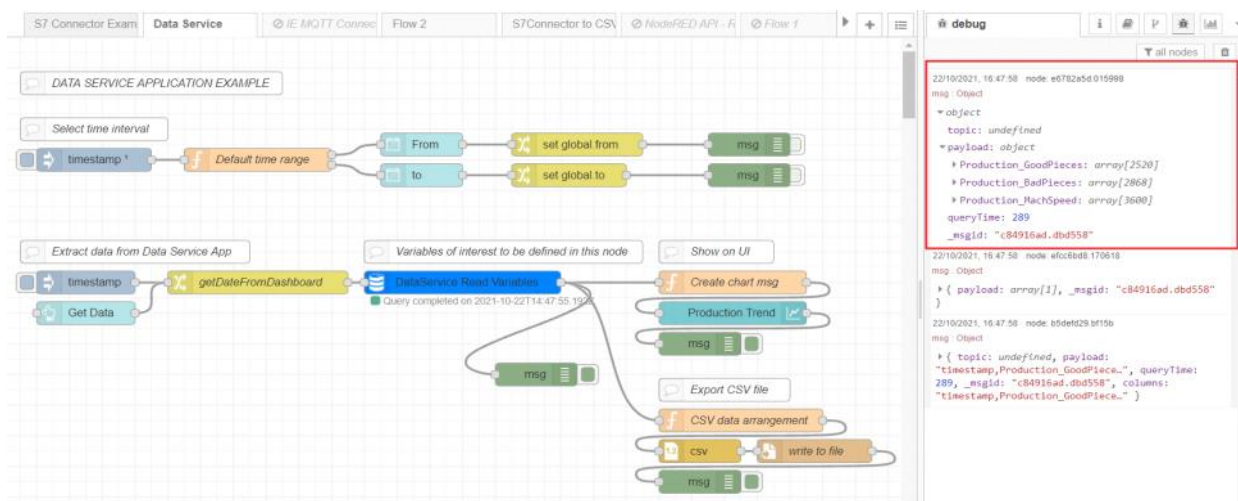
Nel campo **Variables Names** del nodo **DataService Read Variables**, infatti, devono essere esplicitate le **variabili di interesse** da estrarre dal Data Service, **separate da virgole e senza spazi nel mezzo**.

Come si vede nella figura seguente, in questo esempio applicativo sono state configurate tre variabili di interesse: **Production\_GoodPieces**, **Production\_BadPieces**, **Production\_MachSpeed**.

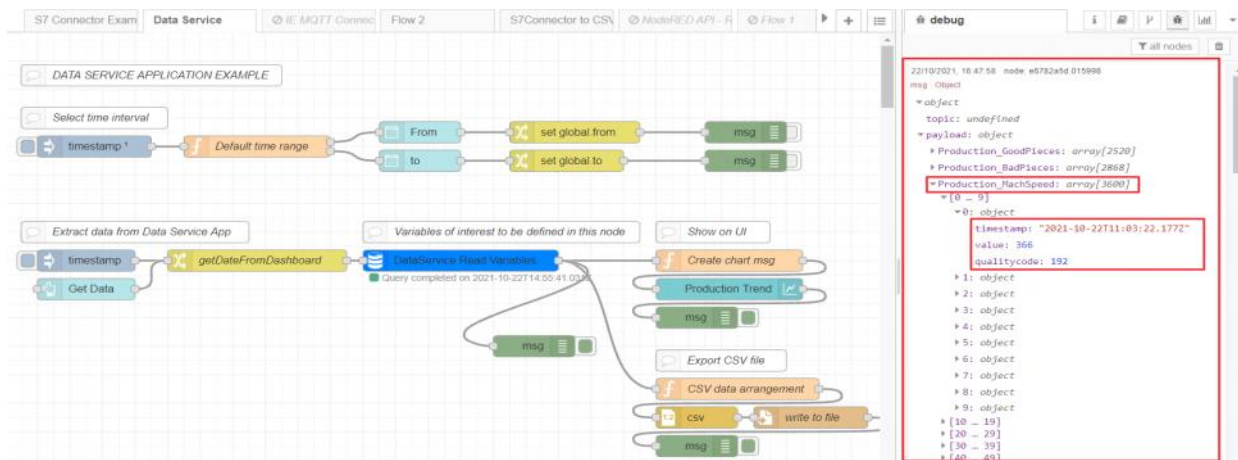


Dopo aver esplicitato le tag di interesse, è necessario attivare l'estrazione dei dati tramite il nodo **inject**.

Lo stato del nodo **DataService Read Variables** passerà da **"Querying data in progress..."** a **"Query completed"**, dando poi in output un messaggio contenente tutti i data points delle variabili di interesse.

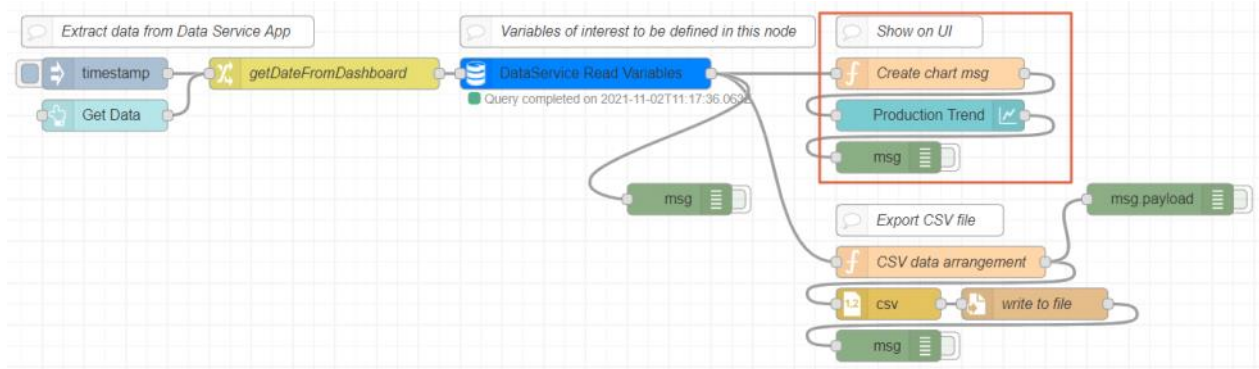


Nel messaggio di output, ad ogni variabile è associato un array di oggetti, dove ogni oggetto rappresenta un data point della tag. Ogni oggetto è un datapoint delle tag estratte dal database di Data Service, caratterizzato da uno specifico timestamp, valore e quality code, come si può vedere dall'immagine sottostante.



## Visualizzazione dati tramite dashboard di SIMATIC Flow Creator

Per visualizzare i dati ottenuti dal flusso precedentemente descritto, è possibile utilizzare la funzionalità **Web Dashboard di SIMATIC Flow Creator** e i nodi della libreria **node-red-dashboard**, dedicata alla rappresentazione di diversi oggetti grafici quali indicatori, campi di testo e grafici. Il nodo **charts-ui** qui utilizzato (**Production Trend**) permette di visualizzare diversi tipi di grafici (linee, barre, torta) sulla Web Dashboard di SIMATIC Flow Creator, sia inviando nuovi dati in tempo reale, che inviando l'intera struttura dati.

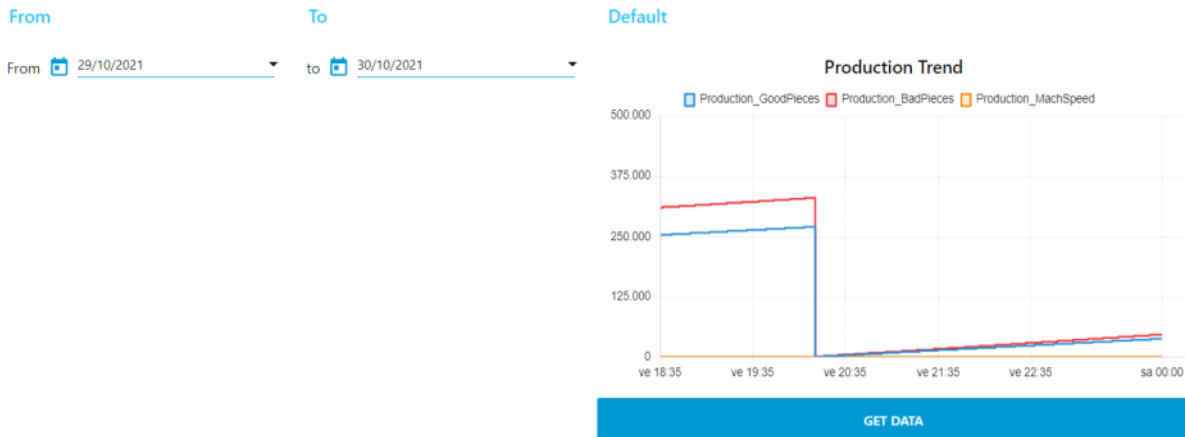


Nel flusso sopra evidenziato, a partire dai dati ricevuti dal nodo **DataService Read Variables**, la funzione **Create chart msg** formatta tutti i dati secondo gli standard del nodo **charts-ui**, con l'obiettivo di creare una struttura dati per rappresentare un grafico a linee contenente tre serie temporali (**Production\_GoodPieces**, **Production\_BadPieces**, **Production\_MachSpeed**) nell'intervallo di tempo selezionato. Ogni serie temporale formattata dalla funzione **Create chart msg** conterrà diversi campioni e i relativi istanti temporali.

Di seguito un esempio del messaggio di output ricevuto dal nodo funzione **Create chart msg**:

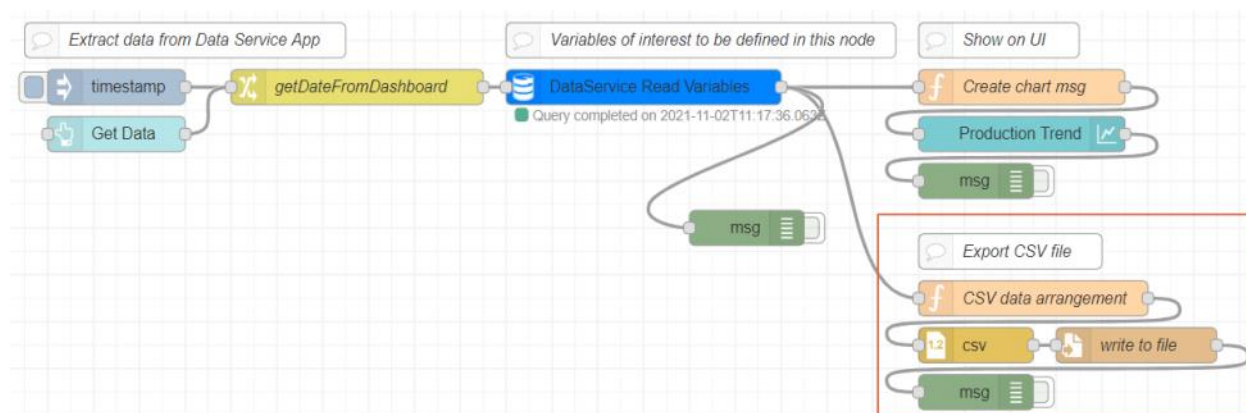
```
2/11/2021, 12:17:38 node: 99d04bcd.554328
msg.payload: array[1]
▼ array[1]
▼ 0: object
  series: array[3]
    0: "Production_GoodPieces"
    1: "Production_BadPieces"
    2: "Production_MachSpeed"
  labels: array[1]
  data: array[3]
    0: array[2555]
      ▼ [0 ... 9]
        0: object
          x: 1635848252097
          y: 75361
        1: object
          x: 1635848253117
          y: 75363
        2: object
```

I dati delle tre variabili di interesse possono essere visualizzati direttamente come tre serie temporali su un grafico a linee, visibile collegandosi alla Web dashboard di SIMATIC Flow Creator, come mostrato nell'immagine seguente:



## Salvataggio dati in CSV

Per consentire l'archiviazione locale dei dati, questo esempio applicativo sfrutta il nodo **csv** della libreria **node-red** per **salvare ed esportare un file in formato CSV** contenente tutte le serie temporali delle variabili **Production\_GoodPieces**, **Production\_BadPieces**, **Production\_MachSpeed** nell'intervallo di tempo selezionato.



Per fare ciò, il nodo **function CSV data arrangement** viene utilizzata per formattare i dati ricevuti dal nodo **DataService Read Variables** nello standard del nodo **csv**. Il nodo function, infatti, considera tutti gli elementi organizzandoli come segue:

```
2/11/2021, 16:01:16 node: d833befe.84b24
msg: Object
  object
    topic: undefined
    payload: array[3603]
      [0 ... 9]
        0: object
          timestamp: "2021-11-02T14:01:07.906Z"
          Production_GoodPieces: 113005
          Production_BadPieces: 138177
          Production_MachSpeed: 302
        1: object
          timestamp: "2021-11-02T14:01:08.897Z"
          Production_GoodPieces: 113008
          Production_BadPieces: 138180
          Production_MachSpeed: 304
        2: object
          timestamp: "2021-11-02T14:01:09.911Z"
          Production_GoodPieces: null
          Production_BadPieces: 138186
          Production_MachSpeed: 306
        3: object
        4: object
        5: object
        6: object
        7: object
        8: object
        9: object
```

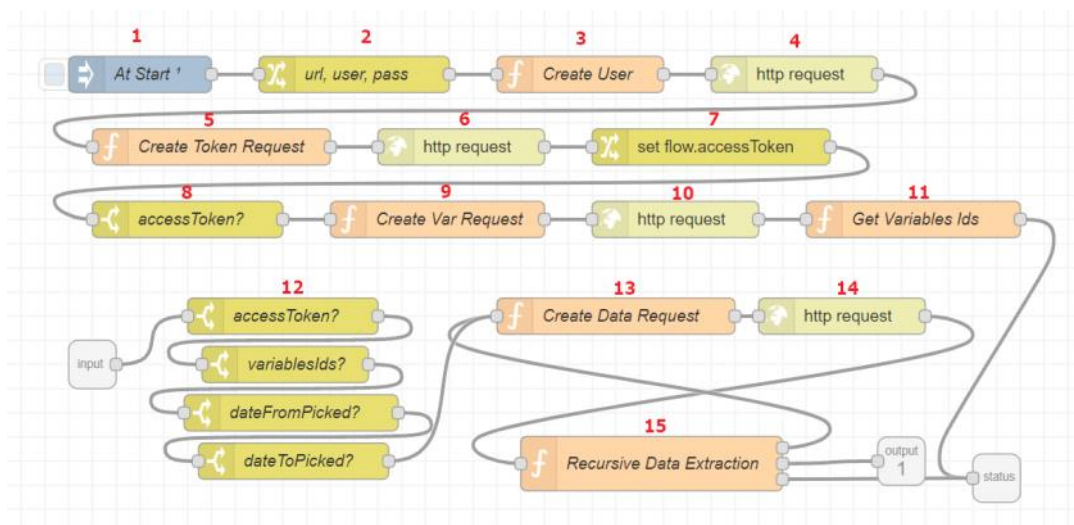
Una volta che i dati sono stati formattati nel modo corretto ed inviati al nodo **csv**, il file CSV viene salvato dal flusso negli **Application Volumes** del dispositivo edge su cui è installata l'applicazione SIMATIC Flow Creator. Questa operazione viene eseguita dal nodo **file** della libreria **node-red**. Per scaricare il file CSV creato, basta semplicemente accedere al dispositivo edge, selezionare l'applicazione SIMATIC Flow Creator all'interno del menu **Management** e cliccare sull'icona di **Download**. Il file verrà così salvato localmente sul proprio PC.

Di seguito un esempio di file CSV salvato dal flusso:

|    | A                | B                     | C                    | D                    |
|----|------------------|-----------------------|----------------------|----------------------|
| 1  | timestamp        | Production_GoodPieces | Production_BadPieces | Production_MachSpeed |
| 2  | 29/10/2021 18:35 | 252961                | 309720               | 312                  |
| 3  | 29/10/2021 18:35 |                       | 309726               | 314                  |
| 4  | 29/10/2021 18:35 |                       | 309732               | 316                  |
| 5  | 29/10/2021 18:35 |                       | 309738               | 318                  |
| 6  | 29/10/2021 18:35 | 252964                | 309741               | 320                  |
| 7  | 29/10/2021 18:35 | 252967                | 309744               | 322                  |
| 8  | 29/10/2021 18:35 | 252970                | 309747               | 324                  |
| 9  | 29/10/2021 18:35 | 252973                | 309750               | 326                  |
| 10 | 29/10/2021 18:35 | 252976                | 309753               | 328                  |
| 11 | 29/10/2021 18:35 | 252979                | 309756               | 330                  |
| 12 | 29/10/2021 18:35 | 252985                |                      | 332                  |
| 13 | 29/10/2021 18:35 |                       | 309762               | 334                  |
| 14 | 29/10/2021 18:35 | 252988                | 309765               | 336                  |
| 15 | 29/10/2021 18:35 | 252991                | 309768               | 338                  |
| 16 | 29/10/2021 18:35 | 252994                | 309771               | 340                  |
| 17 | 29/10/2021 18:35 | 252997                | 309774               | 342                  |
| 18 | 29/10/2021 18:35 |                       | 309780               | 344                  |
| 19 | 29/10/2021 18:35 | 253003                |                      | 346                  |
| 20 | 29/10/2021 18:35 |                       | 309786               | 348                  |
| 21 | 29/10/2021 18:35 | 253006                | 309789               | 350                  |
| 22 | 29/10/2021 18:35 | 253009                | 309792               | 352                  |
| 23 | 29/10/2021 18:35 | 253012                | 309795               | 354                  |
| 24 | 29/10/2021 18:35 | 253015                | 309798               | 356                  |
| 25 | 29/10/2021 18:35 | 253018                | 309801               | 358                  |
| 26 | 29/10/2021 18:35 | 253021                | 309804               | 360                  |

## Sub-Flow DataService Read Variables

In questo paragrafo verrà descritto nel dettaglio il funzionamento del sub-flow **DataService Read Variables**.



Di seguito, per ogni nodo enumerato nella figura soprastante, viene riportata una breve spiegazione.

1. Nodo *inject* per triggerare le operazioni successive.
2. Il nodo imposta il valore delle variabili di flusso di seguito elencate:  
**dataserviceUrl:** "<http://edgeappdataservice:4203>"  
**dataserviceUser:** "nodeUser"  
**dataservicePass:** "nodePass"
3. Nodo che utilizza le variabili di flusso sopra definite per creare uno **user** e definirne le proprietà:  
username: "nodeUser",  
password: "nodePass",  
familyName: "myFamily nodeUser",  
givenName: "nodeUser",  
e-mail: "nodeUser@myemail.com",  
roles: [{application: "edgeappdataservice", role: "admin"}]}
4. Nodo di **richiesta HTTP** utilizzato per inviare (**POST**) i dati dell'utente creato al seguente URL: <http://edgeappdataservice:4203/TokenManagerService/users>.
5. Il nodo sfrutta la **bearer authentication** con **codifica base64** per generare una **richiesta di token** per l'utente creato.
6. Nodo di **richiesta HTTP** utilizzato per inviare (**POST**) i dati del token al seguente URL: <http://edgeappdataservice:4203/TokenManagerService/oauth/token>. In risposta alla richiesta di login e token del nodo precedente, il server genererà un **bearer token**, ovvero una stringa criptata utilizzata nelle richieste successive. Questa stringa verrà infatti utilizzata nell'header di richieste fatte verso risorse protette, come ad esempio i dati nel database del Data Service.
7. Il nodo imposta il token di accesso come variabile di flusso (**accessToken**).
8. Verifica se la variabile di flusso **accessToken** è nulla.
9. Crea una richiesta variabili così da estrarre tutte le variabili dal database del Data Service.
10. Nodo di **richiesta HTTP** utilizzato per ottenere (**GET**) i dati delle variabili al seguente URL <http://edgeappdataservice:4203/DataService/Variables> tramite la API mostrata di seguito.





```

▼ object
  ▶ headers: object
    method: "GET"
    url: "http://edgeappdataservice:4203/DataService/Data?variableIds=[\"561d1b9bf99b474a8560c08e2b1f5c60\", \"6b408b27e48448f990fe8815a7562c85\", \"6ed033533bb74a9298eff3ea6f96a40c\"]&from=2021-10-21T22:37:19.074Z&to=2021-10-21T23:37:19.074Z&order=Ascending"
    _msgid: "4d8e8dd1.4b4e24"
    statusCode: 200
    responseUrl: "http://edgeappdataservice:4203/DataService/Data?variableIds=[\"561d1b9bf99b474a8560c08e2b1f5c60\", \"6b408b27e48448f990fe8815a7562c85\", \"6ed033533bb74a9298eff3ea6f96a40c\"]&from=2021-10-21T22:37:19.074Z&to=2021-10-21T23:37:19.074Z&order=Ascending"
  ▼ payload: object
    data: array[3]
      ▼ 0: object
        variableId: "561d1b9bf99b474a8560c08e2b1f5c60"
        values: array[1432]
      ▼ 1: object
        variableId: "6b408b27e48448f990fe8815a7562c85"
        values: array[1576]
      ▼ 2: object
        variableId: "6ed033533bb74a9298eff3ea6f96a40c"
        values: array[2000]
      ▼ hasMoreData: object
        from: "2021-10-21T23:10:38.697Z"
        to: "2021-10-21T23:37:19.074Z"
      redirectList: array[0]
    responseCookies: object

```

15. Come indicato nelle figure precedenti, l'API **/DataService/Data/{variableId}** consente la lettura dati con un limite massimo di 2000 data points. Per estrarre più data points è possibile sfruttare la proprietà **hasMoreData** del payload, nella quale è contenuto il periodo di tempo con datapoint non inclusi nella risposta.

Lo scopo del nodo (15) è quindi quello di effettuare delle chiamate ricorsive alla stessa API fino alla completa risoluzione di tutti i data points nell'intervallo di tempo richiesto. Quando tutti i data points sono stati recuperati, l'output sarà del nodo sarà il seguente:

```

22/10/2021, 01:46:14 node: e6782a5d.015998
msg: Object
▼ object
  topic: undefined
  ▼ payload: object
    Production_GoodPieces: array[2558]
    Production_BadPieces: array[2838]
    Production_MachSpeed: array[3600]
  queryTime: 337
  _msgid: "31d05d62.d07442"

```

Dove, per ogni variabile estratta, i data points saranno caratterizzati da timestamp, valore e quality code.

```

22/10/2021, 01:46:14 node: e6782a5d.015998
msg: Object
▼ object
  topic: undefined
  ▼ payload: object
    ▼ Production_GoodPieces: array[2558]
      ▼ [0 ... 9]
        ▼ 0: object
          timestamp: "2021-10-21T22:46:10.719Z"
          value: 4008
          qualitycode: 192
        ▼ 1: object
        ▼ 2: object
        ▼ 3: object
        ▼ 4: object
        ▼ 5: object
        ▼ 6: object
        ▼ 7: object
        ▼ 8: object
        ▼ 9: object
      ▼ [10 ... 19]
      ▼ [20 ... 29]
      ▼ [30 ... 39]
      ▼ [40 ... 49]

```