MENTOR

Mobilize Efforts for a harmonized diffusion of New smart and shared Mobility Technologies in the Overall program Regions

ID: 562850

Bike Sharing Merano Data Collector
Specifiche di integrazione nell'Open Data Hub

v1.0, 17.10.2019

Capofila IT / Lead Partner IT	Comune di Merano / Gemeinde Meran
Capofila CH / Lead Partner CH	Comune di Brig-Glis / Brig-Glis
Partner 1 IT	NOI Techpark
Partner 2 IT	SASA
Partner 1 CH	Postauto

Indice

Note preliminari.....4

Metadati stazioni.....4

Metadati tipi.....4

Dati.....5

Note preliminari

Alla luce della struttura dell'API del Data Provider (Digital Mobility Solutions GmbH, piattaforma MOQO), si è deciso in fase di avvio dei lavori di scegliere l'approccio di modellare i singoli veicoli (bici) come "station", secondo il concetto usato nell'Open Data Hub. Non saranno modellate nell'Open Data Hub le stazioni "virtuali" del servizio di bike sharing in quanto non restituite esplicitamente dall'API esposta dalla piattaforma MOQO.

L'end-point alla piattaforma MOQO è https://portal.moqo.de/api

Metadati stazioni

Attraverso il metodo "/cars" sono disponibili molteplici informazioni sul veicolo. In fase di implementazione va effettuata una verifica di quali attributi sono effettivamente valorizzati / presenti rispetto alla specifica fornita, dal momento che la maggior parte si applicano solo al caso in cui i veicoli sono delle auto.

Le informazioni relative alle "station" bici da memorizzare nella tabella station sono quelle indicate in . Tutti gli altri attributi sono da salvare come json nella tabella **metadata**, creando un opportuno riferimento con la stazione di riferimento (colonna station_id).

Campi file webservice	Colonne tabella stations del database
id	**stationcode**
license	**name**
location	**pointprojection**

Tabella 1: Mapping tra gli attributi del web-service della piattaforma MOQO e la tabella "station" del database.

Alcune osservazioni di dettaglio:

- i valori delle colonne **active** e **available** sono da settare a true.
- il valore della colonna **parent_id** non è da valorizzare.
- il valore della colonna **origin** è da settare con **BIKE_SHARING_MERANO**.
- il valore della colonna **stationtype** nel nostro database è da settare come **Bicycle**;
- i valori delle colonne **id** e **meta_data_id** (id corrispondente nella tabella metadata) vanno gestiti automaticamente in fase di scrittura del nuovo record nella tabella;
- è da verificare l'effettiva presenza e valorizzazione del campo **license** nell'API MOQO.

Metadati tipi

I metadati dei tipi gestiti dal sistema vanno salvati nella tabella *type* del database. In questo caso, alcuni attributi legati alla "station" bici sono da memorizzare anche come tipi così da

permetterne una storicizzazione dei dati associati che variano nel tempo. Con riferimenti ai campi dell'API esposta dalla piattaforma MOQO, i tipi da salvare sono i seguenti:

- **available** (indica se la bici è disponibile o meno per un noleggio). Va riutilizzato il tipo "availability" già gestito nell'Open Data Hub.
- **location** (indica la posizione corrente di una bici)
- **available_from** e **available_until** (fornisce un'indicazione eventuale rispetto ad una disponibilità limitata della bici. Nello specifico se una bici è attualmente non disponibile, va considerato il campo available_from per capire quando sarà nuovamente disponibile; viceversa se una bici è libera va considerato il campo available_until per capire fino a quando sarà disponibile).
- **in_maintenance** (indica se la bici è in manutenzione, se non disponibile).

Dati

Lo stato delle biciclette viene aggiornato ad ogni variazione di stato di una bici o in caso di un'operazione di un'utente tramite l'APP (es. richiesta prenotazione di una bici). Dal momento che i dati sono disponibili solo tramite GET (meccanismo pull) e non POST (meccanismo push), bisogna garantire una frequenza di aggiornamento nei dati piuttosto elevata in modo da fornire l'informazione più attualizzata possibile. Si propone di configurare inizialmente il sistema con chiamate all'API MOQO **ogni 5 minuti**.

I dati sono da salvare nelle tabelle measurement / measurementstring (in cui vengono salvati solo i valori più recenti) e measurementhistory / measurementstringhistory. Nello specifico i tipi available e in_maintenance sono da salvare in measurement e measurementhistory, in quanto booleani (0 = available; 1 = not available), mentre i tipi location, available_from e available_until in measurementstring e measurementstringhistory. Alcune osservazioni di dettaglio:

- Il valore della colonna **period** nel nostro database va settata con **300** (vedi considerazioni sopra).
- Il valore della colonna **id** va gestito automaticamente in fase di scrittura del nuovo record nella tabella.
- Il valore della colonna **created_on** va gestito automaticamente in fase di scrittura del nuovo record nella tabella (è il timestamp relativo all'operazione di scrittura). Dal momento che l'API non fornisce alcuna indicazione sul timestamp in quanto i dati rappresentano lo stato attuale delle bici, tale valore è da memorizzare anche per la colonna **timestamp**.
- Il valore della colonna **double_value** è da valorizzare con il valore restituito dall'API MOQO.
- Il valore della colonna **provenance** è da valorizzare rispetto all'ID del record corrispondente nella tabella provenance.

- I valori delle colonne **station_id** e **type_id** sono da valorizzare in maniera appropriata in fase di scrittura dei record.

Sono da considerare i seguenti accorgimenti per una gestione efficiente del salvataggio dei dati:

- Per motivi di privacy, il dato legato alla location è da salvare solo se available = TRUE.