UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Dipartimento di Ingegneria “Enzo Ferrari”

Corso di laurea triennale in Ingegneria Informatica

Gestione dei dati prodotti da una comunità energetica

Relatore:

Prof. Sonia Bergamaschi

Correlatore:

Prof. Domenico Beneventano

Candidato:

Thomas Sommariva

Anno Accademico 2020/2021

Indice

1. Introduzione

In questo elaborato viene descritta la progettazione di un database per la gestione dei dati prodotti da una comunità energetica. Il progetto si è svolto in merito ad una collaborazione in atto tra ENEA e il gruppo di ricerca "DBGroup" del Dipartimento di Ingegneria "Enzo Ferrari" di Unimore.

Partendo da un database relazionale in fase di sviluppo di ENEA, di cui è disponibile uno schema relazionale draft, con indicate le primary key e foreign key, gli obiettivi del lavoro di tesi possono essere sintetizzati come segue

1. effettuare un'analisi ed il reverse engineering di tale schema relazionale draft per ricavarne il corrispondente schema ER
2. Lo schema ER ottenuto rappresenterà una importante documentazione del database in questione
3. inoltre, grazie allo schema ER si possono individuare più facilmente eventuali criticità/problemi dello schema relazionale iniziale; questi problemi vengono discussi con ENEA, e vengono delineate opportune soluzioni. Queste soluzioni vengono prima riportate sullo schema ER ottenendo quindi uno Schema ER modificato e quindi sul relativo schema relazionale tramite la progettazione logica-relazionale.

2. ENEA

In questa sezione, dopo una breve descrizione dell'Agenzia ENEA, verranno introdotti i progetti di ENEA inerenti al lavoro di tesi svolto.

> "L'ENEA è l'**Agenzia nazionale per le nuove tecnologie, l'energia e lo sviluppo economico sostenibile**, *ente di diritto pubblico finalizzato alla ricerca, all'innovazione tecnologica e alla prestazione di servizi avanzati alle imprese, alla pubblica amministrazione e ai cittadini nei settori dell'energia, dell'ambiente e dello sviluppo economico sostenibile"*
>
> *(art. 4 Legge 28 dicembre 2015, n. 221).*

ENEA è un'agenzia nazionale italiana con fini di ricerca che opera nei settori dell'ambiente, dell'energia e delle nuove tecnologie a supporto dello sviluppo sostenibile.

Le sue attività riguardano i seguenti campi:

- Fonti di energia rinnovabili
- Efficienza energetica
- Fusione nucleare
- Sicurezza e salute
- Ambiente e cambiamento climatico
- Tecnologie per il patrimonio culturale

Su queste tematiche ENEA esegue attività di ricerca collaborando con numerosi enti ed istituzioni nazionali ed internazionali attraverso laboratori specializzati e strumentazioni avanzate, sviluppa nuove tecnologie, fornisce servizi ad alto contenuto tecnologico, studi e valutazioni a soggetti pubblici

e privati. ENEA svolge inoltre attività di formazione e informazione con lo scopo di accrescere le competenze di settore.

I progetti di ENEA che vogliamo descrivere si collocano nell'ambito delle cosiddette Comunità energetiche locali, introdotte brevemente nella sottosezione che segue. Le descrizioni dei progetti riportati nel seguito sono state in parte ricavate dalla documentazione fornita da ENEA.

2.1. Comunità energetiche locali

I cambianti climatici che stanno avendo luogo sul nostro pianeta nell'ultimo secolo, hanno evidenziato in maniera inequivocabile gli effetti che la vita dell'uomo sulla terra sta avendo sull'ambiente. È quindi diventata improrogabile una transizione energetica verso un nuovo modello di organizzazione sociale basato su produzione e consumo di energia proveniente da fonti rinnovabili, piuttosto che dai combustibili fossili come è avvenuto fino ad oggi; perché questa transizione possa avere luogo è però necessario che avvenga prima di tutto un cambiamento sociale, economico e culturale.

È proprio in quest'ottica che le comunità energetiche svolgono un ruolo fondamentale trasformando il cittadino da semplice consumatore di energia a *prosumer*, questo termine mutuato dall'inglese indica infatti un utente che non si limita al ruolo passivo di consumatore (consumer), ma partecipa attivamente al processo di produzione (producer). In campo energetico il prosumer è quindi colui che possiede un impianto di produzione di energia rinnovabile, della quale ne consuma una parte. L'energia restante può essere accumulata per poi essere utilizzata dall'utente successivamente, scambiata con i consumatori di energia vicini al prosumer oppure può essere immessa nella rete elettrica.

Quando un gruppo di utenti decide di cooperare nella produzione, nel consumo e nella gestione dell'energia elettrica, questo prende il nome di comunità energetica. Il concetto di comunità energetica racchiude una pluralità di organizzazioni che possono differire per topologia ed obiettivi: una comunità energetica può coinvolgere un condominio, un vicinato oppure un edificio (uffici/ centri commerciali/ etc.) e può avere come scopo quello di fornire ai propri membri energia a prezzi accessibili oppure può dare

priorità al guadagno economico vendendo l'energia prodotta come farebbe una società energetica tradizionale.

2.2. GECO

GECO (Green Energy Community) è un progetto di ENEA che si pone come obiettivi: la riduzione del consumo e dello spreco di materie prime, la promozione di una società equa che contrasti la povertà energetica, la transizione verso metodi di produzione e consumo di energia che siano ecologicamente sostenibili, favorire l'autonomia degli individui ed alimentare uno spirito comunitario basato sulla partecipazione attiva nella gestione delle risorse comuni.

Per raggiungere questi obiettivi GECO sta costituendo una comunità energetica, basata su energie pulite e rinnovabili, nel distretto di Pilastro Roveri a Bologna.

2.3. POLIS-EYE

POLIS-EYE (POLIcy Support systEm for smart citY data governancE) è un progetto di ricerca che intende sviluppare un sistema di supporto decisionale che mira ad ottimizzare l'utilizzo della smart city e a migliorare la vita cittadina.

POLIS-EYE si basa sulla raccolta e gestione di dati provenienti da diverse sorgenti (Open Data, IoT, applicazioni) al fine di fornire servizi di business analytics ed in particolare si occupa di:

- Analitica descrittiva: accesso ai dati raccolti in formato immediatamente interpretabile

- Analitica predittiva: modelli di apprendimento automatico e pattern matching per effettuare previsioni
- Analitica prescrittiva: modelli di decisione in grado di fornire raccomandazioni in contesti specifici.

Questo sistema si basa sull'integrazione di tecnologie per la gestione di big data, deep learning e tecniche di ottimizzazione combinatoria.

I servizi forniti saranno a supporto di pubbliche amministrazioni, cittadini, imprese del settore turistico ed alle attività di consulenza ed esse collegate.

2.4. SELF-USER

SELF-USER è un progetto di ENEA che ha lo scopo di realizzare un impianto innovativo che favorisca la transizione energetica verso il consumo di energie rinnovabili. Così come GECO, anche SELF-USER si colloca nell'ambito delle comunità energetiche, in particolare intende realizzare un sistema che colleghi ad un unico POD (Point Of Delivery) un intero condominio che si trova nel comune di Reggio Emilia ed è costituito da 48 abitazioni. Il condominio comprende utenze elettriche del condominio e dei singoli inquilini, pannelli fotovoltaici, sistemi di accumulo elettrico, centraline meteo e colonnine di ricarica per auto elettriche.

Questo sistema analizza:

- Impianto fotovoltaico: potenza nominale, potenza istantanea e variazione dell'irraggiamento solare in base alle condizioni ambientali rilevate dalla stazione meteo
- Utenze: numero, classificazione (private e condominiali), valori di consumo medio delle utenze, profilo di carico e relativa variazione durante l'anno

- Sistema di accumulo: potenza e capacità

Attraverso questi dati l'impianto è in grado di massimizzare l'autoconsumo del condominio e si stima che la percentuale di energia prodotta sarà pari al 70%-80% dell'energia totale richiesta.

3. Specifiche fornite da ENEA

Per poter gestire l'enorme quantità di dati necessari al funzionamento dei servizi sopra descritti, ENEA ha in atto una collaborazione con il gruppo di ricerca "DBGroup" del dipartimento di ingegneria "Enzo Ferrari" di Unimore, che include la progettazione e la conseguente realizzazione di un database che sia ottimizzato, in particolare, per la gestione delle serie temporali.

Le specifiche erano contenute in un documento tecnico - realizzato nell'ambito dell'attività di GECO – che descrive e documenta una prima versione del database atto a gestire i dati sia per SELF-USER che per GECO. La documentazione fornita includeva sia lo schema relazionale del database sia informazioni riguardanti le principali tabelle del database: il ruolo che queste svolgono nello schema e il formato dei dati che le devono popolare. Per semplicità di rappresentazione, lo schema viene riportato suddiviso in vari sottoschemi inclusi nelle seguenti figure:

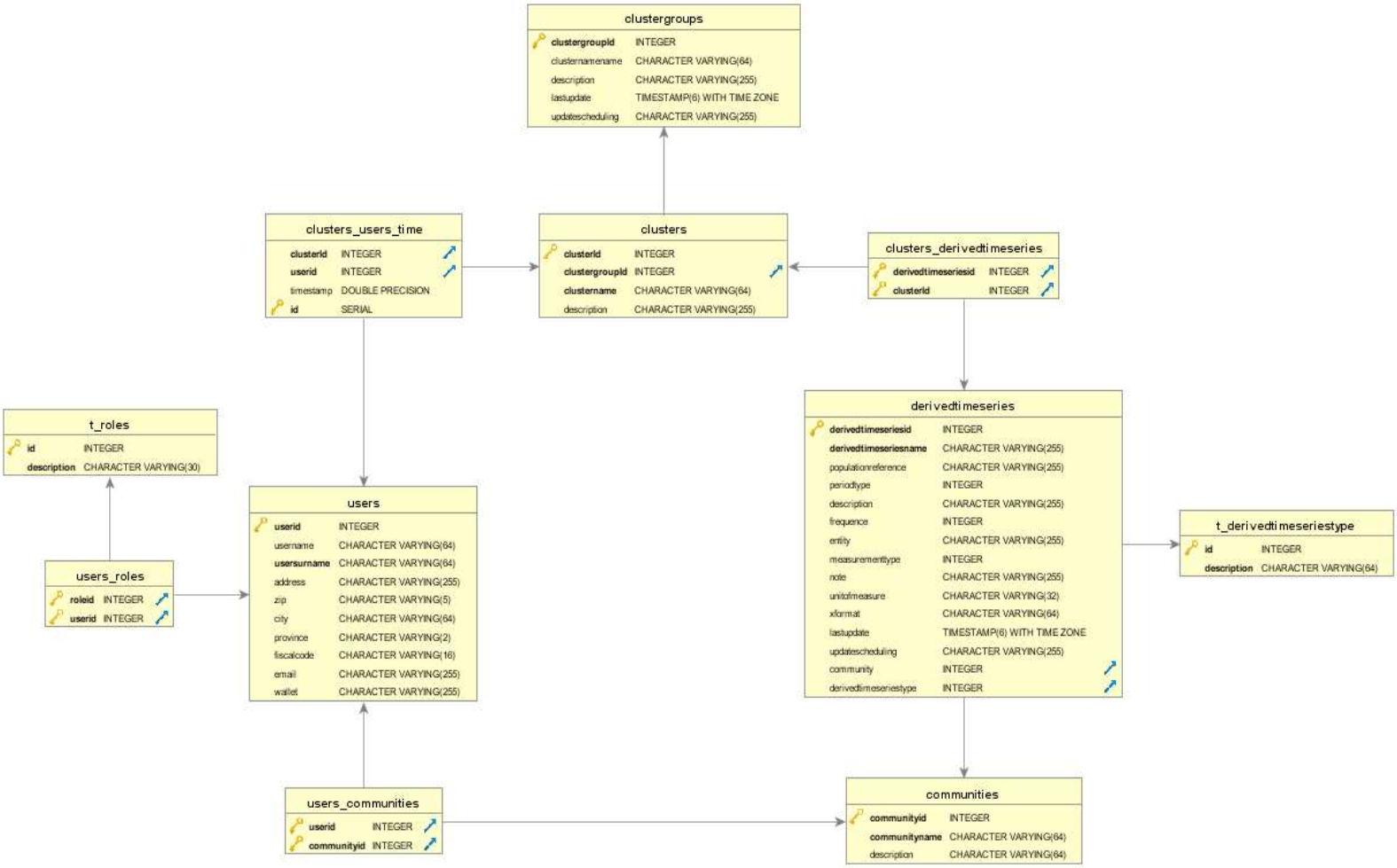

Figura 1 - Schema 1 : Users,communities,clusters

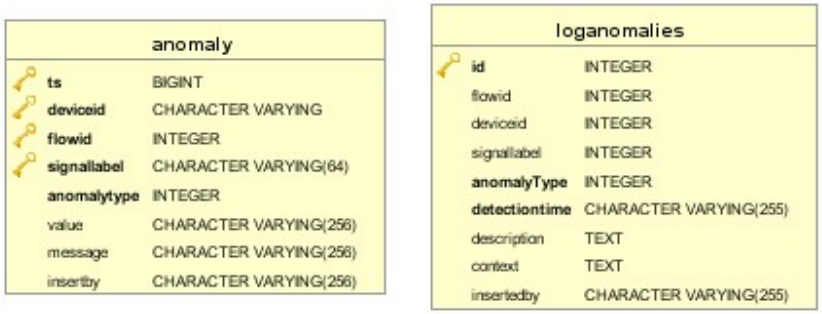

Figura 2 – Schema 1 : Anomalie

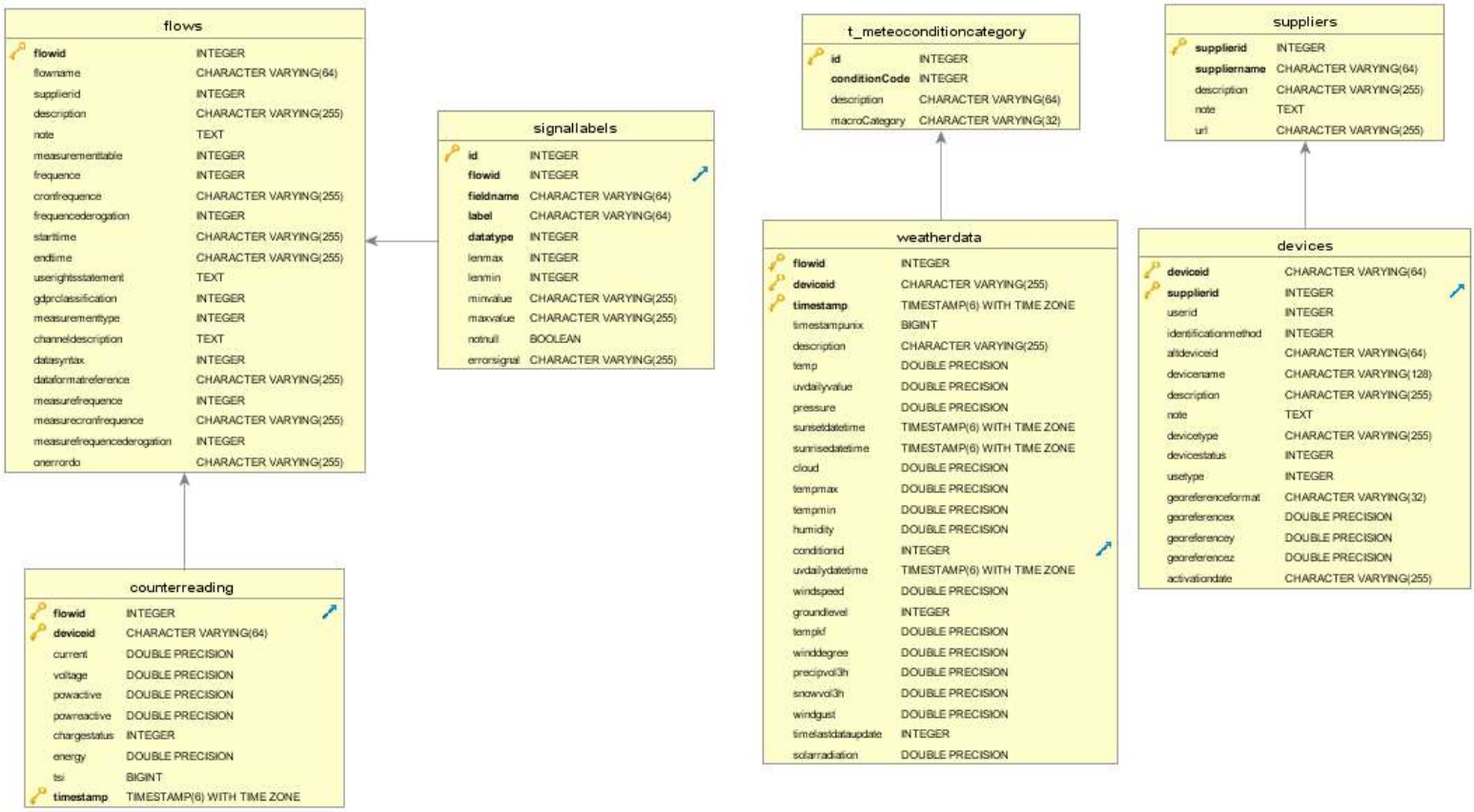

Figura 3- Schema 1: flows, counterreading, wheaterdata, devices

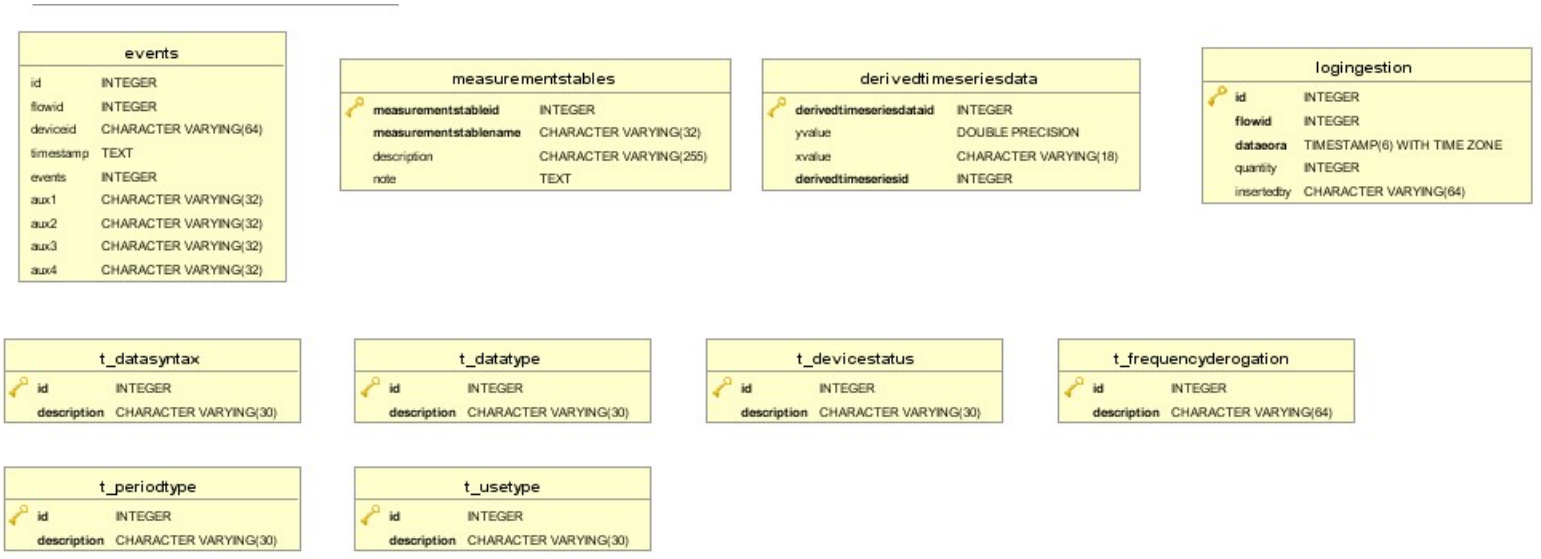

Figura 4 – Schema1: events, derivedtimeseriesdata, logingestion, entità metadato

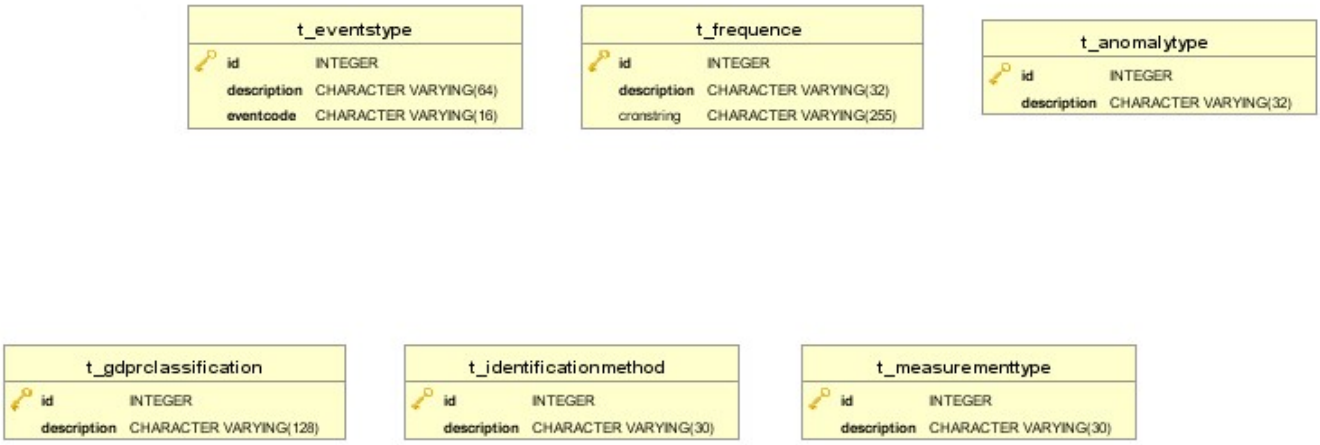

Figura 5 – Schema1 : entità metadato

Dallo schema sopra riportato e dalla documentazione ad esso correlata emerge che l'entità di maggior rilievo del modello è Flows. Un flusso di dati è costituito da dispositivi omogenei ed è associato ad un solo fornitore che è responsabile della sua attività. Non è tuttavia escluso che un medesimo dispositivo ed un medesimo fornitore attivino più flussi con qualità dei dati diversa.

Flows è l'entità responsabile di raccogliere tutti i dati relativi al fornitore, alla comunità energetica, ai consumi rilevati dal contare, ai dati di produzione e a quelli di immagazzinamento energetico; il flusso raccoglie poi una serie di informazioni relative a: tipo di misurazione, frequenza con la quale i dati vengono raccolti, eventuali anomalie, sintassi dei dati e classificazione del regolamento sulla protezione dei dati (GDPR).

Esiste poi un secondo tipo di flusso, sempre rappresentato dalla medesima entità, che raccoglie informazioni relative ai dati meteo raccolti dalla centralina metereologica, questi dati devono essere confrontati insieme ai dati di produzione di energia da fonti rinnovabili al fine di comprendere al meglio in che modo le condizioni metereologiche influiscono sulla produzione energetica e di massimizzare l'autoconsumo.

Flows è quindi l'entità responsabile di raccogliere tutti i dati di maggior rilievo presenti nel database. È su questa entità che si eseguiranno la maggior parte delle operazioni di consultazione ed aggregazione dei dati.

I dati raccolti infatti, non vengono analizzati singolarmente ma viene eseguito un lavoro di aggregazione al fine di valutarli nel loro complesso. L'aggregazione dei dati viene fatta inizialmente sulle medie di ogni singolo POD (Point of Delivery, identifica univocamente il contatore dell'energia) ed arriva fino ad un quadro complessivo della comunità. Ad esempio, il consumo di un singolo POD diventa prima un profilo di consumo, poi un cluster di consumi di più utenti (es. una palazzina) e poi di quartiere o di comunità.

In questo modo è possibile:

- Identificare indicatori di prestazione per il singolo utente o per la comunità
- Visualizzare valori di sintesi per l'utente, l'edificio e la comunità
- Identificare la flessibilità potenziale dei carichi
- Aggregare profili di consumatori per la loro gestione (creazione di cluster di comportamento, analisi di fattibilità della comunità o di un investimento).

A patire dal flusso di dati si costruiscono poi le **serie temporali derivate**: tabelle che possono essere finalizzate a risolvere diversi problemi:

- Rappresentare un dato sintetico, frutto di una preelaborazione di un set più ampio di dati.
  Un esempio tipico sono i profili medi di carico giornaliero calcolati sulle medie di un periodo per un cluster di utenti (frutto di più elaborazioni dei dati a partire dalle letture dei consumi).

- Rappresentare un dato ‘ripulito’ e ‘rilavorato’ a seguito di una serie di controlli che vengono fatti per colmare buchi, eliminare anomalie, etc.

3.1. Tipologia dei dati

I dati raccolti per il funzionamento della local energy community riguardano la descrizione delle utenze energetiche e i relativi parametri di consumo, produzione e immagazzinamento di energia. Questi sono di due tipologie:

- **Dati statici**: raccolti una sola volta (ad esempio al momento dell’iscrizione) e conservati in tabelle di un database relazionale. Questi dati riguardano persone, edifici/appartamenti, e le caratteristiche dei dispositivi di produzione, consumo e immagazzinamento di energia.
- **Dati dinamici**: dati raccolti da sensori in continuo o con periodicità differenti a seconda dei casi o recuperati da sistemi esterni alla local energy community: ad esempio aggregatori esterni, fornitori di energia o del servizio elettrico.

L’insieme dei **dati statici** contiene:

- Anagrafica degli utenti (nome, mail, etc.)
- Tipologia di utenza e caratterizzazione (domestica, aziendale, condominiale)
- Georeferenziazione dell’edificio o degli edifici sottoposti a misurazione/monitoraggio
- Caratteristiche dell’involucro e degli edifici (infissi, dispositivi tecnici, etc.)
- Tipologia di tecnologia per la climatizzazione invernale o estiva
- Definizione dei carichi elettrici specificando potenza installata, differibilità, attenuabilità, classe energetica

- Caratterizzazione delle sorgenti di energia rinnovabile
- Potenza dell'accumulo

L'insieme dei **dati dinamici** contiene:

- Dati di consumo (con differente intervallo di tempo, tipicamente il quarto d'ora)
- Dati di produzione (con differente intervallo di tempo, tipicamente il quarto d'ora)
- Andamento dell'accumulo (carica/scarica con differente intervallo di tempo, tipicamente il quarto d'ora)

3.2. Metodologia di raccolta dei dati

I **dati statici** vengono raccolti secondo le seguenti modalità:

- Iscrizione alla LEC attraverso differenti sistemi, sia digitali che manuali (sito web, app, documenti e moduli cartacei)
- Questionario per la raccolta delle informazioni di caratterizzazione energetica

  Questi dati vengono raccolti quindi su diversi tipo di supporti digitali e vanno successivamente importati (se non sono già presenti) e conservati in DB relazionali.

I **dati dinamici** vengono invece raccolti mediante:

- Sensori collegati ad un aggregatore (smart-box) studiato da ENEA. Lo smart-box invia dati in formato standard ad un sistema di raccolta. I dati possono essere, se necessario, pre-processati dallo smart-box, ad esempio per calcolare le medie su intervalli di tempo ampi
- Sensori collegati ad un contatore smart. I nuovi contatori di energia elettrica sono infatti in grado di dialogare con un dispositivo utente

che raccoglie informazioni e dati inviate alla rete e le manda ad un aggregatore esterno ad ENEA. I dati, in questo caso, devono essere recuperati presso l'aggregatore. In questo caso il formato dei dati è non conosciuto a priori ma dipende dal produttore dei dati stessi.

- Sistemi del distributore che raccoglie i dati di tutti i contatori e li mette a disposizione dei venditori di energia. Anche in questo caso, il formato dei dati non è conosciuto a priori.

3.3. Utilizzo dei dati

I **dati statici** vengono utilizzati per:

- Identificare gli utenti e le loro caratteristiche
- Individuare possibilità di ottimizzare il consumo energetico (cappotto dove non c'è, infissi, etc.)

I **dati dinamici** vengono utilizzati per definire dei differenti profili di produzione/consumo dei singoli utenti che possono essere utilizzati per:

- Benchmarking interno alla comunità
- Identificazione di profili di produzione/consumo compatibili ed ottimizzabili
- Identificazione di profili di flessibilità utilizzabile e vendibile
- Identificazione di potenziali miglioramenti nei comportamenti dell'utente o nel profilo energetico
- Accounting dei consumi e produzione, anche per la ripartizione dei compensi
- Identificazione dei comportamenti virtuosi e loro remunerazione (in token) all'interno della comunità.

I dati dinamici non vengono analizzati singolarmente poiché l'enorme quantità di dati raccolti renderebbe impossibile la loro analisi senza aver prima eseguito un lavoro di aggregazione. Le operazioni sopra elencate vengono quindi fatte a partire da cluster di dati che, a seconda del caso d'uso, possono essere relativi al singolo utente, all'edificio o all'intera comunità.

4. Il Database

Nel capitolo seguente è analizzato nel dettaglio il database: per prima cosa vengono analizzate e descritte le entità che fanno parte dello schema e successivamente le principali relazioni esistenti tra queste, verranno mostrate volta per volta solo le porzioni di schema E/R in analisi. Infine, sono riportati per intero lo schema E/R relativo al database analizzato per ENEA e la conseguente traduzione in relazionale al fine di fornirne una visione di insieme.

4.1. Entità

Nel seguente sotto-capitolo vengono analizzate nel dettaglio le entità che sono state inserite nel database e viene spiegato il loro ruolo all'interno del modello.

A causa dell'elevato numero di tabelle presenti nel database e dell'elevato numero di relazioni tra queste, sarebbe risultato difficoltoso da leggere uno schema aderente alla notazione E/R convenzionale: con il nome dell'entità al centro e gli attributi disposti a raggiera intorno ad esso. Per questa ragione si è scelto di rappresentare le entità mediante l'uso di una tabella in cui il nome dell'entità è riportato in alto e di seguito sono indicati i suoi attributi; gli identificatori interni sono stati marcati con la sigla **PK** (Primary Key) mentre gli identificatori esterni sono stati marcati con la sigla **FK** (Foreign Key); si è scelto di non indicare all'interno dell'entità le foreign key che non prendono parte all'identificazione dell'entità (come nel modello E/R tradizionale).

4.1.1. Anagrafica

Come detto in precedenza, i dati raccolti sono classificati come dati statici e dinamici, i primi non si riferiscono direttamente alla produzione e al consumo di energia, riguardano invece le informazioni anagrafiche sulle persone e sulla strumentazione che prendono parte al processo di produzione e consumo di energia. Questi dati sono raccolti una sola volta e salvati in apposite tabelle del database, le entità che svolgono questo compito sono:

- Communities: sono le comunità energetiche che sono gestite dalla piattaforma
- Users: sono gli utenti della piattaforma. Un utente può appartenere solamente ad una comunità
- Suppliers: sono tutti i servizi, aziende o in generale fornitori che forniscono dati attraverso dei dispositivi o altri meccanismi (ad esempio da siti web)
- Device: sono tutti i dispositivi da cui arrivano o con cui vengono fatte le misurazioni dei dati

Queste entità sono tutte identificate mediante un identificatore interno numerico, fa eccezione l'entità Device, la quale ha un identificatore misto formato da una stringa, contenete il codice POD, e

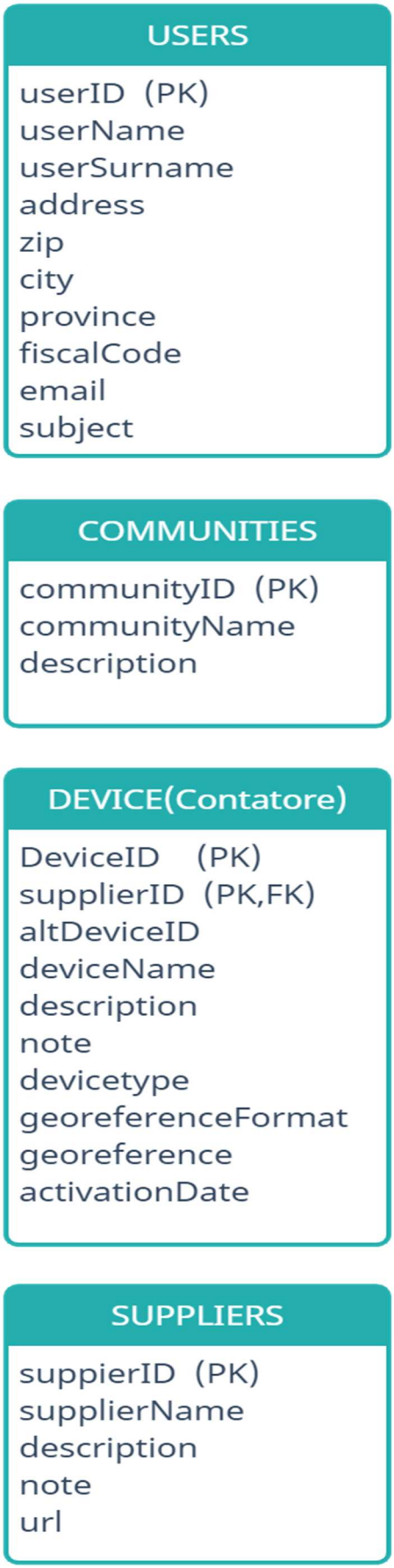

Figura 6 – Tabelle 1-5 : Anagrafica

dall’identificatore di Supplier, questo è dovuto al fatto che un contatore è relativo ad uno e un solo fornitore.

Gli altri attributi di queste entità modellano i dati statici che vengono forniti all’ente al momento della registrazione.

4.1.2. Letture e consumi

Come si è detto, Flows è l'entità principale del modello sulla quale vengono fatte la maggior parte delle operazioni di consultazione, tutti i dati raccolti dai dispositivi vengono organizzati in flussi, successivamente si costruiscono serie temporali derivate e si raggruppano i flussi in cluster. Un flusso è identificato internamente tramite un inter (flowID); contiene informazioni relative alla frequenza di invio e rilevazione dei dati, a cosa fare in caso di errore, ai momenti di inizio e fine di rilevazione dei dati, al canale di invio dei dati e alla relativa sintassi.

Un flusso è costituito da più dispositivi omogenei ed è associato ad un solo fornitore che è responsabile della sua attività. Inoltre, esistono due tipi di flusso: quelli relativi ai dati di consumo, produzione e immagazzinamento di energia elettrica e quelli relativi ai dati meteo. Questi dati provengono rispettivamente dalle tabelle Counter_reading e Weather_data, entrambe le tabelle sono identificate dal dispositivo che ha generato i dati, dal flusso di appartenenza e da un timestamp che indica il momento di rilevazione dei dati. Il compito di indicare la tabella di destinazione del flusso (Counter_reading o Weather_data) è svolta dall'entità Measurements_tables la quale è in

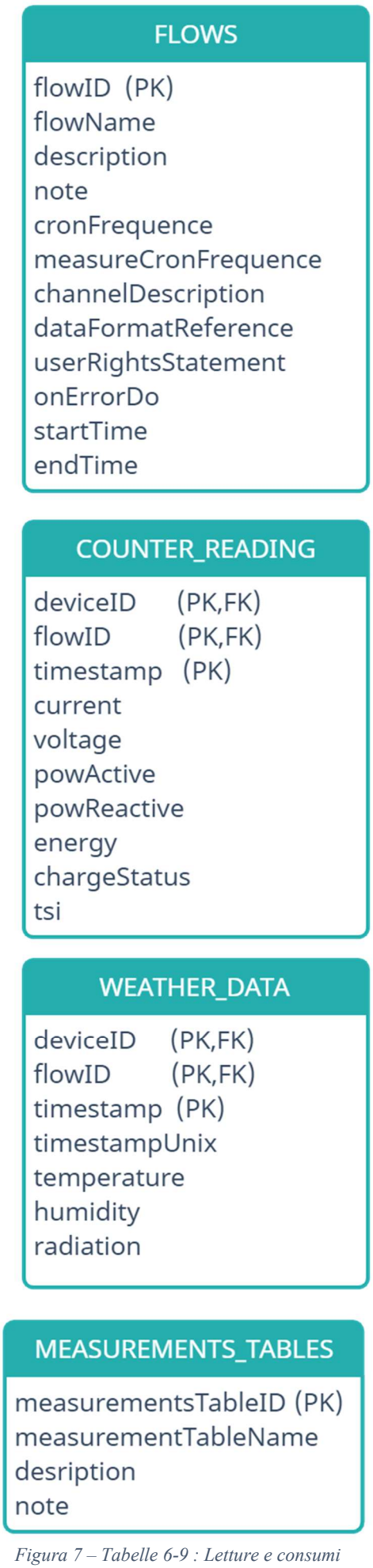

Figura 7 – Tabelle 6-9 : Letture e consumi

relazione uno a molti con Flows e indica in quali tabelle andare a cercare i dati del flusso.

4.1.3. Tabelle a supporto del controllo di errori

Dato che in ogni sistema ci sono diverse cose che possono andare male e causare errori, è fondamentale dotarsi di un sistema che sia quantomeno in grado di rilevare gli errori o ancor meglio di correggerli.

In questo modello la rilevazione degli errori è svolta dalle tabelle Anomalies, Signal_labels e Log_ingestion.

L'entità Anomalies è responsabile di memorizzare tutti gli errori che possono verificarsi all'interno del database, contiene un riferimento al flusso in cui si è verificata l'anomalia, l'orario in cui è stata rilevata, una descrizione testuale dell'anomalia e del contesto in cui questa è avvenuta e un riferimento al sistema o alla persona che ha rilevato l'errore.

L'entità Signal_labels invece svolge il compito di fornire supporto al rilevatore di errori, in questa tabella vengono memorizzate tutte le informazioni relative ai vari campi di un flusso di dati, si salvano informazioni riguardanti i valori ammessi per ogni campo di un flusso di dati, la possibilità che il campo in questione sia o meno Null, il valore di errore per quell'attributo e il formato in cui questo viene memorizzato.

Figura 8 – Tabelle 10-12 : Tabelle a supporto del controllo di errori

Il controllore di errori dovrà quindi andare a leggere ogni riga di un flusso di dati (comprese quelle di Counter_reading e Weather_data), verificare che il valore di ogni colonna sia conforme a quanto indicato nella relativa riga di Signal_labels e in caso contrario dovrà inserire una nuova riga nella tabella Anomalies relativa all'errore individuato.

Log_ingestion ha invece il compito di memorizzare tutti gli errori che non sono relativi ai dati memorizzati ma all'invio degli stessi, dato che, soprattutto in caso di invii multipli, può accadere che alcuni dati vadano persi sul canale di trasmissione dei dati. Come per Anomalies, questa entità contiene un riferimento al flusso in cui si è verificato l'errore e all'orario in cui questo è stato rilevato oltre che un'indicazione di chi ha inserito l'errore nel database; infine è presente un attributo 'quantity' che indica il numero di record o file che sono stati ricevuti correttamente, questo campo va confrontato con quanto indicato nel descrittore del flusso per verificare quante informazioni sono andate perdute.

4.1.4. Serie temporali

Una serie temporale è un insieme di dati ordinato rispetto al tempo che descrive in che modo il sistema analizzato evolve. La frequenza con cui i dati di una serie temporale vengono raccolti dipende dal campo di utilizzo e può essere molto variabile (ogni millisecondo/ogni ora/ogni mese), inoltre i dati possono essere raccolti sia in maniera regolare che irregolare (ad esempio si possono raccogliere solamente quando accade un determinato evento). Le serie temporali, infatti, sono utilizzate in numerosi campi, per esempio: monitoraggio delle prestazioni di un sistema informatico, sistemi di trading finanziario, internet of things, monitoraggio delle condizioni di un ambiente, etc.

Una caratteristica fondamentale delle serie temporali è quindi quella di avere necessariamente un campo "timestamp" che indica il momento nel quale il dato in questione è stato campionato. La presenza di questo campo, tuttavia, non è una condizione sufficiente a classificare un insieme di dati come serie temporale, la peculiarità delle serie temporali è che i dati vengono aggiunti molto frequentemente mentre non vengono mai rimossi o modificati, inoltre i dati vengono inseriti alla fine (*append-only*) in quanto quando ricevo un nuovo dato da inserire questo sarà più recente rispetto a tutti quelli già presenti.

Il fatto che i dati di una serie temporale vengano manipolati in maniera così differente rispetto ai dati ordinari richiede che questi vengano gestiti tramite un DBMS che sia ottimizzato per la loro manipolazione. Per questo progetto è stato scelto di utilizzare TimescaleDB: un'estensione di PostgreSQL pensata appositamente per la gestione delle serie temporali.

In questo modello, i dati relativi a consumo e produzione di energia elettrica da parte degli utenti vengono campionati con cadenza quartoraria, questo significa che ogni giorno vengono prodotti 96 report per ogni dispositivo;

data la mole di dati in questione risulterebbe particolarmente difficoltoso analizzare i dati raccolti singolarmente. Per ovviare a questo problema si è scelto di ricorrere all'utilizzo delle serie temporali, le quali hanno il compito di fornire un'informazione risultante da alcune elaborazioni dei dati raccolti.

Le serie temporali vengono rappresentate nel database dalle entità Derived_time_series e Derived_time_series_data-

Derived_time_series ha il compito di rappresentare la serie temporale, tra i suoi attributi troviamo:

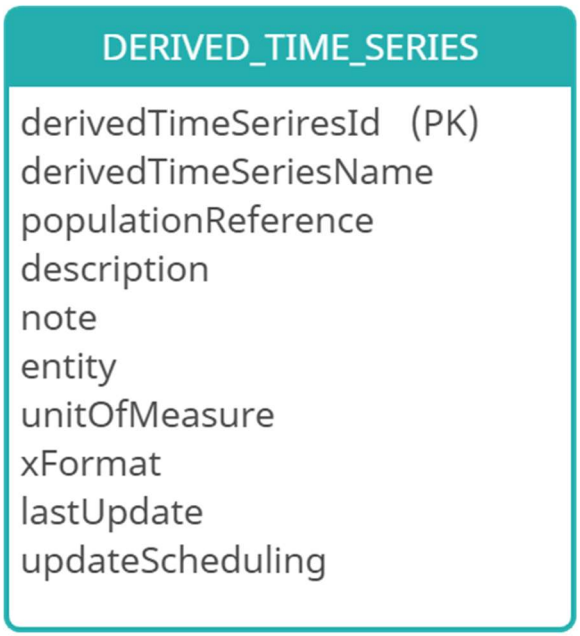

Figura 9 – Tabelle 13-14 : Serie temporali

- una descrizione della popolazione a cui la serie si riferisce: se si tratta di un'utenza singola o di un cluster
- il tipo di dato rappresentato (entità metadato, si veda in seguito)
- l'arco temporale coperto dalla serie (entità metadato)
- l'unità di misura dei dati
- il formato in cui l'intervallo temporale è rappresentato: potremmo infatti trovare un timestamp unix, un numero incrementale per distinguere i valori della serie oppure una rappresentazione testuale dell'orario (es: 11.00 - martedì)
- tipo di misura: valore istantaneo, medio o minimo/massimo sull'intervallo (entità metadato)
- un timestamp che indica l'ultima volta in cui la serie è stata popolata

- frequenza di rilevazione dei dati (entità metadato)
- un'indicazione testuale che indica se è necessario ricalcolare la tabella e la programmazione per farlo (se assente si assume di non dover ricalcolare periodicamente)

Derived_time_series_data è in relazione uno a molti con Derived_time_series e contiene invece i dati veri e propri della serie temporale, questi sono rappresentati sul piano cartesiano: ogni valore della serie è quindi identificato da una coppia di valori (x,y).

L'asse x è lasse dei tempi questo valore è rappresentato in maniera testuale e deve essere interpretato secondo quanto indicato nella tabella Derived_time_series, alla riga relativa al dato in questione e alla colonna 'xFormat'.

Sull'asse y troviamo invece il valore della grandezza rappresentata, a questo deve essere associata l'unità di misura indicata dal campo "unityOfMeasure" della relativa riga di Derived_time_series.

4.1.5. Cluster

Cluster è una parola che deriva dalla lingua inglese, letteralmente significa 'Gruppo', in ambito tecnico-scientifico questo termine indica un gruppo di *oggetti* logicamente connessi tra loro che nel complesso costituiscono un insieme organico.

Nell'ambito di questo progetto *Clusterizzare* significa individuare gruppi di misure, le quali possono riferirsi a comportamenti similari di un utente in relazione ad una entità misurata, ad esempio si possono raggruppare le misurazioni del consumo energetico di un singolo utente relative ai giorni feriali/festivi oppure le misurazioni di produzione energetica nei giorni invernali/estivi. Un secondo tipo di cluster riguarda il raggruppamento di più utenti che sono legati da comportamenti simili nel consumo/produzione di energia, dall'appartenenza alla stessa comunità energetica o dalla residenza nello stesso edificio.

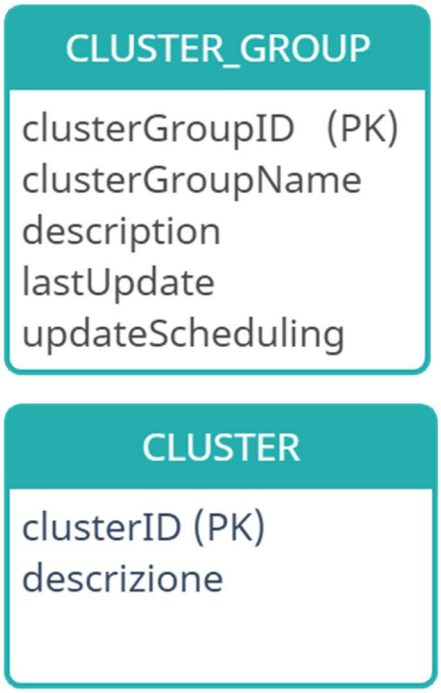

Figura 10 – Tabelle 15-16 : Cluster

Ai cluster generati possono poi essere associate delle serie temporali derivate che hanno il compito di rappresentarne il comportamento in sintesi.

In questo modello il lavoro di clusterizzazione è svolto dalle entità Cluster_group e Cluster:

Cluster_group descrive l'operazione di raggruppamento che ha generato il cluster, tra i suoi attributi troviamo una descrizione testuale dell'operazione di clusterizzazione, data e ora dell'ultimo aggiornamento del cluster e un'indicazione testuale che indica se è necessario ricalcolare la tabella e la programmazione per farlo (se assente si assume di non dover ricalcolare periodicamente).

Cluster è invece l’entità responsabile di identificare il singolo cluster, questa è in relazione con utenti, comunità flussi e dispositivi che lo compongono e con le serie temporali che ne descrivono il comportamento.

4.2. Reverse Engineering

In questo sotto-capitolo verrà effettuato il reverse engineering dello schema relazionale descritto in precedenza per ricavarne il corrispondente schema E/R. A tale scopo sono riportate e analizzate le principali relazioni presenti nello schema relazionale; queste sono presentate prima attraverso una descrizione testuale che ne evidenzia identificatori e foreign key, lo schema viene poi analizzato e infine sono riportati le relative porzioni di schemi E/R; come ultimo passaggio si riporta la traduzione in relazionale di tali schemi E/R. Al fine di fornire una rappresentazione più chiara, si è scelto di riportare in questi schemi scheletro solamente le chiavi primarie mentre nelle relative traduzioni sono indicate solo le foreign key riferite alle tabelle in analisi.

4.2.1. Anagrafica

- Users
  - userid
- Roles
  - roleid
- Community
  - communityid
- Suppliers
  - supplierid
- Flows
  - flowid
  - supplierid → Suppliers *-- riferimento al fornitore*
- users_roles
  - userid → Users *--riferimento all'utente*
  - roleid → Roles *--riferimento al ruolo*
- flows_communities
  - flowid → Flows *--riferimento al flusso*
  - communityid → Community *--riferimento alla comunità*
- suppliers_communities
  - communityid → Community *--riferimento alla comunità*

supplierid → Suppliers *--riferimento al fornitore*

- users_communities

  communityid → Community *--riferimento alla comunità*

  userid → Users *--riferimento all'utente*

Da questa porzione di schema emerge che ogni utente ha un ruolo all'interno del modello ma può averne anche più di uno (ad esempio un utente potrebbe essere contemporaneamente: Registered, Community administrator e Token administrator); inoltre ogni utente può far parte di più di una comunità contemporaneamente, all'interno di ognuna delle quali gli viene assegnato un portafoglio.

Un flusso dati è associato ad un solo fornitore ma può essere relativo a più di una comunità (questo può accadere ad esempio se più comunità hanno uno stesso fornitore di energia o se condividono la centralina meteo) e, ovviamente, una comunità può attivare più di un flusso dati.

Infine, una comunità energetica può avere più di un fornitore, con il decreto Milleproroghe del 2020, infatti, a chi partecipa ad una comunità energetica sono riconosciuti tutti i diritti come cliente finale, compreso quello di scegliere il proprio fornitore e, ovviamente, un supplier può essere il fornitore di energia di comunità differenti.

In questa porzione di schema a prima vista sembra essere presente un ciclo: *Communities →Suppliers_communities →Suppliers →Associato_a →Flows →Flows_communities →Communities*; tuttavia, un Supplier può essere fornitore di energia per più di una community anche senza avere alcun flusso attivo, per questa ragione le tabelle suppliers_communities e flows_communities sono entrambe necessarie e non è possibile considerare una delle due derivata (cioè una view) dell'altra.

D’altra parte se c’è un flusso attivo di un supplier verso una community, il supplier dovrebbe essere fornitore di tale community: questa regola che non risulta esprimibile nello schema ER, andrebbe controllata tramite appositi meccanismi, quali i *trigger*.

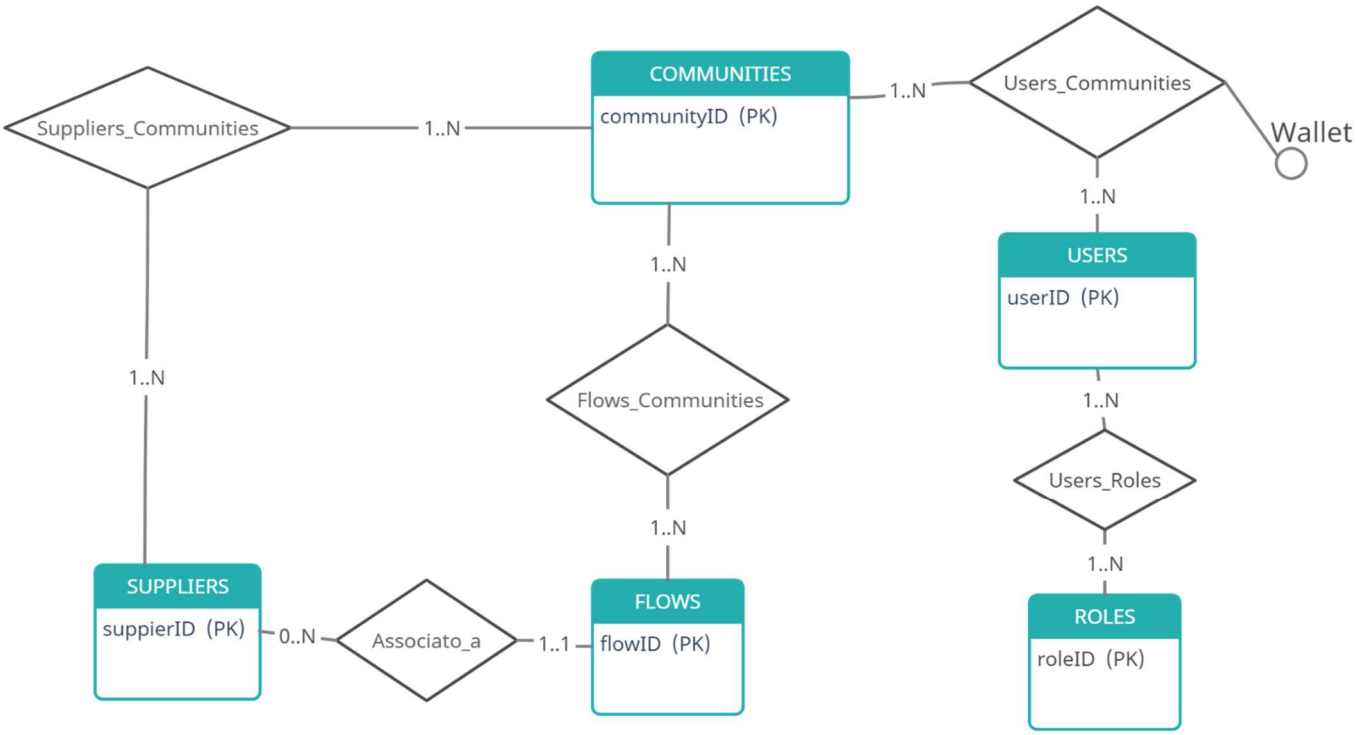

Figura 11 – Schema 2: Anagrafica

Riportiamo di seguito il relativo schema relazionale, ottenuto tramite la classica progettazione logica relazionale, limitatamente alle chiavi e alle foreign-key , ovvero limitatamente alle informazioni riportate nello schema E/R. Lo scopo di tale schema relazionale è duplice:

1. serve per controllare il processo di reverse-engineering effettuato: infatti se non sono state apportate modifiche allo schema E/R si dovrebbe ottenere lo stesso schema relazionale iniziale
2. serve soprattutto per evidenziare come eventuali modifiche/miglioramenti riportati nello schema E/R si riflettano nel corrispondente schema relazionale.

Lo schema relazionale completo con tutti gli attributi verrà riportato poi successivamente.

Suppliers (supplierid)

Flows (flowid, supplierid)

FK: supplierid **REFERENCES** Suppliers

Users (userid)

Roles (id)

Users_Roles (userid, roleid)

FK: userid **REFERENCES** Users

FK: roleid **REFERENCES** Roles

Communities (communityid)

Users_Communities (communityid, userid)

FK: communityid **REFERENCES** Communities

FK: userid **REFERNCES** Users

Flows_Communities (communityid, flowid)

FK: communityid **REFERNCES** Communities

FK: flowid **REFERENCES** Flows

Suppliers_Communities (communityid, supplierid)

FK: communityid **REFERNCES** Communities

FK: supplierid **REFERENCES** Suppliers

4.2.2. Letture e consumi

- Device
  - deviceID
  - supplierID → Suppliers -- *riferimento al fornitore*
- Messurements_table
  - measuremenTstableID
- Flows
  - flowid
  - measurementsTableID →Measurements_Table
  -- *riferimento al tabella delle misure*
- Counter_reading
  - timestamp
  - deviceid → Device -- *riferimento al dispositivo*
  - flowID → Flows -- *riferimento al flusso*
- Weather_data
  - timestamp
  - deviceid → Device -- *riferimento al dispositivo*
  - flowID → Flows -- *riferimento al flusso*

In questa sezione del database sono presenti i due tipi di flusso dati che costituiscono il modello: dati relativi ai consumi (Counter_reading) e dati meteo (Weather_data).

Un dispositivo può essere o un contatore o una centralina meteo, quando si esegue una lettura dei dati raccolti dal dispositivo questo crea una nuova riga nella tabella Counter_reading o Weather_data (a seconda del tipo di dispositivo) e attiva un flusso dati; la tabella Measurements_table è responsabile di indicare a Flows se questo è relativo a consumi energetici o a dati meteo.

Flows →Relativo_a3 →Counter_reading →Lettura →Device →Lettura2 →W eather_data →Relativo_a4 → Flows

Potrebbe sembrare un ciclo, in realtà data la doppia natura di Flows e Device, può essere attivo solo uno tra i due rami:

1. *Flows →Relativo_a3 →Counter_reading →Lettura →Device*
2. *Flows →Relativo_a4 →Weather_data →Lettura2 →Device.*

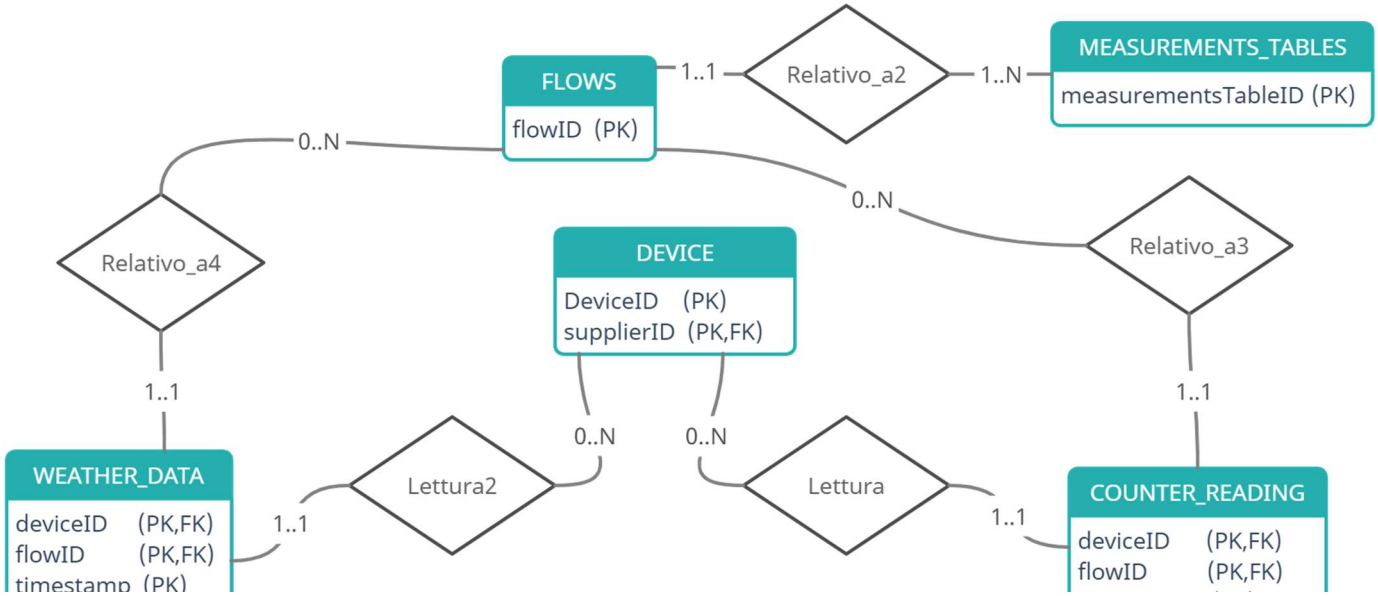

Figura 12 – Schema 3: Letture e consumi

Measurement_tables (measurementstableid)

Flows (flowid, measurementstablesid)

FK: measurementstablesid **REFERENCES** Measurements_tables

Device (deviceid, supplierid, id, altdeviceid, userid)

AK: id

AK: altdeviceid

FK: supplierid **REFERENCES** Suppliers

FK: userid **REFERENCES** Users

Counter_reading (deviceid, flowid, timestamp)

FK: deviceid **REFERENCES** Device

FK: flowid **REFERENCES** Flows

Weather_data (deviceid, flowid, timestamp)

FK: deviceid **REFERENCES** Device

FK: flowid **REFERENCES** Flows

4.2.3. Controllo degli errori

- Flows
  - flowid
- Log_ingestion
  - logingestionID
  - flowID → Flows -- *riferimento al flusso*
- Data_type
  - dataTypeID
- Anomaly_type
  - AnomalyTypeID
- Signal_Labels
  - signallabelid
  - flowID → Flows -- *riferimento al flusso*
  - dataTypeID →dataType -- *riferimento al tipo di dato*
- Anomalies
  - anomaliesID
  - anomalietipeID → Anomaly_type -- *riferimento all'anomalia*
  - signalLabelID → Signal_Labels -- *riferimento a signal_label*

Per ogni campo all'interno del flusso di dati, Signal_labels memorizza quali sono i valori ammessi; il tipo di dato a cui la label si riferisce è invece memorizzato attraverso "l'entità metadato" Data_type. Qualora un campo del flusso di dati non dovesse essere conforme a quanto specificato dalla corrispondente riga di Signal_labels, verrà aggiunta una riga alla tabella Anomalies, la quale sarà in relazione con la tabella Anomaly_type in cui sono memorizzati i tipi di dato. Se dovesse invece verificarsi un errore nell'invio o ricezione di dati, questo sarà memorizzato all'interno della tabella Log_ingestion che è direttamente in relazione con la tabella Flows.

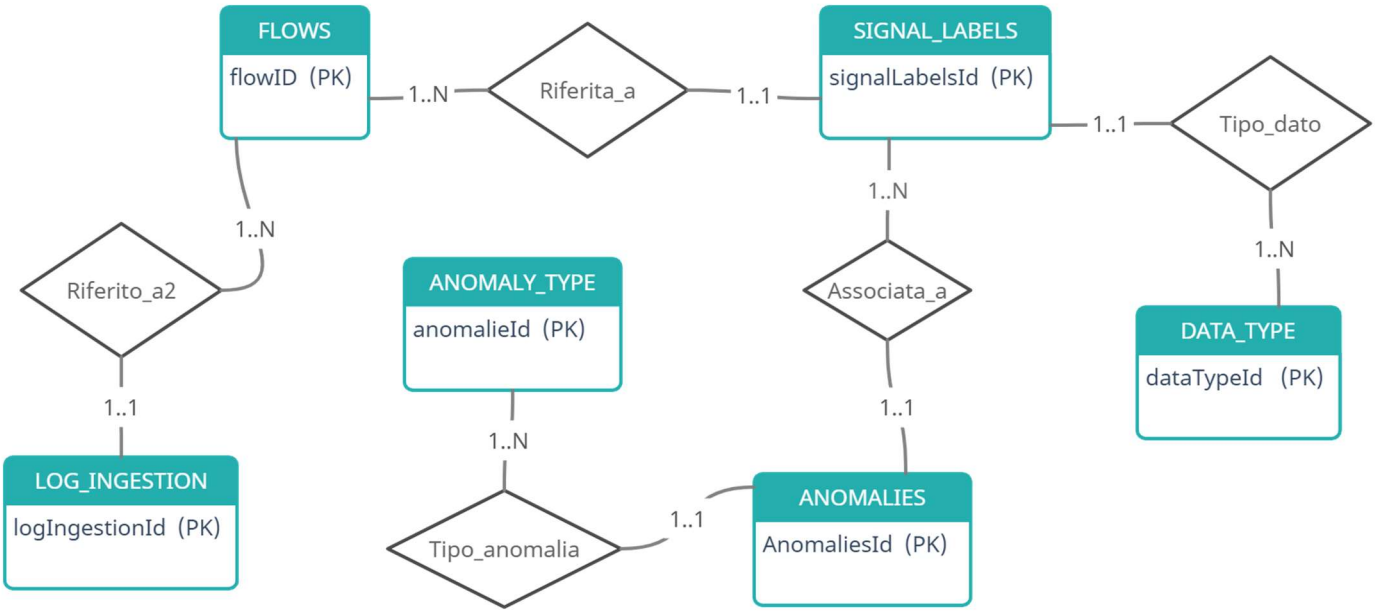

Figura 13 – Schema 4: Controllo degli errori

Flows (<u>flowid</u>)

Data_Type (<u>id</u>)

Signal_lables (<u>signallabelid</u>, flowid datatype)

FK: flowid **REFERENCES** Flows

FK: datatype **REFERENCES** Data_type

Anomaly_type (<u>id</u>)

Anomalies (<u>id</u>, flowid, signallabel, anomalytype)

FK: flowid **REFERENCES** Flows

FK: signallabel **REFERENCES** Signal_labels

FK: anomalytype **REFERENCES** Anomaly_type

Log_ingestion (<u>id</u>, flowid)

FK: flowid **REFERENCES** Flows

4.2.4. Serie temporali

- Device
  deviceID
  supplierID → Suppliers *-- riferimento al fornitore, fa parte della chiave primaria*
- Flows
  flowid
- Communities
  communityID
- Users
  userID
- Period_type
  periodTypeID
- Derived_time_series_type
  derivedTimeSeriesTypeID
- Derived_time_series
  derivedTimeSeriesID
  periodTypeID → Period_type *-- riferimento al tipo di periodo (giorno/settimana/mese/etc.)*
- derivedTimeSeriesTypeID → Derived_time_series_type *--riferimento al tipo di informazione rappresentato dalla curva*
- TimeSeries_Users
  derivedTimeSeriesID Derived_time_series *-- riferimento alla serie temporale*
  userID → Users *-- riferimento all'utente*
- TimeSeries_communities
  derivedTimeSeriesID → Derived_time_series *-- riferimento alla serie temporale*
  communityID → Communities *-- riferimento alla comunità energetica*
- TimeSeries_Device_Flows
  derivedTimeSeriesID → Derived_time_series *-- riferimento alla serie temporale*
  flowID →Flows *--riferimento al flusso*
  deviceID →Device *-- riferimento all'utenza*
- Derived_time_series_data
  derivedTimeSeriesDataID
  derivedTimeSeriesID → Derived_time_series *-- riferimento alla serie temporale*

Una serie temporale può essere relativa ad una comunità o ad una singola utenza, in entrambi casi, si memorizza all'interno della relazione il giorno a cui si riferiscono i primi dati della serie temporale.

Dato che un utente può avere più di un'utenza, si tiene traccia dei dispositivi che partecipano alla serie temporale; una serie temporale può essere riferita a più di un dispositivo (se la serie è relativa ad un comunità) e lo stesso dispositivo può far parte di più di una serie temporale a seconda dell'informazione che si vuole rappresentare; dato che un dispositivo può attivare più di un flusso dati, è importante memorizzare non solo i dispositivi ma anche i flussi di dati relativi alla serie temporale.

Si memorizzano inoltre il tipo di periodo a cui la serie si riferisce (giorno/ mese/ anno/ etc.) e il tipo di informazione che la curva rappresenta (Profilo giornaliero consumi comunità/ Curva obiettivo consumi giornalieri di utente/ etc.), queste informazioni tuttavia non sono memorizzate all'interno della tabella Derived_time_series ma tramite le entità metadato Period_type e Derived_time_serie_type. Come si è detto, i dati della serie temporale sono rappresentati come punti del piano cartesiano, ogni punto che appartiene alla curva è memorizzato nella tabella Derived_time_series_data.

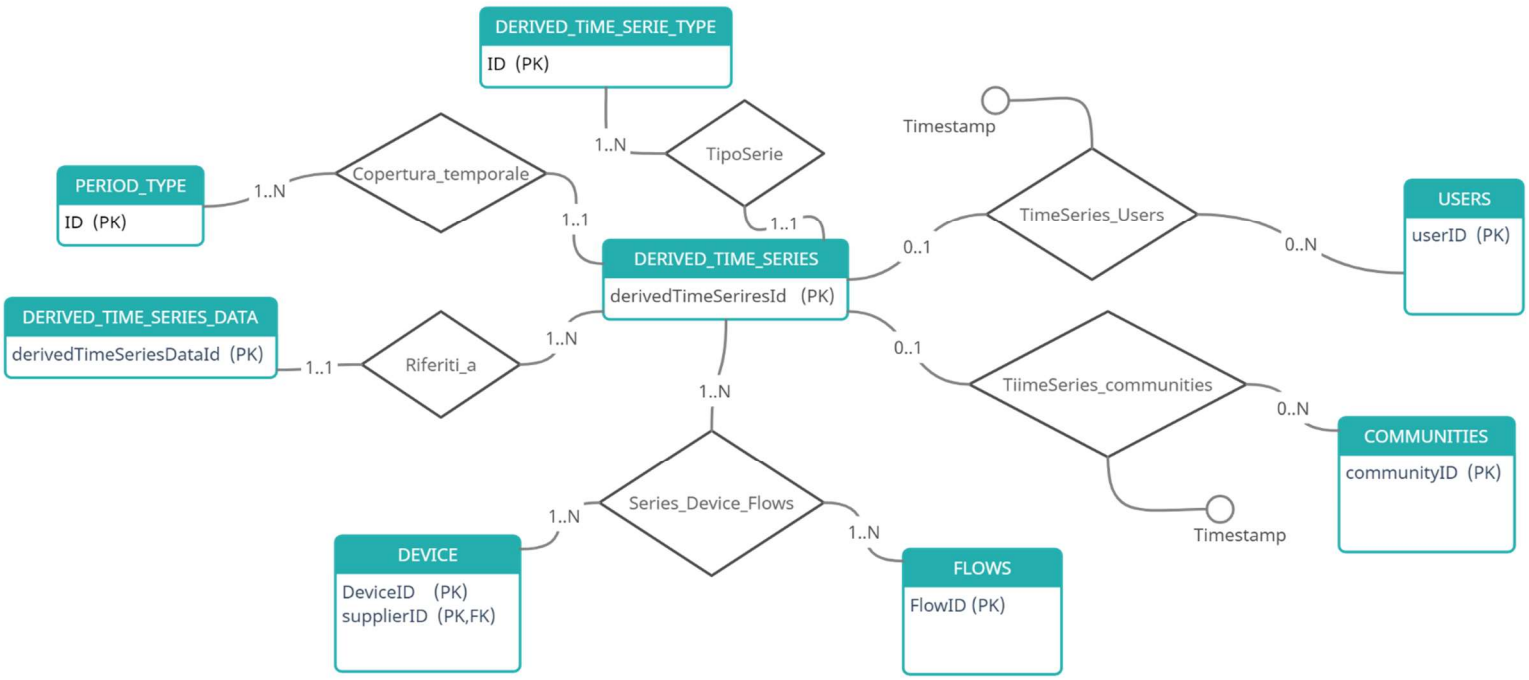

Figura 14 – Schema 5: Serie Temporali

Device (<u>deviceid, supplierid</u>, id, altdeviceid, supplierid)

AK: id

AK: altdeviceid

FK: supplierid **REFERENCES** Suppliers

FK: userid **REFERENCES** Users

Users (<u>userid</u>)

Communities (<u>communityid</u>)

Derived_time_series_type (<u>id</u>)

Period_type (<u>id</u>, description)

Derived_time_series (<u>derivedtimeseriesid</u>, derivedtimeseriestype, periodtype)

FK: derivedtimeseriestype **REFERNCES** Derived_time_series_type

FK: periodtype **REFERNCES** Period_type

Derived_time_series_data (derivedtimeseriedataid, derivedtimeserieid, xvalue, yvalue)

FK: derivedtimeserieid **REFERENCES** Derived_time_series

Derivedtimeseries_Users_time (id, derivedtimeserieid, userid, timestamp)

FK: derivedtimeseriesid **REFERNCES** Derived_time_series

FK: userid **REFERENCES** Users

Derivedtimeseries_Community_time (id, derivedtimeserieid, communityid, timestamp)

FK: derivedtimeseriesid **REFERNCES** Derived_time_series

FK: communityid **REFERENCES** Communities

Derivedtimeseries_Device_time (id, derivedtimeserieid, deviceid, flowid, timestamp)

FK: derivedtimeseriesid **REFERNCES** Derived_time_series

FK: deviceid **REFERENCES** Device

FK: flowid **REFERENCES** Flows

4.2.5. Cluster

- Device
  deviceID
  supplierID → Suppliers -- *riferimento al fornitore, fa parte della chiave primaria*
- Flows
  flowID
- Cluster_Group
  clusterGroupID
- Users
  userID
- Communities
  communityID
- Deriverd_time_series
  DerivedTimeSeriesID
  clusterID → Clusters -- *riferimento al cluster*
- Cluster
  clusterID
  communityID → Community -- *riferimento alla comunità*
  clusterGroupID → Cluster_Group -- *riferimento al gruppo di cluster*
- Cluster_Users
  clusterID → Clusters -- *riferimento al cluster*
  userID → Users -- *riferimento all'utente*
- Cluster_Device_Flow
  clusterID → Clusters -- *riferimento al cluster*
  flowID → Flows -- *riferimento al flusso*
  deviceID → Devices -- *riferimento al dispositivo*

La tabella Cluster rappresenta ogni singolo cluster ottenuto da un'operazione di raggruppamento, tale operazione è invece descritta in Cluster_Group e associata al cluster.

I cluster sono formati da più utenti, gli utenti a loro volta possono far parte di cluster differenti a seconda dei dati che voglio analizzare, per questa ragione è necessario tenere traccia di quali utenti appartengono ad ogni cluster in un'apposita tabella del database: Cluster_Users.

Dato che un utente può avere più utenze, come per le serie temporali, è necessario associare al cluster anche i dispositivi che partecipano al gruppo, inoltre, ancora una volta, dato che ogni dispositivo può attivare più di un flusso dati è necessario tenere traccia non solo dei dispositivi ma anche dei flussi dati che partecipano al cluster.

Un cluster è associato ad una solo Community, non esistono infatti operazioni di raggruppamento che eccedano la dimensione della comunità energetica.

Infine, al Cluster sono associate una o più serie temporali che lo descrivono.

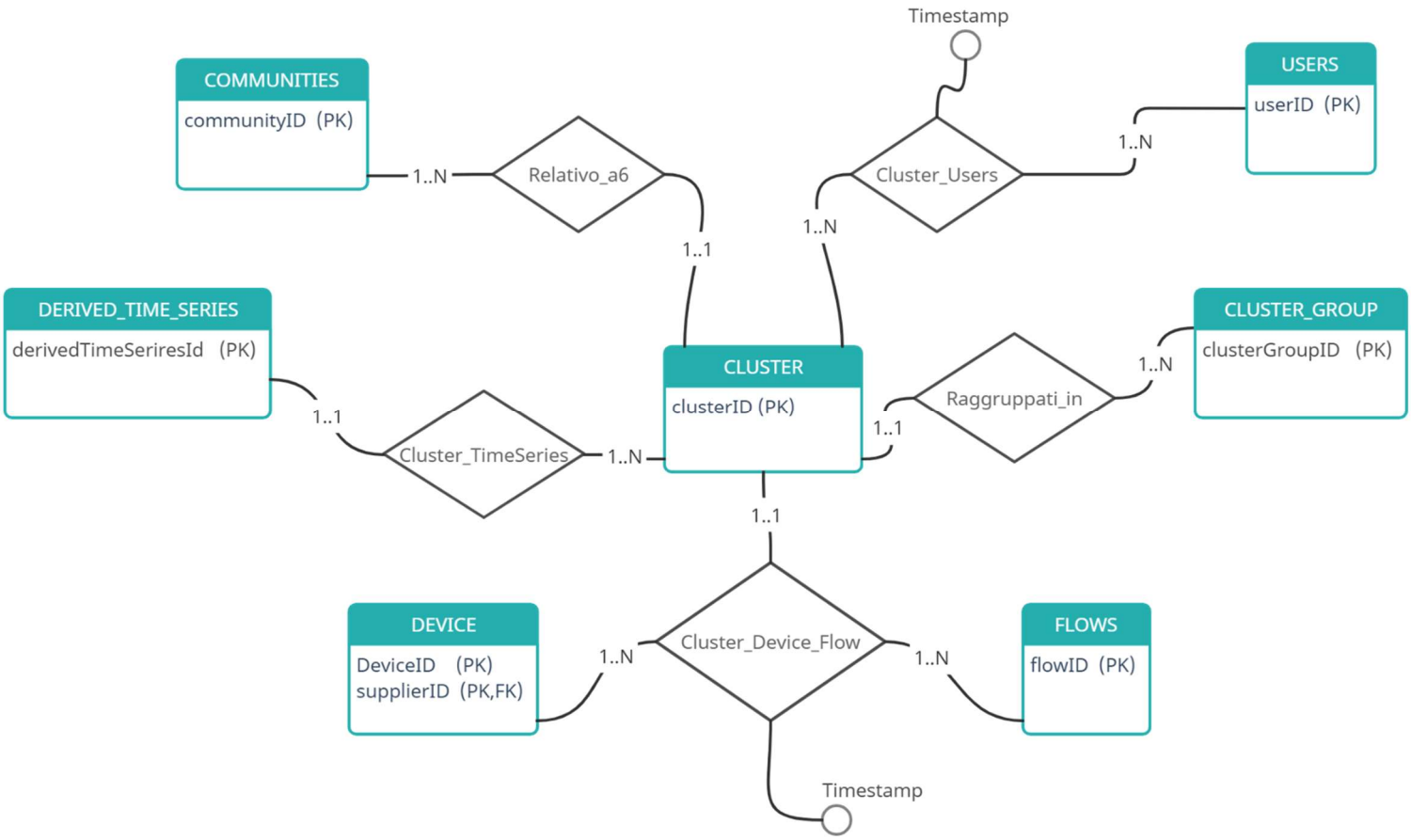

Figura 15 – Schema 6: Clusters

Flows (flowid, supplierid)

FK: supplierid **REFERENCES** Suppliers

Device (deviceid, supplierid, id, altdeviceid, supplierid, userid)

AK: id

AK: altdeviceid

FK: supplierid **REFERENCES** Suppliers

FK: userid **REFERENCES** Users

Users (userid)

Communities (communityid)

Derived_time_series (derivedtimeseriesid, clusterID)

FK: clusterId **REFERENCES** Cluster

Cluster_groups (clustergroupId)

Clusters (clusterId, clustergroupid, communityid)

FK: clustergroupid **REFERENCES** Cluster_groups

FK: communityid **REFERENCES** Communities

Clusters_Users_time (id, clusterId, userid, timestamp)

FK: clusterId **REFERENCES** Cluster

FK: userid **REFERENCES** Users

Clusters_Device_time (clusterId, deviceid, flowid, timestamp)

FK: clusterId **REFERENCES** Cluster

FK: deviceid **REFERENCES** Device

FK: flowid **REFERENCES** Flows

4.2.6. Entità metadato

Nel modello sono infine presenti alcune "entità metadato", queste sono tutte costituite da una coppia chiave(integer)-valore(varchar(30)). L'utilità di queste tabelle è quella di ridurre le dimensioni dei dati salvati nel database memorizzando una sola volta i dati che sono molto ricorrenti; inoltre operando in questo modo ci si mette al riparo da errori dovuti alla possibilità che uno stesso campo testuale venga ripetuto due volte in maniera leggermente diversa o mediante l'uso di sinonimi (es: quartorario/quarto d'ora), questo tipo di differenze non modificherebbe il significato del campo in questione ma creerebbe gravi problemi al momento delle interrogazioni del database; tali errori sarebbero inoltre molto difficili da individuare.

Si riporta di seguito la tabella per la decodifica delle "entità metadato:

- AnomalyType:
    1. Interruzione invio
    2. Errore nel dispositivo di misura
    3. Riferimenti temporali errati
    4. Riferimenti dispositivo errati
    5. Formato non conforme
    6. Misura fuori range

- Devicestatus:
    1. Funzionante
    2. Sospeso
    3. Errore
    4. Non installato

- Frequence:
    1. Secondo
    2. Minuto
    3. Cinque minuti
    4. Quartoraria
    5. Oraria
    6. Sei ore
    7. Giornaliera
    8. Settimanale
    9. Mensile
    10. Annuale
    11. Altro
    12. Una tantum

- Periodtype:
    1. Orario diurno
    2. Giorno
    3. Settimana
    4. Bisettimanale
    5. Mese
    6. Bimestre
    7. Trimestre
    8. Quadrimestre
    9. Semestre
    10. Anno
    11. Biennio
    12. Altro

- Measurementtype:

1. Totale
2. Medio
3. Minimo
4. Massimo
5. Istantaneo

- Usetype:
    1. Alloggio
    2. Parti comuni
    3. Usi condominiali
    4. Usi industriali
    5. Centralina outdoor
    6. Tetto fotovoltaico
    7. Generatore a biomasse

- Role:
    1. Unknown
    2. Registered
    3. Community administrator
    4. Oracle administrator
    5. Token administrator
    6. Econoy administrator

- Frequencyderogation:
    1. Frequenza fissa e obbligatoria
    2. Nessuna regola
    3. Frequenza derogabile in eccesso per condizione specifica

- Gdprclassification:

1. Nessuna rilevanza
2. Dati personali
3. Dati sensibili
4. Dati altamente sensibili

- Datasyntax:
    1. Json
    2. Xml
    3. Csv
    4. Txt
    5. Propietario

- Identificationmethod:
    1. POD
    2. ID del fornitore
    3. ID interno community
    4. Altro

- Derivedtimeseriestype
    1. Profilo giornaliero consumi comunità
    2. Profilo giornaliero potenza attiva comunità
    3. Profilo giornaliero produzione comunità
    4. Profilo giornaliero potenza attiva comunità
    5. Curva obiettivo consumi giornalieri di comunità
    6. Profilo settimanale produzione comunità
    7. Profilo settimanale potenza attiva comunità
    8. Curva attesa produzione giornaliera di comunità
    9. Curva attesa produzione settimanale di comunità
    10. Profilo giornaliero consumi utente

11. Profilo giornaliero potenza attiva utente
12. Profilo giornaliero produzione utente
13. Profilo giornaliero potenza attiva utente
14. Curva obiettivo consumi giornalieri di utente
15. Profilo settimanale produzione utente
16. Profilo settimanale potenza attiva utente
17. Curva attesa produzione giornaliera di utente

Curva attesa produzione settimanale di utente

4.3. Schema E/R completo

Nelle pagine seguenti sono riportati lo schema E/R relativo all'intero database e la conseguente traduzione in relazionale.

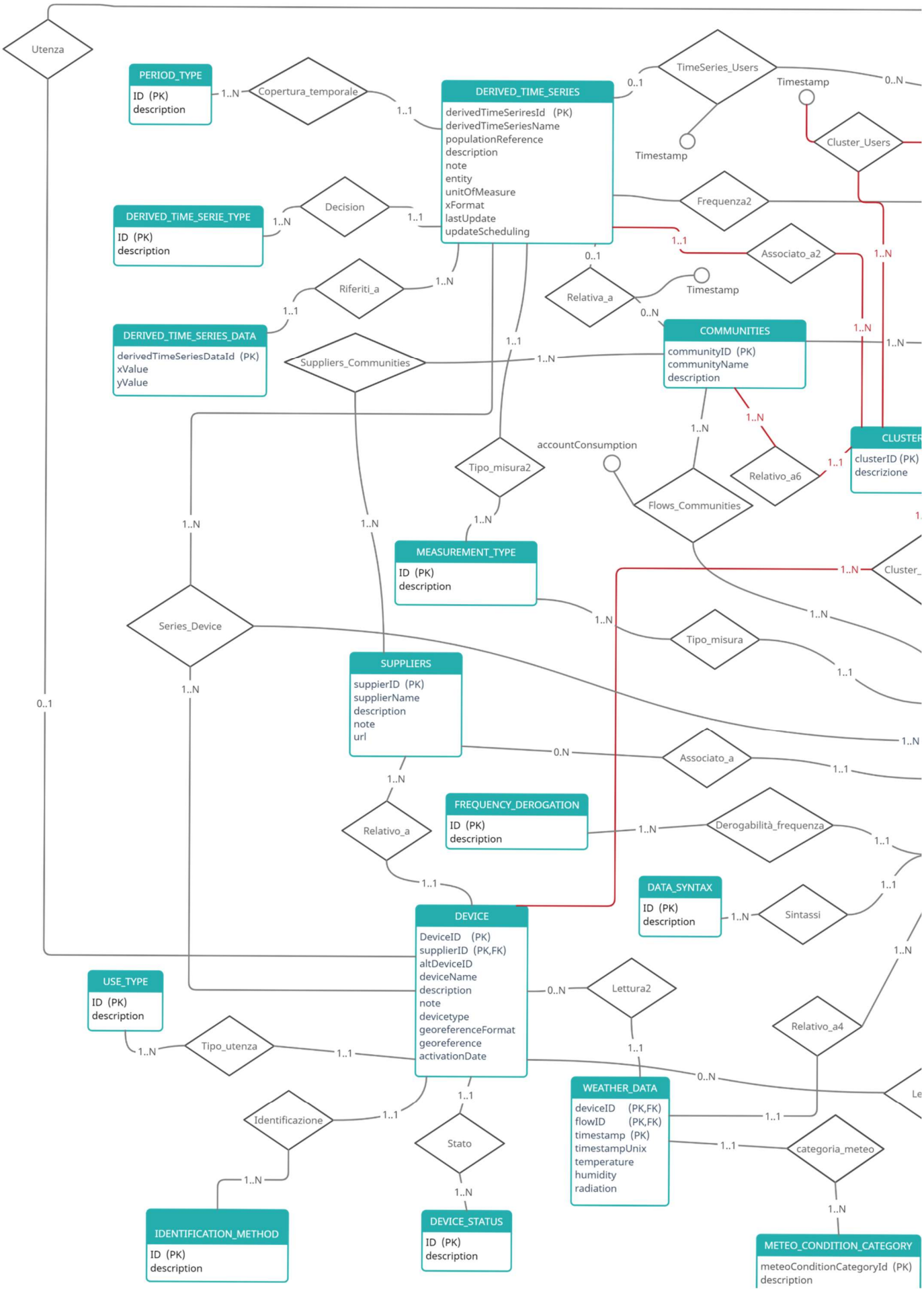

Figura 16 – Schema 7: E/R completo

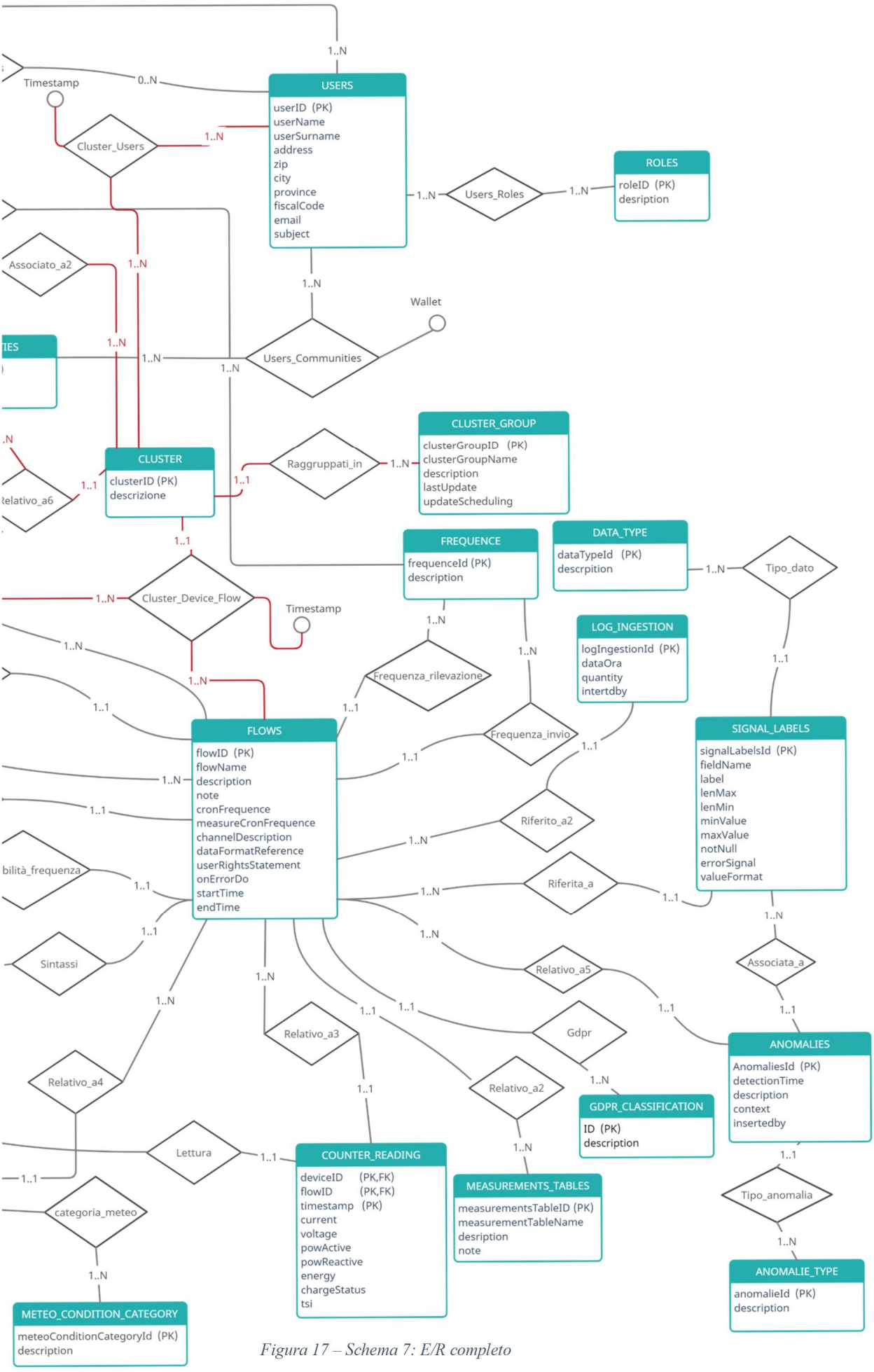

Figura 17 – Schema 7: E/R completo

Vincoli non esprimibili in E/R:

- Il Device e il Flow in relazione con Counter_reading sono relativi allo stesso Supplier.
- Flows deve essere in relazione con uno e uno solo tra Counter_reading e Weather_data.
- Device deve essere in relazione con uno e uno solo tra Counter_reading e Weather_data.
- Derived_time_series può essere riferita ad una comunità o ad un utente, in questo caso se l'utente ha più utenze attive si associa la serie a Device invece che ad User

4.4. Traduzione in relazionale

Suppliers (supplierid, suppliername, description, note, url)

Measurement_tables (measurementstableid, measurementstablename, description, note)

Frequence (id, description)

Frequency_derogation (id, description)

Measurement_type (id, description)

Data_syntax (id, description)

Gdpr_classification (id, description)

Flows (flowid, flowname, supplierid, description, note, measurementstablesid, frequence, cronfrequence, frequencyderogation, measurefrequence, measurecronfrequence, measurefrequencyderogation, measurementtype, channeldescription, datasyntax, dataformatreference, userightsstatement, gdprclassification, onerrordo, starttime, endtime)

FK: supplierid **REFERENCES** Suppliers

FK: measurementstablesid **REFERENCES** Measurements_tables

FK: frequence **REFERENCES** Frequence

FK: frequencyderogation **REFERENCES** Frequency_derogation

FK: measurefrequence **REFERENCES** Frequence

FK: measurefrequencyderogation **REFERENCES** Frequency_derogation

FK: measurementtype **REFERENCES** Measurement_type

FK: datasyntax **REFERENCES** Data_syntax

FK: gdprclassification **REFERENCES** Gdpr_classification

Identification_method (id, description)

Device_status (id, description)

Use_type (id, description)

Device (deviceid, supplierid, id, altdeviceid, identificationmethod, supplierid, userid devicename, description, note, devicetype, devicestatus, usetype, georeferenceformat, georeference, activationdate)

AK: id

AK: altdeviceid

FK: supplierid **REFERENCES** Suppliers

FK: userid **REFERENCES** Users

FK: identificationmethod **REFERENCES** Identification_method

FK: devicestatus **REFERENCES** Device_status

FK: usetype **REFERENCES** Use_type

Counter_reading (deviceid, flowid, timestamp, current, voltage, powactive, powreactive, energy, chargestatus, tsi)

FK: deviceid **REFERENCES** Device

FK: flowid **REFERENCES** Flows

Meteo_condition_category (id, description)

Weather_data (deviceid, flowid, timestamp, timestampunix, temperature, humidity, radiation, conditionid)

FK: deviceid **REFERENCES** Device

FK: flowid **REFERENCES** Flows

FK: conditionid **REFERENCES** Meteo_condition_category

Users (userid, username, usersurname, address, zip, city, province, fiscalcode, email, subject)

Roles (id, description)

Users_Roles (userid, roleid)

FK: userid **REFERENCES** Users

FK: roleid **REFERENCES** Roles

Communities (communityid, communityname, description)

Users_Communities (communityid, userid, wallet)

FK: communityid **REFERENCES** Communities

FK: userid **REFERNCES** Users

Flows_Communities (communityid, flowid, accountconsumption)

FK: communityid **REFERNCES** Communities

FK: flowid **REFERENCES** Flows

Suppliers_Communities (communityid, supplierid)

FK: communityid **REFERNCES** Communities

FK: supplierid **REFERENCES** Suppliers

Data_Type (id, description)

Signal_lables (signallabelid, flowid, fieldname, label, datatype, lenmax, lenmin, minvalue, maxvalue, notnull, errorsignal, valueformat)

FK: flowid **REFERENCES** Flows

FK: datatype **REFERENCES** Data_type

Anomaly_type (id, description)

Anomalies (id, flowid, signallabel, deviceid, anomalyType, detectiontime, description, context, insertedby)

FK: flowid **REFERENCES** Flows

FK: signallabel **REFERENCES** Signal_labels

FK: deviceid **REFERENCES** Device

FK: anomalyType **REFERENCES** Anomaly_type

Log_ingestion (id, flowid, dataeora, quantity, insertedby)

FK: flowid **REFERENCES** Flows

Derived_time_series_type (id, description)

Period_type (id, description)

Derived_time_series (derivedtimeseriesid, derivedtimeseriesname, populationreference, derivedtimeseriestype, periodtype, description, note, frequence, entity, unitofmeasure, xformat, measurementtype, lastupdate, updatescheduling, clusterId)

FK: derivedtimeseriestype **REFERNCES** Derived_time_series_type

FK: periodtype **REFERNCES** Period_type

FK: frequence **REFERNCES** Frequence

FK: measurementtype **REFERNCES** Measurement_type

FK: clusterId **REFERENCES** Cluster

Derived_time_series_data (derivedtimeseriedataid, derivedtimeserieid, yvalue, xvalue)

FK: derivedtimeserieid **REFERENCES** Derived_time_series

Derivedtimeseries_Users_time (id, derivedtimeserieid, userid, timestamp)

FK: derivedtimeseriesid **REFERNCES** Derived_time_series

FK: userid **REFERENCES** Users

Derivedtimeseries_Communities_time (id, derivedtimeserieid, communityid, timestamp)

FK: derivedtimeseriesid **REFERNCES** Derived_time_series

FK: communityid **REFERENCES** Communities

Derivedtimeseries_Device_Flow (id, derivedtimeserieid, deviceid, flowid)

FK: derivedtimeseriesid **REFERNCES** Derived_time_series

FK: deviceid **REFERENCES** Device

FK: flowid **REFERENCES** Flows

Cluster_groups (clustergroupId, clustergroupname, description, lastupdate, updatescheduling, communityid)

FK: communityid **REFERENCES** Communities

Clusters (clusterId, clustergroupid, descrizione, communityid)

FK: clustergroupid **REFERENCES** Cluster_groups

FK: communityid **REFERENCES** Communities

Clusters_Users_time (id, clusterId, userid, timestamp)

FK: clusterId **REFERENCES** Cluster

FK: userid **REFERENCES** Users

Clusters_Device_time (clusterId, deviceid, flowid, timestamp)

FK: clusterId **REFERENCES** Cluster

FK: deviceid **REFERENCES** Device

FK: flowid **REFERENCES** Flows

5. Conclusioni

Lo scopo di questo elaborato è quello di presentare il lavoro svolto durante l'attività progettuale completata assieme al gruppo di ricerca "DBGroup" del Dipartimento di Ingegneria "Enzo Ferrari" di Unimore al fine di progettare un database in grado di gestire i dati prodotti da una comunità energetica.

Partendo da un database relazionale in fase di sviluppo di ENEA, una delle principali attività svolte è stata l'analisi ed il reverse engineering di tale schema relazionale draft, producendo il corrispondente schema ER. Lo schema ER ottenuto rappresenta un importante risultato raggiunto in quanto costituisce una efficace documentazione del database in questione, documentazione che non era disponibile all'inizio del progetto. Inoltre di tale schema ER è stata fatta la completa progettazione logico-relazionale, ovvero il primo passo per l'implementazione del database su DBMS.

Concludo con alcune considerazioni generali, non tecniche, sul dominio applicativo in cui è stato svolto il lavoro di tesi, ovvero le comunità energetiche locali. I recenti cambiamenti climatici che ci troviamo ad affrontare rendono necessaria la transizione verso metodologie ecosostenibili di produzione e consumo di energia. Una soluzione percorribile in questo ambito è sicuramente quella delle comunità energetiche locali: comunità di cittadini che cooperano nella produzione di energia da fonti rinnovabili e ne condividono i frutti.

Si è visto nel corso dell'elaborato come le comunità energetiche producano quotidianamente un'enorme quantità di dati e come questi richiedano poi di essere analizzati, sintetizzati e rappresentati attraverso strumenti matematici talvolta complessi.

Ora che le comunità energetiche locali stanno iniziando a nascere sul nostro territorio, ed in vista di un futuro in qui queste sono destinate ad aumentare

in numero e dimensione, il processo di transizione energetica necessita di strumenti informatici di supporto che siano in grado di gestire questa nuova tipologia di dati e di analizzarli in modo da ottimizzare la produzione ed il consumo di energia.

6. Bibliografia

- *Domenico Beneventano, Sonia Bergamaschi, Francesco Guerra, Maurizio Vincini, "Progetto di Basi di Dati Relazionali. Lezioni ed esercizi", Pitagora Editrice Bologna, 2007*
- *WP230-006-v18-analisi_dati_x_GECO.doc.;* Documento fornito da ENEA
- *NN224-037-v1-Caratterizzazione_dei_dati_per_la_Local_Energy_Communities_-_20201020.docx;* Documento fornito da ENEA
- *NuovoReportEnea_UNIMORE_08_06_2021_v5.docx;* Documento fornito da ENEA
- *WP224-022-v2-RDS_Descrizione_architettura_dati_ENEA_AS-IS_e_TO-BE1.docx;* Documento fornito da ENEA
- *WP224-024-v1-Risorse_di_calcolo_centro_ENEA_Bologna.docx;* Documento fornito da ENEA
- *WP450-002-v4-schema_caso_uso_self_user.doc;* Documento fornito da ENEA
- *www.enea.it , ENEA,* visitato il 15/09/2021
- *www.gazzettaufficiale.it/eli/id/2016/1/18/16G00006/sg ,* visitato il 15/09/2021
- *www.tecnopolo.enea.it,* visionato il 17/09/2021
- *www.gecocommunity.it,* GECO, visionato il 17/09/2021
- *www.selfuser.it,* SELF-USER, visionato il 17/09/2021
- *www.poliseye.it,* POLIS-EYE, visionato il 17/09/2021
- *www.timescale.com,* TimescaleDB, visionato il 18/09/2021

A conclusione di questo elaborato, vorrei dedicare qualche riga a tutti coloro che mi sono stati vicini in questo percorso di crescita personale e professionale.

Innanzitutto, ringrazio la mia relatrice, la professoressa Sonia Bergamaschi, che mi ha seguito con disponibilità in ogni step della realizzazione dell'elaborato, fin dalla scelta dell'argomento.

Grazie anche al mio correlatore, il professor Domenico Beneventano, per i suoi preziosi consigli e per avermi suggerito puntualmente le giuste modifiche da apportare alla mia tesi.

Ringrazio i miei amici per rendere speciale ed indimenticabile il periodo universitario.

Un grazie a tutta la mia famiglia che, dalle Alpi fino a Malta, mi è sempre vicina, anche quando la distanza fisica lo rende difficile.

Grazie a mio fratello Christian perché, nonostante gli scontri, c'è sempre quando ho bisogno di qualcuno su cui contare.

Infine, non posso non ringraziare i miei genitori per esserci sempre, perché da sempre mi sostengono nella realizzazione dei miei progetti e perché senza di loro non avrei raggiunto questo importante traguardo del mio percorso di studi.