Report

Hospitalization-Analysis

Sommario

Introduzione

In questo report, si condivideranno i dettagli dello sviluppo di un progetto dedicato all'analisi di un interessante dataset ospedaliero reperito su Kaggle, una piattaforma online dedicata all'analisi dei dati e al machine learning. Tale dataset, comprende informazioni su oltre quattordicimila pazienti. L'attenzione si è focalizzata su due moduli chiave: l'estrazione e la pulizia dei dati (ETL) e l'analisi multidimensionale (OLAP). Nel primo modulo, si è affrontato l'ingegnerizzazione dei dati e del loro caricamento nel database, mentre nel secondo modulo, si sono integrate tecnologie come Cube.js e Apache Superset per condurre analisi avanzate e visualizzazioni dei dati. Di seguito, verranno esposti i singoli argomenti, per una migliore comprensione del progetto.

Dataset info

Di seguito sono riportate tutte le informazioni riguardanti il dataset che è stato trovato sulla piattaforma "Kaggle". Il link al dataset è il seguente:
https://www.kaggle.com/datasets/ashishsahani/hospital-admissions-data?select=HDHI+Admission+data.csv

Specifica

I dati rappresentano una collezione di oltre quattordicimila informazioni di pazienti ospedalizzati nell'arco di due anni (2017-2019).

Provenienza

I dati sono stati raccolti presso il complesso ospedaliero denominato "Hero DMC Heart Institute" situato in Ludhiana, Punjab, India.

Forma

Il dataset è organizzato in colonne denominate -acronimo, nome, tipo, significato- :

Acronimo	Nome	Tipo	Significato
SNO	Serial-number	Integer	pratica di ospedalizzazione
MRD No.	Admission number	Integer	numero di identificativo paziente
D.O.A	Date of Admission	Datetime	data di ospedalizzazione
D.O.D	Date of Discharge	Datetime	data di dimissione
AGE	Age	Integer	età del paziente
GENDER	Gender	Enum	sesso del paziente • [M] - male • [F] - female
RURAL	Rural	Enum	area urbana di residenza • [U]rban • [R]ural
TYPE OF ADMISSION-EMERGENCY/OPD	Type of admission-emergency/opd	Enum	ospedalizzazione per emergenza, oppure ospedalizzazione per visite cicliche o trattamenti • [E]mergency • [O]PD
DURATION OF STAY	Duration of stay	Integer	giorni di ospedalizzazione
DURATION OF INTENSIVE UNIT STAY	Duration of intensive unit stay	Integer	giorni in terapia intensiva

OUTCOME	Outcome	Varchar	risultato dell'ospedalizzazione • [DAMA] Discharged Against Medical Advice • [DISCHARGE] Dimesso • [EXPIRY] Deceduto
SMOKING	Smoking	Bool	• 0 paziente non tabagista • 1 paziente tabagista
ALCOHOL	Alcohol	Bool	• 0 il paziente non assume periodicamente alcol • 1 il paziente assume periodicamente alcol
DM	Diabetes mellitus	Bool	• 0 il paziente non ha il diabete mellito • 1 il paziente ha il diabete mellito
HTN	Hypertension	Bool	• 0 il paziente non soffre di ipertensione • 1 il paziente soffre di ipertensione
CAD	Coronary artery disease	Bool	• 0 il paziente non soffre di coronaropatia • 1 il paziente soffre di coronaropatia
PRIOR CMP	Cardiomyopathy	Bool	• 0 il paziente non soffre di 0 il paziente non soffre di cardiomiopatia • 1 il paziente soffre di cardiomiopatia
CKD	Chronic kidney disease	Bool	• 0 il paziente non soffre di malattia renale cronica • 1 il paziente soffre di malattia renale cronica
HB	Haemoglobin	Double	valore rilevato di emoglobina
TLC	Total leukocytes count	Double	conteggio totale dei leucociti
PLATELETS	Platelets	Integer	conteggio delle piastrine
GLUCOSE	Glucose	Integer	valore rilevato del glucosio
UREA	Urea	Integer	valore rilevato dell'urea
CREATININE	Creatinine	Double	valore rilevato di creatinina
BNP	B-type natriuretic peptide	Integer	valore rilevato di peptide natriuretico di tipo B (anche detto peptide natriuretico cerebrale)
RAISED CARDIAC ENZYMES	Raised cardiac enzymes	Bool	• 0 il paziente ha un numero di enzimi cardiaci nella norma • 1 il paziente ha un numero di enzimi cardiaci alto
EF	Ejection fraction	Integer	valore rilevato di frazione di eiezione
SEVERE ANAEMIA	Severe anaemia	Bool	• 0 il paziente non soffre di anemia grave

			• 1 il paziente soffre di anemia grave
ANAEMIA	Anaemia	Bool	• 0 il paziente non soffre di anemia • 1 il paziente soffre di anemia
STABLE ANGINA	Stable angina	Bool	• 0 il paziente non ha l'angina stabile • 1 il paziente non soffre di angina stabile
ACS	Acute coronary Syndrome	Bool	• 0 il paziente non soffre di sindrome coronarica acuta • 1 il paziente soffre di sindrome coronarica acuta
STEMI	St elevation myocardial infarction	Bool	• 0 il paziente non ha avuto un infarto al miocardico con sopraslivellamento del tratto ST • 1 il paziente ha avuto un infanto al miocardico con sopraslivellamento del tratto ST
ATYPICAL CHEST PAIN	Atypical chest pain	Bool	• 0 il paziente soffre di un dolore atipico al petto • 1 il paziente soffre di un dolore atipico al petto
HEART FAILURE	Heart failure	Bool	• 0 il paziente non ha avuto un'insufficienza cardiaca • 1 il paziente ha avuto un'insufficienza cardiaca
HFREF	Heart failure with reduced ejection fraction	Bool	• 0 il paziente non ha un insufficienza cardiaca con frazione di eiezione ridotta. • 1 il paziente ha un insufficienza cardiaca con frazione di eiezione ridotta.
HFNEF	Heart failure with normal ejection fraction	Bool	• 0 il paziente non ha un insufficienza cardiaca con frazione di eiezione normale • 1 il paziente ha un insufficienza cardiaca con frazione di eiezione normale
VALVULAR	Valvular heart disease	Bool	• 0 il paziente non soffre di malattie cardiache valvolari • 1 il paziente soffre di malattie cardiache valvolari
CHB	Complete heart block	Bool	• 0 il paziente non ha avuto un blocco cardiaco completo • 1 il paziente ha avuto un blocco cardiaco completo

SSS	Sick sinus syndrome	Bool	• 0 il paziente non ha la sindrome del nodo del seno • 1 il paziente ha la sindrome del nodo del seno
AKI	Acute kidney injury	Bool	• 0 il paziente non ha un danno renale acuto • 1 il paziente ha un danno renale acuto
CVA INFRACT	Cerebrovascular accident infarct	Bool	• 0 il paziente non ha avuto un infarto da incidente cerebrovascolare • 1 il paziente ha avuto un infarto da incidente cerebrovascolare
CVA BLEED	Cerebrovascular accident bleed	Bool	• 0 il paziente non ha un sanguinamento da incidente cerebrovascolare • 1 il paziente ha un sanguinamento da incidente cerebrovascolare
AF	Atrial fibrilation	Bool	• 0 il paziente non ha una fibrillazione atriale • 1 il paziente ha una fibrillazione atriale
VT	Ventricular tachycardia	Bool	• 0 il paziente non ha una tachicardia ventricolare • 1 il paziente ha una tachicardia ventricolare
PSVT	Paroxysmal supra ventricular tachycardia	Bool	• 0 il paziente non ha una tachicardia parossissima sporaventricolare • 1 il paziente ha una tachicardia parossissima sopraventricolare
CONGENITAL	Congenital heart disease	Bool	• 0 il paziente non ha una cardiopatia congenita • 1 il paziente ha una cardiopatia congenita
UTI	Urinary tract infection	Bool	• 0 il paziente non ha un'infezione del tratto urinario • 1 il paziente ha un'infezione del tratto urinario
NEURO CARDIOGENIC SYNCOPE	Neuro cardiogenic syncope	Bool	• 0 il paziente non ha una sincope neurocardiogenica • 1 il paziente ha una sincope neurocardiogenica
ORTHOSTATIC	Orthostatic	Bool	• 0 il paziente non soffre di ipotenzione ortostatica • 1 il paziente soffre di ipotenzione ortostatica

INFECTIVE ENDOCARDITIS	Infective endocarditis	Bool	• 0 il paziente non ha una endocardite infettiva • 1 il paziente ha una endocardite infettiva
DVT	Deep venous thrombosis	Bool	• 0 il paziente non ha una trombosi venosa profonda • 1 il paziente ha una trombosi venosa profonda
CARDIOGENIC SHOCK	Cardiogenic shock*	Bool	• 0 il paziente non ha avuto uno shock cardiogenico • 1 il paziente ha avuto uno shock cardiogenico
SHOCK	Shock	Bool	• 0 il paziente non ha avuto uno shock • 1 il paziente ha avuto uno shock
PULMONARY EMBOLISM	Pulmonary shock	Bool	• 0 il paziente non ha avuto uno shock polmonare • 1 il paziente ha avuto uno shock polmonare
CHEST INFECTION	Chest infection	Bool	• 0 il paziente non ha un'infezione toracica • 1 il paziente ha un'infezione toracica

*Lo shock è definito sia come una pressione sanguigna sistolica < 90mmHg che quando la causa dello shock è qualsiasi cosa non cardiaca.

Pulizia e Normalizzazione dei dati

Analizzando il dataset, si è riscontrata la presenza di alcuni parametri che presentano delle istanze vuote/nulle, e sono:

- BNP, peptide natriuretico cerebrale (4% di istanze vuote, 54% di istanze nulle)
- EF, frazione di eiezione (10% di istanze nulle, 6 istanze su ~13000 sono vuote)
- Glucosio (5% di istanze nulle, 12 istanze su ~13000 sono vuote)
- Piastrine (2% di istanze nulle, 1 istanza su ~13000 è vuota)

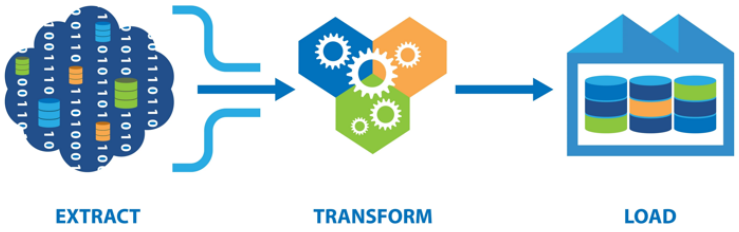
Si è deciso di tenere conto solamente dei record che presentano un numero inferiore o pari a due parametri inerenti alle analisi svolte durante l'ospedalizzazione, con valori uguali null/empty.

Sarebbe stato opportuno valutare una soluzione per i record che presentano un BNP o un EF null/empty, in quanto questi due parametri rappresentano un indicatore per una possibile insufficienza cardiaca. Però, se si scartassero tutti i record che presentano questa caratteristica, si otterrebbe un dataset di dimensione dimezzata (il 54% ~ 64% in meno).
Si potrebbe ovviare a questo problema effettuando un'analisi statistica sui valori di BNP ed EF in base ai restanti valori delle analisi. Tuttavia, in questo caso, sarebbe necessario il supporto di uno specialista che indichi quali sono le regole per poter stabilire un valore approssimato. Pertanto, si è deciso di mantenere i record che presentino valori null/empty sui parametri BNP ed EF.

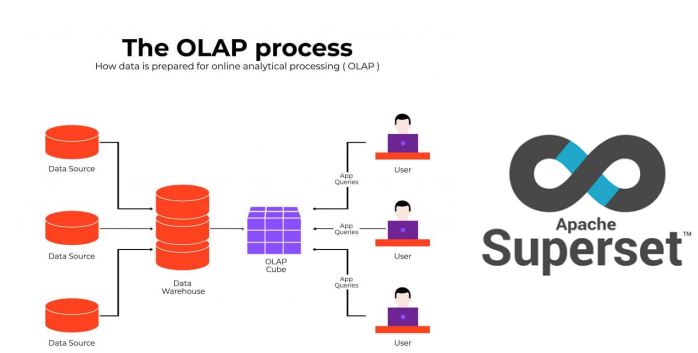
Progetto

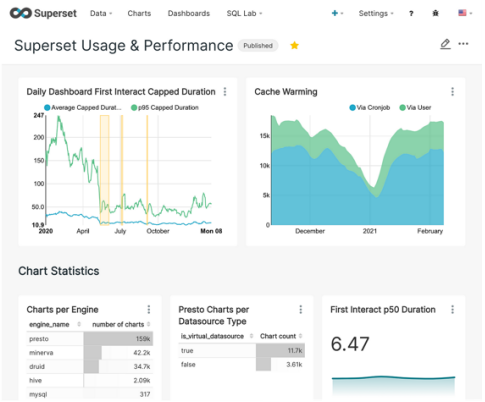
Il progetto verrà suddiviso in 2 moduli:

- Il primo riguarda la lettura dei dati da fonti aperte, il processamento e l'ingegnerizzazione dei dati. Verrà previsto il salvataggio di questi dati in una database e verrà costruita la parte dei servizi per permettere l'interazione con il layer progettato.



- Il secondo modulo vedrà invece l'integrazione e lo sviluppo della parte di data visualization, con lo scopo di fornire analisi multidimensionali dei dati ingegnerizzati nella fase precedente.





Operativamente il progetto verrà suddiviso in 3 parti, che di seguito verranno approfondite:

- ETL
- OLAP
- Superset

Per ognuno dei seguenti moduli è previsto l'utilizzo delle tecnologie elencate:

- Data Processing:
  - Java EE
  - Postgresql
- Data Visualization e analisi multidimensionale:
  - Cube.js
  - Apache Superset
- Running environment:
  - Docker

Il documento avrà in calce ai tre paragrafi, dedicati alle tre sezioni sopra indicate, un paragrafo dedicato all'ambiente nel quale i tre moduli vengono eseguiti e a come esso debba essere configurato.

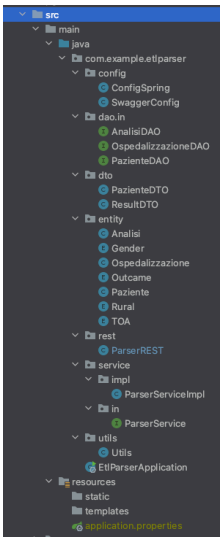
ETL - Extract, transform, load

Il progetto prevede una parte di data mining e pulizia del dataset; questo avverrà seguendo le regole precedentemente indicate. La selezione dei dei dati da fonti aperte avverrà controllando in prima istanza, il formato del file che verrà caricato, il quale se non sarà conforme (.csv) verrà automaticamente scartato. In seguito, verrà controllato se il file csv presenta le colonne necessarie per poter normalizzare le tuple in oggetti; qualora non fosse possibile, il file verrà scartato. Se i dati saranno correttamente formattati, avverrà il caricamento nel database attraverso JPA e verrà salvata una copia del file sul server in una cartella separata, seguendo il seguente formato:

"./FileUploadX/namefile"

dove X rappresenta il numero di file caricati e "namefile" il nome del file. Tutto ciò verrà offerto da un metodo REST esposto in localHost all'indirizzo http://localhost:8080/parser/uploadCVS. Il modulo non prevede una parte front-end, quindi il testing blackbox è avvenuto attraverso Swagger all'indirizzo "http://localhost:8080/swagger-ui/index.html". I test di unità e di integrazione sono stati svolti con l'utilizzo di JUnit e Mockito, e sono disponibili in "/test/java/com.example.etlparser/".

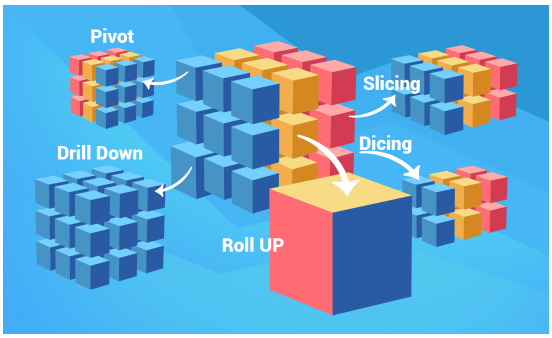
La struttura del progetto è organizzata seguendo il formato aziendale (come in figura).



OLAP - On-Line Analytical Processing

Questa metodologia riveste un ruolo cruciale nella gestione e nell'analisi dei dati, fornendo agli utenti strumenti potenti per esplorare, analizzare e comprendere informazioni complesse in modo intuitivo e tempestivo. Uno degli aspetti chiave di OLAP, è rappresentato dai "cubi", strutture multidimensionali che consentono di organizzare i dati in un formato che riflette la struttura naturale delle informazioni aziendali. Questi cubi vengono spesso chiamati "cubi OLAP" e sono progettati per consentire una navigazione agevole attraverso diverse dimensioni, come tempo, prodotto, area geografica e altri attributi rilevanti per l'analisi. All'interno di questi cubi, gli utenti possono eseguire una serie di operazioni analitiche fondamentali, che consentono di estrarre valore dai dati in modi significativi. Alcune delle operazioni più rilevanti includono:

1. **Drill-down e Roll-up**: Attraverso queste operazioni, gli utenti possono esplorare dati a diversi livelli di dettaglio o aggregarli per ottenere una visione più ampia;
2. **Slice-and-dice**: Questa operazione consente di "affettare" il cubo, selezionando una porzione specifica di dati in base a determinati criteri;
3. **Pivot:** Questa operazione consente agli utenti di scambiare dimensioni o aspetti del cubo per esaminare i dati da prospettive diverse;
4. **Calcolo di Misure**: Gli utenti possono definire nuove misure o indicatori di performance derivati da quelli esistenti, consentendo un'analisi più approfondita e personalizzata;
5. **Analisi di Tendenze e Variazioni**: Utilizzando i cubi OLAP, è possibile identificare tendenze nel tempo e analizzare le variazioni nei dati per prendere decisioni informate;



Apache Kylin

Apache Kylin è un motore OLAP avanzato che garantisce prestazioni eccezionali su grandi volumi di dati. La versione 5.0 introduce importanti miglioramenti, tra cui un'architettura Cloud Native, un motore nativo per l'accelerazione e una nuova interfaccia utente, rendendo Kylin una soluzione completa e all'avanguardia per le esigenze analitiche delle organizzazioni nel contesto dei Big Data. Il progetto prevedeva l'utilizzo di Apache Kylin, il quale richiedeva molti prerequisiti:

- Software
  - Hadoop: cdh5.x, cdh6.x, hdp2.x, EMR5.x, EMR6.x, HDI4.x
  - Hive: 0.13 - 1.2.1+
  - Spark: 2.4.7/3.1.1
  - Mysql: 5.1.7 and above
  - JDK: 1.8+
  - OS: Linux only, CentOS 6.5+ or Ubuntu 16.0.4+
- Hardware
  - The minimum configuration of a server running Kylin is 4 core CPU, 16 GB RAM and 100 GB disk.
    - For high-load scenarios, a 24-core CPU, 64 GB RAM or higher is recommended.

Dopo aver seguito la guida presente nella documentazione (https://kylin.apache.org/5.0/docs/quickstart/deploy_kylin#download-and-install), non è stato possibile effettuare la configurazione in locale, in quanto, dopo aver scaricato, installato e configurato tutto il necessario il comando "hadoop fs" produceva un errore.
Si è pensato, quindi, di utilizzare Kylin in una configurazione docker-compose, seguendo la guida ufficiale: https://hub.docker.com/r/apachekylin/apache-kylin-standalone.
La versione Docker di Kylin, invece riusciva ad essere eseguita, ma non configurata e di conseguenza era impossibile configurare il dataset postgres e/o mysql per poter creare i cubi.
Di seguito verrà riportato un esempio della configurazione che andava implementata per inizializzare Kylin.

```
version: '3'
services:
  app:
    build:
      context: .
    container_name: app
    ports:
      - "8080:8080"
    depends_on:
      - db
    volumes:
      - ./target/ETL-Parser-0.0.1-SNAPSHOT.jar:/app/app.jar
    networks:
      - backend
  pgadmin:
    image: dpage/pgadmin4:latest
    container_name: pgadmin
    ports:
      - "16002:80"
    environment:
      PGADMIN_DEFAULT_EMAIL: danielerusso_@hotmail.it
      PGADMIN_DEFAULT_PASSWORD: 123
    volumes:
      - ./pgadmin/servers.json:/pgadmin4/servers.json
    networks:
      - backend
    depends_on:
      - db
```

```
  db:
    image: 'postgres:13.1-alpine'
    container_name: db
    ports:
      - "16003:5432"
    environment:
      - POSTGRES_USER=admin
      - POSTGRES_PASSWORD=123
      - POSTGRES_DB=postgres
    volumes:
      - postgres:/var/lib/postgresql/data
      - ./database/postgres/init-database.sh:/docker-entrypoint-initdb.d/init-database.sh
      - ./database/postgres/init_sql:/init_sql
    networks:
      - backend

  kylin:
    container_name: kylin
    image: apachekylin/apache-kylin-standalone:5.0-beta
    ports:
      - "7070:7070"
      - "8088:8088"
      - "9870:9870"
      - "8032:8032"
      - "8042:8042"
      - "2181:2181"
    command: -Dkylin.config=./kylin/config/kylin.properties
    environment:
      - JDBC_URL='jdbc:postgresql://db:5432/postgres'
      - JDBC_USER=admin
      - JDBC_PASSWORD=123
      - KYLIN_CONF_HOME=/home/kylin/apache-kylin-5.0.0-beta-bin/conf
    networks:
      - backend
    volumes:
     - ./kylin/conf/kylin.properties:/home/kylin/apache-kylin-5.0.0-beta-bin/conf/kylin.properties
     - ./kylin/lib/ext/postgresql-42.6.0.jar:/home/kylin/apache-kylin-5.0.0-beta-
bin/lib/ext/postgresql-42.6.0.jar
     - ./kylin/conf/datasource/POSTGRESQL.xml:/home/kylin/apache-kylin-5.0.0-beta-
bin/conf/datasource/POSTGRESQL.xml
    links:
      - db

networks:
  backend:
    driver: bridge
volumes:
  postgres:
```

Considerato ciò si è provveduto alla ricerca di un gestore di cubi equivalente che non modificasse le finalità. Dopo una ricerca approfondita ed accurata, si è ritenuto di utilizzare Cube.js che è compatibile con lo stack tecnologico precedentemente indicato.

Cube.js

Cube.js è un framework open source che semplifica sia la connessione tra silos di dati, la creazione di metriche coerenti che l'accessibilità, attraverso API, a diverse applicazioni. Il punto chiave di Cube.js è la possibilità di creare "cubi dati" utilizzando un approccio code-first, dove gli ingegneri definiscono modelli di dati in codice YAML o JavaScript. Questi cubi rappresentano entità aziendali e consentono di definire calcoli specifici tramite misurazioni e dimensioni. Inoltre, Cube.js supporta la creazione di viste, semplificando l'interazione con il modello di dati. Questo approccio agile e code-first, facilita la gestione dei modelli di dati attraverso il controllo di versione, favorendo la collaborazione e la manutenibilità. Cube.js è stato adottato con successo da aziende come Cloud Academy, Cyndx e Cuboh, portando le applicazioni ad uno sviluppo più rapido, tempi di inattività dell'analisi ridotti e prestazioni ottimizzate.
Cube.js mette a disposizione sia cubi che le view per organizzare i dati:

- **Cube**: Rappresenta un set di dati multidimensionale. Può essere pensato come una raccolta di misure e dimensioni che possono essere esplorate e analizzate. Un cubo definisce quali dati sono disponibili per l'analisi e come possono essere aggregati in diverse dimensioni.
- **View**: Rappresentazione specifica dei dati in un cubo. Può essere considerata come una proiezione dei dati del cubo che specifica quali colonne e aggregazioni sono disponibili. Le viste consentono di definire insiemi specifici di dati e calcoli che possono essere utilizzati per rispondere a domande analitiche specifiche.

In breve, mentre il "cubo" rappresenta l'insieme complessivo di dati multidimensionali, la "view" è una rappresentazione specifica di quei dati, progettata per rispondere a esigenze analitiche particolari. Entrambi giocano un ruolo chiave nell'organizzazione e nell'esposizione dei dati in Cube.js.

N:B: nella sezione finale, verranno approfonditi i dettigli di configurazione per la comunicazione tra Cube.js e Postgres oltre a Superset.

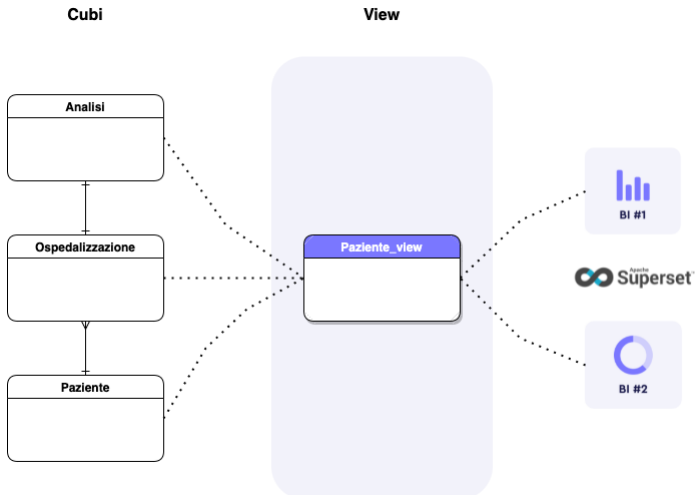
Cubi

Dopo aver inizializzato l'ambiente, si è provveduto alla creazione dei tre cubi che di seguito verranno riportati.

- *Paziente.yml*
- *Ospedalizzazione.yml*
- *Analisi.yml*

View

Per una corretta esposizione dei dati, è stata creata un view da esporre a Superset disponibile qui. Il grafico seguente, può essere esplicativo per capire la struttura del modulo.



Casi d'uso

Di seguito, verranno riportati alcuni casi d'uso che verranno sviluppati e trasformati in query da effettuare sui dati:

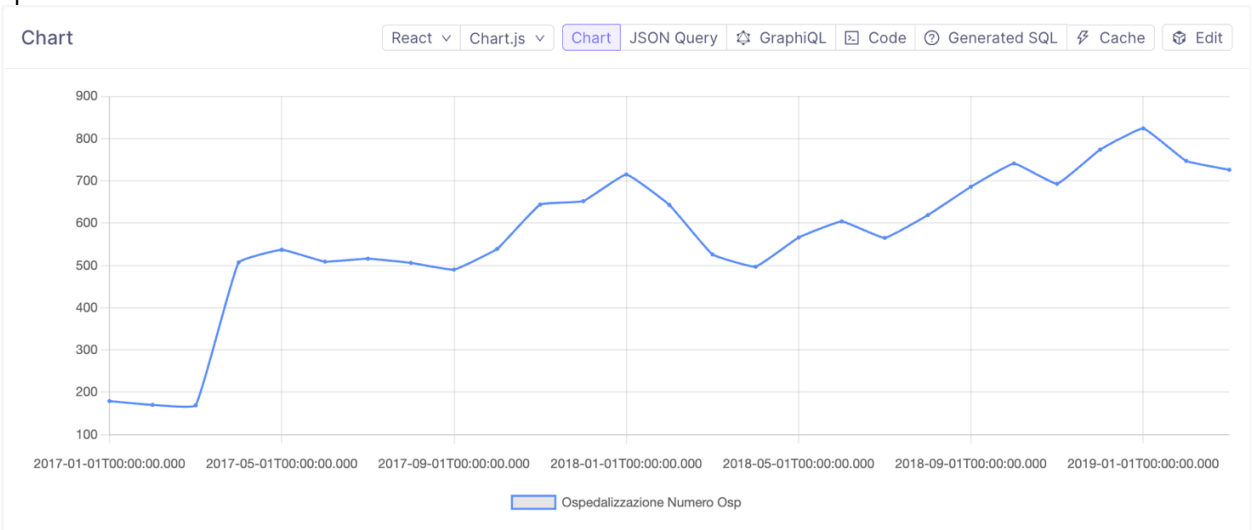
- Contare il numero di ospedalizzazioni per ciascun mese per i due anni dello studio
- Analizzare la provenienza geografica dei pazienti, ovvero Rural o Urban
    - Analizzare se ci siano dei trend in relazione alla provenienza
- Analizzare le patologie che colpiscono di più le persone, come
    - Malattia coronarica [66.96%]
    - Ipertensione [48.59%]
    - Sindrome coronarica acuta [36.57%]
    - Insufficienza cardiaca [28.95%]
    - Insufficienza renale acuta [22.24%]

I casi d'uso seguiranno la seguente formattazione:

- Idea del caso d'uso
- Grafico
- Query in JSON
- Conclusioni

Primo caso d'uso

L'obiettivo del caso d'uso è capire se durante l'arco di due anni, il numero dei casi di ospedalizzazione è aumentato.

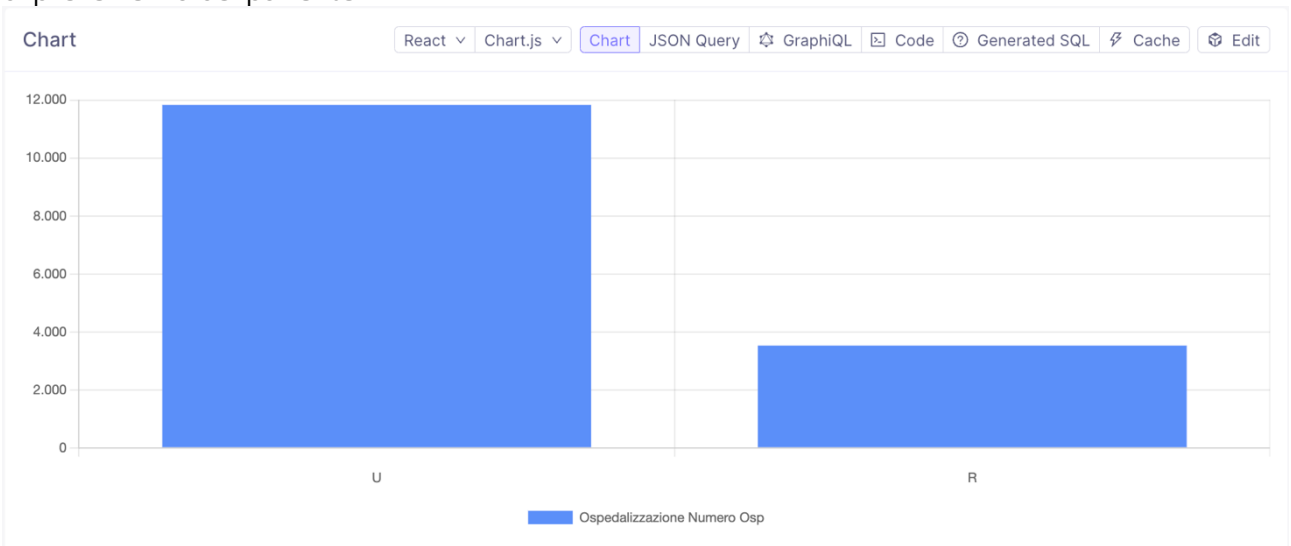



```
{
  "limit": 15000,
  "measures": [
    "ospedalizzazione.numero_osp"
  ],
  "timeDimensions": [
    {
      "dimension": "ospedalizzazione.date_admission",
      "granularity": "month",
      "dateRange": [
        "2017-01-01",
        "2019-03-31"
      ]
    }
  ]
}
```

Il grafico sembra confermare che al passare del tempo il numero di ospedalizzazione aumenti.

Secondo caso d'uso

L'obiettivo del caso d'uso è capire se l'incidenza di ospedalizzazione aumenta in relazione all'area di provenienza del paziente.



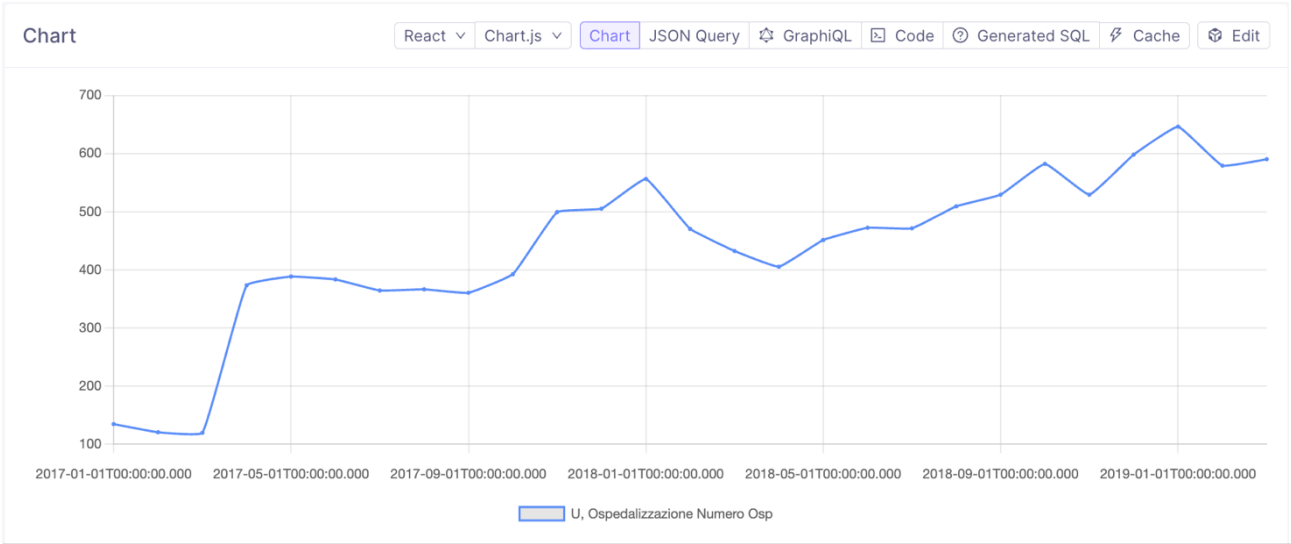

```
{
  "limit": 15000,
  "measures": [
    "ospedalizzazione.numero_osp"
  ],
  "timeDimensions": [
    {
      "dimension": "ospedalizzazione.date_admission",
      "dateRange": [
        "2016-12-01",
        "2019-04-30"
      ]
    }
  ],
  "dimensions": [
    "paziente.rural"
  ],
  "order": {
    "ospedalizzazione.numero_osp": "desc"
  }
}
```

Il grafico sembra indicare che le persone provenienti dai contesi urbani, abbiano maggior probabilità di essere ospedalizzati. Per tale motivo si è deciso di volgere un caso d'uso aggiuntivo, che valuti la presenza di trend relativi alla provenienza dei pazienti.

Analisi avanzata

Si potrebbe pensare di verificare se per singola area di provenienza, questa incida sull'aumento delle ospedalizzazioni.

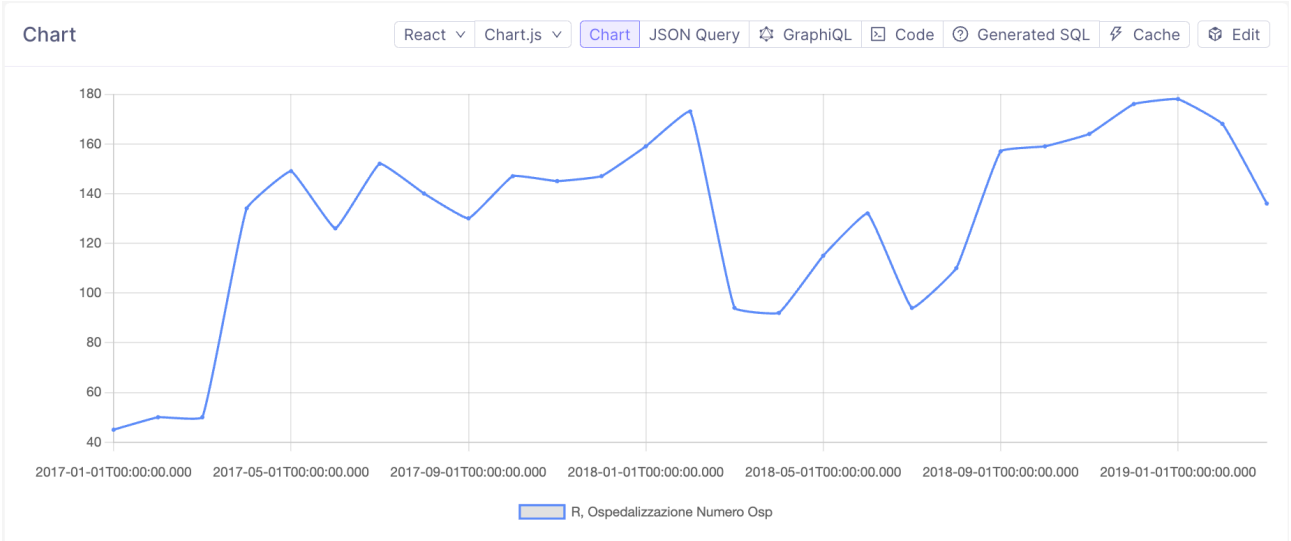
Contesto urbano




```
{
  "limit": 15000,
  "measures": [
    "ospedalizzazione.numero_osp"
  ],
  "timeDimensions": [
    {
      "dimension": "ospedalizzazione.date_admission",
      "granularity": "month",
      "dateRange": [
        "2017-01-01",
        "2019-03-31"
      ]
    }
  ],
  "dimensions": [
    "paziente.rural"
  ],
  "order": {
    "ospedalizzazione.numero_osp": "desc"
  },
  "filters": [
    {
      "member": "paziente.rural",
      "operator": "equals",
      "values": [
        "U"
      ]
    }
  ]
}
```

I dati sembrano mostrare che, per i contesti urbani, i casi di ospedalizzazione aumentino di anno in anno.

Contesto Rurale




```
{
  "limit": 15000,
  "measures": [
    "ospedalizzazione.numero_osp"
  ],
  "timeDimensions": [
    {
      "dimension": "ospedalizzazione.date_admission",
      "granularity": "month",
      "dateRange": [
        "2017-01-01",
        "2019-03-31"
      ]
    }
  ],
  "dimensions": [
    "paziente.rural"
  ],
  "order": {
    "ospedalizzazione.numero_osp": "desc"
  },
  "filters": [
    {
      "member": "paziente.rural",
      "operator": "equals",
      "values": [
        "R"
      ]
    }
  ]
}
```

D'altra parte, le ospedalizzazioni delle persone provenienti dai contesti rurali, sembrano rimanere costati nel tempo.

Terzo caso d'uso

L'idea era quella di soffermarsi su due patologie che, in ambiti diversi, coprono il maggior spettro dei ricoverati, sui quali si hanno dati che permettano di effettuare analisi, ovvero:

- Malattia coronarica
- Insufficienza renale acuta

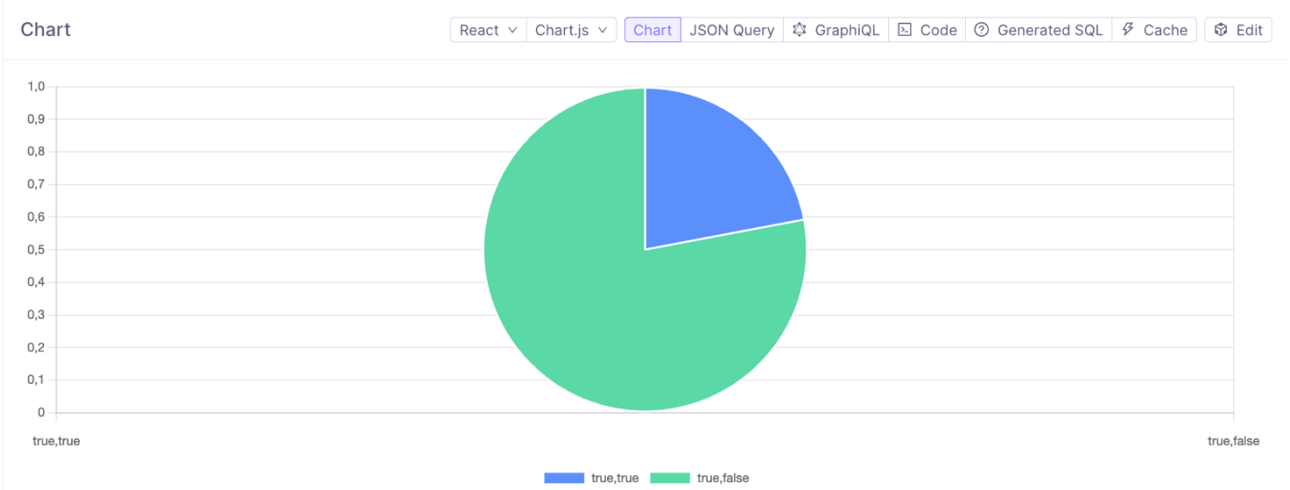
Inizialmente si volevano valutare anche i pazienti affetti da ipertensione, ma bisogna notare che il parametro fondamentale sull'analisi dell'ipertensione, è la pressione arteriosa, misurabile con lo sfigmomanometro, dato che però non è stato riportato nel dataset.

Malattia coronarica

Durante l'analisi della malattia coronarica, è fondamentale avere valori normali di Reised cardiac enzymis i quali molto spesso non risultano essere nei valori limite per tali pazienti.

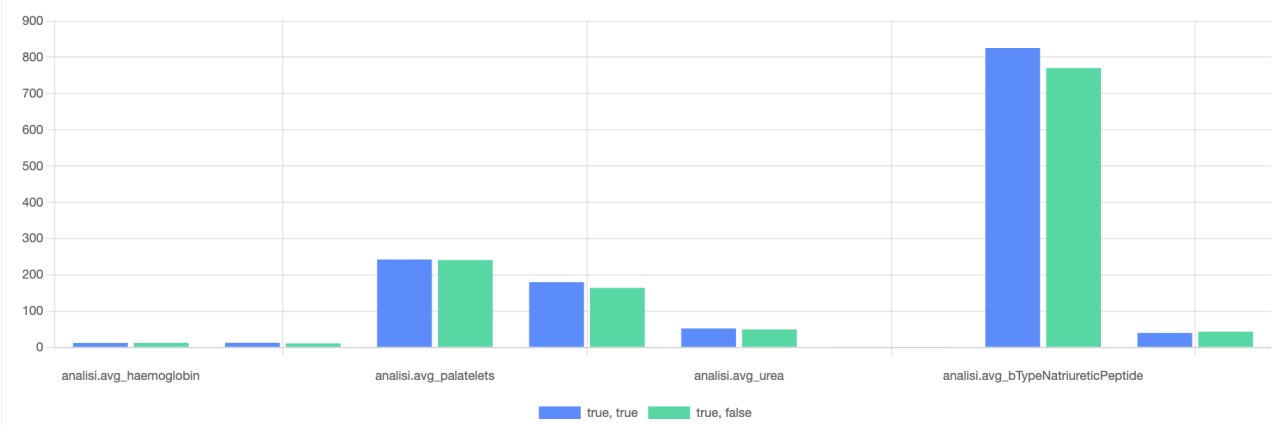
Leggenda:

- **Blu**: Pazienti affetti da malattia coronarica e con un valore di enzimi cardiaci nella norma;
- **Verde**: Pazienti affetti da malattia coronarica e con un valore di enzimi cardiaci NON nella norma;



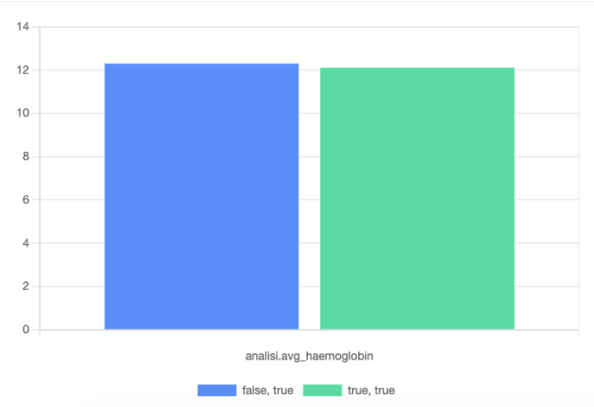

```
{
  "limit": 15000,
  "dimensions": [
    "paziente.coronary_artery_disease",
    "analisi.raised_cardiac_enzymes"
  ],
  "order": {
    "ospedalizzazione.date_admission": "asc"
  },
  "filters": [
    {
      "member": "paziente.coronary_artery_disease",
      "operator": "equals",
      "values": [
        "true"
      ]
    }
  ],
  "measures": [
    "ospedalizzazione.numero_osp"
  ]
}
```

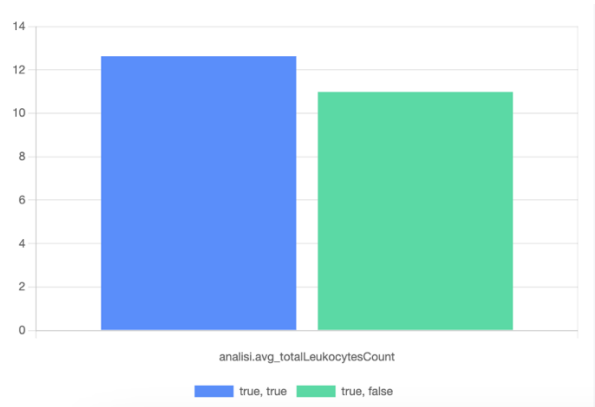
Si può vedere come, chi non ha una giusta quantità di questi enzimi, risulti essere il 77.97% del totale di chi è afflitto da tale patologia; alla luce di questo dato, ho effettuato un confronto medio dei valori delle analisi:

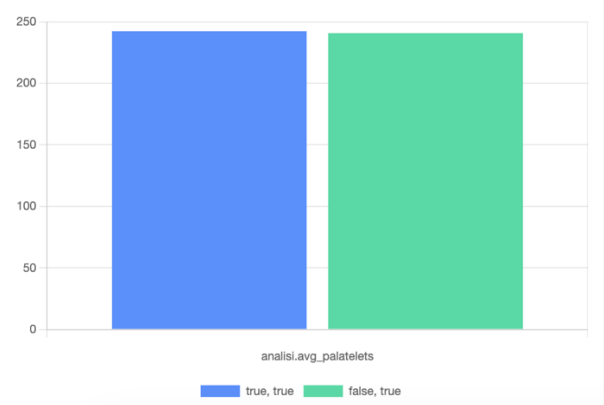


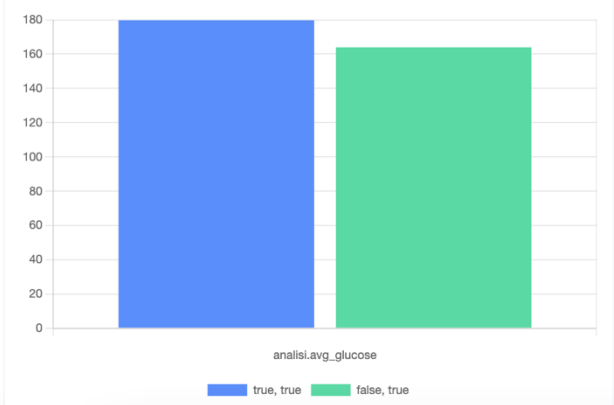

```
{
  "limit": 15000,
  "measures": [
    "analisi.avg_palatelets",
    "analisi.avg_glucose",
    "analisi.avg_urea",
    "analisi.avg_creatinine",
    "analisi.avg_bTypeNatriureticPeptide",
    "analisi.avg_ejectionFraction"
  ],
  "dimensions": [
    "paziente.coronary_artery_disease",
    "analisi.raised_cardiac_enzymes"
  ],
  "order": {
    "ospedalizzazione.numero_osp": "desc"
  },
  "filters": [
    {
      "member": "paziente.coronary_artery_disease",
      "operator": "equals",
      "values": [
        "true"
      ]
    }
  ]
}
```

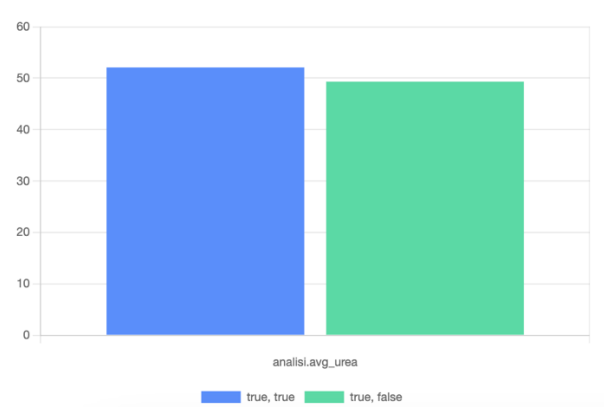
Riscontrando che chi soffre di patologie coronariche, ha valori raised cardiac enzymis non nella norma: piastrine, glucosio, urea, creatinina, b type natruretic peptite e total leukocytes cont valori **inferiori**, mentre per ejection fraction e emoglobina, valori **superiori**. Per una corretta visualizzazione dei dati, verranno riportati i grafici per ogni singolo caratterizzante preso in esame.

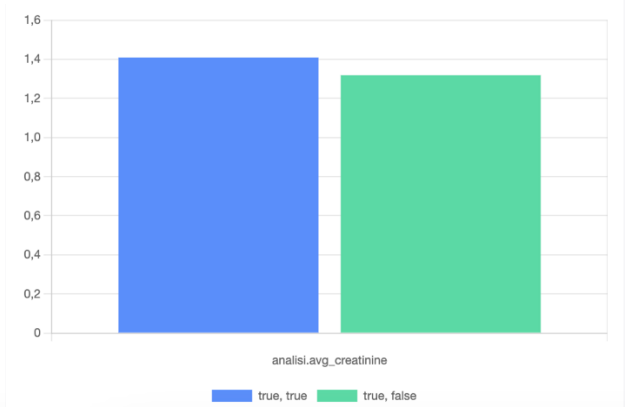


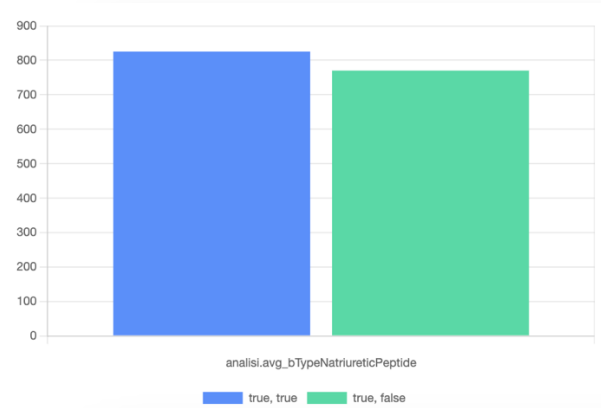


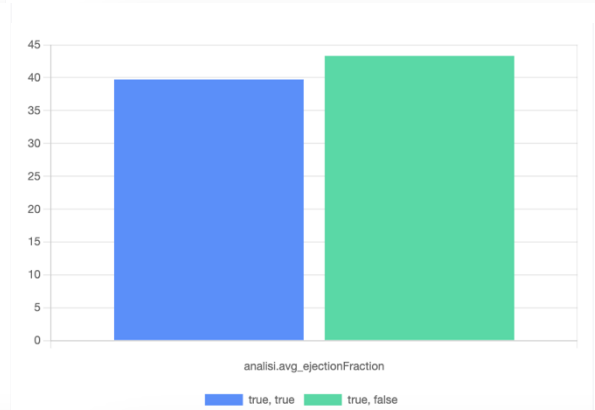












Insufficienza renale acuta

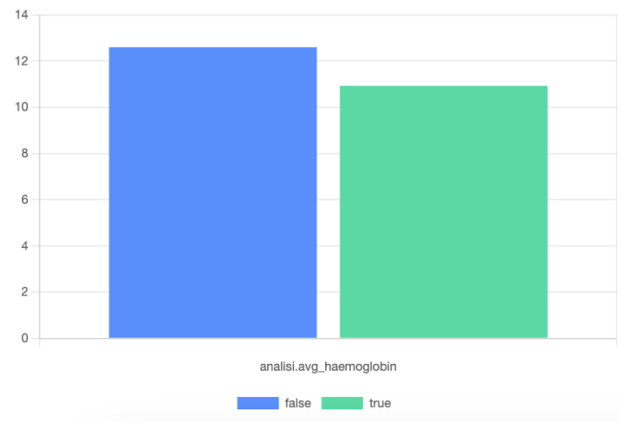
Visto che chi soffre di insufficienza renale acuta, ha la creatinina superiore ai valori normali, si è pensato di confrontare i valori medi delle analisi fra chi soffre di insufficienza renale ed un paziente sano.

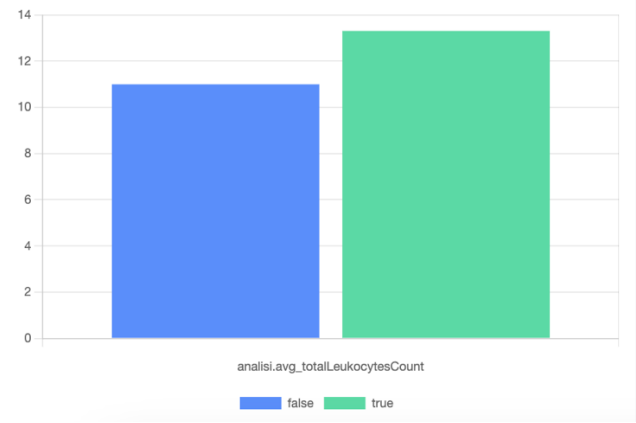
Leggenda:

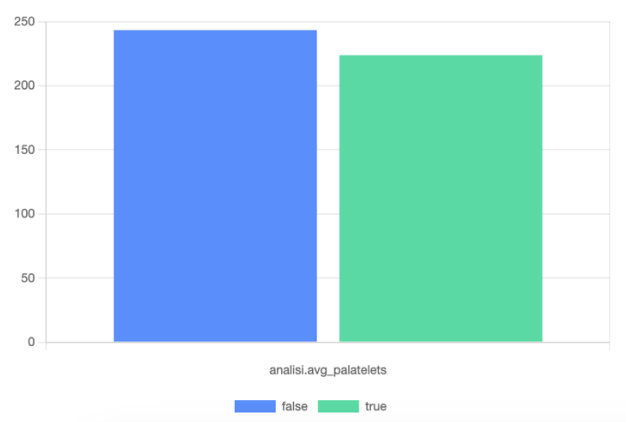
- **Blu**: Pazienti NON affetti da insufficienza renale acuta;
- **Verde**: Pazienti affetti da insufficienza renale acuta;

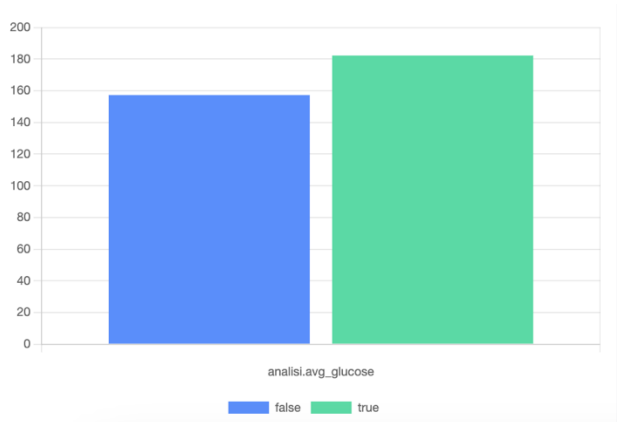

```
{
  "limit": 15000,
  "measures": [
    "analisi.avg_haemoglobin",
    "analisi.avg_totalLeukocytesCount",
    "analisi.avg_palatelets",
    "analisi.avg_glucose",
    "analisi.avg_urea",
    "analisi.avg_creatinine",
    "analisi.avg_bTypeNatriureticPeptide",
    "analisi.avg_ejectionFraction"
  ],
  "dimensions": [
    "ospedalizzazione.acute_kidney_injury"
  ],
  "order": {
    "ospedalizzazione.numero_osp": "desc"
  }
}
```

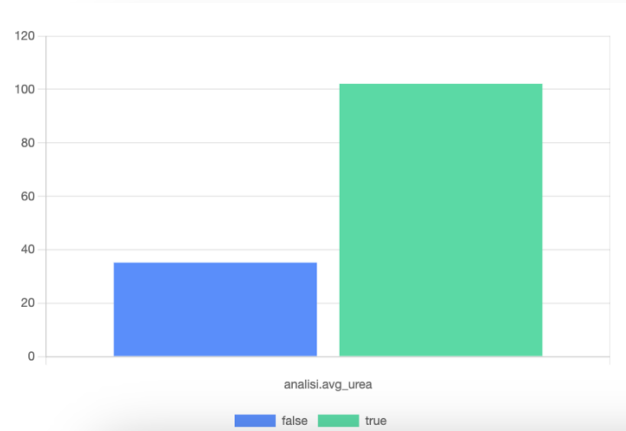
Si può notare come chi soffre di insufficienza renale, ha valori superiori di glucosio, urea, b type natruretic peptite [222%] e creatinine [300%]; mentre inferiori in piastrine e ejection infraction. Per una corretta visualizzazione delle differenze nelle analisi, verranno riportati i grafici per singolo caratterizzante preso in esame.

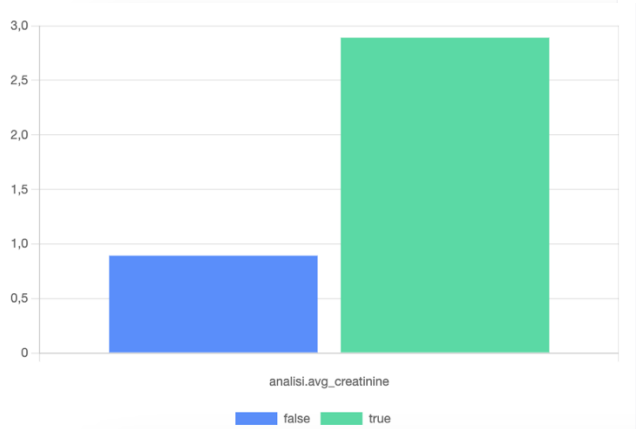


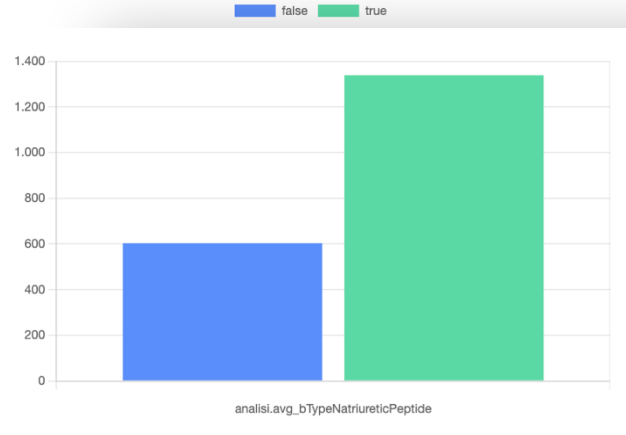


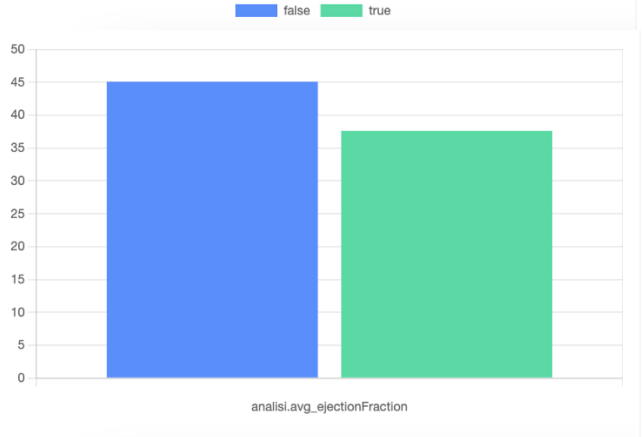










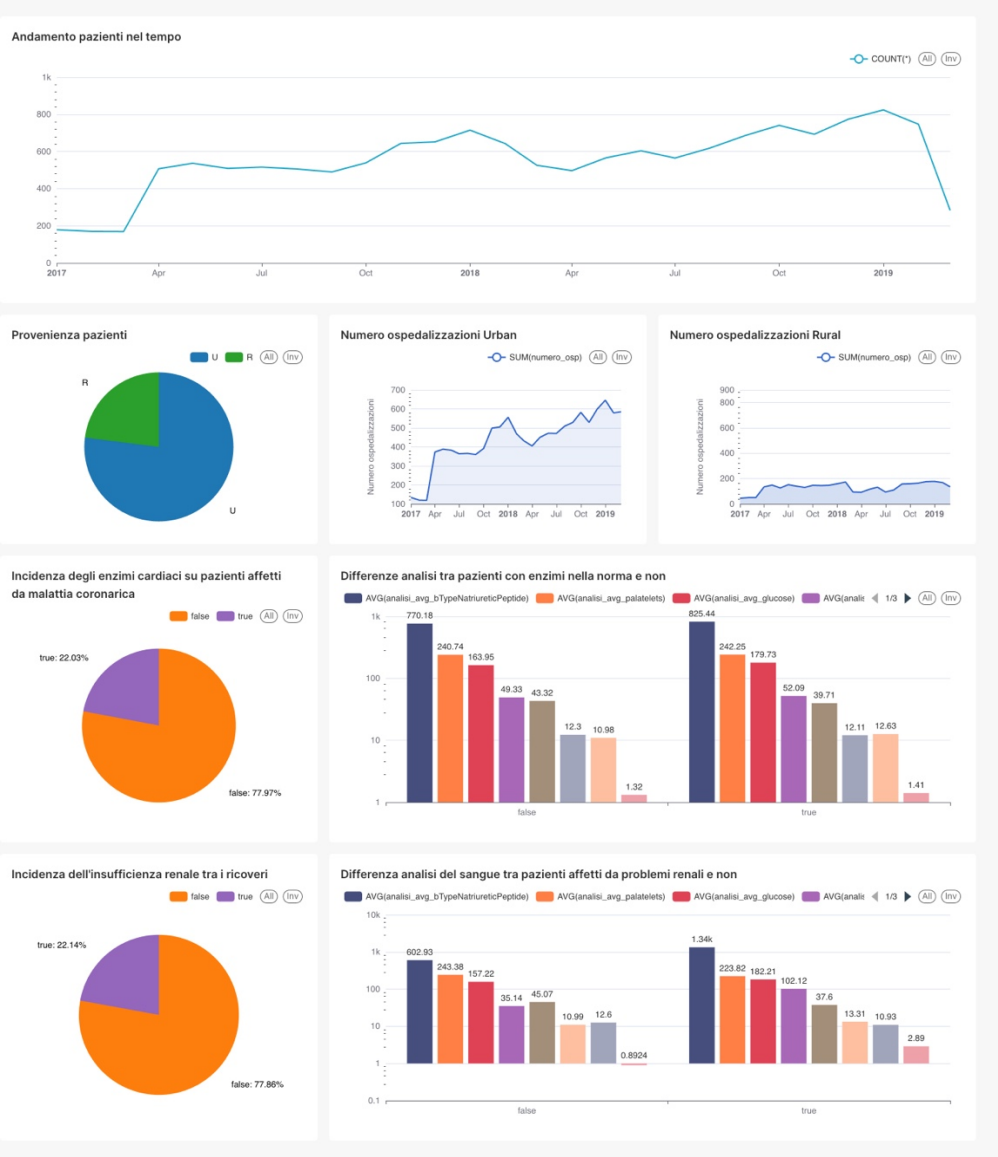


Superset

Per la parte finale del progetto, dedicata alla data visualization, si è deciso di utilizzare Apache Superset, una piattaforma open-source di data visualization e business intelligence progettata per semplificare l'analisi e la visualizzazione dei dati. Creato da Airbnb e successivamente donato alla Apache Software Foundation, Superset offre un'ampia gamma di funzionalità per esplorare, analizzare e condividere i dati in modo interattivo. Ecco alcuni punti chiave sull'Apache Superset che hanno portato a sceglierlo:

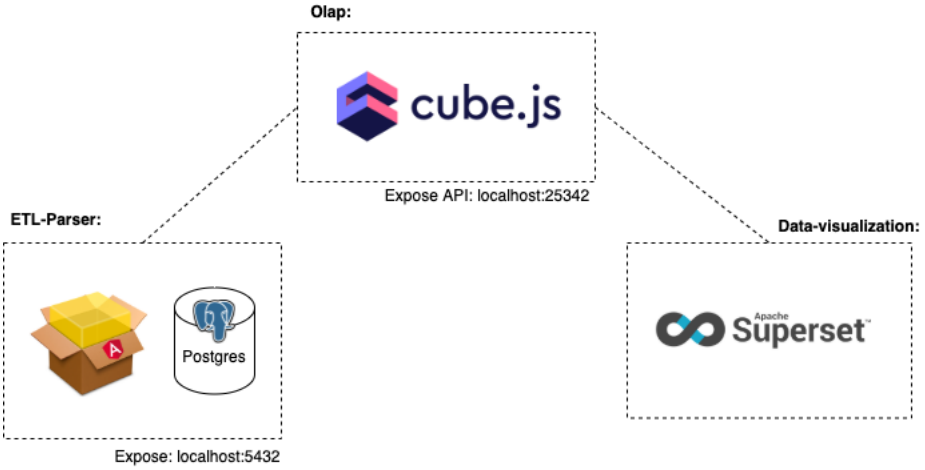
- **Connessioni dati versatili**: Superset può connettersi a una varietà di origini dati, tra cui database SQL, PostgreSQL, data warehouse e fonti di dati come Apache Druid, Google Sheets e Cube.js. Questa flessibilità consente agli utenti di accedere a dati provenienti da diverse fonti, in un unico ambiente.
- **Estensibilità**: Essendo un progetto open source, Superset è altamente estensibile e offre la possibilità di integrare nuove funzionalità e connessioni dati. La comunità attiva di sviluppatori contribuisce continuamente al miglioramento e all'espansione delle capacità di Superset.

Complessivamente, Apache Superset è una soluzione potente per le esigenze di data visualization e business intelligence, offrendo un modo flessibile, intuitivo e collaborativo per esplorare e comunicare i dati aziendali. Dopo la configurazione, è stato possibile effettuare le query precedentemente indicate ma in maniera più precisa e poter organizzare tutte le analisi in una dashboard consultabile in real time. Di seguito, viene riportato il risultato finale.



Configurazione progetto

Per mantenere un applicazione modulare, scalabile e a microservizi si è deciso di strutturare l'infrastruttura utilizzando Docker. Il risultato finale è un docker-compose per ogni modulo.



Per permettere la comunicazione tra i vari docker, va creata una rete bridge attraverso l'apposito comando:

```
docker network create nomeRete
```

Si può controllare se la rete è stata effettivamente creata attraverso l'apposito comando:

```
docker network ls
```

Per facilitare lo start-up e shut-down, sono stati creati 2 script bash che inizializzano i vari docker-compose:

- start.sh
- stop.sh

Ricordiamo che per lanciare i comandi bash, vanno eseguiti da terminale con l'apposito comando:

```
sudo bash nomescript.sh
```

Configurazione ETL

Configurazione application.properties:
Nel file "application.properties" vanno inserite tutte le variabili di ambiente per settare come datasource postgres, il dialetto di sql e le impostazioni del server per l'upload dei file in modalità multi-part. Di seguito verrà riportato tale configurazione:

```
# Database Properties
spring.datasource.url=jdbc:postgresql://DB:5432/postgres
spring.datasource.username=admin
spring.datasource.password=123
#Server setting
spring.servlet.multipart.max-file-size=10MB
spring.servlet.multipart.max-request-size=10MB
server.tomcat.threads.max=200
folderCount=0
# Hibernate Properties
# The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
# Hibernate ddl auto (create, create-drop, validate, update)
spring.jpa.hibernate.ddl-auto=update
```

La configurazione del compose, verrà approfondita nella sezione successiva.

Configurazione docker-compose

Il docker compose per il modulo ETL viene di seguito riportato:

```
version: '3'

services:
  app:
    build:
      context: .
    container_name: app
    ports:
      - "8080:8080"
    depends_on:
      - db
    volumes:
      - ./target/ETL-Parser-0.0.1-SNAPSHOT.jar:/app/app.jar
    networks:
      - retecondivisa
db:
    image: 'postgres:13.1-alpine'
    container_name: db
    ports:
      - "5432:5432"
    expose:
      - "5432:5432"
    environment:
      - POSTGRES_USER=admin
      - POSTGRES_PASSWORD=123
      - POSTGRES_DB=postgres
    volumes:
      - postgres:/var/lib/postgresql/data
    networks:
      - retecondivisa
networks:
    retecondivisa:
       driver: bridge
       external: true

volumes:
  postgres:
```

Configurazione OLAP

Per aggiungere Cube.js, basta creare un docker compose inserendo il seguente codice. Assicurandoci che la rete sia condivisa tra tutti i vari servizi e che quest'ultima sia in modalità bridge.

```
version: '3'

services:
  cube:
    image: cubejs/cube:latest
    ports:
      - 4000:4000
      - 15432:15432
      - 25432:25432
    expose:
      - 25432:25432
    environment:
      - CUBEJS_DEV_MODE=true
    networks:
      - retecondivisa
    volumes:
      - .:/cube/conf

networks:
  retecondivisa:
    driver: bridge
    external: true
```

Ricordiamo che Cube.js fungerà da ponte tra i nostri dati e Superset, quindi andranno configurate:

- La connessione postgres, per prelevare i dati normalizzati e creare i cubi e le view.
- Le API, da esporre per permettere a Superset di consultare i dati.

Di seguito verranno indicati tutti i passaggi per configurarlo in maniera ottimale.

Configurazione Datasource

La configurazione del datasource può essere fatta da GUI, quando avviamo la prima volta Cube.js, nella quale ci verrà chiesto di inserire i seguenti dati:

- Hostname
- Port
- Database
- Username
- Password

Qualora in seguito volessimo apportare delle modifiche, basta andare nel file ".env" dove sono presenti tutte queste informazioni.

```
CUBEJS_DB_HOST=DB
CUBEJS_DB_PORT=5432
CUBEJS_DB_NAME=postgres
CUBEJS_DB_USER=admin
CUBEJS_DB_PASS=123
CUBEJS_DB_TYPE=postgres
…
```

Per vedere le modifiche, basta aggiornare la pagina e non deployare di nuovo tutto il compose.

Configurazione API

Le API SQL sono disabilitate di default. Per abilitare l'API SQL, bisogna impostare CUBEJS_PG_SQL_PORT, un numero di porta a cui si desideri connettersi con uno strumento compatibile con Postgres. Di conseguenza nel file .env andranno aggiunte le seguenti informazioni.

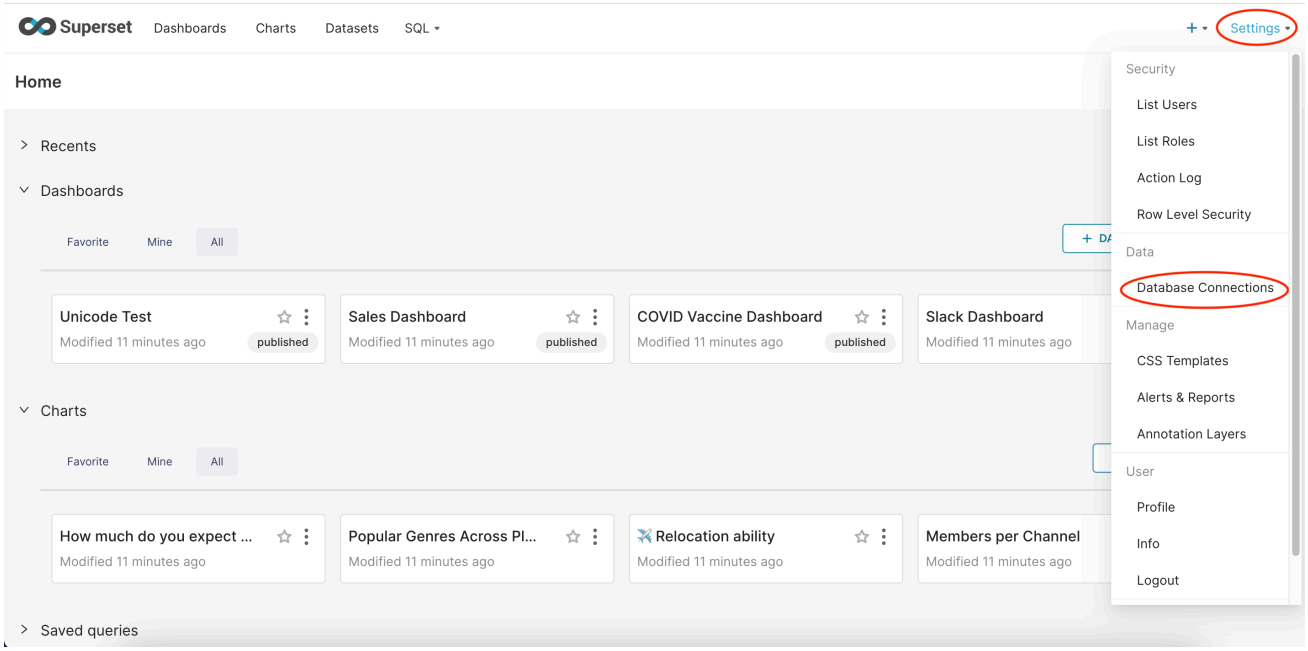
```
...
CUBEJS_EXTERNAL_DEFAULT=true
CUBEJS_SCHEDULED_REFRESH_DEFAULT=true
CUBEJS_DEV_MODE=true
CUBEJS_SCHEMA_PATH=model
CUBEJS_PG_SQL_PORT=25432
CUBEJS_SQL_USER=admin
CUBEJS_SQL_PASSWORD=123
```

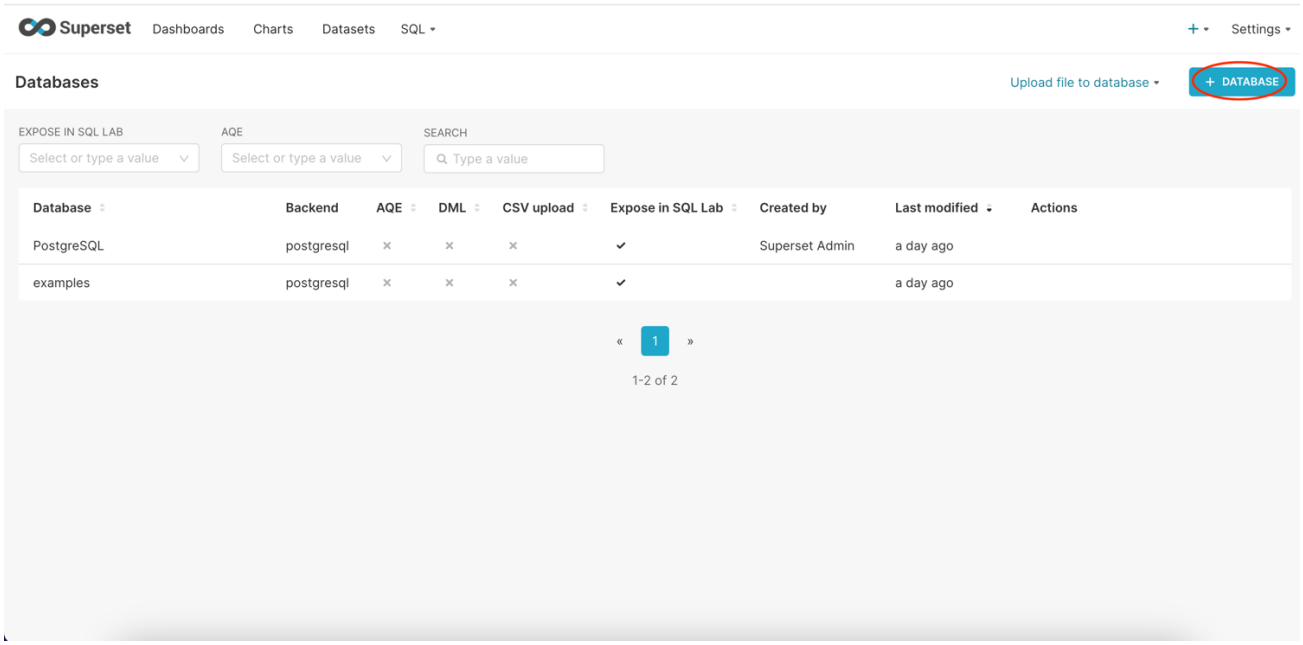
Configurazione Superset

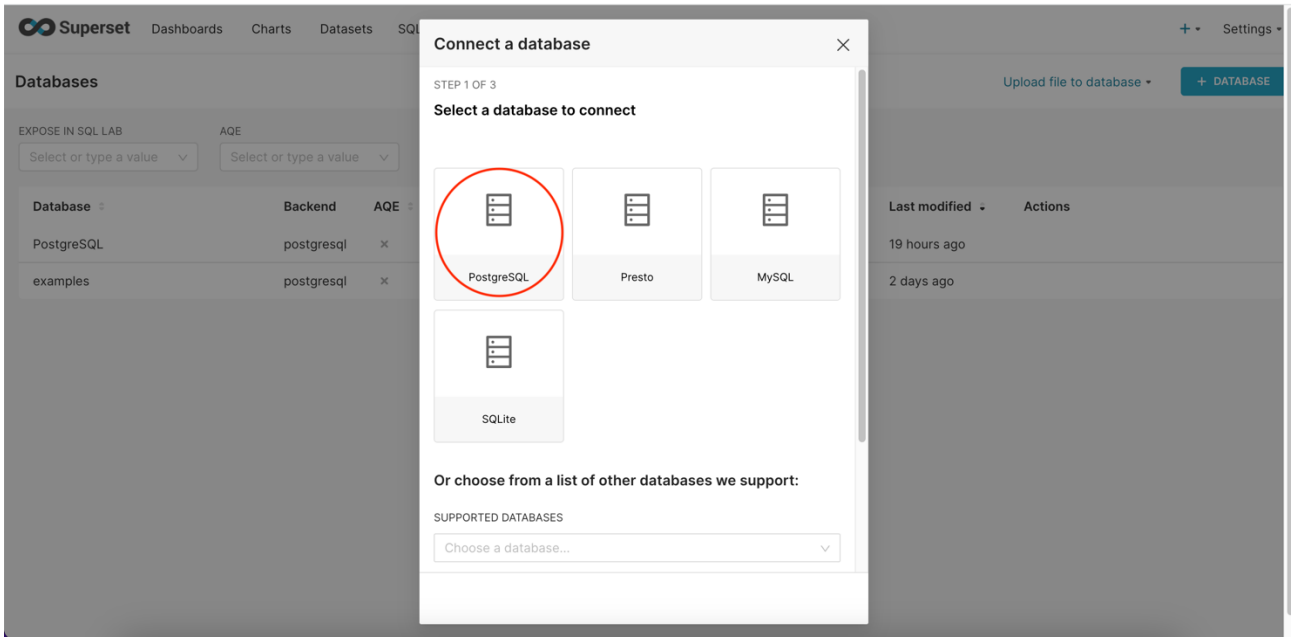
Dopo aver avviato il compose all'indirizzo localhost:8088, sarà disponibile Superset. Successivamente, andrà fatto l'accesso con le seguenti credenziali (di default)

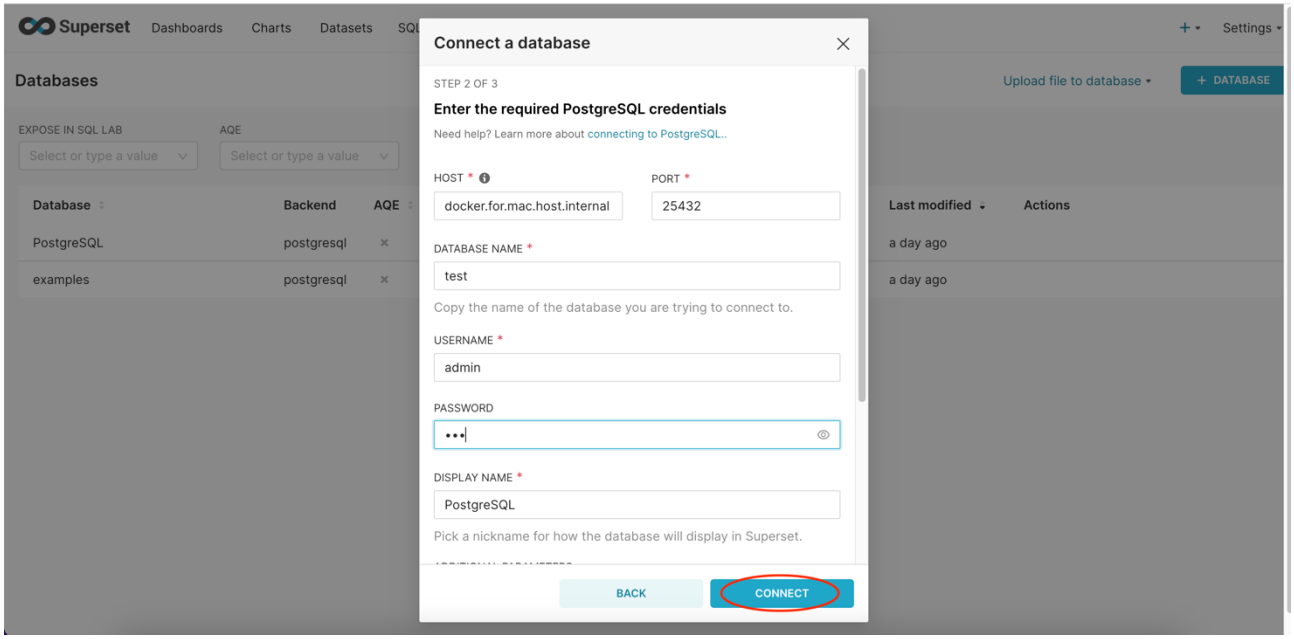
- username: admin
- password: admin

Configurazione database

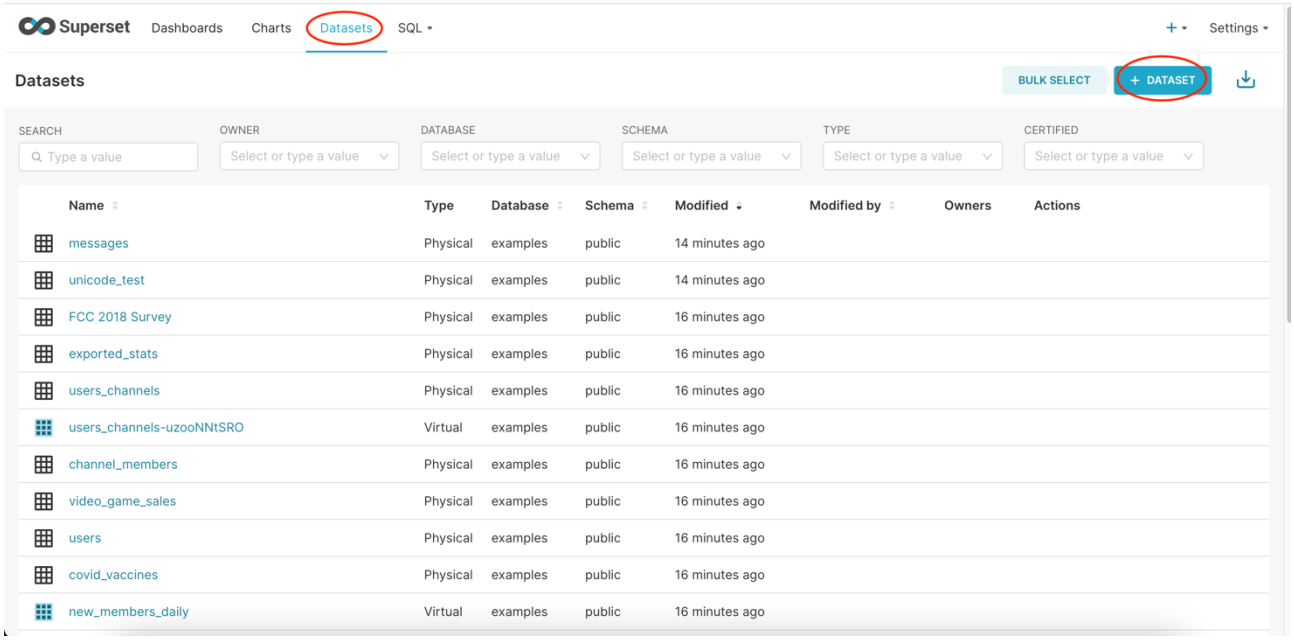


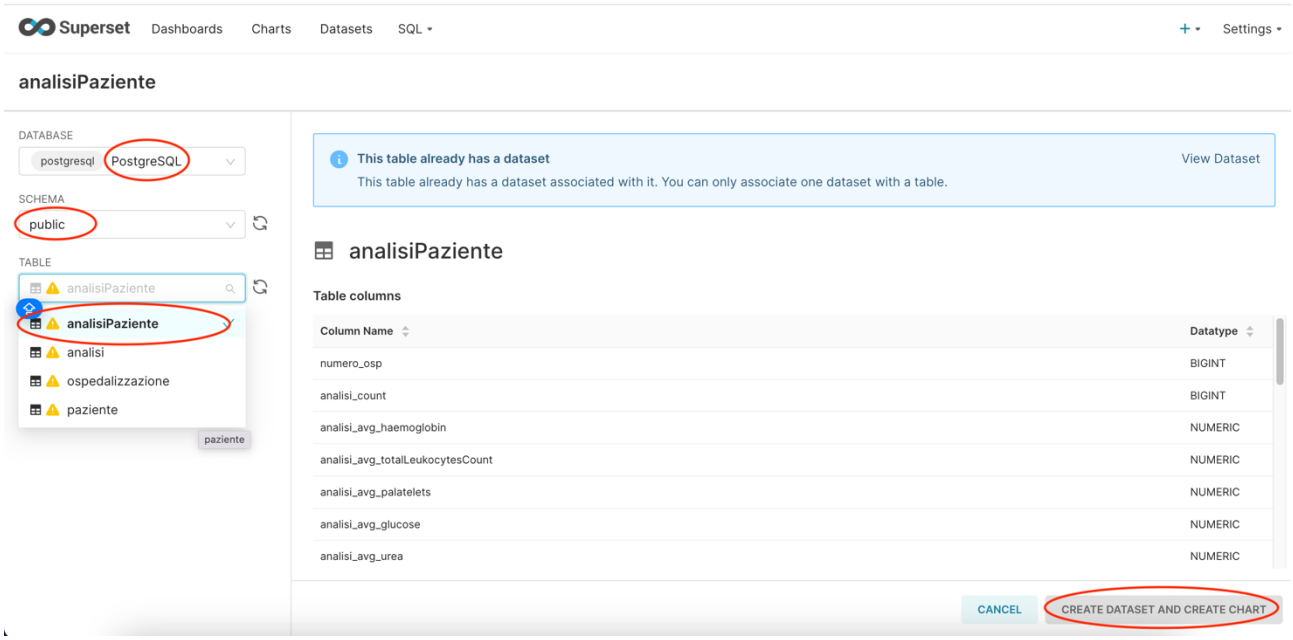



Superset
Dashboards
Charts
Datasets
Settings
Databases
Upload file to database
+ DATABASE
EXPOSE IN SQL LAB
AQE
Select or type a value
Database
Backend
AQE
Last modified
Actions
PostgreSQL
postgresql
19 hours ago
examples
postgresql
2 days ago
Connect a database
STEP 1 OF 3
Select a database to connect
PostgreSQL
Presto
MySQL
SQLite
Or choose from a list of other databases we support:
SUPPORTED DATABASES
Choose a database...


Superset
Dashboards
Charts
Datasets
Settings
Databases
Upload file to database
+ DATABASE
EXPOSE IN SQL LAB
AQE
Select or type a value
Database
Backend
AQE
Last modified
Actions
PostgreSQL
postgresql
a day ago
examples
postgresql
a day ago
Connect a database
STEP 2 OF 3
Enter the required PostgreSQL credentials
Need help? Learn more about connecting to PostgreSQL..
HOST *
PORT *
docker.for.mac.host.internal
25432
DATABASE NAME *
test
Copy the name of the database you are trying to connect to.
USERNAME *
admin
PASSWORD
DISPLAY NAME *
PostgreSQL
Pick a nickname for how the database will display in Superset.
BACK
CONNECT

Creazione Dataset





Creazione Grafico