Python Biella Group

Modern Data Engineering

Stato dell'arte e novità nel mondo dei dati

Concetti teorici ed esempi pratici

a cura di: **Davide Airaghi e Andrea Guzzo**

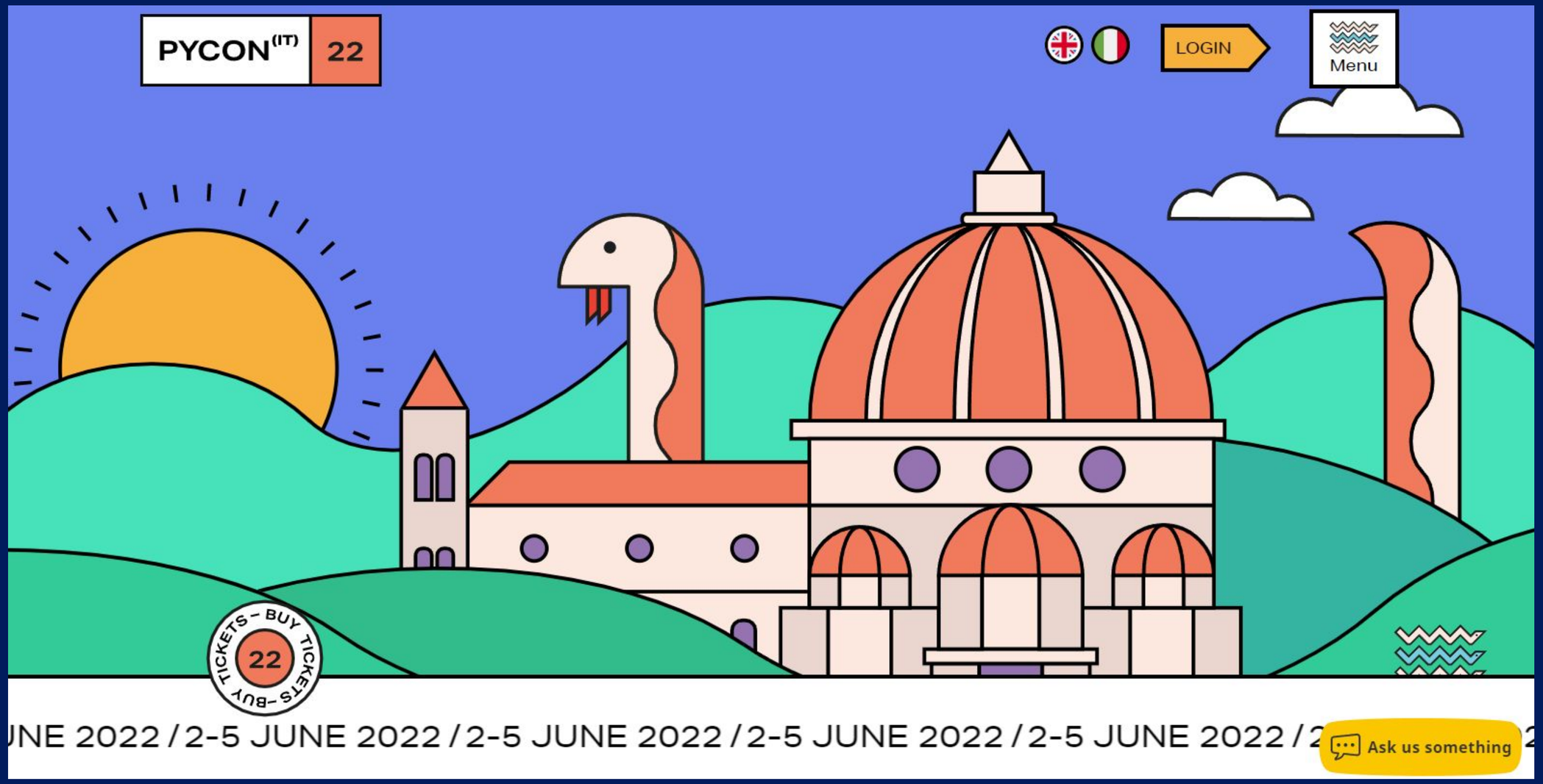
JOIN US!

C’è sempre qualcosa da imparare per migliorarci e crescere... insieme!

- **Nostri canale e materiale: https://linktr.ee/PythonBiellaGroup**

Tutto questo è stato reso possibile grazie a

- **Tutta la community di P.B.G.**
- **Maria Teresa Panunzio**: https://www.linkedin.com/in/maria-teresa-panunzio-27ba3815/
- **Mario Nardi**: https://www.linkedin.com/in/mario-nardi-017705100/
- **Andrea Guzzo**: https://www.linkedin.com/in/andreaguzzo/
- **Davide Airaghi:** https://www.linkedin.com/in/airaghidavide/


PYCON(IT) 22
LOGIN
Menu
BUY TICKETS – BUY TICKETS – BUY TICKETS –
22
UNE 2022 / 2-5 JUNE 2022 / 2-5 JUNE 2022 / 2-5 JUNE 2022 / 2-5 JUNE 2022 /
Ask us something

INIZIAMO!

https://github.com/PythonBiellaGroup/ModernDataEngineering

Python Biella Group

Obiettivo delle serate

Raccontare e descrivere il mondo dell'ingegnerizzazione dei dati

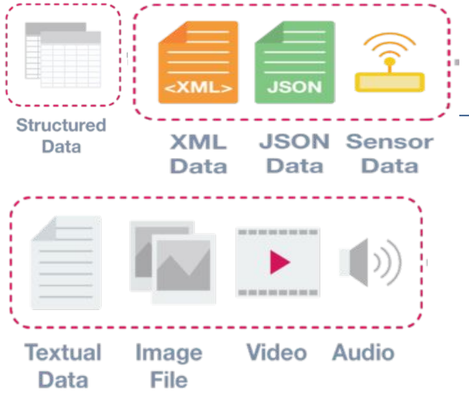
- Aspetti teorici generali
- Semplici esempi con airflow
- Progetto di esempio

Data Engineering

- E' il processo di raccolta, trasformazione, integrazione e convalida dei dati per l'analisi. Per le organizzazioni che vogliono seguire un approccio data driven come fondamento delle loro decisioni questo processo é estremamente importante per consentire ai fruitori dell'informazione di avere il dato:
  - sempre disponibile
  - coerente, affidabile e preciso (in qualità)
  - idoneo per le attività di analisi (dalla BI al ML)
- Elementi:
  - Identificazione delle sorgenti del dato necessario alle esigenze di analisi
  - Trasformazione e messa in qualità del dato (ETL/ELT)
  - Progettazione basi di dati idonee alla fruizione del dato
  - Progettazione di pipeline di elaborazione scalabili,sicure, monitorabili e schedulate

Processo implementativo

1 - Collecting



- Individuazione fonti informative
- Gestire le relazioni con i fornitori del dato
- Valutare se il contenuto informativo del dato rispecchia le necessità di analisi
- Identificare le modalità di ingestion del dato

2 - Pipelining

- Identificazioni trasformazioni,integrazioni e logiche di business da applicare
- Definizione schedulazione pipeline
- Definizione error handling e alerting
- Progettazione basi di dati

3 - Consuming

- Identificazione tempistiche refresh del dato
- Orchestrazione con tool di AI
- Supporto attività Data Scientist/ Data Analyst

Trasformazione del dato - Data cleaning

- Una delle attività più importanti é la messa in qualità del dato grezzo. Questa é un'operazione imprescindibile poiché condizionerà l'analisi e (quindi le decisioni) dell'intera organizzazione (GIGO)

- Comprende:
  - Gestione valori mancanti (NULL)
  - Gestione formati (yyyy-mm-dd)/conversioni (valuta)/formati stringa(uppercasing)
  - Tipizzazione corretta
  - Correttezza di dominio (codice nazione)
  - Unicità di valori nel dataset
  - Ricerca duplicati
  - Valori anomali - outliers

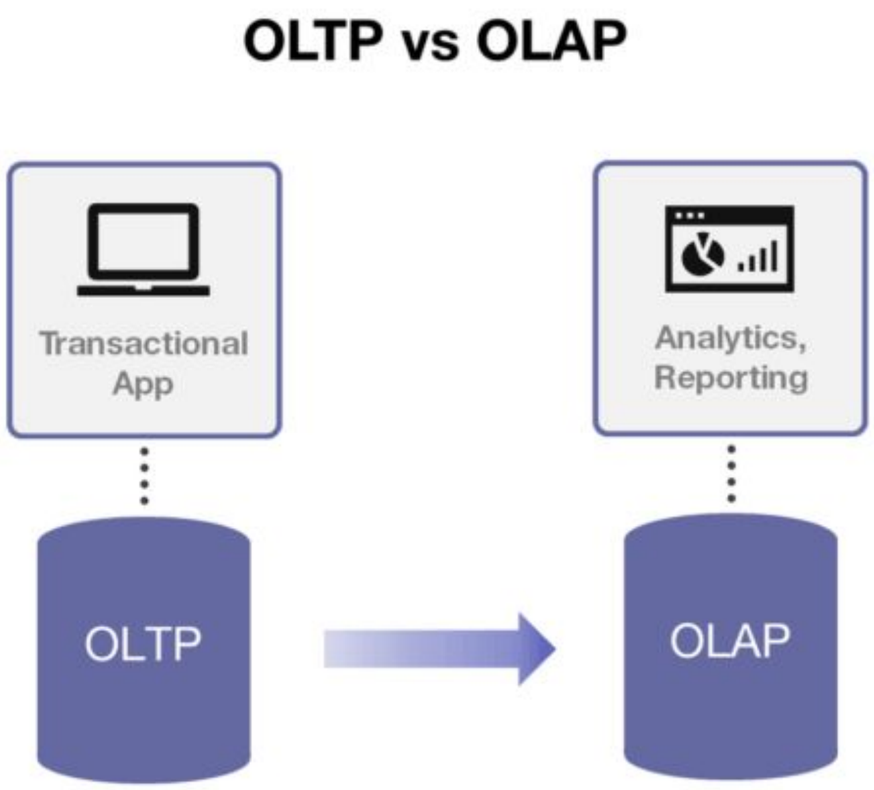
Trasformazione del dato - Data wrangling

- Un’altra attività importante é l'arricchimento e l’allineamento dell’informazione che stiamo trattando
- Comprende:
  - Individuazione sub set informativo rilevante
  - Data enrichment:
    - Campi calcolati (p.e un dato numerico lo moltiplico per un certo fattore)
    - Aggiunta di informazioni (p.e ad un indirizzo aggiungo il CAP)
    - Decodifica dei valori (BI → Biella)
  - Aggregazioni (in caso di dato grezzo estremamente granulare)
  - Normalizzazione

OLTP vs OLAP

OLTP:

- dato transazionale
- normalizzato
- lettura del dato diretta e poco complessa. Si legge il dato corrente
- Operazioni di inserimento, cancellazione e scrittura
- Importante per il monitoraggio delle normali attività aziendali

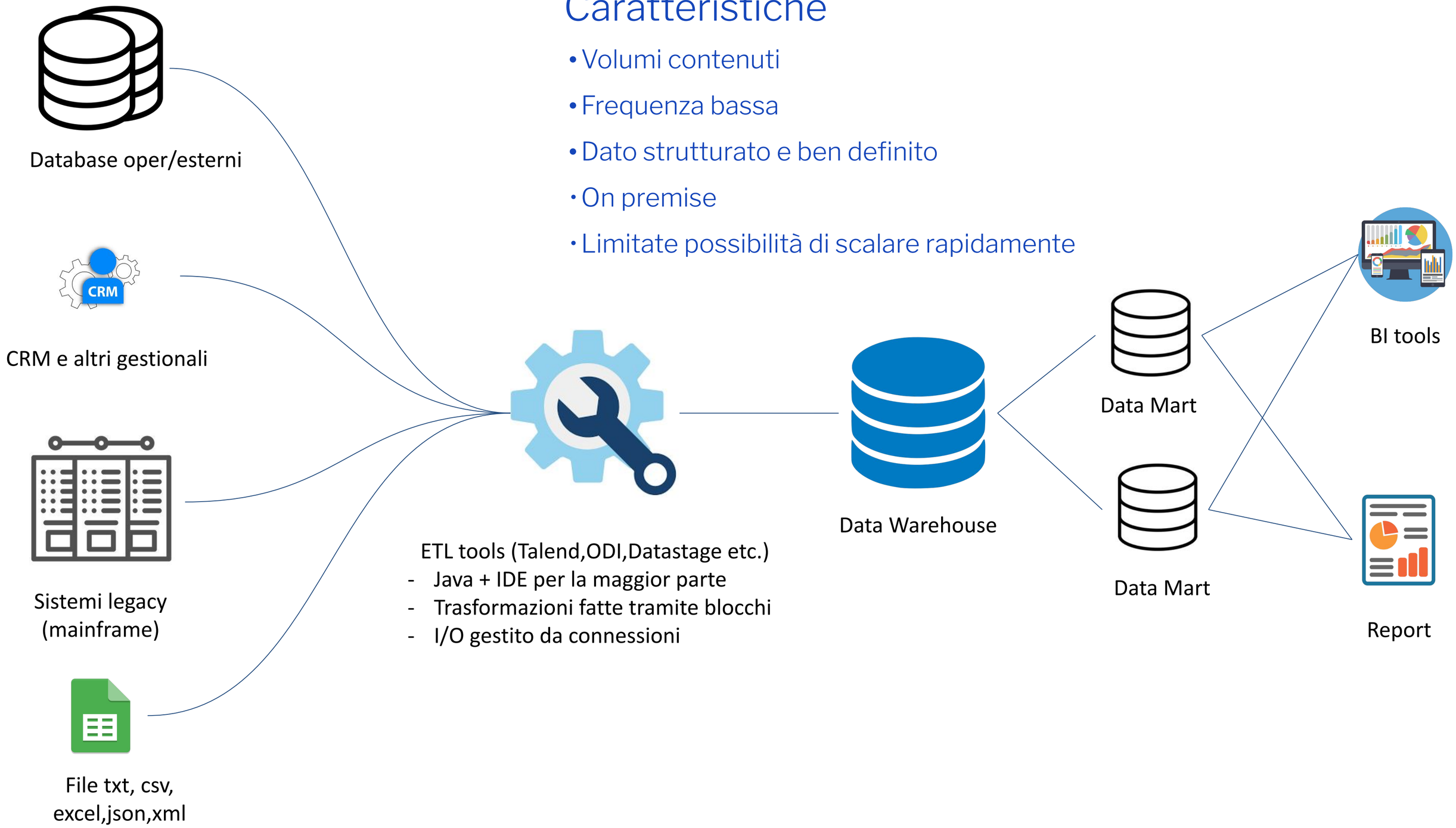


OLAP:

- dato analitico e con profondità storica
- denormalizzato
- lettura del dato molto complessa.
- Operazioni di aggregazione su una mole considerevole di dati
- Importante per la pianificazione e il processo decisionale

Caratteristiche

- Volumi contenuti
- Frequenza bassa
- Dato strutturato e ben definito
- On premise
- Limitate possibilità di scalare rapidamente

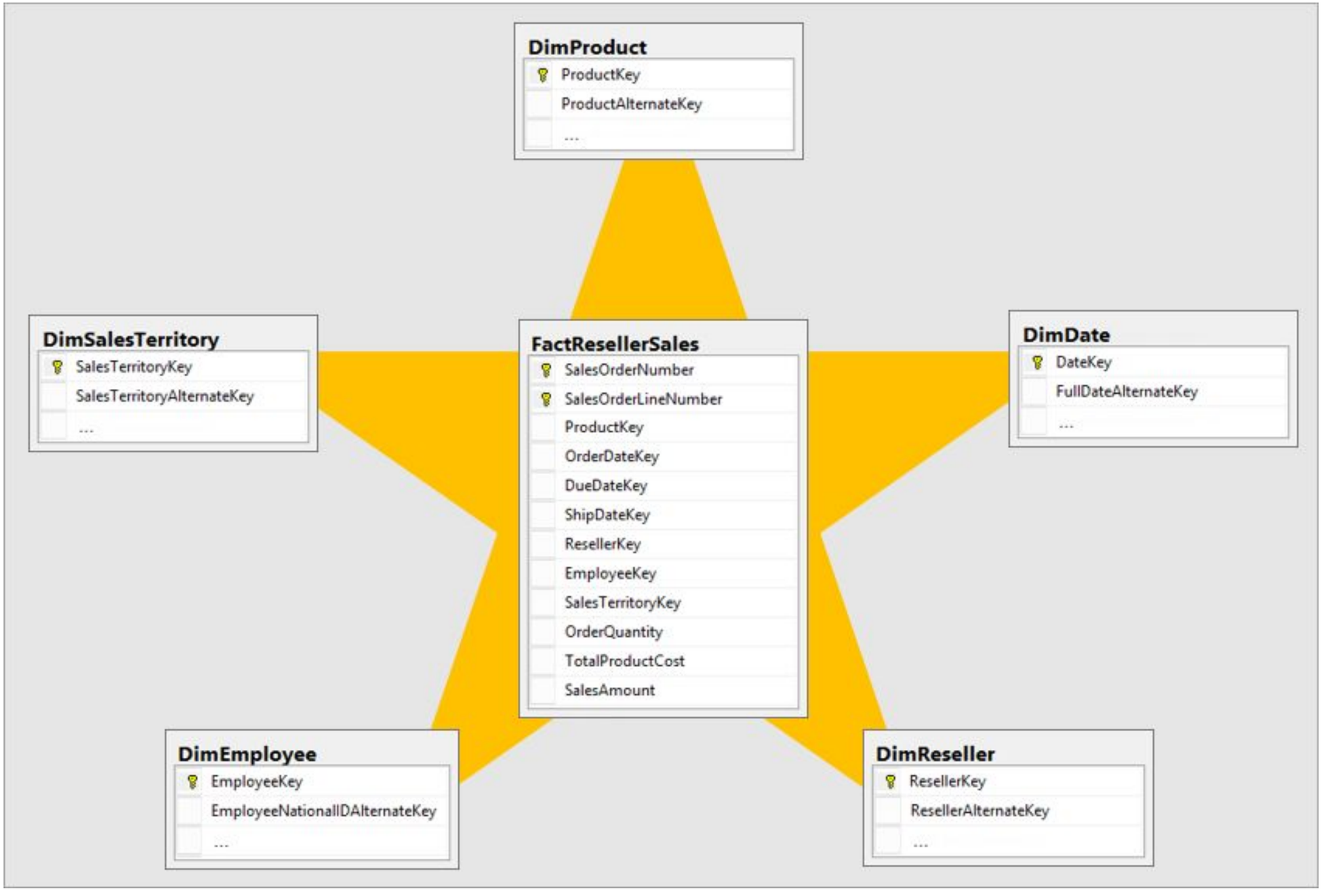


Data Warehouse

- Caratteristiche
  - Orientato agli oggetti: favorisce la possibilità di analisi su particolari topic
  - Integrazione: permette l’integrazione e l'omogeneità del dato ottenuto da diverse sorgenti sia interne che esterne all’organizzazione
  - Non volatilità: garantisce la permanenza del dato acquisito
  - Storicizzazione: permette la storicizzazione del dato e la conseguente analisi temporale
- Modello dato: utilizza lo **star schema** come modello

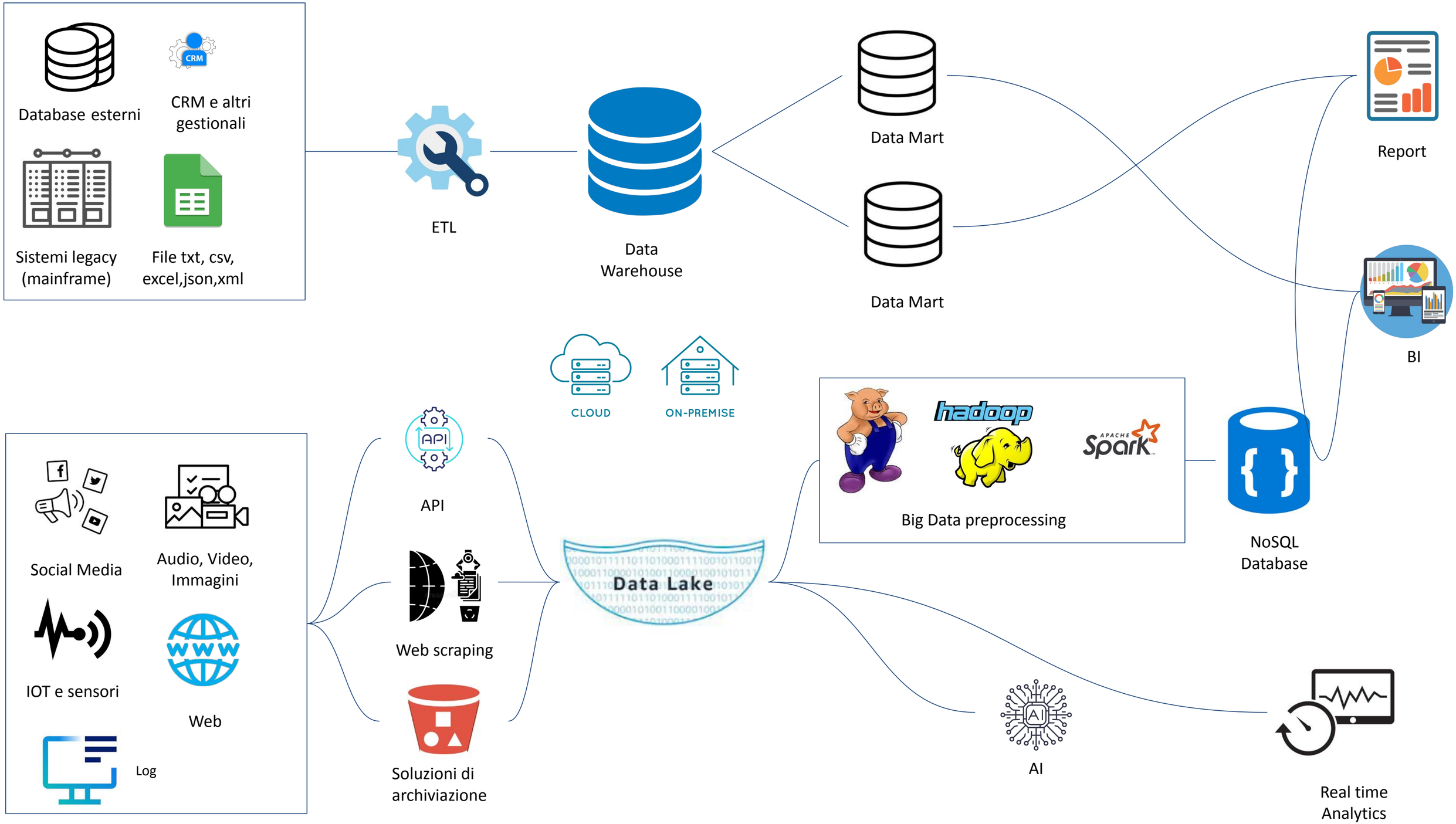
Dimensional modeling - Star schema

- E' un approccio di modellazione del dato molto utilizzato all'interno dei sistemi di BI e DataWarehouse.
- Struttura:
  - Fact table: contiene tipicamente eventi di business (transazioni,vendite etc)
  - Dimension tables: contengono informazioni collegate agli eventi di business sulle quali si potrà improntare l'analisi
- Possibilità offerte:
  - Struttura ottimizzata per interrogazioni OLAP quindi ideale per aggregazioni e filtri
  - Cammino di join semplice anche per utenti meno esperti
  - Facile individuazione di indici che velocizzino l'accesso al dato


DimProduct
ProductKey
ProductAlternateKey
...
DimSalesTerritory
SalesTerritoryKey
SalesTerritoryAlternateKey
...
FactResellerSales
SalesOrderNumber
SalesOrderLineNumber
ProductKey
OrderDateKey
DueDateKey
ShipDateKey
ResellerKey
EmployeeKey
SalesTerritoryKey
OrderQuantity
TotalProductCost
SalesAmount
DimDate
DateKey
FullDateAlternateKey
...
DimEmployee
EmployeeKey
EmployeeNationalIDAlternateKey
...
DimReseller
ResellerKey
ResellerAlternateKey
...

- Volumi elevati (p.e attività web utenti)
- Frequenza elevata (p.e sensori)
- Dato non strutturato e strutturato
    - File csv,json,xml,txt
    - Immagini, audio e video
    - Social media - testo
- Ambiente di lavoro on cloud e on premise
- Nuove modalità di processamento ed ingestion (aka più ambienti e linguaggi da gestire)
- Nuove esigenze di analisi
    - Real time business intelligence
    - Machine learning
    - Process mining

Come evolve l’ architettura?

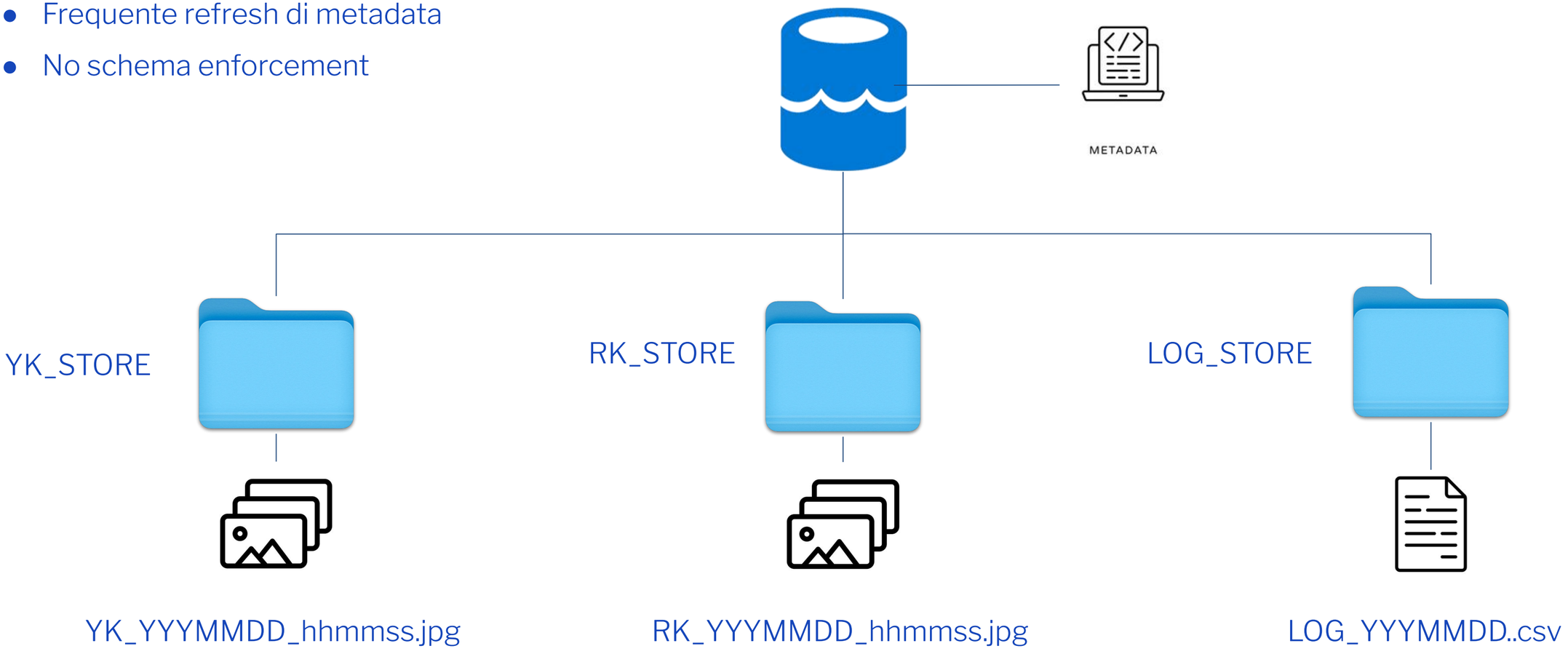

Database esterni
CRM
CRM e altri gestionali
Sistemi legacy (mainframe)
File txt, csv, excel,json,xml
ETL
Data Warehouse
Data Mart
Data Mart
Report
BI
CLOUD
ON-PREMISE
Social Media
Audio, Video, Immagini
IOT e sensori
Web
Log
API
Web scraping
Soluzioni di archiviazione
Data Lake
hadoop
APACHE Spark
Big Data preprocessing
NoSQL Database
AI
Real time Analytics

Data Lake

- Caratteristiche
  - Acquisisce dati in formato sia strutturato che non strutturato
  - Memorizza e protegge i dati in maniera scalabile senza (idealmente) vincoli
  - Indicizza e cataloga i dati in modo da consentirne la fruizione senza alcuna movimentazione
  - Permette la fruizione del dato a sistemi di ML e di analisi
- ELT: il paradigma di processamento del dato é in questo caso ELT (Extract Load Transform)
- Governance: rischio di “data swamp”

Data Lake - struttura e problematiche

- File di dimensioni disparate non consentono un corretto partizionamento
- Dato inconsistente
- Democratizzazione totale ma scarsa conoscenza di dominio
- Problemi di indicizzazione
- Frequente refresh di metadata
- No schema enforcement

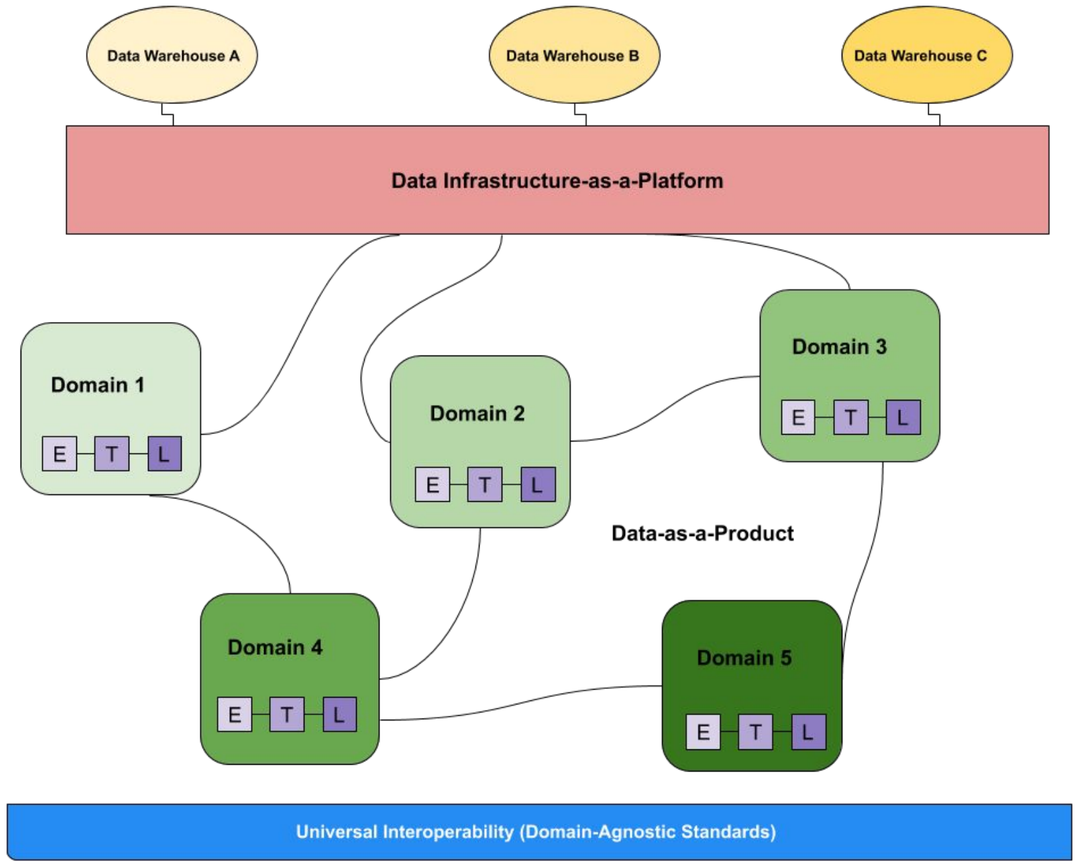


Data Mesh

Domain Driven Design:

- Ogni dominio (CRM,Gestionale, etc) specifico é in grado di comprendere, analizzare e mettere in qualità il dato di sua pertinenza
- Ogni dominio é **responsabile** del dato che mette a disposizione.
- Ogni dominio crea, sviluppa e mantiene dei **data product** che vengono fruiti da altri domini come sorgente informativa e di analisi

  Il paradigma “premia” i domini più utili e più sottoscritti dagli altri con una certa allocazione di budget
- Viene superato il monolite del Data Lake e ancor prima del DataWarehouse
- E’ ora favorita la democratizzazione del dato tramite data product realizzati con il principio **DATSIS**



Data Mesh - DATSIS

- **Discoverable** – i consumatori devono essere autonomi nella ricerca e nell'individuazione dei data product realizzati dai differenti domini
- **Addressable** – i data product devono essere identificati con un nome univoco seguendo una nomenclatura comune
- **Trustworthy** – i data product devono essere realizzati dai domini che detengono i dati e devono mettere a disposizione dati di qualità
- **Self-describing** – i consumatori devono poter usare i dati esposti dai data product senza doverne chiedere il significato agli esperti di dominio
- **Integrated** – i data product devono essere realizzati seguendo degli standard condivisi per essere facilmente integrati nella architettura e riusati per realizzare altri data product
- **Secure** – i data product devono essere sicuri by design e l'accesso al dato deve essere regolamentato a livello centrale tramite politiche standard di sicurezza

Apache Airflow - orchestrazione pipelines

- Permette di creare, monitorare e schedulare pipeline
- Gli elementi di una pipeline sono:
  - Task: attività/ azione da eseguire
  - Dipendenze: quali task dipendono da un certo task
  - Schedulazione: condizioni e tempistiche di esecuzione
- Le pipeline sono definite come DAG (Directed Acyclic Graphs)
  - Directed: i task di una pipeline sono sempre diretti in modo da garantire una chiaro path di esecuzione
  - Acyclic: non ci possono essere task che puntano a un task che é già stato eseguito in precedenza. In questo modo evitiamo la possiiblità di esecuzioni infinite.
  - Graphs: i flussi sono definiti come dei grafi ovvero ogni nodo del grafo rappresenta un task e i rami lo collegano ad altri task definendo dipendenze

Python Biella Group

Modern Data Engineering - 2

Stato dell'arte e novità nel mondo dei dati

Concetti teorici ed esempi pratici

a cura di: **Davide Airaghi e Andrea Guzzo**

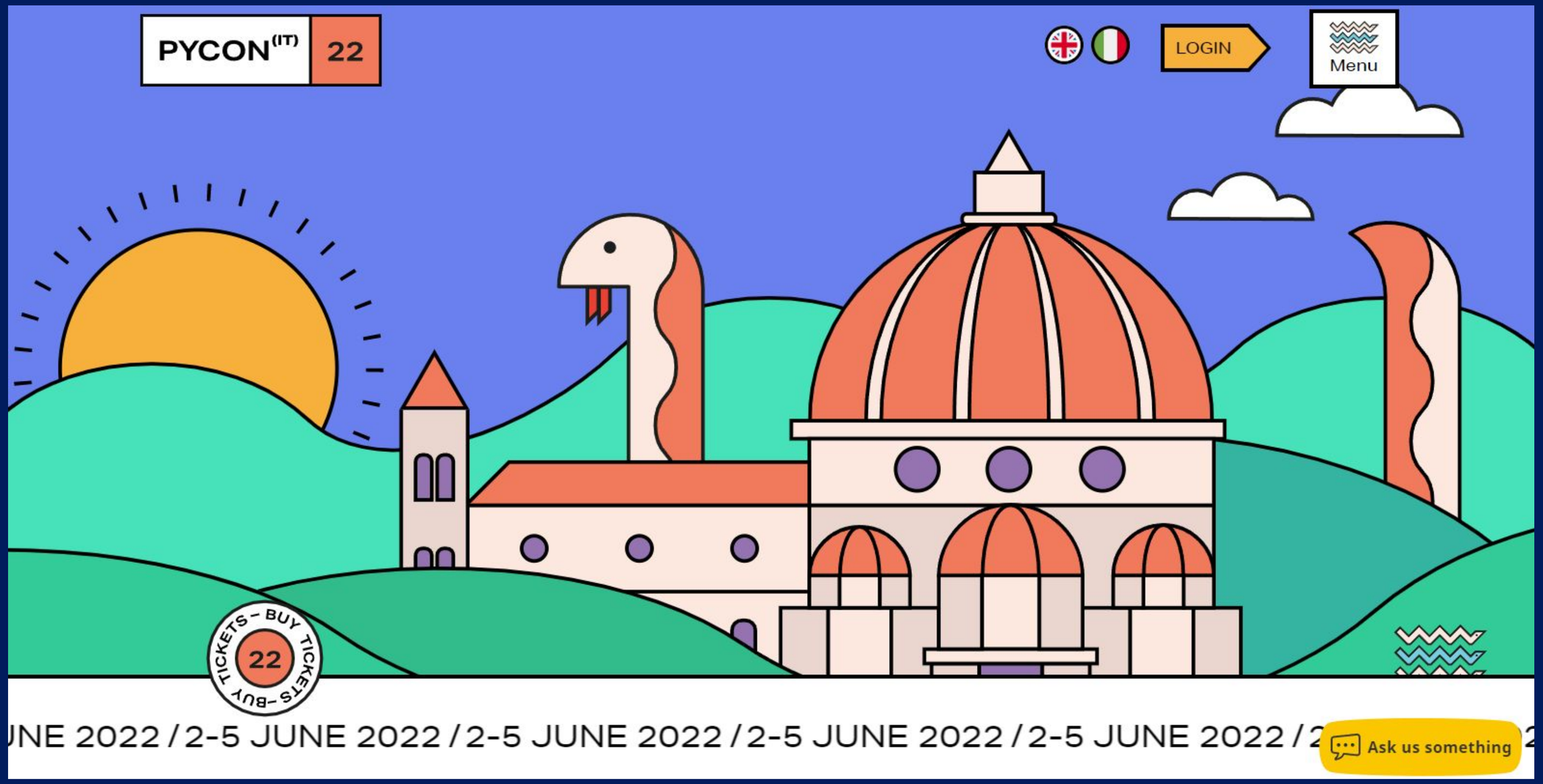
JOIN US!

C’è sempre qualcosa da imparare per migliorarci e crescere... insieme!

- **Nostri canale e materiale: https://linktr.ee/PythonBiellaGroup**

Tutto questo è stato reso possibile grazie a

- **Tutta la community di P.B.G.**
- **Maria Teresa Panunzio**: https://www.linkedin.com/in/maria-teresa-panunzio-27ba3815/
- **Mario Nardi**: https://www.linkedin.com/in/mario-nardi-017705100/
- **Andrea Guzzo**: https://www.linkedin.com/in/andreaguzzo/
- **Davide Airaghi:** https://www.linkedin.com/in/airaghidavide/


PYCON(IT) 22
LOGIN
Menu
BUY TICKETS – BUY TICKETS – BUY TICKETS –
22
UNE 2022 / 2-5 JUNE 2022 / 2-5 JUNE 2022 / 2-5 JUNE 2022 / 2-5 JUNE 2022 /
Ask us something

INIZIAMO!

https://github.com/PythonBiellaGroup/ModernDataEngineering

Python Biella Group

Agenda

01 Introduzione - Open Data

02 Presentazione della soluzione

03 Concetti chiave Apache AirFlow

04 Concetti chiave Streamlit

Materiale e codice sorgente

https://github.com/PythonBiellaGroup/

Open Data

- Rendere dati e informazioni delle PA aperti e accessibili online, in assenza di forme di controllo e restrizioni, come copyright e brevetti, che ne limitano l'integrazione e il riutilizzo

- Interoperabilità: il loro valore aumenta sensibilmente quando data set differenti, prodotti e pubblicati in modo indipendente da diversi soggetti, possono essere incrociati liberamente da terze parti

- Non basta averli liberati, servono collaborazione tra PA, aziende e cittadini nel creare nuove app e servizi

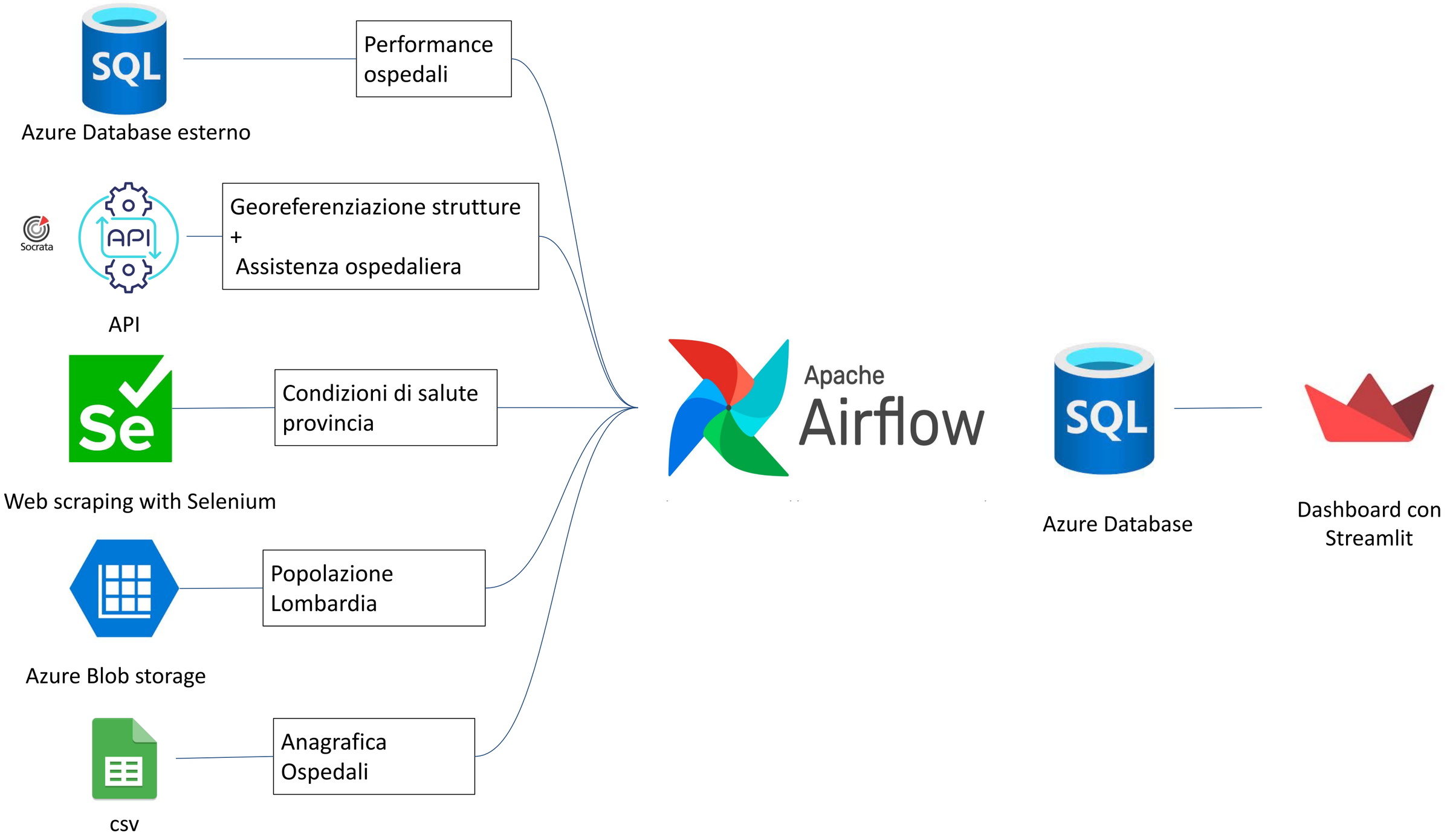
- Possono essere propulsori economici per la creazione e nascita di start up, e posti di lavoro

- Occorre avere un linguaggio comune, una semantica con dati strutturati e chiavi di lettura univoche

Dati

Tutti i dati sono stati raccolti dal sito **Open Data** di regione Lombardia (https://dati.lombardia.it)

- Performance ospedali (ricoveri ripetuti, ritorni in sala operatoria, etc)
- Georeferenziazione strutture (coordinate geografiche strutture)
- Condizioni di salute per provincia (aspettativa di vita,tasso mortalità, etc)
- Popolazione Lombardia (dati demografici)
- Anagrafica ospedali


Azure Database esterno
Performance ospedali
Socrata
API
Georeferenziazione strutture + Assistenza ospedaliera
Web scraping with Selenium
Condizioni di salute provincia
Azure Blob storage
Popolazione Lombardia
csv
Anagrafica Ospedali
Apache Airflow
Azure Database
Dashboard con Streamlit

Apache AirFlow

Riferimenti:

- Installazione in locale o su docker
- Monitoring tramite GUI
- DAG: dynamic acyclic graph
- Operatori:
  - BashOperator: esegue comandi bash
  - PythonOperator: esegue una funzione python
  - EmailOperator: permette l'invio di mail
  - SimpleHttpOperator: invia una richiesta http
  - Sensor: attende il verificarsi di un evento (p.e ricezione di un file)
- Scheduler: processo di airflow che attiva le pipeline

Streamlit

Riferimenti:

- Sviluppo di data app/ML app
- Nessuna competenza di sviluppo frontend: API già pronte all' uso per realizzare la vostra APP.
- Grafici con le librerie più famose: plotly,altair,vega,bokeh,etc.
- Cloud CI/CD pipeline
- Caching
- Input widgets
- Ora parte di Snowflake!

PRONTI?