Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica per il management

BIG DATA E DATA WAREHOUSE PER SUPPORTO ALLE DECISIONI IN AMBITO AZIENDALE: UN CASO DI STUDIO

Relatore:
Chiar.mo Prof.
MARCO DI FELICE

Presentata da:
GIOVANNI VITALE

Correlatore:
Chiar.mo Tutor
RAFFAELE GREZZI

Sessione III
Anno Accademico 2019/2020

Alla mia famiglia, che mi è stata e mi sarà sempre vicina.

Indice

Parte I

Introduzione

Introduzione alla tesi

La mia tesi analizza ed illustra un progetto svolto durante il tirocinio in azienda, comprendendo la teoria correlata; il progetto consiste nell'esecuzione di una migrazione tecnologica di una dashboard per un cliente del settore Manufacturing. Per migrazione tecnologica si intende copiare tabelle e viste, legate alla dashboard per cui è richiesto il progetto, da database di origine di una tecnologia (in questo caso Netezza) a database di destinazione (in questo caso Hana), convertendo il loro linguaggio di definizione SQL. Inoltre, in queste operazioni sono comprese le attività di migrazione dei flussi di dati che alimentano queste tabelle e viste, utilizzando i flussi di origine e modificandoli con le tabelle e viste che vengono migrate precedentemente.
L'azienda ospitante si chiama Iconsulting S.p.A; Iconsulting è un System Integrator indipendente, specializzato nella progettazione di sistemi best-in-class di Data Warehouse, Business Intelligence, Performance Management e Big Data. Essa è nata da un gruppo di ricercatori ed oggi è partner strategico dei venditori tecnologici di riferimento: IBM, Microsoft, Microstrategy, Oracle, SAP e SAS. Iconsulting ha sede a Bologna, Roma, Milano e Londra ed ha all'attivo più di 500 progetti di successo su un portfolio di oltre 150 clienti di elevato standing.
La tesi è caratterizzata da 6 parti: *Introduzione, Stato dell'arte, Progetto, Conclusione e sviluppi futuri, Bibliografia e Sitografia* e, infine, *Ringraziamenti.*
La sezione dedicata alla *Introduzione* è la parte che serve a spiegare ciò che è stato fatto durante il processo di elaborazione della tesi, descrivendo in breve il contenuto dei vari capitoli che la compongono.
La sezione dedicata allo *Stato dell'arte* è la parte che approfondisce la teoria legata al progetto della tesi; questa sezione è composta essenzialmente da tre capitoli collegati fra loro da alcuni nessi logici. Il primo capitolo introduce l'argomento base, fulcro del progetto e della teoria, ovvero i *Big Data*; in particolare tratta della loro definizione, della loro importanza in vari casi d'uso e delle infrastrutture che li ospitano. Di seguito il primo capitolo si collega ad una delle più importanti infrastrutture che ospitano i Big Data, ossia i *Data Warehouse*; il secondo capitolo tratta della loro definizione, della differenza con sistemi di database operativi, della loro architettura multi livello, della loro modellazione e dei loro modelli. Infine, il secondo capitolo lascia spazio ad un argomento attuale, legato all'obiettivo del progetto elaborato, ovvero *IBM NETEZZA VS SAP*

HANA; quest'ultimo capitolo di questa parte tratta delle definizioni delle due tecnologie citate e della motivazione della migrazione dalla tecnologia Netezza alla tecnologia Hana, obiettivo finale del progetto di tesi.
La sezione dedicata al *Progetto* è la parte che serve ad esporre tutto ciò che riguarda il progetto che è stato elaborato; questa sezione è composta da tre capitoli: *Progettazione, Implementazione* e *Validazione*. Il capitolo di *Progettazione* illustra informazioni preliminari del progetto quali i requisiti, gli obiettivi finali, le metodologie utilizzate nella sua elaborazione e l'architettura del processo utilizzato. Il capitolo di *Implementazione* tratta, invece, degli strumenti utilizzati, la loro definizione e il loro impiego durante la progettazione, e delle operazioni svolte, volte al raggiungimento del prodotto finale. Infine, il capitolo di *Validazione* serve a mostrare[1] le varie schermate del prodotto finale ottenuto e a esplicare una loro funzionalità all'interno della dashboard.
La sezione dedicata alla *Conclusione e sviluppi futuri* è la parte conclusiva che descrive alcuni possibili approfondimenti riguardanti il progetto ultimato.
La sezione dedicata alla *Bibliografia e Sitografia* mostra le fonti da cui sono state apprese informazioni o foto, utili per la stesura della tesi.

[1]N.B. Le foto utilizzate per mostrare il prodotto finale sono prive di dati sensibili e, a volte, arricchite con dati fittizi per tutelare il cliente che ha richiesto il progetto.

Parte II

Stato dell'arte

Capitolo 1

Big Data

Al giorno d'oggi si sente sempre più spesso parlare dei Big Data e del loro significato nella rappresentazione di un'evoluzione per diversi settori; Big Data è un termine che descrive un grande volume di dati, strutturati e non strutturati, che ottiene un'azienda ogni giorno [15]. Essi devono essere, poi, analizzati alla ricerca di informazioni di valore che portino a decisioni aziendali migliori e a mosse strategiche di business. È importante sottolineare che lo stesso termine Big Data è ambiguo; la sua traduzione *'grandi dati'* o *'grossi dati'*, infatti, fa pensare ad un'enorme quantità di dati di oggi disponibili in diversi settori e, in automatico, porta a concludere che per rivoluzione Big Data si intendono le opportunità oggi disponibili di avere così tante informazioni al servizio business. Questa conclusione, però, è vera solo in minima parte perché esistono settori dove i dati, per quanto ve ne sia un'enorme quantità, non sono sempre disponibili a tutti e, soprattutto, non vengono sempre condivisi. La vera 'rivoluzione' a cui ci si riferisce parlando di Big Data è la capacità di usare queste informazioni per elaborare, analizzare e trovare riscontri oggettivi su diverse tematiche.

1.1 Definizione di Big Data

Il termine ***'Big Data'*** si riferisce a dati informatici grandi, veloci o complessi, difficili o impossibili da elaborare con i metodi tradizionali. Il concetto di 'Big Data' ha acquisito uno slancio solo all'inizio degli anni 2000, quando l'analista di mercato Doug Laney ha articolato l'attuale definizione di Big Data come le 'tre V':

- **Volume:** oggi, le organizzazioni raccolgono dati da diverse fonti, tra cui transazioni commerciali, dispositivi intelligenti (IoT), apparecchiature industriali, video, social media, ecc. Con volume si fa riferimento, quindi, a questa enorme massa di informazioni che non è possibile raccogliere con tecnologie tradizionali;

- **Velocità:** con la crescita di *Internet of Things*, ovvero Internet delle Cose, i flussi di dati verso le imprese devono essere gestiti in modo tempestivo e a una velocità senza precedenti. Con velocità si intende, quindi, alla rapidità con cui nascono e vengono acquisiti i dati;

- **Varietà:** con questo termine si fa riferimento alle differenti tipologie di dati oggi disponibili, provenienti da un numero crescente di fonti eterogenee, quali sistemi transazionali e gestionali aziendali, sensori, social network, open data. Vi possono essere dati disponibili in tutti i tipi di formati, dai dati strutturati e numerici, ai documenti di testo non strutturati, e-mail, video, audio, dati di stock e transazioni finanziarie.

Con il passare degli anni sono state definite due ulteriori caratteristiche, arricchendo il modello delle 'tre V':

- **Veridicità:** con questo termine ci si riferisce alla qualità dei dati. Poiché i dati provengono da così tante fonti diverse, è difficile collegare, abbinare, pulire e trasformare i dati tra i sistemi. La loro qualità e la loro integrità è un pilastro molto importante per effettuare analisi che siano utili e affidabili;

- **Variabilità:** oltre a crescere in velocità e varietà, i flussi di dati sono imprevedibili, per cui cambiano spesso e variano di continuo. La mutevolezza del loro significato è un aspetto da tenere in considerazione nel momento in cui i dati vengono interpretati.

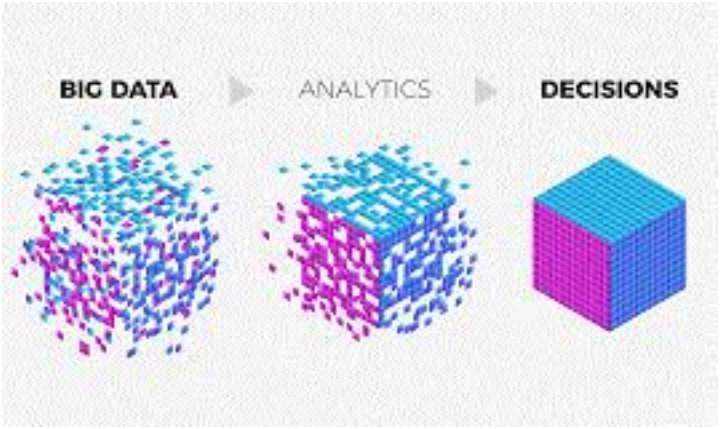
Infine vi è un'ultima caratteristica sviluppata negli ultimi anni, data l'importanza che ha acquisito l'elemento dei Big Data, che si va ad aggiungere alle altre ed inizia sempre per la 'V':

- **Valore:** con questo termine si intende il valore che può essere estratto da un'azienda, ovvero prendere decisioni con più informazioni, rapide e consapevoli, dai Big Data attraverso fondamentali metodologie di Big Data Analytics.

1.2 Importanza e casi d'uso dei Big Data

Come già detto precedentemente, l'importanza dei Big Data non ruota intorno alla loro quantità ma al loro utilizzo. Ad esempio, è possibile accedere ai dati provenienti da qualsiasi fonte, analizzarli e trovare risposte che consentano di:

- Ridurre i costi;
- Accorciare le tempistiche;
- Sviluppare nuovi prodotti;
- Ottimizzare le offerte;
- Prendere decisioni più smart.



Con la combinazione dei Big Data e dei Big Data Analytics, infatti, si possono ottenere risultati di business. Di seguito, alcuni esempi di casi d'uso:

- **Customer analytics:** le aziende possono analizzare il comportamento dei consumatori in ottica di marketing multicanale per migliorare l'esperienza del cliente, aumentare i tassi di conversione, le vendite collaterali, offrire servizi e aumentare la fidelizzazione;
- **Analytics operazionale:** molte organizzazioni hanno l'obiettivo di migliorare le prestazioni operative e fare un uso migliore degli asset aziendali. I Big Data possono aiutare le imprese a trovare nuovi modi per operare in modo più efficiente;
- **Prevenzione delle frodi e dei crimini:** le aziende possono individuare attività sospette attraverso il riconoscimento di pattern che possano indicare un comportamento fraudolento, prevenendone il manifestarsi o individuando il colpevole;

- **Ottimizzazione dei prezzi:** le aziende possono usare i dati per ottimizzare i prezzi applicati a prodotti e servizi, espandendo il proprio mercato o aumentando i ricavi.

1.3 Infrastrutture di archiviazione per i Big Data

Affiché un progetto Big Data possa avere successo, le aziende hanno bisogno di dedicare a questo carico di lavoro un'infrastruttura adeguata e spesso molto specifica, in grado di raccogliere, archiviare ed elaborare i dati per presentarli in una forma utile. Il tutto garantendo la sicurezza delle informazioni mentre sono archiviate e in transito.
Tra le opzioni di archiviazione più usate in ambito Big Data vi sono:

- **Data warehouse:** i tradizionali sistemi su cui le applicazioni aziendali registrano i propri dati, dall'ERP[1] al CRM[2], possono ovviamente costituire una delle fonti da cui le applicazioni Big Data attingono le informazioni;

- **Data lake:** repository di informazioni in grado di contenere volumi di dati estremamente grandi nel loro formato nativo, almeno fino al momento in cui è necessario effettuare elaborazioni e ricavare informazioni per le applicazioni di business. In quel caso, e solo a quel punto, i sistemi Big Data si occuperanno di estrarre da quei dati le informazioni richieste;

- **Cloud storage:** è un modello di conservazione dati su computer in rete dove i dati stessi sono memorizzati su molteplici server virtuali generalmente ospitati presso strutture di terze parti o su server dedicati. Esso consiste nell'astrazione, nell'aggregazione e nella condivisione delle risorse di storage attraverso la rete internet.

Nel Capitolo 2 viene descritto dettagliatamente l'argomento riguardante i data warehouse poiché durante il tirocinio svolto ho lavorato nel campo dei data warehouse.

[1]ERP: acronimo che sta ad indicare le parole inglesi 'Enterprise Resource Planning', ovvero pianificazione delle risorse aziendali; è un termine che identifica i software progettati per assistere e automatizzare diversi processi aziendali critici, tra cui principalmente: contabilità, gestione delle risorse umane, vendite, inventario, produzione logistica e distribuzione.

[2]CRM: acronimo che sta ad indicare le parole inglesi 'Customer Relationship Management', ovvero gestione delle relazioni con i clienti; è un processo strategico che consente di comprendere meglio le esigenze dei clienti e come soddisfare tali esigenze per migliorare il tasso di ritenzione dei clienti, soddisfare le esigenze di supporto, aumentare le vendite e aumentare i profitti.

Capitolo 2

Data warehouse

I data warehouse generalizzano e consolidano i dati in uno spazio multidimensionale. L'implementazione dei data warehouse comprende operazioni di *pulizia dei dati*, *integrazione dei dati* e *trasformazione dei dati*, fasi iniziali importanti per il data mining. Inoltre, i data warehouse forniscono strumenti **OLAP** (*Online Analytical Processing*, ovvero un insieme di tecniche software per l'analisi interattiva e veloce di grandi quantità di dati [2].

2.1 Definizione di data warehouse

Il **data warehousing** fornisce architetture e strumenti per le esecuzioni di business per organizzare, comprendere e utilizzare sistematicamente i loro dati per prendere decisioni strategiche. Negli anni il termine *'data warehouse'* ha avuto diverse definizioni, per cui è difficile fornularne una in modo dettagliato. In generale, un data warehouse si riferisce a un *'repository'* di dati mantenuto separatamente dai database operativi di un'organizzazione. I sistemi del data warehouse permettono un'integrazione di una varietà di sistemi

di applicazione; essi supportano l'elaborazione delle informazioni fornendo una piattaforma solida di dati storici consolidati per le analisi. Secondo un leader nella costruzione dei sistemi di data warehouse:

> *"A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process"*
> -William H. Inmon [1][2]

Con questa definizione Inomn dichiara che il data warehouse è una raccolta di dati orientata al soggetto, integrata, basata sul tempo e non volatile a supporto del processo decisionale nella gestione aziendale.
Vi sono quattro parole chiave: orientato al soggetto, integrato, variante-tempo e non volatile.

- **Orientato al soggetto:** un data warehouse è organizzato intorno a soggetti quali il cliente, il fornitore, il prodotto e le vendite. Si focalizza sulla modellazione e l'analisi dei dati per coloro che dovranno prendere delle decisioni, piuttosto che occuparsi delle operazioni quotidiane e sull'elaborazione delle transazioni dell'organizzazione.

- **Integrato:** un data warehouse viene di solito costruito integrando più fonti eterogenee, come database relazionali, flat file e record di transazioni online.

- **Variante-Tempo:** i dati sono conservati per fornire informazioni in una prospettiva storica (ad esempio 10-15 anni fa). Ogni struttura del data warehouse contiene elementi *tempo*, sia implicito che esplicito.

- **Non volatile:** un data warehouse è un archivio di dati mantenuto fisicamente separato, trasformato dall'applicazione dei dati trovati nell'ambiente operativo. Per questo, esso non ha bisogno di un processo di transazione, di ripristino e dei meccanismi di controllo della concorrenza; esso richiede solo due operazioni di accesso ai dati: caricamento iniziale dei dati e accesso ai dati.

Molte organizzazioni usano le informazioni provenienti dai data warehouse per sostenere le attività decisionali di business, includendo:

1. Aumentare la focalizzazione sul cliente e, quindi, sull'analisi del suo modello di acquisto (ad esempio: la preferenza di acquisto, il tempo di acquisto, i cicli di budget e l'appetito per la spesa);

[1]William H. Inmon: William (Bill) H. Inmon (San Diego, 20 luglio 1945) è un informatico statunitense, considerato il padre del data warehouse ed è il creatore della Corporate Information Factory. Inmon è conosciuto per i suoi seminari sullo sviluppo dei data warehouse e come speaker di molte industrie di sviluppo di software e di calcolo.

2. Riposizionare i prodotti e gestire i portafogli prodotti confrontando l'andamento delle vendite per trimestre, per anno e per aree geografiche al fine di definire delle strategie di produzione;

3. Analisi delle operazioni e ricerca di risorse di profitto;

4. Dirigere le relazioni con i clienti, apportare correzioni ambientali e gestire il costo delle risorse aziendali;

2.2 Differenze tra sistemi di database operativi e data warehouse

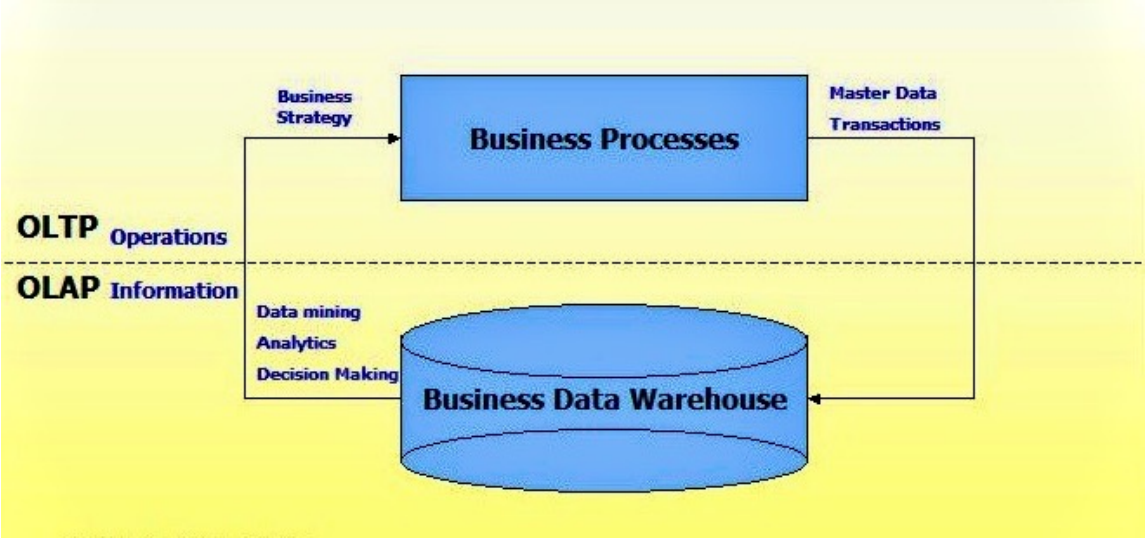


Figura 2.1: Sistemi OLTP VS Sistemi OLAP

Il compito principale dei sistemi di database operativi online è quello di eseguire transazioni online e l'elaborazione di query. Questi sistemi sono chiamati processi di transazioni online **OLTP** (*Online Transaction Processing*); essi coprono la maggior parte delle operazioni giornaliere di un'organizzazione come la vendita, l'inventario, produzione, banca, buste paga, registrazione e contabilità. I sistemi data warehouse vengono utilizzati dai vari utenti al fine di analizzare i dati e di prendere decisioni. Tali sistemi possono organizzare e presentare i dati in vari formati al fine di soddisfare le diverse esigenze dei diversi utenti; questi sistemi sono noti come sistemi **OLAP** (Online Analytical Processing). Le differenze fra sistemi OLTP e sistemi OLAP sono:

- **Utenti e orientamento al sistema:** un sistema OLTP è orientato al cliente, usato per transazioni ed elaborazioni di query da impiegati, clienti e professionisti

informatici. Un sistema OLAP è orientato al mercato, utilizzato, invece, per l'analisi dei dati da *"lavoratori della conoscenza"*, incluso i manager, i dirigenti e gli analisti.

- **Contenuto dei dati:** un sistema OLTP tratta di dati che sono troppo dettagliati per essere utilizzati facilmente per prendere delle decisioni. Un sistema OLAP tratta di una grande mole di dati storici, fornisce servizi di riepilogo e aggregazione e archivia e gestisce le informazioni a diversi livelli di dettaglio; ciò rende più facile le informazioni per prendere delle decisioni.

- **Design del database:** un sistema OLTP adotta un modello ER[2] (*Entity-Relationship*) e un design del database orientato all'applicazione. Un sistema OLAP può adottare o un modello *STAR* o un modello *SNOWFLAKE* e un design di database orientato al soggetto.

- **Vista:** un sistema OLTP si concentra principalmente sui dati correnti di un'impresa o di un dipartimento senza riferirsi a dati storici o ai dati di diverse organizzazioni. Un sistema OLAP spesso comprende più versioni di uno schema di database, a causa del processo evolutivo di un'organizzazione; inoltre, essi si occupano anche di informazioni che provengono da organizzazioni diverse, integrando informazioni da molti archivi di dati.

- **Modelli di accesso:** i modelli di accesso di un sistema OLTP consistono principalmente in transazioni brevi e atomiche; tale sistema richiede un controllo di concorrenza e meccanismi di ripristino. I modelli di accesso di un sistema OLAP sono per lo più operazioni di sola lettura.

Nella Figura 2.2 vi è il riepilogo di tutte le differenze tra i due sistemi.

[2]Modello ER: In informatica, nell'ambito della progettazione dei database, il modello ER (in italiano: modello entità-associazione o modello entità-relazione) è un modello teorico per la rappresentazione concettuale e grafica dei dati ad un alto livello di astrazione, formalizzato dal professore Peter Chen nel 1976. Esso viene spesso utilizzato nella prima fase della progettazione di una base di dati, nella quale è necessario tradurre le informazioni ottenute dall'analisi di un determinato dominio in uno schema concettuale, chiamato schema E-R (schema entità-associazione) o diagramma E-R (diagramma entità-associazione).

Feature	OLTP	OLAP
Characteristic	operational processing	informational processing
Orientation	transaction	analysis
User	clerk, DBA, database professional	knowledge worker (e.g., manager, executive, analyst)
Function	day-to-day operations	long-term informational requirements decision support
DB design	ER-based, application-oriented	star/snowflake, subject-oriented
Data	current, guaranteed up-to-date	historic, accuracy maintained over time
Summarization	primitive, highly detailed	summarized, consolidated
View	detailed, flat relational	summarized, multidimensional
Unit of work	short, simple transaction	complex query
Access	read/write	mostly read
Focus	data in	information out
Operations	index/hash on primary key	lots of scans
Number of records accessed	tens	millions
Number of users	thousands	hundreds
DB size	GB to high-order GB	≥ TB
Priority	high performance, high availability	high flexibility, end-user autonomy
Metric	transaction throughput	query throughput, response time

Figura 2.2: Tabella di comparazione tra sistemi OLAP e sistemi OLTP

2.3 Architettura multilivello del data warehousing

I data warehouse adottano spesso un'architettura a tre livelli (notare la Figura 2.3):

1. Il livello più basso è un server di database warehouse che è quasi sempre un sistema di database relazionale[3]. Strumenti e utilità di back-end vengono utilizzati per inserire i dati nel livello inferiore da database operativi o altre fonti esterne. Questi strumenti e utilità eseguono l'estrazione, la pulizia, la trasformazione dei dati e, inoltre, le funzioni di caricamento e aggiornamento per aggiornare il data warehouse. I dati vengono estratti utilizzando interfacce di programmi applicativi note come **gateway**; esso è supportato dal DBMS[4] sottostante e consente ai

[3]Database relazionale: è un tipo di database di archiviazione che fornisce accesso a data points correlati tra loro. I database relazionali sono basati sul modello relazionale, un modo intuitivo e diretto di rappresentare i dati nelle tabelle.

[4]DBMS: indica le parole inglesi *Database Management System*, ovvero Sistema di gestione di basi di dati; esso è un sistema software progettato per consentire la creazione, la manipolazione e l'interrogazione efficiente di database.

programmi client di generare codice SQL da eseguire su un server. Questo livello contiene anche una **metadata repository**, che immagazzina informazioni sul data warehouse e i suoi contenuti (argomento descritto in modo dettagliato nella sottosezione 2.3.1).

2. Il livello intermedio è un server OLAP che viene tipicamente implementato utilizzando o un modello OLAP relazionale (**ROLAP**), cioè un DBMS relazionale esteso che mappa le operazioni su dati multidimensionali in operazioni relazionali standard, o un modello OLAP multidimensionale (**MOLAP**), cioè un server per scopi speciali che implementa direttamente dati e operazioni multidimensionali (argomento descritto in modo dettagliato nella sottosezione 2.3.2).

3. Il livello superiore è un livello client front-end, che contiene strumenti di query e report, strumenti di analisi e/o strumenti di data mining.

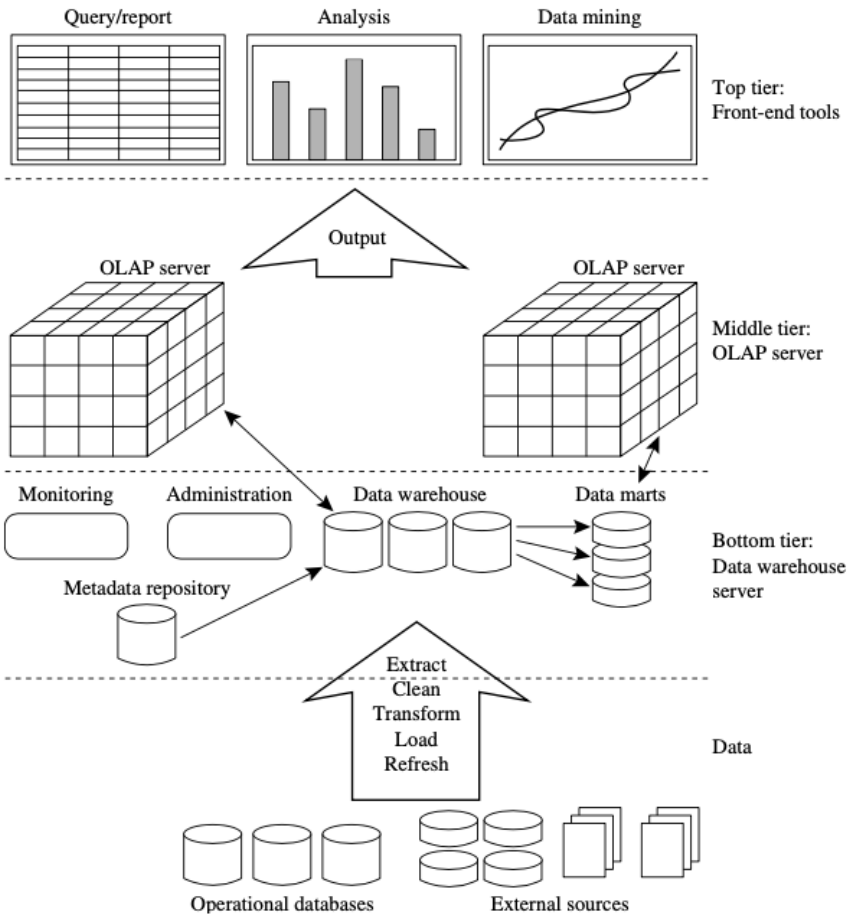


Figura 2.3: Architettura a tre livelli del data warehouse

2.3.1 Metadata repository

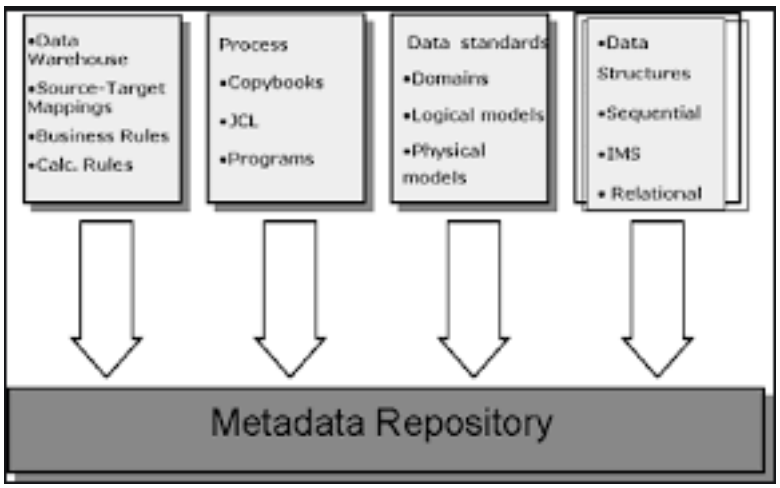


Figura 2.4: Metadata Repository

I metadata sono dati che vengono creati per definire i nomi dei dati e gli oggetti del data warehouse. Una metadata repository dovrebbe contenere:

- Una descrizione della struttura del data warehouse, che include lo schema, la vista, le dimensioni, le gerarchie, le definizioni dei dati derivati e, anche, le posizioni e il contenuto del data mart.
- Metadati operativi, che includono la derivazione dei dati, valuta dei dati e informazioni di monitoraggio.
- Gli algoritmi utilizzati per il riepilogo, che includono algoritmi di definizione di misure e dimensioni, dati su granularità, partizioni, aree tematiche, aggregazione, riepilogo e query e report predefiniti.
- Mappatura dall'ambiente operativo al data warehouse, che include i database di origine e il loro contenuto, descrizioni del gateway, partizioni di dati, estrazione dei dati, pulizia, regole di trasformazione e impostazioni predefinite, aggiornamento dei dati e regole di eliminazione e sicurezza.
- Dati relativi alle prestazioni del sistema, che includono indici e profili che migliorano l'accesso ai dati e le prestazioni di recupero, oltre alle regole per la tempistica e la pianificazione dei cicli di aggiornamento e replica.
- Metadati aziendali, che includono termini e definizioni aziendali, informazioni sulla proprietà dei dati e politiche di addebito.

2.3.2 ROLAP Server VS MOLAP Server

In questa sezione vi sono descritte le differenze tra un server ROLAP e un server MOLAP:

- *ROLAP:* questi sono i server intermedi che si trovano tra un server back-end relazionale e gli strumenti front-end del client. Utilizzano un DBMS relazionale o relazionale esteso per archiviare e gestire i dati del magazzino e un middleware OLAP per supportare i pezzi mancanti. I server ROLAP includono l'ottimizzazione per ciascun back-end DBMS, l'implementazione della logica di navigazione di aggregazione e strumenti e servizi aggiuntivi. La tecnologia ROLAP tende ad avere una maggiore scalabilità rispetto alla tecnologia MOLAP.

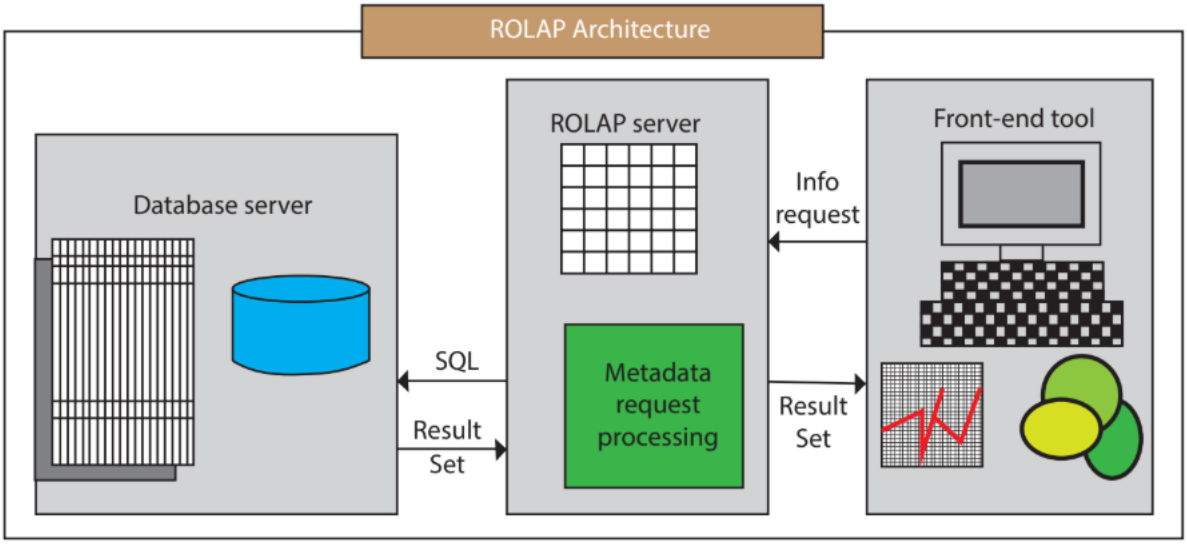


Figura 2.5: Modello ROLAP Server

- *MOLAP:* questi server supportano visualizzazioni di dati multidimensionali tramite motori di archiviazione multidimensionali basati su array. Mappano viste multidimensionali direttamente su strutture di array di cubi di dati. Il vantaggio dell'utilizzo di un cubo di dati è che consente una rapida indicizzazione a dati riepilogati precalcolati.

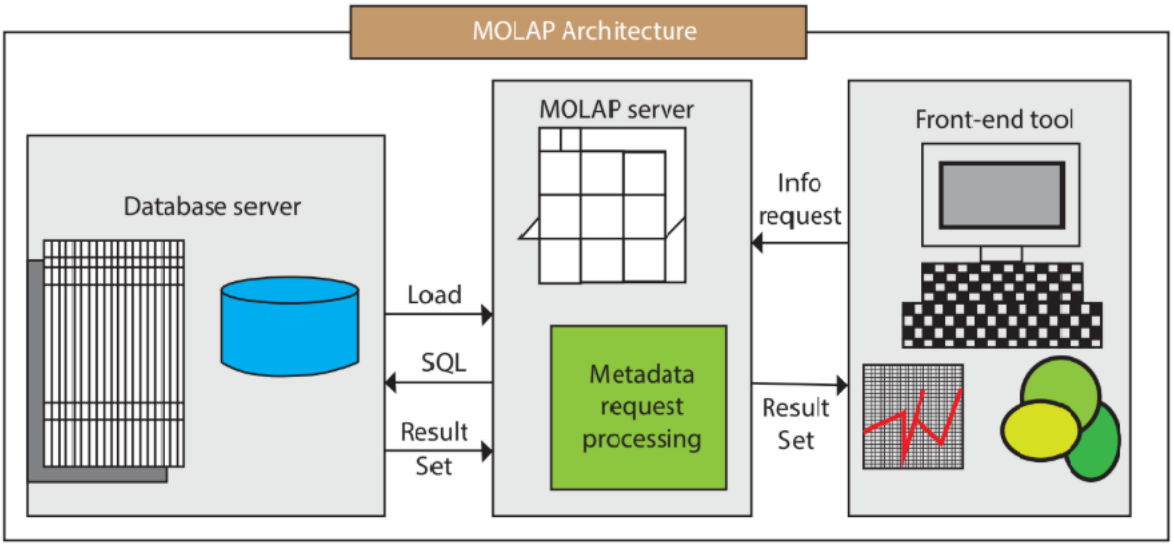


Figura 2.6: Modello MOLAP Server

2.4 Modelli di data warehouse

Da un punto di vista dell'architettura ci sono tre modelli di data warehouse: *Enterprise warehouse, data mart e virtual warehouse.*

2.4.1 Enterprise warehouse

L'enterprise warehouse raccoglie tutte le informazioni sugli argomenti che coprono l'intera organizzazione. Fornisce l'integrazione dei dati a livello aziendale, di solito da uno o più sistemi operativi o fornitori di informazioni esterni, ed ha un ambito interfunzionale. In genere esso contiene dati dettagliati e dati riepilogativi.
Ci sono due tipi di architetture molto importanti: *Corporate Information Factory* di W. H. Inmon, citato precedentemente, e Dimensional Data Warehouse di Raplh Kimball[5].

Corporate Information Factory di Inmon

Nella business intelligence, la Corporate Information Factory è una struttura di raccolta integrale dei dati proposta da William H. Inmon nel 1998. Con Corporate Information

[5]Ralph Kimball (1944) è un autore in materia di data warehousing e business intelligence. È uno degli architetti originali del data warehousing ed è noto per convinzioni a lungo termine secondo cui i data warehouse devono essere progettati per essere comprensibili e veloci. La sua metodologia, nota anche come modellazione dimensionale o metodologia Kimball, è diventata lo standard nell'area del supporto decisionale.

Factory si definisce, pertanto, il congiunto delle informazioni (non solo attuali, ma anche passate) che servono per una lettura trasversale ed innovativa dell'attività di una impresa. La caratterizzano il fatto di essere:

- **Trasversale** perché non corrisponde ad un solo dipartimento;
- **Innovativa** perché cerca di rispondere a domande nuove o capire "modelli" esistenti ma non evidenti;
- **Non solo attuale** perché mantiene uno storico dei dati.

Al fine di creare una Corporate Information Factory sono necessari (oltre ai dati in sé) i metadati che formano da raccordo fra le varie fonti.
Per capire l'architettura si inizia ad osservarla nella Figura 2.8 da sinistra; a sinistra vi si trovano i sistemi operazionali che supportano il business. Questi sistemi alimentano un processo chiamato **ETL**[6] che indica le operazioni di *estrarre, trasformare, caricare.*

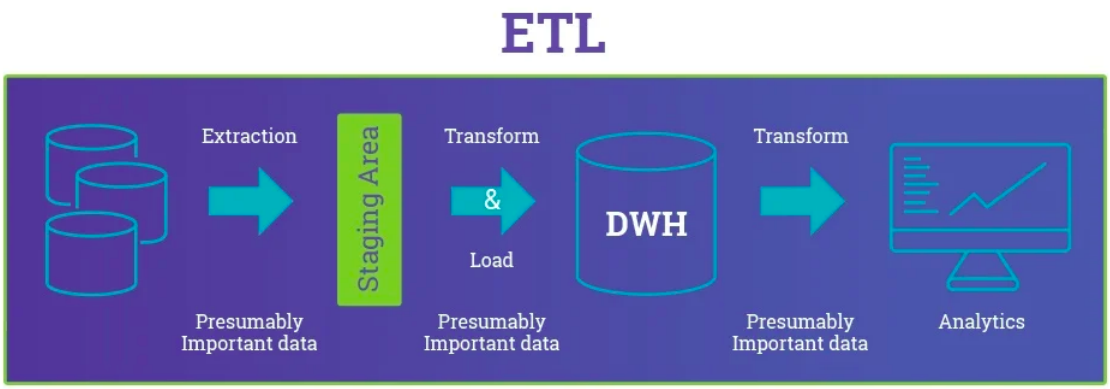


Figura 2.7: Processo ETL

Questo processo consolida le informazioni ottenute dai sistemi operazionali, le integra e le carica in una singola repository chiamata enterprise data warehouse. Quest'ultima

[6]ETL: funzioni svolte da strumenti di back-end per popolare ed aggiornare i propri file di dati. in particolare, le funzioni incluse sono :

- **Estrazione dei dati:** raccoglie dati da origini multiple, eterogenee ed esterne;
- **Pulizia dei dati:** rileva gli errori nei dati e li corregge quando possibile;
- **Trasformazione dei dati:** converte i dati dal formato sorgente al formato warehouse;
- **Caricamento:** ordina, riepiloga, consolida, calcola le visualizzazioni, controlla l'integrità e crea indici e partizioni.
- **Aggiornamento:** propaga gli aggiornamenti dalle sorgenti di dati al warehouse.

è al centro della Corporate Information Factory; essa è una repository integrata di dati atomici. Integrata dai vari sistemi operazionali, essa contiene una rappresentazione consistente e definitiva di una attività di business in un singolo posto. Atomici in natura; il dato nella repository è catturato al livello più basso di dettaglio possibile. L' obiettivo dell'enterprise data warehouse è di aggiungere dati storici ad una varietà di sistemi analitici. Essa è di solito registrata in un database relazionale e Inmon sostiene l'utilizzo di un database in 3° forma normale. Attorno all'enterprise data warehouse vi sono altri componenti, tra cui i data marts, database che supportano una vista dipartimentale delle informazioni. I data marts servono come fulcro delle attività analitiche, che includono query, report e altre attività.

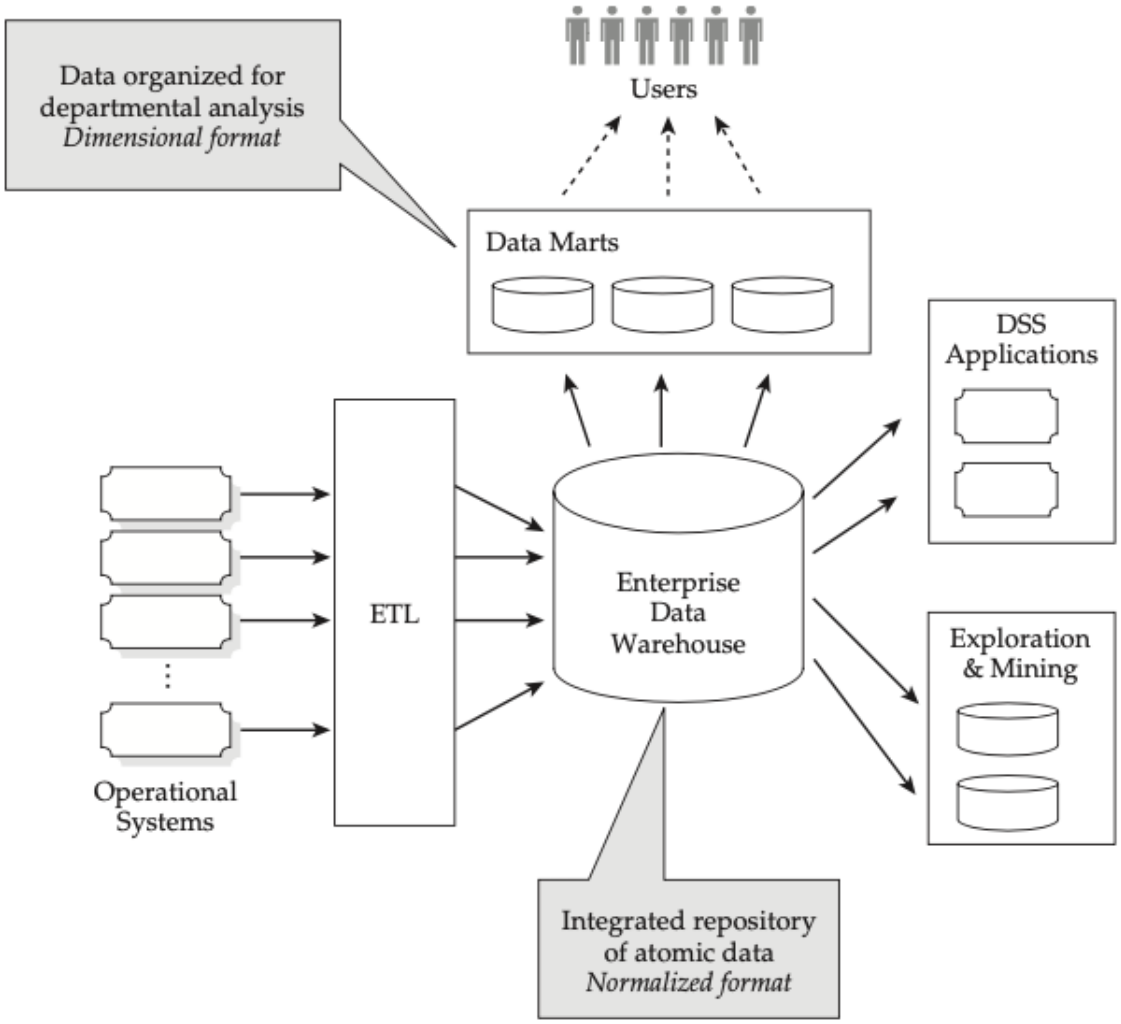


Figura 2.8: Vista del diagramma di Corporate Information Factory

Dimensional Data Warehouse di Kimball

Quest'altra categoria di architettura dei data warehouse è stata creata da Ralph Kimball. Kimball sviluppò un architettura enterprise per i data warehouse costruita sul concetto di design dimensionale. Egli condivise molte caratteristiche della Corporate Information Factory di Inmon; consente una repository integrata per i dati atomici e si basa sul design dimensionale per supportare l'analisi. L'architettura Dimensional Data Warehouse inizia assumendo una separazione tra i sistemi operazionali e quelli analitici. Come prima, i sistemi operazionali sono posizionati a sinistra del diagramma, da osservare nella Figura 2.9; essi servono per salvare dati storici. Spostandoci verso destra vi è il processo ETL che serve per consolidare le informazioni ottenute dai vari sistemi operazionali, le integra e le carica in una singola repository. Al centro, dopo il processo ETL, vi è una repository integrata per dati atomici; essa contiene una visione unica delle attività aziendali, come disegnata da tutta l'azienda e memorizza le informazioni in un formato altamente granulare o atomico. Infine, i data marts sono una area tematica all'interno dei data warehouse.
Questa architettura si differenzia dall'architettura di Inmon per due motivi:

1. L'architettura di Kimball è progettata secondo il principio del modello dimensionale, ovvero esso consiste in una serie di star schema o cubi, che catturano informazioni al livello più basso di dettaglio possibile; questa architettura è progettata usando il principio del modello ER;

2. L'architettura di Kimball è accessibile direttamente dai sistemi analitici.

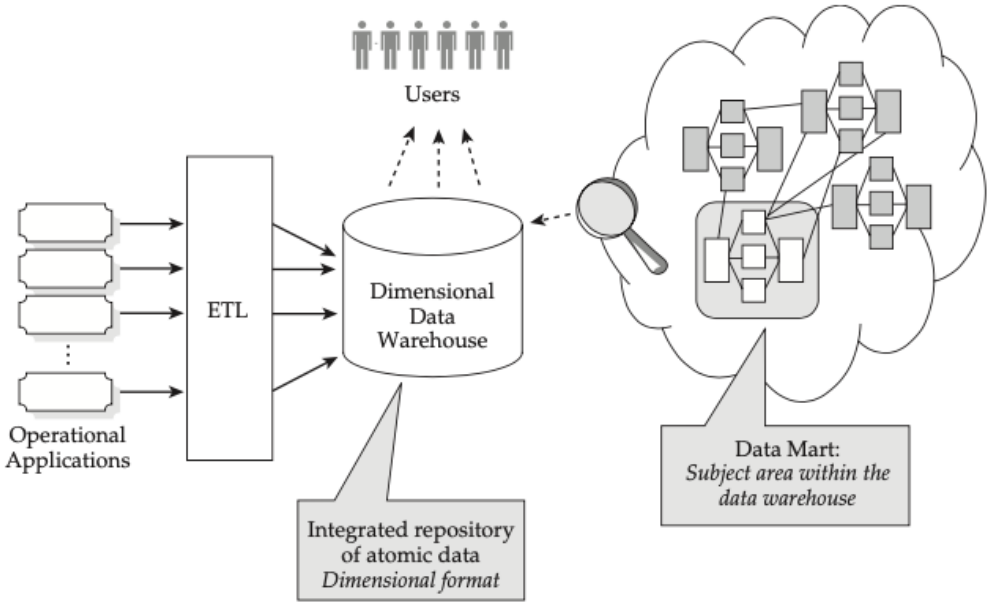


Figura 2.9: Vista del diagramma di Dimensional Data Warehouse

2.4.2 Data mart

Un data mart contiene un sottoinsieme di dati a livello aziendale che è utile per un gruppo specifico di utenti e, quindi, lo scopo è limitato a specifici soggetti selezionati. Ad esempio, un data mart di marketing può limitare i propri argomenti a clienti, articoli e vendite. I dati contenuti nei data mart tendono ad essere riassunti e a seconda della loro origine, i data mart possono essere classificati come indipendenti o dipendenti:

- **Data mart indipendenti:** provengono da dati acquisiti da uno o più sistemi operativi o fornitori di informazioni esterni, o da dati generati localmente all'interno di un particolare dipartimento o area geografica.
- **Data mart dipendenti:** provengono direttamente dai data warehouse aziendali.

L'architettura di data mart può ottenere risultati rapidi e poco costosi a breve termine ma allo stesso tempo può dare origine a costi ed inefficienze a lungo termine. Questa architettura è un data store analitico che non è stato progettato in un contesto enterprise; essa è focalizzata esclusivamente su un argomento (Figura 2.10).
Uno o più sistemi operazionali alimentano un database chiamato data mart. Il data mart può usare il design dimensionale, un modello entità-relazione, o altre forme di progettazione. Strumenti o applicazioni analitiche lo interrogano direttamente, ottenendo informazioni per gli utenti finali. Poiché i suoi risultati sono rapidi e meno costosi, vi possono essere più architetture con data marts per delle organizzazioni (Figura 2.11). Nel frattempo che un singolo data mart può apparire il percorso più efficiente per i risultati, la presenza di molteplici elementi di questo genere espone ad inefficienze poichè molteplici processi ETL caricano dati dalla stessa risorsa di sistema, causando costi di ridondanza di processi da mantenere. Inoltre, dato che i data marts possono essere di differenti tecnologie, spesso acquisiscono l'etichetta di *stovepipe*, che connota la mancanza di compatibilità. Questi problemi fanno sì che essi diventino isole di informazioni. I data marts sono sviluppati per soddisfare una serie ristretta di esigenze, ma non supportano l'analisi interfunzionale; quindi, il risparmio a breve termine lascia il posto a costi a lungo termine.

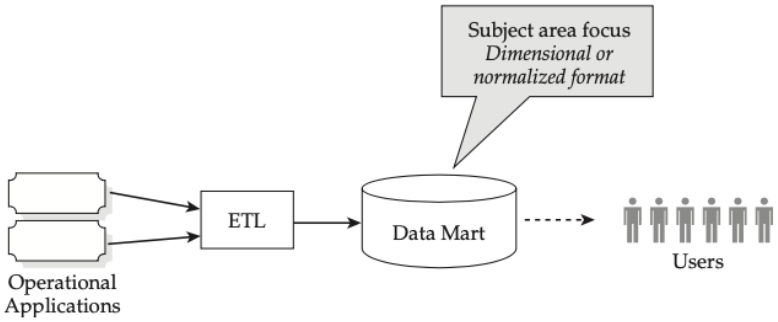


Figura 2.10: Architettura data mart

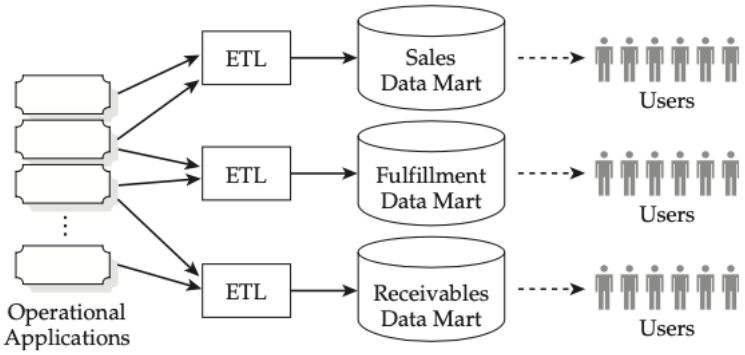


Figura 2.11: Molteplici architetture data marts

	Enterprise Level			Subject Area Level		
	Integrated Repository of Atomic Data	Format	Direct Access	Data Marts	Format	Direct Access
Corporate Information Factory	✓	3NF	No	Physical	Dimensional*	Yes
Dimensional Data Warehouse	✓	Dimensional	Yes*	Logical*	Dimensional	Yes
Data Mart	×	*n/a*	*n/a*	Physical	Dimensional*	Yes

* Optional

Architecture	Advocate	Also Known As	Description	Role of Dimensional Design
Corporate Information Factory	Bill Inmon	• Atomic data warehouse • Enterprise data warehouse	• *Enterprise data warehouse* component is an integrated repository of atomic data • It is *not* accessed directly • *Data marts* reorganize data for departmental use/analysis	Dimensional design used for data marts only
Dimensional Data Warehouse	Ralph Kimball	• Enterprise data warehouse • Bus architecture • Architected data marts • Virtual data marts	• *Dimensional data warehouse* is an integrated repository of atomic data • It may be accessed directly • Subject areas within the dimensional data warehouse sometimes called *data marts* • Data marts not required to be separate databases	All data is organized dimensionally
Data Marts	No takers, yet common	• Data mart • Silo • Stovepipe • Island	• Subject area implementation without an enterprise context	May employ dimensional design

Figura 2.12: Tabella riassuntiva delle 3 architetture descritte nelle sezioni: sottosezione 2.4.1 Enterprise e sottosezione 2.4.2 Data mart

2.4.3 Virtual warehouse

Un virtual warehouse è un insieme di viste su database operativi. Per un'elaborazione efficiente delle query, possono essere materializzate solo alcune delle possibili visualizzazioni di riepilogo. Un virtual warehouse è facile da costruire ma richiede una capacità in eccesso sui server di database operativi.

2.5 Modellazione del data warehouse

La modellazione del data warehouse è il processo di progettazione degli schemi delle informazioni dettagliate e riepilogative del data warehouse. Essa è una fase essenziale della costruzione di un data warehouse per due motivi principali:

1. Attraverso lo schema, i clienti del data warehouse possono visualizzare le relazioni tra i dati del warehouse, per utilizzarli con maggiore facilità.

2. Uno schema ben progettato consente di emergere una struttura di data warehouse efficace, per contribuire a ridurre i costi di implementazione dello stesso e migliorare l'efficienza del suo utilizzo

Gli obiettivi della modellazione di un data warehouse sono: sviluppare uno schema che descriva la realtà, o almeno una parte del fatto, che il data warehouse deve supportare e fare in modo che il data warehouse supporti in modo efficiente query complesse su informazioni a lungo termine.
I data warehouse e gli strumenti OLAP si basano su un modello di dati multidimensionale; questo modello visualizza i dati nella forma di ***data cube***, ovvero in un cubo di dati.

2.5.1 Data cube: un modello di dati multidimensionale

Sebbene di solito si pensa ai cubi come strutture geometriche 3D, nel data warehousing il data cube è n-dimensionale. Esso consente ai dati di essere modellati e visualizzati in più dimensioni. Un data cube è definito da *dimensioni* e *fatti*.
In termini generali, le **dimensioni** sono le prospettive o entità rispetto alle quali un'organizzazione desidera tenere dei registri. Ad esempio, un azienda può creare un data warehouse delle vendite al fine di tenere traccia delle vendite del negozio rispetto alle dimensioni di ora, oggetto, filiale e posizione. Queste dimensioni permettono al negozio di tenere traccia di elementi come le vendite mensili di articoli e le filiali e le località in cui gli articoli sono stati venduti. A ciascuna dimensione può essere associata una tabella, denominata *tabella della dimensione*, che descrive ulteriormente la dimensione. Ad esempio, una tabella della dimensione per l'articolo può contenere gli attributi nome, marca e tipo dell'elemento. Le tabelle delle dimensioni possono essere specificate

da utenti o esperti o generate automaticamente e adattate in base alle distribuzioni dei dati. Un modello di dati multidimensionale è tipicamente organizzato attorno ad un tema centrale, come le vendite. Questo tema è rappresentato da una *tabella dei fatti*. I **fatti** sono misure numeriche; essi possono essere pensati come le quantità con cui vogliamo analizzare le relazioni tra le dimensioni. Esempi di fatti per un data warehouse sulle vendite includono dollari venduti (ovvero importo delle vendite in dollari), unità vendute (numero di unità vendute) e importo preventivato. La tabella dei fatti contiene i nomi dei fatti, o delle misure, nonché le chiavi di ciascuna delle tabelle dimensionali correlate.

	location = *"Vancouver"*			
	item *(type)*			
time *(quarter)*	*home entertainment*	*computer*	*phone*	*security*
Q1	605	825	14	400
Q2	680	952	31	512
Q3	812	1023	30	501
Q4	927	1038	38	580

Figura 2.13: Esempio di visualizzazione 2-D dei dati di vendita di un'azienda in base a *time* e *item*.

	location = "Chicago"				***location*** = "New York"				***location*** = "Toronto"				***location*** = "Vancouver"			
	item				***item***				***item***				***item***			
time	*home ent.*	*comp.*	*phone*	*sec.*	*home ent.*	*comp.*	*phone*	*sec.*	*home ent.*	*comp.*	*phone*	*sec.*	*home ent.*	*comp.*	*phone*	*sec.*
Q1	854	882	89	623	1087	968	38	872	818	746	43	591	605	825	14	400
Q2	943	890	64	698	1130	1024	41	925	894	769	52	682	680	952	31	512
Q3	1032	924	59	789	1034	1048	45	1002	940	795	58	728	812	1023	30	501
Q4	1129	992	63	870	1142	1091	54	984	978	864	59	784	927	1038	38	580

Figura 2.14: Esempio di visualizzazione 3-D dei dati di vendita di un'azienda in base a *time*, *item* e *location*.

Le tabelle presenti nella Figura 2.13 e nella Figura 2.14 mostrano i dati con diversi gradi di riepilogo. Nel campo della ricerca dei data warehousing, un cubo di dati, come quelli mostrati nelle tabelle sopra, viene spesso indicato come un ***cuboide*** (o parallelepipedo). Dato un insieme di dimensioni, possiamo generare un cuboide per ciascuno dei possibili sottoinsiemi delle dimensioni date. Il risultato formerebbe un *reticolo* di cuboidi (Figura 2.15), ciascuno dei quali mostra i dati ad un diverso livello di riepilogo o *group-by*, raggruppamento, e che, infine, vengono indicati come un cubo di dati. Il cuboide che contiene il livello di riepilogo più basso è chiamato ***cuboide di base***. Ad esempio, il cuboide 4-D nella Figura 2.16 è il cuboide di base per tali dimensioni: tempo, elemento, posizione e fornitore. La Figura 2.17 è un cuboide 3-D (non base) per tempo, articolo e posizione, riepilogato per tutti i fornitori. Il cuboide 0-D, che detiene il livello più alto di riepilogo, è chiamato il ***cuboide apice***. Nell'esempio utilizzato, esso è il totale delle vendite, o dollari venduti, riassunti in tutte e quattro le dimensioni. Il cuboide apice è tipicamente indicato da *all*, ovvero da tutto.

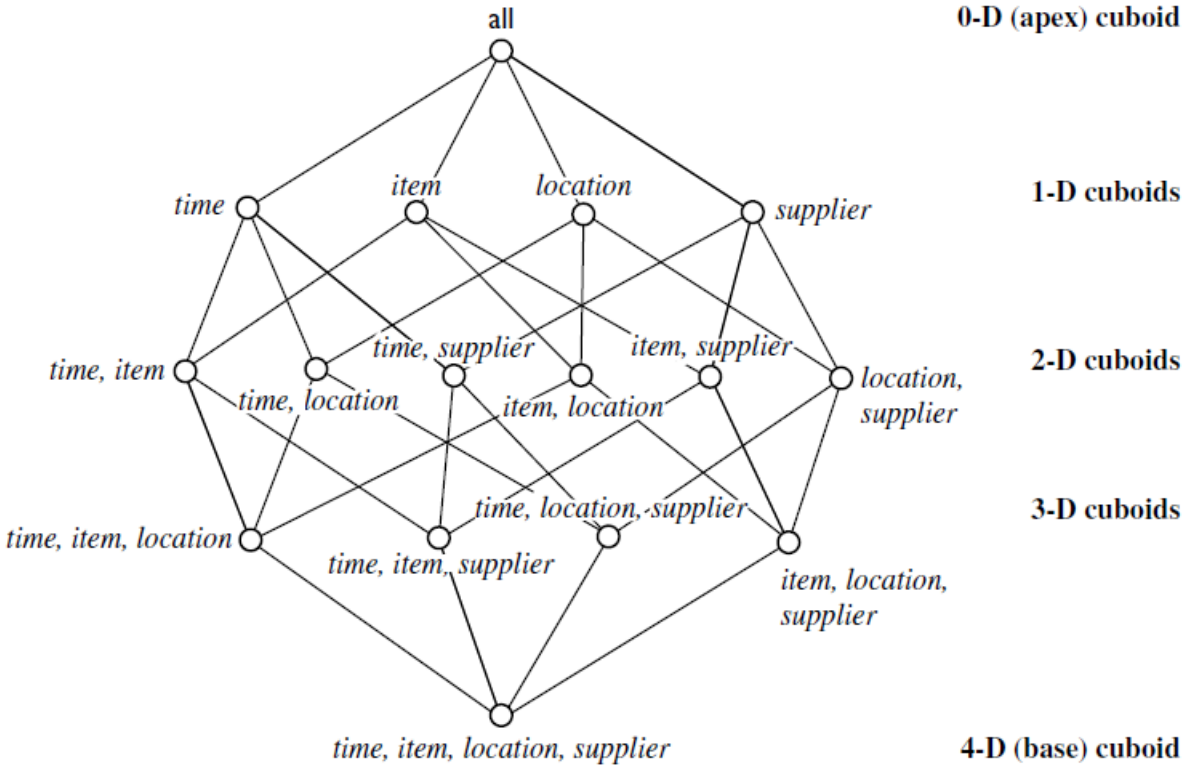


Figura 2.15: Reticolo di cuboidi che costituisce un cubo di dati 4-D, in base alle dimensioni *time*, *item*, *location* e *supplier*. Ogni cuboide rappresenta un diverso grado di riepilogo.

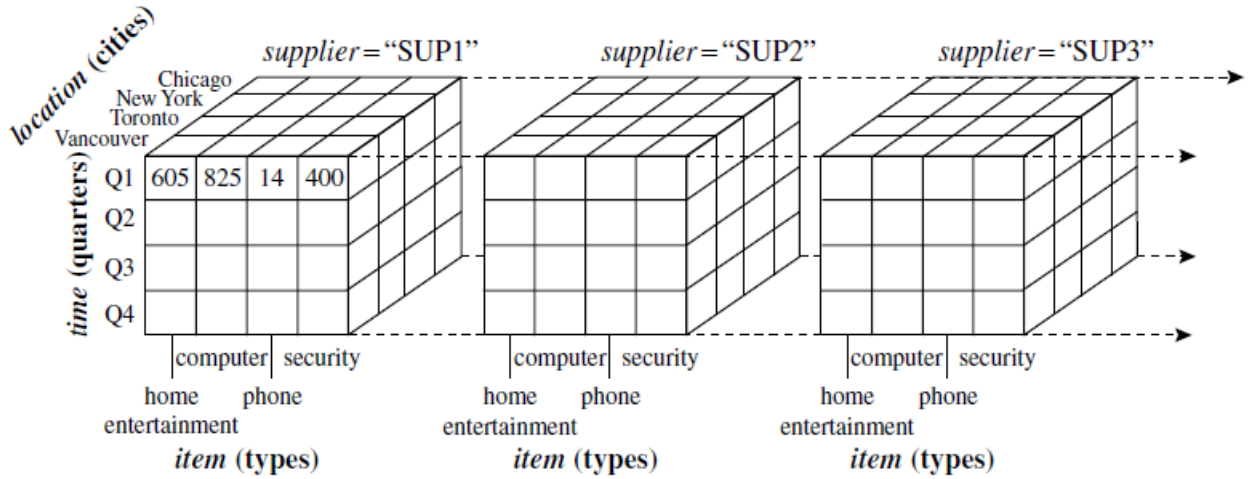


Figura 2.16: Rappresentazione in un cubo di dati 4-D dei dati di vendita, rispetto alle dimensioni di *tempo*, *item*, *location* e *supplier*.

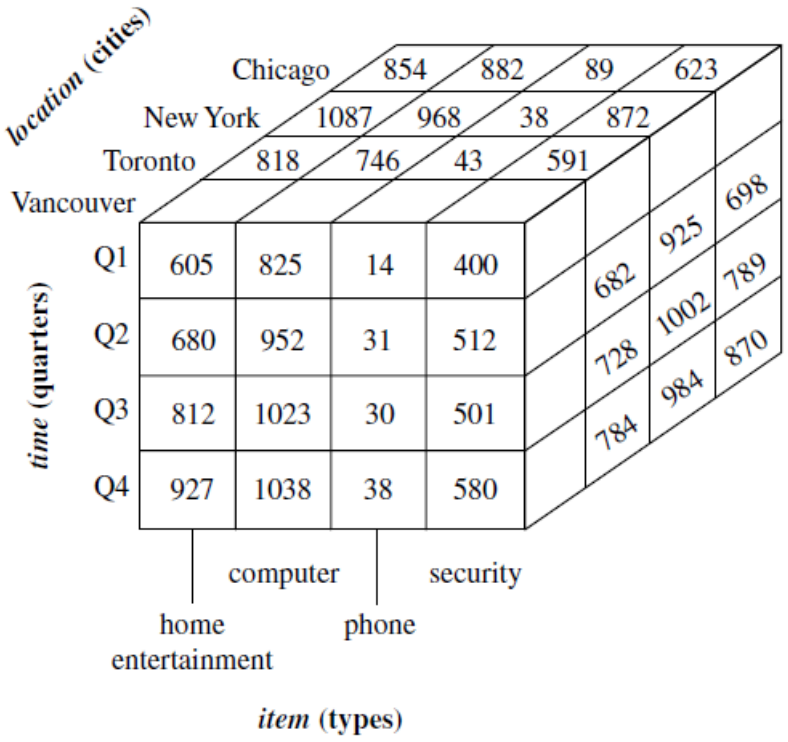


Figura 2.17: Rappresentazione in un cubo di dati 3-D dei dati della Figura 2.14, rispetto alle dimensioni *time*, *item* e *location*.

2.5.2 Star schema, Snowflake schema e Fact constellation schema: schemi per modelli di dati multidimensionale

Un data warehouse richiede uno schema conciso e orientato al soggetto che faciliti l'analisi dei dati online. Il modello di dati più popolare per un data warehouse è un modello multidimensionale, che può esistere sotto forma di uno *Star schema*, un *Snowflake schema* o un *Fact constellation schema*. Di seguito, vi è una descrizione dei tre schema citati [1].

Star schema

Il paradigma di modellazione più comune è lo **Star schema**, ovvero schema a 'stella', in cui il data warehouse contiene:

- Una grande tabella centrale, denominata *tabella dei fatti*, contenente la maggior parte dei dati, senza ridondanza;
- Una serie di tabelle assistenti più 'piccole', denominate *tabelle delle dimensioni*, una per ogni dimensione.

Il grafico di questo schema assomiglia ad un esplosione di una stella, con le tabelle delle dimensioni visualizzate in un modello radiale attorno alla tabella dei fatti centrale (come si può notare nella figura Figura 2.18). Nella figura, ogni dimensione è rappresentata da una sola tabella e ogni tabella contiene una serie di attributi. Ad esempio, la tabella delle dimensioni della *location*, 'posizione', contiene gli attributi: location-key, street, city, province-or-state, country. Questo vincolo può introdurre una certa ridondanza; ad esempio, può succedere che due città siano entrambe appartenenti allo stesso stato e che, quindi, verranno create le voci per tali città nella tabella delle dimensioni della location generando una ridondanza tra gli attributi province-or-state e country.
Nella Figura 2.18, come già anticipato, vi è un esempio di uno Star schema per le vendite di un'azienda. Le vendite sono considerate rispetto quattro dimensioni: *time*, *item*, *branch* e *location*. Lo schema è composto da una tabella dei fatti centrale per le vendite che contiene le chiavi di ciascuna delle quattro dimensioni, insieme a due misure: *dollars-sold* e *units-sold*. Per ridurre al minimo le dimensioni della tabella dei fatti, gli identificatori delle dimensioni (ad esempio *time-key* e *item-key*) sono identificatori generati dal sistema.

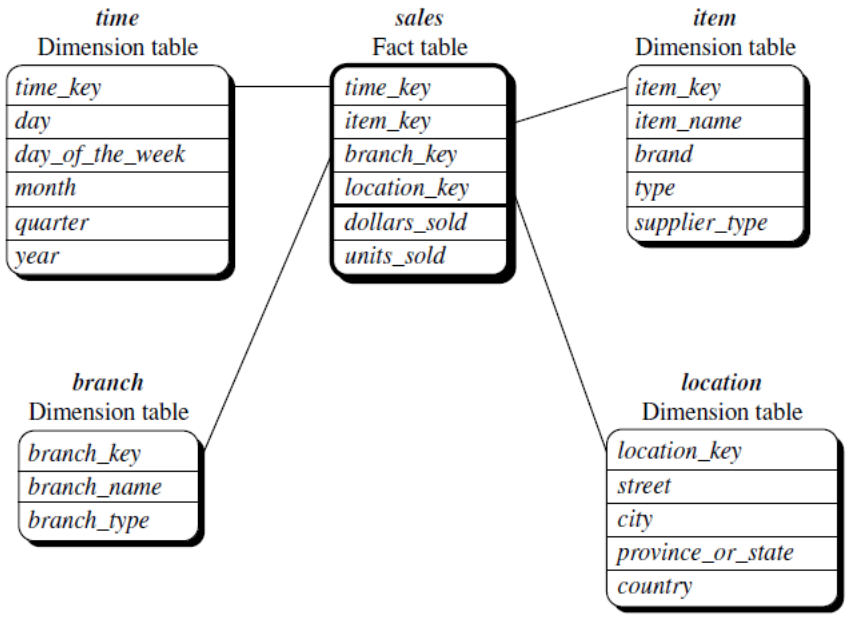


Figura 2.18: Esempio di uno Star schema per le vendite di un'azienda.

Snowflake schema

Lo Snowflake schema (ovvero schema a 'fiocco di neve') è una variante del modello dello Star schema, dove alcune tabelle delle dimensioni sono normalizzate, suddividendo così ulteriormente i dati in tabelle aggiuntive. La normalizzazione è un procedimento volto all'eliminazione della ridondanza informativa e del rischio di incoerenza dal database; questo processo si fonda su un semplice criterio: se una relazione presenta più concetti tra loro indipendenti, la si decompone in relazioni più piccole, una per ogni concetto. Questo tipo di processo non è sempre applicabile in tutte le tabelle, dato che in alcuni casi potrebbe comportare una perdita di informazioni. Il grafico finale dello schema ha una forma simile a un fiocco di neve (come si può notare nella figura Figura 2.19). La principale differenza tra i modelli Snowflake schema e Star schema è che le tabelle delle dimensioni dello Snowflake schema possono essere mantenute in forma normalizzata per ridurre ridondanze e per consentire di risparmiare spazio di archiviazione. Inoltre, lo Snowflake schema può ridurre l'efficacia della navigazione, poiché saranno necessari più *join* per eseguire una query e, di conseguenza, le prestazioni del sistema potrebbero essere influenzate negativamente. Quindi, sebbene esso riduca la ridondanza, nella progettazione del data warehouse non viene utilizzato quanto lo Star schema.
Nella Figura 2.19 vi è un esempio di uno Snowflake schema per le vendite di un'azienda. In questo caso, la tabella dei fatti sulle vendite è identica a quella dello Star schema nella Figura 2.18. La principale differenza tra i due schemi è nella definizione delle tabel-

le delle dimensioni. La tabella della dimensione *item* nello Star schema è normalizzata nello Snowflake schema e il risultato è nella presenza di due nuove tabelle, *item* (articoli) e *supplier* (fornitori). Ad esempio, la tabella della dimensione *item* adesso contiene gli attributi *item-key*, *item-name*, *brand*, *type* e *supplier-key*, dove supplier-key è collegata alla tabella delle dimensioni *supplier*, contenente gli attributi *supplier-key* e *supplier-type*. Allo stesso modo, la tabella della dimensione *location* nello Star schema può essere normalizzato in due nuove tabelle: *location* e *city*. La *city-key* della tabella location si collega alla *city-key* della tabella city. Inoltre si può notare che, se si vuole, si può effettuare un'ulteriore normalizzazione della tabella city degli attributi *province-or-state* e *country* nello Snowflake schema.

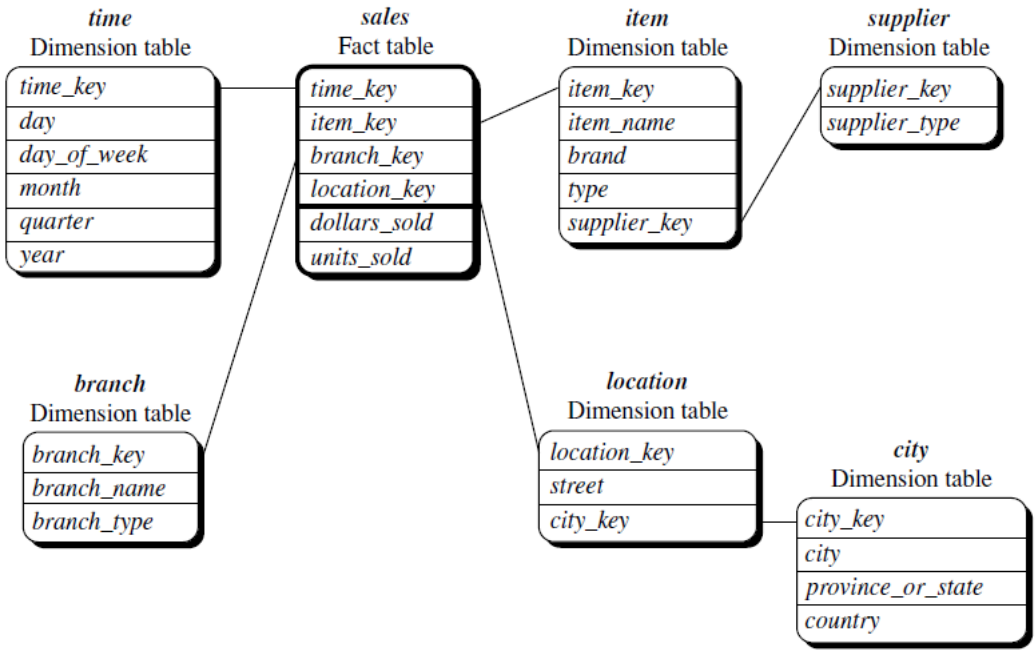


Figura 2.19: Esempio di uno Snowflake schema per le vendite di un'azienda.

Fact constellation

Alcune applicazioni sofisticate possono richiedere la condivisione di più tabelle delle dimensioni con le tabelle dei fatti. Questo tipo di schema può essere visto come una raccolta di Star schema e, quindi, viene chiamato **schema galattico** o **Fact constellation**, 'costellazione di fatti' (come si può notare nella Figura 2.20).
Nella Figura 2.20 vi è un esempio di Fact constellation schema per le vendite di un'azienda. Questo Fact constellation schema specifica due tabelle dei fatti, *sales* e *shipping*. La definizione della tabella *sales* è identica a quella dello Star schema (Figura 2.18).

La tabella *shipping* ha cinque dimensioni (chiavi), *item-key*, *time-key*, *shipper-key*, *from-location*, *to-location* e due misure, *dollars-cost* e *units-shipped*. Il Fact constellation schema consente la condivisione delle tabelle delle dimensioni tra tabelle dei fatti. Ad esempio, le tabelle delle dimensioni *time*, *item* e *location* sono condivise tra le tabelle dei fatti *sales* e *shipping*.

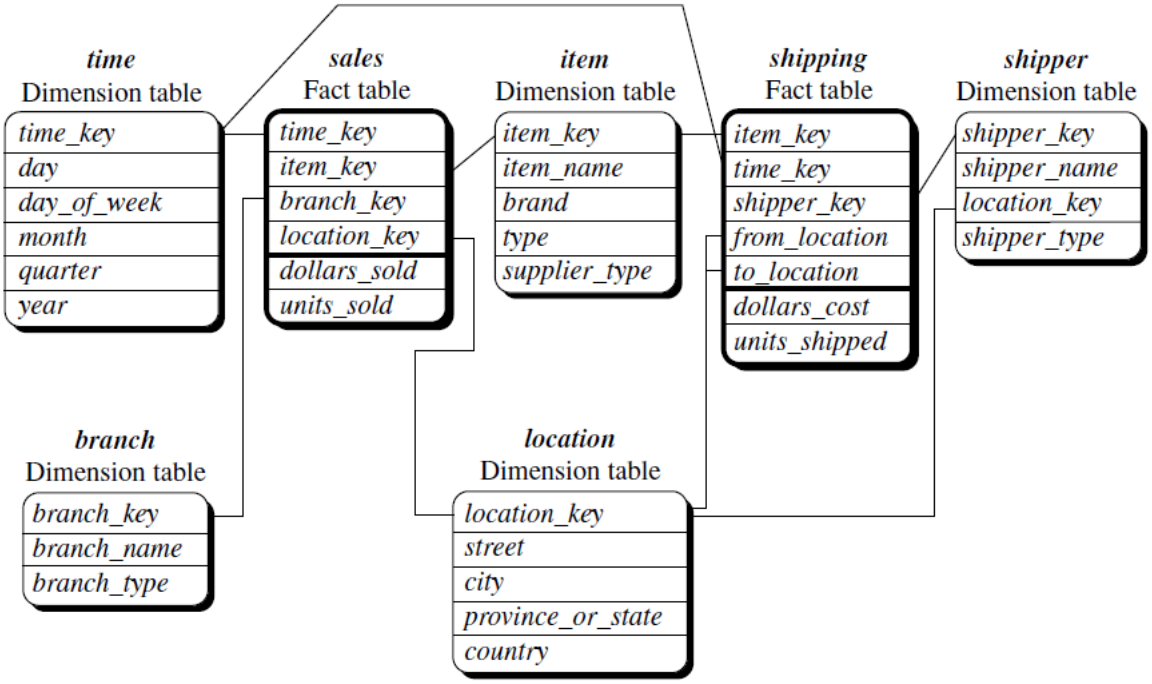


Figura 2.20: Esempio di un Fact constellation schema per le vendite di un'azienda.

Nel data warehousing vi è una distinzione tra la scelta degli schema che vengono adottati per un data warehouse e un data mart:

- Un data warehouse raccoglie informazioni su argomenti che riguardano l'intera organizzazione, come clienti, articoli, vendite, beni e personale, e quindi il suo ambito è a livello aziendale; il suo schema più adatto e quello più comunemente utilizzato è il Fact constellation schema, poiché può modellare soggetti multipli e correlati.
- Un data mart è un sottoinsieme di reparto del data warehouse che si concentra su argomenti selezionati, e quindi il suo ambito è a livello di reparto; il suo schema, invece, più utilizzato è lo Star schema o lo Snowflake schema, poiché entrambi sono orientati alla modellazione dei singoli soggetti, sebbene lo Star schema sia più popolare ed efficiente.

Capitolo 3

IBM NETEZZA VS SAP HANA

L'obiettivo principale del progetto portato a termine durante il tirocinio è stato la migrazione dei flussi appartenenti ai server di NETEZZA ai server di HANA. Di seguito, vi è la descrizione delle due tecnologie e la motivazione di questa migrazione.

3.1 IBM NETEZZA

Figura 3.1: Logo IBM Netezza

Definizione di Netezza

IBM[1] Netezza è un'appliance, ovvero un dispositivo che include un DBMS e un hardware dedicato, ottimizzata per il datawarehouse, interrogabile in SQL standard da gran parte dei client e dei front-end di Business Intelligence. La tecnologia Netezza è basata sull'elaborazione MPP (*Massively Parallel Computing*, ovvero a parallelismo massivo dei dati; per elaborazione a parallelismo massivo dei dati si intende un tipo di calcolo parallelo dove centinaia o addirittura migliaia di processori sono usati in modo coordinato

[1]IBM: acronimo di *International Business Machines Corporation* è un'azienda statunitense, la più antica e tra le maggiori al mondo nel settore informatico. Produce e commercializza hardware, software per computer, middleware e servizi informatici, offrendo infrastrutture, servizi di hosting, cloud computing, intelligenza artificiale, quantum computing e consulenza nel settore informatico e strategico.

all'interno di un unico grande computer [19]. L'elaborazioni delle query viene distribuita su un certo numero di SPU (*Snippet Processing Units*), unità di base di elaborazione e archiviazione su disco; esso è fondamentalmente un microcomputer autonomo costituito da una CPU, processori logici, memoria e archiviazione su disco e ha una logica per cercare rapidamente le informazioni e restituire solo i risultati corrispondenti delle porzioni di dati salvate sul suo disco.
Un sistema Netezza può essere composto da qualche decina a molte centinaia di SPU.

3.2 SAP HANA

Figura 3.2: Logo SAP Hana

Definizione di Hana

SAP[2] HANA è una piattaforma SAP basata sulla tecnologia 'in memory computing'. Si tratta di un'insieme di hardware e software che ha lo scopo di processare significative quantità di dati in tempo reale, usando l'elaborazione In-Memory, quindi in memoria anziché nei dischi. I dati quindi non vengono immagazzinati in database tradizionali ma in HANA [12].
Le sue caratteristiche si possono riassumere in questi elementi:

- In-memory database: tutte le operazioni sono lette e scritte direttamente nella memoria;
- Data memory in colonne: organizzazione in colonne, oltre a quella in righe, per ottimizzare la compressione dati e la velocità di elaborazione;
- Parallelismo: le CPU lavorano su processi paralleli e massivi;
- Aggregazione dinamica: il calcolo viene effettuato in memoria in 'runtime';
- Real Time Analysis: analisi dei dati in tempo reale;

[2]SAP: acronimo di *Systems Applications and Products* in Data Processing; per definizione, è sia il nome del software ERP (*Enterprise Resource Planning*) sia il nome dell'azienda. SAP Software è una multinazionale europea, fondata nel 1972 da Wellenreuther, Hopp, Hector, Plattner e Tschira, e sviluppa soluzioni software per la gestione delle operazioni aziendali e delle relazioni con i clienti.

- Non vi è ridondanza: tutto in live e in memory con una velocità di lettura superiore ai database standard;
- Scalabilità in Cloud: con le integrazioni di AWS (*Amazon Web Services* o *Google Cloud* o *Azure*) è possibile diminuire o aumentare facilmente potenza e memoria;
- Analisi predittiva[3] e Machine Learning: con la gestione in tempo reale dei dati è possibile effettuare analisi predittive con integrazioni con tecnologie di data science.

3.3 Migrazione dei flussi da tecnologia Netezza a tecnologia HANA

La migrazione che ho effettuato per il progetto durante il tirocinio è dovuta al fatto che, come spiegato nella sezione precedente, l'architettura Hana, essendo fondamentalmente un database in memoria centrale, immagazzina tutti i dati nella memoria (RAM) e, quindi, non c'è perdita di tempo nel caricamento dei dati dal disco rigido alla RAM, o durante il processo che mantiene alcuni dati sulla RAM e sul disco rigido contemporaneamente. Tutto è in-memory per tutto il tempo, il che consente alla CPU l'accesso immediato ai dati per l'elaborazione.

[3]Analisi predittiva: termine che comprende una varietà di tecniche statistiche per analizzare fatti storici e attuali e fornire predizioni sul futuro o su eventi sconosciuti.

Parte III

Progetto

Capitolo 4

Progettazione

Questo capitolo è dedicato alla descrizione della progettazione del progetto, in particolare nella spiegazione dei suoi requisiti, dei suoi obiettivi finali, delle metodologie utilizzate e l'architettura che è stata seguita per il raggiungimento del prodotto finale.

4.1 Requisiti del progetto

I requisiti più importanti per giungere alla realizzazione del progetto sono:

- Avere conoscenza delle tecnologie che operano nell'ambito dei data warehouse, quali SAP BW/SAP 4HANA, IBM Netezza e SAP BO;
- Conoscenza della teoria riguardante il mondo delle basi di dati, quali i database e i data warehouse;
- Conoscenza del linguaggio SQL, riguardante l'archiviazione, la manipolazione e il recupero di dati presenti nei database;
- Avere alcune competenze trasversali, quali abilità interpersonali, competenza digitali, teamwork, capacità comunicative, abilità di ricerca, organizzazione, capacità di analisi, problem solving.

4.2 Obiettivi finali

L'obiettivo primario del progetto è eseguire un'importante migrazione tecnologica per un cliente del settore Manufacturing. Ulteriori obiettivi sono: coniugare una formazione metodologica con training quotidiani sulle tecnologie, in affiancamento a tutor e team dedicati.

4.3 Metodologie utilizzate nel corso del progetto

Durante il percorso del progetto, le attività sono state svolte essenzialmente su tre pilastri:

- formazione tecnologica e inserimento su progetto reale (SAP BW/SAP 4HANA, IBM Netezza, SAP BO).
- Metodologie aziendale e modalità operativa AGILE; per modalità operativa AGILE si intende un tipo di approccio alla collaborazione e ai flussi di lavoro fondato su una serie di valori in grado di guidare il nostro modo di procedere. Il lavoro viene organizzato in rilasci rapidi di modifiche, cosicché il cliente possa essere soddisfatto. Questa metodologia impiega un lavoro di gruppo e approcci flessibili per giungere ad un miglioramento continuo; ogni componente del team lavora in autonomia, rimanendo in contatto con i rappresentanti aziendali tramite incontri periodici durante il ciclo di vita del progetto [18].
- Training sulle principali tecnologie aziendali propedeutiche a una formazione a 360° da parte di un tutor.

4.4 Architettura del procedimento utilizzato

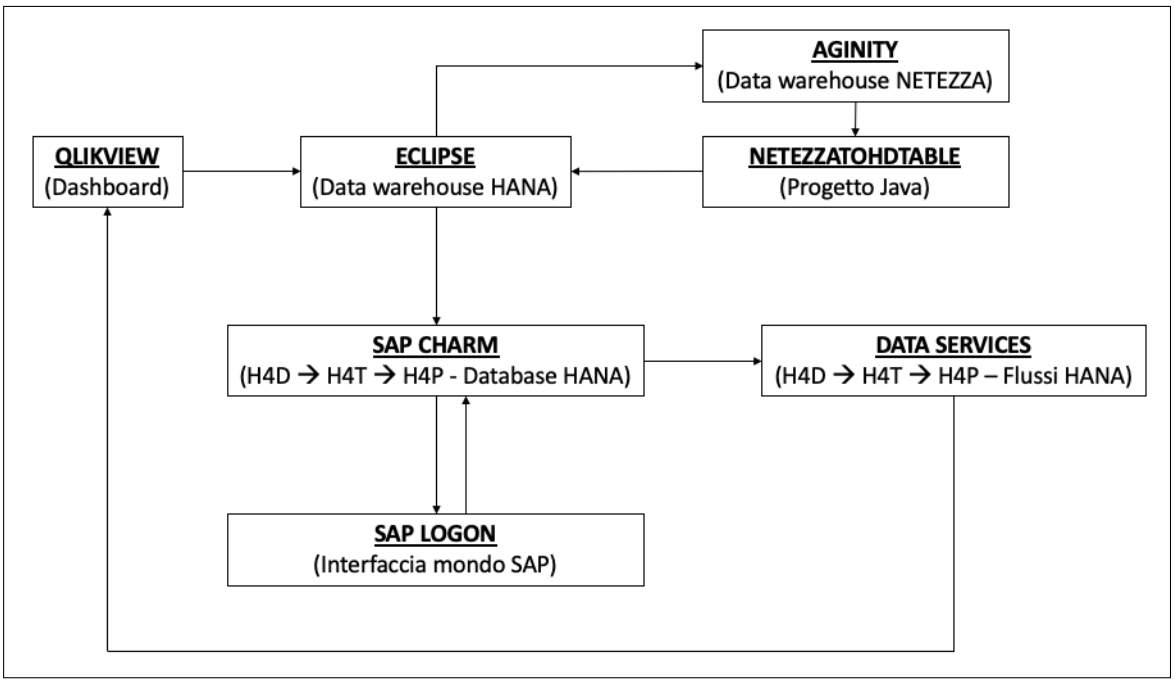


Figura 4.1: Architettura del progetto

Per raggiungere la finalizzazione del progetto è stato indispensabile l'utilizzo di alcuni strumenti, presenti nella figura Figura 4.1, che verranno descritti in questa sezione 5.1. Nella figura citata si possono notare, grazie alla direzione delle frecce, dei procedimenti utilizzati nel corso del progetto, descritti in modo dettagliato in questa sezione 5.2; in sequenza:

1. QlikView $\longrightarrow$ Eclipse.
2. Eclipse $\longrightarrow$ Aginity.
3. Aginity $\longrightarrow$ NetezzaToHdTable.
4. NetezzaToHdTable $\longrightarrow$ Eclipse.
5. Eclipse $\longrightarrow$ SAP Charm.
6. SAP Charm $\rightleftharpoons$ SAP Logon.
7. SAP Charm $\longrightarrow$ SAP Data Services.
8. SAP Data Services $\longrightarrow$ QlikView.

Con questa architettura utilizzata si può notare che siamo andati ad agire su due lati: **lato dashboard**, tutto ciò che si riferisce alla dashboard stessa e, quindi, la componente QlikView, e **lato data warehouse**, tutto ciò che serve per la migrazione delle tabelle da tecnologia NETEZZA a tecnologia HANA e, quindi, i restanti componenti.

Capitolo 5

Implementazione

In questo capitolo vengono descritti in modo dettagliato gli strumenti che sono stati utilizzati nell'elaborazione del progetto, lato dashboard e lato data warehouse. Inoltre vengono anche descritte, in modo cronologico, le operazioni che sono state eseguite più volte, con l'aggiunta di foto per far comprendere meglio l'esecuzione (N.B. le foto utilizzate hanno dei dati che sono oscurati per tutelare il cliente che ha richiesto il progetto).

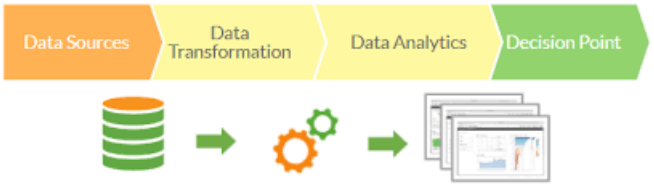
5.1 Strumenti utilizzati

5.1.1 *Lato dashboard*

Lato dashboard, le principali operazioni che sono state svolte riguardano le query presenti negli script del file QVW da modificare che servono per analizzare le tabelle presenti nel data warehouse HANA e le task che servono ad attivare gli script del file QVW; lo strumento utilizzato è stato QlikView.

QlikView

Figura 5.1: Logo QlikView



Al giorno d'oggi, QlikView è uno strumento di analisi che mostra facilmente una panoramica generale dei dati per scoprirne le relazioni anche quando si gestiscono serie di dati ampie e complesse, come nel nostro caso, collegandosi direttamente alla sorgente dati (HANA). La tecnologia del modello di dati associativo su cui si basa QlikView consente di creare un'interfaccia unica per la presentazione interattiva e l'analisi di qualsiasi tipo di informazioni (Dashboard). I sistemi di ricerca delle informazioni tradizionali richiedono spesso un approccio dall'alto verso il basso, mentre QlikView consente di iniziare con qualsiasi parte di dati, caricati in memoria, indipendentemente dalla sua posizione nella struttura dei dati; quindi, esso garantisce velocità nel fornire informazioni rispetto ai sistemi di ricerca tradizionali che rendono più lenta l'acquisizione di informazione ad un utente. Con QlikView è possibile:

- Creare un'interfaccia dell'utente finale flessibile per un information warehouse.
- Ottenere snapshot delle relazioni tra i dati.
- Realizzare presentazioni basate sui dati.
- Creare grafici e tabelle dinamici.
- Eseguire analisi statistiche.
- Collegare descrizioni e applicazioni multimediali ai propri dati.
- Creare sistemi esperti personalizzati.
- Creare nuove tabelle, che uniscono informazioni da molte sorgenti.
- Creare sistemi di Business Intelligence personalizzati.

Nel progetto, QlikView è stato utilizzato per la realizzazione della Dashboard, modificando ogni tab presente nel file QVW iniziale; in ogni tab sono presenti varie query che servono per analizzare delle tabelle presenti nel database di HANA e, una volta eseguite, il proprio risultato viene registrato in file QVD. Un file QVD è un *flat data file*, ovvero un file di dati piatto, con estensione '.qvd'; esso può memorizzare una singola tabella di dati e viene creato nello script di caricamento di un file QVW. La particolarità di un file QVD è che viene compresso durante la creazione utilizzando gli stessi algoritmi utilizzati da QlikView per archiviare i dati in memoria e, quindi, i file possono essere molto piccoli per la quantità di dati che contengono, consentendo di risparmiare e ricaricare tempo e spazio.

Un ulteriore strumento utilizzato nel progetto, legato a QlikView, è la QMC (*QlikView Management Console*), ovvero la console di gestione di QlikView; essa è il portale principale di amministrazione nella configurazione e nella gestione di QlikView Server. Nella sua pagina iniziale vi sono 4 tabs:

- **Status:** contiene informazioni sui tasks che possono essere pianificati e anche informazioni sui diversi servizi QlikView in Windows. Sono inoltre disponibili informazioni sulle statistiche di QlikView Server.
- **Users:** si trovano i dettagli sulla gestione degli utenti e le informazioni sulla gestione dell'accesso alla sezione.
- **Documents:** contiene informazioni sui documenti di origine, sulle attività assegnate e sui documenti dell'utente.
- **System:** comprende la configurazione, la licenza e le attività di supporto.

Durante il progetto, questo strumento è stato utilizzato per creare le attività che servivano ad 'attivare' i vari script di codice presenti nel file QVW e, quindi, a generare i vari file QVD che alimentano la Dashboard.

5.1.2 *Lato data warehouse*

Lato data warehouse, sono state svolte tutte le operazioni di migrazione delle tabelle e dei flussi che le alimentano da tecnologia NETEZZA a tecnologia HANA; gli strumenti che sono stati utilizzati sono:

- **Eclipse**, descritto a pagina 45;
- **Aginity**, descritto a pagina 47;
- **SAP Charm e SAP Logon**, descritto a pagina 49;
- **SAP Data Services**, descritto a pagina 50;
- **NetezzaToHdTable**, descritto a pagina 54.

Eclipse

Figura 5.2: Logo Eclipse

Eclipse è un software gratuito ed è un IDE (*Integrated development environment*), ovvero è un ambiente di sviluppo integrato multilinguaggio e multipiattaforma. Esso contiene un'area di lavoro e un sistema di plug-in estendibile per la personalizzazione dell'ambiente di lavoro.
Durante il progetto è stato installato e utilizzato il tool per l'amministrazione e la gestione di SAP HANA, ovvero SAP HANA tool; esso consente lo sviluppo e la gestione di SAP HANA Database. Tramite la connessione con SAP HANA Server, il tool può accedere ad un database locale o remoto e l'utente, grazie a ciò, può gestire i database di SAP HANA, creare e gestire le autorizzazioni dell'utente stesso, creare nuovi o modificare i modelli esistenti dei dati.
Come si può notare nella Figura 5.3, la schermata iniziale è composta da una toolbar, nella quale sono presenti le operazioni più frequenti, una sezione che riguarda i progetti, i sistemi a cui si è collegati (database) e le rispettive repository, un SQL Editor, spazio nella quale possono essere eseguite query, una sezione dedicata ai risultati delle varie operazioni svolte dalle query e, infine, una console, nella quale vengono mostrate le informazioni riguardanti gli errori, la cronologia di esecuzione delle query, le proprietà dei vari job, ecc.
Nella sezione a lato della schermata iniziale di Eclipse, come si può notare nella Figura 5.3, vi sono tre collegamenti a tre database diversi ma collegati fra loro (H4D/H4T/H4P). Le lettere finali servono ad identificare i loro "ambienti":

- **H4D:** D sta ad indicare la parola inglese *'Development'*, ovvero sviluppo; questo è il database in cui vengono create,modificate,eliminate tabelle, viste, calculation view, ecc.

- **H4T:** T sta ad indicare la parola *'Test'*; questo database serve per testare le varie componenti che sono presenti in alcuni flussi (strumento Data Services) che sono in fase di testing.

- **H4P:** P sta ad indicare la parola inglese *'Production'*, ovvero produzione; questo è il database nel quale vengono importate, dopo varie fasi di testing, le varie componenti, grazie agli strumenti di SAP Charm e SAP Logon (descritti rispettivamente in questa sezione 5.1.2).

Eclipse è servito per la creazione delle tabelle e delle viste mancanti, le quali erano da migrare, e, successivamente, per la loro importazione all'interno dei flussi nella quale erano presenti (argomento presente in questa sezione 5.1.2). Inoltre Eclipse è servito per eseguire delle query all'interno dei database per controllare la presenza delle eventuali tabelle o viste da migrare (su H4D), per analizzare dei dati in fase di test (su H4T) e in produzione (su H4P).

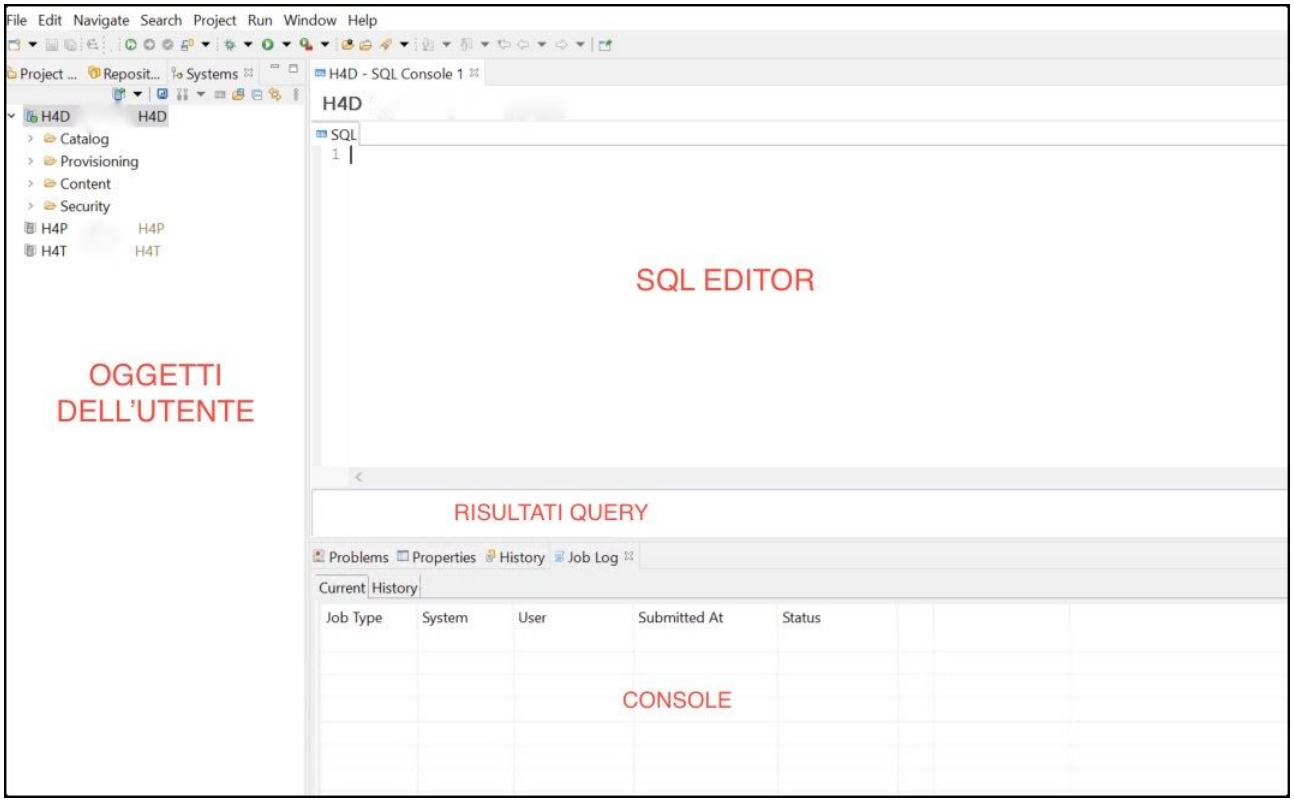


Figura 5.3: Schermata iniziale di Eclipse

Aginity

Figura 5.4: Logo Aginity

Aginity è una workbench per i *dashDB*[1] ed è un'applicazione client gratuita che permette agli utenti di interagire con la loro dashDB.
Gli utenti possono svolgere diverse attività: creare oggetti di un database, creare data query e interagire con i risultati attraverso griglie avanzate, importare ed esportare dati, registrare i risultati delle esecuzioni SQL, parametrizzare le query e molto altro.
La workbench Aginity fornisce agli utenti le seguenti funzionalità principali:

1. Importare dati di file locali in tabelle nuove o già esistenti all'interno di un database.
2. Eseguire query in SQL.
3. Generare script per oggetti di un database, eseguirli nell'editor SQL o salvarli in file.
4. Registra una cronologia delle query scritte in precedenza.
5. Ricerca l'oggetto di un database con cui vuoi lavorare.
6. Autocompleta le righe di codice SQL che si stanno scrivendo.
7. Crea template SQL che hanno parametrizzato dei valori, per rendere facile il salvataggio e il riutilizzo della logica SQL.
8. Salva frammenti di codice frequenti, pronti per essere riutilizzati.
9. Permette all'utente di far controllare e analizzare meglio i dati ottenuti dalle query attraverso ricollocazioni di colonne, ordinamenti, aggregazioni, raggruppamenti, puntamenti e funzioni dei grafici.

[1]dashDB: IBM dashDB sono dei servizi di cloud data warehousing gestiti completamente in memoria per sviluppatori, amministratori di database, analisti aziendali, data scientist, ecc. Essi offrono prestazioni e semplicità di un'applicazione di data warehouse con scalabilità e agilità di un cloud.

10. Integra la componente dei grafici.

Nella Figura 5.5 si può notare una possibile schermata iniziale della workbench che comprende: il Menu Bar, che contiene accessi ad altri menu che servono per la configurazione del sistema, al di sotto una Toolbar, che mostra le opzioni più frequenti utilizzate, l'editor SQL, che serve per scrivere le query, uno spazio nel quale vengono visionati i risultati delle query eseguite, e, infine, un Object Browser, che serve per visualizzare gli oggetti che appartengono ai database.
Lo strumento Aginity è stato utilizzato nel progetto per cercare le tabelle e le viste, presenti nei database di NETEZZA, che dovevano essere migrate; una volta trovate, viene copiato il codice in NETEZZA SQL della definizione della tabella o della vista per essere, in seguito, convertito, grazie anche allo strumento NetezzaToHdTable (argomento presente in questa sezione 5.1.2), in codice HANA SQL, da importare nello strumento Eclipse (argomento presente in questa sezione 5.1.2).

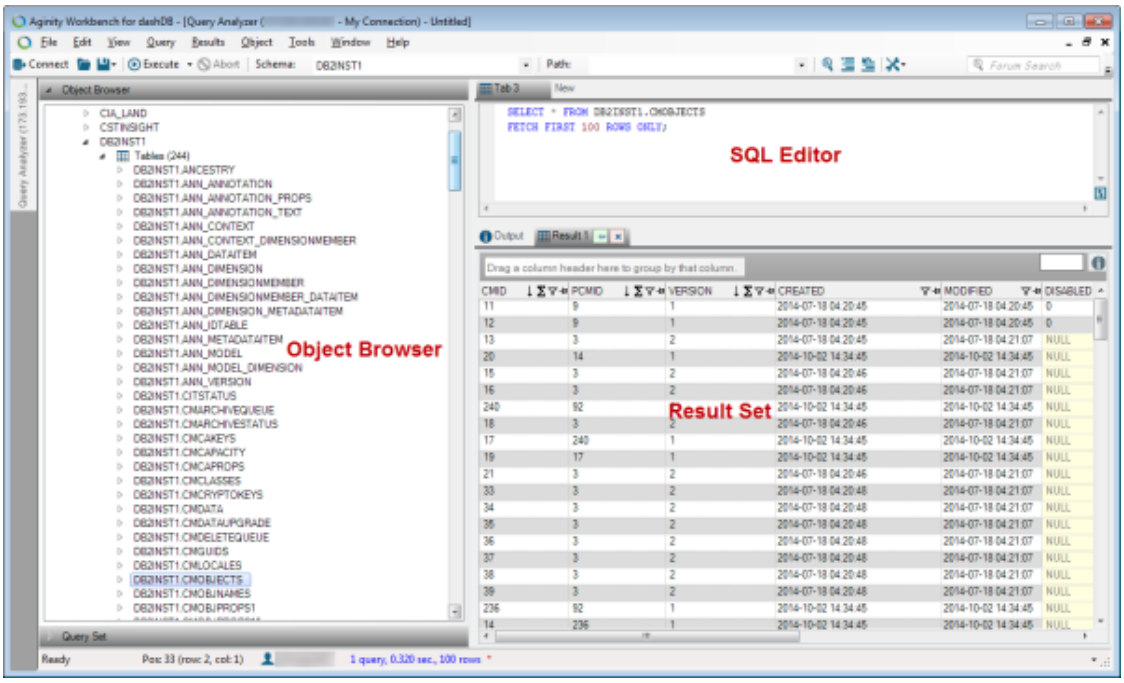


Figura 5.5: Schermata di Aginity

SAP Charm e SAP Logon

(a) *Logo Charm* **(b)** *Logo SAP Logon*

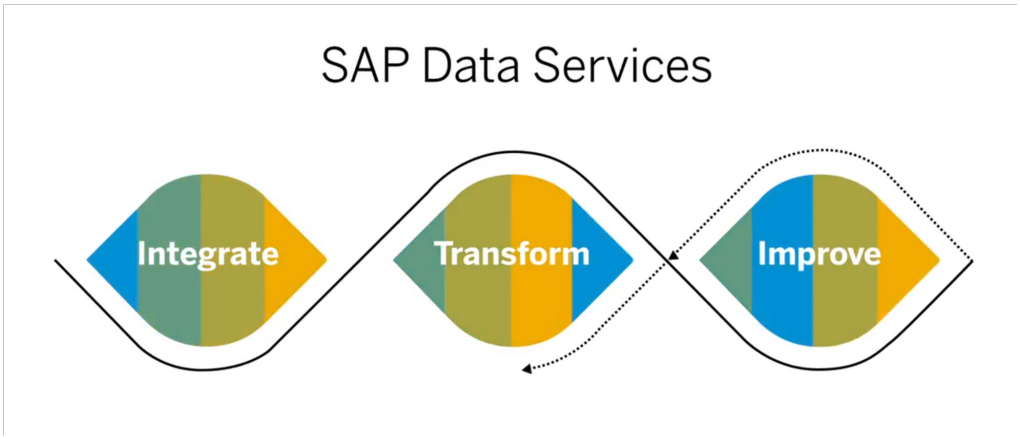
Figura 5.6: Loghi dei due strumenti

Questi due strumenti sono collegati fra loro e rendono un ambiente unico i tre database presenti in Eclipse (H4D - H4T - H4P), collegandoli fra loro; entrambi rappresentano il mondo SAP, e quindi il mondo legato alla gestione delle risorse aziendali e delle pianificazioni delle attività. In particolare, SAP Charm (il secondo termine fa riferimento alle parole inglesi: *Change Request Management*, ovvero la gestione delle richieste di modifica) è uno strumento che gestisce le attività eseguite durante il passaggio da progettazione a test e da test al sistema di produzione; esso consente di tenere traccia delle richieste di modifica e delle richieste di trasporto nel sistema di gestione delle modifiche nell'intera soluzione aziendale.
Durante il passaggio dalla fase di progettazione a quella di tes, serve lo strumento SAP Logon; quest'ultimo è un programma installato localmente che viene utilizzato per accedere ad un sistema SAP. Dopo essersi connessi al sistema B4D di sviluppo, bisogna selezionare le tabelle/viste che sono prese in causa nella richiesta per spostarle in fase di testing. Infine, per passare dalla fase di test alla fase di produzione bisogna comunicare tramite mail la richiesta creata su Charm ai tecnici che se ne occupano, averne l'approvazione e, per ultimo, l'importazione.

SAP Data Services

Figura 5.7: Logo SAP Data Services

SAP Data Services è un applicativo software per il **data integration** e il **data transformation**; esso permette all'utente di sviluppare ed eseguire workflow che acquisiscono dati da sorgenti predefinite, quali applicazioni, web services, flat-files, databases, ecc, e di combinare, trasformare, ridefinire quei dati per poi ricaricarli su target definiti.



Oltre alle classiche funzioni di data transformation, ad esempio join, filtri, aggregazioni, Data Services offre funzionalità avanzate per la text analysis, il data profiling e auditing e alcune operazioni di data quality come il matching e il geocoding. Esso, inoltre, supporta il changed-data capture, capacità importante per ottenere dati in input per i sistemi di data-warehousing. Infine Data services supporta sia i classici processi batch che servizi real-time, che permettono un aggiornamento automatico dei dati da sorgente.

Tra gli oggetti più trattati durante il progetto e più importanti che si trovano in Data Services vi sono: i **Job**, i **Workflow**, i **Dataflow**, gli **Scripts** e i **Datastore**. Questi elementi possono essere creati recandosi nella Local Object Library o usando i pulsanti appositi messi a disposizione dall'interfaccia di Data

Services:

- Un Job è l'unita basilare di Data Services; il job può essere eseguito, schedulato, può contenere script, workflow e dataflow ma non può però contenere direttamente dei flussi operativi.

Figura 5.8: Icona Job

- Un Workflow è la seconda unità di Data Services; esso è considerabile come un elemento organizzativo, in quanto non può contenere flussi operativi, ma solo Dataflow e altri Workflow innestati. Esso, infatti, può contenere uno o più Dataflow al suo interno; ciò ci permette di eseguire diversi Dataflow in serie e in parallelo contemporaneamente.

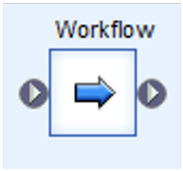


Figura 5.9: Icona Workflow

- Un Dataflow è la vera unità operativa di Data services; solo in esso possono essere utilizzati i vari componenti di query e tutti i componenti custom per realizzare un vero e proprio flusso ETL.

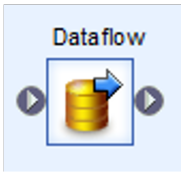


Figura 5.10: Icona Dataflow

- Uno Script è un insieme di righe di codice in una procedura.

Figura 5.11: Icona Script

- Un Datastore è un canale logico che collega Data Services ai database di origine e di destinazione del datastore.

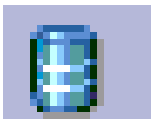


Figura 5.12: Icona Datastore

Inoltre, un ulteriore oggetto è il componente **query**, oggetto base per compiere la maggior parte delle operazioni con i dati; esso ha la funzione principale di mappare i campi delle tabelle sorgenti in quelli delle tabelle di destinazione, trascinandoli dal campo sorgente al campo destinazione.

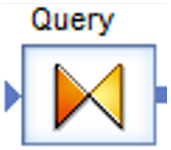


Figura 5.13: Icona Query

Come si può notare nella Figura 5.15, con il componente query è anche possibile applicare dei filtri (clausola where), delle distinct (select), delle group by (raggruppamento), delle order by (ordinamento) e creare delle join fra più tabelle.
Nella Figura 5.14 viene mostrato un esempio di schermata iniziale di Data Services; facendo riferimento ai numeri presenti, si possono notare:

1. **Tool Bar:** barra degli strumenti composta da diverse operazioni che vengono svolte frequentemente, quali il salvataggio, l'apertura di una directory, l'esecuzione di un job, la convalida di un workflow/dataflow, ecc.

2. **Project Area:** area che contiene il progetto corrente, che include Job, Workflow e Dataflow.

3. **Work Space:** area della finestra dell'applicazione in cui definiamo, visualizziamo e modificiamo gli oggetti.

4. **Local Object Library:** libreria che contiene oggetti del repository locale, come trasformazioni, lavoro, flusso di lavoro, flusso di dati, ecc. Essa si differenzia dalla Central repository poichè quest'ultima viene utilizzata per le configurazioni dei repository sui job server, la gestione della sicurezza, il controllo delle versioni e la condivisione degli oggetti, invece la Local repository è dell'utente e può creare, modificare ed eseguire job, workflow e dataflow.

5. **Tool Palette:** palette di strumenti che con i suoi bottoni permette di aggiungere nuovi oggetti all'area di lavoro.

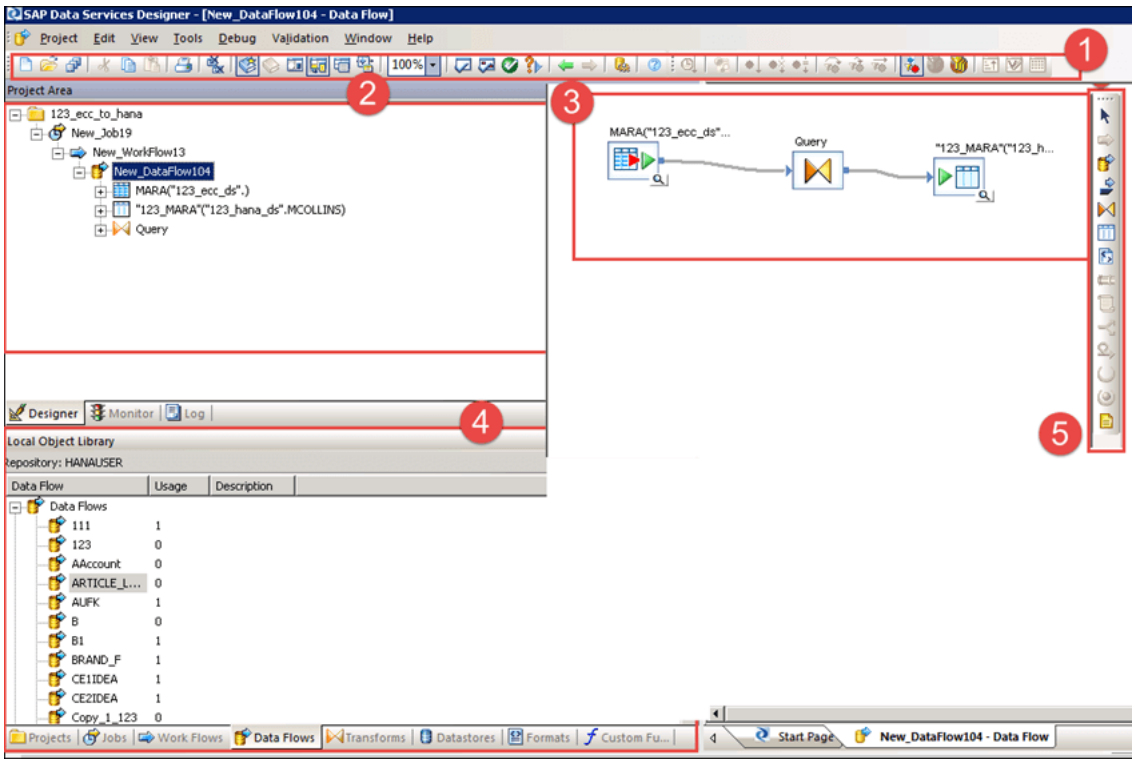


Figura 5.14: Esempio di schermata iniziale

Lo strumento Data Services è servito nel progetto per esportare i flussi che servivano a popolare le tabelle presenti nel file QlikView e che, quindi, erano da migrare e per effettuare la loro migrazione nei server e database di HANA.

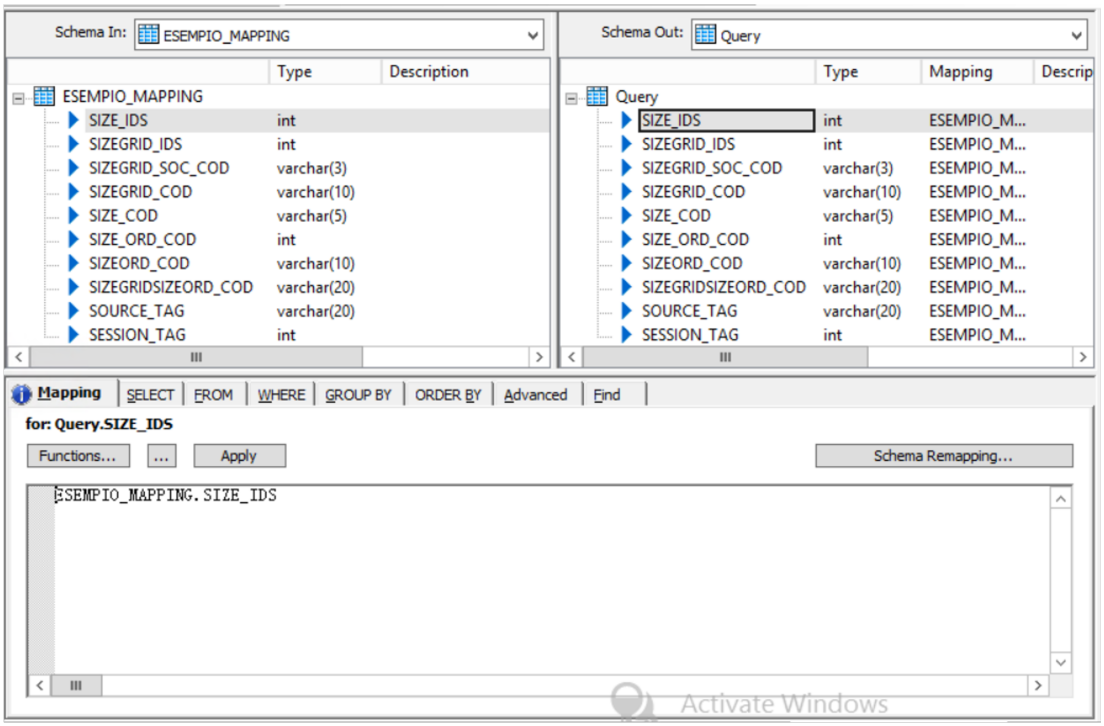


Figura 5.15: Esempio di Query

NetezzaToHdTable

Il programma NetezzaToHdTable è un programma Java creato dal tutor che mi ha seguito durante il tirocinio; questo programma serve per convertire la creazione solo di tabelle da linguaggio NETEZZA SQL a linguaggio HANA SQL. Per eseguirlo, da console bisogna inserire un comando composto dal percorso del file di destinazione, che deve essere convertito, e dallo schema presente in HANA di cui farà parte. Come output si ottiene la conversione del codice di creazione della tabella che bisogna importare nello strumento Eclipse (argomento presente in questa sezione 5.1.2).

5.2 Svolgimento

Nello svolgimento del progetto sono state eseguite determinate operazioni per giungere allo scopo finale richiesto, ovvero, come già specificato precedentemente, effettuare una migrazione dei flussi che alimentano una dashboard di un cliente importante del settore Manifacturing. Per migrazione si intende sostituire nei flussi, di cui si ha bisogno, ogni elemento (tabella, vista, script di codice) e riferimento ai database di NETEZZA con il relativo elemento e riferimento al database di HANA. In questa sezione, in particolare, vi è una delucidazione delle operazioni svolte nel dettaglio, descritte in ordine.

5.2.1 *Prima analisi in QlikView*

In questa prima operazione viene analizzato il file QVW di QlikView, prendendo nota in un file excel delle tabelle che vi sono presenti in ogni script di codice; ogni script ha la sua funzione: vi è lo script che si occupa di analizzare la tabella degli ordini, la tabella del fatturato, la tabella delle consegne, ecc.

5.2.2 *Esportazione ed Importazione job di riferimento in Data services*

Successivamente bisogna esportare i job presenti nella repository del server di NETEZZA, che servono per alimentare le tabelle scritte nella operazione precedente e che, quindi, sono da migrare, ed importarli nella Local Repository nel server di HANA.

5.2.3 *Controllo su Eclipse in H4D*

Una volta dopo aver esportato ed importato i job, occorre controllare nel database H4D la presenza delle tabelle/viste che vi sono nei flussi importati attraverso l'esecuzione di query nella workbench. Possono risultare due i casi in cui ci si ritrova:

1. La tabella/vista è già presente in H4D poiché è già stata migrata in passato per altri flussi di appartenenza (si passa direttamente alla sottosezione 5.2.7).

2. La tabella/vista non è presente e quindi bisogna convertire la sua definizione in linguaggio HANA SQL (proseguire con sottosezione 5.2.4).

5.2.4 *Cercare il codice di creazione delle tabelle/viste da migrare su Aginity*

Questa operazione è stata eseguita nel caso in cui la tabella/vista in questione non è presente nel database H4D su Eclipse.
Attraverso l'esecuzione di una query ci si accerta la sua presenza nei database di NETEZZA:

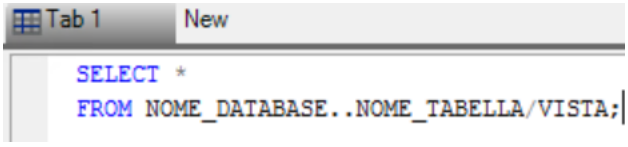


Figura 5.16: Esempio di query su Aginity

Successivamente aprire lo script DDL (*Data Definition Language*), ovvero lo script che serve a generare una tabella/vista, copiarlo e incollarlo in un file, salvandolo con estensione .sql .

5.2.5 *Convertire manualmente o tramite il programma NetezzaToHdTable il codice ottenuto*

Nel caso in cui si tratti di una tabella, si può convertire il suo codice di creazione attraverso il programma java NetezzaToHdTable. Da console bisogna inserire questo comando per eseguire il programma in modo corretto:

java NetezzaToHdTable <percorso file di destinazione da convertire> <schema Hana di appartenenza>

Successivamente, si avrà in output un codice che dovrà essere importato in H4D su Eclipse (Figura 5.17).

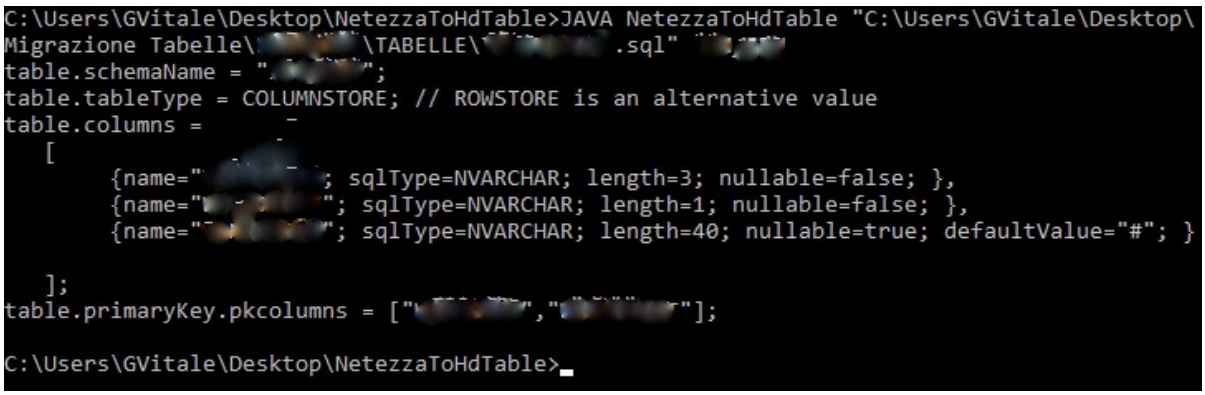


Figura 5.17: Esempio di esecuzione

Nel caso in cui si tratti di una vista, bisogna convertire a mano il suo script di creazione, andando a sostituire alcune sue funzioni di linguaggio NETEZZA SQL con funzioni di HANA SQL e sostituire il nome delle tabelle di NETEZZA presenti con quelle presenti in HANA. Può succedere che alcune tabelle a cui si riferisce la vista non siano già presenti in HANA e, quindi, bisogna importare anch'esse con lo stesso procedimento utilizzato sopra.

5.2.6 *Importazione e attivazione delle tabelle/viste in H4D su Eclipse*

Una volta ottenuto il codice di creazione della tabella/vista bisogna crearle nel database di HANA dedicato allo sviluppo (H4D) e importare il codice.

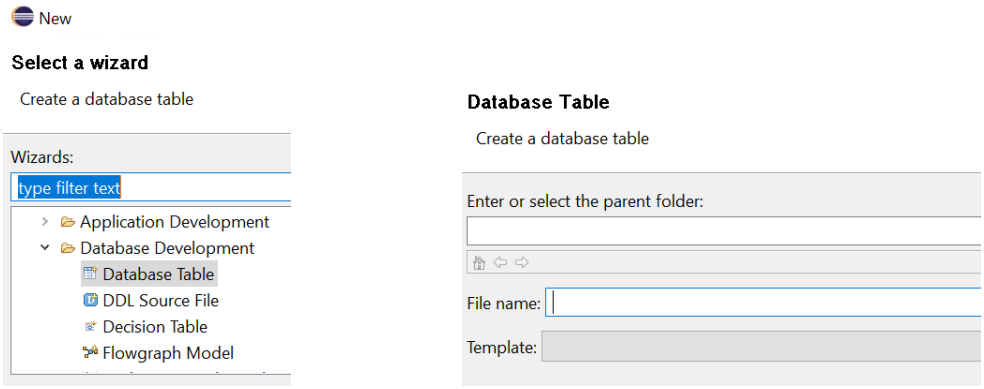


Figura 5.18: Creazione tabella nel database

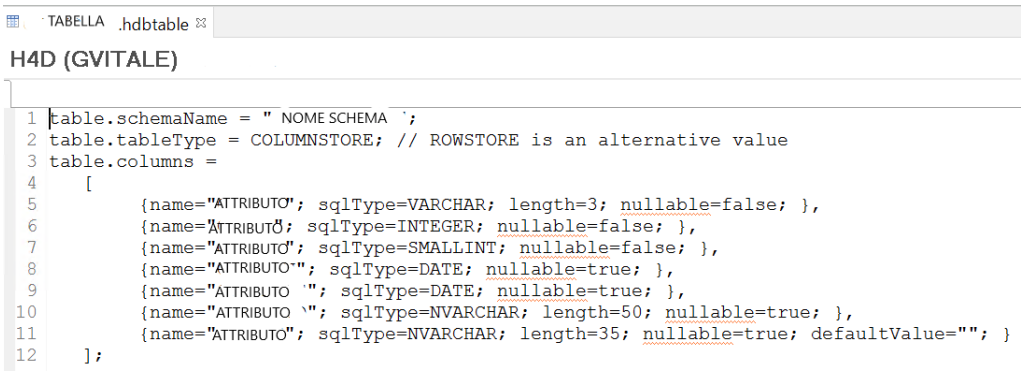


Figura 5.19: Importazione del codice di creazione della tabella/vista

Dopo aver effettuato l'importo, si deve attivare la tabella/vista per controllare se la sintassi è giusta: nel caso fosse giusta, viene attivata e può essere utilizzata, nel caso fosse errata, bisogna correggere la sintassi del codice.

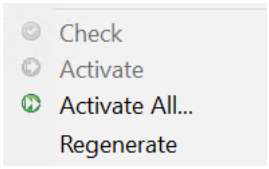


Figura 5.20: Attivazione della tabella/vista

5.2.7 *Importazione delle tabelle/viste in Data services e nei flussi di appartenenza*

Durante questa operazione viene importata la tabella/vista nel datastore di HANA su Data Services, per poi poterla utilizzare all'interno dei job. Questa tabella/vista, presente

nel database di HANA, andrà a sostituire in un job la sua stessa tabella/vista presente nel database Netezza.

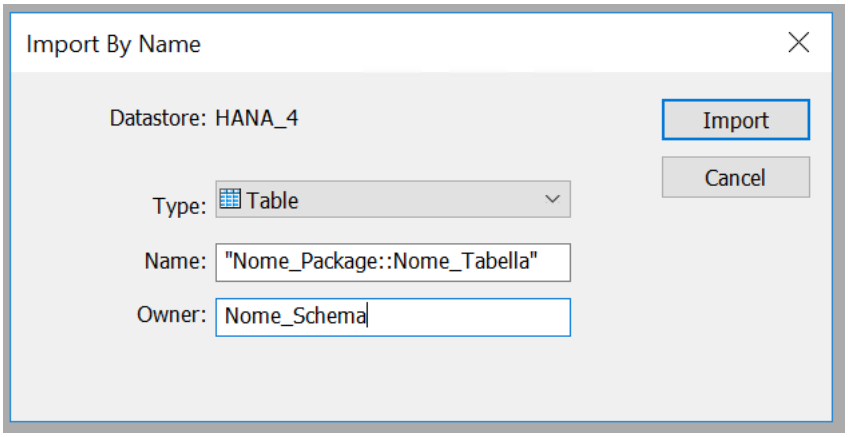


Figura 5.21: Esempio di importazione.

5.2.8 *Fare richiesta su Charm e utilizzo di SAP Logon*

L'operazione successiva è la richiesta su Charm. Come è stato spiegato nella sezione degli strumenti, Charm viene utilizzato per gestire le attività eseguite durante il passaggio dallo sviluppo al test, fino al sistema di produzione finale. Fasi iniziali di una richiesta:

1. **Created**: quando la richiesta viene creata bisogna inserire una descrizione, il manager, il developer, il tester, la priorità e il ciclo a cui appartiene (Figura 5.22);

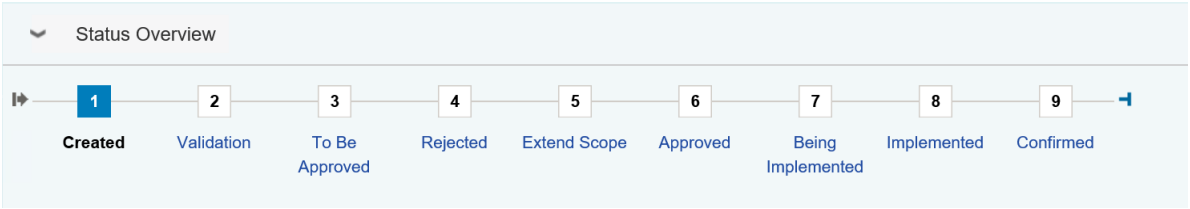


Figura 5.22: Fasi della richiesta

2. **Extend Scope**: una volta aver salvato e validato la richiesta, bisogna inserire uno scopo, ovvero il tipo di richiesta che viene fatta ('Normal change');
3. **Being Implemented**: dopo aver aggiunto lo scopo, passare in fase di sviluppo;
4. **In development**: nella fase di sviluppo della richiesta bisogna passare alla normal change, tramite il suo id che viene generato, e creare una richiesta di trasporto (Figura 5.23);

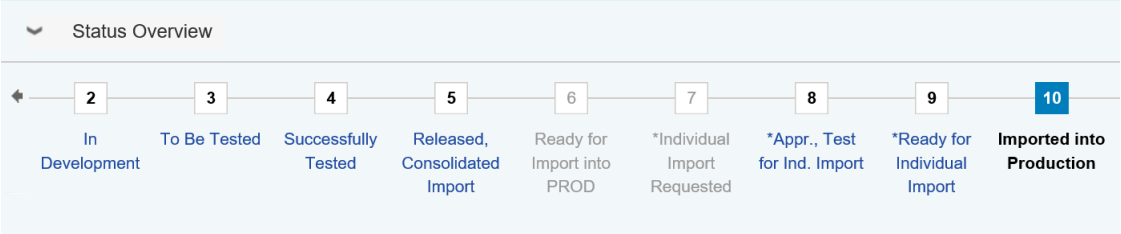


Figura 5.23: Fasi della richiesta (normal change)

5. **Utilizzo di SAP Logon**: in questa fase entra in gioco SAP Logon. Questo strumento permette di selezionare e importare le tabelle/viste nella fase di test;

6. **To Be Tested**: portare la normal change in fase di testing.

5.2.9 *Testing dei job in ambiente di Test*

Una volta dopo aver messo la richiesta in fase di testing, si possono testare i job che si stanno migrando, per verificare eventuali errori che possono risultare. Nel caso in cui riportino errori, bisogna trovare la causa e sistemarla; nel caso in cui funzioni tutto correttamente, si può portare in 'avanti' la richiesta di trasporto che è in corso.

5.2.10 *Richiesta di importazione in Produzione*

Riprendendo le fasi descritte nella sottosezione 5.2.8:

1. **Succesfully Tested**: una volta dopo aver verificato l'esecuzione dei job, si passa alla fase di test raggiunto;

2. **Ready for individual Import**: in questa fase finale per realizzare l'importazione in produzione c'è bisogno del consenso degli amministratori che se ne occupano tramite mail, nella quale bisogna specificare l'id della normal change e le tabelle/viste che bisogna importare;

3. **Imported into Production**: con il consenso degli amministratori la richiesta viene, infine, importata dai tecnici in produzione.

5.2.11 *Testing dei job in ambiente di Produzione*

Importata la richiesta, come il test effettuato nella sottosezione 5.2.9, si possono testare i job in Produzione per verificare eventuali errori. Nel caso in cui riportino errori, bisogna trovare la causa e sistemarla; nel caso in cui funzioni tutto correttamente, si possono

aggiungere i job alla central repository di appartenenza, per eventuali successive schedulazioni. Quando un job viene schedulato significa che viene impostata la sua esecuzione con una frequenza stabilita (ad esempio: ogni giorno alle 8:00, ecc).

5.2.12 *Correzione degli script su QlikView*

Una volta conclusa la fase di test in produzione per verificare la correttezza dei flussi migrati, bisogna modificare ogni script di codice del file QVW su QLikView. Le modifiche più frequenti che sono state apportate sono:

- Cambiare i riferimenti nella clausola FROM alle tabelle presenti nei database Netezza con le tabelle migrate nel database Hana.
- Modificare alcune funzioni non presenti in Hana SQL (ad esempio: la funzione NVL(), funzione che viene utilizzata per sostituire il valore NULL con un altro valore, con la funzione COALESCE());
- Aggiungere dei raggruppamenti (GROUP BY + HAVING) per attributi presenti nella clausola SELECT.
- Modificare il collegamento al database di riferimento di Netezza con il collegamento al database di produzione di Hana.

5.2.13 *Test degli script sulla console di QlikView*

Dopo aver modificato gli script bisogna testarli sulla console di sviluppo di Qlikview; in pratica, si creano dei task che si riferiscono al file QVW, uguali a quelli presenti in produzione, che servono a far partire ogni singolo script tramite un parametro che gli viene passato. Ogni singolo script che al proprio inizio soddisferà questa condizione (IF) con il parametro che viene passato tramite task, viene eseguito; alla fine degli script si creeranno dei file QVD contenenti i dati che si ottengono dal codice eseguito. Nel caso in cui, dopo le varie esecuzioni delle task, nessuna di esse termina con errori, bisogna mettere a confronto i file QVD generati in sviluppo con quella in produzione, in modo da poter controllare se tutto funziona correttamente e non ci siano perdite di dati.

5.2.14 *Sostituire il file QVW modificato con quello in produzione*

In questa operazione finale, dopo aver verificato l'esecuzione corretta delle task nella console di sviluppo, bisogna sostituire il file QVW modificato inizialmente con quello presente nella cartella di produzione; in questo modo, alla conclusione dell'esecuzione delle task schedulate, il risultato finale è la dashboard migrata che può essere utilizzata dal cliente che ne ha fatto richiesta.

Capitolo 6

Validazione

In questa sezione viene mostrato il risultato finale delle operazioni svolte attraverso una serie di foto che rappresentano le schermate principali della dashboard (N.B. le foto che sono state utilizzate per mostrare il prodotto finale hanno alcuni dati oscurati per tutelare il cliente che ha richiesto il progetto). Il risultato finale è la dashboard migrata alla tecnologia Hana e, di seguito, vi sono alcuni screenshot.

6.1 Sezione Home

La sezione Home è una pagina, come si può notare nella Figura 6.1, dedicata a mostrare i dati degli ordini netti e lordi dei resi, i clienti attivi con portafoglio Wholesale e i loro ordini, i dati di vendita di occhiali da sole e occhiali da vista, con la possibilità di filtrare le varie informazioni con diversi periodi:

- YTD: Year To Date, ovvero da un anno fa ad oggi;
- QTD: Quarter To Date, ovvero da un trimestre fa ad oggi;
- MTD: Month To Date, ovvero da un mese fa ad oggi;
- L4W: Last 4 Weeks, ovvero le ultime quattro settimane;
- LW: Last Week, ovvero l'ultima settimana;
- CW: Current Week, ovvero la settimana corrente.

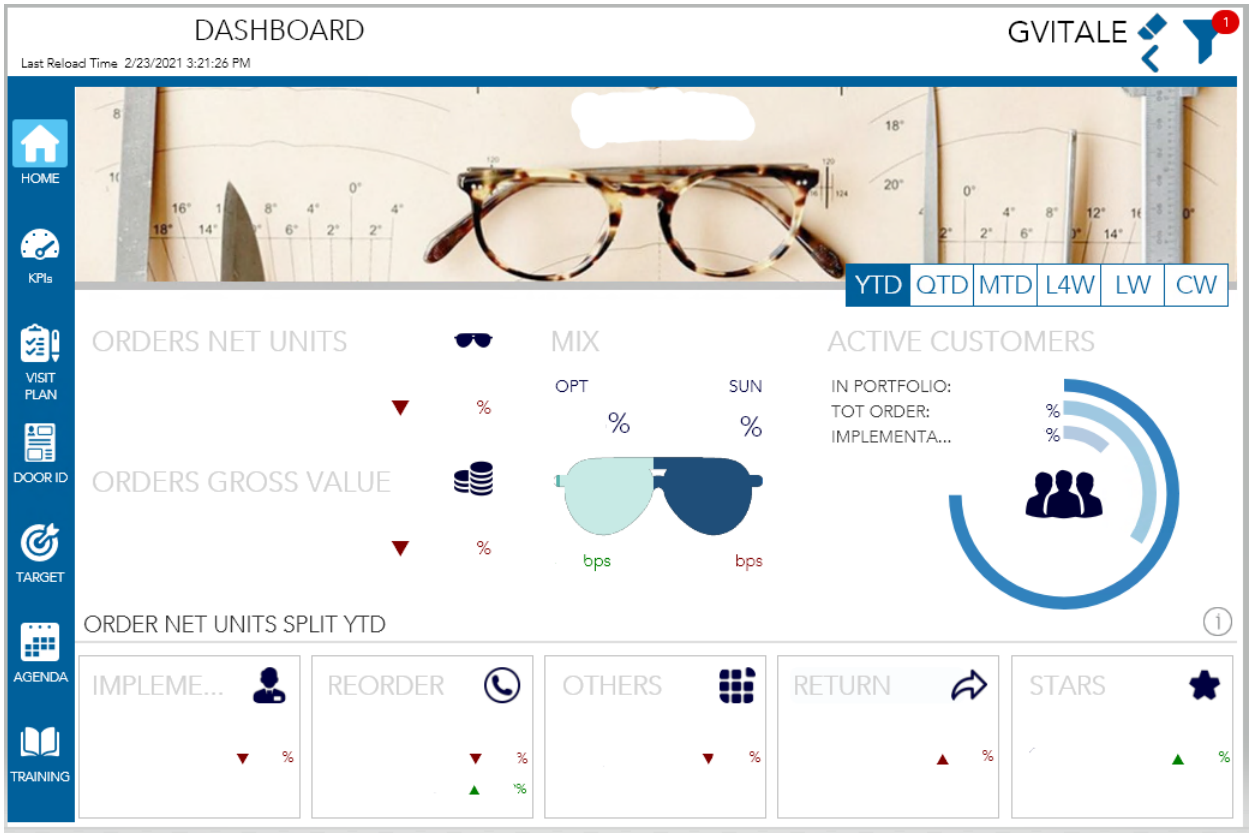


Figura 6.1: Sezione Home

6.2 Sezione KPIs

La sezione KPIs (*Key Permorfance Indicator*, ovvero indicatore chiave delle performance) è una pagina, come si può notare nella Figura 6.2, nella quale l'utente può selezionare metriche ed attributi da controllare, ottenendo un riepilogo degli stessi, con la possibilità di filtrarli per periodi.

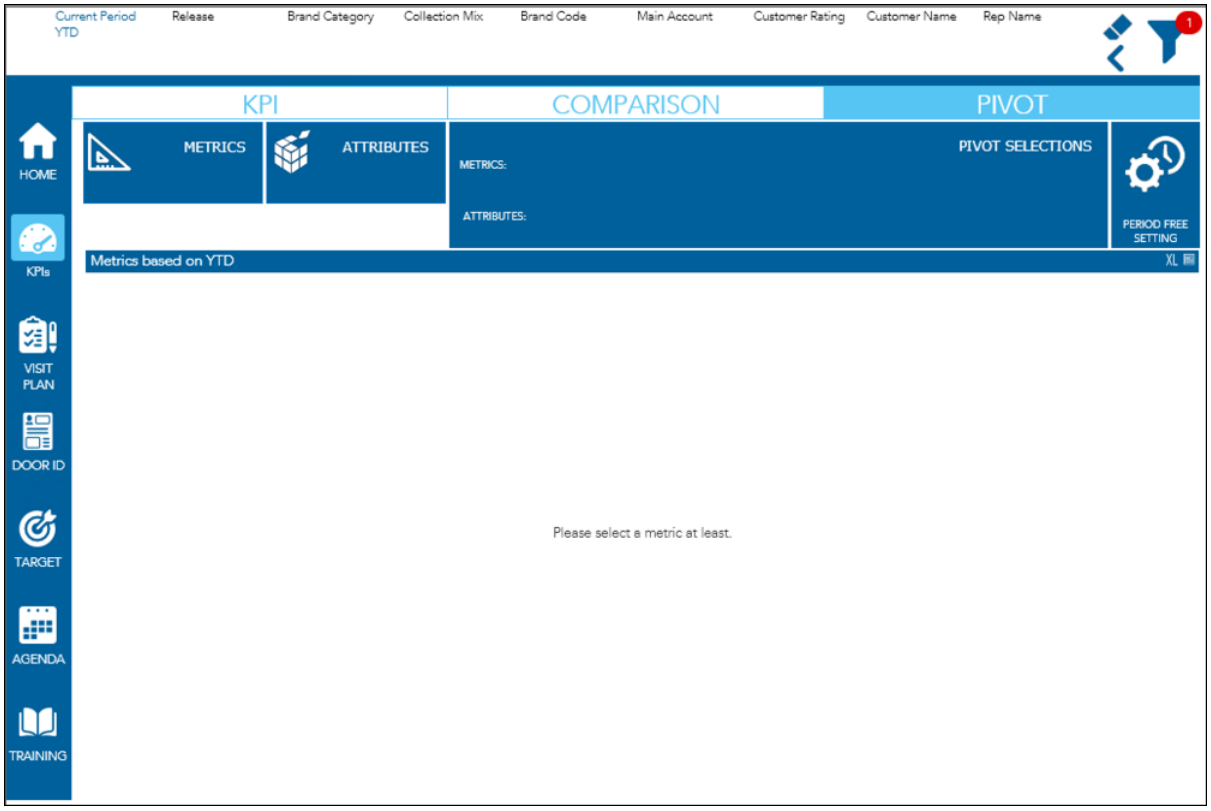


Figura 6.2: Sezione KPIs

6.3 Sezione Visit Plan

La sezione Visit Plan è una pagina, come si può notare nella Figura 6.3, che mostra i dati delle visite effettuate dagli agenti del cliente della dashboard nei vari negozi per *'forza vendita'*, con la possibilità di filtrare per vari elementi legati all'agente.

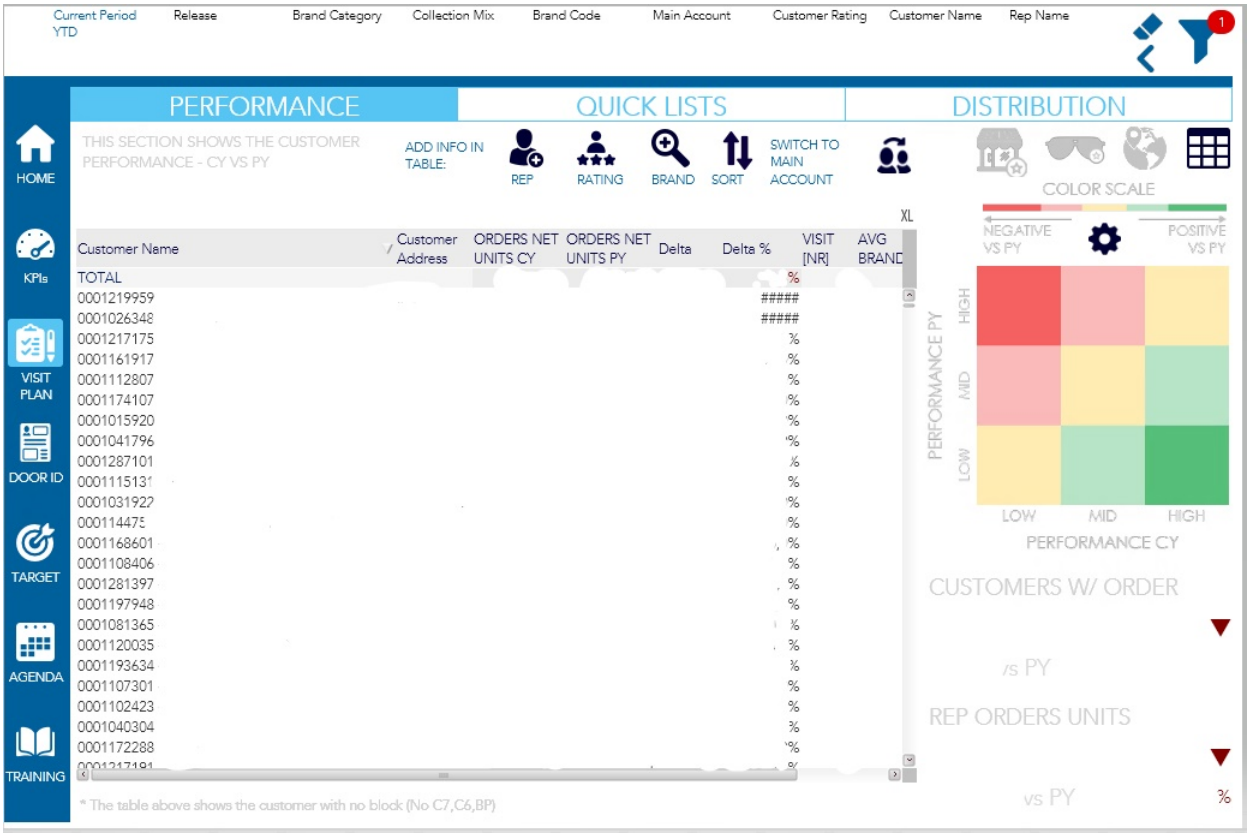


Figura 6.3: Sezione Visit Plan

6.4 Sezione Door Id

La sezione Door Id è una pagina, come si può notare nella Figura 6.4, che mostra all'utente, che si riferisce al Main Account, i dati delle varie *'porte'*, ovvero i vari negozi delle catene di aziende. Essa, quindi, rappresenta un registro dei clienti, mostrando le anagrafiche dei clienti.

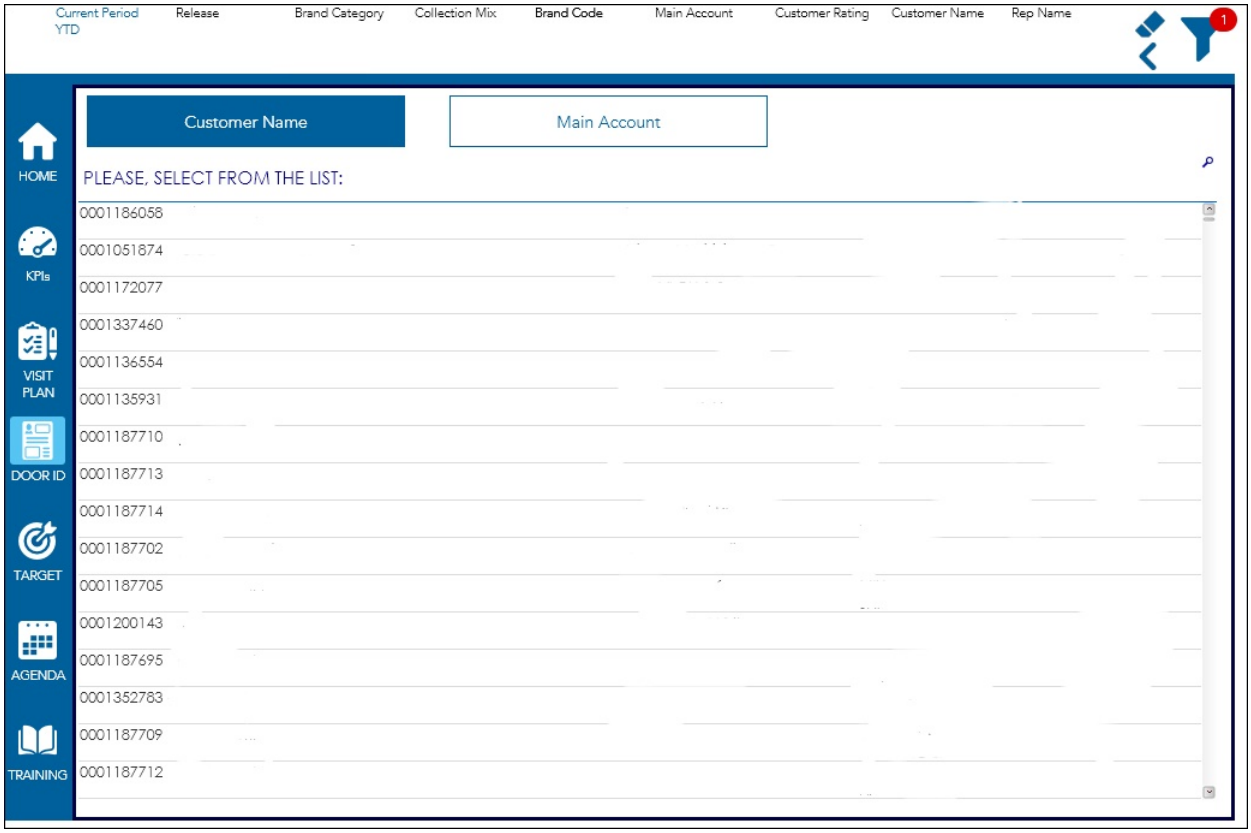


Figura 6.4: Sezione Door Id

6.5 Sezione Target

La sezione Target è una pagina, come si può notare nella Figura 6.5, che permette all'utente di selezionare i vari agenti che vi sono registrati per controllare i loro dati legati agli obiettivi pianificati nel tempo (N.B. I sei nomi e cognomi degli agenti presenti nello screenshot sono nomi e cognomi fittizi).

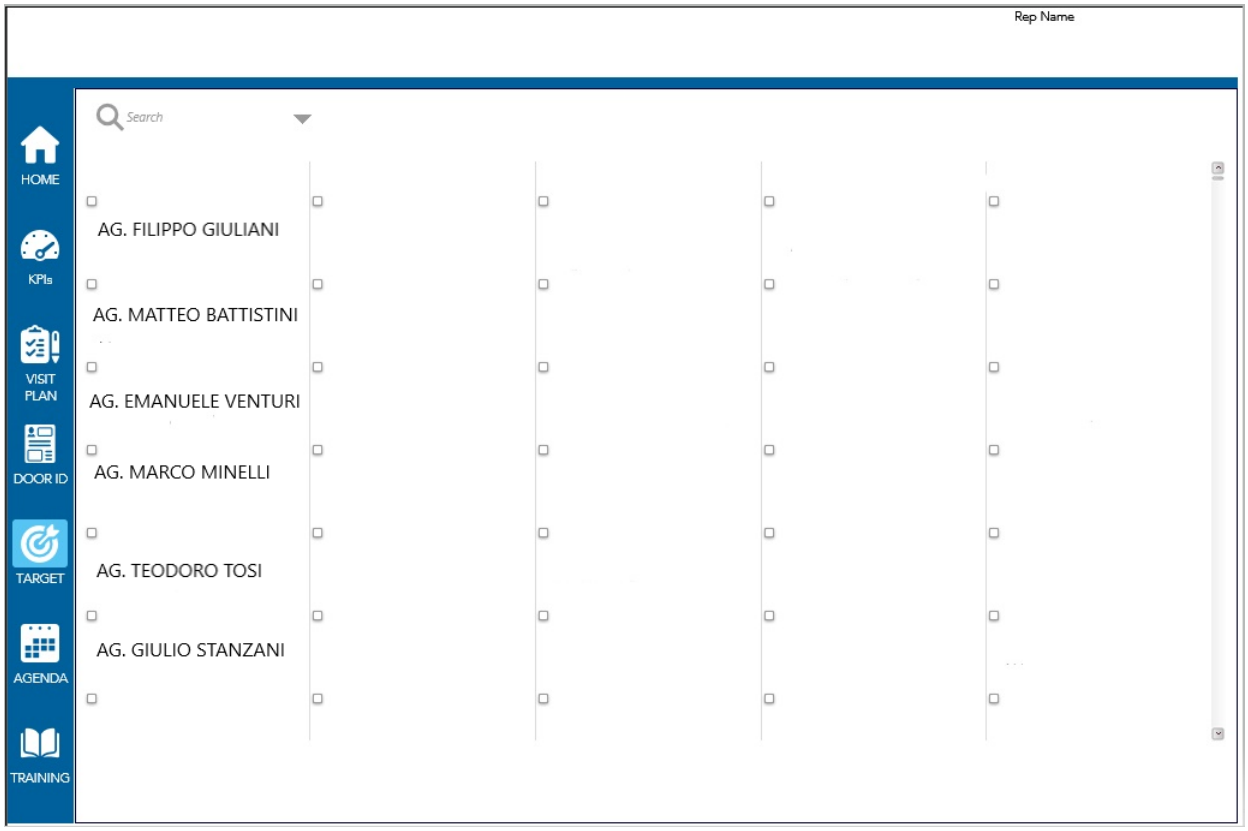


Figura 6.5: Sezione Target

6.6 Sezione Agenda

La sezione Agenda è una pagina, come si può notare nella Figura 6.6, che viene utilizzata come calendario delle visite, con i relativi dati di vendita e di ordini, dei vari agenti che hanno compiuto e che devono compiere (N.B. I sei nomi e cognomi degli agenti presenti nello screenshot sono nomi e cognomi fittizi).

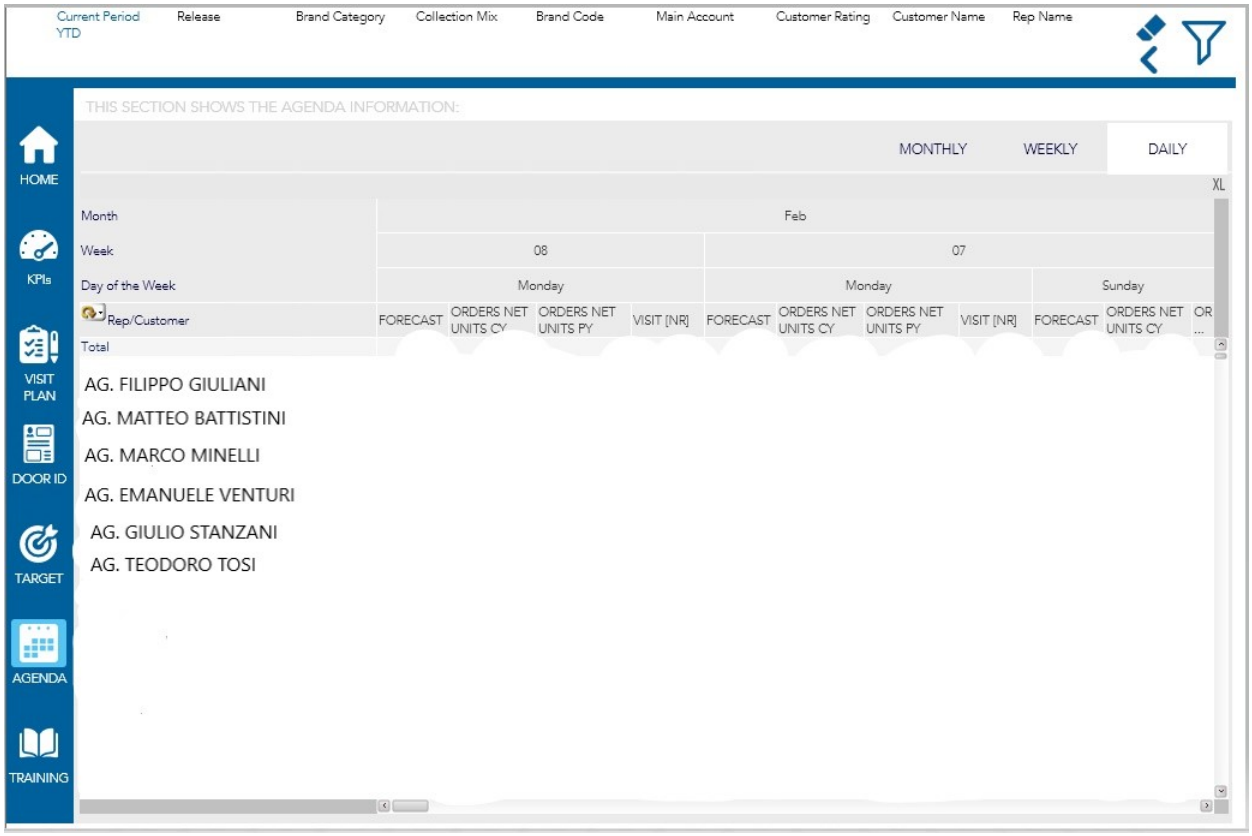


Figura 6.6: Sezione Agenda

6.7 Sezione Training

La sezione Training è una pagina, come si può notare nella Figura 6.7, legata all'addestramento da parte del cliente della dashboard dei clienti; vengono mostrati i dati sotto percentuale e non di 'porte' aperte, ovvero negozi nuovi che partecipano, corsi completati, utenti attivi, utenti registrati, utenti che hanno effettuato l'accesso e conoscenza globale e del brand trasmessa.

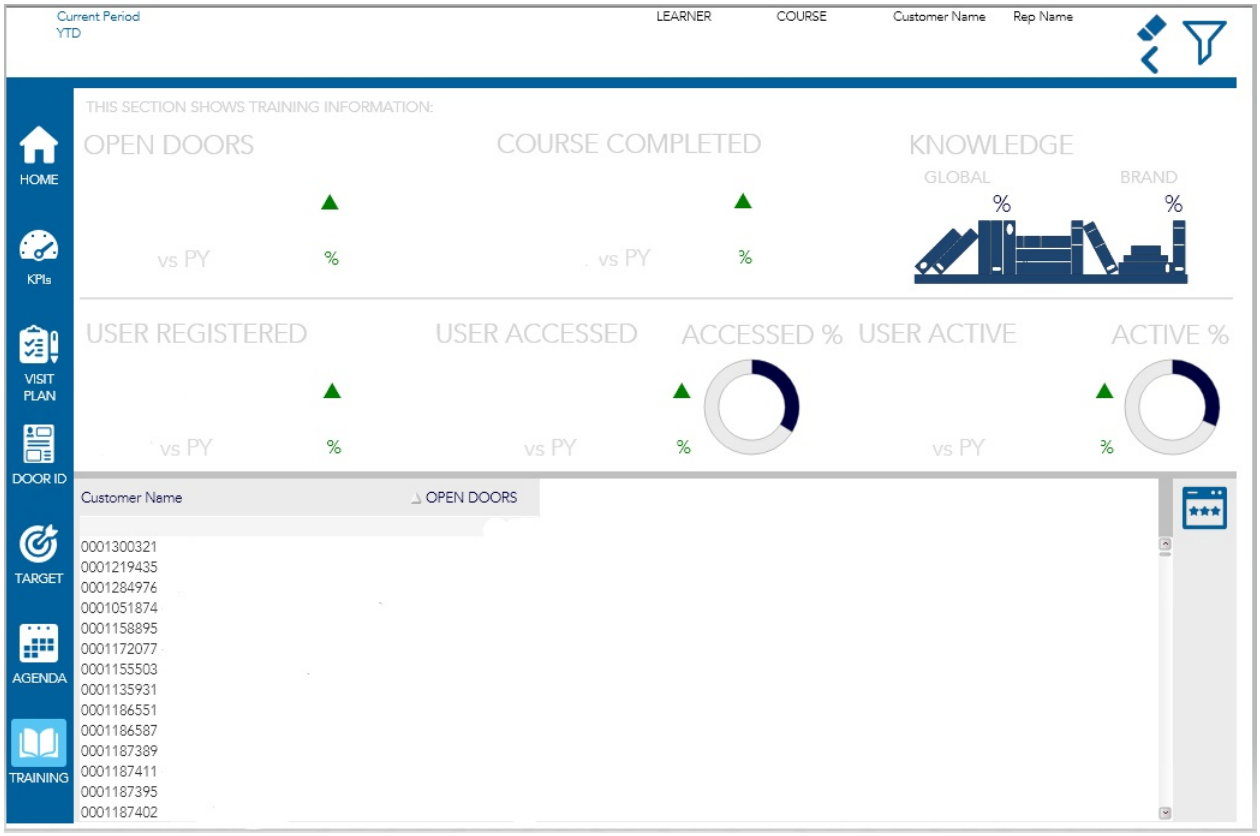


Figura 6.7: Sezione Training

Parte IV

Conclusione e sviluppi futuri

Conclusione della tesi e sviluppi futuri del progetto

Alla fine del progetto di tesi elaborato si può denotare l'importanza che hanno i Big Data nel campo aziendale e l'efficienza della nuova tecnologia nella quale si è svolta la migrazione. In particolare, si può osservare quanto un'enorme mole di dati possa essere importante per un'azienda nell'effettuare decisioni di business, nel controllare utenti appartenenti all'azienda, nel controllare dati giornalieri di vendita e di produzione, ecc. Al giorno d'oggi, il compito dei sistemi informativi è quello di estrarre una nuova conoscenza dai dati, al fine di aiutare i manager a prendere decisioni migliori e in un lasso di tempo ridotto. Come affermato nel primo capitolo, i Big Data permettono di effettuare nuove tipologie di analisi su nuovi dati, sfruttando caratteristiche come volume, varietà, velocità, veridicità, variabilità dei dati e generando valore, elemento di interesse dell'azienda. Allo stesso tempo, l'altro argomento della tesi che rende maggiore il valore dei Big Data sono i Data Warehouse. Oggigiorno aziende di tutte le dimensioni sono sempre alla ricerca di strumenti, strategie e soluzioni per migliorare la propria capacità di essere competitivi nei confronti della concorrenza e il valore aggiunto che può garantire un Data Warehouse è l'elevato livello di qualità delle analisi possibili per la ricerca di migliori decisioni di business.
In conclusione, il prodotto finale del progetto elaborato è una migrazione tecnologica di una dashboard per un cliente del settore Manufacturing. Possibili sviluppi futuri legati a questo progetto di tesi includono:

- Manutenzioni a livello di flussi: inserimento di tecniche di push down, ovvero tecniche per cui la fase di trasformazione dei dati venga totalmente eseguita dal database sottostante, non forzando perciò la memoria di Data Services (strumento descritto in questa sezione 5.1.2). Esse garantiscono prestazioni ai flussi notevolmente migliori, prevenendo i fallimenti di quest'ultimi dovuti ai limiti di memoria dello strumento Data Services.

- Manutenzioni a livello di database: inserimento di nuovi campi di KPIs, ovvero indici dell'andamento di un processo aziendale, e di nuove dimensioni in tabelle, per l'esecuzione di analisi future più dettagliate, con più informazioni a disposizione

dell'azienda. Inoltre, vi possono essere operazioni di inserimento di nuove tabelle con nuovi campi di fatti e dimensioni.

- Manutezione a livello di dashboard: inserimento di nuovi script che analizzano le tabelle e generano la dashboard e, quindi, nuove pagine che mostrano determinati dati, con determinati filtri, grafici, tabelle, voluti dall'azienda.

Parte V

Bibliografia e Sitografia

Bibliografia e Sitografia

[1] (Capitoli 1-2-3) *Star schema*, pdf creato dall'azienda Iconsulting.

[2] (Capitoli 4-5) *Data Mining, Concepts and Techniques*, 3° Edizione, Jiawei Han, Micheline Kamber, Jian Pei (2012).
http://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheli-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf

[3] https://www.sap.com/products/hana/what-is-sap-hana.html?btp=99a18fd0-61c3-4747-971b-6ccb3e151083#overview

[4] https://dl.acm.org/doi/abs/10.5555/2093889.2093965

[5] https://mindmajix.com/definition-and-advantages-of-qvds-in-qlikview

[6] https://help.qlik.com/it-IT/qlikview/April2020/Content/QV_HelpSites/what-is.htm

[7] https://blogs.sap.com/2014/09/22/basic-procedure-for-change-management-ch

[8] https://www.apprisia.com/blog/sap-basis-2/charm-change-request-management-solution-manager/

[9] https://www.guru99.com/sap-ds-sap-data-services-in-sap-hana.html#:~:text=SAP

[10] https://help.sap.com/saphelp_nw73/HELPDATA/EN/91/cdbc2bfd6d451d956067f93928d981/content.htm?no_cache=true

[11] https://innovaformazioneblog.altervista.org/cose-sap-hana/?doing_wp_cron=1610621418.2960579395294189453125

[12] https://sapprofession.com/cosa-e-sap-hana/

[13] https://answers.sap.com/questions/8913711/netezza-vs-hana.html

[14] https://www.cloudtalk.it/big-data-esempi/

[15] https://www.sas.com/it_it/insights/big-data/what-is-big-data.html

[16] https://www.cwi.it/big-data

[17] https://blog.osservatori.net/it_it/le-5v-dei-big-data#:~:text=Nel

[18] https://www.redhat.com/it/devops/what-is-agile-methodology

[19] https://it.wikipedia.org/wiki/Netezza

Parte VI

Ringraziamenti

Ringraziamenti

Infine, siamo giunti alla sezione dedita ai ringraziamenti, momento importante ed emozionante in cui voglio esprimere parole di gratitudine verso persone che mi sono state accanto in questi anni di università.

Prima di tutto vorrei ringraziare e dedicare questo traguardo alla mia famiglia, mio padre Saverio, mia madre Assunta e mia sorella Alessia, per ogni singolo sacrificio e per aver creduto nelle mie capacità, soprattutto nei momenti più difficili.

Poi vorrei ringraziare i miei amici più stretti, che nonostante questo periodo che stiamo vivendo si sono dimostrati dei veri amici, supportandomi e sopportandomi nei momenti di crisi.

Successivamente, vorrei ringraziare anche i miei colleghi di università, in particolare le colleghe Hajar, Alessia e Geralda, per tutte quelle volte che ci siamo incoraggiati a vicenda, anche nei momenti di crisi, tutti quei momenti di studio e di divertimento insieme.

Inoltre, ringrazio l'azienda Iconsulting perché mi ha dato la possibilità di svolgere il tirocinio e l'opportunità di lavorare per loro, portando a termine la stesura della mia tesi.

Infine, un ringraziamento va al professore Marco Di Felice e al correlatore Raffaele Grezzi perché mi hanno saputo seguire in questo mio progetto e si sono dimostrati persone competenti nel loro ruolo professionale.