ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Scuola di Ingegneria e Architettura
Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

ANALISI E SVILUPPO DEL PROCESSO DI MIGRAZIONE DEL DATA WAREHOUSE DI UN'AZIENDA AUTOMOBILISTICA

Elaborato in
BUSINESS INTELLIGENCE

Relatore
Prof. STEFANO RIZZI

Presentata da
FEDERICO ORAZI

Terza Sessione di Laurea
Anno Accademico 2020 – 2021

PAROLE CHIAVE

Business Intelligence

Data Warehouse

Data Platform

Microsoft Azure

Migrazione dati

A Voi che sostenete i miei
sogni e le mie ambizioni.

Indice

Introduzione **ix**

Sommario **xi**

1 Business Intelligence **1**
1.1 Piramide della BI 1
1.2 Architettura di un sistema di BI 3
1.3 Sorgenti operazionali 5
1.4 Processo di ETL 6
1.4.1 Estrazione 7
1.4.2 Pulitura 7
1.4.3 Trasformazione 7
1.4.4 Caricamento 8
1.5 Data Warehouse 8
1.5.1 Modello multidimensionale 9
1.5.2 Modelli logici 10
1.6 BI Analytics 12
1.7 BI nel cloud 12

2 Stack tecnologico **15**
2.1 Data Platform 16
2.1.1 Microsoft Azure 17
2.2 Ingestion 19
2.2.1 Azure Data Factory 19
2.3 Storage 20
2.3.1 Azure Data Lake Storage 21
2.4 Model & Serve 22
2.4.1 SAP BusinessObject 22
2.4.2 Azure Synapse Analytics 25
2.4.3 Azure Analysis Services 27
2.4.4 Power BI 30
2.5 Azure DevOps 33

3 Migrazione del Data Warehouse **35**
3.1 Obiettivo 35
3.2 Analisi AS-IS 36
3.2.1 Architettura AS-IS 36
3.2.2 Analisi preliminare 38
3.3 Analisi TO-BE 42
3.3.1 Architettura TO-BE 43
3.3.2 Ottimizzazioni architetturali 45
3.3.3 Standard di sviluppo 47
3.4 Migrazione delle Data Entity 49
3.4.1 Analisi 50
3.4.2 Mockup 51
3.4.3 Sviluppo 57
3.4.4 Test 66

4 Conclusioni e sviluppi futuri **69**

Ringraziamenti **71**

Riferimenti **73**

Introduzione

L'informazione è un bene prezioso per ogni azienda, necessario per prendere decisioni e controllare le attività aziendali. Essa è frutto di trasformazioni che i sistemi informativi eseguono sui dati. Prima della nascita dei Data Warehouse, le informazioni venivano estrapolate attraverso procedimenti di selezione e sintesi progressiva eseguite direttamente sulla grande mole di dati contenuti nei database aziendali. Il problema di questo processo era principalmente legato alla lentezza e alla poca efficienza. Con l'introduzione dei Data Warehouse è stato possibile raccogliere dati provenienti da sorgenti di varia natura rendendoli direttamente disponibili ai manager che hanno potuto utilizzarli per analisi e valutazioni finalizzate alla pianificazione e al processo decisionale.

Col tempo il processo di Data Warehousing ha subito notevoli trasformazioni dovute principalmente alla necessità di adattarsi alle nuove tecnologie emergenti. La nascita dei Big Data e del Cloud ha innescato il passaggio verso soluzioni che integrano tecnologie di varia natura e le mettono a disposizione per soddisfare le nuove esigenze aziendali. Nascono in questo modo le Data Platform che aprono la strada ad analisi che non si limitano a descrivere *"Cosa è successo"* ma si spingono oltre, ipotizzando *"Cosa sta succedendo, cosa accadrà e perché"*.

L'azienda IConsulting lavora da anni su queste tematiche e ha col tempo collezionato un insieme molto numeroso di clienti per i quali realizza progetti di Business Intelligence di varia natura. In questo contesto si colloca il progetto di tesi. Un'azienda molto nota nel settore automobilistico, che per motivi di privacy da questo momento in poi chiameremo Cliente, necessita di valutare e adottare la miglior strategia evolutiva per il sistema di Business Intelligence attualmente in uso, al fine di supportare i propri obiettivi strategici di business.

È richiesta l'analisi della situazione attuale (AS-IS) del sistema di Business Intelligence del Cliente insieme alla valutazione e l'adozione del miglior processo di evoluzione che produca una nuova architettura (TO-BE) che soddisfi le nuove esigenze. Viene quindi progettato e sviluppato un processo di migrazione dal vecchio sistema a quello nuovo che non si limita solamente a replicare quanto già esistente ma anche a migliorarlo, estendendone le capacità analitiche complessive.

Sommario

Seguendo un percorso suddivisibile essenzialmente in due fasi, la tesi si sviluppa attraverso tre capitoli. La prima fase, dedicata all'approfondimento di tematiche teoriche, è descritta attraverso i primi due capitoli. Il terzo capitolo, invece, è dedicato alla descrizione delle attività di analisi e agli sviluppi.

Capitolo 1 Nel primo capitolo vengono introdotti i concetti di Business Intelligence e Data Warehousing delineando il percorso evolutivo e le motivazioni che al giorno d'oggi hanno portato dall'utilizzo di soluzioni prevalentemente on-premise ad un'integrazione con il mondo del Cloud Computing.

Capitolo 2 Il secondo capitolo si collega a quanto descritto nel precedente e ha lo scopo di introdurre il concetto di Data Platform come naturale evoluzione dei sistemi di Business Intelligence tradizionali. Inoltre, vengono presentate le diverse tecnologie coinvolte in tutto il processo di realizzazione del progetto per le quali ci si sofferma a descrivere gli aspetti fondamentali.

Capitolo 3 Il terzo capitolo entra nel merito del progetto realizzato. Viene descritto il processo di analisi che, partendo dalla situazione attuale (AS-IS), ha portato alla definizione dell'architettura del nuovo sistema di Business Intelligence (TO-BE) insieme agli standard e alle linee guida da seguire a progetto avviato. Successivamente, l'attenzione si focalizza sull'attività di sviluppo e quindi sulla migrazione di due delle Data Entity del sistema originario.

Capitolo 1

Business Intelligence

Ogni azienda ha bisogno di prendere decisioni per raggiungere il proprio obiettivo di business. Tali decisioni devono essere guidate dai dati per poter ottenere risultati positivi. Poiché nel corso degli anni la mole di dati a disposizione è aumentata a dismisura, l'utilizzo dell'informatica per gestirli ha assunto un ruolo sempre più di spicco nel processo decisionale aziendale. Se in passato il suo principale utilizzo era quello di supportare le attività dell'azienda per renderle più veloci ed economiche, oggi ha un impatto cruciale sul business poiché consente ai manager di prendere decisioni tattico/strategiche e di individuare elementi critici nell'organizzazione.

Con il concetto di Business Intelligence si intende l'insieme di strumenti e procedure che consentono a un'azienda di trasformare i propri dati di business in informazioni e conoscenza utili al processo decisionale. Le informazioni così ottenute sono utilizzate dai decisori aziendali per definire e supportare le strategie di business, così da operare decisioni consapevoli e informate con l'obiettivo di trarre vantaggi competitivi, migliorare le prestazioni operative e la profittabilità e, più in generale, creare valore per l'azienda.

1.1 Piramide della BI

Dati, informazione e conoscenza non sono la stessa cosa. Quando si parla di **dati** si intendono i dati prodotti dalle sorgenti transazionali/operazionali. Sono caratterizzati dal fatto di avere numerosità elevata e di essere tipicamente di bassa qualità. I dati vengono trasformati in **informazione** a seguito di un processo di raffinamento che ha come obiettivo quello di rimuovere inconsistenze, errori e ridondanze. L'informazione viene sottoposta ad un ulteriore processo di raffinamento per ottenere **conoscenza**. La conoscenza rappresenta l'obiettivo del processo di analisi: si tratta di qualcosa di sintetico e ad alto valore che viene utilizzato per guidare il processo decisionale.

La sequenza di operazioni che consente di trasformare i dati grezzi in informazione e conoscenza prende il nome di **Data Pipeline** e viene sintetizzata all'interno della **Piramide della BI** mostrata in Figura 1.1. La piramide è suddivisa in livelli, a partire dal basso troviamo:

- *Livello 1* - è il livello delle sorgenti dati. Le sorgenti possono essere di varia natura, dai database gestionali ai dati opensource recuperabili tramite Internet;
- *Livello 2* - è il livello dell'**Operational Data Store (ODS)** ovvero di quella componente dell'architettura di un Data Warehouse nella quale vengono caricati i dati delle sorgenti dopo essere stati ripuliti da eventuali errori, normalizzati e integrati. L'ODS rappresenta il luogo ideale per eseguire query di reportistica operativa utili alle attività quotidiane dell'azienda;
- *Livello 3* - è il livello del Data Warehouse che contiene informazioni che possono essere elaborate attraverso interrogazioni di tipo **OLAP (On-Line Analytical Processing)**. Questo tipo di interrogazioni analitiche sono molto utili ai manager poiché coinvolgono tipicamente grosse quantità di dati per produrre un risultato sintetico utile al processo decisionale. Sui dati contenuti nel DW è anche possibile realizzare **cruscotti** ovvero finestre di indicatori che mostrano l'andamento del business;
- *Livello 4* - è il livello in cui viene applicato il Data Mining sulle informazioni estratte dal DW. L'obiettivo è quello di individuare pattern ricorrenti e sfruttarli per prendere decisioni per il futuro;
- *Livello 5* - è il livello delle analisi What-If ovvero di quelle analisi rivolte al futuro che cercano di prevedere l'impatto che si otterrà sul business a seguito di una certa decisione.

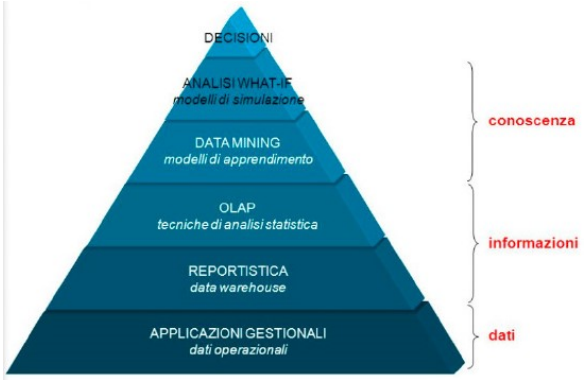


Figura 1.1: Piramide dalla BI

1.2 Architettura di un sistema di BI

Da un punto di vista strutturale, esistono tre possibili architetture che possono essere utilizzate per realizzare un sistema di BI:

- **Architetture a 1 livello** - presentano un unico livello fisico, quello delle sorgenti. Poiché il DW non possiede un vero e proprio livello fisico, le query OLAP eseguite dai manager devono necessariamente essere elaborate da un middleware che sia in grado di tradurle in tempo reale in un insieme di query da eseguire sui DB operazionali.

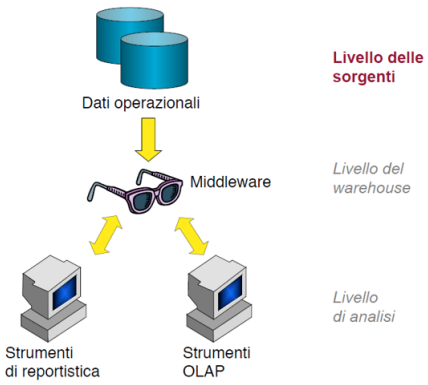


Figura 1.2: Architettura a 1 livello

Questa architettura è superata da anni poiché non rispetta il principio di separazione dei database sorgente dal DW, causando rallentamenti nell'esecuzione delle query operative sui database sorgente quando vengono eseguite delle query OLAP.

- **Architetture a 2 livelli** - presentano due livelli fisici, quello delle sorgenti e quello del DW. In questo caso, le sorgenti possono essere di varia natura e, attraverso un processo di ETL, i dati in esse contenuti vengono estratti, sottoposti a trasformazioni e caricati nel DW. Quest'ultimo è tipicamente costituito da un insieme di più Data Mart. Un Data Mart rappresenta una frazione del DW rilevante per una certa area di business, per un certo reparto aziendale, per un certo gruppo di utenti.

  I vantaggi principali offerti da questo tipo di architettura sono moltiplici:

  - l'interrogazione analitica OLAP effettuata sul DW non interferisce con il livello operazionale;
  - il livello del warehouse contiene sempre informazioni di buona qualità disponibili per essere interrogate;

- l'organizzazione logica del DW può seguire il modello multidimensionale che è più facilmente comprensibile ai non esperti del settore.

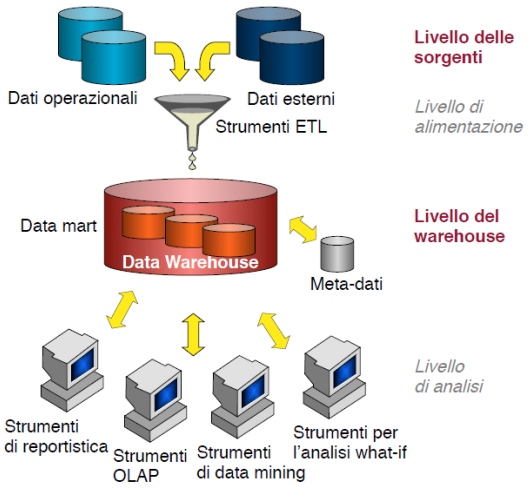


Figura 1.3: Architettura a 2 livelli

- **Architetture a 3 livelli** - in questo caso si rende fisico anche il livello dell'alimentazione. Per fare ciò, viene introdotto l'Operational Data Store (ODS) dentro il quale vengono caricati i dati operazionali dopo essere stati corretti, normalizzati e integrati.

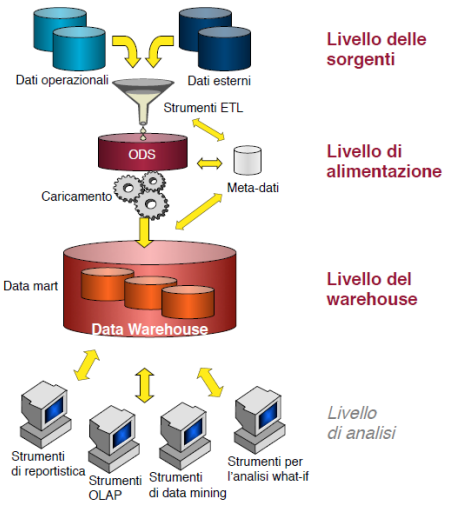


Figura 1.4: Architettura a 3 livelli

Lo svantaggio principale dell'utilizzare questo tipo di architettura è che richiede una quantità di memoria maggiore rispetto alle architetture precedenti. I vantaggi, invece, sono principalmente dati dalla presenza dell'ODS che semplifica il processo di ETL consentendo di spezzarlo in due fasi distinte e gestibili separatamente, inoltre, rappresenta il luogo ideale per l'esecuzione delle query di reportistica operativa.

Di seguito vengono analizzate nel dettaglio le componenti principali che è possibile individuare in una tipica architettura di un sistema di BI.

1.3 Sorgenti operazionali

Le sorgenti operazionali rappresentano il punto di partenza dell'intera architettura. In un'azienda le sorgenti operazionali possono essere diverse e ognuna di queste può memorizzare dati appartenenti a contesti differenti. Le operazioni che vengono eseguite sulle sorgenti sono dette transazioni. Le sorgenti prendono anche il nome di **sistemi OLTP** poiché il carico di lavoro che le coinvolgono è principalmente di tipo *OLTP (On-Line Transaction Processing)*. Le query OLTP sono elaborazioni interattive finalizzate alle transazioni sia in lettura sia in scrittura e tipicamente coinvolgono una quantità ridotta di dati. Il carico di lavoro OLTP solitamente non varia nel tempo ed è incapsulato all'interno delle logiche applicative.

I sistemi OLTP sono adatti a supportare i processi di business ma non per la valutazione e l'analisi degli stessi. Il motivo è che le sorgenti sono organizzate in modo separato l'una dall'altra, ognuna può essere realizzata con una tecnologia diversa e contenere dati appartenenti a domini differenti. Per l'analisi dei processi di business è necessario ricorrere ai **sistemi analitici OLAP**. Questi ultimi sono ottimizzati per effettuare analisi sui dati e forniscono strumenti per eseguire il reporting degli stessi. In Figura 1.5 vengono riportate le principali differenze rispetto ai sistemi OLTP.

	OLTP	OLAP
Function	Day to day operation	Decision support
Database Design	Application oriented	Subject oriented
Data	Current, up-to-date detailed, flat relational, isolated	Historical, summarized multi-dimensional, consolidated
Usage	Repetitive	Ad-hoc
Access	Read/Write	Lots of scans
Unit of Work	Short, simple transaction	Complex query
Database Size	Gigabytes	Terabytes
Metric	Transaction throughput	Query throughput, response

Figura 1.5: Sistemi OLTP vs Sistemi OLAP

Prima di poter utilizzare i sistemi analitici sono necessari ulteriori passi intermedi, infatti, risulta necessario trasformare i dati contenuti nelle sorgenti in modo che vengano corretti da eventuali errori ed integrati tra loro. Il processo che si occupa di questo passaggio prende il nome di *ETL (Extract, Transform, Load)*.

1.4 Processo di ETL

La procedura di ETL è cruciale in un sistema di BI poiché si fa garante della qualità dei dati di cui si disporrà all'interno del DW. Il ruolo dell'ETL è quello di alimentare una sorgente dati singola, dettagliata, esauriente e di alta qualità (l'ODS) che possa a sua volta alimentare il DW. Nelle architetture a 2 livelli alimenta direttamente il DW senza passare per l'ODS. Sono quattro le operazioni che il processo di ETL svolge:

- Estrazione;
- Pulitura;
- Trasformazione;
- Caricamento.

Durante l'esecuzione di questo processo, viene impiegata un'area intermedia, chiamata **Staging Area**, che l'ETL utilizza come database di servizio per memorizzare temporaneamente i dati elaborati a seguito delle trasformazioni applicate e che successivamente verranno trasferiti sull'ODS o sul DW, a seconda del tipo di architettura utilizzata.

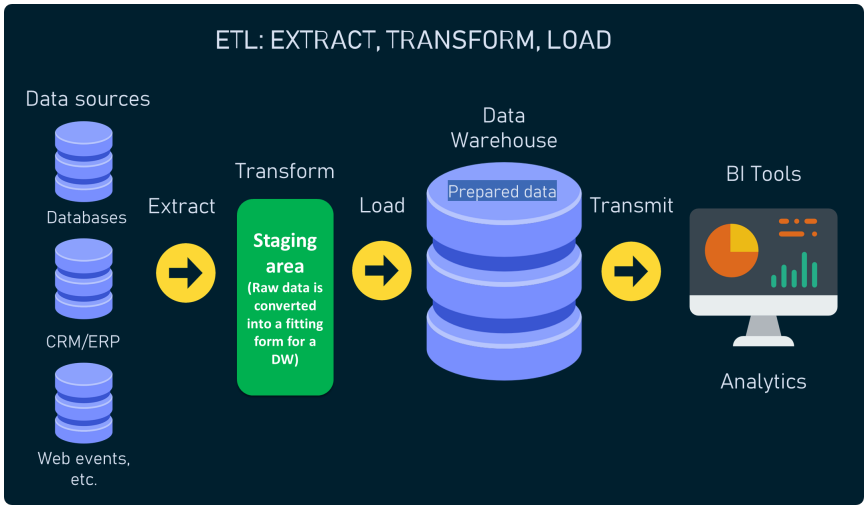


Figura 1.6: ETL con dettaglio sulla Staging Area

1.4.1 Estrazione

In questa fase viene eseguita l'estrazione dei dati dai DB sorgente. I dati vengono temporaneamente memorizzati all'interno della Staging Area dove vengono sottoposti ad operazioni di pulizia e trasformazione prima di essere caricati all'interno dell'ODS. L'estrazione può essere di due tipi:

- **Statica** - vengono estratti tutti i dati. Rappresenta una fotografia delle informazioni contenute nelle sorgenti operazionali e viene solitamente eseguita per il primo popolamento del DW;
- **Incrementale** - vengono estratti solamente i record che hanno subito modifiche dalla data di ultima estrazione. In questo caso, è necessario mantenere informazioni sulla precedente estrazione, in modo tale da poter determinare quali record sono stati modificati. L'esecuzione può essere effettuata in modo immediato o ritardato.

1.4.2 Pulitura

L'obiettivo della fase di pulitura è quello di migliorare la qualità dei dati e quindi di correggere gli errori eventualmente presenti. Tipicamente, il processo di pulitura prevede la gestione di:

- dati duplicati;
- inconsistenze tra valori logicamente associati;
- dati mancanti;
- uso non previsto di campi;
- valori impossibili o errati;
- errori di battitura.

1.4.3 Trasformazione

La trasformazione ha come obiettivo quello di rendere uniforme e standard il formato dei dati. Le diverse sorgenti, infatti, memorizzano informazioni in formato differente e di conseguenza, per poter effettuare l'integrazione, risulta necessario trasformare i dati in un formato uniforme. Se si adotta un'architettura a 3 livelli, la trasformazione viene suddivisa in due fasi:

- Fase 1, dalle sorgenti all'ODS - viene applicata la conversione di formato ai dati sorgente per renderli uniformi, dopodiché i dati vengono integrati ed eventualmente selezionati nel caso in cui nelle sorgenti siano presenti dati non utili;
- Fase 2, dall'ODS al DW - i dati vengono eventualmente denormalizzati e aggregati ad un livello di dettaglio differente rispetto a quello delle sorgenti. Vengono inoltre derivate nuove misure a partire da quelle già esistenti.

1.4.4 Caricamento

Rappresenta l'ultima fase del processo di ETL. I dati vengono caricati dal livello riconciliato al DW finale. Le modalità di caricamento sono principalmente due:

- **Refresh** - il DW viene ricaricato completamente. I vecchi dati vengono cancellati per lasciare spazio ai nuovi. Questa modalità viene solitamente utilizzata per effettuare il primo caricamento del DW;
- **Update** - vengono caricati nel DW solamente quei dati che hanno subito una modifica dalla data dell'ultimo caricamento. I dati che sono stati modificati non vengono cancellati o aggiornati, garantendo così la storicizzazione del DW.

1.5 Data Warehouse

Il Data Warehouse ha il compito di raccogliere dati provenienti da sorgenti di varia natura e di renderli disponibili ai manager che possono utilizzarli per eseguire analisi e valutazioni finalizzate alla pianificazione e al processo decisionale.

Come anticipato in precedenza, i sistemi operazionali vengono costruiti rispettando i criteri della normalizzazione che garantiscono una maggior efficienza riducendo la ridondanza dei dati. Un database progettato in 3NF potrebbe avere tabelle con un numero elevato di relazioni. Di conseguenza, la realizzazione di un report analitico a partire da un sistema di questo tipo, potrebbe rallentare l'esecuzione della query dato l'elevato numero di join necessari per recuperare le informazioni. La struttura del DW è appositamente pensata per evitare questo tipo di inconvenienti, riducendo il tempo di risposta ed incrementando le performance delle query per il reporting e l'analisi dei dati. Il modello con il quale viene costruito un DW prende il nome di modello multidimensionale.

1.5.1 Modello multidimensionale

Il modello Entity-Relationship (ER) tradizionalmente utilizzato per la progettazione concettuale di database, non può essere adottato come fondamento per la realizzazione di un DW. Tale modello descrive la struttura del dominio applicativo, ma non esprime concetti come la multidimensionalità o la gerarchia dei livelli di aggregazione. Il modello multidimensionale è l'astrazione attraverso la quale è possibile rappresentare ed interrogare i dati nel DW. Esso consente una fruizione più agevole da parte degli utenti e una maggiore flessibilità di interrogazione. Il dato, infatti, deve aderire ad un modello più intuitivo per i non esperti rispetto al modello relazionale e risulta particolarmente adatto per essere interrogato attraverso query OLAP. Gli elementi di base che caratterizzano il modello multidimensionale sono i seguenti:

- **Fatto** - rappresenta un fenomeno di business che accade dinamicamente nell'azienda e che i manager sono interessati a monitorare;
- **Cubo** - la metafora attraverso la quale immaginiamo di rappresentare un fatto;
- **Misura** - rappresenta una proprietà numerica di un fatto che descrive un aspetto quantitativo di interesse per l'analisi;
- **Dimensione** - può essere pensata come un asse di un sistema cartesiano tramite il quale l'utente può accedere ai fatti. Un cubo può disporre di più dimensioni ognuna delle quali descrive una coordinata d'analisi del fatto;
- **Attributo dimensionale** - rappresenta una proprietà, con dominio finito, di una dimensione;
- **Gerarchia** - ogni dimensione può essere la radice di una gerarchia di attributi usati per aggregare i dati memorizzati nel cubo.

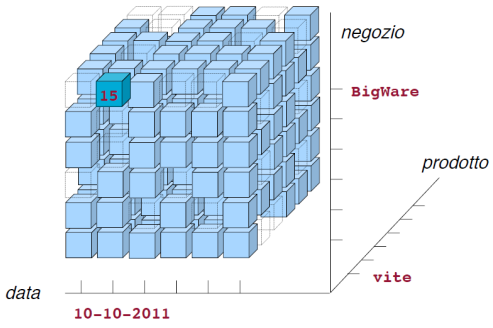


Figura 1.7: Esempio di cubo

Un possibile modello concettuale grafico per rappresentare schemi multidimensionali è il *Dimensional Fact Model (DFM)*. Il DFM possiede le seguenti caratteristiche:

- fornisce pieno supporto alla progettazione logica;
- rende disponibile un ambiente nel quale eseguire valutazioni sul carico di lavoro in modo intuitivo;
- facilita la comunicazione tra progettista e utente finale;
- costituisce un documento stabile da cui partire per il progetto logico;
- può essere utilizzato a posteriori come documentazione espressiva e non ambigua.

Gli elementi di base del modello multidimensionale vengono rappresentati nel DFM attraverso schemi di fatto.

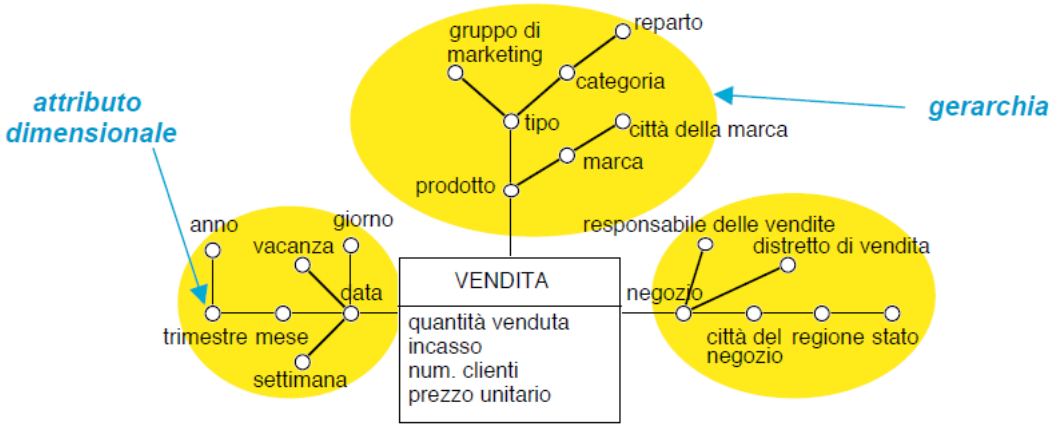


Figura 1.8: Esempio di schema di fatto che modella le vendite di prodotti

1.5.2 Modelli logici

I dati possono essere memorizzati su DW secondo due principali modelli logici:

- **ROLAP (Relational OLAP)** - in questo caso, per rappresentare informazioni multidimensionali, vengono utilizzati database relazionali. Poiché il modello relazionale utilizzato per memorizzare i dati non corrisponde al modello multidimensionale richiesto per poter esprimere interrogazioni OLAP, è necessario ricorrere ad un motore multidimensionale che si

occupi di tradurre le query OLAP in query SQL. Le query OLAP tradotte potrebbero produrre problemi di prestazioni dovute al gran numero di join eventualmente richiesti. Per risolvere il problema, tipicamente le tabelle del DW vengono denormalizzate, riducendo in questo modo il numero di join coinvolti nelle query;

- **MOLAP (Multimensional OLAP)** - i dati multidimensionali vengono memorizzati su una piattaforma che nativamente li memorizza secondo il modello multidimensionale. In questo caso, non è richiesto l'utilizzo di un intermediario che si occupi della traduzione delle query poiché sia il front-end sia il back-end adottano lo stesso modello di rappresentazione dei dati. Il problema principale di cui soffre questo modello è quello della *sparsità*, infatti, una percentuale anche molto elevata di celle dei cubi che compongo il DW può essere vuota e, nonostante ciò, occupano ugualmente spazio in memoria.

Per ovviare alle problematiche dei due modelli descritti in precedenza, ad oggi sono anche disponibili soluzioni ibride che cercano di raccogliere i vantaggi di entrambi i modelli mitigandone gli svantaggi. In questo caso si parla di **HOLAP (Hybrid OLAP)**.

Per rappresentare dati multidimensionali in database relazionali esistono due possibili schemi:

- **Star Schema** - la tabella dei fatti si trova al centro dello schema e le dimensioni sono collegate ad essa tramite relazioni di un solo livello. Tipicamente, le dimensioni vengono denormalizzate rispetto allo schema di partenza per ridurre il numero di join richiesti nelle query analitiche.

- **Snowflake Schema** - la tabella dei fatti si trova al centro dello schema e le dimensioni vengono normalizzate in tutto o in parte.

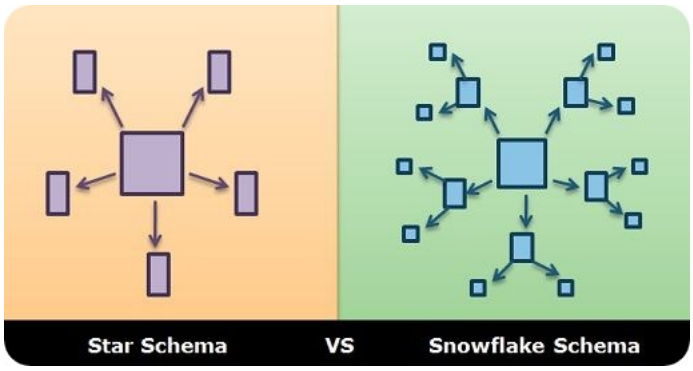


Figura 1.9: Star schema vs Snowflake schema

1.6 BI Analytics

Rappresenta l'ultimo livello dell'architettura di un sistema di BI e consiste nell'insieme di tecnologie e strumenti che consentono all'utente di esprimere interrogazioni sui dati contenuti all'interno del DW. A questo livello i dati possono essere rappresentati anche con indicatori di sintesi (KPI) che consentono ai manager di avere una visione più aggregata di un determinato fenomeno semplificando e velocizzando il processo decisionale. Sul mercato sono disponibili diversi tools per la realizzazione di report e dashboard che adottano la modalità di visualizzazione dei dati più opportuna e gradevole in base alle richieste del cliente.

1.7 BI nel cloud

Le soluzioni di BI per l'analisi delle informazioni e il mapping dei dati seguono tradizionalmente un approccio on-premise che prevede che tutte le componenti dello stack tecnologico utilizzato vengano acquistate e mantenute direttamente dall'azienda che ne fa uso. La nascita dei Big Data e l'evoluzione tecnologica hanno portato ad un cambiamento nelle esigenze di analisi aziendali che hanno reso le soluzioni di BI tradizionali non più sufficienti per il loro scopo. Emerge dunque la necessità di soluzioni più veloci, scalabili, sicure e disponibili on-demand che permettano di colmare il gap con il passato e portare la BI ad un livello successivo. L'utilizzo del cloud computing ha reso possibile questo passaggio di livello.

Il cloud computing consiste nella distribuzione on-demand di risorse IT tramite Internet, con una tariffazione basata sul consumo. Piuttosto che acquistare, possedere e mantenere i data center e i server fisici, è possibile accedere a servizi tecnologici, quali capacità di calcolo, storage e database, sulla base delle proprie necessità affidandosi a un fornitore cloud. La cloud BI offre tutta una serie di vantaggi, tra cui:

- **Scalabilità** - le risorse cloud sono spesso espandibili per soddisfare le esigenze aziendali. I piani tariffari variano di conseguenza in base al numero di utenti e alle funzioni richieste;

- **Elasticità** - grazie al cloud computing non è necessario allocare in anticipo una quantità di risorse maggiore di quante ne siano effettivamente necessarie così da gestire i picchi nei livelli di attività aziendali in futuro. La quantità di risorse viene automaticamente scalata in risposta al carico di lavoro;

- **Semplicità di deployment** - la semplicità è data dal fatto che non è richiesta l'installazione di hardware e software aggiuntivi;
- **Mobile-friendly** - le soluzioni di cloud BI sono accessibili anche tramite dispositivi mobili quali smartphone e tablet;
- **Riduzione degli overhead** - le soluzioni cloud riducono al minimo il numero di server e personale necessari per gestire le operazioni quotidiane. Il tempo risparmiato nelle attività che una volta spettavano al reparto IT, può essere speso per altre funzioni all'interno dell'azienda;
- **Risparmio sui costi** - il cloud permette di evitare spese di capitale (per esempio, per data center e server fisici) in favore di una spesa variabile, pagando solo per le risorse IT realmente consumate.

Capitolo 2

Stack tecnologico

Come anticipato nel capitolo precedente, in un mondo in cui la quantità di dati prodotti cresce in modo esponenziale, le aziende devono essere in grado di ricavare sempre più valore da tali dati e dunque le esigenze tendono ad essere sempre maggiori. Le piattaforme di BI tradizionali sono principalmente in grado di fornire agli utenti report storici completi e strumenti di analisi ad hoc di facile utilizzo. I data warehouse costituiscono l'elemento fondamentale nelle piattaforme di BI tradizionali e spesso svolgono un ruolo di collegamento tra le varie tecnologie coinvolte nel sistema. I dati vengono standardizzati, ripuliti, trasformati e caricati nel DW prima di essere inseriti nei vari report e dashboard.

Se in passato poteva essere sufficiente interrogare dati storici e ricavare valore da essi, ad oggi sono richieste anche funzionalità aggiuntive tra cui:

- **analisi on-demand** - gli utenti richiedono funzionalità self-service per mettere in relazione e analizzare set di dati specifici in base alla propria comprensione, in qualsiasi momento, per qualsiasi scopo;
- **analisi predittive** - le funzionalità di reporting storico forniscono solo un pezzo del puzzle: informazioni su ciò che è accaduto in passato. Per diventare veramente data-driven, le aziende hanno bisogno di analisi predittive e di informazioni sul futuro. Con i modelli predittivi, le aziende possono utilizzare modelli e previsioni per prendere decisioni in base ai propri dati;
- **analisi di dati di diverso tipo** - mentre le piattaforme di BI tradizionali sono principalmente focalizzate sui dati strutturati, gli utenti di oggi hanno bisogno di analizzare anche dati semi-strutturati e non strutturati.

In questo capitolo, vengono descritti gli aspetti fondamentali delle diverse tecnologie coinvolte in tutto il processo di realizzazione del progetto.

2.1 Data Platform

Allo scopo di colmare il gap tra la BI tradizionale e quella moderna, sono nate delle piattaforme che raggruppano tecnologie di varia natura e che le mettono a disposizione in maniera integrata per gestire e analizzare i dati in modo da soddisfare le nuove esigenze aziendali. Si parla in questo caso di Data Plaftorm. Una Data Platform è un insieme integrato di tecnologie che soddisfa collettivamente le esigenze di dati end-to-end di un'organizzazione. Consente l'acquisizione, lo storage, la preparazione, la consegna e la governance dei dati, garantendo inoltre un livello di sicurezza per utenti e applicazioni. Una Data Platform mette a disposizione gli strumenti necessari per rispondere non solo alla domanda *"Cosa è successo"* ma anche alle domande *"Cosa sta succedendo, cosa accadrà e perché"* aprendo la strada ad analisi predittive utili al raggiungimento degli obiettivi aziendali.

Esistono diversi cloud provider che rendono disponibili intere Data Platform tramite il cloud. Utilizzare una Cloud Data Platfom, permette di gestire il dato astraendo tutte le logiche tecniche sulle quali si basa la piattaforma. La Cloud Data Platform fornisce direttamente accesso ai servizi senza doversi preoccupare di come questi sono effettivamente implementati.

Ogni Cloud Data Platform sul mercato dispone dei propri strumenti/servizi per la gestione e l'analisi dei dati ma, in generale, condividono tutte una stessa organizzazione. La moderna architettura di una Data Platform può essere riassunta come mostato in Figura 2.1. Vengono distinti quattro passi fondamentali:

- **Ingestion** - il primo passo è quello di acquisire i dati dalle sorgenti;
- **Storage** - dopo che i dati sono stati acquisiti, devono essere archiviati in un formato durevole e facilmente accessibile;
- **Preparation & Train** - i dati vengono esplorati ed eventualmente trasformati per essere utilizzati per addestrare modelli predittivi;
- **Model & Serve** - la fase finale consiste nel realizzare un modello che sia facilmente interrogabile attraverso strumenti di reportistica.

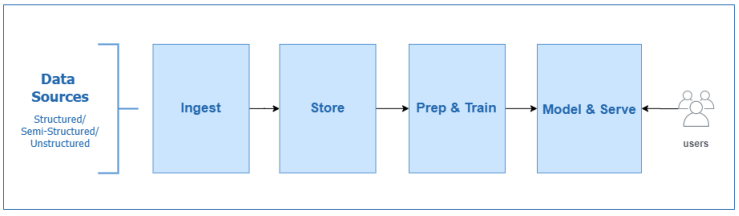


Figura 2.1: Flusso dei dati in una Data Platform

Un aspetto molto importante da notare è che la componente di calcolo e quella di archiviazione sono mantenute separate e indipendenti. Questo significa che vengono applicate politiche di addebito differenti: per l'archiviazione è necessario pagare solo per la quantità di storage effettivamente utilizzata mentre la potenza di calcolo viene pagata in termini di risorse (CPU, memoria, IO) richieste.

Tra i vari cloud provider esistenti, quello utilizzato per la realizzazione di questo progetto è Microsoft Azure. Di seguito ne vengono descritte le principali caratteristiche.

2.1.1 Microsoft Azure

Azure è una piattaforma di cloud computing che mette a disposizione un insieme di servizi in continua espansione per creare soluzioni per raggiungere gli obiettivi aziendali. I servizi di Azure vanno da semplici servizi di hosting nel cloud all'esecuzione di computer completamente virtualizzati per la realizzazione di soluzioni software personalizzate. Azure offre una vasta gamma di servizi basati su cloud come l'archiviazione remota, l'hosting di database e la gestione centralizzata degli account. Inoltre, mette a disposizione funzionalità per AI e Internet of Things (IoT).

Il funzionamento di Azure si basa sulla *virtualizzazione*. La virtualizzazione consente di separare l'hardware di un computer dal suo sistema operativo utilizzando un livello di astrazione chiamato *Hypervisor*.

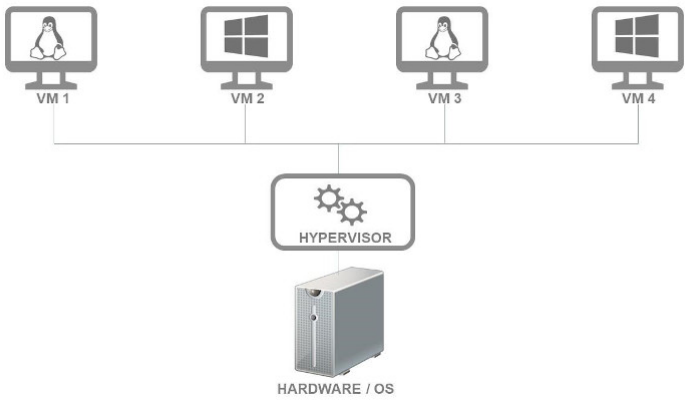


Figura 2.2: Hypervisor & Virtual Machines

L'Hypervisor emula tutte le funzioni di un reale computer e della sua CPU all'interno di una macchina virtuale, ottimizzando le capacità dell'hardware. Consente l'esecuzione di diverse macchine virtuali contemporaneamente e ogni macchina virtuale è in grado di eseguire un sistema operativo differente.

Azure sfrutta le tecniche di virtualizzazione e le ripete su scala globale all'interno dei Data Center Microsoft sparsi in tutto il mondo. Ogni Data Center possiede diversi Rack che al loro interno includono diversi server. Ogni server include un Hypervisor in grado di eseguire diverse macchine virtuali. I server sono collegati tra loro attraverso uno switch di rete. All'interno di uno dei server di ogni rack viene eseguito uno speciale software chiamato *Fabric Controller (FC)*. Il FC ha il compito di monitorare e gestire i server del rack e di coordinare le risorse per le applicazioni software. Ogni FC è collegato ad un altro software chiamato *Orchestrator*. L'Orchestrator è il responsabile della gestione di tutto ciò che avviene in Azure, incluso fornire risposte alle richieste dell'utente. L'utente effettua richieste utilizzando la web API dell'Orchestrator. La web API può essere acceduta attraverso diversi strumenti tra cui la user interface del portale Azure. Quando un utente esegue una richiesta per utilizzare una VM, l'Orchestrator impacchetta tutto quello che serve e lo invia ad uno dei server rack dove viene ricevuto dal Fabric Controller. Il Fabric controller crea la macchina virtuale, dopodiché l'utente può collegarsi ad essa.

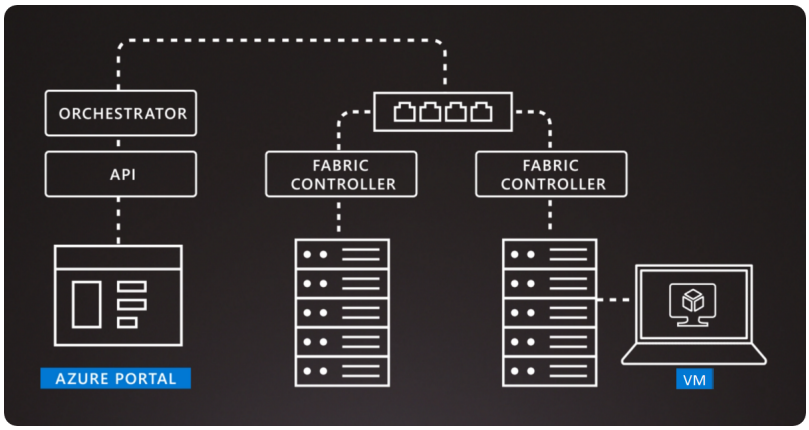


Figura 2.3: Orchestrator & Hypervisor

Oltre alla creazione di macchine virtuali, Azure consente di accedere ad un numero sempre crescente di servizi Cloud. Questi ultimi includono:

- potenza di calcolo on-demand;
- memorizzazione di dati in maniera scalabile e sicura;
- gestione e analisi di Big Data;
- servizi per IoT e AI.

Tutto questo ovviamente offerto secondo la formula *"pay as you go"*, caratteristica ormai molto comune nel mondo del cloud computing.

2.2 Ingestion

L'Ingestion rappresenta il primo passo in qualsiasi Data Platform Architecture e consiste nell'acquisizione dei dati dalle sorgenti. Queste ultime possono avere nature differenti e contenere dati di diverso tipo. I dati potrebbero provenire da sorgenti on-premise oppure cloud, potrebbero essere dati stazionari (batch) o in movimento (stream), strutturati o non strutturati. In generale, le complessità che devono essere affrontate in questa fase sono innumerevoli.

In Microsoft Azure uno dei possibili strumenti per gestire l'acquisizione dei dati è Azure Data Factory. Di seguito ne vengono descritte le principali caratteristiche.

2.2.1 Azure Data Factory

Azure Data Factory (ADF) è un servizio cloud-based di ETL e di integrazione che consente di creare flussi di lavoro basati sui dati per orchestrare lo spostamento e la trasformazione di questi ultimi su larga scala. Si tratta dello strumento che Microsoft Azure mette a disposizione per la Data Integration e la Data Orchestration.

Azure Data Factory gestisce lo spostamento e la trasformazione dei dati tra vari archivi e risorse di calcolo. È possibile creare e pianificare flussi di lavoro (detti Pipeline) che consentono di eseguire l'acquisizione dei dati da svariate tipologie di sorgenti. L'obiettivo di Data Factory è quello di recuperare dati da una o più sorgenti e trasformarli in un formato che possa essere più facilmente processato.

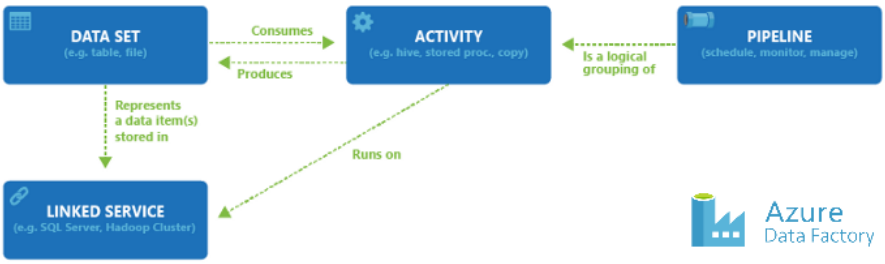


Figura 2.4: Componenti principali di ADF

La Figura 2.4 mostra le principali componenti di Azure Data Factory. Di seguito vengono descritte più nel dettaglio.

- **Pipeline** - una Pipeline rappresenta un raggruppamento logico di attività ognuna delle quali svolge una singola unità di lavoro. Insieme, le attività contenute in una pipeline svolgono un task.

- **Activity** - le Activity rappresentano i singoli passi computazionali che vengono svolti in una Pipeline.
- **Dataset** - un Dataset è la rappresentazione del dato. I Datasets rappresentano i dati che devono essere acquisiti (Data Source) o memorizzati (Sink) dalle Activities.
- **Linked Service** - simili alle connection string, definiscono la informazioni di connessione necessarie a Data Factory per connettere risorse esterne.
- **Integration Runtime** - rappresentano il ponte tra le Activities e i Linked Services. Un IR è l'ambiente di esecuzione utilizzato da Data Factory per eseguire Activities.
- **Trigger** - un Trigger determina quando una Pipeline deve essere eseguita. Possono essere schedulati, periodici o anche associati a specifiche tipologie di eventi.

Questo strumento viene tipicamente utilizzato per l'esecuzione di job di ETL/ELT (Data Integration, Analytics, Ingestion di dati verso un DW), attività di sincronizzazione periodiche, esecuzione di SSIS package e migrazione di on-premise SSIS package verso Azure.

2.3 Storage

I classici database difficilmente sono in grado di convogliare e rendere analizzabile la grossa mole di dati di cui si dispone al giorno d'oggi. Il problema principale è che tali tecnologie presuppongono di avere a che fare con un modello dati univoco mentre nel mondo dei Big Data è necessario integrare ed unire dati che possono essere anche molto diversi tra loro. Risulta necessario adottare tecnologie più evolute per far fronte alle nuove esigenze e ricorrere all'utilizzo del cloud computing per creare soluzioni di BI scalabili e adatte alla gestione dei Big Data.

Per sopperire alle mancanze dei database, nasce l'idea di **Data Lake** inteso come unico repository centralizzato nel quale i dati vengono memorizzati nella loro forma grezza e vengono processati attraverso query eseguite al momento del bisogno. Un Data Lake è in grado di memorizzare una quantità variabile di formati, da dati non strutturati, semi-strutturati, a dati strutturati.

Utilizzando un DataLake i costi iniziali di importazione del dato vengono notevolmente abbattuti poiché in esso vengono direttamente caricati i dati grezzi. In questo caso, infatti, non si parla più di ETL ma di ELT poiché

l'attività di trasformazione dei dati viene posticipata al momento in cui tali dati vengono richiesti dall'utente.

2.3.1 Azure Data Lake Storage

Azure Data Lake Storage (ADLS) è un file system elastico, scalabile e sicuro che supporta la semantica HDFS e funziona con l'ecosistema Apache Hadoop. Fornisce spazio di archiviazione illimitato adatto per un'ampia varietà di dati. È progettato per l'esecuzione di sistemi di larga scala che richiedono una grande capacità di elaborazione per analizzare grandi quantità di dati.

ADLS è adatto per archiviare tutti i tipi di dati provenienti da diverse fonti come dispositivi, applicazioni e molto altro. Consente agli utenti di archiviare dati strutturati e non. Inoltre, non richiede la definizione di uno schema prima che i dati vengano caricati. Queste caratteristiche lo rendono perfetto per essere utilizzato nel mondo dei Big Data. Ogni file ADLS è suddiviso in blocchi e questi blocchi sono distribuiti su più nodi. Non ci sono limiti sul numero di blocchi e di nodi.

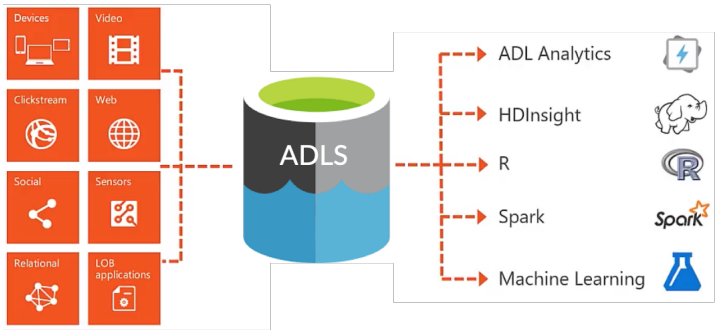


Figura 2.5: ADLS - Esempi di sorgenti e di strumenti di analisi

Una delle caratteristica che rende ADLS una delle soluzioni di archiviazione basate su Cloud più interessanti, è il suo modello *"pay-per-use"* senza costi iniziali. Consente agli utenti di pagare solo per il numero di gigabyte effettivamente occupati e per il numero di transazioni (lettura e scrittura) effettuate su tali dati.

Recentemente Microsoft ha introdotto ADLS Gen2 che è un superset di ADLS Gen1 e include nuove funzionalità dedicate alle analisi eseguite su Azure Blob Storage. Fino ad ora gli utenti dovevano scegliere tra un object store (Azure Blob Storage) o un file system (ADLS Gen1). ADLS Gen2 unifica object storage e file system per fornire l'accesso simultaneo agli stessi dati.

2.4 Model & Serve

Questa fase del flusso dei dati della Data Platform include tutti gli strumenti che consentono di modellare i dati in modo che gli utenti possano effettivamente interrogarli. Questo significa creare un modello dei dati e un'unica fonte di verità su cui l'azienda possa fare affidamento. È a questo punto che intervengono la BI, gli strumenti di reportistica e di analytics.

La piattaforma di analisi e reportistica BI utilizzata per la quasi totalità dei report già esistenti nell'azienda Cliente è **SAP BusinessObject**. Tra gli strumenti che Microsoft Azure mette a disposizione, in questo progetto sono stati utilizzati: **Azure Synapse Analytics** per il Data Warehousing; **Azure Analysis Services** per la modellazione dei dati; **Power BI** per la reportistica. Di seguito ne vengono descritte le principali caratteristiche.

2.4.1 SAP BusinessObject

SAP BusinessObject è una suite di strumenti per reportistica, visualizzzazione e condivisione di dati. È composta da diversi tools, pensati principalmente per tre scopi:

- **Reporting** - condividere informazioni dettagliate e formattate, principalmente come tabelle, con utenti interni o esterni all'organizzazione;
- **Dashboard & Apps** - creare dashboard e applicazioni dinamiche e personalizzabili per un supporto al processo decisionale "a colpo d'occhio";
- **Data discovery** - individuare, scoprire e analizzare dati.

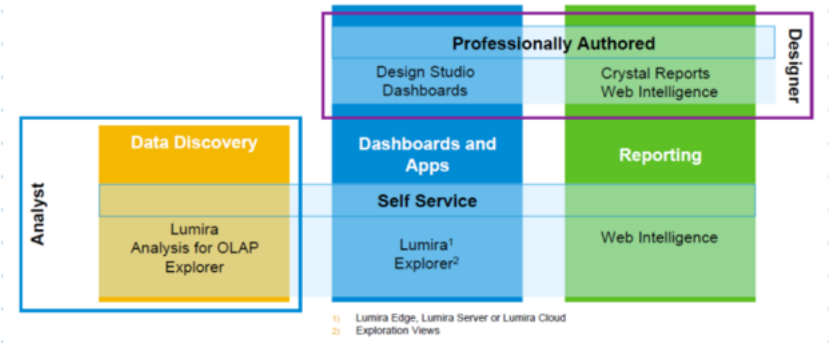


Figura 2.6: Suddivisione ad alto livello dei principali tools di SAP BO

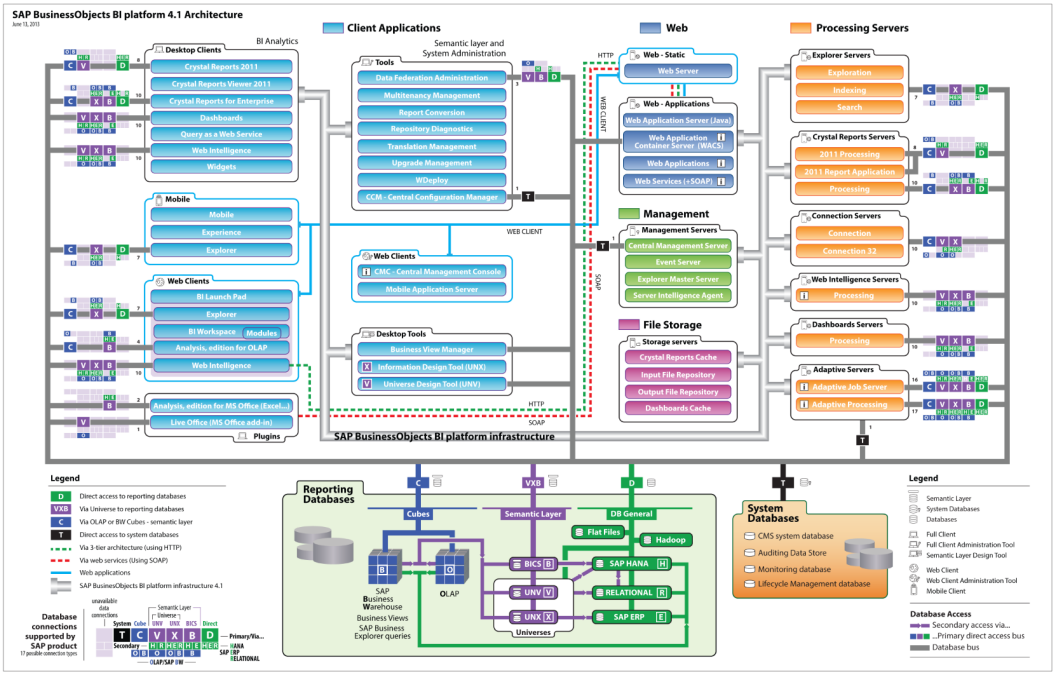


Figura 2.7: Architettura di SAP BO

L'architettura di SAP BO è molto articolata e viene mostrata in Figura 2.7. Può essere suddivisa in sei livelli principali:

- **Data Tier** - in figura è rappresentato dallo strato più basso, è costituito dal Reporting Database e da un insieme di database di sistema. Il Reporting Database viene utilizzato come sorgente dati per la creazione di report e dashboard. Solitamente è costituito da database relazionali ma è anche possibile utilizzare file di testo, documenti di Office o sistemi OLAP;

- **Processing Tier** - in figura è rappresentato dallo strato di colore arancione, è il livello che analizza i dati e produce report e altri tipi di output. Costituito da un insieme di server in esecuzione su uno o più host, è l'unico livello che accede direttamente ai database che contengono i dati;

- **Storage Tier** - in figura è rappresentato dallo strato di colore fucsia, è responsabile della gestione dei file, come documenti e report. Gestisce anche la memorizzazione della cache;

- **Management Tier** - in figura è rappresentato dallo strato di colore verde, coordina e controlla tutti i componenti che compongono la piattaforma di BI;

- **Web Tier** - costituito dalle componenti di colore blu insieme ai Web Client, contiene applicazioni distribuite su un application server Java, accessibili agli utenti finali tramite un browser Web. Offre una serie di servizi Web che forniscono funzionalità di BI agli strumenti software tramite l'application server. Esempi di servizi sono l'autenticazione della sessione, la gestione dei privilegi utente, ecc.

- **Client Tier** - costituito dalle componenti di colore azzurro, contiene tutte le applicazioni client desktop che interagiscono con la piattaforma di BI per fornire una varietà di funzionalità di reporting, analisi e amministrazione.

Attraverso la **Central Management Console (CMC)** è possibile eseguire la maggior parte delle attività amministrative quotidiane, tra cui la gestione degli utenti, dei contenuti e del server. Tutti i documenti e i file contenuti in SAP BO sono considerati **Oggetti**. Questi ultimi sono organizzati in **Cartelle** e **Categorie**. Le cartelle possono essere pubbliche o personali. È buona norma organizzare le cartelle secondo una struttura già esistente nell'azienda (esempio: rispecchiando i reparti aziendali). Tramite il CMC è possibile gestire la creazione e l'organizzazione degli oggetti e delle cartelle e definire i permessi di accesso dagli utenti verso di essi. È possibile inoltre gestire tante altre attività come l'amministrazione dei Processing Servers, la gestione dell'Auditing e l'organizzazione delle Connessioni e degli Universi.

Attraverso le **Connessioni** vengono definiti dei parametri che permettono alle applicazioni di connettersi ai DB (relazionali o OLAP) per creare poi i report. Gli **Universi**, definiti sulla base delle connessioni, consentono agli utenti di accedere ai database senza dover accedere alle strutture dati sottostanti. Di fatto, costituiscono un livello semantico costruito al di sopra dei database. La creazione degli Universi viene effettuata attraverso l'**Information Design Tool (IDT)**. Si tratta di uno strumento di progettazione di SAP BO che consente di estrarre, definire e manipolare i metadati provenienti da diversi origini utilizzando una connessione OLAP o relazionale, per creare Universi. L'Universo BO costituisce quindi una raccolta organizzata di oggetti (un modello) che consente agli utenti di analizzare e creare report utilizzando i dati aziendali in un linguaggio non tecnico, indipendentemente dalle origini dati e dalle strutture sottostanti. Un Universo si compone tipicamente di tre elementi:

- **Connessione** - definisce in che modo l'Universo può accedere a un'origine dati relazionale o OLAP;

- **Data Foundation** - schema che definisce le tabelle e i join rilevanti di uno più database relazionali;

- **Business Layer** - raccolta di metadati che fornisce un'astrazione delle entità di database relazionali o di cubi OLAP comprensibile per l'utente aziendale.

Una volta realizzato un universo si procede con la sua pubblicazione. A questo punto è possibile realizzare report che si basano su di esso. Lo strumento **Web Intelligence** consente di creare query e report utilizzando gli Universi creati con IDT. È disponibile come applicazione desktop oppure attraverso l'applicativo Web **BI Launch Pad**.

2.4.2 Azure Synapse Analytics

Azure Synapse Analytics è un servizio di analisi che combina Data Integration, Data Warehousing, Business Functions e Big Data Analytics. Il core di questo servizio è rappresentato dal **SQL pool** (precedentemente chiamato Azure SQL Data Warehouse). Si tratta di un Data Warehouse che utilizza un'architettura MPP (Massive Parallel Processing) per eseguire query complesse su grandi quantità di dati. Ne esistono di due tipi:

- **Dedicated SQL pool** - le risorse sono dedicate e sempre disponibili, che le si stia utilizzando o meno;
- **Serverless SQL pool** - le risorse vengono attivate solo se necessario.

L'architettura può essere sintetizzata come mostrato in Figura 2.8.

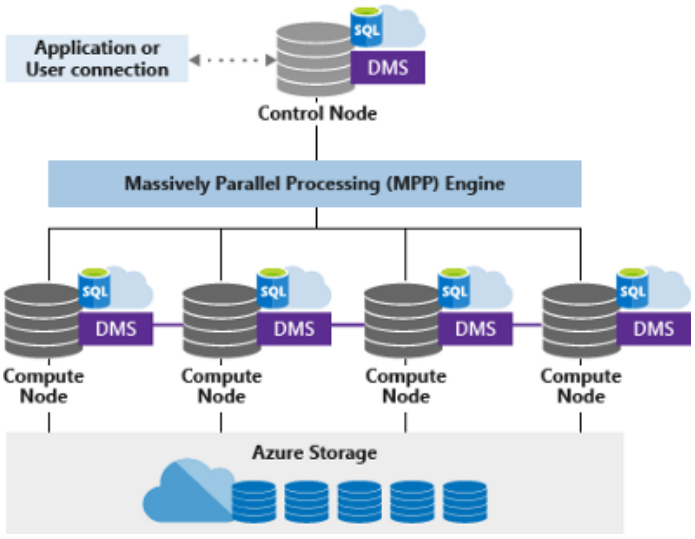


Figura 2.8: Dedicated SQL pool architecture

Il **Control Node** rappresenta la mente del Dedicated SQL Pool e costituisce la componente di front-end che interagisce con applicazioni e connessioni esterne. Il **motore MPP** viene eseguito sul Control Node per ottimizzare e coordinare l'esecuzione delle query parallele. I **Compute Nodes** eseguono le query e vengono utilizzati distribuendo equamente il carico. A seconda della quantità di DWU (Data Warehouse Units) scelta, i Compute Nodes possono variare in numerosità. In generale, più sono numerosi più potenza di calcolo si riesce ad ottenere. Il **Data Movement Service** (DMS) è il servizio che si occupa di gestire lo spostamento dei dati tra Compute Nodes. Alla base dell'architettura, troviamo una componente di storage sulla quale vengono memorizzati i risultati delle esecuzioni delle query parallele.

I dati vengono distribuiti tra i Compute Nodes sotto forma di shards. Quando una tabella viene creata, è possibile scegliere lo sharding pattern da utilizzare:

- **Replicated tables** - in questo caso, una copia completa dell'intera tabella viene replicata su ogni nodo. In questo modo non è necessario trasferire dati tra un nodo e l'altro quando una query deve essere eseguita ma può risultare un approccio problematico in presenza di database molto grandi;

- **Round-Robin table** - i dati di una tabella vengono distribuiti equamente tra i vari nodi seguendo un approccio che prevede l'estrazione causale del primo nodo per poi proseguire sequenzialmente con i successivi;

- **Hash-Distributed table** - scelta una colonna della tabella da distribuire, viene applicata una funzione hash che utilizza i valori di tale colonna per assegnare ogni riga ad un nodo.

Si possono distinguere tre principali tipologie di tabelle in base all'approccio di indexing utilizzato:

- **Clustered Columnstore Index** - le tabelle vengono indicizzate sulla base delle colonne e questo consente una maggiore compressione;

- **Clustered Index** - anche detto Clustered B-Tree, si tratta di un indice ad albero costruito a partire da una colonna della tabella. In presenza di query che esprimono filtri su tale colonna, la struttura ad albero consente di raggiungere velocemente i dati richiesti;

- **Heap Tables** - i dati non vengono ordinati e non esiste alcun indice. Se una tabella è etichettata come heap allora per trovare una riga in essa è necessario scorrere tutta la tabella.

Il partizionamento delle tabelle consente di dividere i dati in gruppi più piccoli. Nella maggior parte dei casi, le partizioni vengono create a partire da una colonna contenente date. Il partizionamento è supportato da tutti i tipi di tabelle (inclusi Clustered Columnstore, Clutered Index e Heap) e da tutti i tipi di distribuzione (hash, round robin, replicated tables). Può favorire la manutenzione dei dati e le prestazioni delle query. È importante porre particolare attenzione sul numero di partizioni create per ogni tabella poiché la creazione di una tabella con troppe partizioni, può compromettere le prestazioni in alcune circostanze. Questo è particolarmente vero per le tabelle Clustered Columnstore. Affinché il partizionamento sia utile, è importante capire quando utilizzarlo e il numero di partizioni da creare. Non esiste una regola rigida su quante partizioni generare, dipende dai dati a disposizione e da quante partizioni vengono caricate contemporaneamente. È importante considerare quante righe apparterranno a ciascuna partizione. Per una compressione e prestazioni ottimali delle tabelle Clustered Columnstore, tipicamente è necessario considerare un minimo di 1 milione di righe per partizione. Prima della creazione delle partizioni, il Dedicated SQL Pool applica già una forma di partizionamento delle tabelle dato dalla distribuzione di queste ultime tra i vari Compute Nodes disponibili. Qualsiasi partizionamento aggiunto ad una tabella, si aggiunge a tale distribuzione.
Esistono due principali modalità per caricare dati all'interno del SQL pool:

- **Single Client** - le modalità Single Client sfruttano strumenti come Azure Data Factory per eseguire un caricamento di massa (bulk);
- **Parallel Readers** - le modalità Parallel Readers si appoggiano a **PolyBase** per leggere dati da una sorgente esterna (ad esempio da un Azure Data Lake Storage) e caricarli nel SQL Pool. PolyBase ignora il Control Node e carica i dati direttamente nei Compute Nodes.

Oltre al SQL pool, Azure Synapse Analytics integra una vasta gamma di servizi che supportano la maggior parte delle attività del processo di Data Warehousing. Nel progetto in questione ci si è limitati ad utilizzare questo strumento esclusivamente per il SQL pool.

2.4.3 Azure Analysis Services

Azure Analysis Services è una Platform as a Service (PaaS) che permette di realizzare data models nel cloud. Si compone di un database in-memory e di un motore di elaborazione dati. Utilizza funzionalità avanzate di modellazione per combinare dati provenienti da più sorgenti, definire metriche e proteggere i dati in un unico modello tabellare. Il modello dati offre agli utenti un modo

più semplice e veloce per eseguire analisi utilizzando strumenti come Power BI ed Excel.

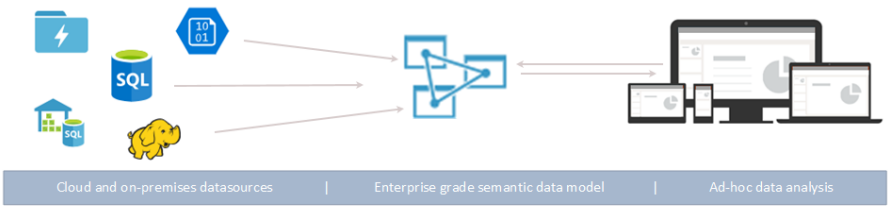


Figura 2.9: Analysis Services tra sorgenti e strumenti di analisi

Nasce originariamente su SQL Server nella sua versione on-premise. Rispetto a quest'ultima, la versione in cloud non supporta il modello multidimensionale per cui è possibile realizzare solamente modelli tabular.

Analysis Services è disponibile in tre differenti livelli di servizio: **Developer**, **Basic** o **Standard**. Ad ognuno di questi sono associati una serie di piani di costo che variano in base alla potenza di elaborazione e alla capacità di memoria richieste. Quando viene creato un server Analysis Services dal portale di Azure, è necessario specificare il livello di servizio e il piano di cui si necessita. In ogni momento è possibile scalare al livello e/o al piano che si preferisce.

I modelli possono essere realizzati direttamente con **Visual Studio** creando un progetto Analysis Services Tabular. Nel momento della creazione del progetto è necessario specificare le connessione alle sorgenti a partire dalle quali sarà poi possibile importare le tabelle che verranno utilizzate per la realizzazione del modello. Dovranno poi essere indicate le relazioni che sussistono tra le tabelle insieme alle eventuali misure e colonne calcolate. Queste ultime vengono definite utilizzando il linguaggio **DAX (Data Analysis Expressions)**. Le formule DAX includono funzioni, operatori e valori per eseguire calcoli avanzati e query sui dati contenuti nelle tabelle del modello. Tipicamente lo si utilizza per definire:

- **Misure** - si tratta di calcoli dinamici i cui risultati cambiano a seconda del contesto. Nella creazione di report, vengono utilizzate misure che supportano la combinazione e il filtraggio dei dati del modello tramite più attributi. Una volta creata una misura, è anche possibile utilizzarla all'interno delle formule di altre misure. Poiché i valori delle misure sono associate ad un contesto, vengono ricalcolate ogni volta che quest'ultimo subisce una variazione;

- **Colonne calcolate** - si tratta di una colonna che si aggiunge fisicamente ad una tabella esistente. La formula DAX definisce i valori che verranno

assegnati alla colonna. I valori vengono calcolati per ogni riga della colonna non appena viene specificata la formula DAX da utilizzare. La colonna calcolata, una volta aggiunta alla tabella, costituisce una sua estensione e dunque viene concretamente memorizzata sul database;

- **Tabelle calcolate** - si tratta di una tabella che viene derivata da uno o più tabelle presenti nello stesso modello. La formula DAX, in questo caso, specifica i valori che devono essere raggruppati nella nuova tabella. Una tabella calcolata viene ricalcolata se una delle tabelle da cui estrae i dati viene aggiornata o sottoposta a refresh;
- **Row-level security** - si tratta di una formula DAX che restituisce un valore booleano per stabilire quali righe di una tabella possono essere restituite nel risultato di una query da parte di utenti a cui è associato uno specifico ruolo. Quando si definisce un row-level security utilizzando una formula DAX, di fatto si crea un set di righe consentito. Ciò non nega l'accesso alle altre righe ma semplicemente non vengono restituite come parte del set di righe consentito.

Il linguaggio DAX può essere anche utilizzato per la definizione di query. A differenza delle formule DAX, che possono essere create solo in modelli dati tabular, le query DAX possono essere eseguite anche su modelli multidimensionali di Analysis Services. Una query DAX è un'istruzione, simile all'istruzione di SELECT in T-SQL. Il tipo più elementare di query DAX è un'istruzione di valutazione. Per esempio, nel Listato 2.1 viene mostrata una query DAX che restituisce l'elenco dei prodotti con un `SafetyStockLevel` inferiore a 200, ordinati in maniera crescente per `EnglishProductName`. È possibile creare misure come parte della query. Queste esisteranno solo per la durata della query.

```
EVALUATE
 ( FILTER ( 'DimProduct', [SafetyStockLevel] < 200 ) )
ORDER BY [EnglishProductName] ASC
```

Listato 2.1: Esempio di query DAX

Nel momento del deploy di un modello, è necessario indicare come destinazione il server Analysis Services precedentemente creato. In questo modo, il modello verrà distribuito su tale server e quindi successivamente acceduto e utilizzato tramite strumenti di reporting come Power BI. Nel caso specifico dell'utilizzo di Power BI, un'alternativa a questo procedimento è eseguire il deploy del modello su un repository Power BI Premium e poi accedere ad esso da Power BI Desktop passando per il Service. Dettagli su Power BI vengono descritti nel paragrafo successivo.

2.4.4 Power BI

Power BI è una raccolta di servizi software che interagiscono per trasformare sorgenti dati non correlate in informazioni coerenti, visivamente coinvolgenti e interattive. Le componenti principali di Power BI sono tre:

- **Power BI Desktop** - l'applicazione desktop che consente di connettersi, trasformare e visualizzare i dati. È possibile connettersi a diversi tipi di sorgente e creare oggetti visivi che poi possono essere condivisi con altre persone all'interno di un'organizzazione;
- **Power BI Service** - si tratta di una SaaS utilizzata principalmente per condividere con altri i report realizzati con Power BI Desktop;
- **Power BI Mobile** - Power BI offre un set di App per dispositivi mobili iOS, Android e Windows 10. Nelle App per dispositivi mobili è possibile connettersi e interagire con i dati archiviati nel Cloud e in locale.

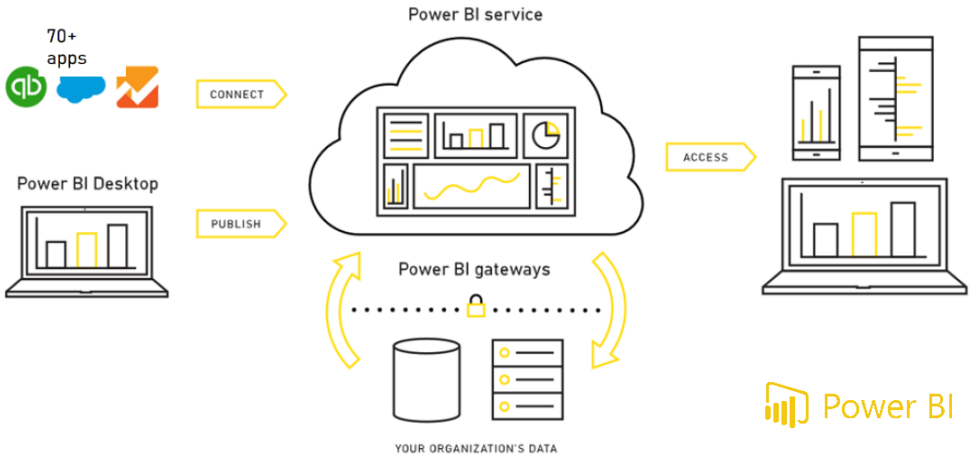


Figura 2.10: Power BI - Componenti principali

Il tipico flusso di lavoro prevede di connettersi alle sorgenti dati con Power BI Desktop e realizzare un report. Una volta completato, il report viene pubblicato su Power BI Service e in questo modo lo si condivide con tutti gli utenti che hanno accesso al servizio. Attraverso i Power BI gateways, il Service è in grado di collegarsi alle sorgenti per consentire la visualizzazione dei report. A questo punto, sui dispositivi mobili è possibile visualizzare ed interagire con i report pubblicati.

Vengono offerte anche funzionalità di ETL e modellazione dei dati che, come mostrato in Figura 2.11, sono rese possibili dall'integrazione in Power BI degli strumenti Power Query e Analysis Services. **Power Query** può essere utilizzato come strumento di ETL e quindi è il responsabile dell'estrazione, trasformazione e caricamento dei dati all'interno di Analysis Services che viene invece utilizzato per la creazione di un modello dati interrogabile. In generale, però, è preferibile eseguire queste attività a monte dell'architettura della Data Platform e utilizzare Power BI solo per il consumo dei dati.

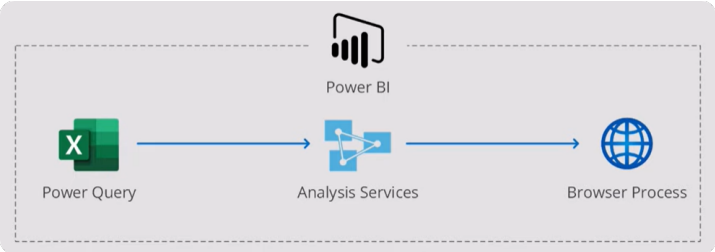


Figura 2.11: Power BI - ETL e modellazione dati

Esistono due modalità con cui è possibile caricare dati in Power BI:

- **Import query** - tutti i dati vengono caricati su Power BI in modo da essere indipendenti dalle sorgenti. Per aggiornare i dati è necessario eseguire l'operazione di Refresh;
- **Direct query** - viene mantenuto attivo un collegamento verso le sorgenti in modo che le query eseguite su Power BI vengano ridirezionate ed eseguite direttamente su di esse.

Le operazioni che possono essere eseguite su Power BI dipendono da tre fattori:

- il tipo di licenza e subscription che si sta utilizzando;
- dove è archiviato il contenuto;
- i ruoli e i permessi assegnati.

Ogni utente del Power BI Service, dispone di una licenza gratuita, una licenza Pro o una licenza Premium. La licenza **Premium** consente di archiviare contenuti in un contenitore virtuale chiamato *Capacity*. Chiunque disponga dell'autorizzazione, può visualizzare il contenuto archiviato nella Premium Capacity senza acquistare singole licenze Power BI Pro o Premium per singolo utente. La Premium Capacity consente la distribuzione dei contenuti da parte degli utenti Pro senza richiedere licenze Pro per i destinatari che visualizzano

il contenuto. Tipicamente, la persona che crea il contenuto nella Premium Capacity, utilizza una licenza Pro per connettersi alle sorgenti, modellare i dati e creare report e dashboard che vengono salvati in un workspace della Premium Capacity. Gli utenti senza una licenza Pro possono comunque accedere al workspace a patto che venga loro assegnato un ruolo in quel workspace. All'interno di un workspace è possibile assegnare ruoli che determinano la misura in cui i colleghi possono interagire con il contenuto. Esempi di ruoli sono: Viewer, Contributor, Member e Administrator.

Un ulteriore strumento messo a disposizione è **Power BI Report Builder** per la creazione di Paginated Reports da condividere sul Power BI Service. I Paginated Reports sono particolari tipologie di report formattati appositamente per adattarsi bene ad una pagina. Vengono infatti spesso utilizzati per rapporti operativi o per la stampa di moduli come fatture o trascrizioni. Visualizzano tutti i dati in una tabella, anche se la tabella si estende su più pagine. Le componenti fondamentali che costituiscono Power BI Report Builder sono le seguenti:

- **Data Source** - costituiscono le connessioni verso le sorgenti. Nel caso in cui si disponga di un modello memorizzato in un repository Power BI, è possibile collegarsi a tale modello in modo da poter recuperare i dati da esso;

- **Dataset** - il termine "dataset" viene utilizzato sia in Power BI, sia in Power BI Report Builder ma assume significati differenti. Un dataset in Report Builder rappresenta il risultato di una query che restituisce i dati di cui un report ha bisogno a partire da una data source. È possibile creare uno o più dataset all'interno di un report e ognuno di questi deve avere un nome univoco. La query che permette di costruire un dataset può essere definita nei linguaggi SQL, DAX o MDX a seconda del tipo di sorgente utilizzata. Nel caso in cui la sorgente sia un modello Analysis Services è possibile utilizzare solo i linguaggi DAX o MDX. Esiste anche la possibilità di utilizzare lo strumento Query Designer che fornisce supporto nella creazione della query;

- **Parameter** - i parametri vengono tipicamente utilizzati per applicare filtri sui report. Esistono due tipi di parametri: **Report Parameter** e **Query Parameter**. I primi vengono tipicamente utilizzati per filtrare i dati al tempo di esecuzione del report. I valori di questi parametri vengono specificati dall'utente nel momento in cui il report viene eseguito. I Report Parameters devono essere mappati verso corrispettivi Query Parameters di un dataset. Questi ultimi sono parametri direttamente iniettati all'interno di una query di un dataset e permettono effettivamente di filtrarlo.

2.5 Azure DevOps

Azure DevOps offre servizi per pianificare il lavoro, collaborare allo sviluppo del codice e creare e distribuire applicazioni. Consente alle organizzazioni di creare e migliorare i prodotti a un ritmo più rapido rispetto a quanto possibile con i tradizionali approcci di sviluppo software.

Tra i servizi che Azure DevOps mette a disposizione troviamo:

- **Azure Repos** - fornisce repository Git o TFVC per il controllo di versione;
- **Azure Pipelines** - offre servizi di build and release per supportare la continuous integration e delivery delle applicazioni;
- **Azure Boards** - offre una suite di strumenti Agile per supportare il lavoro di pianificazione e monitoraggio, difetti del codice, problemi vari utilizzando metodi Kanban e Scrum;
- **Azure Test Plans** - offre vari strumenti per effettuare il testing di App, inclusi test manuali/esplorativi e continuous testing;
- **Azure Artifacts** - consente al team di condividere pacchetti come Maven, npm, NuGet e altri da origini pubbliche e private e di integrare la condivisione di tali pacchetti all'interno delle pipelines.

Azure DevOps è disponibile in due versioni:

- **Azure DevOps Services** - rappresenta l'offerta cloud di Azure DevOps. Offre un servizio scalabile, affidabile e disponibile a livello globale;
- **Azure DevOps Server** - rappresenta l'offerta on-premise di Azure DevOps. Si basa su un SQL Server back-end. Questa soluzione viene scelta quando si ha la necessità che i dati rimangano all'intero della rete.

Anche se entrambe le offerte offrono gli stessi servizi essenziali, Azure DevOps Services introduce i classici vantaggi derivanti dall'utilizzo di una soluzione Cloud. In compenso, alcune funzionalità di Azure DevOps Server non sono disponibili in Azure DevOps Services.

Per il progetto di tesi si è fatto uso del servizio Azure Repos per la realizzazione di un repository Git nel quale vengono caricati i modelli dati. In questo modo, i modelli risultano facilmente accessibili da tutti i membri del team e allo stesso tempo, sfruttando il meccanismo del branching, più membri del team possano lavorare contemporaneamente ad attività che coinvolgono lo stesso modello senza entrare in conflitto.

Il repository creato è accessibile attraverso Visual Studio dal quale è possibile eseguire la clone e, successivamente, tutte le principali attività di versioning del codice: dalla creazione di branch al commit delle modifiche. In questo caso, non è stato seguito un Git workflow specifico: ci si è limitati a mantenere su un branch Master la versione stabile del modello creando dei branch separati ogni volta che risultava necessario applicarvi delle modifiche per supportare funzionalità aggiuntive.

Capitolo 3

Migrazione del Data Warehouse

3.1 Obiettivo

L'obiettivo del Cliente è quello di valutare e adottare il miglior processo evolutivo per il proprio sistema di Business Intelligence, al fine di supportare gli obiettivi strategici di business. È richiesta un'analisi della situazione attuale (AS-IS) del sistema di BI in uso che permetta di valutare e adottare una nuova architettura (TO-BE) che soddisfi le nuove esigenze. Deve quindi essere progettato e sviluppato un processo di migrazione dal vecchio sistema a quello nuovo che non si limiti solamente a replicare quanto già esistente ma anche a migliorarlo, estendendone le capacità analitiche complessive. I principali benefici attesi sono i seguenti:

- disponibilità di una Enterprise Data Platform che apra la strada a progetti interfunzionali con un modello semantico comune;
- centralizzazione, riutilizzabilità e gestione comune dei dati;
- riduzione del Time-to-Market per lo sviluppo di nuovi progetti di business analytics;
- disponibilità di un'architettura dati moderna, flessibile, scalabile, affidabile e sicura;
- disponibilità di una Data Platform con supporto per iniziative in ambito Big Data, Advanced Analytics, Data Science e IoT;
- linee guida per la definizione di Data Governance Office con specifici ruoli e responsabilità;
- coordinamento collaborativo tra i reparti IT e Business;
- cultura dei dati condivisa in tutta l'organizzazione.

3.2 Analisi AS-IS

Lo scopo dell'analisi AS-IS è quello di valutare la situazione attuale del sistema di BI in uso nell'azienda Cliente. Si struttura come segue:

1. *Mapping delle applicazioni e dell'architettura tecnologica corrente* - con il supporto di professionisti dell'azienda Cliente, si è seguito uno schema logico architetturale collaudato su molte esperienze pregresse che ha consentito di identificare i sistemi e le applicazioni analitiche attualmente in uso insieme alle principali interdipendenze tra questi ultimi;

2. *Analisi delle sorgenti dati e dei flussi* - sono state identificate tutte le sorgenti insieme alle principali trasformazioni (business rules) applicate ai flussi dati ETL;

3. *Collezione, analisi e classificazione dei report esistenti* - analizzando gli strumenti di front-end, si è provveduto a censire tutti gli oggetti analitici e a rilevare i principali KPI;

4. *AS-IS Service Level Agreement* - è stata eseguita una misura del valore del sistema, valutando lo stato dell'arte.

3.2.1 Architettura AS-IS

In Figura 3.1 viene mostrata l'architettura attualmente in uso nell'azienda Cliente.

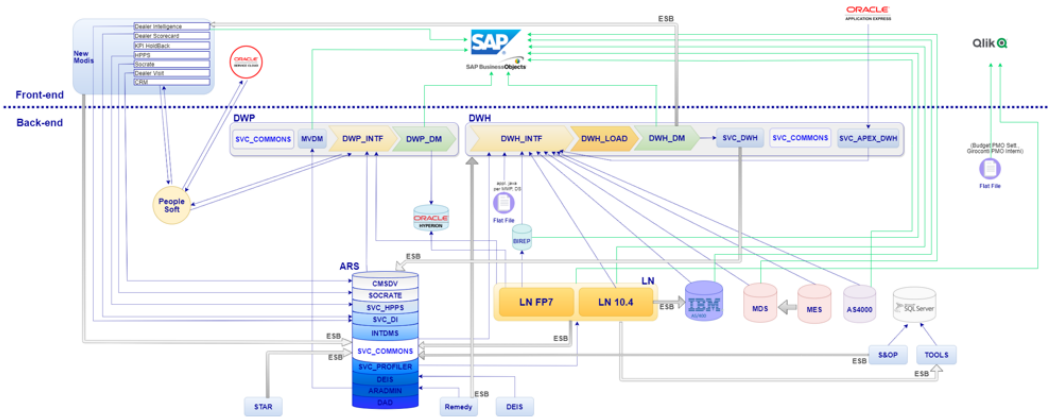


Figura 3.1: Architettura AS-IS completa

La piattaforma di analisi e reporting BI utilizzata per la creazione della quasi totalità dei report presenti è **SAP BusinessObject**. **Qlik Sense** è un

applicativo di reportistica secondario dedicato alla creazione di report relativi a richieste/ordini di acquisto, budget PMO settimanalizzato e giroconti PMO interni.

Le sorgenti dati risiedono su database **Oracle** e consistono principalmente di data set LN e CRM. **Infor LN** è il sistema gestionale utilizzato dal Cliente, a partire dal quale vengono prodotti la maggior parte dei dati. È basato su Oracle ed attualmente è presente in due versioni: **LN FP7** e **LN 10.4**. Lo strumento utilizzato per l'ETL è **Data Stage 11**. Per la pianificazione e l'esecuzione dei flussi vengono utilizzati degli script in esecuzione tramite **Crontab**. La Figura 3.2 mostra una suddivisione in livelli dell'architettura.

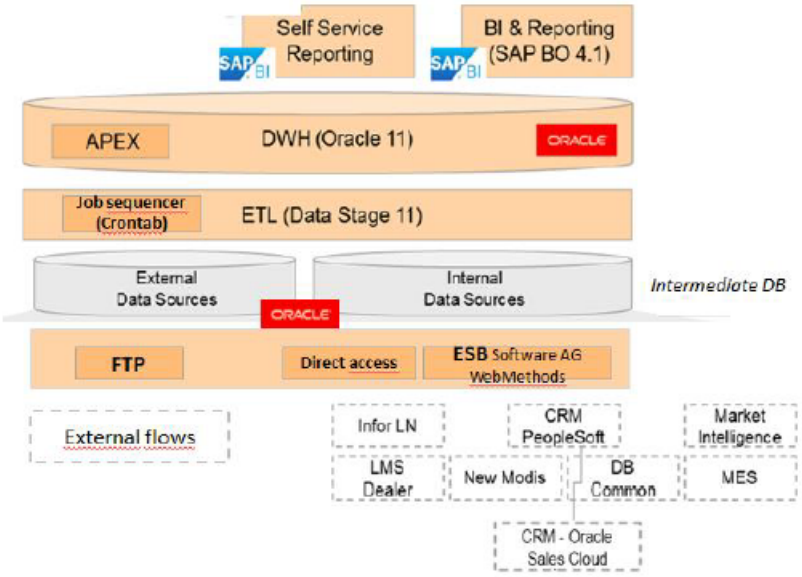


Figura 3.2: Suddivisione in livelli dell'architettura AS-IS

Come si può notare viene adottata un'architettura a 3 livelli alla base della quale è presente il livello delle sorgenti. I dati prodotti da queste ultime, vengono caricati tramite diverse modalità all'interno di un database intermedio a partire dal quale, attraverso l'utilizzo di Data Stage, i dati vengono caricati nel Data Warehouse. In cima all'architettura troviamo SAP, utilizzato per il reporting.

Il Data Warehouse si basa su DB Oracle 11 ed è composto da tre layers (DWH_INTF, DWH_LOAD, DWH_DM). La maggior parte delle business rules presenti sono implementate in DWH_INTF o in DWH_LOAD. Il layer DWH_DM è quello che contiene i dati su cui si appoggiano effettivamente i report realizzati con SAP BusinessObject. In Figura 3.3 viene mostrata la struttura e i flussi di caricamento.

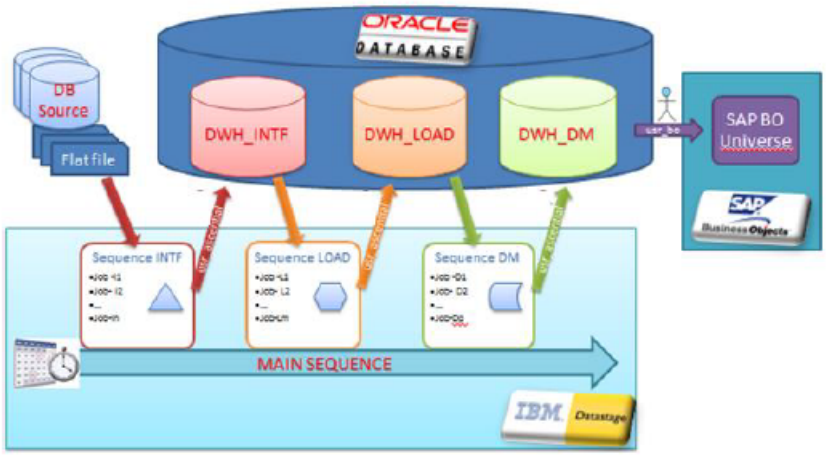


Figura 3.3: Suddivisione in livelli del Data Warehouse

3.2.2 Analisi preliminare

Lo scopo dell'analisi preliminare è quella di definire perimetro del problema e obiettivi in maniera più precisa.

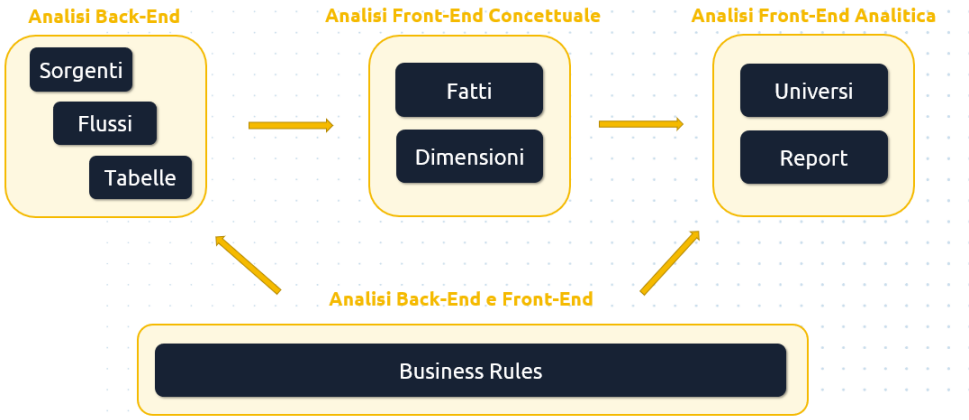


Figura 3.4: Analisi preliminare - Diagramma delle attività

Come mostrato in Figura 3.4, l'analisi preliminare può essere suddivisa in quattro macro-attività:

- **Analisi Back-End** - è stata eseguita l'analisi dell'architettura AS-IS con conseguente produzione di un diagramma architetturale e di un glossario dell'architettura. Il diagramma architetturale prodotto è quello mostrato in Figura 3.1 mentre il glossario descrive singolarmente i singoli componenti del diagramma. Viene inoltre effettuato il censimento dei flussi ETL analizzandone le caratteristiche. Dall'analisi emerge che il numero di flussi è molto elevato (3710 processi);

- **Analisi Front-End Concettuale** - è stato realizzato un macro-modello di analisi concettuale (DFM) evidenziando fatti, dimensioni e gerarchie;

- **Analisi Front-End Analitica** - censimento, analisi, classificazione e statistiche sui report presenti su SAP BO e i relativi universi. È stata eseguita una prima analisi della reportistica presente su SAP BO con l'obiettivo di capire quali report fossero effettivamente utilizzati e quali fossero invece obsoleti. Dopodiché, per ogni report, sono state raccolte importanti informazioni tra cui: l'entità, l'universo, la cartella, la tipologia, il grado di unicità, ecc. Infine, è stato stimato il numero di giorni necessari per l'implementazione. Tutti i report sono stati raggruppati in **Data Entity**. Una Data Entity definisce un gruppo di report che afferisce ad uno stesso fatto di business. I fatti di business sono a loro volta raggruppati in Business Area. In Tabella 3.1 viene mostrato l'elenco completo delle Data Entity suddivise per Business Area;

- **Analisi Back-End e Front-End** - sono state identificate le principali Business Rules e definiti i Service Levels dell'architettura AS IS. L'obiettivo principale del censimento delle Business Rules è analizzare la complessità delle regole presenti all'interno dei flussi ETL e dei report. Per ogni Business Rule sono stati censiti: l'ambito, il layer di appartenenza (back-end o front-end), la complessità e l'effort. I Service Levels sono variabili standard con le quali è possibile misurare il valore di un sistema di supporto alle decisioni di una Data Platform in termini di premesse valutabili tecnicamente tramite diversi accorgimenti che abilitano tali servizi. L'obiettivo è supportare il processo di evoluzione in atto rispetto alle aspettative di fruizione, valutando lo stato dell'arte. Con l'analisi dei Service Levels è stato verificato che ci sono potenzialità non raggiunte a causa della mancanza di alcune funzionalità di livello inferiore ma fondamentali.

Business Area	Data Entity	AS-IS Reports	TO-BE Reports
Sales & Commercial Logistics	Preowned	12	7
	Demo Policy	2	1
	CR, CCF, Stock	31	21
	Orders	17	13
	Portfolio	9	7
	PO e Order Call	14	9
	Cancellations	6	5
	Pre-order	2	1
	Pre-order OPT	1	1
	OPT e Tailormade	45	17
	Other	37	27
HR	Gestione accessi	4	1
Academy	Dealer compliancy	6	4
	Gestione corsi	12	7
AFC	Controllo di gestione	63	4
	RDA	1	1
	Other	2	2
Spare Parts	Fatturato	20	10
	Back order	4	1
	Magazzino	12	2
	Supply Chain	1	1
	Trasporti inbound e outbound	8	3
	Monitoraggio attività	3	2
	Other	8	6
Customer Service / After Sales	Fatturato	12	7
	Garanzie	6	5
	Service Entry	27	19
	Other	5	4
Quality & Purchasing	Ispezioni	9	9
	Magazzino	1	1
	Resi rete	3	2
	MAPS-SIGIP	8	4
	Other	6	1
	Benestare ECS	10	5
	CQC ECS	1	1
	Deroghe ECS	4	1
	ROS ECS	4	3
	Tracciabilità sicurezza	1	1
Logistics	Logistica	25	8
	CQC ECS	1	1
	Magazzino	8	5
	EOL China	1	1
	Fattori industriali	2	2
	Tracking avanzamenti di linea	1	1
	Other	4	4
ICT Remedy	Monitoraggio attività	3	1
	Ticket Remedy	4	1

Tabella 3.1: Elenco delle Data Entity suddivise per Business Area

Il grado di unicità dei singoli report, estrapolato durante l'Analisi Front-End Analitica, viene utilizzato per stabilire quali report risultano ridondanti o possono essere unificati ad altri report. Dai risultati di questa analisi emerge la stima del numero di report TO-BE indicata in Tabella 3.1. Vengono inoltre evidenziate le Data Entity su cui ci si è concentrati per il progetto di tesi.

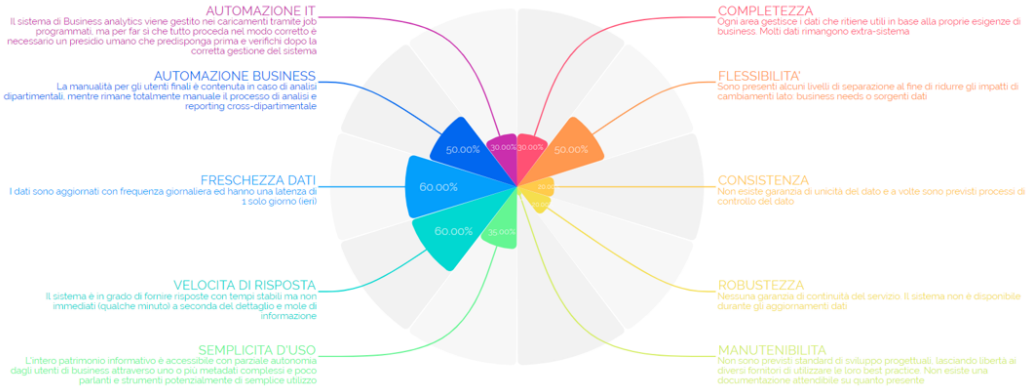


Figura 3.5: Diagramma dei Service Levels

In figura 3.5 viene riassunto lo stato attuale dei sistemi e delle applicazioni di analisi disponibili nell'azienda Cliente dal punto di vista dei Service Levels. Questi ultimi vengono misurati tenendo conto di 10 differenti assi. Il punteggio lungo ciascun asse indica la qualità del sistema per quanto riguarda l'aspetto trattato da quell'asse e viene misurato tenendo conto degli attuali standard di mercato insieme all'esperienza pluriennale di IConsulting su questo tipo di progetti. Considerando quello mostrato in figura come lo stato di partenza, è possibile evidenziare le aree di miglioramento. Quelle che risultano avere il miglior rapporto Effort/Benefit sono le seguenti:

- **Manutenibilità (5%)** - non esiste né una documentazione tecnica né funzionale che sia attendibile;
- **Automazione IT (30%)** - non è presente un sistema di controllo di quadratura tecnica automatica per tutti i dati presentati a sistema. Inoltre, non viene utilizzato un tool grafico per monitorare le eccezioni emerse dal caricamento dei dati.

Alla luce di quanto emerso dall'analisi preliminare, l'approccio proposto per la realizzazione del progetto è un'attività di "Reverse Engineering" collaborativa con il personale esperto dell'azienda Cliente.

3.3 Analisi TO-BE

Lo scopo di questa analisi è quello di valutare il miglior processo evolutivo per il sistema di Business Intelligence utilizzato dal Cliente. L'analisi TO-BE si struttura come segue:

1. *Definizione dei casi d'uso sui nuovi requisiti di business* - sulla base dei requisiti aziendali raccolti durante le recenti valutazioni e sulla base dell'esperienza di IConsulting nel settore automobilistico e del lusso, viene definita una lista di casi d'uso validi come proposta per gli utenti aziendali;

2. *Definizione dei casi d'uso sugli Analytics esistenti* - sulla base dell'analisi AS-IS, viene definito un elenco di casi d'uso sugli attuali strumenti di Analytics;

3. *Presentazione dei casi d'uso agli utenti di business* - la lista dei casi d'uso precedentemente definita viene presentata ai business stakeholders;

4. *Definizione dei requisiti dell'architettura TO-BE* - sulla base dei casi d'uso discussi con il Cliente e sfruttando le precedenti esperienze di IConsulting in progetti analoghi, viene definito l'elenco dei requisiti dell'architettura TO-BE in termini di requisiti architetturali, applicativi e organizzativi;

5. *Orientamento architetturale e POC* - vengono proposti possibili scenari architetturali differenti, identificando i requisiti di sistema e le possibili soluzioni. Su richiesta esplicita, IConsulting supporta il Cliente nella valutazione Vendor/Software attraverso POC di approfondimento su strumenti specifici;

6. *Data Governance Guidelines* - vengono condivise con il Cliente le linee guida sulla governance dei dati e viene definito un approccio per gestire questa attività.

3.3.1 Architettura TO-BE

L'architettura TO-BE nasce da un'attività di analisi svolta da IConsulting in collaborazione con il Cliente durante la quale sono state condivise alcune linee guida e tendenze tecnologiche rilevanti che hanno portato alla definizione di un'architettura logica di riferimento che soddisfacesse tutte le esigenze precedentemente emerse. A partire da questa architettura logica, sono stati declinati diversi scenari tecnologici sui quali si sono focalizzati approfondimenti e discussioni per presentare i pro e i contro di ciascuno scenario. Una volta definito l'orientamento verso le tecnologie da utilizzare, sono stati organizzati POC di approfondimento su strumenti specifici coinvolgendo anche i SW vendors. I POC sono stati realizzati dai SW vendors e IConsulting ha agito come mediatore per definire i requisiti chiave comuni e aiutare nella valutazione dei risultati.

L'architettura TO-BE che è stata concordata si allinea a quella mostrata in Figura 3.6.

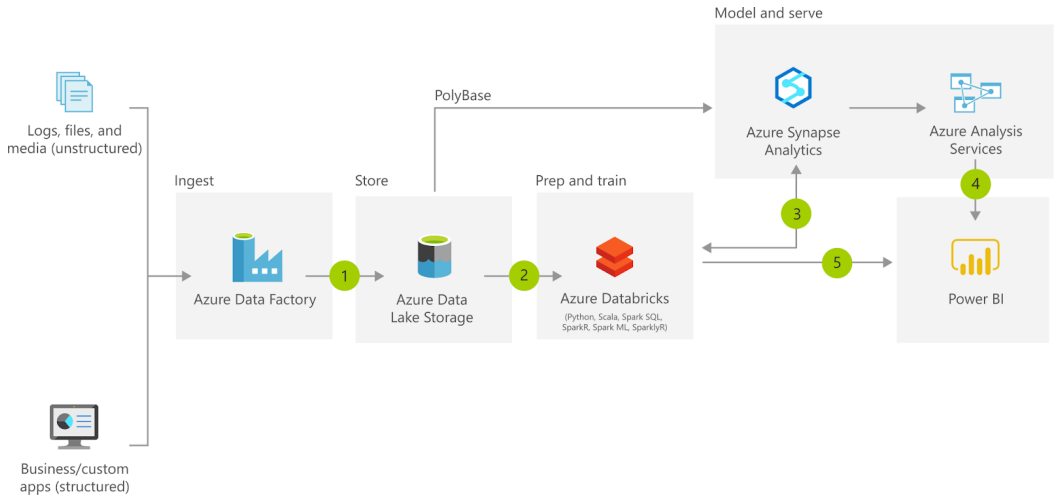


Figura 3.6: Diagramma dell'architettura TO-BE

Per la fase di Ingestion, **Azure Data Factory** viene utilizzato come strumento di ETL/ELT. Diversamente da come mostrato nell'immagine, i dati sorgente vengono nella maggior parte dei casi caricati direttamente su **Azure Synapse Analytics**. Solo in presenza di tabelle dalle dimensioni molto elevate, è previsto il caricamento intermedio su **Azure Data Lake Storage** per poi procedere con il caricamento su Azure Synapse Analytics attraverso l'utilizzo di **PolyBase**. Questo passaggio intermedio consente di ottimizzare le performance.

PolyBase è lo strumento di SQL Server per la Data Virtualization. Con Data Virtualization si intende la possibilità di interrogare dati che si trovano all'esterno di un database come se fossero locali. L'esempio più semplice consiste nel rendere disponibile alle query una tabella di database di terze parti (esempio: Oracle) come se fosse una tabella di SQL Server. Con l'aiuto di PolyBase, Azure Synapse Analytics è in grado di importare ed esportare dati dal Data Lake sfruttando la velocità offerta dalle tecniche di indicizzazione Clustered Columnstore senza la necessità di dover creare un processo di ETL separato o di utilizzare uno strumento dedicato per l'importazione dei dati. Ulteriori caratteristiche che permettono a PolyBase di migliorare le performance complessive sono:

- la possibilità di delegare parte della computazione alla sorgente esterna per ottimizzare la query - l'ottimizzatore della query prende una decisione basata sul costo che si avrebbe nell'eseguire il push del calcolo verso Hadoop. Il push del calcolo si traduce in job MapReduce che sfruttano le risorse computazionali distribuite di Hadoop;

- la possibilità di eseguire un trasferimento dati parallelo sfruttando i vari nodi su cui è distribuito l'Azure Data Lake Store.

Nell'architettura TO-BE, **Azure Analysis Services** viene utilizzato per la modellazione tabellare dei cubi del DW. I modelli realizzati attraverso Analysis Services vengono sottoposti a controllo di versione attraverso l'utilizzo di **Azure DevOps**.

Per la reportistica vengono sfruttati gli strumenti **Power BI** e **Power BI Report Builder**. Il primo viene utilizzato in presenza di report che presentano KPI complessi e per i quali gli utenti finali necessitano di un ventaglio analitico più ampio: si ricorre a Power BI nel momento in cui si vogliono estendere le capacità analitiche di un report esistente aggiungendo ad esempio grafici ed indicatori. Il secondo viene utilizzato in presenza di report tabellari per i quali non è richiesto di estendere le capacità analitiche ed è sufficiente replicare quanto già esistente.

In Figura 3.6, viene mostrato anche lo strumento **Azure Databricks** che è stato utilizzato per attività di machine learning non legate alla migrazione e quindi non incluse nel progetto di tesi.

3.3.2 Ottimizzazioni architetturali

Riflessioni su Analysis Services e Synapse Analytics

Una possibile alternativa architetturale che è stata valutata, consiste nel caricare i dati provenienti dalle sorgenti direttamente su Azure Analysis Services, senza ricorrere al livello rappresentato da Azure Synapse Analytics. In precedenza, infatti, si è visto che Analysis Services dispone di un database in-memory che può essere direttamente utilizzato per contenere i dati di analisi.

Figura 3.7: Alternativa architetturale senza Azure Synapse Analytics

Conseguenza della natura di un modello Tabular Analysis Services unico e collocato su un ambiente Azure, è la quantità di memoria utilizzata dal modello. Analysis Services dispone di tre differenti livelli di servizio a ciascuno dei quali è associata una serie di piani di costo che variano in base alla potenza di elaborazione e alla capacità di memoria richieste. Attualmente, il dimensionamento massimo del servizio offerto da Microsoft corrisponde al livello S9. Con questa configurazione, vengono resi disponibili 400GB di memoria. Vista l'elevata numerosità di tabelle presenti sulle sorgenti e poiché si prevede una crescita costante nel tempo della quantità di dati, questa soluzione viene scartata per lasciare spazio a quella che include un layer aggiuntivo rappresentato da Azure Synapse Analytics. I dati quindi verranno importati dalle sorgenti a Synapse, mentre Analysis Services verrà utilizzato esclusivamente per la realizzazione del modello. Power BI si appoggerà poi su tale modello per lo sviluppo dei report.

Accesso ai dati con Power BI

In precedenza si è visto che Power BI mette a disposizione due modalità di accesso ai dati: *Import* e *Direct Query*. Sono state valutate attentamente entrambe e, nonostante la prima richieda un adeguato dimensionamento dell'ambiente Power BI per far spazio alle tabelle del modello, possiede come vantaggio principale la maggiore velocità di risposta. Per questo motivo è stata preferita rispetto alla seconda modalità che richiede tempi di attesa più lunghi per il recupero del dato.

Gestione e distribuzione delle tabelle in Synapse

Allo scopo di parallelizzare le operazioni che vengono effettuate sui diversi nodi che compongono l'infrastruttura di Synapse, è necessario scegliere il tipo di distribuzione più adatto in base al contenuto e alle caratteristiche delle tabelle. La scelta della distribuzione corretta permette di ottimizzare le performance e di evitare problemi legati alla saturazione della memoria temporanea. L'approccio "Round Robin" è quello che ottimizza la velocità di caricamento dei dati ed è stato utilizzato per popolare il primo strato dell'architettura del nuovo DW. Per gli strati successivi sono stati utilizzati gli approcci di distribuzione "Hash-Distributed" e "Replicated". La scelta del primo rispetto al secondo è dipesa dalla dimensione della tabella e dal tipo di operazioni da eseguire su di essa. La distribuzione Hash è da preferire in presenza di tabelle di grandi dimensioni per le quali sono previste frequenti operazioni di inserimento, aggiornamento ed eliminazione. La distribuzione Replicated è preferibile per tabelle di dimensione non troppo elevata utilizzate in join che richiederebbero altrimenti lo spostamento dei dati. È stata inoltre modificata la modalità di gestione delle tabelle, che di default viene impostata secondo l'opzione di indexing "Heap Tables", scegliendo un approccio più efficiente basato su "Clustered Columnstore Index".

Processo di ETL

Poiché l'azienda Cliente richiede esplicitamente che il tempo di attesa che intercorre tra il caricamento dei dati nella nuova architettura a partire dalle sorgenti fino alla loro effettiva disponibilità in Power BI, sia il più breve possibile, si è deciso di implementare un meccanismo di Ingestion basato su un approccio pull. È stata creata sulle sorgenti Oracle una tabella di log nella quale, per ogni flusso Data Stage, viene registrato il timestamp di inizio e di fine caricamento della tabella. Poi è stata creata una pipeline Data Factory che monitora costantemente la tabella Oracle e la confronta con un'altra tabella di log che tiene traccia, per ogni tabella caricata nella nuova architettura,

dello stato di caricamento e della data. Il risultato del confronto permette di distinguere quali dati risultano nuovi e quali invece sono già stati caricati consentendo quindi di limitare solo ai primi l'effettivo caricamento. Maggiori dettagli verranno forniti in seguito.

3.3.3 Standard di sviluppo

Un obiettivo che IConsulting si pone è quello di realizzare un sistema efficace ed efficiente nella gestione dei flussi di dati aziendali. Per questo motivo è stata applicata una metodologia che divide su vari livelli la struttura della piattaforma dati da realizzare.

La suddivisione in layers offre una serie di vantaggi. I principali sono i seguenti:

- ogni livello ha un compito molto specifico e mira ad aumentare la manutenibilità, la tracciabilità e la ricostruibilità dei dati. I dati evolvono gradualmente verso una "versione" più pregiata secondo un flusso lineare unidirezionale facilmente seguibile;
- ciascun livello utilizza i dati di output del livello precedente, seguendo una prospettiva di raffinamento lineare e incrementale delle informazioni;
- ogni fonte tratta le informazioni una volta, quindi la ridondanza viene evitata lungo tutta la supply chain del livello.

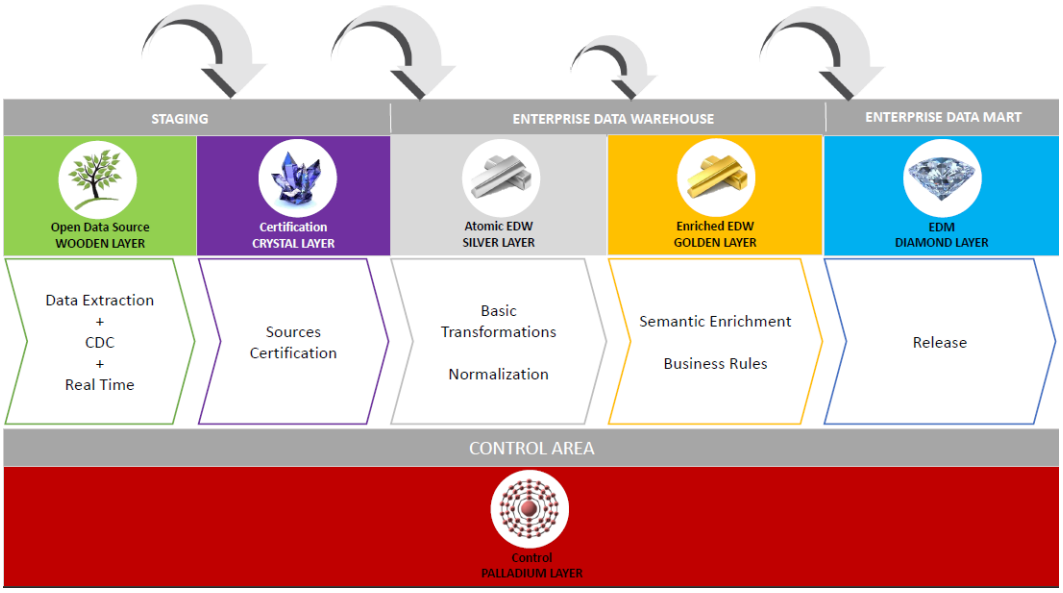


Figura 3.8: Enterprise Data Warehouse Layering

I dati destinati al consumo e che possono essere utilizzati come sorgente per altri sistemi esterni sono quelli che in Figura 3.8 sono contenuti nel **Diamond Layer**. Questo livello contiene un sottoinsieme mirato e ottimizzato dei dati provenienti dai livelli precedenti che rappresentano l'intero patrimonio informativo. Il **Control Layer** è alla base delle performance di tutti gli altri livelli. Ad esso sono delegati vari compiti, come il monitoraggio, il tracciamento dei flussi per impedire esecuzioni concorrenti non consentite, il tempo e lo stato di esecuzione dei vari flussi. Grazie a questa struttura si ha la garanzia che ogni volta che un dato varca tutte le porte rappresentate da un layer, quel dato risulta affidabile e certificato.

Come mostrato in Figura 3.9, l'architettura del Data Warehouse è costituita da cinque differenti database:

- **STG** - lo **Staging** gestisce il trattamento di certificazione delle sorgenti dati. Corrisponde all'equivalente dell'ODS;
- **EDW** - i dati nell'**Enterprise Data Warehouse** vengono sottoposti a trasformazioni, normalizzazione e arricchimenti con KPI calcolati a partire da complesse regole di business;
- **EDM** - l'**Enterprise Data Mart** rappresenta uno schema dedicato per la pubblicazione delle informazioni destinate al consumo. Può essere più o meno trasparente rispetto all'EDW;
- **CTL e MAN** - contengono le tabelle standard del Control Layer. Tra queste troviamo tabelle di configurazione e tabelle di log.

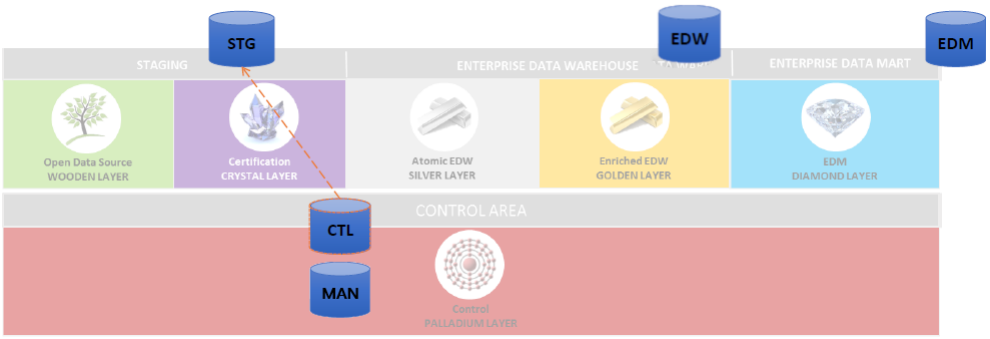


Figura 3.9: Database Schema

Il motivo di questa suddivisione in livelli, al di là dell'ordine logico, è che tipicamente i diritti di accesso a questi schemi sono diversi a seconda del tipo di utenza. Lo strato STG rappresenta il confine con i responsabili dei sistemi gestionali, l'EDW è un'area più tecnica e riservata, l'EDM è aperto a tutti gli utenti finali.

3.4 Migrazione delle Data Entity

Come anticipato in precedenza, le Data Entity sulle quali ci si è concentrati nel progetto di tesi sono **Orders** e **Portfolio**. La migrazione di una singola Data Entity, può essere vista come un sotto-progetto figlio del più generale progetto di migrazione del Data Warehouse. Per ogni Data Entity, infatti, è necessario seguire il medesimo processo di sviluppo che può essere essenzialmente suddiviso in quattro fasi:

1. **Analisi**;
2. **Mockup**;
3. **Sviluppo**;
4. **Test**.

Di seguito vengono descritte le singole fasi del processo di migrazione. Premesse da tenere in considerazione, sono le seguenti:

- per semplicità, in ogni sezione vengono presi in considerazione solo i dettagli più interessanti della fase di volta in volta descritta e vengono considerate contemporaneamente entrambe le Data Entity nonostante, nella realtà, lo sviluppo sia avvenuto sequenzialmente (prima Orders e poi Portfolio) poiché guidato dalla schedula definita nelle fasi iniziali di progetto;
- l'obiettivo del progetto di migrazione non include la rimodellazione del Data Warehouse attualmente in uso quindi, durante lo sviluppo, non è emersa la necessità di ricorrere a particolari tecniche di modellazione concettuali (come ad esempio il DFM). Sono state mantenute le stesse tabelle e relazioni presenti nel vecchio sistema facendo attenzione a non duplicare il dato in presenza di report AS-IS contenenti entità (esempio: dimensioni conformi) già caricate in precedenza nella nuova architettura ma che si riferiscono a tabelle differenti. In questo caso l'idea è quella di riutilizzare le entità già presenti, previo accordo con il Cliente.

3.4.1 Analisi

Durante questa fase si analizzano i singoli report della Data Entity e si produce un documento Excel che ne descrive le caratteristiche. Da adesso in poi, questo documento verrà riferito come **Documento di Analisi**. Si parte da un template Excel predefinito e lo si compila inserendo informazioni riassuntive riguardo a tutti i report della Data Entity. Il Documento di Analisi ha un ruolo chiave poiché viene successivamente utilizzato per guidare le fasi successive della migrazione.

Si parte dal presupposto che tutti i report del sistema di Business Intelligence originario fanno riferimento ad uno specifico universo BO e sono tutti caratterizzati dal fatto di essere tabellari, infatti, nessuno presenta grafici o indicatori. Tutte le informazioni dei report sono raggruppate all'interno di tabelle. Il motivo di questa scelta è legato al tipo di utilizzo che fino ad ora il Cliente ha fatto di essi. Principalmente si è trattato di un uso operativo dove la necessità era quella di visualizzare i dati filtrandoli in vari modi e poter esportare eventualmente i risultati in vari formati (es. Excel). Con il nuovo sistema, uno degli obiettivi è proprio quello aumentare, dove possibile, le capacità espressive dei report aggiungendo indicatori e diagrammi.

Ogni report può essere suddiviso in una o più pagine, ognuna delle quali può contenere una o più tabelle e fare riferimento ad una o più query che alimentano le tabelle recuperando i dati dalle sorgenti Oracle. In alcuni casi, viene data la possibilità all'utente di specificare manualmente dei filtri (prompt) che agiscono direttamente sulle query e modificano il risultato restituito e visualizzato nelle tabelle.

Tra le principali informazioni che vengono raccolte nel Documento di Analisi troviamo:

- **caratteristiche descrittive del report** - nome, path BO, universo BO, numero di pagine AS-IS, numero di pagine TO-BE. Queste ultime due informazioni servono a determinare se nel report originario esistono pagine che, per motivi di ridondanza o per assenza di utilità, non è richiesto di avere anche nel corrispettivo report nel nuovo sistema;

- **elenco delle query** - vengono riportate tutte le query raggruppate per report. Le query verranno poi utilizzate durante le fasi di Sviluppo e di Test per la realizzazione del Data Warehouse e le verifiche di consistenza;

- **elenco dei prompt** - vengono indicati tutti i filtri che l'utente ha la possibilità di applicare ad un report insieme al formato dei valori che essi accettano in input. Questa operazione è importante poiché gli stessi filtri dovranno essere riportati anche nei corrispettivi report nel nuovo sistema;

- **elenco KPI** - si analizzano le singole tabelle dei vari report alla ricerca di colonne che possono svolgere il ruolo di KPI. Esempi tipici sono gli importi, il numero di vendite, ecc;

- **note di similarità tra i report** - un passo importante, che verrà concretizzato nella fase successiva, è quello di decidere quali report possono essere fusi con altri. In molti casi, infatti, il sistema di Business Intelligence originario presenta report ridondanti che devono essere unificati. Molte volte questa operazione non è scontata poiché possono esistere report che presentano informazioni analoghe ma con caratteristiche leggermente diverse per i quali è necessario chiedere feedback agli utenti di business prima di procedere con la fusione.

3.4.2 Mockup

In questa fase, viene eseguita un'analisi più approfondita dei singoli report della Data Entity. Uno degli obiettivi è comprendere quali tra i report presenti possono essere fusi in un unico report e quali devono invece essere mantenuti separati. Alcuni report, infatti, possono presentare informazioni ridondanti che non ha senso mantenere duplicate. Inoltre, poiché tutti i report sorgente sono di tipo tabellare, per i report che presentano KPI complessi è necessario valutare se produrre un mockup che consenta di aumentarne il potere analitico introducendo indicatori e diagrammi. Questa attività può richiedere un eventuale consulto con il Cliente. Per svolgere l'attività di fusione dei report ridondanti, ci si lascia guidare dalle informazioni emerse nella fase precedente e dai risultati delle analisi preliminari eseguite prima dell'avvio del progetto. Per quanto riguarda invece lo sviluppo di report dal potere analitico maggiorato, vengono applicate le best practice previste nell'ambito della Data Visualization [3] per integrare ad essi diagrammi che ne massimizzino l'efficacia. Per i report che non presentano KPI complessi e per i quali non risulta necessario estendere il potere analitico, ci si limita a migrarli sul nuovo stack tecnologico così come sono, cercando di mantenere le nuove versioni il più possibile fedeli a quelle vecchie sia dal punto di vista estetico sia dal punto di vista delle possibilità analitiche.

Complessivamente, la migrazione delle Data Entity Orders e Portfolio coinvolge venti report. I dettagli possono essere visualizzati nella Tabella 3.1. Di questi venti report, quelli che presentano KPI complessi sono due (*"Orders Trend"* e *"Portfolio Analysis"*). I mockup per i due report in questione vengono realizzati utilizzando lo strumento **Power BI**. Per i 18 report restanti vengono realizzati mockup che non aggiungono potere analitico ma si limitano

a rispecchiare quanto attualmente presente su SAP BO. Per questi ultimi viene utilizzato lo strumento **Power BI Report Builder**.

Mockup Orders Trend

Il report *"Orders Trend"* nasce dalla fusione di tre report AS-IS:

- *"Orders In Trend"* - mostra il numero di veicoli ordinati;
- *"Orders Canc Trend"* - mostra il numero di ordini veicolo cancellati;
- *"Net Orders Trend"* - mostra il numero di veicoli ordinati, al netto delle cancellazioni.

In questi tre report, le misure vengono rappresentate in funzione del modello del veicolo, in una certa area geografica e in periodo temporale specifico. I possibili filtri applicabili dall'utente sono sul modello del veicolo, sulla nazione e sul periodo di analisi. Poiché presentano somiglianze e misure correlate, si decide di unificarli in un unico report Power BI che permetta all'utente di visualizzare contemporaneamente o una alla volta le misure di interesse (Orders In, Cancellations, Net Orders). In Figura 3.10 viene proposto il mockup realizzato. I dati in esso contenuti costituiscono un campione selezionato a partire dal totale.

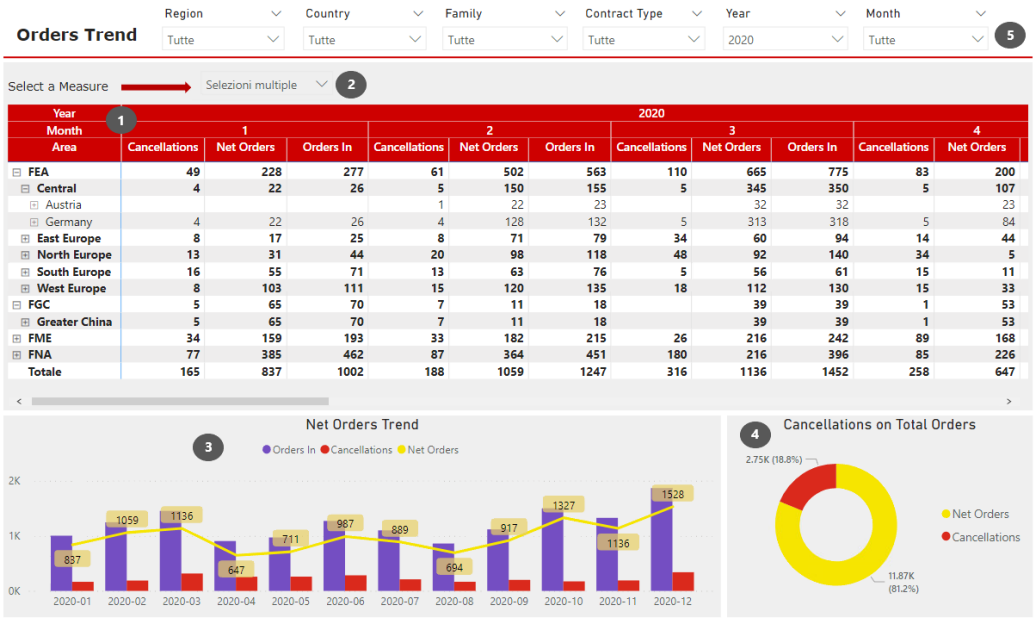


Figura 3.10: Mockup Orders Trend

Seguendo la numerazione indicata in Figura 3.10, le componenti salienti del mockup sono:

1. **Tabella del report** - riporta i dati tabellari del report originale. Viene eseguito il pivoting degli anni e dei mesi e quindi indicati i valori delle tre misure di interesse suddivisi secondo la gerarchia geografica Area-Region-Country sulla quale può essere eseguita l'operazione di drill-down;

2. **Selettore misure** - questo selettore consente di scegliere quali delle tre misure visualizzare in tabella. Il risultato della scelta impatta solo sulla tabella e non anche sui grafici del report;

3. **Trend degli ordini al netto delle cancellazioni** - si tratta di un grafico a linee con istogramma a colonne raggruppate che visualizza sulle colonne gli ordini in ingresso e le cancellazioni mentre, sull'unica linea presente, mostra l'andamento degli ordini al netto delle cancellazioni;

4. **Cancellazioni sul totale degli ordini** - visualizza la porzione di ordini annullati sul totale di ordini registrati;

5. **Barra dei filtri** - nella parte alta del report l'utente ha la possibilità di specificare filtri che agiscono su tutte le componenti grafiche (Visual). Questo apre la strada ad una vasta gamma di analisi delle quali l'utente può usufruire autonomamente. I filtri indicati sono gli stessi che erano presenti nei report originali di SAP BO ai quali è stato aggiunto un filtro sul tipo di contratto.

Una funzione importante offerta da Power BI è l'interazione tra i Visual. Oltre a poter indicare filtri che agiscono su tutte le componenti grafiche contemporaneamente, l'utente ha la possibilità, interagendo con uno dei Visual, di propagare tale interazione anche verso tutti gli altri. Nella Figura 3.11 l'utente seleziona il paese *"Germany"* dalla tabella e il risultato è che i grafici del report vengono filtrati di conseguenza. Questa funzionalità può essere personalizzata al bisogno scegliendo eventualmente di limitare l'interazione solo tra alcuni Visual o nessuno. L'insieme delle funzioni offerte da Power BI lo rendono uno strumento adatto alla realizzazione di report in linea con i principi definiti dal *Visual Information Seeking Mantra* [6] di Ben Shneiderman.

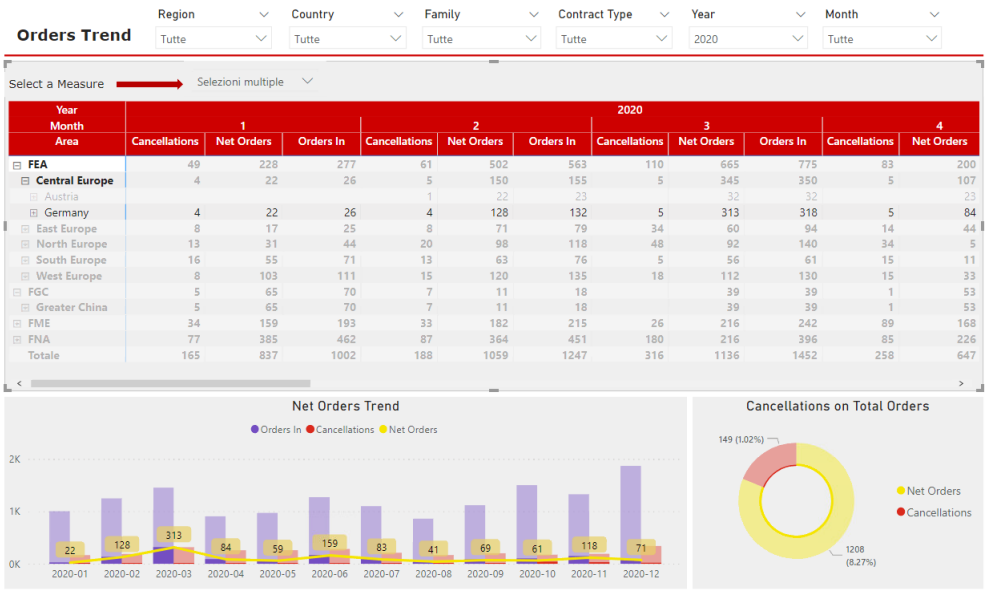


Figura 3.11: Mockup Orders Trend - Interazione tra Visual

Mockup Portfolio Analysis

Il termine Portfolio viene utilizzato nel contesto del Cliente per indicare il Portfolio contratti, ovvero il numero di contratti attivi. Un contratto risulta attivo finché il veicolo viene consegnato al cliente o finché il contratto viene annullato. Il report *"Portfolio Analysis"* contiene l'elenco di tutti i contratti attivi e per ognuno specifica informazioni quali la posizione geografica in cui il contratto è stato sottoscritto, l'anzianità del contratto, dettagli sul veicolo in oggetto, ecc. In Figura 3.12 viene proposto il mockup realizzato. Anche in questo caso, i dati in esso contenuti costituiscono un campione selezionato a partire dal totale. Le componenti salienti del mockup sono:

1. **Mappa dei contratti attivi** - viene mostrata una mappa navigabile dove è possibile visualizzare come si distribuiscono nel globo i contratti sottoscritti e attualmente attivi. La dimensione delle bolle determina la quantità di contratti attivi in quell'area. È possibile eseguire operazioni di drill-down e roll-up per visualizzare informazioni più o meno dettagliate. Passando con il cursore su una delle bolle viene visualizzato un Tooltip che mostra il nome dell'area in questione e il numero puntuale di contratti attivi in quella zona;

2. **Contratti attivi per modello di veicolo** - l'obiettivo è visualizzare dei dati statici (ovvero puntuali in un certo istante di tempo) di un'unica variabile di riferimento (il numero di contratti attivi). Per questo motivo

viene utilizzato un Clustered Bar Chart che distribuisce i contratti attivi tra i vari modelli e consente di effettuare confronti a colpo d'occhio;

3. **Anzianità dei contratti attivi** - l'anzianità di un contratto attivo si stabilisce come il numero di giorni che intercorrono tra la data di inserimento del contratto e la data del report. Se un contratto viene risolto o annullato non viene più considerato nell'elenco dei contratti attivi e dunque non incide più sul grafico;

4. **Pannello di filtro e di dettaglio** - viene predisposto un pannello dedicato all'applicazione dei filtri che, anche in questo caso, rispecchiano quelli già esistenti sul report originale di SAP BO.

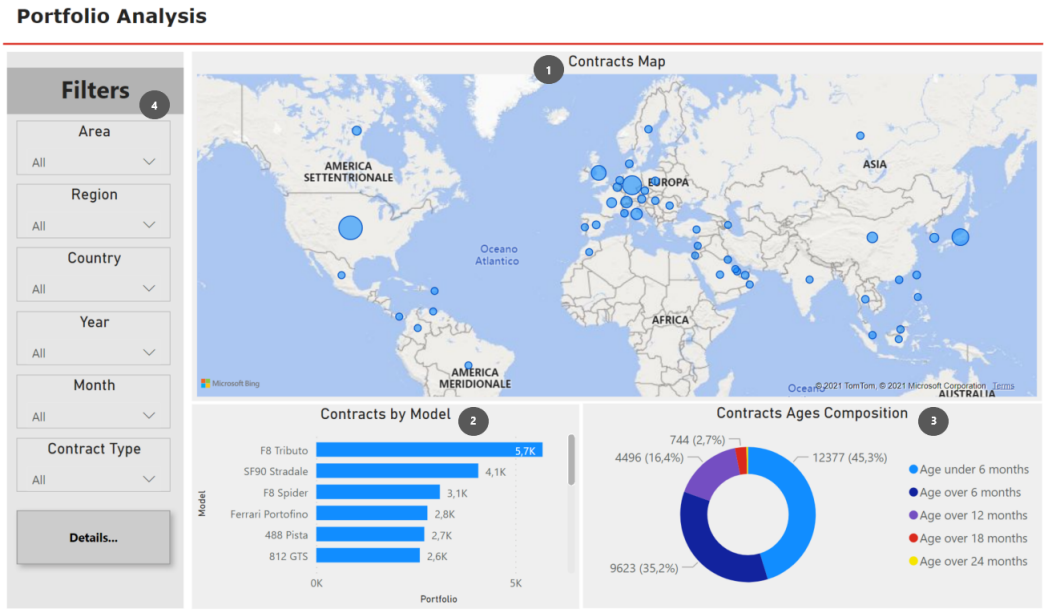
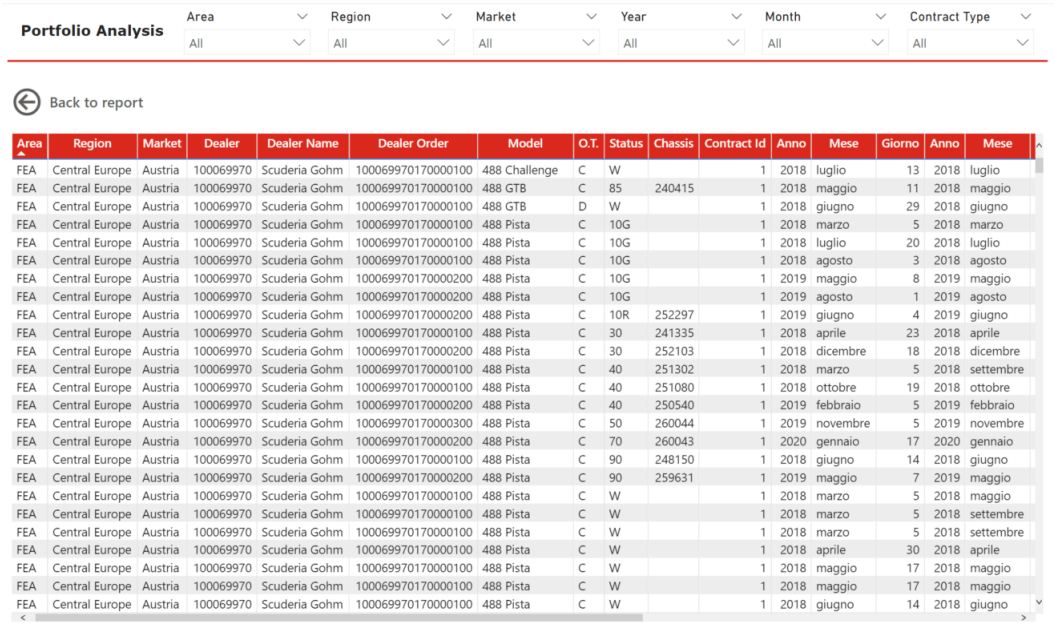


Figura 3.12: Mockup Portfolio Analysis

In questo caso, il report è suddiviso su due pagine Power BI, infatti, se l'utente effettua un click sul pulsante *"Details..."* viene direzionato verso la schermata mostrata in Figura 3.13 dove è possibile visualizzare nel dettaglio la tabella che costituiva il report originario nel vecchio sistema. Da notare, nella parte alta del mockup, la presenza degli stessi filtri della pagina precedente. I filtri vengono sincronizzati in modo che passando da una pagina all'altra rimangano attivi. Un apposito pulsante consente all'utente di tornare alla pagina precedente.



Portfolio Analysis

Area: All | Region: All | Market: All | Year: All | Month: All | Contract Type: All

Back to report

Area	Region	Market	Dealer	Dealer Name	Dealer Order	Model	O.T.	Status	Chassis	Contract Id	Anno	Mese	Giorno	Anno	Mese
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Challenge	C	W		1	2018	luglio	13	2018	luglio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 GTB	C	85	240415	1	2018	maggio	11	2018	maggio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 GTB	D	W		1	2018	giugno	29	2018	giugno
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	10G		1	2018	marzo	5	2018	marzo
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	10G		1	2018	luglio	20	2018	luglio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	10G		1	2018	agosto	3	2018	agosto
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000200	488 Pista	C	10G		1	2019	maggio	8	2019	maggio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000200	488 Pista	C	10G		1	2019	agosto	1	2019	agosto
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000200	488 Pista	C	10R	252297	1	2019	giugno	4	2019	giugno
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	30	241335	1	2018	aprile	23	2018	aprile
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000200	488 Pista	C	30	252103	1	2018	dicembre	18	2018	dicembre
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	40	251302	1	2018	marzo	5	2018	settembre
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	40	251080	1	2018	ottobre	19	2018	ottobre
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000200	488 Pista	C	40	250540	1	2019	febbraio	5	2019	febbraio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000300	488 Pista	C	50	260044	1	2019	novembre	5	2019	novembre
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000200	488 Pista	C	70	260043	1	2020	gennaio	17	2020	gennaio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	90	248150	1	2018	giugno	14	2018	giugno
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000200	488 Pista	C	90	259631	1	2019	maggio	7	2019	maggio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	W		1	2018	marzo	5	2018	maggio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	W		1	2018	marzo	5	2018	settembre
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	W		1	2018	marzo	5	2018	settembre
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	W		1	2018	aprile	30	2018	aprile
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	W		1	2018	maggio	17	2018	maggio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	W		1	2018	maggio	17	2018	maggio
FEA	Central Europe	Austria	100069970	Scuderia Gohm	100069970170000100	488 Pista	C	W		1	2018	giugno	14	2018	giugno

Figura 3.13: Mockup Portfolio Analysis - Tabella di dettaglio

L'insieme delle decisioni prese in questa fase, insieme ai mockup realizzati, vengono inseriti all'interno di un documento che da adesso in poi verrà riferito come **Documento di Mockup**. Quest'ultimo viene successivamente condiviso con il cliente che provvederà ad analizzarlo fornendo feedback ed eventualmente ad approvarlo.

3.4.3 Sviluppo

In questa fase viene effettivamente sviluppata la migrazione delle Data Entity verso la nuova architettura. Le attività che vengono svolte possono essere raccolte in tre gruppi:

1. Processo di ETL;
2. Modello dati;
3. Report finali.

Processo di ETL

Come anticipato precedentemente, il nuovo Data Warehouse è suddiviso in tre livelli attraversando i quali il processo di ETL elabora e trasforma i dati per ottenere informazioni di maggiore qualità. Ogni tabella coinvolta nei report delle Data Entity precedentemente analizzate, deve essere migrata all'interno del nuovo Data Warehouse rispettando la suddivisione in livelli. È importante fare attenzione a non migrare tabelle condivise con altre Data Entity che sono già state migrate in precedenza. Complessivamente, per le Data Entity Orders e Portfolio è richiesta la migrazione di 38 tabelle alcune delle quali sono condivise da entrambe le Data Entity.

Lo sviluppo del processo di ETL prevede inizialmente l'effettiva creazione degli strati del Data Warehouse. Per prima cosa è necessario occuparsi della **creazione dello strato ODS** (riferito come STG) e dell'aggiornamento della **tabella di configurazione ODS** (*"MAN.BO_ConfigTableToLoad"*). Quest'ultima contiene informazioni di configurazione sui caricamenti delle tabelle Oracle, tra cui le query SQL per recuperare i dati. Le tabelle ODS mantengono gli stessi nomi tabella/colonne presenti nelle tabelle sorgenti, vengono distribuite secondo la modalità Round Robin e aggiungono in coda le colonne:

- *"SOURCE_SYSTEM"* - sistema sorgente (LN, CRM, DWH, fornitore esterno, ecc.);
- *"SOURCE_TABLE"* - nome della tabella sorgente;
- *"TS_INS"* - data e ora di caricamento del record.

SOURCE_SYSTEM	SOURCE_TABLE	TS_INS
LN 10.4	BAAN.TTFCMG011840	2020-10-27 23:27:20.000

SOURCE_SYSTEM	SOURCE_TABLE	TS_INS
DWH	DWH_DM.DM_PREOWNED_ORD_COM	2021-11-03 14:59:25

SOURCE_SYSTEM	SOURCE_TABLE	TS_INS
Analitic Company	DF_RV_AGE_data.20211021.csv	2021-10-21 09:38:34.000

Figura 3.14: Esempi di colonne aggiunte in coda alle tabelle ODS

Il passo successivo prevede la **creazione dello strato EDW**. Le trasformazioni che vengono applicate in questo strato possono essere suddivise in due fasi:

- sotto-strato **Atomic EDW** - per ogni tabella caricata in ODS viene creata una tabella DWA all'interno della quale vengono applicate trasformazioni di base (conversioni di tipi, rinomina dei campi, ecc.) e normalizzazione;

- sotto-strato **Enriched EDW** - per ogni tabella DWA viene creata una tabella DWE nella quale vengono applicati arricchimenti semantici (join tra tabelle, ecc.) e business rules.

Le tabelle DWA e DWE vengono distribuite secondo le modalità Hash o Replicated in base alla dimensione della tabella. Poiché le tabelle DWE possono essere frutto di join tra diverse tabelle degli strati precedenti, per ognuna di esse è necessario inserire la lista delle dipendenze all'interno della **tabella di configurazione DWE** (*"MAN.BO_ConfigDWE"*).

L'ultimo passo prevede la **creazione dello strato EDM** all'interno del quale vengono principalmente realizzate delle viste a partire dalle tabelle DWE. È il momento ideale per applicare la rinomina dei campi con nomi di business, l'aggiunta di eventuali chiavi composte da utilizzare per costruire successive relazioni nei cubi, l'applicazione di eventuali filtri. Le viste espongono informazioni dedicate al consumo e verranno successivamente utilizzate come sorgenti per la realizzazione dei modelli dati. Il principale ruolo delle viste è quello di disaccoppiare le query, espresse tramite i report, dalla tabella DWE concreta. In generale, se una tabella DWE è condivisa da più Data Entity, a partire da essa vengono comunque create viste differenti. Tramite le viste abbiamo la possibilità di limitare la visibilità sulla tabella concreta agli utenti che utilizzano i report di una specifica Data Entity. Tramite la tabella concreta abbiamo la possibilità di applicare Business Rules che si riflettono automaticamente su tutte le viste che si appoggiano ad essa.

Una volta realizzata la struttura a livelli del Data Warehouse è possibile procedere con l'effettiva esecuzione del processo di ETL. In precedenza è stata anticipata l'esigenza da parte del Cliente di ridurre al minimo i tempi di attesa che intercorrono tra il caricamento dei dati nella nuova architettura alla loro effettiva disponibilità in Power BI. Questo richiede di poter caricare ed elaborare i dati non appena sono "pronti" sulle sorgenti. Il metodo migliore per fare ciò probabilmente è pianificare dei flussi event-driven. Poiché nel nostro caso, questa strada non risulta praticabile, viene simulato il comportamento event-driven interrogando le sorgenti ciclicamente. In pratica, il sistema verifica la presenza di dati "pronti" eseguendo un polling a intervalli regolari verso le sorgenti. Per implementare questo meccanismo è stato necessario creare una tabella di log sulle sorgenti Oracle che tenesse traccia dei flussi Data Stage eseguiti quotidianamente. Sul nostro sistema è stata poi creata una seconda tabella di log all'interno della quale viene registrato l'andamento di ogni caricamento. La combinazione delle due tabelle di log ci permette di capire quali caricamenti devono essere lanciati. Il meccanismo di polling consiste semplicemente nell'interrogare ciclicamente queste due tabelle. Inoltre, poiché alcune delle trasformazioni che vengono eseguite sul nostro sistema dipendono dal completamento di questi caricamenti e si vorrebbe eseguire queste trasformazioni il prima possibile, sono stati progettati ulteriori meccanismi di polling che, interrogando il nostro log interno, verificano quando è possibile eseguire una trasformazione.
Ogni meccanismo di polling condivide i seguenti aspetti:

- viene eseguito da un trigger una volta al giorno;
- verifica la presenza di tabelle processabili interrogando una o più tabelle di log (polling);
- poiché l'operazione di polling è costosa, prima di ripeterla, se non è trascorso un certo intervallo di tempo dall'inizio del polling in corso, il meccanismo deve attendere un numero di secondi pari al tempo rimanente per raggiungere quell'intervallo di tempo. In questo modo, il polling può essere eseguito ad intervalli regolari;
- il ciclo di polling termina al raggiungimento di un tempo limite, anche se non sono state recuperate tutte le tabelle richieste;
- le pipeline vengono realizzate con Azure Data Factory mentre Azure Synapse Analytics viene utilizzato per lo storage.

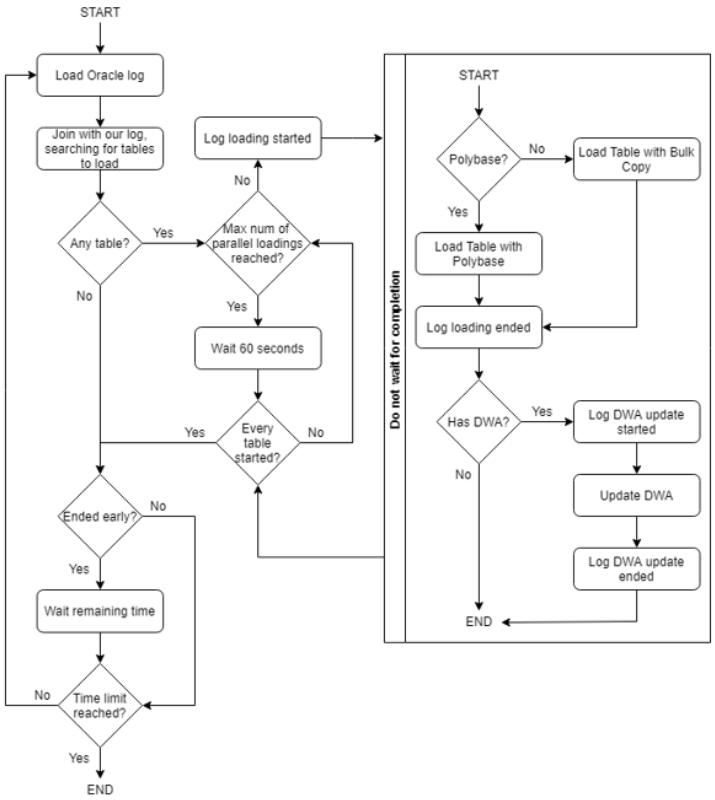


Figura 3.15: Caricamento dei dati e update delle tabelle DWA

In Figura 3.15 viene descritto il funzionamento del meccanismo di polling per il caricamento dei dati dalle sorgenti. Nella parte più a sinistra del flowchart viene mostrato il ciclo di polling. Al suo interno, vengono combinate le tabelle di log (quella presente su Oracle e quella interna alla nostra architettura) e si verifica se sono presenti tabelle che possono essere caricate. Se almeno una è presente, il sistema esegue un determinato processo (visibile al centro e a destra del flowchart) al termine del quale vengono eseguiti i due controlli temporali accennati in precedenza.

Il resto del processo è progettato in modo da consentire l'avvio di un numero limitato di caricamenti in parallelo. Questa scelta è motivata dal fatto che si vuole evitare di appesantire troppo la coda di operazioni da far svolgere Synapse essendo quest'ultimo utilizzato anche per gestire query provenienti da altri progetti. La pipeline non aspetta che i caricamenti vengano completati, il suo unico scopo è lanciare più caricamenti possibili, nel minor tempo possibile. Per fare ciò è necessario risalire a quanti caricamenti sono attualmente in esecuzione interrogando la tabella di log interna. In essa, infatti, i caricamenti vengono registrati subito prima dell'avvio e immediatamente dopo il completamento.

Infine, per il caricamento effettivo delle tabelle, viene utilizzata la classica *"Copy Data Activity"* di Azure Data Factory sfruttando la tecnologia Polybase per le tabelle di dimensioni elevate. Subito dopo il caricamento in ODS, dove possibile, i dati vengono spostati direttamente nel layer DWA.

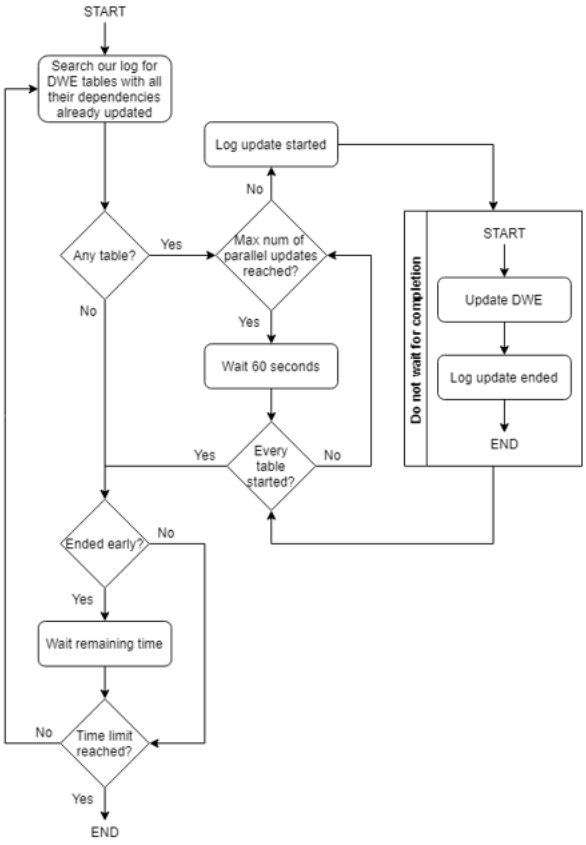


Figura 3.16: Update delle tabelle DWE

In Figura 3.16 viene descritto il funzionamento del meccanismo di polling per effettuare l'aggiornamento dello strato DWE. Il comportamento risulta essere molto simile al caso precedente, quello che cambia è l'operazione di verifica della presenza di nuove tabelle da aggiornare che viene eseguita considerando solo la tabella di log interna alla nostra architettura. I record in essa contenuti, combinati con le dipendenze indicate nella tabella di configurazione DWE, permettono di capire quando le tabelle di questo strato sono processabili dal nostro sistema.

Le pipeline vengono messe in esecuzione nel momento dell'attivazione di un trigger periodico che è stato impostato per scattare ogni giorno a mezzanotte. Di default viene impostato un intervallo di polling di 5 minuti e il tempo limite per il completamento viene configurato per scadere alle 8:30:00.

Modello dati

Secondo le linee guida definite nelle fasi iniziali di progetto, è previsto che per ogni Data Entity venga realizzato un modello dati a sé stante in modo tale da avere modelli più piccoli e facilmente gestibili rispetto ad adottare un unico modello monolitico. Alcune delle Data Entity possono utilizzare tabelle comuni e questo porta inevitabilmente alla presenza di ridondanza tra i vari modelli che vengono creati. Poiché l'architettura adottata separa il Data Warehouse distribuito nel SQL Pool di Synapse dai modelli realizzati in Analysis Services, non sussistono problemi legati all'aggiornamento delle tabelle da parte del processo di ETL: quest'ultimo aggiorna le tabelle presenti sul Data Warehouse e automaticamente i modelli, che sono realizzati a partire da esse, disporranno dei nuovi dati.

Come anticipato in precedenza, i modelli si appoggiano alle viste presenti nello strato EDM del Data Warehouse. Per ogni modello viene realizzato un differente Analysis Services Tabular Project in Visual Studio all'interno del quale si configura come sorgente dati il SQL Pool di Synapse Analytics. Da quest'ultimo vengono caricate le viste EDM che poi vengono messe in relazione in maniera analoga alle operazioni di join presenti nelle query sorgente. Una volta realizzata la struttura del modello, si procede con l'assegnazione di un alias ad ogni tabella e con l'eventuale aggiunta di colonne calcolate e misure. Ogni progetto Visual Studio viene collegato ad un unico repository Azure DevOps per il versioning dei modelli.

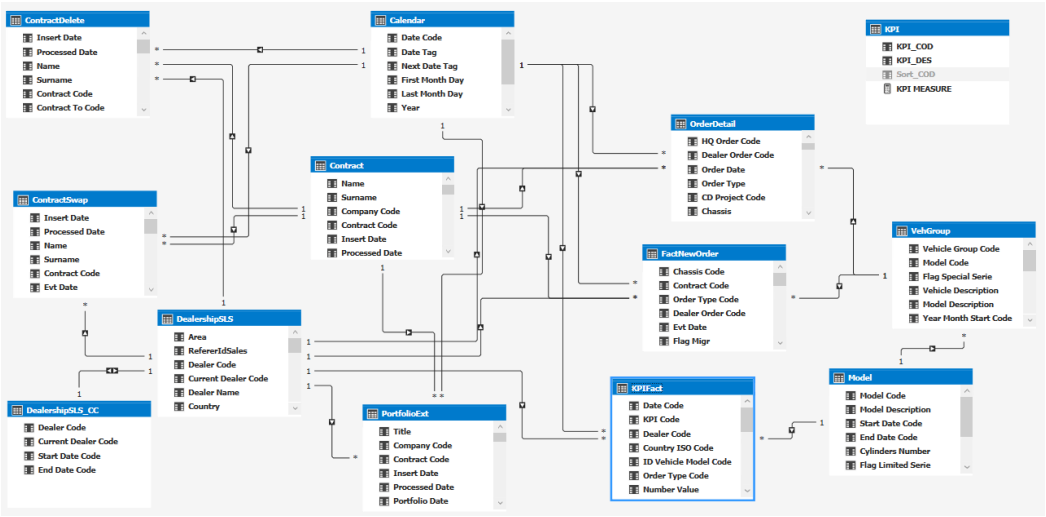


Figura 3.17: Data Entity Orders - Modello Dati

A scopo esemplificativo, in Figura 3.17 viene riportato il modello realizzato per la Data Entity Orders. Il modello presenta diversi fatti che condividono

alcune dimensioni. La tabella *"KPIFact"* è il fatto sulla base del quale è stato realizzato il report il cui mockup è raffigurato in Figura 3.10. Si tratta di una tabella contenente più di un milione di righe che al suo interno include KPI provenienti da diverse Business Area dell'azienda Cliente, infatti, viene condivisa da più Data Entity (inclusa Porfolio) e quindi compare in più modelli. Ogni riga della tabella corrisponde ad un evento. I campi *"KPI Code"* e *"Number Value"* consentono rispettivamente di distinguere il KPI di riferimento e il corrispettivo valore. Nel caso specifico della Data Entity Orders, i KPI di interesse sono quelli con codice 4 e 5 che identificano rispettivamente specifiche istanze di ordini in ingresso e di cancellazioni. In Figura 3.18 viene mostrato un dettaglio su una porzione del contenuto della tabella. Nella parte alta dell'immagine viene messa in evidenza la formula DAX per creare la misura che calcola il numero di ordini in ingresso. In maniera analoga, viene realizzata la misura che calcola il numero di cancellazioni. A partire da queste possono poi essere realizzare ulteriori misure che saranno accessibili da Power BI per la realizzazione dei report.

[KPI Code] f_x Orders In:= CALCULATE(SUM('KPIFact'[Number Value]), 'KPIFact'[KPI Code]=4)

	KPI Code	Dealer Code	Country ISO...	ID Vehicle...	Order Type ...	Number Value	KPI I...	Country	Region	Area	ContractType	Measure	Date...
1	4	100006280	US	F142 DCT	C	1		U.S.A.	USA	FNA	Customer	Ordini In	201307
2	4	100006185	US	F142 DCT	C	1		U.S.A.	USA	FNA	Customer	Ordini In	201505
3	4	100006370	US	F142 DCT	C	1		U.S.A.	USA	FNA	Customer	Ordini In	201502
4	4	100006145	US	F142 DCT	C	1		U.S.A.	USA	FNA	Customer	Ordini In	201201
5	4	100006335	US	F142 DCT	C	1		U.S.A.	USA	FNA	Customer	Ordini In	201303
6	4	100006500	US	F142 DCT	C	1		U.S.A.	USA	FNA	Customer	Ordini In	201303
7	4	100006279	US	F142 DCT	C	1		U.S.A.	USA	FNA	Customer	Ordini In	201301

Orders In: 140576 | Orders Canc: 22644 | Net Orders: 117932 | % Orders Canc: 16.1080127475529

Figura 3.18: Data Entity Orders - Contenuto tabella KPIFact

Nel modello è possibile notare che la tabella chiamata *"KPI"* è priva di relazioni. Si tratta di una tabella inserita manualmente che permette di implementare lo Slicer per il filtraggio dei dati sulla base della misura scelta (Orders In, Cancellations, Net Orders), mostrato nel mockup in Figura 3.10.

[KPI_COD] f_x KPI MEASURE:= SWITCH(FIRSTNONBLANK('KPI'[KPI_COD] ,TRUE()), "Orders In", KPIFact[Orders In], "Cancellations" , KPIFact[Orders Canc], "Net Orders", KPIFact[Net Orders] , 0)

	KPI_COD	KPI_DES	Sort_COD	Add Column
1	Cancellations	Cancellations	2	
2	Net Orders	Net Orders	3	
3	Orders In	Orders In	1	

KPI MEASURE: 22644

Figura 3.19: Data Entity Orders - Contenuto tabella KPI

La funzionalità di filtraggio è resa possibile dalla combinazione del campo *"KPI_COD"* e della misura *"KPI MEASURE"* mostrati in Figura 3.19. Il primo viene utilizzato nello Slicer per consentire all'utente di scegliere la misura di interesse, la seconda restituisce dinamicamente i valori associati alla misura scelta che potranno essere visualizzati in tabella.

Una volta completato, il modello viene pubblicato come dataset all'interno di un repository su Power BI Service. In questo modo viene reso accessibile tramite Power BI Desktop e Power BI Report Builder ed è quindi possibile procedere con la realizzazione dei report finali.

Sviluppo report finali

Per quanto riguardo lo sviluppo dei report con Power BI, una volta collegati al Service e selezionato il modello dati di interesse come sorgente, è possibile procedere con la realizzazione delle dashboard in linea con i mockup precedentemente descritti e indicati all'interno del Documento di Mockup. In questo caso, i dati che popoleranno i Visual inseriti nella dashboard non saranno dati campione ma reali.

Allo stesso modo, utilizzando lo strumento Power BI Report Builder, è possibile collegarsi al Service e selezionare il modello dati di interesse come sorgente. A partire da esso possono quindi essere realizzati dataset che conterranno i dati per popolare le tabelle dei report. A scopo esemplificativo, in Figura 3.20 viene riportata la schermata di progettazione di uno dei 18 report da realizzare con Power BI Report Builder.

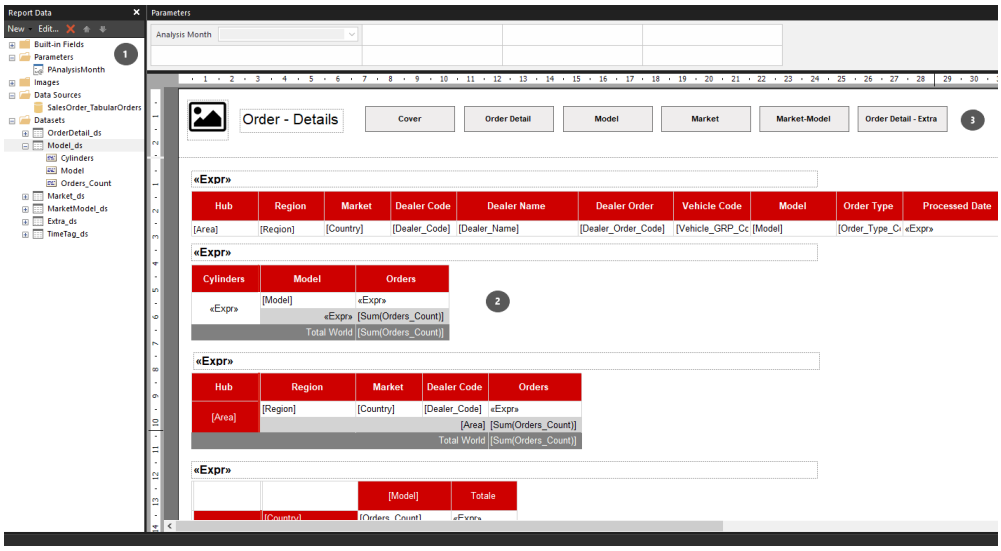


Figura 3.20: Orders - Schermata di progettazione di un paginated report

Seguendo la numerazione proposta in Figura 3.20, gli elementi di interesse sono i seguenti:

1. **Report Data** - vengono specificati Data Sources, Datasets e Parameters. Viene realizzato un Dataset per ogni tabella contenuta nel report e uno per ogni parametro. Infatti, è possibile notare la presenza del Dataset *"TimeTag_ds"* la cui query recupera i dati che verranno utilizzati per popolare il parametro *"PAnalysisMonth"* ovvero quello utilizzato per consentire all'utente di filtrare i dati su base temporale;

2. **Tabelle del report** - vengono progettate le tabelle che conterranno i dati restituiti dai Dataset. Le tabelle presentano la stessa forma assunta nei report originali della vecchia architettura. Sono organizzate per Row Groups e Column Groups sulla base dei quali vengono aggregate le misure. Utilizzando un linguaggio interno allo strumento, è possibile specificare espressioni (*"Expr"*) per visualizzare il dato in forma elaborata;

3. **Bookmarks** - nell'intestazione vengono aggiunti dei pulsanti che consentono di saltare da una tabella all'altra quando il report viene eseguito. Bisogna considerare, infatti, che nonostante nella schermata di progettazione le tabelle risultino relativamente compatte, con l'esecuzione del report verranno popolate da migliaia di righe rendendo difficile lo spostamento da parte dell'utente.

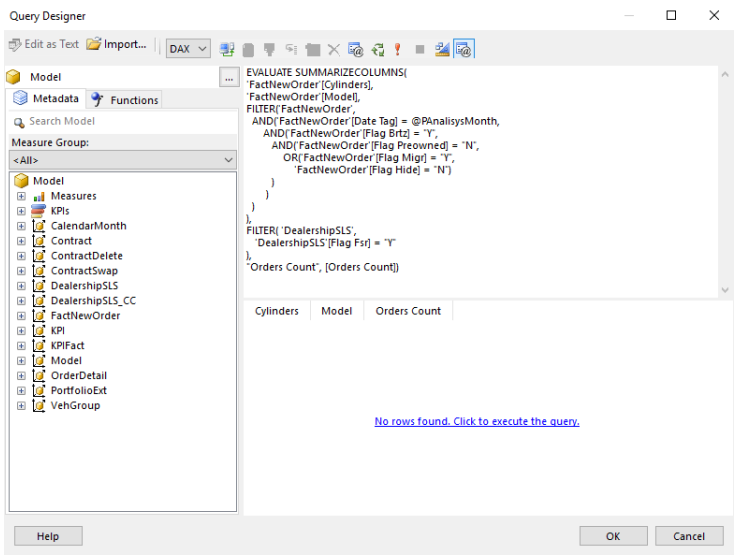


Figura 3.21: Orders - Query DAX del Dataset *"Model_ds"*

Come anticipato in precedenza, ogni Dataset rappresenta il risultato di una query che viene eseguita sulla sorgente dati. Poiché quest'ultima è costituita da un modello Analysis Services, sono supportati i linguaggi DAX e MDX per lo sviluppo della query. Come esempio, in Figura 3.21 viene proposta la query DAX utilizzata per realizzare il Dataset *"Model_ds"*. La funzione *"SUMMARIZECOLUMNS"* restituisce una tabella contenente le colonne e le misure specificate come argomento mentre la funzione *"FILTER"* applica un filtro alla tabella risultante.

Una volta realizzati, i report vengono pubblicati sullo stesso repository di Power BI Service nel quale si trova il modello dati. L'utente finale potrà direttamente utilizzare i report attraverso un browser.

3.4.4 Test

Una volta realizzati tutti i report finali di una Data Entity, viene eseguita una prima sessione di test interni rivolta alla verifica della qualità dei dati (**Data Quality**). L'obiettivo è verificare che i dati visualizzati nei report prodotti corrispondano a quelli che sono presenti sui report originali. Può capitare che i dati non vengano visualizzati nello stesso modo o che alcune misure non corrispondano. In questi casi, è necessario effettuare un'analisi mirata per individuare il problema. Questa analisi viene condotta secondo un approccio definito "a imbuto" poiché inizialmente verifica la correttezza dei report sviluppati ad un livello di dettaglio generale per poi procedere eventualmente concentrandosi su casi sempre più specifici fino ad isolare i record che causano il problema. Fintanto che l'analisi si trova ad un livello di dettaglio generale, si ricorre all'utilizzo dell'SQL per verificare che i dati presenti sul Data Warehouse distribuito su Synapse corrispondano a quelli presenti sulle sorgenti dati Oracle. A mano a mano che si scende nel dettaglio, può essere richiesto di affiancare alle analisi SQL, analisi manuali su esportazioni in Excel dei report che presentano problemi. Le analisi manuali vengono condotte sia utilizzando gli strumenti messi a disposizione da Excel sia attraverso script Python di carattere esplorativo.

La migrazione di una Data Entity si conclude con l'esecuzione di una sessione di **User Acceptance Test (UAT)** durante la quale i report realizzati vengono condivisi con i referenti dell'azienda Cliente che ne verificano la correttezza confrontandoli con quanto descritto nel Documento di Mockup precedentemente validato e approvato. Nel caso in cui i test abbiano esito positivo, il Cliente accetta i risultati ottenuti. In caso contrario, fornisce feedback e suggerisce modifiche che vengono prese in carico il prima possibile.

Tutte le attività di migrazione di una Data Entity vengono sviluppate in un ambiente dedicato allo sviluppo. In tutto esistono tre ambienti, ognuno dei

quali possiede differenti requisiti, motivo per cui è necessario creare un set di risorse Azure dedicato per ciascuno di essi. Nello specifico, gli ambienti sono:

- **Sviluppo**;
- **Test**;
- **Produzione**.

Una volta superati gli UAT, tutto ciò che è stato realizzato in ambiente di sviluppo, viene portato in ambiente di test dove vengono ripetuti gli stessi test di Data Quality effettuati in ambiente di sviluppo. In questo caso, lo scopo è quello di individuare eventuali problemi che possono emergere nell'impiegare gli strumenti dell'architettura in un ambiente mai utilizzato e non contaminato. Completate le verifiche in ambiente di test, si può procedere a spostare quanto realizzato in ambiente di produzione. Il contenuto di questo ambiente costituirà effettivamente quello che verrà consegnato al cliente come parte della soluzione finale.

Capitolo 4

Conclusioni e sviluppi futuri

L'attività di progetto, che si è concentrata sulla migrazione delle Data Entity Orders e Porfolio, è stata portata a termine con successo. L'implementazione ha seguito le classiche fasi di un progetto di BI applicate nel mondo Cloud dell'ecosistema Microsoft Azure, dallo sviluppo del processo di ETL alla realizzazione dei report finali interrogabili dall'utente. Il nuovo sistema consentirà all'azienda Cliente di prendere decisioni di business più consapevoli grazie alla disponibilità di un ventaglio analitico più ampio e alla possibilità di effettuare analisi più mirate. La nuova Data Platform aprirà la strada ad ulteriori progetti che potranno contare su un'architettura dati moderna, flessibile, scalabile, affidabile e sicura.

Per gli sviluppi futuri è previsto di applicare anche alle restanti Data Entity lo stesso processo che è stato utilizzato con quelle oggetto di tesi. Secondo le stime, il progetto di migrazione dovrà essere completato integralmente entro la fine del 2022. In tale data, verrà dismesso il vecchio gestionale Infor LN per lasciare spazio ad un nuovo gestionale SAP. Quando quest'ultimo entrerà in funzione, verrà dismesso anche il vecchio Data Warehouse Oracle e sostituito con quello nuovo basato su stack Microsoft Azure. I vecchi dati verranno utilizzati come dati storici e potranno essere oggetto di analisi future rivolte al supporto delle decisioni di business.

Ringraziamenti

Il ringraziamento più grande va ai miei genitori, Franca e Sauro, che mi hanno sostenuto per tutto il percorso di studi e che in questi mesi mi sono stati particolarmente vicini, supportandomi emotivamente per la realizzazione di questa tesi.

Grazie a mio fratello, Alessandro, le cui ambizioni mi ricordano giorno dopo giorno i motivi per i quali sto affrontando tutto questo e a mia cognata, Mirela, che crede in me e in quello che faccio al punto da offrirmi costantemente nuove idee e spunti per crescere.

Grazie agli amici di una vita, fedeli compagni di avventure, che sono stati un'ancora di salvezza lungo questo percorso poiché mi hanno consentito di staccare la spina al momento del bisogno e di pensare anche ad altro al di fuori dello studio.

Grazie ai miei compagni di corso che hanno condiviso con me questa esperienza nel bene e nel male. Abbiamo affrontato insieme tanti ostacoli e il condividere con voi le difficoltà mi ha permesso di superarle divertendomi e rimanendo sano di mente.

Un ringraziamento speciale è rivolto ai miei tutor Prof. Stefano Rizzi e Daniele Ligorio insieme a tutti i membri del team di IConsulting, cito nello specifico Giacomo Marchi. Mi avete fornito il supporto necessario per la realizzazione di questa tesi guidandomi in tutti i passi del tirocinio. L'avete reso un'esperienza estremamente formativa e utile per il mio futuro professionale.

Questa tesi chiude un cammino durato quasi sei anni. Sento di essere profondamente cambiato rispetto agli inizi. Nonostante mi venga la pelle d'oca a ripensare a tutti gli esami e le difficoltà superate, sono pronto ad affrontare le sfide del nuovo percorso intrapreso. Di quello appena concluso mi porto a casa una maggiore fiducia nelle mia capacità e tanti ricordi che rimarranno indelebili. È stata dura ma ce l'abbiamo fatta!

Riferimenti

Bibliografia

[1] Matteo Golfarelli, Stefano Rizzi, *Data warehouse: Teoria e pratica della progettazione*, McGraw-Hill, 2006.

[2] Barry Devlin, *Data warehouse: from architecture to implementation*, Addison-Wesley Longman, 1997.

[3] Andrew Abela, *Advanced Presentations by Design*, Pfeiffer, 2013.

[4] Matteo Golfarelli, Stefano Rizzi, *A model-driven approach to automate data visualization in big data analytics*, Information Visualization, 2020.

[5] Madhu Sudhan S, Chandra J, *IBA Graph Selector Algorithm for Big Data Visualization using Defence Dataset*, IJSER, 2013.

[6] Ben Shneiderman, *The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations*, University of Maryland, 1996.

[7] Brock Craft, Paul Cairns, *Beyond Guidelines: What Can We Learn from the Visual Information Seeking Mantra?*, IEEE, 2005.

Sitografia

[8] "*ETL vs ELT: Key Differences Everyone Must Know*", Altexsoft — `https://www.altexsoft.com/blog/etl-vs-elt`

[9] "*Modern Business Intelligence - The path to Big Data Analytics*", Deloitte — `https://www2.deloitte.com/content/dam/Deloitte/tr/Documents/deloitte-analytics/Modern%20Business%20Intelligence.pdf`

[10] "*Azure Fundamentals*", Microsoft Docs — `https://docs.microsoft.com/en-us/learn/certifications/azure-fundamentals`

[11] "*How does Azyre Cloud work*", Alan George Meile — https://medium.com/@alan.meile/how-does-azure-cloud-work-7479b6f12979

[12] "*Azure Data Factory Docs*", Microsoft Docs — https://docs.microsoft.com/it-it/azure/data-factory/

[13] "*A Quick Guide to Azure Data Factory*", Ruwan Sri Wickaramarathna — https://medium.com/@ruwansriw/about-azure-data-factory-604eb1028702

[14] "*Introduction to Azure Data Lake Storage Gen2*", Microsoft Docs — https://docs.microsoft.com/en-us/azure/storage/blobs/data-lake-storage-introduction

[15] "*Introduction to Azure Synapse Analytics*", Eleonora Fontana — https://medium.com/codex/introduction-to-azure-synapse-analytics-ff317e782f7b

[16] "*Dedicated SQL pools in Azure Synapse Analytics*", Eleonora Fontana — https://medium.com/codex/dedicated-sql-pools-in-azure-synapse-analytics-d3ac7c394a95

[17] "*DAX overview*", Microsoft Docs— https://docs.microsoft.com/it-it/dax/dax-overview

[18] "*What is Power BI?*", Microsoft Docs — https://docs.microsoft.com/it-it/power-bi/fundamentals/power-bi-overview