POLITECNICO DI TORINO

Collegio di Ingegneria Informatica, del Cinema e Meccatronica

Corso di Laurea Magistrale
in Ingegneria Informatica (Computer Engineering)

Tesi di Laurea Magistrale

La Business Intelligence in Rai Pubblicità

Studio del framework di ETL e realizzazione del data model Qlik nell'ambito di un caso reale

Relatore
prof.ssa Elena Baralis

Tutor Aziendale
Mauro Sandri

Candidato
Enrico Manenti

Aprile 2018

Alla mia famiglia.

PREFAZIONE

La tesi esposta nelle prossime pagine rappresenta un lavoro svolto in ambito aziendale. Pertanto, non essendo un'opera di studio puramente teorica e di ricerca sperimentale, sarà raccontata con carattere descrittivo cercando di spiegare nel dettaglio non solo le varie fasi di sviluppo ma anche le complicazioni, le incongruenze e le esigenze che si possono riscontrare in ambiente lavorativo.

L'attività, avviata con un tirocinio curriculare nel mese di giugno 2017, si è protratta fino a marzo 2018 ed è stata svolta presso Rai Pubblicità S.p.A. Si è operato insieme al collega Mario Amato ma agendo su rami paralleli, motivo per cui il capitolo 8, maggiormente di sua competenza, è raccontato con un minor numero di dettagli. L'intero lavoro è stato supervisionato da Mauro Sandri, tutor aziendale, che ha guidato le varie fasi del progetto. La professoressa Elena Baralis, relatrice, è stata invece la coordinatrice dal punto di vista didattico.

Dopo un'introduzione sull'ambito di applicazione, si aprirà una parentesi sui software utilizzati e sull'architettura corredata, esaminandone le peculiarità. Prima di approfondire il racconto del progetto sviluppato, verrà trattato il framework di caricamento dati aziendale, tratteggiandone le caratteristiche principali. Quanto implementato verrà esposto nei capitoli successivi. Nello specifico si tratta di un'applicazione per l'analisi degli investimenti della clientela di cui verranno raccontate tutte le fasi di sviluppo: il reperimento dati, la creazione degli oggetti di database necessari, la costruzione del data model, l'organizzazione del front-end. Infine, verranno descritti i passaggi che hanno condotto all'integrazione di R, tramite cui si è dotato il sistema software aziendale di un nuovo potente tool di calcolo.

Per questioni di privacy, i dati esposti rappresentano una maschera dei dati effettivamente trattati. Questa manipolazione è stata necessaria in quanto non è possibile esporre in maniera pubblica dati sensibili.

INDICE

INTRODUZIONE

In ambito aziendale ci si trova spesso a dover prendere delle decisioni. Le sorti del business sono influenzate fortemente oltre che dalle idee, dalla risolutezza e dalla brillantezza di chi compie le scelte, anche dai mezzi che aiutano ad assumere le corrette valutazioni. La strumentazione utilizzata, infatti, influenza la qualità delle scelte intraprese. Inoltre, spesso, bisogna tenere conto dei tempi ristretti che intercorrono tra il momento di riflessione e quello di azione.

Compiere decisioni può risultare complicato specialmente quando c'è una considerevole mole di dati alla base. Il principale problema da affrontare diventa, quindi, quello di estrapolare le informazioni a partire da questi dati. Sebbene dato e informazione siano due termini spesso utilizzati come sinonimi, hanno un significato differente. Un dato è un'entità atomica che registra l'accadimento di qualcosa. La parola dato deriva, infatti, dal latino "datum" e significa "fatto". Un dato è sostanzialmente un simbolo grezzo che deve essere interpretato per fornire una conoscenza. L'informazione è, così, il risultato dell'elaborazione di uno o più dati. Il dato, quindi, prende valore soltanto inserito in un contesto, elaborato per uno scopo. Come dice Drucker (1998) "la differenza tra dati e informazioni consiste nel fatto essenziale che queste ultime sono dati organizzati per un particolare scopo"[1]. La conoscenza dei meccanismi e delle regole che si genera dal flusso delle informazioni è, invece, qualcosa di ancora più complesso da definire. Per Blair (2002) "la conoscenza si origina attraverso il confronto, le implicazioni, le connessioni e la comunicazione di dati tra persone, gruppi e organizzazioni"[1]. In generale, l'obiettivo che si vuole raggiungere è che il passaggio da dato a informazione, da fatto a conoscenza, sia il più puntuale e consistente possibile e con tempistiche near-real time. Diventa così fondamentale non avere a che fare con la staticità di un report cartaceo ma essere supportati da un tool dinamico, agile e funzionale ma allo stesso tempo anche semplice e facile da usare.

Nel contesto Rai Pubblicità la sfida intrapresa nell'ultimo anno è stata proprio quella di dissuadere i business users dall'utilizzare i vecchi modelli stampati contenenti un numero fisso e time-dependent di dati indicizzati e muoverli verso un approccio dinamico: dinamicità intesa sia nell'uso, consentendo una consultazione delle informazioni a tutto tondo, saltando con semplicità da una informazione all'altra, sia per ciò che riguarda la variabile temporale, con dati sempre aggiornati.

Il lavoro di questa tesi si innesta proprio in questo contesto e mira a fornire un'app di supporto alle decisioni aziendali rispondendo in maniera ampia alle richieste utente.

[1] De Toni A. F., Fornasier A., *La guida del Sole 24 Ore al Knowledge Management*, Gruppo 24 Ore, Milano, 2012

Dare supporto alle decisioni aziendali è proprio il target della Business Intelligence (BI).

Cos'è la Business Intelligence?

Per Business Intelligence si intende il processo di trasformazione di dati e informazioni in conoscenza finalizzata ad adottare strategie vincenti nel settore di riferimento in cui si opera, ottenendo così vantaggi competitivi. La BI è, quindi, l'insieme di tecnologie e processi che consentono alle persone di un'organizzazione, con background differenti, di accedere ai dati, interagire con essi e addentrarsi nell'analisi per gestire il business, migliorando le performance e operando in maniera efficace. Non solo: è anche strumento di scoperta di nuove opportunità verso cui spingersi.

Partendo dai dati, il percorso messo in atto dalla BI consiste di tre momenti: inserire questi dati in un contesto ed elaborarli per ottenere informazioni; dare un significato a queste informazioni per ricavarne conoscenza; unire la conoscenza raggiunta al sapere proprio, all'intuizione, alla perspicacia e alla creatività per avere un giudizio globale sulle cose. Ciò porta a una maggiore padronanza del business, più coscienza sulle decisioni e un migliore controllo sulle azioni da intraprendere.

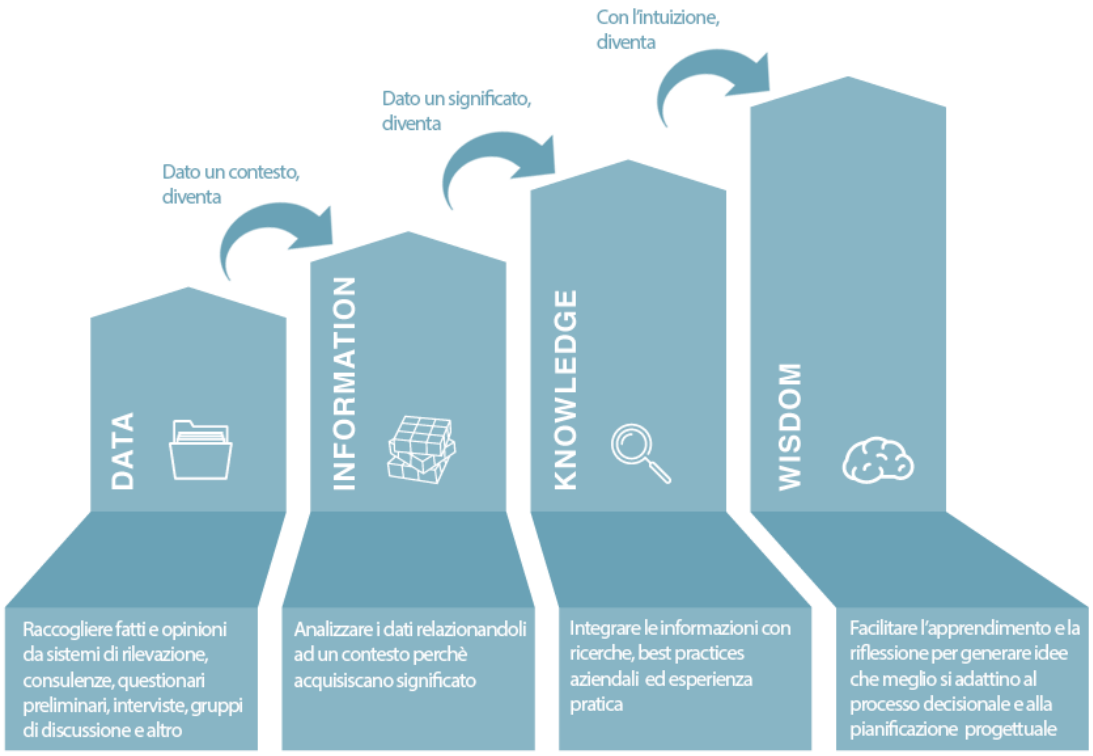

Figura I.1: Business Intelligence – Dai dati alla conoscenza applicabile

Il risultato che si prefigge la business intelligence è, quindi, quello di consentire a chi opera in un'azienda, a qualunque livello, di avere un'introspezione sul settore di cui si interessa, una finestra dinamica sul proprio business. Al giorno d'oggi, in cui prendere la

decisione vincente col giusto tempismo assume un'importanza capitale, la business intelligence gioca un ruolo fondamentale nelle operazioni quotidiane di una compagnia. Se usata in larga misura, consente, appunto, di apportare un valore aggiunto migliorando le performance e aumentando i guadagni. La BI, infatti, aiuta le aziende a scoprire e valutare nuovi affari e opportunità esplorando i dati e testando le teorie ragionate.

Obiettivo della business intelligence è, in sintesi, rendere le informazioni facilmente accessibili e quindi valutabili. È importante così che ciò che viene offerto e visualizzato sia chiaro, comprensibile e il più intuitivo possibile all'utilizzatore finale, non allo sviluppatore che conosce i meccanismi. Per cui i tools e le applicazioni di BI devono essere facili e semplici da usare, con brevi tempi di risposta alle richieste utente in modo da fornire un'user interaction fluida e naturale. In poche parole, bisogna garantire semplicità, velocità e soddisfazione nell'utilizzo. Per realizzare l'obiettivo di rendere le informazioni facilmente fruibili, gli strumenti di BI consentono di organizzare ciò che si vuole esporre in tabelle, grafici, figure, report, presentazioni e dashboards in modo da riassumere le informazioni e renderle maggiormente d'impatto. Una componente fondamentale della BI è quindi rappresentata dalla data visualization, cioè l'insieme delle tecniche di raffigurazione dei dati utili a interpretare in maniera semplice e intuitiva l'andamento aziendale. Ovviamente, fondamentale è anche la parte di data modeling, cioè di predisposizione della nuvola dati d'interesse. Serve, quindi, determinare quali dati esporre e che collegamenti ci sono tra di essi. Se questa fase non viene eseguita in maniera appropriata, le informazioni presentate potrebbero infatti risultare incoerenti o, peggio, presentare delle falle.

Riassumendo, la business intelligence, grazie a una fruizione più immediata e intuitiva delle informazioni aziendali, consente, quindi, di analizzare in maniera più consona i dati, aumentare il controllo di tutte le fasi del lavoro, permettere la scoperta di nuovi ambiti di intervento e opportunità di mercato, migliorare la fase di ragionamento strategico e ottimizzare i processi decisionali guadagnando un vantaggio determinante verso i competitors.

CAPITOLO 1
SOFTWARE UTILIZZATI

Per avere chiarezza del flusso di elaborazione del progetto sviluppato è bene fornire una panoramica degli strumenti utilizzati. In questo capitolo si analizzano allora i software adoperati, fornendo le principali caratteristiche, le potenzialità e spiegando in dettaglio le peculiarità e le qualità per cui sono stati scelti. Vengono inoltre mostrati gli aspetti e gli attributi più rilevanti in modo da avere una visione d'insieme sulle possibilità offerte. In questo modo risulteranno più chiare le scelte implementative adottate e i processi realizzati che saranno discussi nei successivi capitoli.

1.1 Qlik Sense

Qlik Sense è un software della famiglia Qlik, società leader nell'ambito BI e data analytics. È un tool estremamente potente con una duplice funzione: data discovery, in quanto, offrendo un'esplorazione del dato senza limiti e un feedback istantaneo, consente agli utenti di lavorare alla velocità del pensiero, di lasciarsi guidare dall'intuizione ed addentrarsi in un deep insight; reporting, dal momento che mette a disposizione un numero ampio di oggetti grafici per presentare e visualizzare i dati facendo emergere evidenze e relazioni. Esiste sia nella versione desktop che in quella server.

Sono quattro i grandi pregi che l'hanno reso tanto apprezzato e diffuso: l'integrazione di più sorgenti eterogenee, la gestione dei dati in-memory, il motore associativo e la "potenza del grigio". Analizziamo queste caratteristiche singolarmente per scoprirne il valore.

Qlik Sense consente di connettersi ai dati in qualsiasi sorgente grazie a un'ampia scelta di connettori. Quando si crea una connessione, questa viene salvata in Qlik Sense per consentire di selezionare e caricare rapidamente dati dalle sorgenti utilizzate più spesso. È possibile connettersi a database, dati di social media, file locali, file remoti e file web. Indipendentemente dalla sorgente, ci sarà omogeneità tra i dati importati.

I dati vengono caricati in memoria usando un algoritmo di compressione accelerato. Dopo ogni istruzione di caricamento, Qlik trasforma i dati estratti in due tipi di tabelle: tabelle dei dati e tabelle dei simboli. Per ogni campo viene creata una tabella dei simboli contenente una riga per ogni valore distinto di quell'attributo. Ogni record comprende un puntatore e il valore stesso del campo, che sia numerico o testuale. Fondamentalmente le tabelle dei simboli sono così tabelle di ricerca per i valori dei campi. Le tabelle dei dati non conterranno più i dati stessi ma solo i puntatori e, dal momento che i puntatori possono essere utilizzati per cercare il valore reale nelle tabelle dei simboli, nessuna

informazione viene chiaramente persa. Questi puntatori non sono però puntatori ordinari, sono bit-stuffed pointers, cioè hanno solo il numero di bit necessario per rappresentare il campo, non uno di meno, non uno di più. Quindi se un campo contiene quattro valori distinti, l'indice è lungo solo due bit, essendo due i bit che occorrono a rappresentare quattro valori. Per cui la tabella dei dati diventa molto più piccola di quanto sarebbe stata altrimenti. Questo è il motivo per cui la curva di utilizzo di RAM tende ad appiattarsi al crescere del volume dei dati come è visibile dalla figura sottostante.

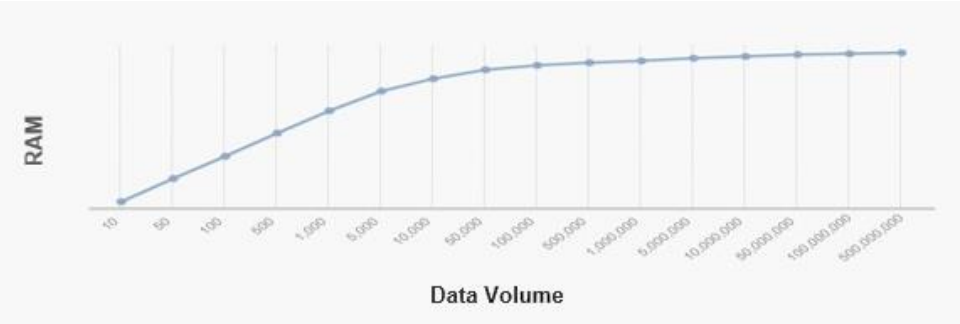

Figura 1.1: Utilizzo di RAM al crescere del volume dati

Il motore associativo QIX (Qlik Indexing Engine) è il vantaggio tecnologico principale. È stato esplicitamente progettato per consentire agli utenti di vedere l'intera storia che vive all'interno dei propri dati, esplorare in forma libera, sondare tutte le possibili associazioni, applicare selezioni interattive e ricerche di parole chiave. Dopo ogni clic, il motore QIX ricalcola istantaneamente e dinamicamente tutte le analisi al contesto attuale e lo fa tanto rapidamente quanto l'utente può formulare domande. In questo modo si è stimolati a porre nuovi quesiti e, mossi dall'intuizione, si è spinti ad addentrarsi sempre più nell'analisi e nella scoperta. Un'altra caratteristica distintiva del motore Qlik associativo è il mantenimento di un ambiente unificato per tutte le analisi su un'intera applicazione, senza dover collegare oggetti tra loro o eseguire più queries. Quando un utente effettua una selezione o esegue una ricerca, tutte le visualizzazioni, le osservazioni e le associazioni vengono immediatamente aggiornate. Questo dà agli utenti la possibilità non solo di interagire con i singoli grafici, ma anche di comprendere l'impatto delle loro domande sull'analisi circostante, a diversi livelli di dettaglio, alla velocità del pensiero.

La "potenza del grigio" è una capacità unica e potente che consente agli utenti di vedere nelle loro analisi non solo i dati correlati, relativi alle loro selezioni, ma anche i valori non correlati. Queste informazioni spesso trasmettono le indicazioni più interessanti, come prodotti che non sono stati venduti o clienti che non hanno acquistato, aiutando gli utenti a scoprire aree o condizioni precedentemente impreviste o rischiose. Quello che succede è che dopo ogni clic il motore QIX ricalcola istantaneamente tutte le analisi al contesto attuale e mette in risalto le relazioni tra i dati utilizzando spunti di colore facili da capire: verde (selezionato), bianco (associato) e grigio (non correlato). Le associazioni possono così essere di natura positiva (un valore correlato a un altro) o negativa (un valore che non è correlato). È importante notare che i valori non correlati (grigi) forniscono informazioni tanto quanto quelli positivi (bianchi) in quanto spesso indicano nuove opportunità o aree incognite.

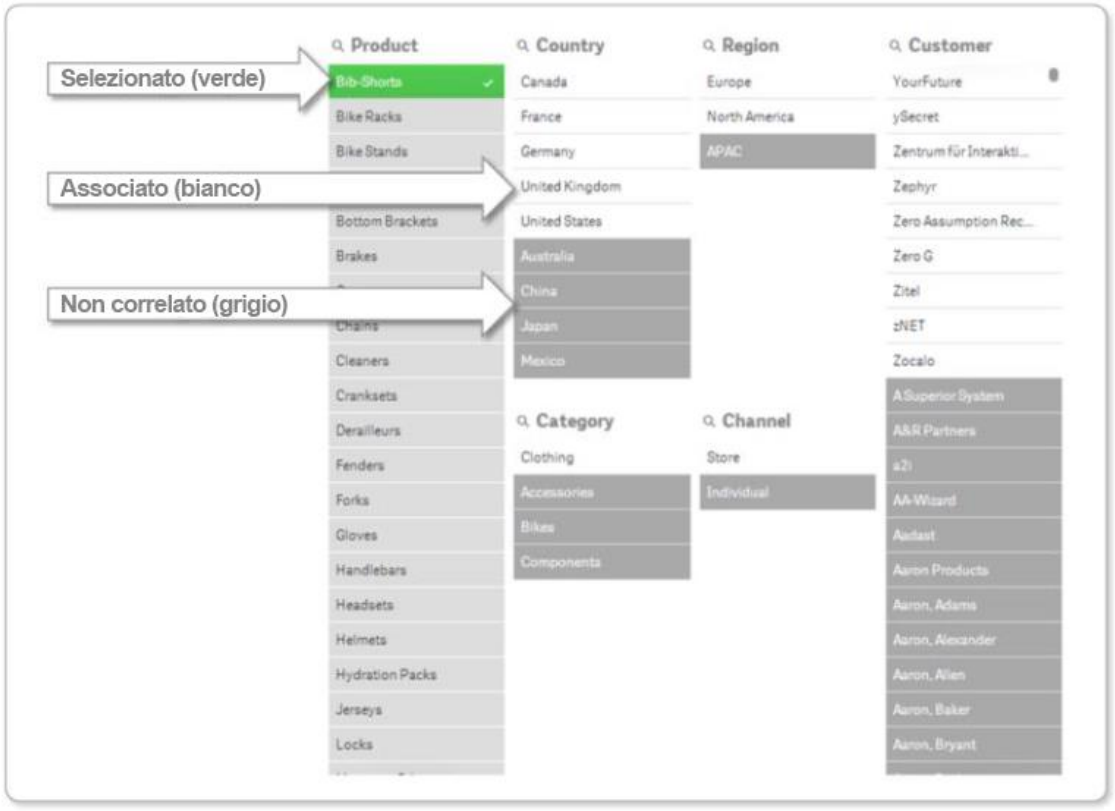

Figura 1.2: La “potenza del grigio”

I componenti fondamentali di Qlik Sense sono l’hub e la QMC (Qlik Management Console). L'hub è lo spazio in cui sono contenute tutte le apps, cioè i progetti sviluppati che l'utente è autorizzato a vedere e utilizzare. Il contenuto all'interno dell'hub è organizzato in streams. Uno stream è una raccolta di apps a cui può accedere un gruppo di utenti specifico. Ovviamente è possibile che gli utenti abilitati a un dato stream dispongano di diritti di accesso differenti. Di default, Qlik Sense include due streams: “Everyone”, in cui a tutti gli utenti sono assegnati diritti di lettura e pubblicazione, e “Monitoring Apps”, contenente due apps di monitoraggio per Qlik Sense. Tutti gli utenti poi possiedono la loro area denominata “Lavoro” a disposizione per i contenuti personali; ciò assicura una chiara separazione tra le informazioni in cantiere (non ancora pubblicate) e quelle considerate affidabili (pubblicate). La Qlik Management Console rappresenta a tutti gli effetti il pannello di amministrazione. Fornisce un set di strumenti molto avanzati di configurazione. Consente di definire permessi e regole e di creare modelli di accesso differenti.

Qlik Sense offre anche altre tre interessanti caratteristiche. La prima è lo storytelling, un nuovo metodo di condivisione dell’analisi svolta. Sostanzialmente è una presentazione che permette di sottolineare gli elementi appresi, le evidenze riscontrate, le intuizioni avute e lo fa consentendo di calarsi direttamente nel contesto dell’app. Ciò permette di unire funzionalità di report, presentazione e analisi esplorativa. La seconda è la possibilità di creare dei custom objects o usarne di fatti da altri. Questi consentono di ampliare le funzionalità di visualizzazione di Qlik Sense utilizzando tecnologie web standard:

HTML, CSS, JavaScript. Infine, attraverso le sue API aperte, Qlik offre la capacità di integrazione con altri software consentendo lo scambio diretto di dati tra il motore QIX e i motori di calcolo di terze parti. Ciò permette di visualizzare i calcoli avanzati, realizzati da strumenti esterni, all'interno di Qlik Sense, in real-time e parallelamente all'esplorazione dell'utente.

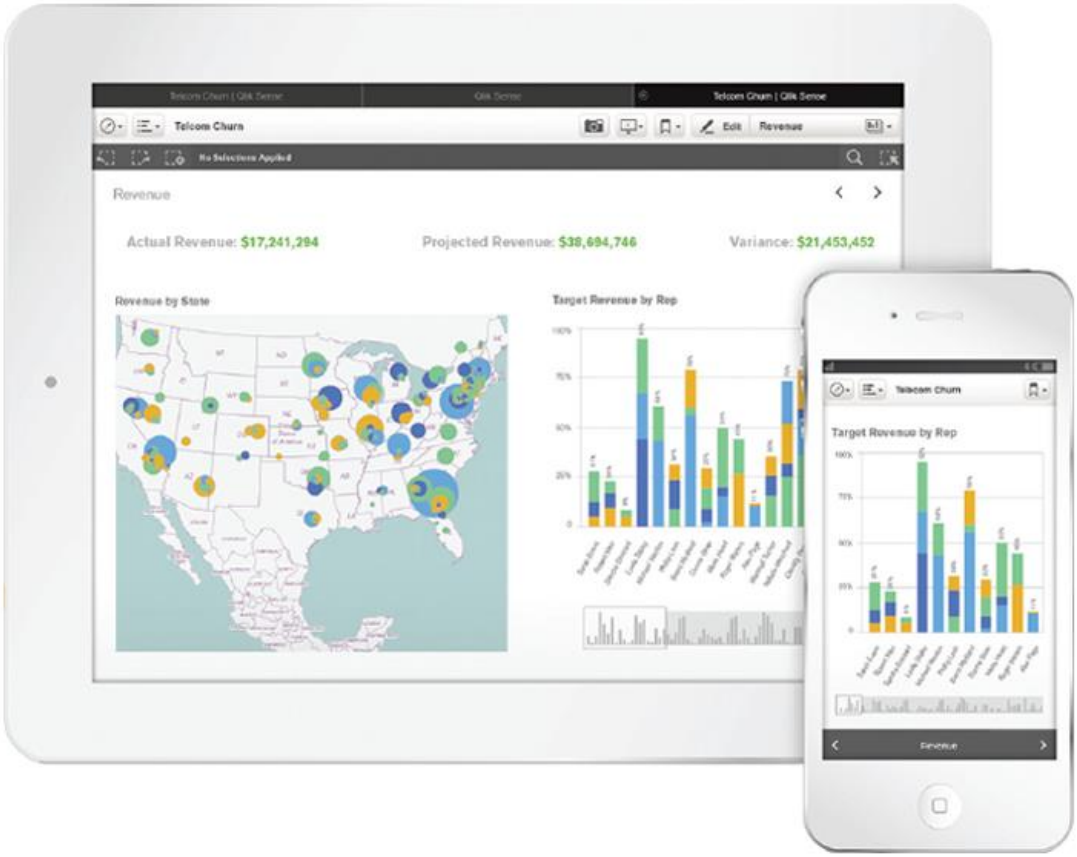

Figura 1.3: Qlik Sense

1.2 SQL Developer

SQL Developer è un software free della famiglia Oracle. È un IDE (integrated development environment), cioè un ambiente con interfaccia grafica che semplifica lo sviluppo e la gestione dei database Oracle sia nelle distribuzioni tradizionali che cloud. Consente agli utenti, così come agli amministratori, di eseguire le proprie attività sui database in pochi click, minimizzando i tempi di lavoro. Mette a disposizione un editor altamente potente per lavorare con SQL, PL/SQL, stored procedures Java ed XML. Consente di lanciare queries, generare piani di esecuzione, esportare dati nella piattaforma desiderata (XML, Excel, HTML, PDF, ecc..), debuggare, testare e documentare. Offre, inoltre, un pannello di amministrazione per gestire a tutto tondo un database sia nella fase di configurazione che in quella di auditing e recovery, semplificando il lavoro ai DBA (database administrators).

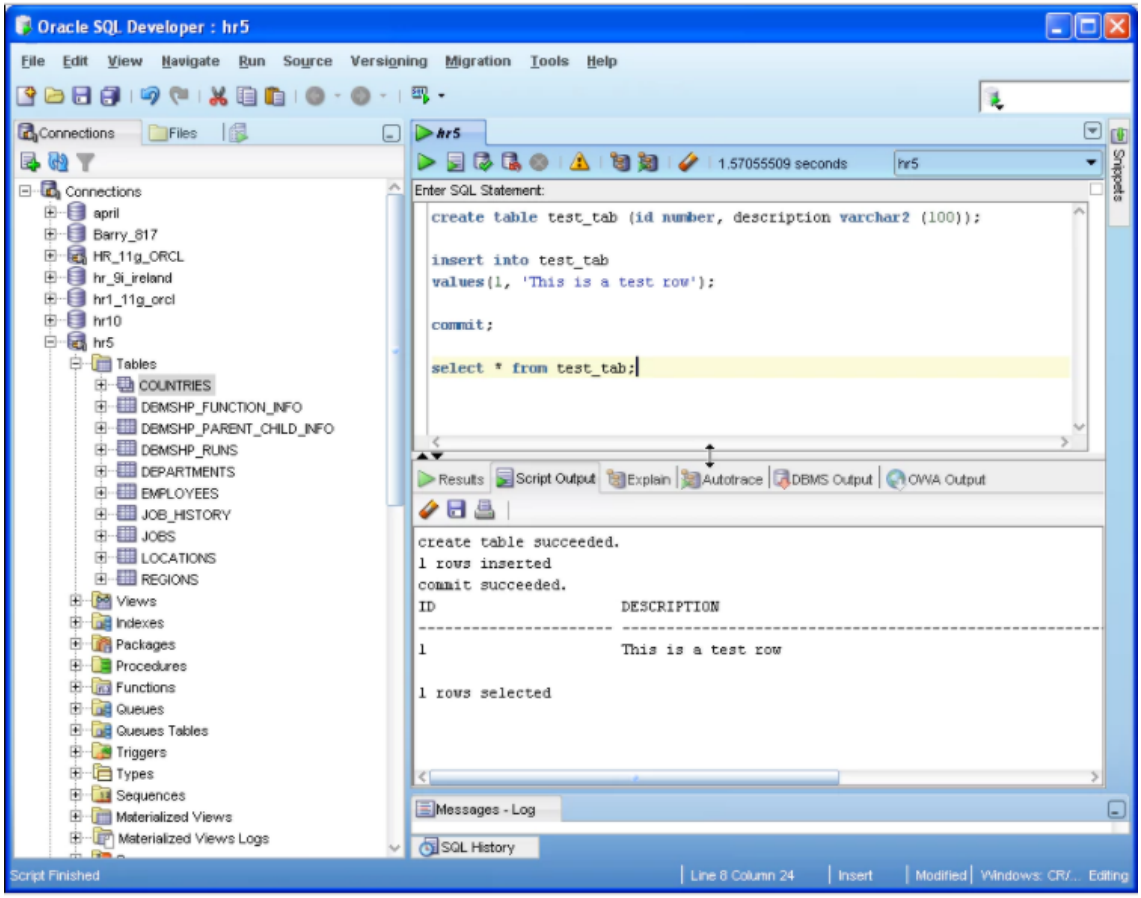

Figura 1.4: SQL Developer

1.3 R

R è un linguaggio di programmazione e un ambiente di sviluppo free per il calcolo statistico e la grafica. Consta di una suite integrata di funzioni software per la manipolazione dei dati, la computazione e la visualizzazione. Include: un valido sistema di gestione e storage dei dati; un vasto insieme di operatori per calcoli su arrays e matrici; un'ampia raccolta coerente e integrata di tools intermedi per l'analisi dei dati; strutture grafiche per lo studio e l'osservazione comparativa congiuntamente a tecniche di rappresentazione su schermo o su copia cartacea; un linguaggio di programmazione articolato ed efficace ma semplice da usare che include istruzioni condizionali, cicli, funzioni ricorsive definite dall'utente e strutture di input e output. Il termine "ambiente" intende caratterizzarlo per quello che è, cioè un sistema completo e coerente in rispetto a quanto spesso accade con altri software di analisi dei dati che sono un complesso di strumenti molto specifici e non flessibili. Anche se il linguaggio è fornito con un'interfaccia a riga di comando, sono disponibili diversi IDE che consentono di lavorare con R in un ambiente più esauriente che offre, oltre ad una console su cui scrivere ed eseguire direttamente del codice, altri strumenti quali debug, cronologia, plotting tools e gestione dei workspaces. Compila e gira su un'ampia varietà di piattaforme UNIX, Windows e MacOS. R e le sue librerie implementano un ricco assortimento di tecniche

statistiche e grafiche, tra cui la modellazione sia lineare che non lineare, l'analisi delle serie temporali, la classificazione, il clustering e altri. Un ulteriore punto di forza di R è la grafica statica, che può produrre grafici di qualità adatti a pubblicazioni, comprensivi di simboli matematici e formule dove necessario. La grafica dinamica e interattiva è disponibile tramite pacchetti aggiuntivi. R è infatti altamente ampliabile e può essere esteso tramite dei packages.

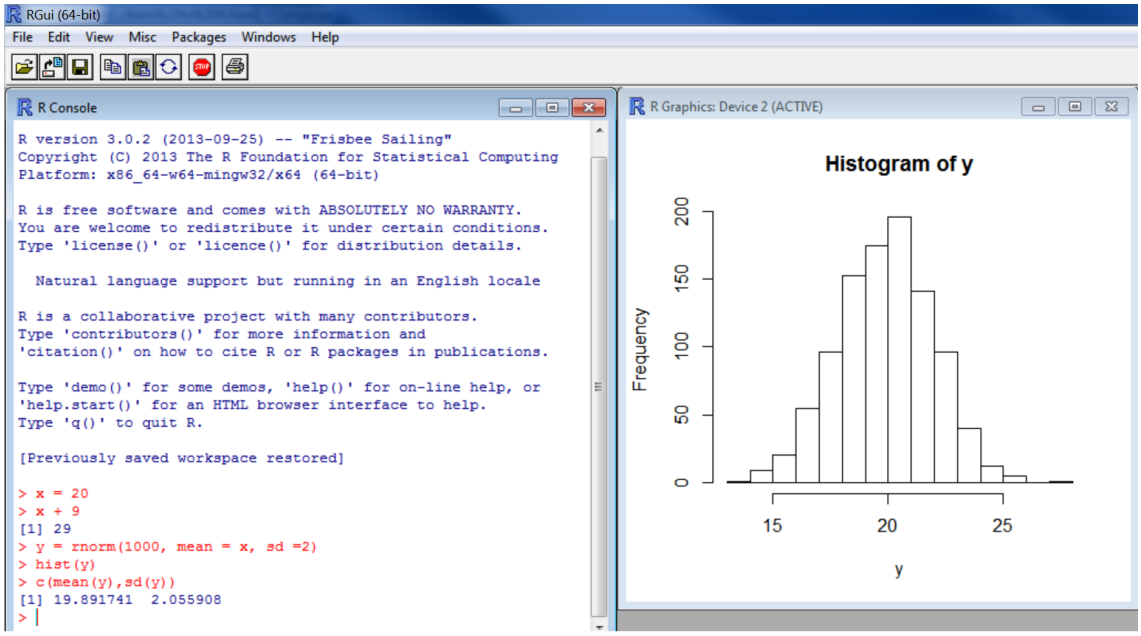

Figura 1.5: R

CAPITOLO 2
ARCHITETTURA HARDWARE E SOFTWARE AZIENDALE

Prima di entrare nel cuore della tesi, è bene soffermarsi sulla tecnologia hardware e software in uso in azienda. Questo capitolo è quindi utile per focalizzare il contesto di sviluppo del progetto e porre l'accento su alcuni aspetti di carattere logistico che stanno alla base di certe scelte d'implementazione.

2.1 Architettura hardware

Per quanto riguarda l'hardware, si espone solo ciò che concerne il mondo Qlik, in quanto è stato l'unico universo con cui si è entrati in contatto. Ci sono, sostanzialmente, due ambienti: ambiente di produzione e ambiente di test (chiamato anche ambiente di collaudo). Ovviamente l'ambiente di collaudo è utilizzato per sperimentare, provare e testare mentre l'ambiente di produzione serve a mantenere ciò che è pronto per essere pubblicato e rilasciato. L'ambiente di produzione si fonda su un server fisico e un disco NAS 250 GB CIFS.

Il server fisico ha i seguenti requisiti:

- Intel Xeon CPU e7-8850 v2 @ 2.30GHz (4 processori);
- 96 GB RAM;
- Windows Server 2012 R2 Standard;
- disco da 580 GB, non estendibile, costituito da due partizioni logiche:
    - C: contiene il sistema operativo e il software Qlik Sense (attualmente sono occupati 40 degli 80 GB totali);
    - D: contiene la base dati dei QVD (QlikView Data, file contenenti i dati esportati da Qlik) (attualmente sono occupati 20 degli 500 GB totali).

Il disco NAS è estendibile ed è utile per avere:

- spazio per deposito dati utente aggiuntivi;
- eventuale ulteriore spazio per QVD (se il disco principale è saturo).

L'ambiente di test è composto dal solo server fisico che è un clone del server di produzione.

I server, sia di produzione che di collaudo, sono raggiungibili, rispettivamente, agli indirizzi IP 172.20.1.35 e 172.20.32.110. Sono quindi accessibili o collegandosi dalla rete locale Rai Pubblicità o dall'esterno tramite VPN. Il nome del server di produzione è ZTOCAV914, mentre quello di collaudo è PTOCAV905. Entrambi stanno sul dominio ict.corp.rai.it. Dal momento che il server di produzione contiene le apps rilasciate accessibili con i dovuti permessi all'utenza finale, è raggiungibile, per semplicità mnemonica, anche tramite il seguente alias: biqliksense.raipubblicita.it.

2.2 Architettura software

Come riportato nel paragrafo precedente, parlando dell'impianto hw per Qlik, sia il server di produzione sia quello di collaudo hanno a bordo il software Qlik Sense. La versione installata è la September 2017 patch 1 (11.14.4).

L'architettura Qlik Sense implementata è basata, quindi, su un unico server fisico ospitante il Qlik Sense Central Node.

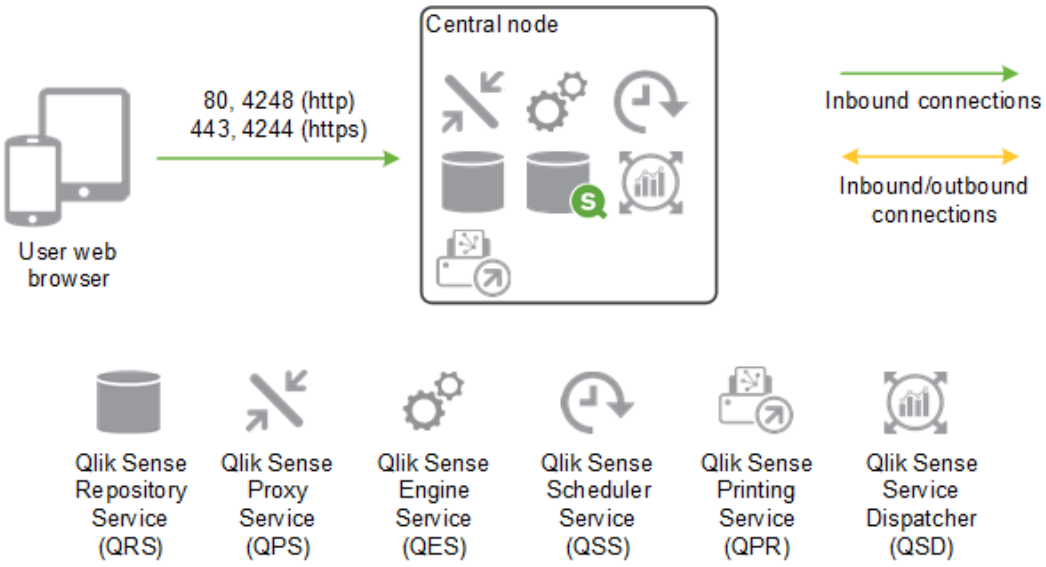

Figura 2.1: Qlik Sense Central Node

Come mostrato nella figura soprastante, i servizi in esecuzione sul Central Node sono:

- Qlik Sense Repository Service (QRS) – gestisce la persistenza e la sincronizzazione delle apps e i dati di configurazione. È il cuore del Central Node in quanto è indispensabile a tutti gli altri servizi per eseguire e servire le apps. Inoltre il QRS memorizza le strutture delle apps e i percorsi dei file binari (cioè i dati memorizzati nel local file system);

- Qlik Sense Repository Database (QRD) – è utilizzato per leggere e scrivere dati nel repository;
- Qlik Sense Proxy Service (QPS) – gestisce l'autenticazione, le sessioni utente e il bilanciamento del carico di lavoro;
- Qlik Sense Scheduler Service (QSS) – gestisce i reloads periodici delle apps e altri tipi di ricaricamento innescati dagli eventi;
- Qlik Sense Engine Service (QES) – è il servizio applicativo che si occupa di tutta la computazione e la logica;
- Qlik Sense Printing Service (QPR) – gestisce l'esportazione in Qlik Sense;
- Qlik Sense Service Dispatcher (QSD) – è un controller di servizio utilizzato per avviare e gestire i seguenti servizi:
    - Broker Service – funge da interfaccia e intermediario tra i servizi lanciati dal QSD;
    - Data Profiling Service – è usato per accedere e modificare il modello di caricamento delle apps;
    - Hub Service – controlla quale contenuto un utente è abilitato a consultare in base ai permessi forniti;
    - Migration Service – garantisce che le apps possano essere utilizzate nella versione di Qlik Sense correntemente installata;
    - Capability Service – gestisce la configurazione delle funzioni di sistema.

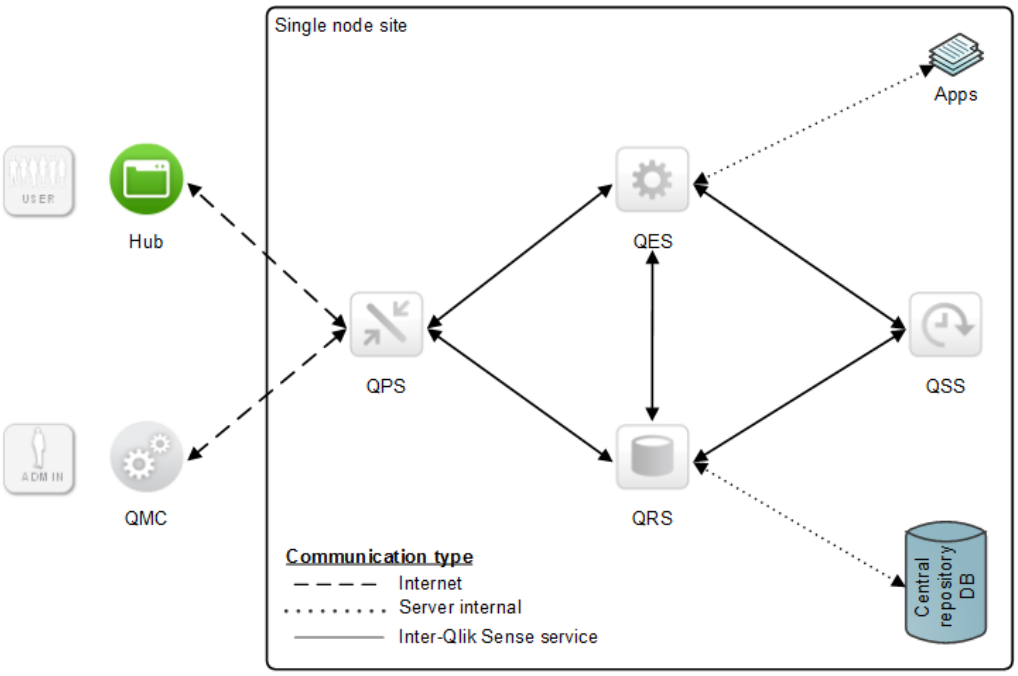

Figura 2.2: Interazione tra servizi nel Qlik Sense Central Node

Passando a parlare di data governance, il sistema per la gestione di basi di dati (DBMG – DataBase Management System) adottato è l'Oracle Database, nella versione 11g. Oracle Database fa parte dei RDBMS (Relational DataBase Management System) ed è quindi un sistema di database basato sul modello relazionale.

Addentriamoci adesso nel patrimonio informativo aziendale. Nel mondo Rai Pubblicità ci sono quattro grandi ambienti: sviluppo, test, collaudo (chiamato anche pre-produzione) e produzione. Sviluppo è l'ambiente nel quale si compilano gli oggetti, si verifica la sintassi e si correggono eventuali errori. Qui i dati presenti sono solamente un campione del totale, in modo da facilitare e velocizzare le operazioni di verifica preliminari. In test, invece, la mole di dati è considerevole e aggiornata. Questo ambiente è quindi utile a fare un check più approfondito e preciso. In collaudo, solitamente, si dà modo all'utente di accedere; ciò consente di svolgere gli ultimi controlli col supporto di colui a cui è destinato il prodotto finale. Infine, produzione è l'ambiente contenente gli oggetti verificati e rilasciati. Dal momento che per Qlik si dispone di sole due licenze (sono due i server fisici su cui il software è installato, come spiegato nel paragrafo precedente), il server Qlik di collaudo punterà all'ambiente di collaudo mentre il server Qlik di produzione punterà all'ambiente di produzione.

Sebbene spesso si usi il termine database per riferirsi a tutto ciò che riguarda un insieme di dati strutturato omogeneo, sarebbe più corretto distinguere il database, inteso come la collezione dei file fisici in cui sono memorizzati i dati, dall'istanza che, invece, rappresenta l'insieme delle aree di memoria e dei processi in background necessari ad accedere ai dati, quindi quanto è necessario al database per funzionare. Caso tipico è avere una sola istanza di database su una macchina ma non è raro trovarne anche più di una. Sarebbe più esatto, quindi, parlare di istanza di database e non semplicemente di database. Nello specifico, nel contesto Rai Pubblicità, per ognuno dei quattro ambienti descritti sopra ci sono tre istanze: istanza del database commerciale (DB commerciale), istanza del data warehouse (DWH), istanza SAP. Quest'ultima è utile per mantenere i sistemi contabili ma non rientra nell'ambito di competenza del progetto, per cui non sarà approfondita ulteriormente. Al contrario si fornisce una spiegazione più accurata e rigorosa delle altre due istanze.

Figura 2.3: Istanze di database Rai Pubblicità

Il DB commerciale contiene tutti i sistemi transazionali aziendali. Qui possiamo quindi trovare i dati provenienti dal mondo TV/radio, dal mondo cinema, dal mondo web, le anagrafiche, i portafogli, i sistemi di booking, i dati di fatturazione, i flussi autorizzativi. Il DB commerciale ha pertanto l'obiettivo di registrare, in tempo reale, i dati con il quale viene alimentato. È alla base dei sistemi OLTP (On Line Transaction Processing) aziendali che memorizzano, modificano e mostrano dei records in real-time. Le operazioni di lettura e scrittura concorrenti devono garantire le famose quattro proprietà: atomicità, coerenza, isolamento e durabilità (ACID). Per questo il dato è normalizzato,

l'informazione viene inserita e modificata una volta sola, non ci sono ridondanze. Fondamentale è garantire il dato, la sua integrità.

Al contrario, il data warehouse è progettato con l'obiettivo di ottimizzare le prestazioni, agevolare i confronti. La velocità di risposta a queries complesse ha un'importanza maggiore rispetto all'organizzazione dei dati. Per questo il dato è denormalizzato ed è spesso il risultato di aggregazioni a diverso livello di dettaglio, favorendo così operazioni di analisi e raffronto. Il data warehouse è la fonte dei sistemi OLAP (On-Line Analytical Processing) che, fornita una serie storica di dati, mirano ad analizzarli da diverse prospettive o dimensioni creando quello che si definisce cubo multidimensionale o ipercubo.

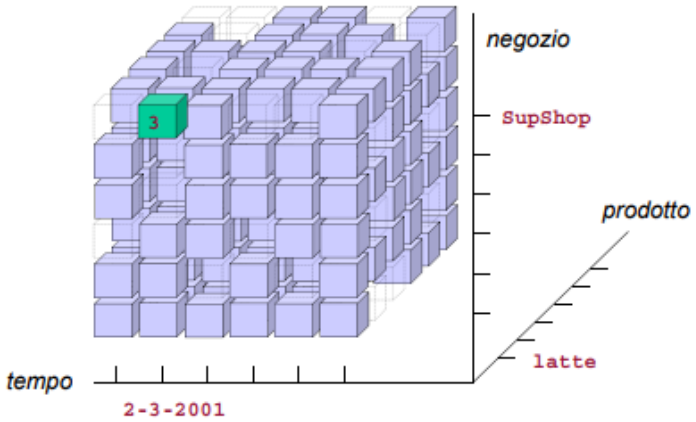

Figura 2.4: Esempio di ipercubo[2]

Il data warehouse viene aggiornato tramite un caricamento notturno giornaliero e i dati contenuti hanno una profondità storica di 5 anni. Sono presenti i dati di venduto Rai Pubblicità e le anagrafiche ribaltate dal DB commerciale utili a fornire un contesto alle informazioni presenti sul DWH. L'utilità del ribaltamento è quella di evitare i database links, cioè connessioni tra due istanze di database differenti, nelle queries di analisi quotidiana perché, ovviamente, si inciderebbe negativamente sulle prestazioni nei tempi di risposta alle interrogazioni. Oltre ai dati interni, nel data warehouse aziendale è scaricato il flusso dati proveniente dalla Nielsen, società che si occupa di ricerche di mercato relative a ciò che le persone guardano e acquistano. La Nielsen rileva, quindi, i dati di mercato, non solo di Rai Pubblicità ma anche delle altre concessionarie. Avere questi dati permette così sia di fare dei confronti tra il mondo interno e quello dei competitors, sia di individuare i clienti che si stanno attivando di cui non abbiamo informazioni e che potrebbero quindi diventare nuovi investitori. Naturalmente ciò che viene acquisito dalla Nielsen avrà delle anagrafiche proprie. Per questo il data warehouse è arricchito da oggetti di ri-mappatura che consentono di elaborare i dati provenienti dal mondo Nielsen e arricchirli con le anagrafiche Rai Pubblicità. In questo modo si riesce a

[2] Golfarelli M., Rizzi S., *Data warehouse. Teoria e pratica della progettazione*, McGraw Hill, Milano, 2006

trattare dati provenienti da fonti eterogenee e a confrontarli per trarne interessanti spunti di analisi ed evidenze di rilievo.

	OLTP	OLAP
Finalità	Supporto all'operatività	Supporto al processo decisionale
Utenti	Molti, livello operativo	Pochi, livello direzionale
Modalità di utilizzo	Guidata, per processi e stadi successivi	Interrogazione ad hoc
Quantità di dati per operazione elementare	Bassa: centinaia di record per ogni transazione	Alta: milioni di record per ogni query
Qualità	In termini di integrità	In termini di consistenza
Orientamento	Per processo/applicazione	Per soggetto
Frequenza di aggiornamento	Continua, tramite azioni	Sporadica, tramite funzioni esplicite
Copertura temporale	Dati correnti	Storica
Ottimizzazione	Per accessi in lettura e scrittura su una porzione di dati	Per accessi in sola lettura su tutta la base di dati

Figura 2.5: Confronto tra sistemi OLTP e OLAP[3]

Ad una istanza possono essere abilitati a connettersi più utenti con permessi di accesso, configurabili, che possono variare da utente a utente. Le connessioni alle varie istanze sono gestite tramite un file TNS (Transparent Network Substrate), ovvero un file di testo che serve a configurare le connessioni lato client verso i database Oracle, e richiedono l'inserimento dell'username e della password dell'utente interessato. Ad ogni utente è associato uno schema che è un contenitore, inizialmente vuoto, messo a disposizione per racchiudere gli oggetti di cui quell'utente è proprietario. Riferendoci al mondo Qlik, sono due gli utenti configurati per ogni istanza: QLIKUSR e QLIKADM. QLIKUSR ha accesso in sola lettura a tutti i dati. Il suo schema sarà popolato da tutti gli oggetti che andremo a richiamare in Qlik. Lo schema QLIKADM, invece, è occupato da tutte le tabelle di configurazione del framework di ETL custom. Quest'ultimo concetto è proprio l'argomento del prossimo capitolo.

[3] *Database Master*, < http://databasemaster.it/dbms-oltp-dbms-olap/ >, ultima consultazione: 24/01/2018

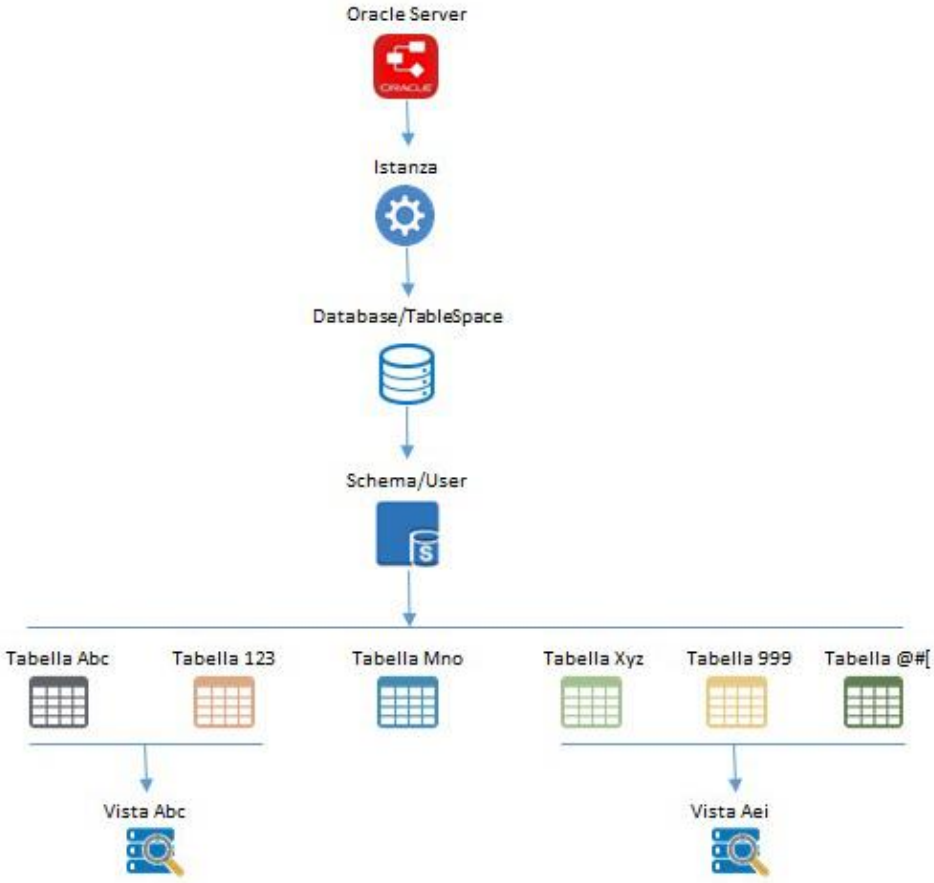

Figura 2.6: Struttura Oracle Server

CAPITOLO 3
FRAMEWORK DI ETL CUSTOM

Per quanto concerne la partecipazione alla progettazione e all'implementazione del sistema aziendale di BI, il primo passo è stato quello di aver studiato e contribuito all'ottimizzazione del framework di ETL che consente di centralizzare e uniformare il caricamento dati su Qlik Sense migliorandone anche le prestazioni. In questo capitolo si analizza inizialmente l'importanza di questo processo per poi mettere a nudo le qualità della soluzione adottata e approfondirne gli steps che la compongono.

3.1 Extract, Transform and Load (ETL)

ETL è l'acronimo inglese di Extract, Transform and Load, le tre fasi costituenti il processo di recupero, predisposizione e caricamento dei dati per un sistema OLAP. Il fine è quindi quello di prelevare dei dati da fonti anche eterogenee, compiere delle modifiche e darli in pasto al complesso destinatario. La distinzione dell'intero procedimento in tre momenti diversi si sposa con le caratteristiche differenti dei tre stadi che, adesso, andiamo a sviscerare singolarmente.

L'estrazione è il gradino iniziale di questo processo e rappresenta la fase di acquisizione dei dati. Le sorgenti possono essere file (CSV, JSON, XML), database transazionali o altri sistemi informatici gestionali. Si può scegliere di prendere l'intero dataset o solo una porzione. In generale, ci sono due modalità di estrazione: statica, in cui si fa una fotografia dei dati operazionali, o incrementale nella quale si preleva solo ciò che è variato dall'ultima estrazione. In questa fase sarebbe opportuno anche tenere conto delle prestazioni, cercando di impattare il meno possibile sui sistemi sorgente. L'output di questo step è l'insieme dei dati estratti, consolidati con un formato adatto al passaggio successivo.

La trasformazione è la fase centrale del processo di ETL sebbene sia opzionale in quanto è possibile non eseguire nessuna manipolazione sui dati estratti e programmare direttamente il caricamento. Solitamente questo stadio è composto da due sotto-fasi: pulizia ed elaborazione vera e propria. La pulizia mira a migliorare la qualità dei dati eliminando inesattezze, inconsistenze e difetti dovuti a valori errati/non previsti per un attributo, dati duplicati, dati mancanti, incoerenze tra dati logicamente associati. L'elaborazione, invece, ha lo scopo di far sì che i dati rifiniti siano il più possibile aderenti alle regole di business del sistema di analisi cui l'ETL è rivolto. Le modifiche includono: traduzione di dati codificati e conversioni per garantire l'omogeneità rispetto all'integrazione delle diverse fonti; derivazione di nuovi campi da dati calcolati e/o

arricchiti; join per accoppiare dati recuperati da più sorgenti; raggruppamenti con l'obiettivo di diminuire la granularità dei dati. Bisogna porre particolare attenzione alle aggregazioni in quanto, se da una parte non vogliamo avere un dettaglio eccessivo che porterebbe a un degrado delle performance delle queries effettuate, dall'altro bisogna mantenere la granularità adatta ad effettuare le dovute analisi.

Infine, l'ultima fase è quella del caricamento in cui c'è la memorizzazione del dato estrapolato ed elaborato nelle strutture predisposte. Avviene quindi la propagazione degli aggiornamenti effettuati, con il dato che viene reso disponibile al sistema target finale. Il caricamento può avvenire secondo due modalità: refresh, nella quale i dati vengono riscritti integralmente andando a sostituire quelli vecchi; update, in cui si registrano solo i cambiamenti senza alterare i dati già esistenti.

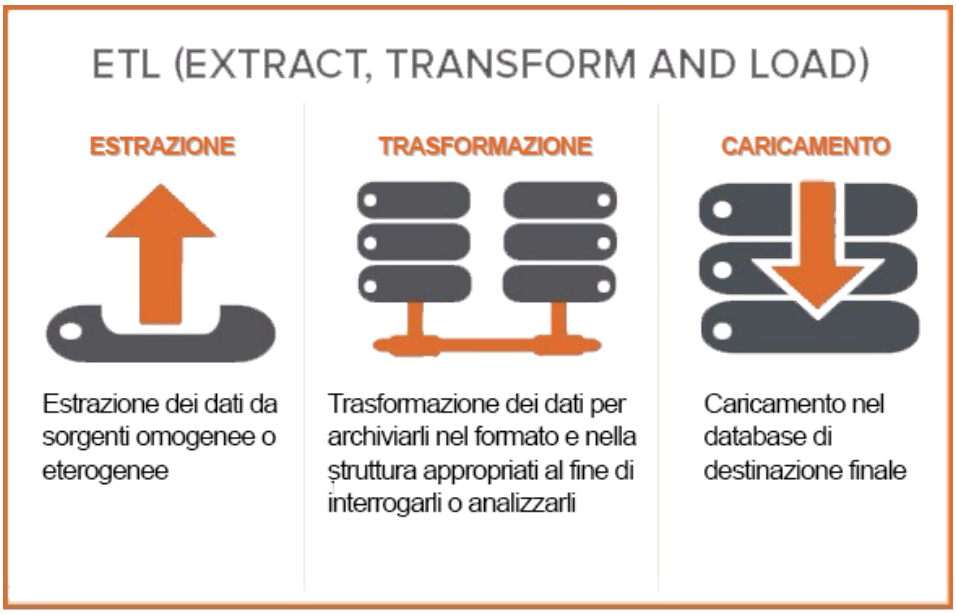

Figura 3.1: ETL in sintesi

Dotarsi di uno strumento di ETL assume una notevole rilevanza nelle performance delle valutazioni dei trend aziendali in quanto consente di semi-automatizzare le operazioni di estrazione, trasformazione e caricamento dei dati. Si riducono così i tempi di sviluppo poiché si eseguono operazioni standard, ben definite. Si migliora inoltre la manutenzione dell'intera struttura dal momento che, essendoci delle fasi pre-determinate e ripetitive di operatività, risultano più semplici le modifiche e le correzioni di bug ed errori in cui spesso siamo supportati dagli stessi tools di ETL che danno la possibilità di osservare graficamente il flusso dati. Ciò consente di visionare perfettamente i passaggi eseguiti rendendo più facile l'interpretazione delle variazioni apportate ai dati caricati e l'impatto delle trasformazioni. Una caratteristica interessante che gli strumenti di ETL dovrebbero offrire e che spesso incorporano è quella di consentire l'interazione di fonti dati disomogenee e ciò è un grosso vantaggio in quanto permette di integrare il patrimonio informativo aziendale senza la necessità di mettere in piedi sistemi di estrazione dati differenti.

3.2 Panoramica sul framework di ETL aziendale

La necessità di avere un framework di ETL personalizzato nasce dall'esigenza di disporre di un sistema di estrazione, manipolazione e messa a disposizione dei dati per Qlik Sense che sia: centralizzato, in modo da avere uno strato sorgente unico per tutti i progetti che si vogliono realizzare; semi-automatizzato, per rendere più meccaniche e rapide le operazioni preliminari; flessibile, in modo che si adatti al crescere delle fonti alimentanti e delle apps. È inoltre fondamentale che sia gestita opportunamente la sicurezza sia a livello di profilazione utente, in modo che siano controllati l'accesso all'hub di Qlik e alla QMC (Qlik Management Console) sia a livello di segregazione del dato per utente.

Il tipo di architettura implementata prevede una soluzione modulare strutturata a livelli definita "4 Tier QVD Based". Si suddivide quindi l'operatività dell'ambiente server in quattro livelli a valle dei quali il risultato sarà sempre uno strato di QVD. Il motivo dell'uso dei QVD risiede, principalmente, in due ragioni: velocità nella lettura dei dati, dal momento che il formato, essendo Qlik nativo, è ottimizzato per essere usato negli script di caricamento Qlik e ciò permette di incrementare la velocità di lettura dei dati da 10 a 100 volte rispetto all'utilizzo di altre sorgenti; riduzione considerevole dei dati trasferiti da sorgenti esterne, consentendo di diminuire drasticamente il carico di lavoro sui server di database e il traffico in rete.

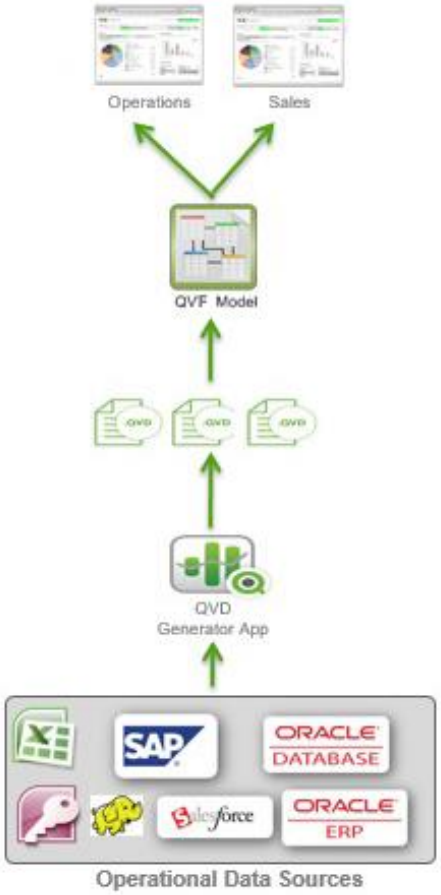

Figura 3.2: Dalle sorgenti eterogenee ai QVD

I livelli di elaborazione individuati sono i seguenti:

1. Estrattore ETL 01 centralizzato
2. Estrattore ETL 02 centralizzato
3. Estrattore ETL 01 app
4. Estrattore ETL 02 app

Funzionalmente il framework è pensato per raccogliere ed elaborare con i primi due livelli tutte le informazioni che si ritengono trasversalmente utili, ossia ciò che viene individuato come informazione condivisibile da più applicazioni (sempre almeno due apps). Questi primi due livelli vengono definiti, appunto, centralizzati e mirano a creare il common data layer. Mentre con l'ETL 01 centralizzato l'obiettivo è quello di caricare i dati così come sono dalle fonti esterne al fine di disaccoppiare lo strato sorgente dal mondo Qlik, l'ETL 02 centralizzato consente di applicare sui dati delle trasformazioni convenienti a più apps.

I livelli 3 e 4 sono definiti, invece, di applicazione o di progetto. Nel dettaglio, il livello 3 è propedeutico a raccogliere tutti i dati attinenti al progetto/applicazione che si sta sviluppando e questi comprendono i QVD provenienti dal common data layer, archiviati dall'estrattore ETL 02 centralizzato e/o, nel caso di necessità proprie dell'app, fonti dati esterne specifiche. Il 4° livello consente, invece, un'ultima azione di data modeling e/o di aliasing sui campi.

Il livello detto applicativo, cioè l'app vera e propria, si appoggerà sul risultato a valle di questi quattro processi.

Il motivo per cui si hanno degli ETL centralizzati e degli ETL di app risiede, come accennato precedentemente, nel fatto che, al crescere del piano di sviluppo della BI, si mira a creare un data lake che sia di riferimento per tutti i progetti che si andranno ad implementare in quanto dovrà contenere, in maniera propriamente strutturata, tutto il patrimonio informativo aziendale. Avere invece sia degli ETL di livello 1 che degli ETL di livello 2 è giustificato dall'importanza che assume il possedere un'architettura a più strati con un'area di staging. Questa infatti è una zona di transito che consente di separare la fase di estrazione dalle fasi successive e consolidare i dati provenienti da sistemi differenti. Ciò si traduce nella possibilità di realizzare operazioni complesse di trasformazione e pulizia dei dati e nella presentazione di un modello dati ancora molto vicino a quello sorgente. L'unico svantaggio è rappresentato dall'introduzione della ridondanza che significa praticamente ulteriore spazio occupato in memoria ma, date le risorse disponibili, questo non è un problema di riguardo.

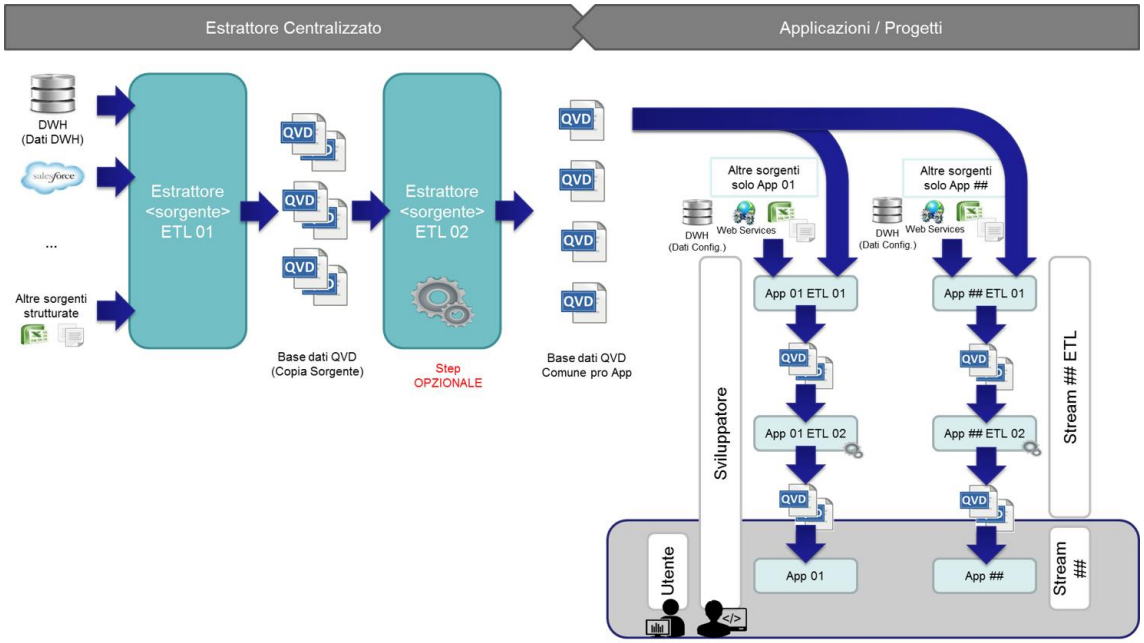

Figura 3.3: Architettura di riferimento ETL custom

Per sostenere le logiche del framework fin qui descritte è necessario predisporre di un'adeguata alberatura delle directories in modo da suddividere in maniera congrua gli output delle diverse fasi.

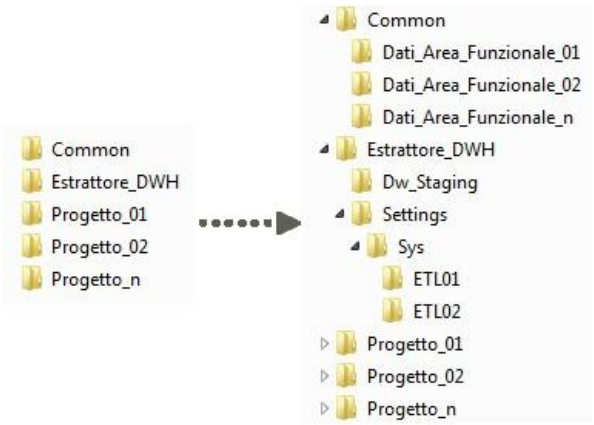

Figura 3.4: Gerarchia delle cartelle del framework

La cartella *Estrattore_DWH* è l'area destinata ad ospitare il risultato delle operazioni compiute dall'estrattore centralizzato e quindi, nello specifico:

- nella sotto-cartella *Dw_Staging* ci saranno i QVD estratti dall'ETL 01 sorgente costituenti la cosiddetta "Base dati (copia sorgente)";
- nelle sotto-cartelle *Settings/Sys/ETL01* e *Settings/Sys/ETL02* si troveranno i QVD contenenti i log dei livelli 1 e 2.

Nella directory *Common* saranno invece ospitati i QVD risultanti dal livello di data modeling centralizzato (cioè l'estrattore ETL 02), tematizzando le estrazioni per usi specifici in apposite folders, ad esempio *Dati_Area_Funzionale_01*.

Allo stesso livello di alberatura delle cartelle *Estrattore_DWH* e *Common* ci saranno le cartelle relative ai singoli progetti/applicazioni (*Progetto_01*, *Progetto_02*, *Progetto_n*).

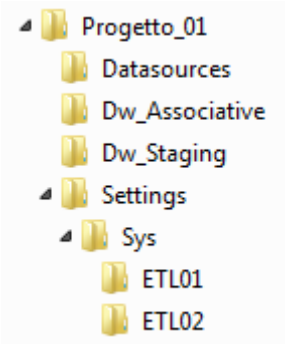

Figura 3.5: Cartella di progetto

Come si può osservare dalla figura soprastante, all'interno delle directories di progetto viene mantenuta la stessa struttura che si ha nella cartella *Estrattore_DWH* in modo tale che la gestione delle estrazioni delle fasi 3 e 4 sia identica a quella delle fasi 1 e 2. Sono però presenti due ulteriori sub-directories:

- *Datasources*, contenente ulteriori fonti dati, specifiche per l'app, che, insieme a ciò che è presente in *Common*, costituiscono la sorgente dati per il progetto;
- *Dw_Associative*, che conterrà il risultato dell'elaborazione di livello 2 di app.

Le applicazioni ovviamente attingeranno ai QVD presenti nella corrispettiva cartella *Dw_Associative* di progetto per costituire la nuvola dati di interesse.

Il framework, con i suoi quattro livelli, non è altro che un insieme di file QVF, cioè applicazioni Qlik Sense, che si appoggiano a strutture di configurazione implementate sull'istanza DWH. Questi oggetti sono compilati nello schema QLIKADM e, solo tramite l'utenza proprietaria che ha accesso sia in lettura che in scrittura, è possibile configurarli. I file QVF menzionati sopra sono dei template standard che consentono, a partire dalle configurazioni effettuate sulle tabelle di settings, la generazione dinamica del codice Qlik Sense finalizzato a realizzare una delle fasi di ETL. Esistono template per l'ETL 01, per l'ETL 02 e per il livello applicativo. In generale, ci sono due QVF per gestire l'ETL centralizzato (livelli 1 e 2) e tre QVF per ogni app che si vuole implementare (QVF di ETL 01, QVF di ETL 02, QVF di app). I QVF di estrattore sorgente sono identificati dal PROJECTID = 1, mentre i QVF di app da numeri interi crescenti. Sulla QMC sono impostati i tasks di reload in modalità concatenata cosicché il primo richiama progressivamente tutti gli altri portando al corretto refresh di tutte le apps. Questa operazione è attualmente settata per avvenire durante la notte di domenica. Ovviamente è anche possibile ricaricare manualmente ogni fase a proprio piacimento.

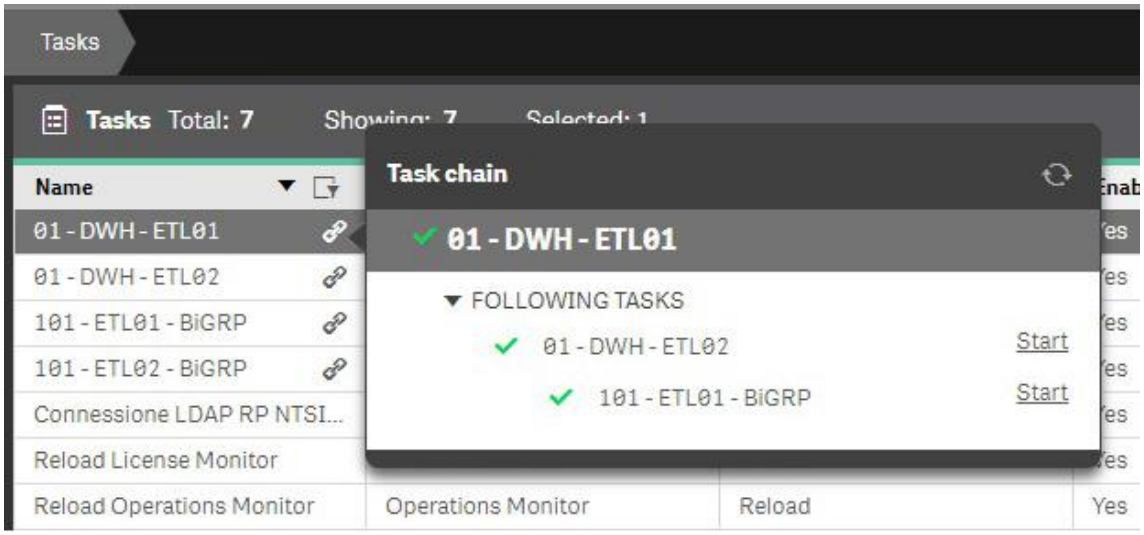

Figura 3.6: Tasks di ricaricamento dati

I tasks sono gestiti dal Qlik Sense Scheduler Service (QSS). Come è possibile vedere dalla figura sottostante, oltre ai tasks di reload degli ETL, è impostato un task di tipo "User synchronization" che consente di importare gli utenti e tutte le informazioni loro correlate da una user directory. Nel momento in cui si crea una nuova istanza di UDC (User Directory Connector), infatti, il sistema crea automaticamente un task di sincronizzazione con un trigger schedulato.

Name	Associated resource	Type	Enabled	Status
01 - DWH - ETL01	EstrDWH_001.01_ALL_ETL01	Reload	Yes	Success
01 - DWH - ETL02	EstrDWH_001.01_ALL_ETL02	Reload	Yes	Success
101 - ETL01 - BiGRP	BiGRP_101.02_ALL_ETL01	Reload	Yes	Success
101 - ETL02 - BiGRP	BiGRP_101.02_ALL_ETL02	Reload	Yes	Success
Connessione LDAP RP NTSI...	Connessione LDAP RP NTSI...	User synchronization	Yes	Success
Reload License Monitor	License Monitor	Reload	Yes	Success
Reload Operations Monitor	Operations Monitor	Reload	Yes	Success

Figura 3.7: Tasks settati sulla QMC

Nei prossimi paragrafi scenderemo verticalmente sulla conformazione degli ETL 01 e 02, analizzandone le caratteristiche e descrivendo con abbondanza di dettagli sia la struttura delle tabelle di settings e come queste vanno correttamente riempite che i template col codice Qlik Sense.

3.3 ETL 01 – Estrattore tabelle

Il principale obiettivo dell'ETL 01 è quello di tirare su i dati dalle fonti consentendo, nel caso dell'estrattore centralizzato, di non sovraccaricare i sistemi sorgente nella fase di scarico informazioni mentre, nel caso dell'estrattore di app, di prelevare solo i dati di

interesse per quel determinato progetto. L'ETL 01 è quindi finalizzato a gestire la fase di staging e mira sia a semplificare che a monitorare il processo di estrazione dei dati e di aggiunta di nuove fonti. Inoltre agevola la manutenzione e migliora le performance delle applicazioni (fornendo accesso ai QVD anziché accesso diretto al DB). Pertanto ciò che deve mettere a disposizione il nostro ETL 01 custom è un modo semplice e inequivocabile di:

- stabilire le connessioni con i sistemi sorgente;
- definire le tabelle da estrarre e le relative specifiche di importazione;
- applicare clausole di where sulle singole tabelle;
- specificare i campi da estrarre ed indicare eventuali campi calcolati;
- assegnare alias ai campi;
- dare la possibilità di storicizzare mensilmente o annualmente le tabelle;
- consentire delle estrazioni custom per i casi particolari (es. query SQL);
- impostare delle variabili globali di appoggio;
- monitorare le estrazioni (cardinalità, tempi, tabelle estratte);
- memorizzare in formato QVD i record sets caricati dalle diverse fonti dati (DWH, database relazionali, fonti esterne).

Per realizzare ciò ci si appoggia su opportune tabelle di settings che, debitamente compilate, consentono di andare a specificare tutti le configurazioni necessarie. In particolare, si utilizzano cinque tabelle: ETL_GLOBAL_VARIABLES, ETL_01_CONNECTION, ETL_01_TABLETOEXTRACT, ETL_01_QUERY, ETL_01_VARIABLES. Il campo PROJECTID è comune a tutte le tabelle ed è fondamentale in quanto consente di individuare univocamente un ETL. L'ETL centralizzato ha PROJECTID uguale a 1, mentre, per convenzione, gli ETL di app hanno un PROJECTID compreso tra 101 e 999 per ciò che è in produzione, superiore a 1000 per ciò che deve essere ancora testato.

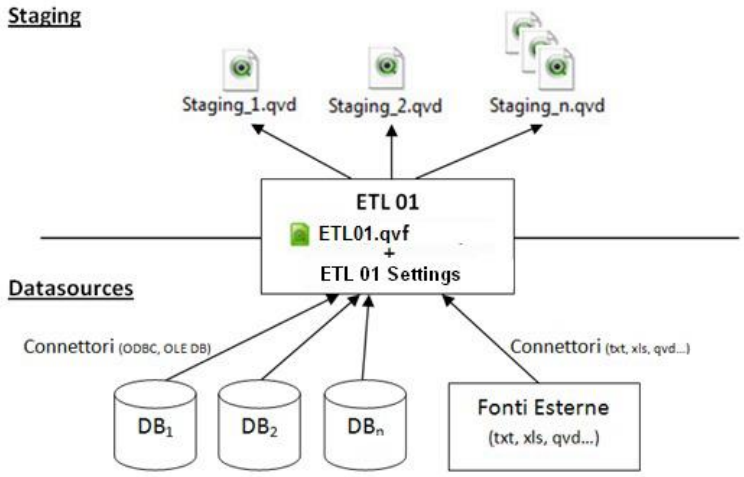

Figura 3.8: Schema ETL 01

Scendiamo adesso nelle peculiarità delle singole tabelle spiegandone utilità e struttura.

La tabella ETL_GLOBAL_VARIABLES permette di parametrizzare l'esecuzione degli ETL. Tali parametri o variabili possono essere utilizzati dagli ETL 01 ma anche da ETL di secondo livello (ETL 02) centralizzando la gestione dei parametri comuni. Di default devono essere presenti le variabili relative ai paths (caricamento delle data sources, memorizzazione dei QVD) e al monitoraggio. Oltre al PROJECTID, i campi settabili sono:

- USAGE_TYPE: indica il livello di ETL che utilizza la variabile. I valori possibili sono ETL01, ETL02, PRES (presentazione, cioè il livello applicativo) o una combinazione di essi;
- VARIABLE_NAME: specifica il nome della variabile nello script. Per convenzione tutte le variabili devono avere il prefisso "vs_" e vengono cancellate a fine reload, quindi esistono limitatamente alla durata dello script. Se se ne vuole prolungare il ciclo di vita occorre crearne una copia col prefisso "v_";
- VARIABLE_VALUE: identifica il valore da assegnare alla variabile;
- DESCRIPTION: è un campo descrittivo.

Per il corretto funzionamento dello script di ETL 01 è necessario che nella tabella ETL_GLOBAL_VARIABLES siano presenti le seguenti variabili di default:

- vs_Prefix_Name_Staging: il valore deve essere settato a "St_" e rappresenta il prefisso usato per le tabelle storate dall'ETL 01 nell'area di staging. I QVD saranno registrati con il nome [vs_Prefix_Name_Staging][nometabella].qvd;
- vs_Path_QVD_DwStaging: indica il path in cui saranno memorizzati i QVD elementari (quelli di staging);
- vs_Path_QVD_Sistema_ETL01: è la cartella in cui si andranno a salvare i QVD di sistema (log, monitoraggio) dell'ETL 01;
- vs_Path_ExternalData: è la directory in cui andare a prelevare i file relativi alle fonti dati esterne;
- vs_SelectAmbiente: consente di scegliere l'ambiente da cui estrarre i sorgenti. I valori possibili sono: Sviluppo, Test e Produzione;
- vs_LoadingType: può assumere i valori "FULL" (completo) o "INCR" (incrementale) e delinea il tipo di caricamento;
- vs_NumOfScriptLogStored: indica il numero di script log generati dall'ETL, storati e visualizzati nella presentation dell'ETL (il valore di default è 10);
- vs_NumOfMonitoringLogStored: indica il numero di monitoring log generati dall'ETL, storati e visualizzati nella presentation dell'ETL (il valore di default è 200).

Ovviamente è necessario che ci sia coerenza tra la struttura delle directories e i paths specificati nelle variabili globali.

ETL_01_CONNECTION consente di impostare le connessioni ai database relazionali (tramite stringa di connessione) e/o alle fonti esterne (specificando il path). Di seguito i

campi costituenti lo schema di questa tabella:

- ENVIRONMENT: individua l'ambiente a cui ci si collega. Può assumere uno dei seguenti valori: Sviluppo, Test, Produzione. Le connessioni considerate in fase di reload saranno tutte e sole quelle che avranno valore corrispondente a quello della variabile globale "vs_SelectAmbiente" presente in ETL_GLOBAL_VARIABLES;
- STRINGSOURCE: consente di indirizzare la lettura sul connettore corretto;
- TYPEDB: identifica la tipologia di data source (relazionale, esterna, ODS (Operational Data Store));
- CONNECTIONSTRING: è la stringa di connessione al DB. È un campo necessario se la sorgente è di tipo relazionale;
- SUBPATHQVD: specifica una sotto-cartella dove memorizzare i QVD di staging. Se il campo non è valorizzato i QVD saranno memorizzati a partire da quanto indicato nella variabile globale "vs_Path_QVD_DwStaging";
- SUBPATHEXTERNALSOURCE: come sopra ma con riferimento alle fonti esterne;
- TOLOAD: è un flag (valori possibili: 'Y' o 'N') che consente di specificare se la connessione è da effettuare o meno. È utile per una manutenzione più rapida;
- DESCRIPTION: è un campo descrittivo.

ETL_01_TABLETOEXTRACT contiene l'elenco delle tabelle da estrarre per ogni fonte dati specificando i parametri di eventuali clausole where, il campo sul quale eseguire l'estrazione incrementale, i parametri per la storicizzazione in QVD e le specifiche di importazione (solo per fonti esterne). In questo caso i campi configurabili sono tanti ma, a seconda dei casi, non sono tutti da compilare. Eccoli di seguito:

- SOURCE: identifica la sorgente e indirizza la lettura sul connettore corretto. Deve esserci corrispondenza tra i valori immessi in questo campo e quelli presenti alla voce STRINGSOURCE delle tabella ETL_01_CONNECTION;
- PHYSICAL_NAME: individua la sorgente. È il nome della tabella se la sorgente è di tipo relazionale o il nome del file compreso di estensione se la fonte è esterna;
- TABLE_NAME: è il campo contenente il nome per il salvataggio in QVD. Di default i file QVD sono memorizzati come [vs_Prefix_Name_Staging][nome tabella].qvd;
- DESCRIPTION: è un campo descrittivo;
- TO_EXTRACT_FULL: è un flag (valori possibili: 'Y' o 'N') utile ad indicare le tabelle che devono essere estratte quando la modalità di caricamento è "FULL";
- TO_EXTRACT_INCR: come sopra ma per estrazione incrementale;
- SPECIFICIMP: consente di indicare le specifiche di importazione per le fonti esterne (TXT, XLS, QVD…). Ad esempio nel caso di un file Excel occorre esplicitare quale foglio andare a prendere;
- TO_FREEZE: è un flag (valori possibili: 'Y' o 'N') che permette di precisare se

la tabella è da storicizzare. Storicizzando verranno memorizzati tanti QVD quanti sono i periodi presi in considerazione;

- FREEZING_TYPE: indica se la storicizzazione deve essere eseguita per anno o per mese;
- FREEZING_FIELD: specifica il campo di storicizzazione: conterrà una data (per storicizzazione mensile) o un anno (per storicizzazione annuale);
- FREEZED_FROM: identifica la data (storicizzazione mensile) o l'anno (storicizzazione annuale) da cui iniziare a storicizzare la tabella;
- FREEZED_TO: come sopra ma per segnare il punto di fine della storicizzazione;
- WHERE_CLAUSE: è un flag (valori possibili: 'Y' o 'N') che segnala se occorre processare una clausola di filtro sui dati sorgente;
- WHERE_ON_SELECT_OR_LOAD: indica se la clausola where deve essere applicata nella SELECT oppure nella LOAD;
- CLAUSE: è il codice Qlik relativo alla clausola where;
- NOTE: è un campo disponibile per annotazioni varie;
- LASTUPDATEFIELD: identifica il campo che nella tabella specifica tramite una data l'inserimento di nuovi records. È utilizzato per il caricamento incrementale;
- LASTCOUNTFIELD: consente di individuare il campo contenente il contatore di tipo numerico da utilizzare ai fini del caricamento incrementale.

La tabella ETL_01_QUERY permette di gestire i campi estratti da ogni tabella. In particolare, permette di decidere quali campi estrarre, assegnare alias e aggiungere nuovi campi calcolati. L'uso di questa tabella è facoltativo in quanto se una tabella è presente in ETL_01_TABLETOEXTRACT e non viene richiamata in ETL_01_QUERY lo script esegue una "LOAD *" importando tutti i campi. Lo schema di ETL_01_QUERY è il seguente:

- TABLE_NAME: identifica la tabella e corrisponde al campo omonimo di ETL_01_TABLETOEXTRACT;
- TOREAD: è un flag (valori possibili: 'Y' o 'N') che consente di specificare se il campo è da estrarre o meno. È utile per una manutenzione più rapida;
- FIELD: è il nome del campo;
- ALIAS: se valorizzato, il nome campo sarà sostituito da quello dell'alias qui inserito;
- TOPRECLOAD: è un flag (valori possibili: 'Y' o 'N') che indica se occorre applicare al campo una funzione in fase di Preceding Load;
- FUNCTION_NAME: se TOPRECLOAD='Y', qui bisogna dettagliare la funzione da applicare al campo nella Preceding Load;
- ALIASPRECLOAD: se valorizzato, si attribuisce un alias al risultato della funzione inserita nel campo precedente;
- NOTE: è un campo disponibile per prendere appunti;

Infine ETL_01_VARIABLES contiene le variabili proprie del singolo ETL o eccezioni rispetto alle variabili globali. Inserendo una variabile già presente nella tabella ETL_GLOBAL_VARIABLES questa assumerà il nuovo valore per lo specifico ETL 01 sovrascrivendo quello impostato in ETL_GLOBAL_VARIABLES. Sono solo tre i campi costituenti questa tabella:

- VARIABLE_NAME: specifica il nome della variabile nello script. Come già detto parlando della tabella ETL_GLOBAL_VARIABLES, per convenzione tutte le variabili devono avere il prefisso "vs_" e vengono cancellate a fine reload, quindi esistono limitatamente alla durata dello script. Se se ne vuole prolungare il ciclo di vita occorre crearne una copia col prefisso "v_";
- VARIABLE_VALUE: identifica il valore da assegnare alla variabile;
- DESCRIPTION: è un campo descrittivo.

Il template di ETL 01, leggendo quanto impostato in queste tabelle, andrà a popolare lo script dinamico di caricamento dati e fornirà in output i QVD. Nello specifico, il QVF di ETL 01 è costituito da più sezioni tematiche. La prima di queste è il "Main" ed è utile a settare le variabili di sistema e il PROJECTID. Tutte le altre sezioni contengono definizioni di routine, tranne "Controllo" che invece le richiama. Qui infatti è contenuto il flusso di esecuzione dello script.

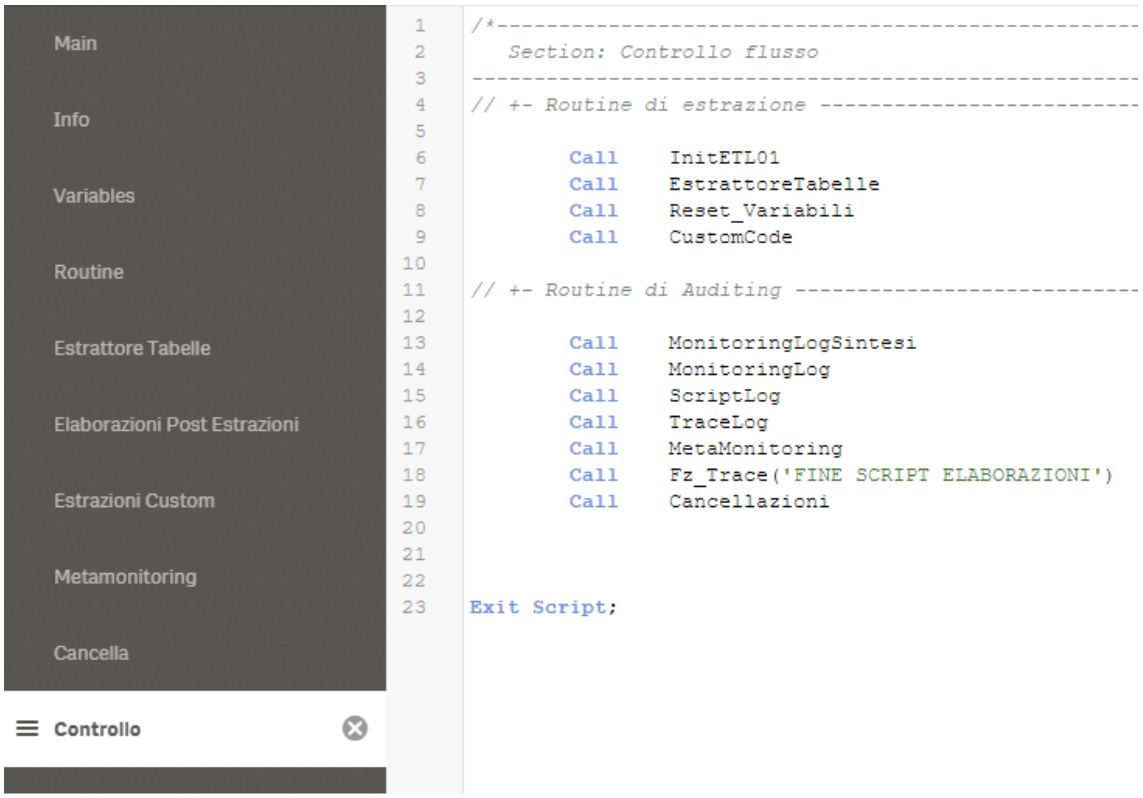

Figura 3.9: Sezione "Controllo" dello script di ETL 01

Come è possibile vedere dalla figura 3.9, inizialmente viene richiamata la routine "InitETL01" che non fa altro che memorizzare data e ora di inizio elaborazione e andare a ricavare dallo storico di log il numero di estrazione corrente. Inoltre tramite la routine "LoadVariabili" vengono settate le variabili caricate in ETL_01_VARIABLES ed ETL_GLOBAL_VARIABLES associate al PROJECTID attuale.

Successivamente si esegue "EstrattoreTabelle", cuore centrale del programma. Il primo passo è la lettura della tabella ETL_01_CONNECTION. Per ogni riga di connessione si esegue la routine di estrazione. A seconda di quanto impostato nella variabile globale "vs_LoadingType" si andranno a leggere dalla tabella ETL_01_TABLETOEXTRACT tutte e sole le righe aventi il flag ad 'Y' nel campo TO_EXTRACT_FULL o TO_EXTRACT_INCR. Prima di procedere al caricamento vero e proprio delle tabelle occorre però gestire eventuali estrazioni incrementali o richieste di storicizzazione. Per fare questo sono state definite due routine chiamate rispettivamente "Incrementale" e "Storicizzazione_Init". Attraverso la prima si valorizzano le variabili utili ad eseguire il caricamento incrementale. Scendendo nel dettaglio, per ogni tabella presa in considerazione si vanno ad ispezionare i campi LASTUPDATEFIELD o LASTCOUNTFIELD per controllare se sono popolati e dal QVD di log si estraggono l'ultima data o l'ultimo contatore valorizzato per la tabella in osservazione. In questo modo si andrà a generare una variabile che sarà utilizzata nella clausola di where e specificherà di caricare solo le righe per cui il valore di LASTUPDATEFIELD o LASTCOUNTFIELD è maggiore dell'ultima data o contatore valorizzato. La routine "Storicizzazione_Init" invece consente di settare correttamente le variabili utili a gestire il ciclo di storicizzazione. Queste sono necessarie dal momento che lo step successivo è un ciclo for che compirà tanti giri quanti sono i periodi di storicizzazione (uno solo se non bisogna storicizzare). Dentro il ciclo c'è il codice per effettuare il caricamento vero e proprio. Innanzitutto si inizializza la variabile contente la condizione di where con cui si precisa di estrarre solo ciò che riguarda il periodo che si vuole storicizzare in quel QVD. Poi, leggendo da ETL_01_QUERY, per ogni tabella considerata si tirano su tutti i campi desiderati gestendo opportunamente i casi in cui è presente un alias. Infine, attraverso la routine "Generazione_Codice", si va a creare la condizione di where definitiva andando a concatenare tutte le variabili predisposte in precedenza e, a seconda del tipo di connessione, se relazionale o esterna, si imposta la clausola di from. Questa può essere costituita semplicemente dal nome tabella o, alternativamente, dalla concatenazione del path che indirizza nella directory delle fonti esterne con il nome della sorgente e relativa specifica di importazione. Fatto anche quest'ultimo passaggio può essere chiamata la routine "Esecuzione_Codice" deputata all'effettivo caricamento. Il salvataggio in QVD avviene nella fase seguente. La funzione incaricata a fare ciò è la "Fz_StoreTable" che prende come parametro in ingresso il nome col quale si sceglie di memorizzare il QVD. Qui, prima di tutto, se l'estrazione è di tipo incrementale si concatena lo storico. Poi si fa lo store al percorso indicato dalla variabile "vs_Path_QVD_DwStaging" a cui si concatena un eventuale sub-path nel caso in cui il campo SUBPATHQVD della connessione considerata sia valorizzato. Insieme a ciò si tiene traccia dell'estrazione memorizzando in opportune variabili i dati utili al monitoraggio (numero di campi e records caricati, tempo e durata di estrazione…). Una volta che è stato generato il QVD si fa il drop della tabella dal momento che non è più necessario tenerla in memoria.

Terminato il caricamento si resettano le variabili per far sì che tutto torni come al

principio, pronto per il prossimo reload.

Le estrazioni che si possono realizzare tramite framework sono tutte e sole quelle che hanno un codice del tipo "LOAD … FROM" o "SQL SELECT … FROM". Ci sono però circostanze particolari che non si adattano a questa soluzione. È il caso di estrazioni tramite SAP OLAP Connector, estrazioni estemporanee da queries con sintassi SQL, cicli di estrazione su più fonti esterne. Per questo tipo di estrazioni non è prevista la compilazione della tabella ETL_01_TABLETOEXTRACT ma è presente una sezione all'interno del template di ETL 01 nel tab "Estrazioni Custom". Qui è definita una routine denominata "CustomCode" che viene richiamata dopo il reset delle variabili.

```
Sub CustomCode
    Call Fz_Trace('ESTRAZIONI CUSTOM')


    //Esempio esecuzione di codice di estrazione CUSTOM:
    //Chiamata a connessione
    call Fz_ConnessioneCustom ('Mdb')   //'Mdb' è una connessione parametrizzata

    //Codice di estrazione custom
    CustomTable:
    SQL SELECT
        *
    FROM `Test_Table_2`;

    //Store e drop della tabella con funzioni di auditing
    Call Fz_StoreTable('CustomTable')   //'CustomTable'è il nome della tabella

End Sub
```

Figura 3.10: Sub-routine "CustomCode"

Sopra è riportato il template della sezione custom. Ci sono sostanzialmente tre blocchi: il primo, tramite la chiamata a "Fz_ConnessioneCustom", consente di impostare la connessione; il secondo è quello dove occorre inserire il codice custom; il terzo è utile a realizzare lo store della tabella. Prevedere una sezione custom nel nostro framework consente quindi di rispondere ad un'esigenza concreta e, contemporaneamente, permette di mantenere due vantaggi importanti: la parametrizzazione delle connessioni tramite l'uso della tabella ETL_01_CONNECTION; il mantenimento della funzione di auditing e di monitoring.

Le altre routine richiamate in sequenza mirano a realizzare i QVD di log e monitoraggio che saranno memorizzati al path specificato dalla variabile globale "vs_Path_QVD_Sistema_ETL01".

L'output del template di ETL 01 non è costituito solo dai QVD contenenti i dati caricati ma pure da un foglio che esplicita i dettagli di log. Nello specifico questo front-end, chiamato "Monitoring Estrazioni", prende quanto memorizzato nei QVD di log e, utilizzando gli strumenti di data visualization messi a disposizione da Qlik, mostra in maniera figurata i dati di log consentendo un più agile monitoraggio.

Quello che viene presentato è:

1. L'informazione su data e ora degli ultimi reloads con la possibilità di selezionare l'estrazione da analizzare.
2. L'indicazione del numero di estrazione tramite l'uso del contatore progressivo, incrementato per ogni elaborazione ETL 01 avvenuta.
3. Un line chart che mostra le cardinalità delle tabelle estratte. La rappresentazione grafica permette di incrociare le numerosità dei vari oggetti e avere evidenza delle variazioni all'avanzare delle estrazioni.
4. Una tabellina rappresentante il numero di records estratti per ogni oggetto, confrontato con lo stesso dato per l'estrazione precedente. Tramite opportuna colorazione è possibile segnalare delle anomalie: in rosso se nella tabella sono presenti meno records rispetto all'ultima estrazione; in verde se nella tabella sono presenti più records rispetto all'ultima estrazione; in bianco se nella tabella è presente lo stesso numero di records rispetto all'ultima estrazione.
5. Una tabella di riepilogo che sintetizza, per estrazione, la situazione delle tabelle tirate su. Nel dettaglio sono riportati: il nome della tabella originaria; il path di store (cartella di destinazione); il nome del QVD memorizzato; il numero di righe (records) e colonne (campi) estratti; data e ora di store (quindi data e ora di creazione del QVD); durata dell'estrazione.

In basso è riportato uno screen del foglio "Monitoring Estrazioni" che illustra quanto spiegato.

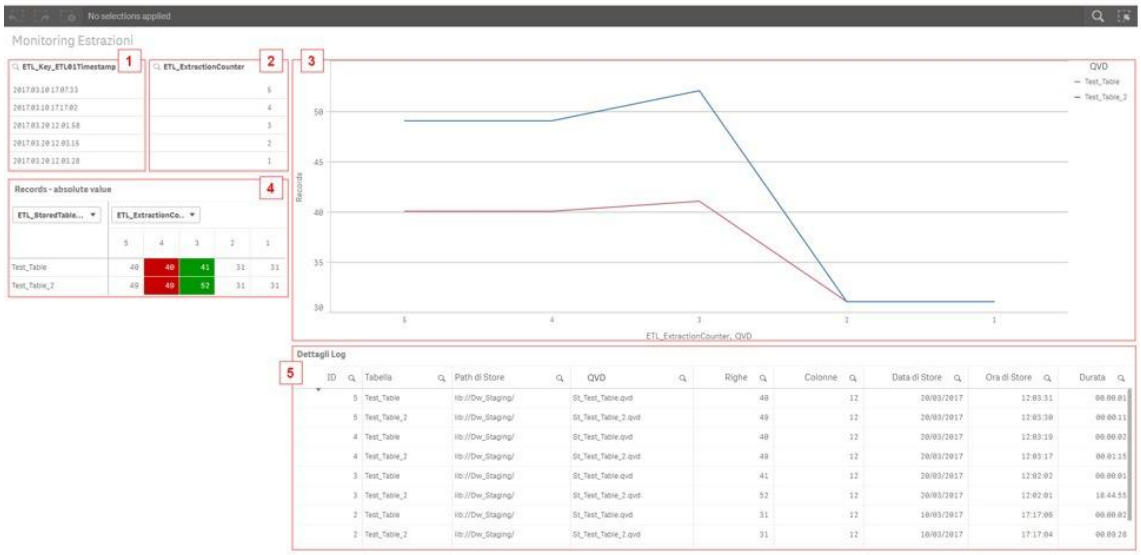

Figura 3.11: Foglio "Monitoring Estrazioni"

3.4 ETL 02 – Normalizzazione e creazione del modello associativo

L'ETL 02 ha l'obiettivo di gestire l'elaborazione, la trasformazione e la modellazione di quanto tirato su dallo stadio precedente. Mentre l'ETL 02 centralizzato è un livello opzionale in quanto, se non ci sono esigenze particolari di modifiche contestuali a più progetti, può essere saltato, l'ETL 02 di app è una fase fondamentale poiché è qui che si va a predisporre il data model dell'applicazione. Inoltre proprio nell'ETL 02 di app avviene la rinomina automatica dei campi secondo la definizione riconosciuta dall'utenza. Il nostro ETL 02 custom si prefigge quindi di semplificare la fase di definizione del data model, la sua manutenzione e di offrire supporto al monitoraggio. In sintesi consente di:

- impostare variabili globali (variabili utilizzate da più ETL di primo o secondo livello);
- modificare il tracciato record di un'entità proveniente dallo staging, scegliendo quali campi caricare e le eventuali chiavi da creare;
- mappare e definire alias per tutti i campi degli oggetti coinvolti nel data model;
- aggiungere campi calcolati alle tabelle caricate dall'area di staging;
- eseguire aggregazioni, join o mapping all'interno delle entità predisposte in una sezione specifica dello script denominata "Custom";
- memorizzare in formato QVD i record sets;
- monitorare la cardinalità del data model successivamente ad ogni caricamento.

Così come l'ETL 01, anche l'ETL 02 è sostanzialmente un file QVF, cioè un'applicazione Qlik Sense che, poggiandosi a delle tabelle di settings, consente di generare dinamicamente lo script utile a gestire la fase di manipolazione del dato e costruzione del modello. Le tabelle di supporto utilizzate dall'ETL 02 sono: ETL_GLOBAL_VARIABLES, ETL02_INDEX, ETL02_TABLES, QLIK_CATALOGO_INFORMAZIONI. Mentre le prime tre sono accomunate dall'avere il campo PROJECTID che identifica il progetto cui fanno riferimento (1, se è l'estrattore centralizzato, tra 101 e 999 se è un'app in produzione, maggiore di 1000 se è un'app da collaudare), la QLIK_CATALOGO_INFORMAZIONI è una struttura universale quindi non caratterizzante un determinato progetto.

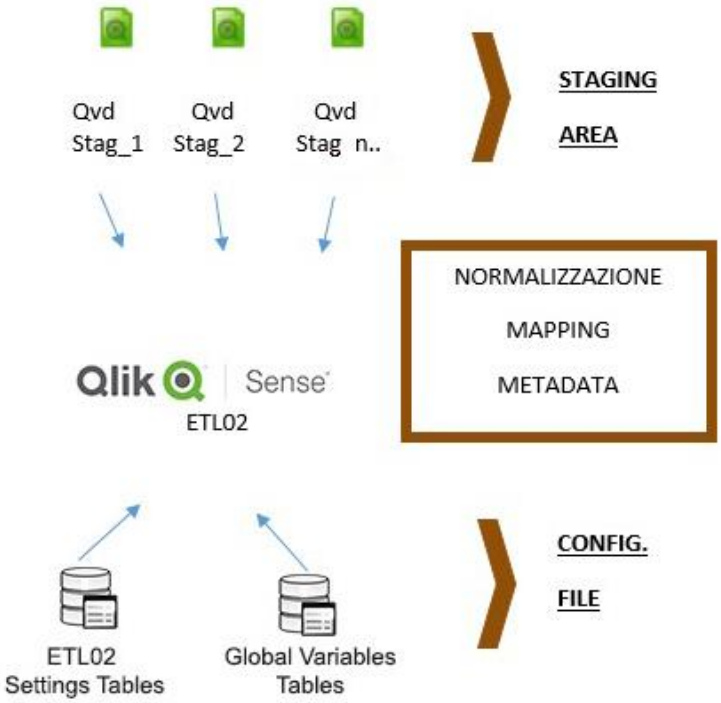

Figura 3.12: Schema ETL 02

Analizziamo adesso la conformazione di ogni tabella per carpirne le utilità.

La tabella ETL_GLOBAL_VARIABLES è stata già descritta nel paragrafo precedente per cui si rimanda alle pagine passate per conoscerne le peculiarità. Affinché lo script di ETL 02 funzioni regolarmente è necessario che in questa tabella siano settate le seguenti variabili di default:

- vs_Prefix_Name_Staging: il valore deve essere impostato a "St_" e rappresenta il prefisso identificante i QVD prelevabili dall'area di staging;
- vs_Prefix_Name_DwAssociativo: deve avere valore "Dw_" e delinea il prefisso usato per le tabelle che saranno storate dall'ETL 02 o nella cartella *Common* se il PROJECTID è 1, o nella directory *Dw_Associative* propria del progetto che si sta sviluppando in tutti gli altri casi. I QVD saranno memorizzati con il nome [vs_Prefix_Name_DwAssociativo][nometabella].qvd;
- vs_Path_QVD_DwStaging: identifica il path da cui andare ad estrarre i QVD di staging;
- vs_Path_QVD_DwAssociativo: indica il percorso in cui saranno memorizzati i QVD risultanti (quelli post-elaborazione);
- vs_Flag_App: è un flag (valori possibili: 0 o 1) utile a specificare se occorre compiere l'operazione di renaming. Assume valore 0 se non si vuole applicare nessuna modifica ai nomi degli attributi (è il caso dell'ETL 02 sorgente), 1 se invece si preferisce effettuare la ridenominazione;

- vs_Hide_Prefix: tutti i nomi di campo che iniziano con questa stringa di testo verranno nascosti alla stessa maniera di tutti i system fields;
- vs_Path_QVD_Sistema_ETL02: è la cartella in cui si andranno a salvare i QVD di sistema (log, monitoraggio) dell'ETL 02;
- vs_NumOfScriptLogStored: indica il numero di script log generati dall'ETL, storati e visualizzati nella presentation dell'ETL (il valore di default è 10);
- vs_NumOfMonitoringLogStored: indica il numero di monitoring log generati dall'ETL, storati e visualizzati nella presentation dell'ETL (il valore di default è 200).

La tabella ETL02_INDEX permette di impostare la lista di entità da caricare dall'area di staging, includendo eventuali clausole di caricamento o note. I campi che la costituiscono sono:

- TABLE_NAME_VL: identifica il nome con cui verrà salvato il file di output derivante da questa fase di normalizzazione. Ogni tabella conterrà il prefisso "Dw_" e avrà quindi questa struttura: [vs_Prefix_Name_DwAssociativo][nome tabella].qvd;
- QVD_STAGING: specifica quale sia il file QVD dell'area di staging che viene elaborato (attenzione a riportare il nome completo compreso di prefisso);
- TABLE_TOLOAD: consente di decidere quale tabella in elenco includere/escludere in questo frangente. Questo campo pertanto è utilizzato per una manutenzione più celere ma non solo. Infatti, oltre a poter contenere 'N' tramite cui si esclude la relativa tabella dall'elaborazione ETL 02, può assumere anche i valori 'Y' o 'R'. Con 'Y' si processeranno i campi come definiti in ETL02_TABLES. Con il valore R ("Raw" mode) la tabella di staging verrà letta senza alcuna elaborazione e salvata in formato QVD così com'è; l'unica clausola considerata sarà la where;
- DESCRIPTION: è un campo descrittivo;
- WHERE_CONDITION: è un flag (valori possibili: 'Y' o 'N') che, se impostato, consente di attivare una condizione di where;
- CLAUSE: se il precedente campo assume valore 'Y', qui occorre riportare la clausola di where;
- NOTE: spazio disponibile per appunti vari.

La tabella ETL02_TABLES contiene il tracciato record della relativa tabella elencata in ETL02_INDEX e una serie di colonne che consentono di fare aliasing, elaborazioni sui campi, definizione di chiavi (anche composte). In questo modo si determina la struttura dei file di output. Nello specifico questa tabella ha la seguente struttura:

- TABLE_NAME: contiene il riferimento alla tabella inserita come TABLE_NAME_VL in ETL02_INDEX;
- QVD_FIELD: identifica il nome del campo all'interno del QVD di primo livello;
- VL_FIELD: precisa il nome con cui viene ridefinito il campo originale al termine dell'elaborazione di secondo livello;

- FIELD_TOLOAD: è un flag (valori possibili: ‘Y’ o ‘N’) che consente di specificare se il campo è da considerare o meno. È utile per una manutenzione più rapida;
- DESCRIPTION: è un campo descrittivo;
- KEY_FIELD: è un flag (valori possibili: ‘Y’ o ‘N’) che definisce se un campo deve essere chiave tra due o più entità;
- ORDER_COMPOSITEKEY: determina l’ordine posizionale con il quale il campo viene inserito in una chiave composta (l’indicizzazione parte da 1);
- KEY_NAME: rappresenta il nome che viene assegnato alla chiave all’interno del modello associativo.

Possono essere fatte elaborazioni in fase di caricamento su ogni singolo campo. In tal caso vengono accettate tutte le funzioni messe a disposizione da Qlik. In conclusione quindi un campo può essere caricato in diversi modi; di seguito vengono riportati alcuni esempi delle possibili combinazioni:

- lettura di un campo dal QVD di staging ed assegnazione di un alias: FIELD_TOLOAD=’Y’ e VL_FIELD valorizzato con l’alias desiderato;
- lettura di un campo dal QVD di staging ed applicazione di una trasformazione: come sopra e funzione di elaborazione inserita nel campo QVD_FIELD;
- lettura di un campo anche come chiave tra due entità: FIELD_TOLOAD=’Y’, VL_FIELD valorizzato con l’alias prescelto, flag KEY_FIELD=’Y’, ORDER_COMPOSITEKEY settato con un valore tra 1 ed N, KEY_NAME contenente il nome chiave voluto;
- lettura di un campo solamente come chiave: FIELD_TOLOAD=’N’, flag KEY_FIELD=’Y’ e i campi ORDER_COMPOSITEKEY e KEY_NAME configurati come già discusso sopra.

Ovviamente poi occorre che vi sia coerenza fra il tracciato record dei QVD di primo livello e l’elenco di campi riportati nelle tabelle ETL02_INDEX ed ETL02_TABLES.

La QLIK_CATALOGO_INFORMAZIONI è una tabella globale in cui sono censite le informazioni del DWH Rai Pubblicità. Possiamo quindi definirla come un data dictionary. È la sorgente per l’operazione di renaming ed è costituita dai seguenti campi:

- CODICE: è la codifica tecnica utilizzata nelle varie fasi dell’ETL;
- DESCRIZIONE_UTENTE: conterrà il valore da esporre a front-end;
- FLAG_ATTIVO: è un flag (valori possibili: ‘Y’ o ‘N’) che specifica se il campo è da rinominare;
- FLAG_VISIBILITA_UTENTE: altro flag (valori possibili: ‘Y’ o ‘N’) che indica se il campo deve essere nascosto (‘N’) o se è invece visibile nell’applicazione (‘Y’).

Nello script non si ricorrerà direttamente a questa tabella ma ad una vista, la VMP_QLIK_CATALOGO_INFORMAZIONI, che tira su solo i campi della QLIK_CATALOGO_INFORMAZIONI con FLAG_ATTIVO=’Y’, cioè i campi da

rinominare. Si fa ciò per disaccoppiare la consultazione del patrimonio informativo aziendale dalla pratica che se ne fa in Qlik.

Andando ad approfondire il template si può facilmente notare che, come per l'ETL 01, ci sono diverse sezioni. La prima è sempre l'area denominata "Main" in cui si setta il PROJECTID e altre informazioni basilari di sistema, mentre la più importante è ancora la sezione "Control" contenente il flusso esecutivo di chiamata a tutte le sub-routine. Tra quest'ultime la prima che viene invocata è la "InitETL02" che, come il suo corrispettivo nell'ETL di primo livello, registra data e ora di inizio elaborazione, determina il numero di estrazione corrente e, tramite la routine "LoadVariabili", imposta le variabili associate al PROJECTID corrente. Si configurano poi due variabili di controllo atte a specificare di fare lo store in QVD delle tabelle prodotte e il drop delle stesse dalla nuvola dati.

Comincia da qui lo script di elaborazione. Se la variabile "vs_Flag_App" ha valore pari ad 1 significa che si vuole applicare la rinomina dei campi. Pertanto viene richiamata la routine "CatalogoInformazioni" che non fa altro che andare a leggere dalla VMP_QLIK_CATALOGO_INFORMAZIONI codice e descrizione utente degli attributi censiti e generare contestualmente la tabella di mapping. Nello specifico, se il campo FLAG_VISIBILITA_UTENTE è settato ad 'N', alla descrizione utente si prepone l'hide prefix cioè la stringa di testo che, posta all'inizio di un nome campo, denota che l'attributo verrà nascosto. Dopo questo passaggio preliminare si richiama la sub-routine "Mapping_Table", funzione deputata a caricare ed elaborare i campi contenuti nei QVD di staging prescelti. Il primo passaggio da compiere è l'individuazione dei QVD da estrarre. Ciò si realizza andando a leggere i records della ETL02_INDEX che presentano il campo TABLE_TOLOAD configurato ad 'Y'. Per ogni tabella qui indicata occorre andare a estrapolare gli attributi specificati in ETL02_TABLES e operare le dovute trasformazioni. Si comincia dal predisporre i campi chiave, cioè quelli per cui KEY_FIELD ha valore 'Y'. In ETL02_TABLES ci saranno tanti valori distinti per l'attributo KEY_NAME quante saranno le chiavi che si vogliono generare. Per ognuna di queste si conta il numero di occorrenze in modo da determinare quanti campi la andranno a comporre e, se questo numero è maggiore di uno, dopo aver ordinato per il campo ORDER_COMPOSITEKEY, si concatenano i corrispettivi valori di QVD_FIELD separati dal carattere '|' andando a costituire la chiave concatenata. Lo step successivo è l'individuazione di tutti gli altri campi da caricare ed elaborare. In questo caso occorre solamente leggere dalla ETL02_TABLES i records con flag FIELD_TOLOAD ad 'Y'. Le trasformazioni da applicare saranno già incluse nel campo QVD_FIELD. Per completare il giro riguardante ogni tabella, nel caso sia stata configurata una where condition è necessario anche andare ad impostare una variabile contenente tale clausola. Tutto è ora pronto per la realizzazione dei QVD finali: vengono tirate su dall'appropriata tabella nell'area di staging sia le chiavi, costruite come spiegato precedentemente, che tutti i campi desiderati applicando, se necessario, la condizione di where. Prima di andare a storare la tabella tramite la funzione "Fz_StoreTable" occorre procedere però alla rinomina dei campi. Si invoca allora la sub-routine "FieldsHidePrefix" che sostanzialmente annette l'hide prefix a tutti campi. Solo successivamente si concretizza la ridenominazione sfruttando la tabella di mapping creata in precedenza a partire dal catalogo informativo. Si opera così in maniera tale che tutto ciò che non è censito nel data dictionary non è visibile all'utenza finale. Di norma infatti se un campo non è stato ancora registrato è perché è un attributo in manutenzione e dunque è corretto nasconderlo. Per

quanto riguarda invece le righe della tabella ETL02_INDEX che presentano valore di TABLE_TOLOAD a 'R', la procedura è la stessa di quella descritta poc'anzi ma semplificata in quanto bisogna prelevare tutti i records del QVD indicato senza applicare trasformazioni o definire chiavi.

Conclusa questa fase viene richiamata la sezione "Custom". Quest'area si presenta vuota in quanto è a totale disposizione del developer e consente di gestire tutti i casi spinosi. In questo modo si può liberamente scrivere qui il codice per realizzare join, aggregazioni, trasformazioni da eseguire in più passi. Solitamente questa sezione la si organizza pensandola come il punto di avvio delle sub-routine definite per la circostanza così da strutturare il tutto in maniera lineare e comprensibile favorendo le operazioni di manutenzione.

Infine, come avveniva per l'ETL 01, vengono richiamate una serie di sub-routine in cascata che puntano a costituire i QVD di log e monitoraggio che saranno salvati al percorso identificato dalla variabile globale "vs_Path_QVD_Sistema_ETL02".

Il risultato dei log viene utilizzato per generare un foglio denominato "Principal" che consente, dopo ogni caricamento, un controllo a valle dell'elaborazione. Come è visibile dalla figura sottostante, questo sheet è strutturato in quattro macro-zone:

1. Contiene un prospetto riportante le date di produzione dei QVD.
2. Come sopra ma con i timestamp.
3. È costituita da un grafico lineare mostrante al variare dell'extraction counter il numero di righe e colonne dei QVD generati.
4. Presenta un quadro che rivela per ogni elaborazione le cardinalità delle tabelle estratte. Tramite opportuna colorazione si evidenziano possibili anomalie:
    - ROSSO: nella tabella sono presenti meno records rispetto all'ultima estrazione;
    - VERDE: nella tabella sono presenti più records rispetto all'ultima estrazione;
    - BIANCO: nella tabella è presente lo stesso numero di records rispetto all'ultima estrazione.

Figura 3.13: Foglio "Principal"

3.5 Template di app

A questo punto è stato predisposto l'intero flusso informativo che ha condotto dai dati puri presenti sulle sorgenti alla loro versione elaborata e strutturata pronta a costituire l'associative model del progetto.

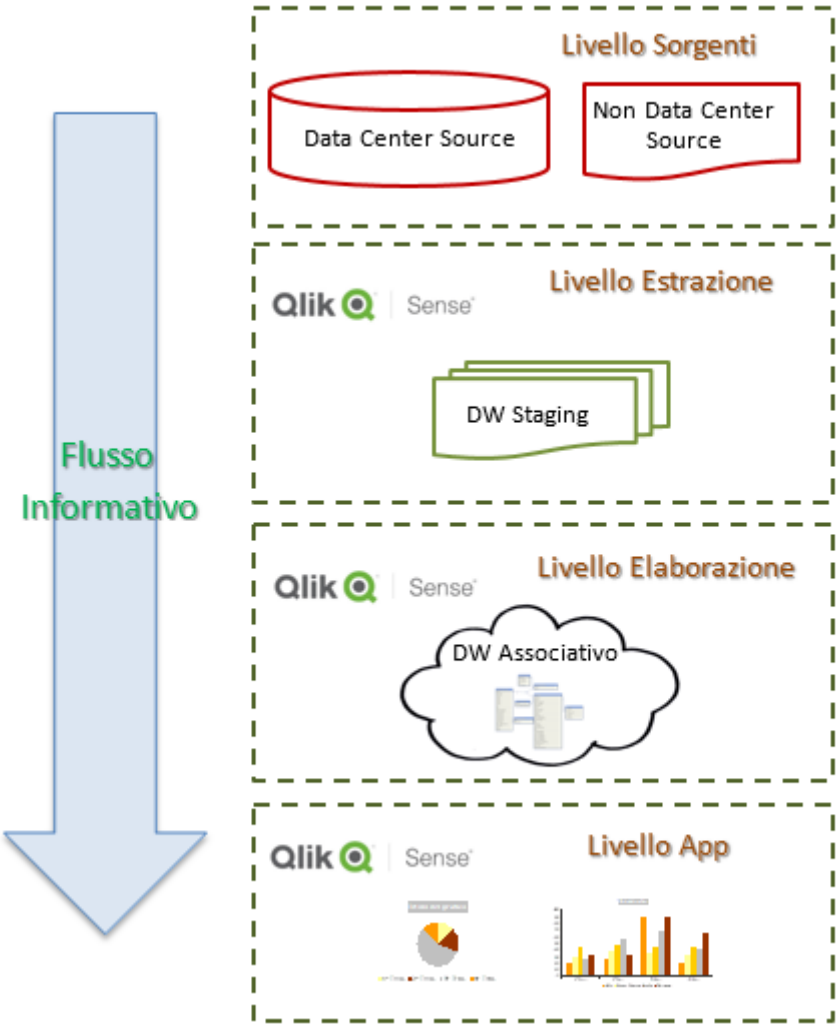

Figura 3.14: Flusso logico delle informazioni

Il livello applicativo, trattato nell'accezione di strato di "End User", si appoggia al risultato a valle delle quattro elaborazioni appena descritte. Qui sono previste soltanto le operazioni di data visualization e di gestione della sicurezza e della segregazione del dato.

Nello script di app occorre così solamente fare un "LOAD *" dei QVD predisposti al livello precedente e contenuti nella cartella *Dw_Associative* di progetto. In questo modo si tirano su i dati richiesti e si dà forma automaticamente al data model definito.

Nello specifico, il template di app è costituito da sei sezioni. La prima è, come sempre, il "Main" che si propone di realizzare le stesse operazioni compiute anche negli altri livelli. C'è poi l'area "Sub" contenente la definizione delle sub-routine. La sezione successiva, denominata "Start", dà l'avvio al caricamento richiamando le azioni iniziali: "LoadVariabili" e "AddSectionAccess". La prima funzione è addetta a estrarre dalla ETL_GLOBAL_VARIABLES le variabili del livello presentazione. Solitamente l'unica qui definita è "v_Hide_Prefix" che specifica il prefisso di occultamento per i campi che si vogliono nascondere. "AddSectionAccess" è invece la sub-routine che si occupa di applicare la Section Access in modo da mettere in pratica le limitazioni di visibilità previste per i vari profili utente. Poiché questo è un tema abbastanza delicato, vi sarà dedicato l'intero prossimo paragrafo. Seguente a "Start" ci sono due sezioni chiamate rispettivamente "SU_Dimensioni" e "SU_FTable" allestite per consentire il caricamento delle tabelle di dimensioni e fatti di interesse. Infine c'è "End" che richiama "LoadExpression", incaricata della definizione delle variabili di progetto dettagliate in ETL_01_VARIABLES, e "Cancellazioni" che invece si impegna a rimuovere tutte le variabili le quali, cominciando per "vs_", come da convenzione, sono variabili di script. Quest'ultimo passo conclude il template.

Per ultimo, nella figura sottostante è riportato schematicamente come si innesta il framework tra lo strato sorgente e le applicazioni Qlik Sense.

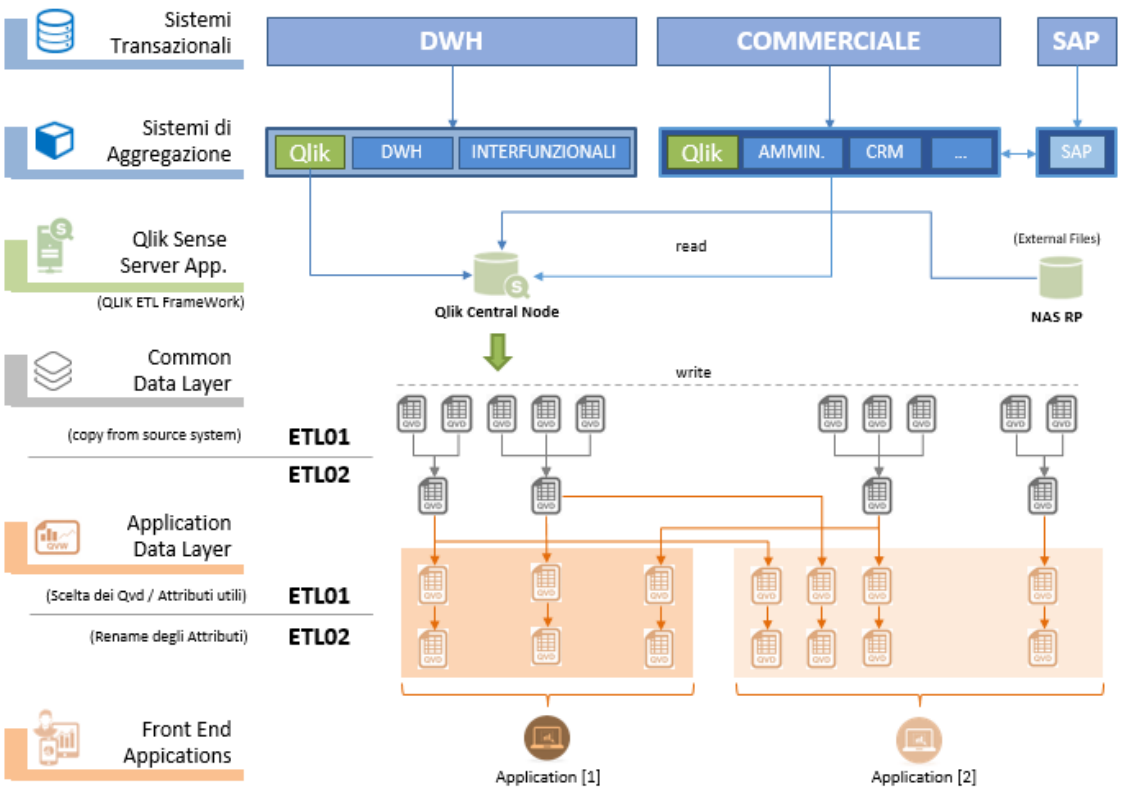

Figura 3.15: Mappa completa dell'architettura

3.6 Profilazione dell'utenza – Limitazione di operatività e visibilità sui dati

Fin qui è stato descritto il contributo benefico che apporta il framework di ETL nell'estrazione e nella manipolazione dei dati. Non meno importante è però la gestione della sicurezza. Nel momento in cui si espongono delle informazioni diventa essenziale avere il pieno controllo sugli accessi e sul set di informazioni che si vuole rendere visibile a ogni determinato tipo di utenza. Non occuparsi di questo tema avrebbe un impatto fortemente negativo sul risultato finale perché vorrebbe dire aver creato un sistema fallace che mostra dati sensibili anche a chi non dovrebbe conoscerli.

Gli aspetti da tenere in considerazione sono due: limitazione dell'operatività degli utenti assegnando permessi specifici; limitazione della visibilità sui dati operando una corretta segregazione del dato.

Il primo aspetto è gestito nella QMC e comporta un'appropriata configurazione degli utenti.

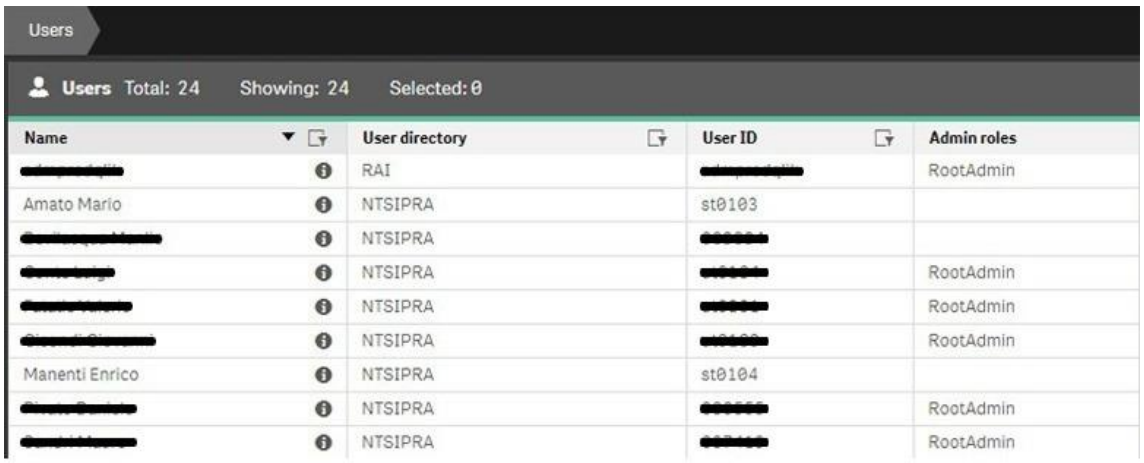

Figura 3.16: Utenti Qlik Sense

Ogni utente è identificato da un user ID, da un username e da un generico tipo di accesso (ADMIN/USER). Con l'installazione di default di Qlik Sense, il Qlik Sense Proxy Service (QPS) include un modulo che gestisce l'autenticazione degli utenti Windows e che supporta l'uso di Kerberos[4] e NTLM[5]. Per questo motivo, l'user ID è sempre rappresentato dall'utenza Microsoft Windows. L'username invece è settabile. Il tipo di accesso può essere ADMIN se si vuole che l'utente abbia pieno controllo di tutte le risorse, USER se, viceversa, si preferisce che i permessi siano personalizzabili.

[4] Kerberos è un protocollo di rete per l'autenticazione. È progettato per fornire autenticazione forte per le applicazioni client-server usando la crittografia a chiave segreta.

[5] NTLM (NT LAN Manager) è una suite di protocolli di sicurezza Microsoft che forniscono autenticazione agli utenti. L'autenticazione è di tipo challenge-response.

In ciò, vengono in soccorso due concetti rilevanti nella corretta configurazione di un server Qlik: custom properties e security rules. Una custom property è una caratteristica da assegnare a un tipo di risorsa. Le custom properties create allo stato attuale sono:

- RP_StreamAccess: si assegna a un utente per specificare gli streams a cui ha accesso ("ALL", se si vuole che acceda a tutti gli streams);
- RP_AccessStreamLevel: è diretta agli streams e indica se il controllo degli accessi è implementato a livello di stream o di app;
- RP_AppUsers: nel caso in cui la precedente proprietà assuma valore "NO", accostata ad un'app consente di impostare i nomi utente abilitati ad accedere a quell'app;
- RP_UserType: identifica le tipologie di profilo utente. Quelle possibili sono: USR_* (utente business) e IT_* (utente IT). L'asterisco può assumere uno dei seguenti valori: R (read), W (write), P (publish), A (app – creazione), DEV (developer – sviluppatore). O una combinazione di essi.

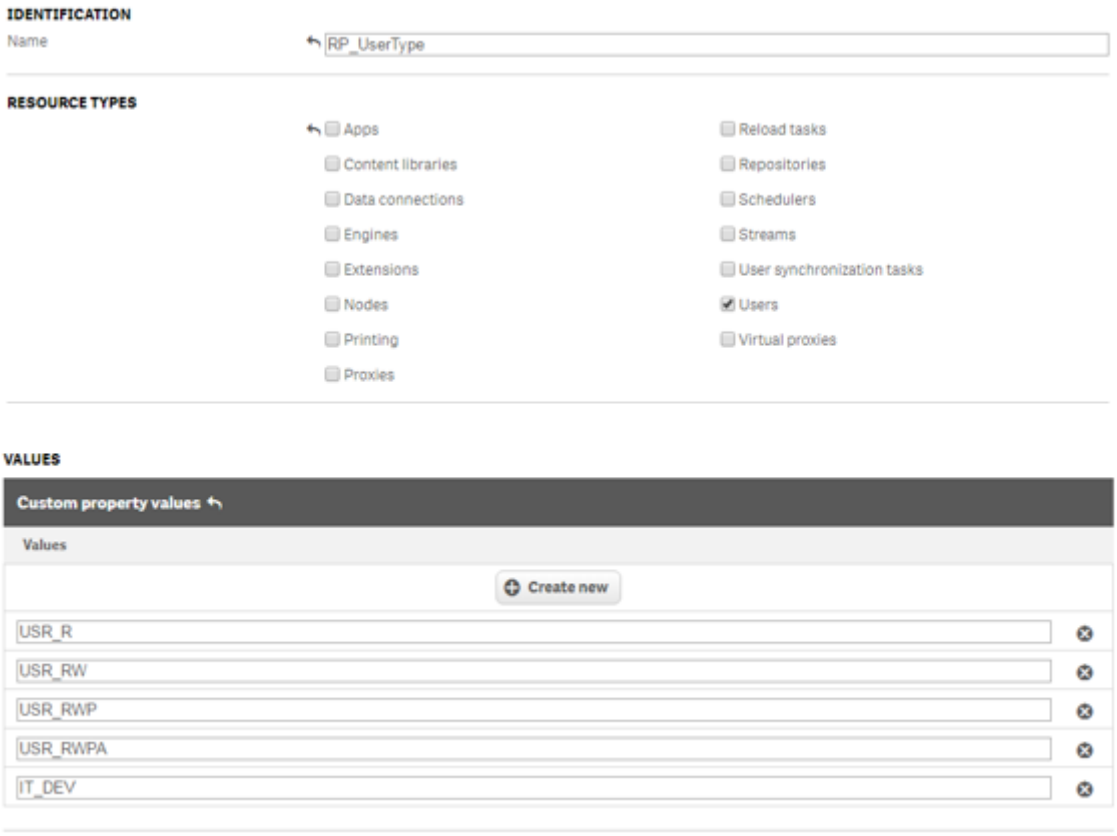

Figura 3.17: Esempio di custom property

Le security rules implementano di fatto un ACL (Access Control List). Consentono appunto di fissare i diritti d'accesso in accordo al tipo di utente. Per crearne una occorre indicare la risorsa a cui è destinata, l'azione che consente, la condizione che deve essere

verificata (qui si sfruttano le custom properties) e il contesto (solo nell'hub, sono nella QMC, in entrambi). Le security rules definiscono quindi a livello di stream, app o app object cosa può fare una determinata tipologia di utenza.

Campo	Codice
Name	RP_App_Create_IT
Description	Create App for IT
Resource	App_*
Action	Create, Update, Delete, Change Owner
Condition	(!user.IsAnonymous() and !user.@RP_UserType.Empty() and user.@RP_UserType like "IT_*")
Context	Both in hub and QMC

Figura 3.18: Esempio di security rule

In figura 3.19 si può osservare il flusso che descrive l'uso di una custom property. Quando si va a creare una custom property si definiscono uno o più valori assegnabili a una risorsa. Nel momento in cui questi valori sono stati opportunamente applicati, le risorse possono essere identificate in base a specifiche qualità. Queste vengono poi usate nelle condizioni delle security rules per definire i diritti d'accesso da valutare ogni volta che un utente chiede di accedere a una risorsa.

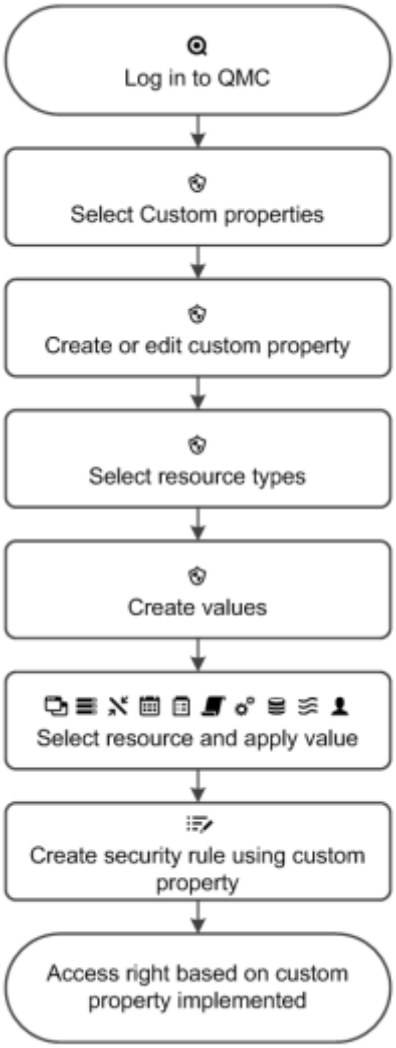

Figura 3.19: Flusso descrittivo di una custom property

Per quanto concerne il limite di visibilità sui dati, è possibile applicare una segregazione orizzontale che vuol dire limitare i records di competenza dell'utenza finale o una segregazione verticale che si traduce nel nascondere alcuni campi. Le istruzioni dello script che gestiscono la visibilità sono incluse nella sezione relativa al controllo degli accessi, riconoscibile nello script perché viene inizializzata dall'istruzione "Section Access".

Se nello script è definita una sezione relativa al controllo degli accessi, la parte dello script destinata al caricamento dei dati dell'app deve fare parte di una sezione differente, inizializzata dall'istruzione "Section Application".

La gestione della segregazione avviene mediante delle tabelle di sicurezza che devono essere opportunamente riempite e caricate. Queste tabelle stanno sotto lo schema QLIKADM e vengono lette in una sezione specifica del template di app. Il framework quindi consente di applicare le limitazioni di visibilità nell'app mediante Section Access, generata automaticamente da script dinamico.

La tabella principale di configurazione è la QLIK_UTENTI. Qui vengono specificati, per ogni utente, i tipi di segregazione da applicare. I campi costituenti questa tabella sono i seguenti:

- USERID: identifica l'utente per cui si applica la segregazione;
- ACCESS: specifica il tipo di accesso. I valori possibili sono ADMIN (utenza tecnica) o USER (utenza business);
- campi per riduzione orizzontale;
- campi per riduzione verticale.

Il numero di campi per riduzione orizzontale può essere personalizzato. Nel contesto Rai Pubblicità sono stati definiti quattro cluster: Cliente, Agenzia, Struttura di vendita, Tipo prodotto pubblicitario. Questi rappresentano i criteri di limitazione attuabili. Nella tabella QLIK_UTENTI quindi ci saranno quattro colonne denominate rispettivamente CLIENTE_GROUP, AGENZIA_GROUP, STRVEN_GROUP, TPPUBBL_GROUP che consentono di specificare, per ogni cluster, il relativo gruppo di appartenenza. La definizione dei gruppi di limitazione è in ulteriori quattro tabelle, una per cluster: QLIK_TIPO_VISIBILITA_CLIENTE, QLIK_TIPO_VISIBILITA_AGENZIA, QLIK_TIPO_VISIBILITA_STRVEN, QLIK_TIPO_VISIBILITA_TPPUBBL. Ognuna di queste tabelle è costituita da due campi: il primo identifica un gruppo, il secondo determina i valori visibili per quel gruppo. Con il gruppo "ALL" si dà indicazione di non applicare nessuna segregazione orizzontale per quel cluster. Il riempimento di queste quattro tabelle è automatizzato da una procedura contenuta nel package "qlikadm.PA_QLIK_ETL_SECTION_ACCESS". Il senso di rendere automatico questo processo è quello di tirare fuori dalle anagrafiche le regole di visibilità. In questo modo, dato un venditore cliente, il suo CLIENTE_GROUP avrà come valori associati solo i codici dei clienti di cui quel venditore si occupa o, dato un venditore agenzia, il suo AGENZIA_GROUP avrà come valori associati tutti e soli i codici delle agenzie di cui quel venditore ha l'incarico alla vendita. Lo stesso ragionamento può essere applicato per strutture di vendita e tipi di prodotto pubblicitario e quindi esteso agli altri cluster. Pertanto la limitazione di visibilità sarà diretta conseguenza delle caratteristiche del fruitore.

Concretamente i records vengono nascosti collegando i dati della sezione relativa al controllo degli accessi ai dati reali: la selezione di valori da visualizzare/escludere viene controllata tramite uno o più campi aventi nomi comuni nella Section Access e nella Section Application. Avvenuta l'autenticazione dell'utente, Qlik Sense esegue un tentativo di copia delle selezioni effettuate nei campi della Section Access all'interno dei campi della Section Application, rispettando esattamente gli stessi nomi di campo (i nomi di campo devono essere scritti in maiuscolo in quanto, per impostazione predefinita, tutti i nomi di campo e i valori di campo vengono convertiti in caratteri maiuscoli nella sezione relativa al controllo degli accessi). Dopo avere effettuato le selezioni, Qlik Sense nasconde definitivamente all'utente tutti i dati esclusi.

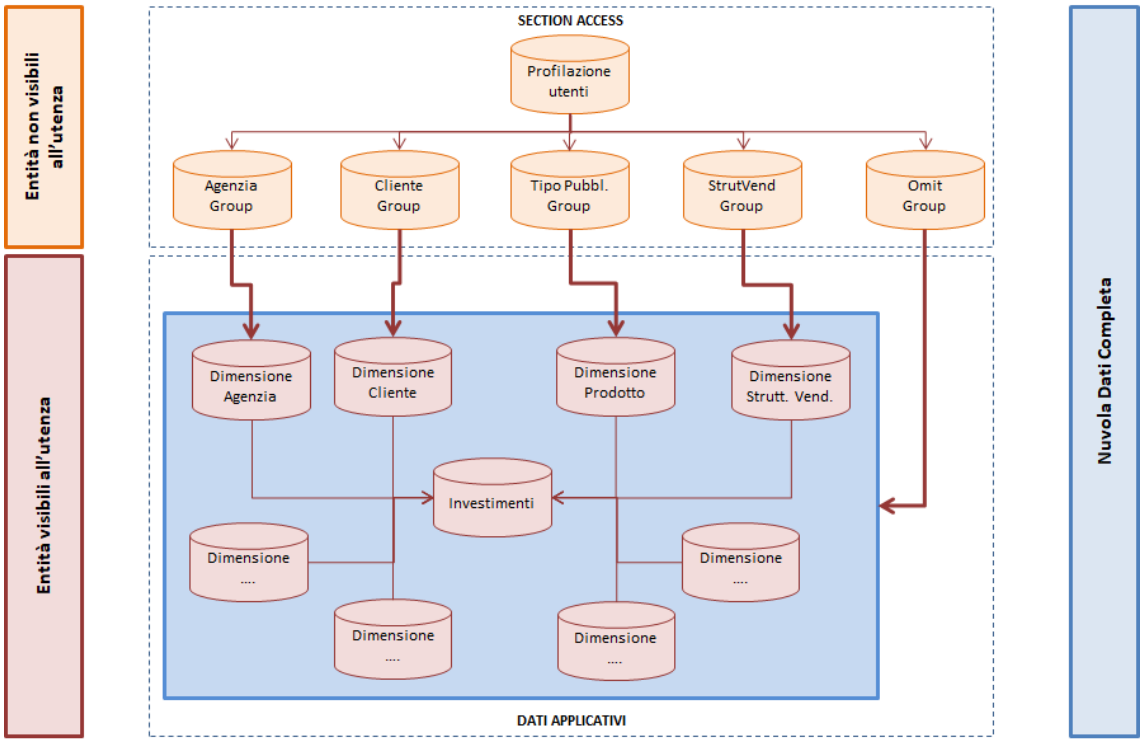

Figura 3.20: Metodo di applicazione della Section Access

Per quanto riguarda invece la segregazione verticale, questa viene gestita mediante un ulteriore campo della tabella QLIK_UTENTI denominato OMIT_GROUP che consente di definire, per utente, un gruppo di OMIT. La definizione dei gruppi di OMIT avviene nella tabella QLIK_TIPO_VISIBILITA_OMIT che, a differenza delle tabelle di visibilità sopra descritte, deve contenere, per ogni gruppo di OMIT che si vuole identificare, i campi da tagliare (agisce quindi con logica inversa rispetto alla segregazione orizzontale). Per gestire i profili che possono vedere tutti i campi della nuvola dati si utilizza il gruppo "NONE". I campi vengono nascosti dal motore di Qlik Sense utilizzando il campo di sistema OMIT.

Le tabelle per la gestione della Section Access vengono scaricate in formato QVD dall'estrattore centralizzato e messe a disposizione delle applicazioni nella cartella *Common*. La procedura di caricamento della Section Access, definita nel QVF di app, è suddivisa in due parti: controllo delle tabelle da caricare e caricamento delle tabelle. Nello specifico, ogni applicazione dovrà controllare che nella tabella QLIK_CONFIG_PROFILI_APP sia presente, con il proprio PROJECTID, almeno un valore per CAMPO_SECTION_ACCESS. Se non è presente alcun valore non sarà caricata la Section Access. In caso contrario verrà prima caricata la tabella QLIK_UTENTI e poi il programma controllerà quale tabella di limitazione di visibilità andare a caricare usando la seguente procedura:

- se viene trovato il valore "CLIENTE" verrà caricata la tabella QLIK_TIPO_VISIBILITA_CLIENTE;

- se viene trovato il valore “AGENZIA” verrà caricata la tabella QLIK_TIPO_VISIBILITA_AGENZIA.
- se viene trovato il valore “STRUTTURA_VENDITA” verrà caricata la tabella QLIK_TIPO_VISIBILITA_STRVEN;
- se viene trovato il valore “TIPO_PUBBLICITA” verrà caricata la tabella QLIK_TIPO_VISIBILITA_TPPUBBL.
- se viene trovato il valore “OMIT” verrà caricata la tabella QLIK_TIPO_VISIBILITA_OMIT.

Nella figura sottostante è presentato parte dello script di Section Access che mostra quanto descritto sopra.

```
Sub AddSectionAccess
  Section Access;

  LIB CONNECT TO $(vs_library_ETL_DB);
  // Carico la tabella QLIK_CONFIG_PROFILI_APP per verificare se il progetto prevede la Section Access.
  QLIK_CONFIG_PROFILI_APP:
  LOAD
    CAMPO_SECTION_ACCESS        AS CampoSectionAccess
  WHERE PROJECTID = $(vs_ProjectID);
  SQL
    SELECT *
    FROM QLIK_CONFIG_PROFILI_APP;


  if NoOfRows('QLIK_CONFIG_PROFILI_APP') = 0 then
    Drop Table QLIK_CONFIG_PROFILI_APP;
    Exit Sub;
  else
      call T_Utenti;

      LET vs_endfor = NoOfRows('QLIK_CONFIG_PROFILI_APP');

      for vs_i = 0 to $(vs_endfor)-1

          LET vs_SA = peek('CampoSectionAccess', $(vs_i), 'QLIK_CONFIG_PROFILI_APP');

            if vs_SA = 'OMIT' then
                call T_OMIT;
            elseif vs_SA = 'CLIENTE' then
                call T_Cliente;
            elseif vs_SA = 'AGENZIA' then
                call T_Agenzia;
            elseif vs_SA = 'TIPO_PUBBLICITA' then
                call T_Prodotto;
            elseif vs_SA = 'STRUTTURA_VENDITA' then
                call T_Territorio;
            end if
      next vs_i

  end if
  LET vs_SA =;
  LET vs_i =;
  LET vs_endfor =;
  DisConnect;
  Drop Table QLIK_CONFIG_PROFILI_APP;

end Sub
```

Figura 3.21: Script di Section Access

Nella fase di creazione/modifica di un foglio l'utenza finale accede alle informazioni della nuvola dati attraverso:

- i campi sorgente: limitati a quanto di propria competenza (segregazione verticale); visibili e selezionabili solo al momento della creazione degli oggetti grafici;
- le dimensioni e le misure definite nella Master Library (spazio in cui vengono memorizzate le voci principali, cioè le risorse create e salvate per essere riutilizzate).

Ad oggi Qlik Sense non consente la limitazione di visibilità su dimensioni e misure. Per cui la selezione di una o più dimensioni derivate da campi che non rientrano nella propria sfera di competenza comporta la visualizzazione di un oggetto senza valori (NULL) mentre la selezione di una o più misure derivate da campi cui non è consentito l'accesso comporta la visualizzazione di un oggetto con valore "N.D.".

CAPITOLO 4

PRIMA FASE DI SVILUPPO APP: REQUISITI UTENTE E DEFINIZIONE DEGLI STEPS DI PROGETTO

Questo capitolo rappresenta l'inizio del racconto del lavoro realizzato per implementare l'applicazione di BI sviluppata. Si fa un excursus dell'attività svolta, partendo dai requisiti da soddisfare e passando per l'organizzazione e la suddivisione delle fasi operative con l'obiettivo di portare a termine quanto richiesto nei tempi dovuti e con i risultati sperati.

4.1 Requisiti utente

Nel momento in cui viene commissionato un lavoro è bene fornire anche le linee guida da seguire, le prerogative da considerare e le scadenze da rispettare. Nell'ambito dei data analytics e della business intelligence bisogna mostrare anche il quadro generale delle informazioni che si vorrebbe fossero esposte e, se possibile, eventuali preferenze su grafici e figure da utilizzare.

Ciò che è stato chiesto di realizzare è un'applicazione denominata "Scheda Accordi Quadro".

"L'accordo quadro è un accordo concluso tra una o più stazioni appaltanti e uno o più operatori economici, il cui scopo è quello di stabilire le clausole relative agli appalti da aggiudicare durante un dato periodo, in particolare per quanto riguarda i prezzi e, se del caso, le quantità previste".[6]

Come afferma Alessandro Massari: "l'accordo quadro è quindi un vero e proprio modello consensuale che con flessibilità permette di stipulare contratti per lavori e/o acquisti ripetitivi, omogenei e di piccola entità".[7]

Quindi, il fine del progetto è quello di realizzare un'applicazione di analisi degli investimenti della clientela che fornisca supporto all'utenza nel momento in cui deve relazionarsi con i clienti nelle fasi di trattativa, consentendo una rapida consultazione

[6] art.3, comma 13, d.lgs. n.50/2016, *Codice dei contratti pubblici*

[7] Massari A., Montalti M., Oliveri A. P., *L'accordo quadro negli appalti pubblici. Analisi teorico-normativa e modelli operativi*, a cura di Massari A., Maggioli Editore, Rimini, 2013

delle informazioni a cui si è interessati, una navigazione fluida dei contenuti e la possibilità di realizzare confronti dalle prospettive desiderate.

Come linea guida degli elementi da implementare sono stati forniti due file Excel denominati rispettivamente “MASTER scheda per accordi quadro.xlsx” e “AQC – Dati per layout.xlsx”. Mentre il primo documento riporta sinteticamente le informazioni da presentare, il secondo fornisce suggerimenti su possibili rappresentazioni dei dati.

HOLDING XXX		EVENTI	
Responsabili contatto (tutti i mezzi)		Saremo	
TELEVISIONE GENERALISTA	Verdi Giovanna	Mondiali	
RADIO	Rossi Mario	Olimpiadi	
Settore Nielsen	**ALIMENTARI**	Europei	
N.B. escluso San Marino			
Fatturato per mezzo	**ambiente**		
Fatturato totale	sv	Statistica Cliente	Netto lordizzato; per ogni mezzo; con eventi
Fatturato al netto di eventi	sv	Statistica Cliente	Netto lordizzato; per ogni mezzo; senza eventi
% incidenza sul fatturato vs totale fatturato Rai	calcolo	Statistica Cliente	"Fatturato totale" / Netto lordizzato; per ogni mezzo; totale Rai
Posizione in graduatoria	calcolo	Statistica Cliente	
Secondi	sv	Statistica Cliente	Secondi; per ogni mezzo; con eventi
% incidenza sul fatturato vs totale secondi Rai			
Lordo	sv	Statistica Cliente	Lordo listino; per ogni mezzo; con eventi
Sconto cliente	sv	Statistica Cliente	%Sc.cliente; per ogni mezzo; con eventi
Sconto totale	sv	Statistica Cliente	% sconto totale; per ogni mezzo; con eventi
Redditività netta	calcolo	Statistica Cliente	("Fatturato totale" / "Secondi") ; per ogni mezzo; con eventi
Redditività lorda	calcolo	Statistica Cliente	("Lordo" / "Secondi") ; per ogni mezzo; con eventi
Sanatorie (in valore)	sv	Statistica Cliente	Sanatoria; per ogni mezzo; con eventi
% I.S. (in valore)	sv	Statistica Cliente	Netto lordizzato; per ogni mezzo; con eventi; Categ. Prod. INIZIATIVE SPECIALI / "Fatturato totale"
% I.S. (in valore) primi 50 clienti TV			
Omaggi Contrattuali (??????)	calcolo	Statistica Cliente	????????????????????????????
Concorrenza (Nielsen)			

Figura 4.1: Parte del file “MASTER scheda per accordi quadro.xlsx”

Come si può notare dalla figura sovrastante, data una holding (cioè una società che sta a capo di un gruppo di imprese) o un cliente, si vogliono visualizzare innanzitutto i responsabili, cioè i venditori che si occupano di quel dato cliente e il settore di riferimento.

Nella sezione successiva del documento sono poi citate diverse informazioni da mostrare relative agli investimenti su Rai Pubblicità.

Nell’area sottostante sono richiesti invece i dati riguardanti il confronto con la concorrenza. Qui le informazioni desiderate sono principalmente di due tipi: quote percentuali e relativi margini di scostamento.

Viene poi citato il flow, ovvero il flusso degli investimenti per prodotto, il quale deve essere visualizzato con profondità settimanale e in comparazione ai competitors.

Proseguendo nel file si passa a menzionare GRP, CPG, posizioni nel break e, infine, viewability e CPM. Questi sono tutti dati specifici dell’advertising. Per conoscerne il significato tecnico si rimanda al “GLOSSARIO”.

In una sezione distinta sono poi nominate due informazioni di rilievo di cui tenere conto nella realizzazione dell’app: un elenco di quattro eventi da considerare, evidentemente, come speciali e un nota bene che specifica di escludere ciò che riguarda San Marino.

L’altro file esplicita, invece, l’inquadramento che dovrebbe avere l’app. All’inizio si vuole consentire la scelta di un cliente. Successivamente si vogliono mostrare le

informazioni di contesto del cliente selezionato e, tramite filtri, determinare le dimensioni su cui effettuare l'analisi (anno, mezzo…). C'è poi una sezione che dovrebbe presentare i dati economici del mondo interno, mentre un'altra area dovrebbe esporre le quote e quindi l'analisi del mercato in rispetto alla concorrenza. Sono poi richieste altre sezioni per analizzare secondi, GRP e CPG magari utilizzando grafici lineari per vederne gli andamenti. Infine, in un foglio a parte, si dà un suggerimento su come dettagliare il flow: la richiesta è di avere un'unica tabella che, dato un cliente, settimana per settimana, mostri gli investimenti per linea di prodotto su Rai Pubblicità e sui principali competitors.

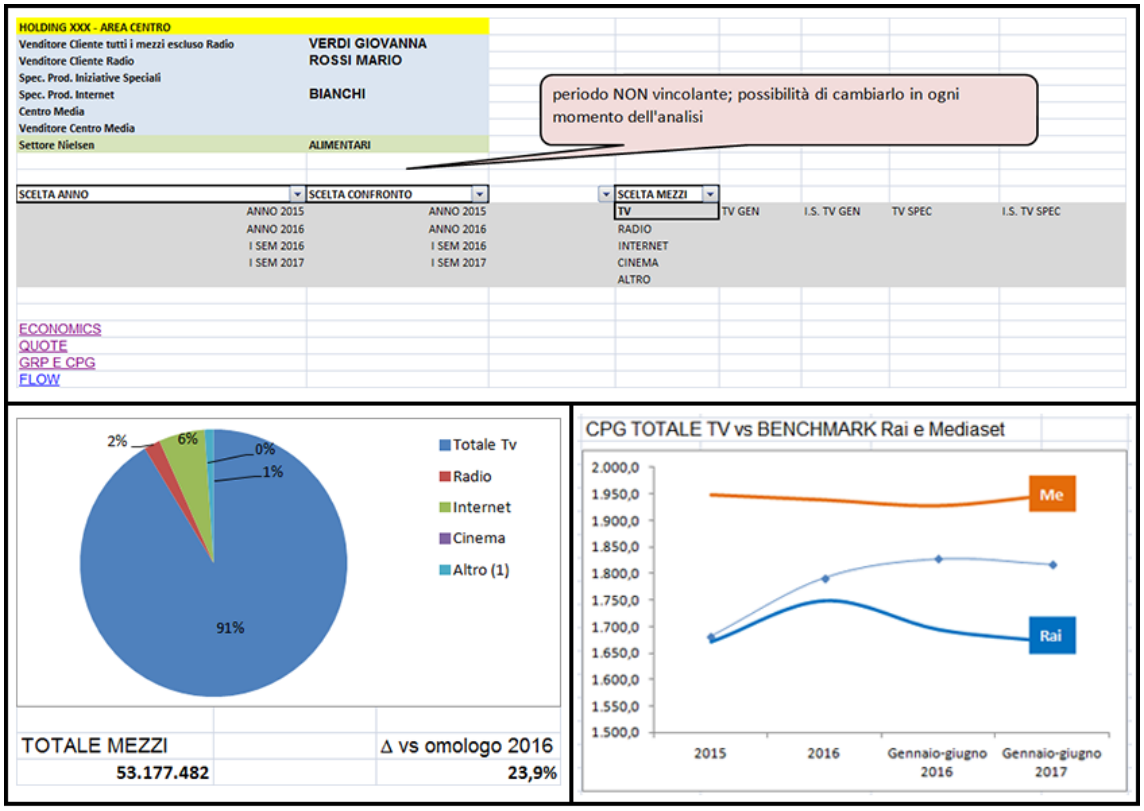

Figura 4.2: Focus del file "Dati per layout.xlsx": scelta cliente, economics, GRP e CPG

In aggiunta a questi due documenti, ne è stato fornito un terzo dal nome "Clienti Accordo Quadro.xlsx" che, tramite una colonna denominata "FL_ACCORDO_QUADRO" (valori possibili 'S' o 'N'), specifica le holding/i clienti per i quali si dovrebbe stipulare un accordo quadro.

4.2 Definizione degli steps di progetto

La prima fase del lavoro è stata quella del dialogo con l'utenza committente per comprendere il significato preciso di certe informazioni da visualizzare ed eliminare le ambiguità. Ad esempio, quando si parla di settore, si vogliono esporre tutti i settori di cui

quel cliente si occupa o solo quello prevalente? Dall'interazione è emerso che è preferibile mostrare solo il settore dominante. Questo è, infatti, utile ad analizzare il comportamento di quel cliente rispetto agli altri clienti caratterizzati dallo stesso settore prevalente. Allo stesso modo è stata indispensabile una fase di tuning delle richieste per determinare le formule necessarie al calcolo di tutto ciò che deve essere visualizzato.

Definite le richieste utente, dopo opportuno dialogo e avallo dell'utenza finale, la prima scelta compiuta è stata quella di scorporare l'analisi del flow dal resto dell'applicazione. La ragione di questa decisione risiede, principalmente, in due motivazioni: il flow fa un'analisi per prodotto e quindi si esce dai confini dell'accordo quadro che fa, invece, un'osservazione per cliente; in secondo luogo il flow vuole scendere a dettaglio settimanale, cosa non richiesta nel resto dell'applicazione che, invece, vuole esplorare i dati per anno e, in certi casi, scendere nella dimensione temporale a livello mensile ma mai settimanale. L'idea ragionata è stata quindi quella di creare due apps, una per l'analisi degli accordi quadro e un'altra per l'analisi dei flussi di investimento per linea di prodotto ma con la prima che, tramite link, punta alla seconda in modo tale che i filtri impostati siano mantenuti e il salto risulti indolore.

Comprese le richieste, analizzati i punti di attenzione su cui focalizzarsi e compiuta questa prima scelta sono state definite le fasi di lavoro per muoversi con la giusta pianificazione nella realizzazione di questo ambizioso progetto. Il primo passo da compiere è sicuramente quello del reperimento dei dati che include la ricerca della sorgente corretta e la creazione degli oggetti necessari. Questa fase è da preambolo alla creazione del modello associativo in Qlik Sense in quanto, per una corretta navigazione ed esplorazione della nuvola dati, bisogna sia predisporre correttamente il dato alla base che agire in maniera diligente e accurata nella costruzione del modello. Raccolti i dati, si può allora passare sul framework di ETL, centralizzando la fase di estrazione e manipolazione dei dati in accordo con quanto illustrato nel terzo capitolo. Definito l'aspetto data modeling, occorre poi concentrarsi sulla data visualization poiché, senza una rappresentazione grafica accattivante, intuitiva e con un filo logico naturale, la fruizione delle informazioni diventa complessa e stancante. Il front-end conclude lo sviluppo dell'applicazione. Ovviamente, realizzato il progetto, è necessario il confronto con l'utenza finale per un giudizio sul lavoro svolto, una relazione costruttiva sugli aspetti da migliorare e la valutazione di eventuali modifiche da apportare e aspetti originali da integrare.

CAPITOLO 5

SECONDA FASE DI SVILUPPO APP: RECUPERO DATI E CREAZIONE DEGLI OGGETTI NECESSARI

I prossimi paragrafi trattano un tema molto caldo nella realizzazione di un'applicazione di BI: il reperimento dei dati e la predisposizione degli oggetti utili a costituire la nuvola dati da esporre. Partendo dalle richieste, saranno analizzate le fonti per comprendere qual è il dato corretto da estrapolare. Verranno poi creati gli oggetti che rappresenteranno il punto di origine per la costruzione del modello, argomento del prossimo capitolo.

5.1 Reperimento dati

L'indagine delle sorgenti è una fase chiave nel percorso di conoscenza del dato. Da ciò si determina, infatti, da dove prelevare gli elementi richiesti e se è garantita una piena disponibilità. Per questa operazione ci si è appoggiati a due strumenti: il Supporto Vendite, un tool gestionale aziendale che espone un'interfaccia grafica di interrogazione del DWH; SQL Developer che, tramite il pannello di gestione, consente di monitorare le varie istanze di database, accedere agli oggetti presenti sui vari schema ed esaminarne il DDL (Data Definition Language, cioè lo script di creazione degli elementi di DB) ma anche eseguire queries per compiere delle ricerche mirate.

Possiamo raggruppare sinteticamente i dati che si vogliono esporre nelle seguenti macro-categorie: dato di investimento interno con relative anagrafiche; dato di investimento di mercato con relative anagrafiche; anagrafiche di transcodifica. Come spiegato nel paragrafo 2.2, i dati di mercato provengono dalla Nielsen, società specializzata in rilevazioni, stime e quote. Nel DWH sono contenuti sia il dato sugli investimenti Rai Pubblicità sia il dato Nielsen, così come le varie anagrafiche e quelle di ri-mappatura tra mondo interno e universo Nielsen che consentono una comparazione uniforme tra sorgenti eterogenee, favorendo i confronti. A primo impatto, quindi, il data warehouse contiene tutto ciò che ci è necessario. Facendo però un'introspezione intrusiva sui dati da esporre, sulle formule costitutive e sui valori di benchmark da tenere in considerazione, ci si rende conto che, per la costruzione di questa app, ci sono delle lacune. Nello specifico, il DWH non dispone delle informazioni su GRP e CPG, né delle posizioni nel break, così come dei dati sulla viewability. Mentre i dati sulle posizioni nel break sono estraibili dal DB commerciale, quindi sostanzialmente non sono stati ribaltati sul DWH ma sono comunque prelevabili tramite opportune queries, le altre richieste non sono

soddisfacibili in quanto le informazioni desiderate non sono attualmente presenti in nessuna istanza sotto il nostro controllo. Come comportarsi allora? Il confronto su questo argomento ha fatto emergere la coscienza che questi dati, attualmente assenti, in un futuro prossimo saranno a nostra disposizione e ribaltati nel DWH. Allo stato attuale, allora, sono stati forniti degli Excel che rappresentano un sample del dato.

Per quanto riguarda il GRP, i documenti assegnati sono uno per specifica holding e hanno denominazione del tipo "Grp e Grp 30##YYYYMM##HOLDING X". Dal nome del file è quindi ricavabile qual è la holding presa in considerazione. Inoltre, l'anno e il mese riportati nel titolo indicano il periodo di acquisizione dei dati cosicché è possibile determinare facilmente la freschezza del contenuto. Il fatto di avere il doppio cancelletto come separatore è stata una nostra esplicita richiesta dal momento che rende sicuro lo split. Da un'analisi compiuta, infatti, è risultato che nessuna holding ha nella propria ragione sociale il doppio cancelletto mentre ci sono casi in cui sono presenti altri caratteri speciali. Il doppio cancelletto ci assicura perciò, in fase di divisione della stringa, che tutto ciò che sta dopo la coppia di cancelletti è il nome della holding. Ogni file contiene sei colonne: "Channel", che specifica il concessionario (sarà sempre RAI, essendo un dato interno) e il mezzo (TV generalista o TV specializzata); "Target" che identifica il gruppo cui è diretta una determinata campagna pubblicitaria; "Year" che individua l'anno di rilevazione; "Month\Variables" per precisare il mese di rilevazione; "Grp" che contiene il dato numerico sul GRP; "Eq. Grp" che, invece, contiene il dato di GRP equivalente cioè il GRP ri-parametrizzato a 30''. Una prima versione di questi file presentava un unico campo atto a specificare il periodo di rilevazione ed era espresso in semestri ma, dal momento che la nostra analisi per gli accordi quadro sarà a livello mensile, è stato richiesto di fornire questo dato nella forma mese-anno.

Il discorso sul target deve essere approfondito. Come spiegato dettagliatamente nel glossario, il GRP misura la pressione pubblicitaria in riferimento a una determinata tipologia di pubblico, che possiamo definire il target group. Per questo motivo ogni rilevazione di GRP deve portare con sé l'indicazione sul target cui mira, altrimenti il dato è privo di contesto e, quindi, diventa insignificante. In merito a ciò è stato allora fornito un altro file Excel denominato "Target Clienti Accordi Quadro.xlsx" che riassume, per un certo numero di holding, i target di riferimento cioè i gruppi prevalenti cui si punta far arrivare il prodotto pubblicitario. Il file è così composto solo da due colonne: la prima che indica la holding, la seconda che rivela il target di riferimento. Una holding può avere più target di riferimento, per cui più righe possono essere correlate alla stessa holding. In generale, in Rai Pubblicità sono stati definiti 86 target che dividono l'audience in categorie per sesso, età, classe socio-economica e altre informazioni quali l'eventuale presenza di bambini o la possibile suddivisione negli acquisti in famiglia.

Per quanto riguarda invece il CPG, questo è una diretta derivata del GRP. Come chiarito infatti nel glossario, CPG sta per "Cost Per GRP", quindi, avendo il dato sugli investimenti (presente sul DWH) e il dato sul GRP (ricavabile dai file Excel descritti poco sopra), il CPG è immediatamente calcolabile senza la necessità di disporre di nuove fonti. Quello che è stato richiesto è, però, anche un valore di CPG di benchmark che, per due target specifici ("Adulti 25-54" e "Adulti 35-64"), segni un valore di riferimento per Rai e Mediaset escludendo dalla rilevazione le reti per bambini. Allo stato attuale, non c'è modo per distinguere il carattere delle reti né c'è stata fornita una discriminante per farlo. Pertanto sono stati provvisti ulteriori documenti con denominazione del tipo

“TV_CPGTREND##YYYYMM”. Come per i file sui GRP, anche in questo caso nel nome del file è riportato il periodo di acquisizione. Il contenuto è costituito da quattro fogli che distinguono i valori presenti per concessionario (Rai o Mediaset) e mezzo (TV generalista o TV specializzata) ed espongono cinque colonne: “Anno”; “Mese”; “Inv. Netto” riportante il netto investito per il mese considerato; “Adulti 25-54 Grp’s 30’’”; “Adulti 35-64 Grp’s 30’’”. Le ultime due colonne contengono, quindi, i valori di GRP 30’’ per i due target standard definiti. In questo modo, avendo il dato sugli investimenti e quello sui GRP, tramite una semplice divisione è possibile ottenere i valori di CPG di benchmark.

L’ultimo dato mancante è quello sulla viewability. A tal proposito è stato fornito un ultimo documento dal nome “VIEWABILITY CLIENTE SETTORE##201706” che, come esplicitato nel titolo, contiene i valori registrati a giugno 2017, quindi con riferimento al primo semestre del 2017. Il file è composto da sette colonne: “Anno”; “Mese”; “Clienti”; “Settori”; “Gross Impressions”, “Measured Impressions”; “Measured Views”. Le prime due colonne sono superflue in quanto indicano anno e mese di rilevazione dei dati, informazione comunque già ricavabile dal nome del file. Le successive due esplicitano il cliente e il settore di riferimento di quel cliente cui si riferiscono i valori presenti negli ultimi tre campi. Questi contengono dati riguardanti il mondo web per il cui significato si rimanda al glossario. L’unica cosa che importa sapere è che la viewability è ottenibile dividendo measured views e measured impressions.

Ricevuti questi file Excel, un primo lavoro necessario è stato quello di uniformare quanto contenuto in questi documenti rispetto a quanto presente sul DWH e, contemporaneamente, adoperarsi in un’operazione di pulizia del dato. Ad esempio, da un’analisi svolta sul documento concernente la viewability è emersa l’evidenza che i dati anagrafici contenuti non rispecchiano né l’anagrafica Rai Pubblicità nè quella Nielsen. Interagendo con i mittenti del file, si è scoperto che questi dati provengono da ComScore, società specializzata nel fornire misurazioni cross-platform per audience, marchi e comportamento dei consumatori in diversi settori commerciali del web. Al fine di consentire all’utenza finale una navigazione coerente, si è quindi proceduto a compiere un’opportuna transcodifica delle anagrafiche andando a mappare i clienti ComScore con le holding contenute nei sistemi Rai Pubblicità. Da ciò è nato il file “Decodifica Clienti Viewability.xlsx”, costituito da due colonne: “Clienti ComScore” e “Holding RP” che, realizzando il mapping, consente di collegare i dati sulla viewability alla nuvola dati. L’altro intervento dovuto è stato quello della pulizia del dato atto a eliminare le inconsistenze tra valori logicamente associati. Per questo nel file “Target Clienti Accordi Quadro.xlsx”, che riporta una serie di ragioni sociali, sono state operate delle correzioni concernenti prevalentemente i caratteri speciali in modo da avere uniformità tra quanto qui contenuto e la semantica propria del DWH.

5.2 Creazione viste

Nel paragrafo precedente è stato trattato il recupero dei dati analizzando le fonti estemporanee per ciò che non è presente sul data warehouse. Per quanto riguarda il DWH occorre, però, predisporre gli oggetti. La scelta intrapresa è stata quella di creare quanto necessario al mondo Qlik sotto lo schema QLIKUSR. In questo modo si ottiene il grosso

vantaggio della centralizzazione che si traduce in una manutenzione più agevole e celere, dal momento che gli oggetti saranno tutti nella stessa locazione e non sarà necessario creare sinonimi. L'utente QLIKUSR ha accesso in lettura a tutto il contenuto del DWH, quindi non è un problema prelevare i dati appropriati e da questi dare origine agli oggetti desiderati e compilarli. Si è deciso di creare solo viste, quindi tabelle virtuali che forniscono una rappresentazione logica di una o più tabelle fisiche. Il motivo risiede principalmente in tre ragioni: il disaccoppiamento dall'oggetto originale, in modo che le modifiche su questi elementi di DB non impattino sulle strutture alla base; l'agevolazione nella gestione dei permessi, cosicché l'accesso alle tabelle sia regolato in maniera diversa rispetto all'accesso alle viste; la normalizzazione dell'informazione. Inoltre si riscontra una semplificazione nella stesura delle queries.

5.2.1 Le regole di nomenclatura

A meno di impedimenti derivanti dalla natura della fonte dati, le problematiche legate alla naming convention sono così risolte direttamente sulle viste.

A tal proposito sono state definite delle regole di nomenclatura. Nello specifico, sono stati identificati dei prefissi e dei suffissi da applicare sia agli oggetti che ai singoli campi a seconda di determinate caratteristiche intrinseche. I prefissi stabiliti per le viste sono:

- VAN_: per le viste anagrafiche che conterranno, quindi, solo codice, descrizione ed un eventuale campo di ordinamento;
- VDM_: per le viste dimensionali che ingloberanno dati provenienti da più anagrafiche aventi legami gerarchici o tematici;
- VDT_: per le viste dati, cioè quelle che saranno utilizzate come tabelle dei fatti;
- VTM_: per le viste calendario, quindi quelle che rappresenteranno la dimensione temporale;
- VPF_: per le viste dei portafogli, contenenti il ventaglio della clientela propriamente strutturato.

Inoltre, si userà il plurale per le viste anagrafiche e dati, il singolare per tutte le altre tipologie.

Per quanto riguarda invece i singoli campi, sono stati definiti dei prefissi per inquadrare opportunamente la tipologia di appartenenza:

- ID_: campo chiave;
- DATA_: campo data;
- DATAUPD_: campo data d'aggiornamento;
- IMP_: campo importo;
- NR_: campo numerosità;
- ORD_: campo ordinamento;
- FL_: campo flag.

Allo stesso modo sono stati indicati dei suffissi:

- _COM: campo commerciale;
- _RIALL: campo riallineato;
- _SA: campo di Section Access (per approfondimenti riferirsi al paragrafo 3.6).

5.2.2 Fatti, dimensioni e misure

Prima di entrare nel dettaglio delle viste, è bene aprire una parentesi sul concetto di fatti, dimensioni e misure. In questo modo sarà più semplice comprendere il contenuto di una vista.

Un fatto rappresenta un processo di business. Modella, quindi, un insieme di eventi di interesse che evolvono nel tempo. Ogni fatto viene misurato durante un momento significativo dell'erogazione di un processo. Il fatto è quindi una collezione di dati da analizzare e, così, ciascuna riga della fact table memorizza un insieme di misure associate a una particolare combinazione di membri, presa all'intersezione di tutte le dimensioni. Una dimensione rappresenta una prospettiva rispetto alla quale effettuare l'analisi, cioè un'informazione del contesto in cui è stata catturata una misurazione. È, quindi, costituita da un insieme di attributi, organizzati in opportune gerarchie, che categorizzano un fatto e le misure associate al fine di consentire agli utenti di rispondere ai quesiti di business. La funzione primaria delle dimensioni è triplice: fornire filtraggio, raggruppamento e classificazione. Una tabella dei fatti contiene, quindi, business facts (le misure) e le chiavi esterne che creano un riferimento alle primary keys delle tabelle dimensionali. Queste contengono attributi descrittivi (i campi) che sono tipicamente campi testuali (o numeri discreti che si comportano come fossero testo), solitamente organizzati in livelli di aggregazione. Al contrario, una misura è un valore numerico che rappresenta una proprietà quantitativa del processo cui si riferisce.

In sintesi:

- un fatto modella una collezione di eventi che muta nel tempo (es. le vendite);
- una dimensione descrive le prospettive di analisi di un fatto (es. prodotto, negozio, data) ed è caratterizzata da attributi di tipo categorico;
- una misura rappresenta una proprietà numerica di un fatto (es. incasso, quantità venduta).

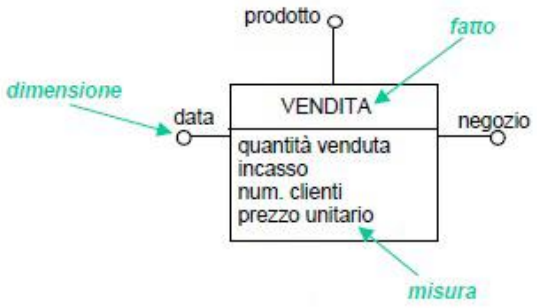

Figura 5.1: Fatti, dimensioni, misure[8]

5.2.3 Panoramica sulle viste

Passiamo adesso rapidamente in rassegna le varie viste create in modo che, quando nel prossimo capitolo si parlerà della costituzione del modello e ci si riferirà ovviamente a questi oggetti, se ne conoscerà già il contenuto.

La VDT_INV_PUBBLICITARI è la vista che mostra i dati di venduto Rai Pubblicità. La principale fonte sorgente è la tabella CC_DATI_MAGAZZINO da cui estrapoliamo tutto il contenuto. Dal momento che questa vista dovrà "parlarsi" con quella che espone i dati di mercato, è stato necessario prendere in considerazione anche la NLS_ST_RELAZ_RETI che mappa l'anagrafica delle reti Rai Pubblicità con l'anagrafica delle reti Nielsen e le tabelle NLS_ST_MEZZI, NLS_RELAZ_MEZZI_TP_PRODOTTO, VI_CC_CFG_RELAZ_DGC_NODI_NL che, messe opportunamente in join tramite il codice di dettaglio commerciale ed il codice di origine ricavo con la CC_DATI_MAGAZZINO, forniscono il codice raggruppamento mezzi Nielsen che è il modo con cui la Nielsen concentra i mezzi per macro-categorie. I campi tirati su, quindi, sono i codici che identificano: il soggetto (cioè il prodotto pubblicizzato), il cliente, la holding (cioè il gruppo di aziende cui eventualmente appartiene anagraficamente un cliente), il mezzo, il media (padre del mezzo), il raggruppamento mezzi Nielsen, la rete, la rete Nielsen, lo stato di vendita. Ci sono, poi, la data di trasmissione (quindi la data di riferimento) e tutte le misure divise per importi (netto, netto lordizzato, lordo, netto impressions vendute, netto impressions effettivamente erogate, sconto commerciale, sconto totale, sconto sanatoria, sconto recupero) e campi numerosità (secondi, passaggi, impressions vendute, impressions erogate). Si espongono inoltre le chiavi costruite ad hoc per concatenare le dimensioni e tutta una serie di altri campi presi dalla CC_DATI_MAGAZZINO tirati su e messi a disposizione per usi futuri e progetti di nuove apps.

La VDT_INV_MERCATO è, invece, la vista contenente i dati di mercato provenienti dalla Nielsen cui, tramite un'operazione di ri-mappatura, sono agganciate le anagrafiche Rai Pubblicità in modo che i dati esposti abbiano evidenza delle corrispondenze Nielsen-Rai Pubblicità. In questo caso ci si è appoggiati a una vista materializzata, cioè una vista scritta fisicamente su disco, cui è già stato applicato il complesso intervento di ri-

[8] Golfarelli M., Rizzi S., *Data warehouse. Teoria e pratica della progettazione*, McGraw Hill, Milano, 2006

mappatura. I dati esibiti sono, come nel caso della VDT_INV_PUBBLICITARI, i codici che inquadrano l'ambito di investimento. Quelli rilevanti sono: codice cliente, codice cliente Nielsen, codice holding, codice raggruppamento concessionari Nielsen (che rappresenta il modo con cui la Nielsen riunisce i vari concessionari pubblicitari), codice raggruppamento mezzi Nielsen, codice rete Nielsen, codice settore Nielsen (identificante il settore merceologico), codice stato di vendita. C'è poi il riferimento temporale, espresso in anno e mese. Insieme a questi campi, assume rilevanza anche un flag denominato FL_CARICAMENTO che delinea il tipo periodo Nielsen. A livello cronologico, infatti, la Nielsen struttura la linea temporale in due periodi: periodo consolidato che, tipicamente, si estende fino a due mesi prima dalla data corrente e periodo EIS che costituisce, invece, l'ultimo lasso di tempo. Nel periodo consolidato tutti i dati sono considerati definitivi mentre, nel periodo EIS, la Nielsen non garantisce la rilevazione e i dati forniti sono provvisori e ancora soggetti a modifica. Ovviamente, sono poi esposte tutte le misure: importo lordo Nielsen, importo netto Nielsen, importo netto marketing Nielsen, numero di impressions erogate, numero di secondi. Ciò che viene ricevuto dalla Nielsen è il lordo da cui, tramite dei fattori di abbattimento, si calcola il netto. Durante il periodo EIS i fattori di abbattimento ottenuti potrebbero non essere precisi. Per questo motivo il reparto marketing ne fornisce altri. Il calcolo del netto nel periodo EIS avviene così tenendo conto di tale valutazione. Questa è la ragione per cui si distinguono il netto Nielsen, che si riferisce al periodo consolidato, e il netto marketing Nielsen che, invece, riguarda il periodo EIS. Il primo è calcolato soltanto a partire da quanto fornisce la Nielsen, il secondo subisce un'operazione di correzione.

La VDT_DATI_NIELSEN contiene solo i dati Nielsen puri. A differenza della VDT_INV_MERCATO, esterna il dettaglio temporale per anno, mese e settimana (la settimana è, però, fornita per i soli mezzi TV e radio) con l'appendice del flag caricamento utile a specificare il periodo Nielsen. Il riferimento all'investimento viene espresso a livello di marca (cioè di prodotto pubblicizzato). Sono poi esplicitati, mediante codice: l'holding, il cliente Nielsen, il raggruppamento concessionari Nielsen, il raggruppamento mezzi Nielsen, la rete Nielsen, il settore Nielsen. Le misure esposte sono le stesse della VDT_INV_MERCATO: importo lordo Nielsen, importo netto Nielsen, importo netto marketing Nielsen, numero di impressions erogate, numero di secondi.

Le differenze tra quest'ultima vista e la VDT_INV_MERCATO riguardano il fatto che nella VDT_DATI_NIELSEN non vengono agganciate le anagrafiche Rai Pubblicità, che il dettaglio temporale ha profondità settimanale e che si scende a livello di marca. In questo modo, a seconda del caso di utilizzo, si può scegliere di impiegare una o l'altra vista: per confronti col mondo interno c'è la VDT_INV_MERCATO; se, invece, si vuole analizzare solo ciò che proviene dalla Nielsen si può usare la VDT_DATI_NIELSEN che offre il massimo livello di dettaglio. Entrambe le viste saranno adoperate nel nostro progetto in fasi differenti.

Queste appena descritte sono le tre fact tables prodotte. Tutte le altre viste sono dimensional tables, quindi espongono il contesto, e si dividono per strutturazione interna in quelle che, per semplicità, sono chiamate: VAN, cioè le viste anagrafiche pure che trattano un dato elemento in pancia al patrimonio informativo; VDM che riuniscono i dati provenienti da più VAN associati da una dipendenza strutturale o settoriale; VPF, ossia le viste dei portafogli. Il motivo di creare sia VAN che VDM risiede nel fatto che talvolta vogliamo portare nel data lake di un progetto un singolo elemento, altre volte, invece,

preferiamo trascinare dentro l'intera gerarchia. Attraverso questa scelta implementativa lasciamo così una certo grado di libertà di utilizzo in base all'ambito di applicazione. Le VPF nascono, invece, solo per un ordine concettuale.

Le VAN create sono le seguenti: VAN_AGENZIE_COMMERCIALI, VAN_AREE, VAN_CATEGORIE_PUBBLICITARIE, VAN_CLASSI_INTERVENTO, VAN_CLIENTI_COMMERCIALI, VAN_CLIENTI_NIELSEN, VAN_EDITORI, VAN_EVENTI, VAN_HOLDING, VAN_MEDIA_PUBBLICITARI, VAN_MEZZI_PUBBLICITARI, VAN_MEZZI_PUBBLICITARI_NIELSEN, VAN_PRODOTTI_PUBBLICITARI, VAN_RAGG_CONCESSIONARI_NIELSEN, VAN_RAGG_MEZZI_NIELSEN, VAN_RETI, VAN_RETI_NIELSEN, VAN_SETTORI_NIELSEN, VAN_TESTATE_EDITORIALI, VAN_TIPI_EVENTO, VAN_TIPI_PUBBLICITA, VAN_VENDITORI.

I nomi sono esplicativi del contenuto. Ciò che viene esposto è sempre il codice, la descrizione e, talvolta, un campo ordinamento nel caso in cui si voglia specificare una disposizione personalizzata diversa da quella letterale/numerica. L'unico punto di attenzione su cui vale la pena soffermarsi è la VAN_HOLDING. La scelta adottata qui è stata, infatti, quella di esporre, per ogni cliente, la holding cui appartiene ma con una particolarità: nel caso in cui un dato cliente non faccia parte di nessuna holding, si mostra il proprio codice e la propria descrizione. In questo modo sostanzialmente si afferma che una holding può essere considerata sia come un raggruppamento di clienti che come un singolo cliente.

Le VDM riuniscono più VAN i cui contenuti sono accomunati da una relazione anagrafica o di contenuto, creando le dimensioni propriamente dette. Di questa tipologia di vista ne sono state create diverse. La VDM_EVENTO espone le informazioni sugli eventi nella loro accezione massima dettagliandone anche il tipo e la data di inizio e fine. La VDM_MARCHE_CLIENTI_NIELSEN consente di esplicitare il legame che sussiste tra le marche (cioè i prodotti pubblicizzati) e i clienti Nielsen. Con la VDM_MEZZI_RP_MEZZI_NIELSEN, invece, si crea l'associazione tra l'anagrafica dei mezzi Rai Pubblicità e quella Nielsen. La VDM_PRODOTTO_PUBBLICITARIO mostra la gerarchia del prodotto pubblicitario. Partendo dal livello più basso (il tipo pubblicità), si sale passando dal mezzo fino ad arrivare al media e si associa la categoria del prodotto. Stesso discorso, ma riguardante il prodotto editoriale, viene fatto nella VDM_PRODOTTO_EDITORIALE: per ogni rete si specifica la testata editoriale cui appartiene e il corrispondente editore. La VDM_RETI_RP_RETI_NIELSEN realizza il mapping tra anagrafica delle reti Rai Pubblicità e anagrafica delle reti Nielsen ma, rispetto alla VDM_MEZZI_RP_MEZZI_NIELSEN, qui il lavoro è semplificato dal fatto che il rapporto è (0,1):(1,1) (cioè ad ogni rete Nielsen corrisponde una o nessuna rete Rai Pubblicità). La VDM_SOGGETTI_CLIENTI è il corrispettivo della VDM_MARCHE_CLIENTI_NIELSEN per il mondo interno: vengono quindi esposti tutti i soggetti (cioè i prodotti pubblicizzati) e come questi sono relazionati con i clienti.

L'ultima tipologia di vista realizzata è chiamata VPF. Nello specifico, la VPF_AGENZIE esplicita il rapporto tra clienti e agenzie allo stato attuale. Mostra quindi, per ogni agenzia con cui Rai Pubblicità è in contatto, i clienti che segue e su che mezzo. Un cliente, infatti, può scegliere di dare incarico ad un'agenzia per un dato mezzo e ad un'altra agenzia per un mezzo differente. La VPF_VENDITORI_CLIENTI e la

VPF_VENDITORI_AGENZIE, invece, rendono manifeste le relazioni correnti che sussistono tra i venditori e la clientela intesa sia come i clienti propri che come le agenzie che rappresentano un certo numero di clienti. Qui si riportano informazioni supplementari inerenti i venditori: l'area di interesse, la classe d'intervento (cioè la specializzazione) e il mezzo di cui si occupano. Sono esplicitati solo i codici ma, attraverso le VAN, si può ricavare la descrizione associata.

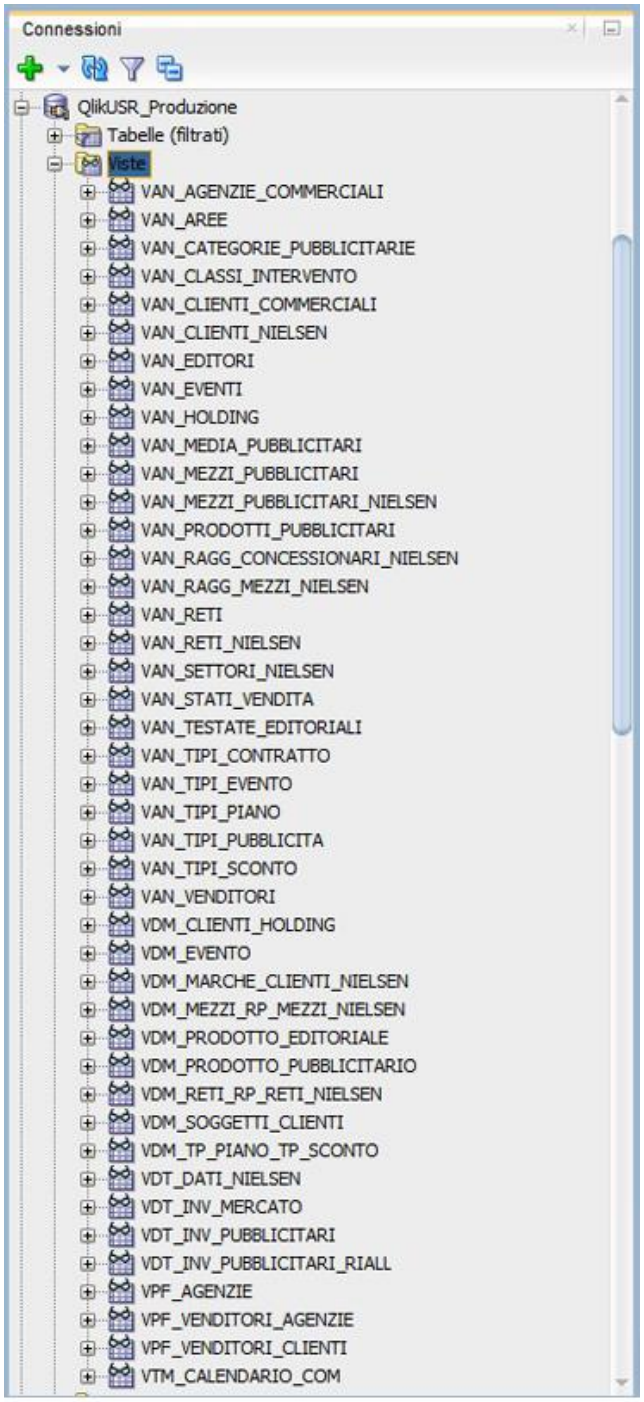

Figura 5.2: Elenco viste create su schema QLIKUSR dell'istanza DWH

Parlando nel paragrafo precedente del reperimento dei dati, si è detto che, per quanto riguarda le posizioni nel break, questa informazione non è presente sul DWH ma è recuperabile dall'istanza del DB commerciale. A fronte di ciò, sono allora state create due viste nello schema QLIKUSR dell'istanza DB commerciale deputate a prelevare questo dato per il mezzo TV e il mezzo radio chiamate rispettivamente VDT_INV_PUBBL_POS_BREAK_TV e VDT_INV_PUBBL_POS_BREAK_RADIO. Quello che viene richiesto è la percentuale di posizioni pregiate. Sono considerate posizioni pregiate solo la prima e l'ultima all'interno di un break. Quindi, quello che ci interessa estrarre è, per ogni spot, un flag che specifichi se è in una posizione considerata di rilievo o meno. Si tirano su allora dalla tabella degli spot TV o radio: il piano, la versione, il dettaglio, il numero di spot e la sua posizione all'interno di un break. Poi, tramite la partition by, si estraggono la prima e l'ultima posizione del break cui lo spot appartiene. I primi quattro campi delineano in maniera univoca uno spot e vengono concatenati con il codice media ("T" per la TV, "N" per la radio) costituendo la chiave comunicato. Il flag viene invece creato settando il valore 'Y' se la posizione dello spot è la prima o l'ultima nel break, 'N' altrimenti. Nella figura sottostante è possibile vedere il codice di creazione di una delle due viste. Si riporta questo come esempio per tutti gli altri.

```
CREATE OR REPLACE FORCE VIEW "QLIKUSR"."VDT_INV_PUBBL_POS_BREAK_TV" ("CHIAVE_COMUNICATO", "FL_POSIZIONE_PRIVILEGIATA") AS
SELECT
       'T'||'|'||LPAD(from_spots.dp_pr_cod_piano,8,'0')||'|'||LPAD(from_spots.dp_pr_vers_piano,2,'0')
       ||'|'||LPAD(from_spots.dp_cod_dett,4,'0')||'|'||LPAD(from_spots.cod_spot,6,'0')||'|'            AS CHIAVE_COMUNICATO,
       CASE WHEN from_spots.posiz = from_spots.min_pos
             THEN  'Y'
            WHEN from_spots.posiz = from_spots.max_pos
             THEN  'Y'
            ELSE 'N'
       END                                                                                             AS FL_POSIZIONE_PRIVILEGIATA
FROM  (SELECT
       s.dp_pr_cod_piano,
       s.dp_pr_vers_piano,
       s.dp_cod_dett,
       s.cod_spot,
       s.posiz,
       MIN(s.posiz) over(PARTITION BY s.bm_gb_bf_cod)  min_pos,    --> Prima posizione a livello di break
       MAX(s.posiz) over(PARTITION BY s.bm_gb_bf_cod)  max_pos     --> Ultima posizione a livello di break
     FROM  spots_tv  s
     WHERE s.stato_vend      <>  'ANN'
     AND   s.cod_disattivazione  IS NULL)  from_spots;
```

Figura 5.3: Creazione vista VDT_INV_PUBBL_POS_BREAK_TV

CAPITOLO 6
TERZA FASE DI SVILUPPO APP: IL DATA MODEL

Questo capitolo costituisce il fulcro del lavoro realizzato. Una volta che i dati sono stati recuperati, bisogna plasmare il reticolo e creare le corrette associazioni in modo da realizzare un modello coerente con quanto si vuole esporre e che non tagli fuori parte dei dati. Poiché il modello sta alla base dei contenuti visualizzati, deve essere sensibile alle selezioni utente cosicché, dinamicamente, risponda in maniera esatta e concorde con quanto desiderato. Il primo paragrafo tratta le caratteristiche che dovrebbe rispettare un data model Qlik, fornisce le possibilità di sviluppo quando si può agire in maniera propria e specifica le best practices e cosa, invece, è sconsigliato fare. In seguito, nel secondo paragrafo, si cala quanto detto nel contesto specifico dell'app realizzata analizzando le complessità con cui ci si è dovuti imbattere e le revisioni che hanno condotto alla versione definitiva del modello dati.

6.1 Lo stato dell'arte

Prima di scendere verticalmente nella costruzione del modello dati dell'applicazione sviluppata, occorre fare una digressione sul data modeling Qlik. In generale, non esiste un modo univoco di foggiare un modello. Questo perché la modellazione è totalmente dipendente dai requisiti: i sistemi con cui si lavora, le skills acquisite, la sicurezza da garantire, le funzionalità da realizzare, la flessibilità, il tempo, il denaro a disposizione, la mole di dati da trattare ma, soprattutto, i requisiti aziendali, le richieste dei business users. Perciò le best practices non sono universali ma vanno applicate in base alla situazione, al contesto.

6.1.1 Confronto tra modello associativo e modello relazionale

Nel primo capitolo, parlando di Qlik, è stato introdotto il concetto di modello associativo, presentandolo come innovazione ed elogiandolo come uno dei pregi di Qlik. Ma in cosa si distingue dal modello relazionale?

In poche parole, i database relazionali e le queries SQL non sono stati progettati per un deep data analytics. È sì vero che il linguaggio SQL è necessario per estrarre dati da molte fonti ma avere un approccio basato su queries SQL anche per la parte di analisi è un

grosso difetto in quanto si avrebbe un'esplorazione lineare e limitata e un'osservazione solo su sottoinsiemi di dati parziali. Le fonti dati, inoltre, devono essere riunite utilizzando join SQL e le ipotesi circa su quali tipi di domande avranno gli utenti devono essere fatte in anticipo. Tutti gli altri dati sono lasciati indietro. Così, se le informazioni vengono caricate da più fonti, gli strumenti basati su queries possono rischiare di causare la "perdita" di dati a causa di join eseguiti al momento del caricamento. In genere si definisce un'origine primaria e si introducono solo sottoinsiemi di dati da origini secondarie che corrispondono ai valori nell'origine primaria. Ciò può causare la perdita di dati, di cui gli utenti aziendali potrebbero non essere a conoscenza. C'è anche il rischio di calcoli errati, poiché i valori potrebbero essere raddoppiati o addirittura triplicati se le queries e i join non sono definiti correttamente. Ottenere il risultato desiderato richiede in genere agli utenti di avere una forte familiarità con il modello dati e le competenze adatte a strutturare correttamente le queries. Non avendo il set di dati completo, è difficile ottenere un quadro generale di come le cose sono correlate. Inoltre, a prescindere dalla robustezza del database, le queries richiedono tempo per essere eseguite. Un sistema lento e non reattivo porta a perdere il filo logico dei pensieri e questo ha come risultato un decadimento della capacità intuitiva, propulsore dei passi successivi. Se un utente desidera cambiare il punto di analisi in base a quello che scopre, è probabile che debba ricostruire queries complesse, il che significa spesso tornare alle sorgenti. Questo schema è definito "ask, wait, answer" dal momento che ogni nuovo tipo di domanda ha un periodo di attesa.

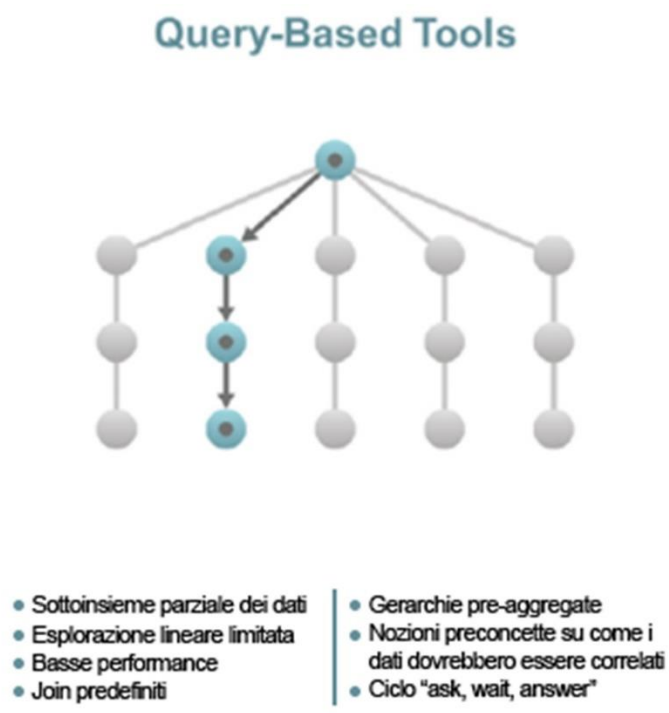

Figura 6.1: Query-based tools

Il modello associativo di Qlik, invece, è progettato specificamente per l'esplorazione e

l'analisi interattiva e in forma libera. Consente agli utenti di scorgere l'intera storia che vive all'interno dei propri dati e di sondare tutte le possibili associazioni esistenti, attraverso tutte le fonti dati. Gli utenti possono così esplorare liberamente utilizzando selezioni interattive e ponendo domande in qualsiasi direzione senza restrizioni o limiti. Dopo ogni clic, il motore associativo di Qlik ricalcola istantaneamente tutte le analisi al contesto attuale. Gli utenti beneficiano in questo modo di un feedback istantaneo e ciò consente loro di acquisire nuove informazioni e intraprendere i passi successivi nell'analisi. Ciò significa che gli utenti sono liberi di cercare, esplorare e muovere il loro bacino di osservazione in base a ciò che vedono, al loro intuito, senza limitazioni e senza dover tornare alle fonti e attendere. Si scoprono in modo coerente margini di approfondimento che, se non previsti in anticipo, sono ignorati usando gli strumenti basati su queries. A differenza di questi, infatti, il modello associativo non limita gli utenti a gerarchie predefinite o nozioni preconcette su come i dati devono essere associati e consente loro, quindi, di addentrarsi nell'indagine e comprendere appieno il modo in cui i dati sono realmente correlati. Inoltre, sfruttando la cosiddetta "potenza del grigio" illustrata nel paragrafo 1.1, si permette agli utenti di vedere nelle loro analisi non solo i dati riguardanti le selezioni applicate, ma anche i valori non correlati. Queste informazioni spesso trasmettono conoscenze interessanti e aiutano gli utenti a realizzare intuizioni e spunti prima non considerati. Viceversa, con gli strumenti basati su queries, questi valori verrebbero semplicemente filtrati, lasciando agli utenti solo un set di dati parziale e una storia incompleta.

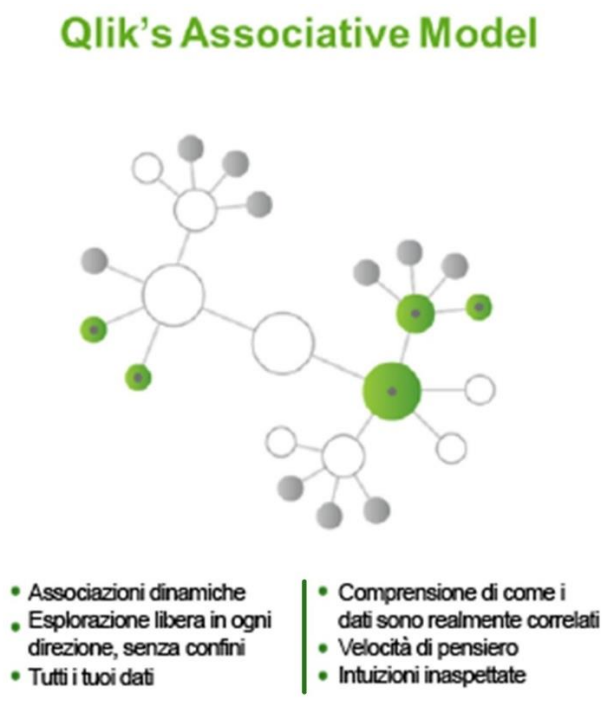

Figura 6.2: Modello associativo

Il ciclo "ask, wait, answer" si trasforma in "ask, get instant feedback, evaluate results".

Avendo il set di dati completo, la risposta è immediata e così l'esplorazione e l'analisi diventano un processo continuo, fatto di domande consequenziali e comprensione. Più gli utenti fanno domande, più, spinti dall'intuizione, aggiungono contesto e più diventano informati, acquisiscono conoscenza.

Figura 6.3: Ciclo "ask, get feedback, evaluate"

Nella figura sottostante è possibile osservare, in maniera sintetica e tabellare, i punti di forza e le discrepanze tra i due modelli.

Entità Diagramma / Relazioni DB **Relazionale**		Entità Diagramma / Relazioni **Associativo**	
Join Predefinite	Schemi complessi	Schemi semplificati	Associazioni dinamiche
Livelli pre-aggregati	Queries complesse	Queries semplificate	Navigazione libera
Analisi predefinite	Comprensione complessa	Immediata comprensione DM	Correlazioni semplificate
Visione scenari parziali	Data storage efficiente	Basso livello di normalizzazione	Esplorazione dei dati completa

Figura 6.4: Relazionale vs Associativo

6.1.2 Pro e contro delle diverse tipologie di schema

Una volta compresa la potenza e l'efficienza del modello associativo, occorre definire la migliore struttura. Sostanzialmente gli schemi realizzabili sono tre: single table, star schema, snowflake schema. Nei passaggi successivi, discutendo delle caratteristiche di queste tre possibilità, si daranno per assodati i concetti di fatti e dimensioni: per riprenderli si rimanda al sotto-paragrafo 5.2.2.

Single table vuol dire disporre di una sola enorme tabella contenente tutta la nuvola dati.

Nello star schema una tabella dei fatti è associata con numerose dimensioni e riflette,

pertanto, una stella. Le tabelle dimensionali stesse sono costituite da gerarchie di dimensioni e contengono, quindi, un insieme di attributi. Lo schema a stella è, così, altamente denormalizzato. I fatti contengono le chiavi esterne che consentono di relazionarsi alle dimensioni. In uno schema a stella solo un collegamento stabilisce la relazione tra la tabella dei fatti e una qualsiasi delle tabelle delle dimensioni e, pertanto, l'architettura è a un solo livello. Il vantaggio dello schema a stella è il ridimensionamento, l'aumento delle prestazioni e una facile comprensione dei dati.

Lo snowflake schema è sostanzialmente un'estensione dello schema a stella. Il termine fiocco di neve descrive, infatti, una struttura di schema a stella normalizzata. In questo modello le tabelle delle dimensioni non sono necessariamente completamente appiattite. Per cui le tabelle dimensionali molto grandi vengono normalizzate in più tabelle sub-dimensionali (dimensioni secondarie) creando una gerarchia. Queste gerarchie consentono di eseguire il drill-down dei dati dalle gerarchie più in alto alle gerarchie più in basso. Avendo un'architettura a più livelli, la latenza di accesso ai dati è superiore nello snowflake schema rispetto allo star schema. Di contro, però, poiché lo schema a stella viene denormalizzato, lo spazio occupato in memoria dallo snowflake sarà minore dello star. Lo schema a fiocco di neve si utilizza, solitamente, quando le tabelle dimensionali diventano molto grandi.

Mentre l'idea della single table è sicuramente da scartare perché è troppo dispendiosa a livello di consumo di RAM ed ha una pessima efficienza a run-time, la scelta tra star schema e snowflake schema dipende dai requisiti aziendali e dalle strutture delle tabelle. Nella figura sottostante è possibile vedere graficamente le principali differenze in termini di performance, complessità e flessibilità delle tre possibili soluzioni.

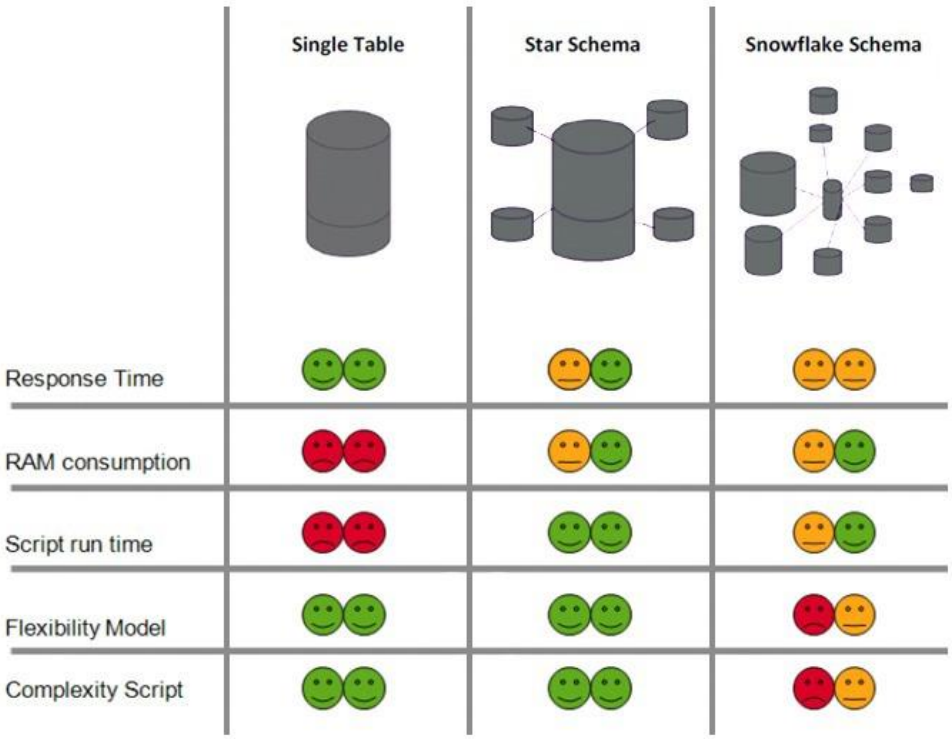

Figura 6.5: Confronto tra le varie tipologie di schema

Generalmente è consigliato gestire le relazioni gerarchiche di molti a uno in una tabella dimensionale singola anziché a più livelli. Lo schema a fiocco di neve, oltre a creare complessità e confusione per gli utenti direttamente esposti alle strutture delle tabelle, non è ottimale per le prestazioni delle queries in quanto, per risolverle, bisogna collegare più tabelle. Lo snowflake, inoltre, impone la gestione delle chiavi che collegano le tabelle normalizzate. Questo processo può diventare estremamente complesso quando le relazioni gerarchiche collegate sono soggette a modifiche. D'altro canto, lo schema a fiocco di neve può risparmiare spazio sostituendo stringhe di testo ripetute con i codici: i guadagni, però, sono trascurabili, specialmente alla luce del prezzo pagato a livello di script per la complessità delle queries. Per quanto riguarda le prestazioni, lo schema a stella è quindi migliore ma, se consideriamo la memoria, è lo snowflake a risultare vittorioso. In linea generale, quindi, le best practices Qlik stabiliscono che gli sviluppatori dovrebbero modellare i dati in uno schema a stella, per la massima efficienza e velocità. La morale della storia è, infatti, che tanto più Qlik deve saltare da una tabella all'altra mentre si esegue un calcolo, tanto maggiore è lo sforzo richiesto. Quindi perché non avere una single table? Ovviamente, si tratta di ottenere il giusto equilibrio tra memoria e CPU. Per applicazioni molto piccole probabilmente andrebbe pure bene avere una singola tabella ma, una volta che l'applicazione inizia a ingrandirsi, si scopre che l'ottimizzazione è una priorità.

Le regole generali sono:

- una singola grande tabella consuma un quantità esosa di RAM;
- il tempo di risposta nello schema snowflake può essere lento poiché Qlik deve svolgere un lavoro più impegnativo. Si aumenta, infatti, il costo di ricostruzione dell'informazione. Lo schema snowflake è quindi normalmente sconsigliato. La riduzione di spazio occupato è, infatti, scarsamente benefica, in quanto l'occupazione maggiore di memoria è dovuta alla tabella dei fatti;
- uno star schema è un buon equilibrio (ma non sempre perfetto).

In linea di massima, quindi, Qlik performa meglio in un modello a stella che presenta una struttura denormalizzata. Questo è dovuto principalmente alla brevità delle relazioni tra le tabelle (data links) che ottimizza il calcolo nelle espressioni.

6.1.3 Come comportarsi in presenza di più tabelle dei fatti?

Spesso, quando ci si imbatte in progetti complessi, ci si trova a dover avere a che fare con una grossa mole di dati e più eventi da gestire. In questo caso occorre valutare bene le scelte possibili e i loro impatti nelle performance e nella comprensione del modello. La decisione sarà quindi totalmente dipendente dalla tipologia e dal volume dei dati trattati ma anche dal contesto. Per gestire più fact tables abbiamo tre opzioni: join, concatenate, link table.

L'operazione di join prende due tabelle e le unisce in una. I records della tabella risultante sono combinazioni dei records presenti nelle tabelle originali associati per valori comuni di campi omonimi, formando quella che può essere definita un'unione naturale. I prefissi Left, Right, Inner possono essere usati per limitare i dati. Il problema è che ciò taglia parte

dei contenuti di una o più tabelle. Si dovrebbe allora operare un outer join che consente di mantenere tutti i dati ma porta a generare tabelle di grandi dimensioni che rallentano il caricamento. Inoltre, se sono presenti dei duplicati, c'è la possibilità di generare inavvertitamente nuovi records. Non si possono neanche definire alternative nel caso di mancato match tra le due tabelle. Infine, si perde il riferimento originario agli oggetti, quindi logicamente il modello risulta più confusionario. Per cui il join è adatto ad aggiungere campi a records già esistenti ma è sconsigliato usarlo per unire più tabelle dei fatti.

La "Concatenate" è un'istruzione che forza la concatenazione con una tabella esistente specificata o con l'ultima tabella logica creata. In linea di principio realizza la stessa operazione dell'istruzione SQL "UNION", ma con due differenze: innanzitutto il prefisso "Concatenate" può essere utilizzato indipendentemente dal fatto che le tabelle abbiano nomi di campo identici o meno; in secondo luogo non vengono rimossi i duplicati cosicchè il numero di records nella tabella risultante sarà la somma dei records nelle tabelle originarie. Per di più, se due o più tabelle caricate presentano esattamente gli stessi campi, Qlik Sense concatenerà automaticamente il contenuto delle tabelle. È possibile prevenire ciò ponendo, tra il load di una tabella e l'altra, l'istruzione "NoConcatenate". Quindi, in sintesi, è consigliabile usare la concatenate quando le fact tables hanno struttura identica.

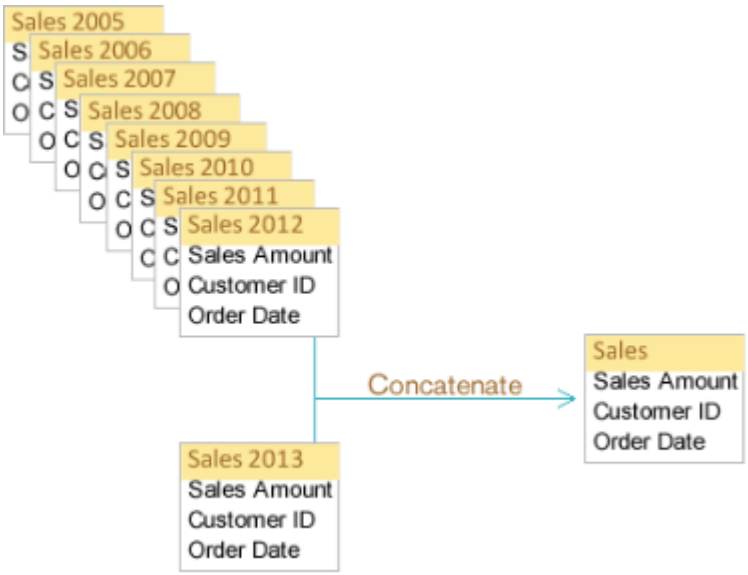

Figura 6.6: Esempio di concatenate

È pensabile sfruttare questa soluzione anche quando ci sono tabelle dei fatti con una granularità mista: i dati presenti riguardano lo stesso evento ma sono a livello di dettaglio diverso. Ad esempio, si specifica da una parte il prodotto, il cliente e la data d'acquisto, dall'altra la categoria del prodotto, la regione di appartenenza del cliente e il mese. In questo caso la soluzione consiste nel concatenare le tabelle in una tabella dei fatti comune e utilizzare chiavi generiche per le dimensioni. La concatenazione avverrà dopo aver costruito i records opportunamente. Nello specifico, se si tratta di una riga della tabella a

granularità più fine, per ogni campo si dettaglierà il valore e il relativo gruppo di appartenenza mentre, se si tratta di un record della tabella a granularità più grossolana, basterà specificare solamente il valore. Le generic keys saranno le chiavi che collegheranno con le dimensioni e saranno definite in modo che i loro valori possano rappresentare sia le singole chiavi che gruppi di chiavi o qualsiasi chiave. In questa maniera è possibile creare un modello di dati più flessibile risolvendo le problematiche di modellazione nel contesto di mixed granularity.

L'ultima soluzione è quella della link table. Questa è una tabella di congiunzione contenenti i campi comuni provenienti da due o più tabelle. Per costruire una link table occorre procedere in questo modo: caricare le fact tables concatenando i campi in comune per costruire un campo chiave ma evitando di tirare su i singoli attributi; creare la link table caricando i valori comuni distinti dalle tabelle dei fatti e costruendo la chiave composta; caricare le dimensioni. L'uso delle link tables ha anche il vantaggio di mantenere separate le tabelle dei fatti, perciò il modello resterà auto-esplicativo e facilmente comprensibile. Questa soluzione presenta, però, anche degli svantaggi: non è banale da implementare in quanto, come riportato sopra, richiede più passaggi e, allo stesso modo, non è neanche facile da manutenere. Per di più, creare una link table vuol dire aggiungere un nuovo elemento al data model il che significa addurre un carico maggiore sia nella fase di ricalcolo dell'analisi al nuovo contesto sia a livello di memoria. È presumibile, appunto, che la link table sia una tabella piuttosto ingente. Infatti, da una prospettiva logico-matematica, il collegamento delle tabelle è identico ad un outer join ma mantenendo staccate le tabelle.

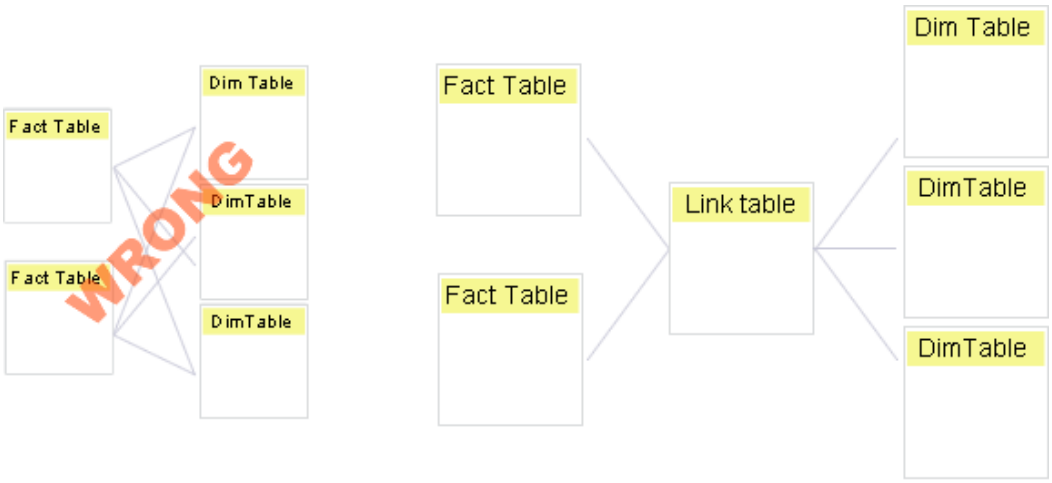

Figura 6.7: Esempio di link table

Quindi, riassumendo:

- il join è adatto al consolidamento di dati provenienti da più tabelle in una sola tabella finale ma si rischia di generare inavvertitamente nuovi records e, a meno di applicare un outer join che produrrebbe una tabella finale di grandi dimensioni e un modello meno intuitivo, di tagliare delle righe. Per tali ragioni questa possibilità non è adatta alla connessione di fact tables;

- la concatenazione dovrebbe essere utilizzata se si hanno tabelle contenenti fondamentalmente lo stesso tipo di entità ma diversi data sets. L'approccio è semplicistico e le prestazioni eccellenti;
- se, invece, si ha una situazione in cui le fact tables contengono eventi differenti e una selezione in una delle tabelle dovrebbe implicare la riduzione a un sottoinsieme di records nelle altre tabelle, allora dovremmo collegare i fatti tramite link tables. Sfruttando questa soluzione il modello dati resta più comprensibile.

Per cui, quando ci sono più tabelle dei fatti, ciascuna con più dimensioni, l'opzione più efficiente è consolidare tutti i fatti in una tabella, se possibile. Se questo non è realizzabile, occorre usare una o più link tables e ciò porterà come risultato uno star schema modificato o un modello dati a fiocco di neve. Nella figura sottostante sono riportati schematicamente vantaggi e svantaggi delle tre possibili soluzioni.

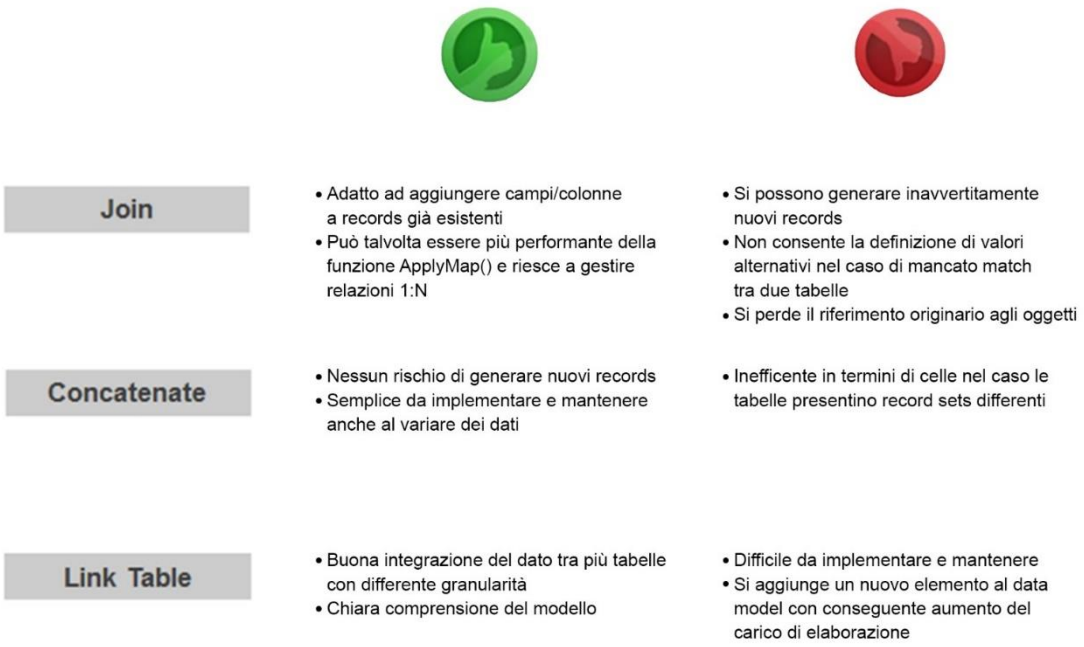

Figura 6.8: Join vs Concatenate vs Link Table

6.1.4 Best practices

Tra le regole generali da seguire, ce ne sono alcune raccomandate mentre altre sono obbligatorie in quanto servono a garantire il corretto funzionamento del modello associativo.

Sono sicuramente da evitare i riferimenti circolari, cioè quando nei dati sono presenti dei loop. In questo caso le tabelle vengono legate in modo che esistano più percorsi di associazioni tra due campi e ciò potrebbe comportare ambiguità di interpretazione degli stessi. Qlik Sense, per preservare la correttezza dei suoi meccanismi, risolve il problema

dei riferimenti circolari interrompendo il loop con una tabella logicamente disconnessa, che è solitamente la tabella più lunga. Così facendo, però, il modello effettivo sarà diverso dal modello logicamente concepito e le selezioni non impatteranno come dovuto. Per gestire opportunamente i riferimenti circolari occorre, allora, modificare lo script di caricamento dei dati assegnando un nome univoco a uno dei campi omonimi. Questo può essere realizzato con l'aliasing o preponendo al load statement l'istruzione "Qualify" tramite cui si aggiunge il nome tabella come prefisso ai diversi nomi campo.

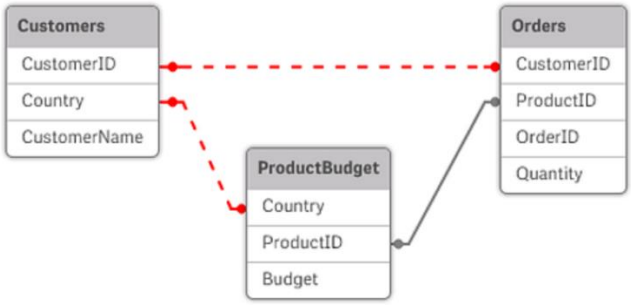

Figura 6.9: Esempio di riferimento circolare

Altra pratica da rispettare è quella di eliminare le chiavi sintetiche. Quando due o più tabelle di dati hanno due o più campi in comune, questo indica una relazione di chiavi composite. Qlik Sense gestisce questa condizione mediante la creazione automatica di chiavi sintetiche. Queste chiavi sono campi anonimi che rappresentano tutte le combinazioni ricorrenti della chiave composita. La presenza di più chiavi sintetiche indica spesso un modello dati non corretto, anche se non necessariamente. In generale, è sempre buona norma evitarle. Bisogna prima verificare che i campi omonimi non abbiano in realtà ruoli differenti (ad esempio "Codice", "Descrizione" o "Data" potrebbero indicare cose diverse a seconda della tabella di appartenenza). Occorre poi rinominare i campi comuni in maniera univoca e, se necessario, creare le chiavi composte o concatenando semplicemente i vari attributi o usando la funzione "Autonumber" che genera un numero intero in base all'ordine di lettura dei dati dalla fonte. Se si necessita di utilizzare chiavi permanenti indipendentemente dall'ordinamento dei dati sorgente, è opportuno utilizzare le funzioni "hash128", "hash160" o "hash256".

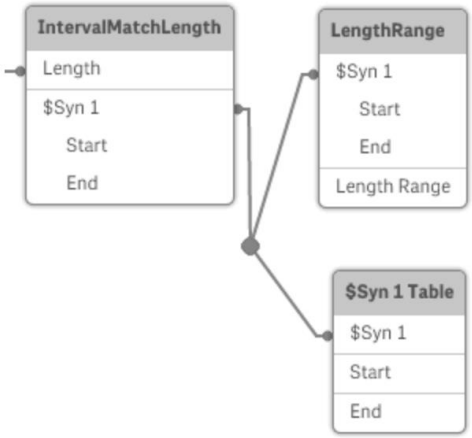

Figura 6.10: Esempio di chiave sintetica

Per non sovraccaricare inutilmente il sistema durante il caricamento e la memoria in utilizzo durante l'esecuzione, è consigliato estrarre solo i campi realmente necessari ed evitare dettagli inutili aggregando al livello opportuno.

Infine, in molti casi è possibile risolvere un task, per esempio delle aggregazioni, costruendo un modello dati più ricco nello script di caricamento o eseguendo le aggregazioni direttamente nelle espressioni dei grafici. Come regola generale, si possono riscontrare le prestazioni migliori quando le trasformazioni sono fatte a back-end.

6.2 La costruzione del modello

Approfondite le tecniche da perseguire per operare in maniera corretta nel data modeling, occorre passare dalla teoria alla pratica e applicare quanto detto per la realizzazione del modello dati dell'app commissionata.

Le funzionalità native di Qlik permettono che il data model venga automaticamente definito associando le entità tramite la definizione di relazioni tra campi omonimi. Tuttavia, con l'aumentare del volume dei dati e al crescere del numero di oggetti coinvolti, anche il modello evolve in complessità. È importante quindi comprendere come gestire e manipolare le problematiche al fine di definire un modello dati Qlik che possa soddisfare la user experience finale ottimizzando performance ed efficienza. Oltre a poter constatare tempi di risposta lenti, il problema principale di un data model mal congegnato è quello di dare origine a visualizzazioni inesatte, fornendo informazioni parzialmente corrette o, peggio, totalmente sbagliate. Organizzare correttamente i dati all'interno di Qlik è il segreto del successo. Nel mondo della BI vige la regola dell'80/20: l'80% dello sforzo va ai dati, il 20% all'interfaccia utente. Questo precetto, che può sembrare solo un cliché, si concretizza nella pratica dal momento che, se i dati sono strutturati correttamente, la creazione degli oggetti dell'interfaccia utente nel front-end di Qlik diventa semplice; risulta facile aggiungere e modificare grafici ed espressioni per ottenere quanto desiderato. Se, invece, i dati sono articolati male, si inizia a scoprire che le espressioni non funzionano come previsto, all'improvviso si è bloccati su un quesito che

dovrebbe essere molto semplice e tutto il tempo risparmiato al momento dell'importazione dei dati lo si perde, così, nella risoluzione dei problemi.

Come definito nel paragrafo 4.2, si è scelto di suddividere l'analisi del flow dal resto dell'app. Per tale ragione si è dato origine a due modelli: uno più complesso che riguarda la "Scheda Accordi Quadro", tema delle prossime pagine e su cui sarà focalizzata la maggior parte dell'attenzione, e uno più semplice concernente il flow che sarà invece trattato in un sotto-paragrafo distinto.

6.2.1 Passi da muovere

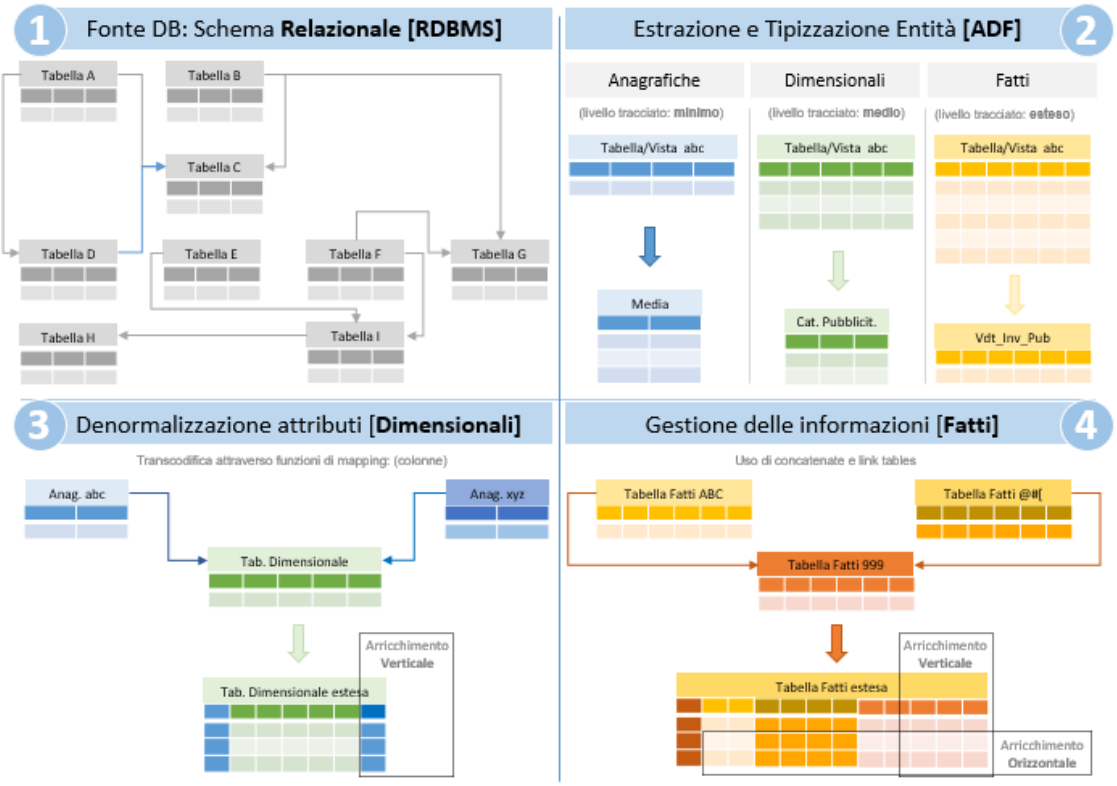

Figura 6.11: Dal modello relazionale a quello associativo

Nel capitolo precedente, discutendo del reperimento dei dati, si è scoperto che le fonti sorgenti sono tre: il data warehouse aziendale; l'istanza del database commerciale per ciò che non è ancora stato ribaltato sul DWH; dei file Excel estemporanei per dati non ancora presenti nel patrimonio informativo aziendale. Nel paragrafo 5.2 è stato illustrato il processo che ha condotto alla creazione delle viste, fonti originarie per il modello. Con riferimento alla figura sopra riportata, è quindi già stato praticato il passaggio dallo stadio uno allo stadio due dal momento che la realizzazione delle viste ha rappresentato la fase di estrazione dei dati e di tipizzazione delle entità divise per fatti, dimensioni e anagrafiche. Adesso, per proseguire in questa attività, occorre compiere un'operazione di arricchimento verticale sulle dimensioni e una di adeguamento, strutturazione e

congiunzione delle fact tables. I prossimi sotto-paragrafi cominciano proprio a parlare degli interventi apportati sui fatti.

6.2.2 Gestione delle fact tables

Quando si comincia a plasmare un modello dati è conveniente partire dalle tabelle dei fatti, cuore da cui tutto si dirama. Per l'app "Scheda Accordi Quadro", la base è sicuramente rappresentata dal dato interno sugli investimenti, tirato su, come spiegato nel sotto-paragrafo 5.2.3, nella VDT_INV_PUBBLICITARI, fonte della fact table denominata "Investimenti". Oltre ai valori economici, qui sono contenute anche altre misure che, per singolo spot, esplicitano, ad esempio, il numero di secondi in onda e il numero di passaggi. Il dato che indica però se uno spot è trasmesso in una posizione pregiata (prima o ultima) di un break, come chiarito nel precedente capitolo, è un'informazione del mondo TV e radio non ancora ribaltata sul DWH, per cui deve essere estrapolata dal DB commerciale. Sono allora state create due viste, la VDT_INV_PUBBL_POS_BREAK_TV e la VDT_INV_PUBBL_POS_BREAK_RADIO che espongono questo dato, rispettivamente per la TV e per la radio. È dunque possibile arricchire la tabella degli investimenti con questa informazione. La chiave di aggancio è rappresentata dal campo CHIAVE_COMUNICATO che, composto dalla concatenazione di più attributi, dettaglia in maniera univoca uno spot e, pertanto, è singolo per ogni record. Verificare l'univocità del campo di congiunzione è essenziale per evitare di cadere nell'errore di creare righe duplicate nell'operazione di join. L'intervento apportato è stato quindi quello di concatenare prima le due viste provenienti dall'istanza del DB commerciale e poi metterle in join con la VDT_INV_PUBBLICITARI. Nello specifico si è sfruttato un left join, in modo da assicurarci che la cardinalità della VDT_INV_PUBBLICITARI resti invariata.

Una delle best practices suggerite è quella di limitare il caricamento esclusivamente a quanto strettamente necessario, estraendo solo i campi utili e aggregando al livello opportuno. Per questo motivo, dal momento che la vista contiene anche campi non richiesti e che il dettaglio massimo è per soggetto ma il punto di vista della nostra analisi vuole essere per cliente, l'operazione successiva è quella di applicare una clausola di group by. Nel fare ciò, si tiene conto delle prospettive di osservazione che si vogliono fornire. Si raggruppa quindi per: codice cliente, codice holding, codice raggruppamento mezzi Nielsen, codice rete Nielsen, ID evento, ID prodotto pubblicitario, ID prodotto editoriale, anno, mese, year to date (YTD). Riguardo il codice holding bisogna puntualizzare che, giacché si vuole esporre, come spiegato nella costruzione dell'anagrafica delle holding, per ogni cliente l'holding di appartenenza o se stesso nel caso non faccia parte di nessun raggruppamento, questo campo è costruito tenendo conto di questa scelta. Anno e mese sono estrapolati dalla data trasmissione mentre l'year to date specifica se giorno e mese di caricamento sono posteriori ai corrispettivi valori della data di trasmissione. L'YTD è essenziale nel confronto tra anni dal momento che comparare l'anno corrente con anni precedenti avrebbe poca valenza senza tener conto della data attuale. Quando si applica una condizione di group by occorre poi, ovviamente, usare degli operatori di aggregazione sulle misure. In questa circostanza si impiega solo la sum, a volte direttamente, a volte per dare origine a nuove misure calcolate. È questo

il caso del lordo sconto cliente, dato dalla somma del netto con lo sconto commerciale, o del campo IMP_OMAGGI che è uguale all'importo sconto commerciale nel caso il netto abbia valore non positivo, zero altrimenti.

Oltre al dato interno, l'altra grande richiesta riguarda il dato di mercato, esposto dalla VDT_INV_MERCATO e indispensabile a realizzare i confronti con gli altri concessionari. Questa vista dà vita alla tabella dei fatti denominata, appunto, "Mercato". Anche in questo caso, nell'ottica di attuare quanto consigliato dalle best practices e ottimizzare i processi, si estraggono solo i campi indispensabili e si aggrega a livello di dettaglio opportuno. Precisamente si raggruppa per: codice cliente, codice cliente Nielsen, codice holding, codice raggruppamento concessionari Nielsen, codice raggruppamento mezzi Nielsen, codice rete Nielsen, codice settore Nielsen, flag caricamento, anno e mese. Sul codice holding si attua lo stesso stratagemma impiegato nella VDT_INV_PUBBLICITARI. Il flag caricamento invece rappresenta il tipo periodo Nielsen e, per questo, se ha valore pari a uno, vuol dire che il periodo è consolidato, se è uguale a due significa che il periodo è EIS. L'uso della group by implica l'applicazione degli operatori di aggregazione sulle misure. Il netto Nielsen sarà dato dalla somma dei singoli valori della misura corrispondente, così come il netto marketing Nielsen. Considerando che è noto che il netto Nielsen si riferisce al periodo consolidato e quello marketing Nielsen al periodo EIS, viene creata una nuova misura denominata IMP_NETTOAQ_NLS che somma il netto Nielsen se il flag di caricamento ha valore uno (quindi il periodo è consolidato), il netto marketing Nielsen altrimenti.

La prima idea balzata alla mente è stata quella di fondere queste due tabelle dei fatti in una sola. Il pensiero nasce dalla riflessione che il dato di mercato fornito dalla Nielsen contempla anche il concessionario Rai Pubblicità. Scendendo nell'analisi c'è, però, un problema di fondo: la rilevazione della Nielsen non è sempre puntuale (cambiano infatti il numero di passaggi), quindi il dato rappresenta una stima, non c'è la garanzia dell'esattezza dei valori. Nella "Mercato" si potrebbe allora sostituire quanto concerne Rai Pubblicità con il dato interno, sempre rigoroso e preciso. Qui sorge tuttavia un altro impedimento. Il dato provvisto dalla Nielsen è a livello di marca, il dato Rai Pubblicità è per soggetto. Marca e soggetto designano entrambi il prodotto pubblicizzato ma le anagrafiche sono differenti e il mapping non è esatto, il rapporto è, infatti, N:M. Se marca e soggetto potessero essere sempre correttamente mappati potremmo agire, ma questo non è sempre garantito. Per cui tale idea è stata abbondonata e si è scelto di mantenere separate queste due tabelle dei fatti e adottare la strategia della link table per congiungerle. Dal momento che il dato interno deve essere confrontato con quello di mercato, la tabella "Investimenti" è stata arricchita con due nuove colonne: la prima, settata staticamente, indica che il codice raggruppamento concessionari Nielsen è quello di Rai Pubblicità; la seconda, tramite ApplyMap (funzione di script usata per mappare l'output di un'espressione a una tabella di mapping caricata in precedenza), andando a leggere dalla "Mercato", determina se il mese e l'anno considerati fanno parte del periodo consolidato o del periodo EIS delineando il tipo periodo Nielsen del record vagliato.

Le altre fact tables derivano dagli Excel forniti. Il dato sul GRP viene estrapolato tramite un foreach dalla serie di file provvisti aventi denominazione "Grp e Grp 30##YYYYMM##HOLDING X". Vengono lette tutte le colonne e, utilizzando una SubField, si estrae dal titolo del documento il nome della holding cui quei dati si riferiscono. Si apportano poi dei mapping allo scopo di uniformare il contenuto con il

resto della nuvola dati. Sui fatti devono essere presenti solo i codici e gli ID, non le descrizioni che per definizione stanno nelle dimensioni. Per questo, il nome holding viene mappato con il corrispettivo codice utilizzando la VAN_HOLDING. La colonna CHANNEL può assumere i valori "RAI GEN" o "RAI SPEC", quindi identifica, in pratica, il raggruppamento mezzi Nielsen. Tramite un'ApplyMap che sfrutta una tabella caricata inline, si passa così da "RAI GEN" e "RAI SPEC" ai corrispettivi codici raggruppamenti mezzi Nielsen identificanti TV generaliste e TV specializzate. Lo stesso tipo di operazione viene compiuta per il mese che, da letterale, deve essere convertito in formato numerico. Vengono inoltre aggiunte due colonne: la prima, dal momento che i dati sono interni, specifica il codice raggruppamento concessionari Nielsen impostandolo staticamente a quello di Rai Pubblicità; la seconda, invece, serve a precisare il tipo target. Se il target del record considerato è "Adulti 25-54" o "Adulti 35-64" significa che è uno dei target di riferimento per la nostra analisi. Altrimenti, utilizzando il file "Target Clienti Accordi Quadro.xlsx" si va a verificare se la coppia HOLDING-TARGET della riga presa in atto è contenuta all'interno del documento. Ciò significa che quel target è definibile come un "Target Reale Pianificazione Cliente", cioè rappresenta il gruppo cui mira prevalentemente far arrivare il proprio prodotto quella determinata holding. In tutti gli altri casi, il target viene classificato come "Altro Target".

Per quanto riguarda il dato sulla Viewability, questo viene estrapolato dal file "VIEWABILITY CLIENTE SETTORE##201706". Si leggono tutte le colonne e se ne aggiungono, staticamente, due: il codice raggruppamento concessionari Nielsen e il codice raggruppamento mezzi Nielsen impostati, rispettivamente, a 1196 (codice di Rai Pubblicità) e 4 (codice identificante il mezzo internet, essendo la viewability un dato web). Dal momento che, come riportato nel paragrafo 5.1, l'anagrafica presente in questa fonte è quella ComScore, tale tabella dei fatti è dissociata dal resto del modello. Per ovviare a ciò è stato predisposto il file "Decodifica Clienti Viewability.xlsx" che, ad ogni cliente ComScore riportato nel documento sulla viewability, associa la corrispettiva holding Rai Pubblicità. Quello che viene praticato, dunque, è un'ApplyMap in cascata che, sfruttando il file appena descritto, mappa prima il cliente ComScore con la relativa holding Rai Pubblicità e poi la ragione sociale della holding con il corrispondente codice. In questo modo si aggiunge un nuovo campo che consente di associare i dati qui presenti al resto della nuvola.

Il lavoro necessario a predisporre la tabella dei fatti contenente i valori di benchmark di CPG è più ostico. I file forniti con denominazione "TV_CPGTREND##YYYYMM" suddividono infatti i contenuti in quattro fogli, scorporati per concessionario (Rai o Mediaset) e mezzo (TV generaliste e TV specializzate). Occorre allora leggere tutte le colonne da ciascuno dei fogli e, poggiandosi a una tabella inline, aggiungere due ulteriori campi che, a seconda del nome del foglio, espongono il corretto concessionario e mezzo. Usando poi, come nei casi precedenti, l'ApplyMap, si sostituisce alla descrizione il codice così da avere il codice raggruppamento concessionari Nielsen e il codice raggruppamento mezzi Nielsen. L'ultima operazione da realizzare riguarda le due colonne "Adulti 25-54 Grp's 30'"" e "Adulti 35-64 Grp's 30'"", già rinominate come "Adulti 25-54" e "Adulti 35-64", che indicano il dato di GRP 30'' per i due target di riferimento. Esiste, quindi, un'unica misura (il GRP ri-parametrizzato a 30 secondi) osservata da due differenti punti della stessa dimensione. Per questo è bene cambiare la struttura e passare ad avere due colonne, una indicante la misura e l'altra denotante la prospettiva. In ciò viene in aiuto la

funzione “CrossTable” che consente di trasformare una tabella incrociata in una tabella lineare. Ciò significa che una tabella larga con diverse colonne viene trasformata in una tabella alta, in cui le intestazioni delle colonne vengono inserite in una colonna con attributo singolo. Occorre specificare il campo che conterrà i valori di attributo (lo chiameremo TARGET_30 e potrà comprendere solo “Adulti 25-54” o “Adulti 35-64”), il campo che conterrà i valori dei dati (GRP_30) e il numero di campi qualificati che precede la tabella da trasformare in un formato generico (nel nostro caso i campi da escludere dalla conversione sono i primi cinque).

Alle fine di queste operazioni il quadro che si delinea è quello di cinque fact tables: “Investimenti”, “Mercato”, “Grp”, “Viewability” e “Cpg 30”. Nel prossimo paragrafo vedremo come metterle insieme.

6.2.3 Creazione delle link tables

Allo stato attuale il modello ci appare come nella figura sottostante.

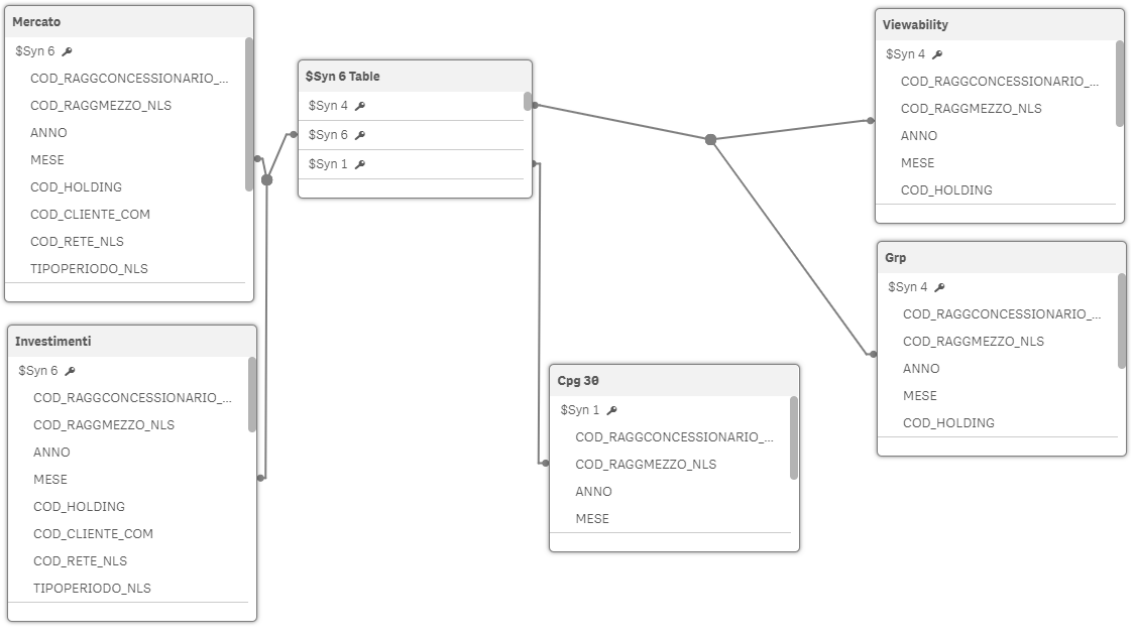

Figura 6.12: Modello con le sole fact tables

Fondamentalmente tra le cinque fact tables c’è una synthetic table contenente le tre chiavi sintetiche generate. Queste nascono perché tra le tabelle dei dati ci sono più campi in comune, quindi c’è una relazione di chiavi composite. La synthetic table conterrà pertanto anche gli attributi condivisi. Le chiavi sintetiche nate sono tre in quanto sono tre le combinazioni ricorrenti della chiave composita: una riguarda i campi in comune tra la tabella “Investimenti” e quella “Mercato”, un’altra si riferisce all’insieme di attributi collettivi di “Grp” e “Viewability” e, l’ultima, concerne i campi condivisi della tabella “Cpg 30”. Avere delle synthetic keys è spesso sintomo di un modello mal strutturato.

Connettere tutte le fact tables in questo modo è infatti profondamente sbagliato perché si collegano fatti a granularità differente allo stesso livello generando una synthetic table che, incorporando tutte le possibili combinazioni degli attributi comuni, sarà ovviamente molto ingente. Analizziamo adesso le possibili soluzioni da applicare in uno scenario come questo.

Creare un'unica tabella dei fatti è escluso; significherebbe riunire più eventi insieme il che è un errore sia a livello di performance, perché vorrebbe dire far esplodere la cardinalità e saturare la RAM, sia a livello concettuale perché è buona norma tenere separati fatti distinti. Sostanzialmente quello che abbiamo davanti sono cinque tabelle aventi tre livelli di granularità differenti: la "Investimenti" e la "Mercato" che scendono a dettaglio di cliente; "Grp" e "Viewability" che, invece, comprendono dati a livello di holding; "Cpg 30" che, contenendo i valori di benchmark, è di riferimento per tutti i clienti. Nel sotto-paragrafo precedente sono state illustrate le motivazioni per cui non è possibile generare un'unica tabella dei fatti a partire dalla "Investimenti" e dalla "Mercato". Usare una "Concatenate", dal momento che i campi esposti sono in una certa misura discordanti, è ugualmente sconsigliato. Per cui la soluzione migliore è quella di impiegare una link table che associa queste due tabelle ed espone gli attributi comuni. Lo stesso ragionamento può essere ripetuto per "Grp" e "Viewability" che, essendo fatti indipendenti tra loro osservati da prospettive in parte diverse, non hanno motivo di essere concatenati, ma è meglio relazionarli tramite una tabella di congiunzione. Un'ultima link table è infine utile a contenere le dimensioni collettive a tutti i fatti.

Quindi lo schema risultante sarà composto, oltre che dalle cinque fact tables finora descritte, anche da tre link tables che, connesse tra loro, consentono di collegare opportunamente i fatti evitando la generazione di chiavi sintetiche. Le tre tabelle di congiunzione si chiameranno rispettivamente: "Link Investimenti-Mercato", utile ad associare la "Investimenti" e la "Mercato" e che scende a livello di dettaglio cliente; "Link Grp-Viewability" che unisce "Grp" e "Viewability", con aggregazione per holding; "Link Cpg Benchmark", cui è connessa la tabella "Cpg 30", per le dimensioni comuni.

Per costruire in maniera corretta una link table il primo passo è quello di generare la chiave composta. Questa può essere assemblata concatenando i campi comuni tra le tabelle dei fatti. Tra "Investimenti" e "Mercato" gli attributi che ritroviamo in entrambe le tabelle sono: il codice cliente, il codice holding, il codice raggruppamento concessionari Nielsen, il codice raggruppamento mezzi Nielsen, il codice rete Nielsen, l'anno, il mese, il tipo periodo Nielsen. La chiave così creata, denominata KEY_LINK_TABLE_INV_MER, sarà data dalla concatenazione di tutti questi campi separati dal carattere '-'. Viene solo escluso il codice holding dal momento che, essendo l'aggregato a granularità più grossolana del codice cliente, è superfluo nella garanzia dell'univocità dei records. La KEY_LINK_TABLE_GRP_VW sarà invece composta dai campi comuni tra la tabella "Grp" e la tabella "Viewability": codice holding, codice raggruppamento concessionari Nielsen, codice raggruppamento mezzi Nielsen, anno e mese. Per ultimo KEY_LINK_TABLE_BENCHMARK_CPG conterrà, divisi da '-', codice raggruppamento concessionari Nielsen, codice raggruppamento mezzi Nielsen, anno e mese, cioè i campi comuni a tutt'e cinque le fact tables.

Definite le chiavi, occorre creare le link tables e modificare le tabelle dei fatti cancellando i campi comuni che saranno ora esposti dalle tabelle di congiunzione. La "Link

Investimenti-Mercato" sarà costituita dalla concatenazione dei records distinti della "Investimenti" e dalla "Mercato" afferenti i campi KEY_LINK_TABLE_INV_MER, KEY_LINK_TABLE_GRP_VW, codice cliente commerciale, codice rete Nielsen e tipo periodo Nielsen. La KEY_LINK_TABLE_INV_MER consente l'associazione con le due tabelle dei fatti appena menzionate, la KEY_LINK_TABLE_GRP_VW permette invece di definire il collegamento con la link table a granularità meno fine, mentre gli altri attributi sono tutti e soli quelli comuni esclusivamente a "Investimenti" e "Mercato" ma non alle altre fact tables. La "Link Grp-Viewability" sarà costruita allo stesso modo, ma prelevando i dati non solo dalla "Grp" e dalla "Viewability" ma anche dalla "Investimenti" e dalla "Mercato" poiché, essendo a granularità maggiormente grossolana, deve consentire l'accesso alle tabelle a granularità più fine senza tagliare nessun record. Conterrà la KEY_LINK_TABLE_GRP_VW, la KEY_LINK_TABLE_BENCHMARK_CPG e il codice holding. Infine, la "Link Cpg Benchmark", essendo la tabella di congiunzione a livello di aggregazione massimo, sarà realizzata a partire da tutt'e cinque le fact tables e conterrà la KEY_LINK_TABLE_BENCHMARK_CPG, l'ID_CALENDARIO (fabbricato a partire da mese e anno tramite la funzione "MakeDate" e utile a creare la relazione con la dimensione temporale), il codice raggruppamento concessionari Nielsen e il codice raggruppamento mezzi Nielsen.

Nelle tabelle dei fatti dovranno poi essere mantenute le chiavi di congiunzione alla link table corrispondente e i campi propri ma eliminati tutti gli attributi comuni. Questa operazione è imprescindibile, altrimenti non risolveremmo il problema delle synthetic keys. Inoltre, bastano le chiavi composte ad identificare correttamene i records. Infatti la selezione di un valore di un campo comune porterà alla selezione delle chiavi corrispondenti contenenti quel dato valore per quel dato campo e, in questo modo, verranno considerate le righe giuste dalle tabelle dei fatti.

Alla fine di questa operazione, il modello ha acquisito la conformazione mostrata in figura 6.13.

Figura 6.13: Modello con fact tables collegate tramite link tables

Adesso non ci resta altro che aggiungere le dimensioni.

6.2.4 Le dimensioni

Costruiti e relazionati i fatti, occorre modellare anche le dimensioni, cioè le prospettive da cui sono analizzati questi eventi. Alcune dimensioni sono complesse e necessitano di essere esplicitate in tabelle proprie, altre, invece, sono rappresentate da un solo attributo e, per questo, vengono definite come dimensioni degeneri. In questo caso le soluzioni realizzative sono due: o integrare questi campi nelle fact tables o creare delle junk dimensions, cioè insiemi di più dimensioni degeneri non legate da dipendenze funzionali.

La scelta è ricaduta sulla prima possibilità dal momento che questi attributi sono di volume contenuto ed è sempre bene, lì dove possibile, limitare il modello evitando l'inserimento di ulteriori tabelle quando non necessario. In tutti gli altri casi vengono, invece, create nuove tabelle, originate a partire da due tipologie di viste illustrate nel paragrafo 5.2: le VAN, cioè le viste anagrafiche pure contenenti solo codice, descrizione e un eventuale campo di ordinamento di un determinato oggetto; le VDM, cioè le viste dimensionali che inglobano dati provenienti da più anagrafiche aventi legami gerarchici o tematici. Le prime saranno associate ai fatti tramite codice, le seconde tramite chiave (ID). Alcune dimensioni, quelle comuni a più fact tables, saranno connesse alle tabelle di congiunzione, le altre, invece, caratterizzanti determinati avvenimenti, saranno collegate direttamente ai fatti.

Scendiamo adesso in verticale sulle dimensional tables create.

"Eventi" racchiude le informazioni riguardo le manifestazioni e le loro tipologie. Preleva i dati dalla VDM_EVENTO ed è collegata alla "Investimenti" tramite l'ID_EVENTO. Nelle specifiche ricevute sono evidenziati quattro accadimenti (i mondiali e gli europei di calcio, le olimpiadi e il festival di Sanremo) da tenere maggiormente in considerazione. Per questa ragione non ci si è poggiati alla tipologia di evento già delineata sulla vista, ma si è creato un nuovo campo denominato TIPO_EVENTO che assume valore "Evento Speciale", se la manifestazione è una di quelle sopra citate, "Altro Evento" diversamente.

Alla "Investimenti" sono associate poi, tramite chiave, anche le dimensioni "Prodotti pubblicitari" e "Prodotti editoriali" che estraggono i contenuti rispettivamente dalla VDM_PRODOTTO_PUBBLICITARIO e dalla VDM_PRODOTTO_EDITORIALE. È bene prestare attenzione a tirare su solo i campi realmente utili sia per evitare di portarci dietro qualcosa di infruttuoso sia perché potrebbero crearsi dei problemi sul modello. Ad esempio, infatti, la vista del prodotto editoriale espone pure gli attributi codice rete e descrizione rete ma questi due campi saranno contenuti in una dimensione separata per le reti e quindi esibirli anche qui significherebbe creare un riferimento circolare.

Alla "Mercato" sono legati la "Clienti Nielsen" e la "Settori Nielsen". La prima delinea codice e descrizione dell'anagrafica dei clienti secondo Nielsen. La seconda fa lo stesso per i settori andando ad estrarre solo quelli riconosciuti attualmente. Vengono infatti prelevati solamente i records che hanno il flag di validità settato a 'S'. Entrambe le tabelle sono linkate ai fatti tramite codice.

Finora sono state discusse solo le dimensioni proprie di alcuni eventi. Passiamo adesso alle dimensioni comuni, collegate alle link tables. La "Link Investimenti-Mercato", che scende fino al dettaglio cliente, sarà perciò connessa alla "Clienti commerciali", dimensione che dettaglia l'anagrafica dei clienti secondo Rai Pubblicità e che si fonda sulla VAN_CLIENTI_COMMERCIALI. La suddetta tabella di congiunzione sarà legata anche con la dimensione "Reti". Questa tabella è costruita a partire dalla VDM_RETI_RP_RETI_NIELSEN che espone la relazione tra l'anagrafica delle reti Nielsen e quella Rai Pubblicità. Il legame è su base codice rete Nielsen che è chiave per la vista. Infatti ad ogni rete Nielsen può essere associata una o nessuna rete Rai Pubblicità. Questa dimensione è arricchita da un ulteriore campo denominato FL_RETI_AQ tramite cui si evidenziano le reti che presentano nel nome la stringa "SAN MARINO". In questo modo sarà possibile filtrare su questa condizione, andando a realizzare quanto richiesto dalle specifiche, ossia la possibilità di escludere dall'analisi ciò che riguarda San Marino.

Una tra le tante richieste utente è quella di poter avere a disposizione le informazioni concernenti i rapporti di vendita dei clienti. Un cliente può entrare direttamente in contatto con il mondo Rai Pubblicità o dare mandato per un dato mezzo su cui ha intenzione di investire ad un'agenzia. Per questo motivo è stata creata un'ulteriore dimensione chiamata "Portafoglio Agenzie" che espone la relazione corrente tra le agenzie e i clienti seguiti. Considerato che l'associazione vuole essere per cliente, per chiarezza espositiva sulla "Link Investimenti-Mercato" si è preferito duplicare il campo COD_CLIENTE_COM rinominando la copia come KEY_PF_AGENZIE andando così a costituire la chiave di aggancio con la dimensione delle agenzie. Questa è creata a partire dalla VPF_AGENZIE. Sulle viste di tipo VPF la scelta presa è stata quella di portare a bordo esclusivamente i codici che identificano in maniera univoca i dati; in fase di presentazione non possono però essere mostrati i codici. Occorre pertanto fare un arricchimento verticale per impreziosire la dimensione con le descrizioni. A tal proposito si usano delle ApplyMap che, sfruttando delle tabelle di mapping caricate in memoria create a partire dalle relative VAN, mappano ogni codice con la rispettiva descrizione. Oltre alle informazioni sulle agenzie, bisogna aggiungere anche quelle sui venditori. Questi, a seconda dell'interlocutore con cui trattano, possono essere divisi in venditori clienti e venditori agenzia. Sono allora state predisposte due ulteriori dimensioni chiamate rispettivamente "Portafoglio Venditori Clienti" e "Portafoglio Venditori Agenzie" connesse direttamente, la prima a "Clienti commerciali" tramite il codice cliente, la seconda a "Portafoglio Agenzie" per mezzo del codice agenzia. Anche in questo caso i portafogli venditori, basandosi sulle viste VPF_VENDITORI_CLIENTI e VPF_VENDITORI_AGENZIE, esibiranno soltanto i codici per cui è necessario, come fatto prima, compiere un'operazione di arricchimento tramite mapping per annettere anche le descrizioni. Menzione particolare merita il mapping del codice mezzo presente su tutti i portafogli. L'intera analisi è basata sui raggruppamenti mezzi Nielsen, per cui anche in questo caso si preferisce esibire questa informazione e non i mezzi Rai Pubblicità, dato a granularità eccessivamente fine. La questione è che non tutti i mezzi dell'anagrafica interna sono raggruppabili secondo l'anagrafica Nielsen. Ciò è sicuramente possibile per i mezzi per cui ci sono investimenti. Pertanto la decisione ottemperata è stata quella di portare sui portafogli solo i mezzi che compaiono sui fatti per cui è quindi sempre possibile realizzare il mapping. Questo è corretto anche logicamente perché far vedere che un dato venditore si occupa anche di mezzi su cui non ci sono stati investimenti negli ultimi cinque anni è un'informazione assolutamente superflua.

La "Link Grp-Viewability" che trasposta il codice holding sarà connessa, tramite tale attributo, alla dimensione "Holding". Questa tabella è eretta a partire dalla VAN_HOLDING. Ci sono però altre tre informazioni riguardanti le holding non presenti sulla vista: la specifica che consiglia o meno la stipula di un accordo quadro, il settore prevalente di investimento e il target di riferimento. Il primo dato può essere estrapolato dal file Excel "Clienti Accordo Quadro.xlsx" e, tramite ApplyMap, annesso ai campi già presenti. Il secondo, invece, deve essere calcolato. Andando ad operare sulla "Mercato", si raggruppa per holding e settore Nielsen e si sommano gli investimenti. Poi, per ogni holding, si determina, tra i vari settori per cui ha impiegato denaro, il massimo investimento. Infine si va prelevare il settore per cui l'investimento è più alto e quello sarà il settore prevalente per quella determinata holding. Tramite mapping anche questa informazione viene aggiunta alla dimensione "Holding". L'ultimo dato è prelevabile dal

file “Target Clienti Accordi Quadro.xlsx” ma, contrariamente agli altri due, non può essere annesso alla dimensione “Holding” tramite mapping. La ragione di ciò risiede nel fatto che una holding può avere più target di riferimento e quindi se facessimo un’ApplyMap assoceremmo solo uno dei valori, il primo memorizzato per quella holding, cascando così in errore. Ragion per cui viene creata una dimensione secondaria denominata “Target riferimento Holding”.

Alla tabella di congiunzione restante, la “Link Cpg Benchmark”, sono connesse le dimensioni comuni a tutte le facts: “Concessionari”, “Raggruppamento mezzi” e “Calendario”. Le prime due sono costruite a partire rispettivamente dalla VAN_RAGG_CONCESSIONARI_NIELSEN e dalla VAN_RAGG_MEZZI_NIELSEN ed espongono codice, descrizione e un campo ordinamento. Per l’ultima, invece, serve intraprendere un discorso più approfondito, affrontato nel prossimo sotto-paragrafo. Il modello, completo di dimensioni, è visibile nella figura sottostante.

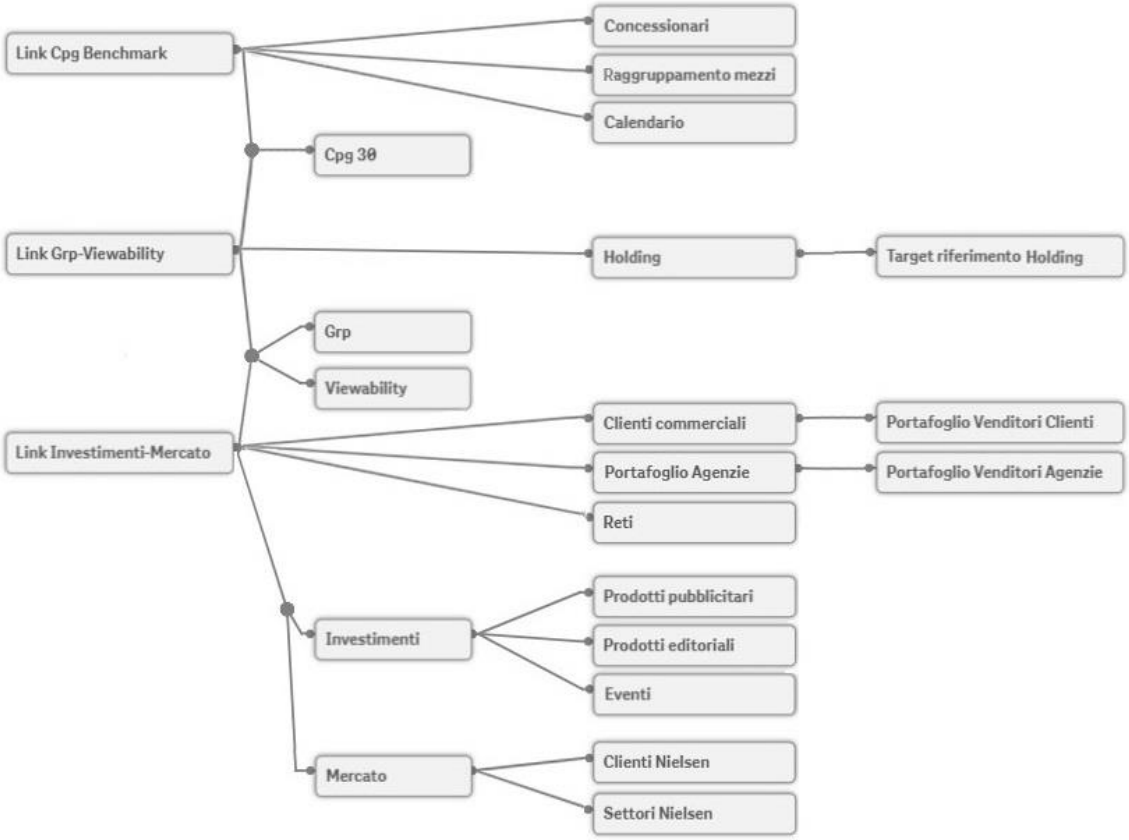

Figura 6.14: Modello compreso di dimensioni

6.2.5 Il master calendar

Per qualsiasi analisi il tempo è un fattore importante. Gli utenti saranno sempre interessati a visualizzare i dati per un determinato periodo e ad eseguire studi comparativi tra due lassi temporali. In un data warehouse solitamente è presente una tabella per la dimensione

tempo che, connessa ai fatti, fornisce il contesto temporale. Nel nostro caso è stata predisposta una vista denominata VTM_CALENDARIO_COM che, però, espone il calendario commerciale. Questo, diversamente dal calendario solare, suddivide gli anni non in mesi ma in cicli e termina le settimane al sabato. Le settimane a cavallo tra due cicli apparterranno interamente al ciclo precedente o a quello successivo a seconda che ci siano più giorni in uno o nell'altro. Il calendario commerciale quindi è utilizzabile per scopi interni di fatturazione ma non è adatto quando ci sono dati provenienti dall'esterno, come nel nostro caso, perché perderebbe di senso. Serve pertanto predisporre una tabella propria che definisca il calendario solare. Non fare ciò sarebbe un grosso errore perché, avendo più fact tables, senza una dimensione temporale comune, la selezione di un periodo su una delle tabelle dei fatti non produrrebbe risultati sulle altre. Inoltre, poiché il nostro obiettivo è anche quello di rendere le selezioni e le aggregazioni il più semplici possibili per i nostri utenti, nella dimensione temporale, oltre alla data originale, possiamo includere attributi come mese, trimestre, semestre ed anno. Non avrebbe senso creare una tabella con un numero sproporzionato di righe tenendo conto di anni concettualmente distanti dalla nostra analisi. È bene allora dare origine alla dimensione temporale partendo dai fatti cosicchè siano presenti solo le date effettivamente utili. Usare però solo le righe delle fact tables può portare a visualizzazioni erronee o ingannevoli. Infatti, se, ad esempio, nel mese di agosto non ci fossero investimenti, in un grafico lineare che ne mostra l'andamento mensile, non vedremmo un picco in giù in corrispondenza di agosto, ma si salterebbe direttamente da luglio a settembre. È chiaro che questo non è il risultato che vogliamo dare. Viene dunque creato un master calendar seguendo dei passi ben precisi in modo da ottenere quanto voluto. Per prima cosa da tutte le tabelle dei fatti si estraggono il primo e l'ultimo giorno dell'anno per ogni record. Poi si va a selezionare il minimo tra i primi giorni dell'anno ed il massimo tra gli ultimi. Queste due informazioni delineano il range di date a cui siamo interessati. Tramite un ciclo while si generano tutte le date comprese tra i due limiti. A partire da queste si crea la tabella del master calendar composta, oltre che dalle date, anche dai rispettivi anni, semestri e mesi, costruiti utilizzando le funzioni di script messe a disposizione da Qlik Sense. In figura 6.15 è possibile osservare parte dello script di generazione del master calendar.

```
StartEndDates:
LOAD
   MIN(ID_CALENDARIO_START)                                    AS FirstDate,
   MAX(ID_CALENDARIO_END)                                      AS LastDate
RESIDENT Calendario_Fact;


LET v_FirstDate = NUM(PEEK('FirstDate', 0, 'StartEndDates'));
LET v_LastDate  = NUM(PEEK('LastDate', 0, 'StartEndDates'));


Drop Tables [StartEndDates], [Calendario_Fact];


TempCal:
LOAD
   DATE($(v_FirstDate) + ROWNO() -1)                           AS  TEMP_DATE
AUTOGENERATE 1
   WHILE ($(v_FirstDate) + ROWNO() -1 < $(v_LastDate));


Calendario:
LOAD
   TEMP_DATE                                                   AS ID_CALENDARIO,
   YEAR(TEMP_DATE)                                             AS ANNO,
   'S' & CEIL(MONTH(TEMP_DATE) / 6)                            AS SEMESTRE,
   NUM(MONTH(TEMP_DATE), '00')                                 AS MESE,
   DATE(MONTHSTART(TEMP_DATE), 'MM-YYYY')                      AS [MESE-ANNO]
RESIDENT TempCal;


Drop Table [TempCal];
```

Figura 6.15: Master calendar

Questa nuova tabella andrà a costituire quella che nel modello illustrato al precedente sotto-paragrafo è denominata come “Calendario” andando a completare il quadro delle dimensioni esposte.

6.2.6 Riduzione dei livelli dimensionali

Parlando di pregi e difetti dei possibili schemi, si è detto fin da subito che molto dipende dal contesto e dai casi specifici ma che il modello a stella rappresenta il compromesso migliore in quanto garantisce il giusto equilibrio in termini di memoria ed efficienza. Infatti lo snowflake schema è sicuramente più parco nel consumo di memoria ma con prestazioni comunque peggiori in quanto più si deve saltare da una tabella all'altra maggiore è lo sforzo richiesto nell’eseguire un calcolo. Fin dal principio quindi il nostro obiettivo è stato quello di ricondurci ad uno star schema: in questo momento, però, siamo ben lontani dall’averlo. Non è sicuramente possibile arrivare ad uno schema a stella perfetto, in quanto abbiamo più tabelle dei fatti. Tuttavia il modello attuale, che è a tutti gli effetti uno snowflake, è sicuramente migliorabile. Sono infatti presenti delle dimensioni secondarie che fanno esplodere orizzontalmente il modello. L’intervento da fare è pertanto quello di riunire le gerarchie in tabelle dimensionali uniche,

denormalizzando. Le dimensioni che devono essere appiattite sono quelle relative ai clienti, alle agenzie e alle holding. L'operazione realizzata, basata sul left join, fonde così la "Clienti commerciali" con la "Portafoglio Venditori Clienti", la "Portafoglio Agenzie" con la "Portafoglio Venditori Agenzie" e la "Holding" con la "Target riferimento Holding" mantenendo sempre il nome della tabella di sinistra. In questo modo si arriva ad un'architettura con un solo livello dimensionale che, presentando però più tabelle dei fatti, può essere definita come schema a stella modificato. Sotto è possibile osservare la versione finale del modello.

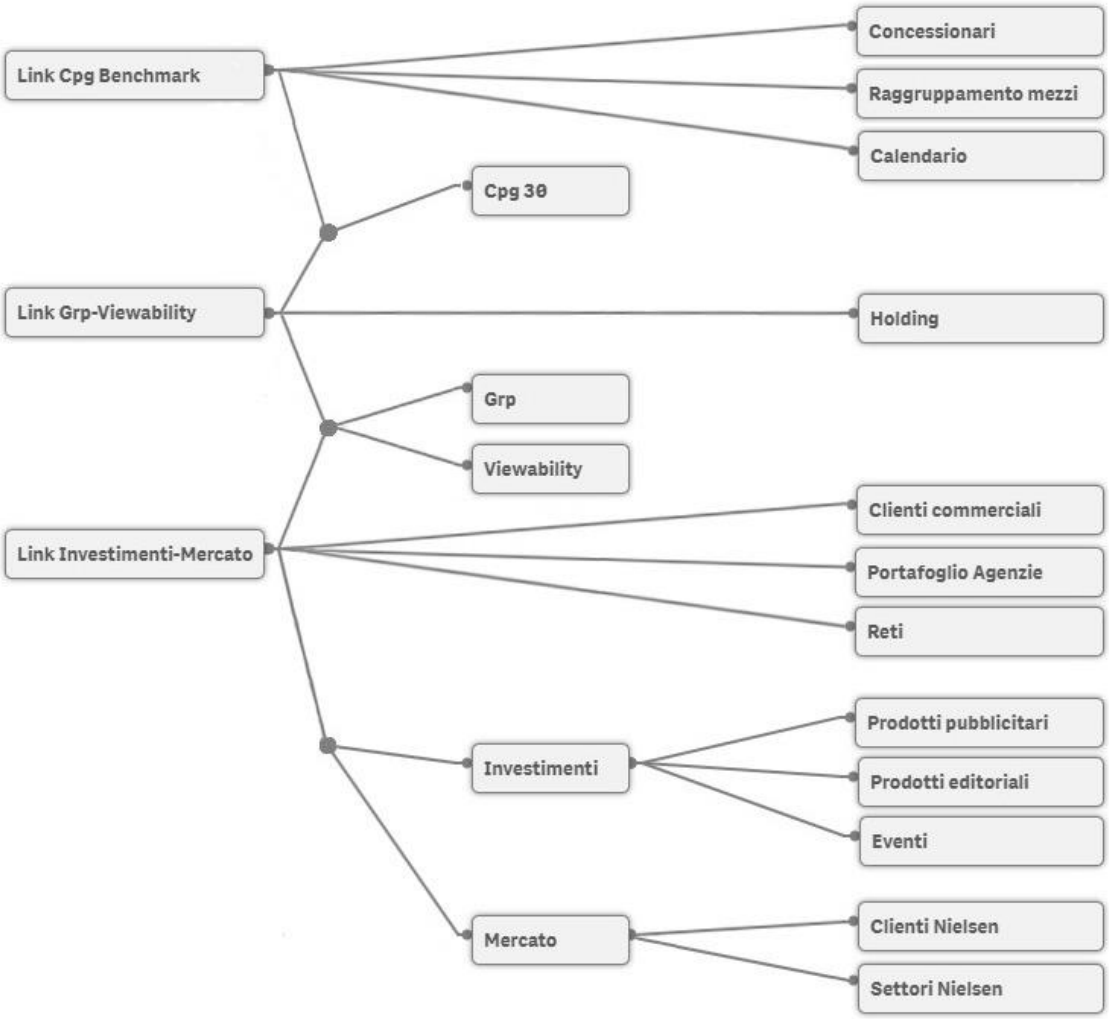

Figura 6.16: Modello - versione definitiva

6.2.7 Il data model dell'app "Flow"

Nel paragrafo 4.2 è stato spiegato del perché si è preferito scorporare l'analisi del flow da resto dell'app. Occorre quindi definire e costruire un data model separato. Volendo mostrare solo il flusso degli investimenti per prodotto e fare una comparazione con i

competitors, non è necessario portare in pancia il dato interno, per cui ci basta avere un'unica fact table contenente il dettaglio sugli investimenti fornito dalla Nielsen. È stata quindi predisposta una tabella denominata "Dati Nielsen" che mostra quanto di interesse. I dati sono estratti dalla VDT_DATI_NIELSEN che, a differenza della VDT_INV_MERCATO utilizzata nell'app "Scheda Accordi Quadro", scende a livello di marca. Così è possibile avere il dettaglio degli investimenti per linea di prodotto. Ovviamente, anche in questo caso, si limita il caricamento esclusivamente a quanto necessario e si aggrega esponendo le prospettive di osservazione che si vogliono esibire. Nello specifico si raggruppa per: codice marca Nielsen, codice holding, codice raggruppamento concessionari Nielsen, codice raggruppamento mezzi Nielsen, codice rete Nielsen, codice settore Nielsen, flag caricamento, anno, mese e settimana. Logicamente, sono poi esposte le misure: importo netto Nielsen, importo netto marketing Nielsen, importo netto AQ Nielsen. Quest'ultimo viene calcolato, come già realizzato e spiegato nel sotto-paragrafo 6.2.2, sommando il netto Nielsen se il flag di caricamento ha valore uno (quindi il periodo è consolidato), il netto marketing Nielsen altrimenti. A quest'unica tabella dei fatti sono collegate tutte le dimensioni. "Holding", "Concessionari", "Raggruppamento mezzi", "Reti" e "Settori Nielsen" sono costruite come illustrato in 6.2.4. La tabella "Calendario", costituente la dimensione del tempo, viene assemblata seguendo i passi dettagliati nel sotto-paragrafo 6.2.5, andando a prelevare primo ed ultimo giorno di interesse e generando tutte le date ivi comprese. Rispetto al caso precedente si aggiunge però il campo che puntualizza la settimana, dal momento che l'analisi del flow vuole spingersi fino a questa profondità temporale. A fronte della generazione di questa nuova tabella, sui fatti non si mostrano più anno, mese e settimana ma un campo chiave denominato ID_CALENDARIO costituito dalla concatenazione dei singoli attributi e che realizza l'associazione con la "Calendario". L'unica dimensione totalmente nuova è quella denominata "Marche Clienti Nielsen". Questa si fonda su una vista, la VDM_MARCHE_CLIENTI_NIELSEN, che stabilisce la relazione tra le marche e i corrispettivi clienti Nielsen che producono quei prodotti. I campi esposti da questa dimensione sono codice, descrizione e tipo della marca e codice e descrizione del cliente Nielsen. Considerato che il livello più basso è chiaramente rappresentato dalla marca, la connessione con la tabella dei fatti è per codice marca Nielsen. Avendo un'unica fact table circondata da tutte le dimensioni, come è possibile riscontrare in figura 6.17, il modello realizzato è un perfetto star schema.

Figura 6.17: Data model dell'app "Flow"

Questo conclude la trattazione del data model.

CAPITOLO 7

QUARTA FASE DI SVILUPPO APP: USO DEL FRAMEWORK DI ETL AZIENDALE

Nei precedenti capitoli è stato descritto il flusso evolutivo che ha condotto dalle richieste utente fino alla definizione del data model. Prima di spingersi nell'ambito della data visualization occorre predisporre la nuvola dati dell'app sfruttando il framework di ETL descritto nel terzo capitolo. In questo modo le operazioni necessarie a foggiare il modello associativo vengono realizzate in maniera uniforme seguendo i pattern prestabiliti e, contestualmente, si va anche a rimpolpare il data lake per il mondo Qlik. Il risultato di questo processo fornisce in output i QVD di progetto che, caricati tramite il load script dell'applicazione, fanno sì che tutti i dati di interesse correttamente strutturati siano a disposizione, pronti per essere usati nella realizzazione degli oggetti grafici da presentare. Ad ogni fase dell'ETL sarà dedicato un paragrafo a se stante.

7.1 Configurazione dell'ETL 01 centralizzato

L'ETL 01 centralizzato ha un solo principale grande obiettivo: prelevare i dati dalle fonti, al fine di scorporare l'elaborazione in Qlik dal livello sorgente, evitando così di sovraccaricare i sistemi esterni. Per cui le operazioni richieste sono solo quelle di settings delle connessioni alle sorgenti e la definizione delle tabelle da estrarre.

Prima di tutto occorre però definire le variabili globali nella ETL_GLOBAL_VARIABLES. Bisogna sicuramente inserire tutte quelle indispensabili al funzionamento dell'ETL 01 indicate al paragrafo 3.3. La variabile "vs_LoadingType" assumerà valore "FULL" dal momento che, per definizione, di volta in volta occorre estrarre la fonte completa. I paths settati saranno quelli indicati in 3.2 quando si è discusso della necessità di disporre di un'adeguata alberatura delle cartelle per smistare correttamente i risultati prodotti dai vari livelli. Tutte le altre variabili assumeranno i valori di default specificati nella trattazione dei paradigmi costituenti del framework.

Le connessioni da configurare sono di due tipi: relazionali o esterne. Quelle appartenenti alla prima tipologia avranno una connection string in forma "LIB CONNECT TO 'NomeConnessione'" in cui "NomeConnessione" sarà "DWH_RP_BI", "DB Commerciale" o "DB_ETL_Toolkit" a seconda che la connessione OLE DB punti allo schema QLIKUSR dell'istanza DWH, allo schema QLIKUSR dell'istanza DB commerciale o allo schema QLIKADM dell'istanza DWH contenente tutte le tabelle di settings. Ovviamente è imprescindibile aver creato queste connessioni su Qlik prima di

configurare la ETL_01_CONNECTIONS. Per quanto riguarda invece le connessioni di tipo esterno, non è necessario includere una stringa di connessione, per cui la differenziazione si fa solo per forma mentis in relazione al tipo file: “XLSX”, “XLS”, “CSV”, “TXT”.

La tabella ETL_01_TABLETOEXTRACT deve essere compilata con tutte le viste necessarie al progetto create e discusse in 5.2.3. Da notare che è necessario caricare anche le tabelle essenziali per la Section Access, cioè quelle definenti gli utenti e i cluster di visibilità, e la VMP_QLIK_CATALOGO_INFORMAZIONI utile per il renaming delle informazioni.

L’ETL 01 centralizzato deve estrapolare i dati dalle sorgenti per intero senza compiere nessuna selezione o trasformazione perché il suo fine è solo quello di passare dai formati eterogenei in input ai QVD in output. Per questo motivo la tabella ETL_01_QUERY in questo frangente deve essere lasciata vuota. Infatti, se una tabella è presente in ETL_01_TABLETOEXTRACT e non viene richiamata in ETL_01_QUERY, lo script esegue una “LOAD *” importando tutti i campi.

Allo stesso modo non ci sono variabili specifiche riferenti l’ETL 01 sorgente, per cui anche la ETL_01_VARIABLES in questa fase deve restare sgombra.

Naturalmente è fondamentale che tutte le righe inserite allo stato attuale nelle tabelle di configurazione presentino PROJECTID uguale ad 1.

7.2 Configurazione dell’ETL 02 centralizzato

L’ETL 02 centralizzato è un livello per definizione opzionale in quanto è utile solamente ad applicare trasformazioni condivisibili da più applicazioni. Questo è però proprio il nostro caso dal momento che si vuole arricchire un QVD originato da una vista del DWH con informazioni aggiuntive provenienti dal DB commerciale.

Occorre aggiungere le variabili globali proprie di questa fase. In particolare, si setta la “vs_Flag_App” a 0 in modo che non venga applicata nessuna rinomina dei campi, e si imposta il path di salvataggio dei QVD risultanti dal livello sorgente a *lib://Dw_Main/Common/*. Da qui gli estrattori di app potranno prelevare i QVD di interesse. “Dw_Main” è una folder connection che punta alla cartella del server in cui sono contenuti tutti i QVD prodotti dal framework.

Successivamente si compila la ETL02_INDEX andando ad aggiungere tutti i QVD presenti nell’area di staging dell’estrattore centralizzato. Il campo TABLE_TOLOAD deve contenere il valore ‘R’ in quanto bisogna leggere in raw mode, cioè tirando su tutti i campi senza specificare selezioni e trasformazioni in ETL02_TABLES che in questo momento sarà allora vuota.

L’elaborazione è compiuta nella sezione “Custom” dello script in quanto si tratta di un join, operazione non configurabile tramite le tabelle di settings del framework. Per una questione di ordine, l’intervento viene realizzato all’interno di una sub-routine denominata “JoinMagazzinoPosBreak” richiamata nella “Custom”. Ciò che vogliamo fare, come spiegato in 6.2.2, è arricchire la tabella degli investimenti con le informazioni

sulle posizioni pregiate. In primo luogo si concatenano i due QVD ottenuti al passo precedente, costruiti a partire dalle due viste del DB commerciale ed esplicitanti il dato sulle posizioni pregiate per la TV e per la radio. Poi, preso il QVD degli investimenti, lo si mette in left join con il risultato della "Concatenate". A valle di ciò avremo la tabella originale arricchita di una nuova colonna. Questa tabella può essere memorizzata in formato QVD nella directory *Common* mantenendo il nome originale, cioè DATI_INV_MAGAZZINO. In questo modo tutti i progetti che preleveranno questo QVD lo troveranno rimpinguato di un dato in più e potranno liberamente scegliere se sfruttarlo o meno.

7.3 Configurazione dell'ETL 01 app

Quando si definisce un ETL di app il primo step è la definizione di un nuovo PROJECTID. Per convenzione gli ID dei progetti in collaudo hanno un valore compreso tra 1001 e 9999. Dal momento che attualmente un'altra app è in fase di testing, il PROJECTID assegnato è il 1002.

L'ETL 01 di app è deputato a tirare su solo i dati convenienti al progetto in corso di sviluppo. Quindi, a differenza dell'ETL 01 sorgente, bisogna anche dettagliare i campi che si vogliono estrarre.

Il primo passaggio è sicuramente rappresentato dalla compilazione della ETL_GLOBAL_VARIABLES. Vale la pena soffermarsi su tutte le variabili specificanti i percorsi. In particolare i file esterni sono caricati su una cartella del NAS, per cui "vs_Path_ExternalData" punterà a tale folder. Sul server è stata definita una directory di progetto avente nome *1002-AccordoQuadroCliente* organizzata come da standard definito in 3.2. Quindi tutte le varie cartelle di store sono strutture a partire da tale folder.

In questa fase gli unici tipi di sorgente sono QVD o fonti estemporanee (nello specifico file Excel). Per tale ragione vengono definite solo due connessioni: la prima denominata "QVD" e la seconda "XLSX" entrambe con stringa di connessione vuota e che distinguono concettualmente la sorgente in base al formato del file da prelevare.

In ETL_01_TABLETOEXTRACT vanno inserite tutte le tabelle che, in quanto rilevanti per la nostra app, devono essere estratte. Queste sono rappresentate dai QVD presenti in *Common* ma anche dagli Excel discussi nel paragrafo 5.1. Questi ultimi vengono estrapolati solo in questa fase poiché sono utili limitatamente a questo progetto e quindi non ha senso portarli nel data lake comune. Allo stato attuale il template di ETL 01 è predisposto per tirare su come file esterni solo i QVD. È stato pertanto necessario apportare un intervento atto a consentire il caricamento tramite framework anche di file con altra estensione. Se il tipo di connessione è esterna si va a distinguere la clausola di from a seconda della specifica di importazione. In questo modo si riesce a gestire anche l'estrazione di file che non sono QVD. Nella pagina successiva lo screen riportante la modifica realizzata.

```
// A seconda del tipo di connessione imposto il codice da eseguire
if('$(vs_TipoConnessione)' = 'RELAZIONALE' or '$(vs_TipoConnessione)' = 'ODS')   then
    Set vs_SQLCode      =   'SQL SELECT';
    Set vs_SQLCodePL    =   'Load';
    Set vs_FromCode     =   $(vs_Fisico);

elseif('$(vs_TipoConnessione)' = 'ESTERNA') then
    Set vs_SQLCode      =   'Load';
    Set vs_SQLCodePL    =   'Load';


if('$(vs_SpecificaImportazione)' = 'qvd') then
    Set vs_FromCode     =   [$(v_Path_ExternalDataQvd)$(vs_SubPathQVD)$(vs_Fisico).$(vs_SpecificaImportazione)] ($(vs_SpecificaImportazione));
else
    Set vs_FromCode     =   [$(v_Path_ExternalData)$(vs_SubPathExtDB)$(vs_Fisico)] ($(vs_SpecificaImportazione));
endif

end if
```

Figura 7.1: Modifica allo script di ETL 01 app

Alcuni tra i file Excel necessitano di un lavoro di adeguamento e di essere caricati a blocchi in quanto contengono lo stesso dato da punti di vista differenti. È il caso dei dati riguardati il GRP, la viewability e il CPG di benchmark, come spiegato nel sotto-paragrafo 6.2.2. Per tale ragione questo genere di prelievo si concretizza nella sezione "Estrazioni Custom" del template andando a scrivere qui il codice di load e il successivo store in formato QVD nell'area di staging dell'app.

In ETL_01_QUERY si settano i campi delle tabelle riportate in ETL_01_TABLETOEXTRACT da tirare su. In questo modo si mette in pratica quanto suggerito da una tra le best practices: la limitazione del caricamento a quanto effettivamente necessario. In questa fase vengono anche compiute delle piccole modifiche in modo da garantire una certa uniformità. Nello specifico, sulla DATI_INV_MAGAZZINO e sulla DATI_INV_MERCATO contenenti rispettivamente il dato sugli investimenti e quello di mercato, in conformità con quanto già deciso nella realizzazione dell'anagrafica delle holding, il codice holding viene costruito esponendo se stesso nel caso non sia NULL o il codice cliente altrimenti. Si sostituisce poi al flag caricamento il tipo periodo Nielsen rispettando la regola illustrata in 5.2.3. Poi, considerando che tutte le fact tables a livello temporale espongono i dati per mese, nella DATI_INV_MAGAZZINO si sostituisce alla data trasmissione relativo mese ed anno.

Infine ETL_01_VARIABLES viene configurata con una serie di variabili utili a front-end.

7.4 Configurazione dell'ETL 02 app

L'ETL 02 di app è la fase cruciale del flusso in quanto qui si plasma il data model. È bene quindi agire con riguardo e compiere tutte le trasformazioni dovute sfruttando appieno il framework sia tramite le tabelle di settings che la sezione "Custom".

La tabella ETL_GLOBAL_VARIABLES deve essere arricchita con quanto rilevante per questo livello. Bisogna aggiungere le variabili di path restanti e settare "vs_Flag_App" ad 1 affinché possa avvenire la rinomina dei campi. Inoltre deve essere aggiunto l'hide prefix che, impostato a '%', consente di nascondere i campi non censiti o quelli che non si vuole rendere visibili come, ad esempio, i codici o le chiavi.

ETL02_INDEX è la tabella designata a definire le fonti, cioè quali QVD di staging sono

di interesse e i relativi output. Tutte le righe riguardanti le tabelle utili alla Section Access e al renaming presenteranno il campo TABLE_TOLOAD ad 'R' considerato che queste tabelle devono essere estratte per intero senza particolari elaborazioni. Tutti gli altri records avranno invece l'attributo TABLE_TOLOAD valorizzato ad 'Y' in modo da poter specificare in ETL02_TABLES quali sono i campi da estrarre, anche nel caso in cui l'intenzione sia di tirarli su tutti. In questo modo si ha assoluta chiarezza di ciò che viene prelevato e come viene trattato. Come illustrato in 6.2.3 le diverse tabelle dei fatti, che presentano più campi in comune, devono essere opportunamente collegate. La scelta compiuta è stata quella di farlo utilizzando delle link tables. Allora in questa fase occorre riportare che ogni fact table di staging produrrà in output due tabelle: quella propria e un'altra temporanea contenente i campi comuni, fonte costituente delle varie link tables finali.

L'ultima tabella da configurare è la ETL02_TABLES in cui vanno inseriti tutti i campi da estrapolare e come questi vanno gestiti. È qui che devono essere definite le trasformazioni volute sebbene non è possibile configurare sulle tabelle di settings le operazioni più complesse come le aggregazioni e i join per le quali si dovrà ricorrere alla sezione "Custom". Le funzioni di elaborazione possono essere inserite direttamente nel campo QVD_FIELD e si possono sfruttare tutte le script functions messe a disposizione da Qlik. Le fact tables risultanti preleveranno dai corrispondenti QVD di staging solo i campi propri e useranno i campi comuni esclusivamente per costruire la chiave composta di associazione con la rispettiva tabella di congiunzione. Come detto poc'anzi, da ogni tabella dei fatti si genererà anche una tabella temporanea contenente i campi comuni e le chiavi possibili. A questo punto è stato predisposto il terreno per compiere le ultime elaborazioni tramite script nella sezione "Custom" del template.

Quest'area, costituita da un'unica sub-routine, viene popolata da invocazioni a ulteriori sub-routine definite in sezioni proprie e che compongono il flusso elaborativo. La prima di queste è "Custom_GroupBy". Sulle fact tables, infatti, sono sì stati esclusi i campi comuni e costruite le chiavi composte ma non è ancora stata realizzata nessuna operazione di aggregazione che, se fatta a livello di dettaglio opportuno, consente di snellire le tabelle. È proprio questo il lavoro compiuto in questa sub-routine. Le successive funzioni definite mirano, invece, a ultimare le dimensioni. "Custom_SettorePrevalente" realizza il calcolo del settore dominante di investimento per ogni holding, come raccontato in 6.2.4. "Custom_Join", invece, concretizza la riduzione di un livello dimensionale come spiegato nel sotto-paragrafo 6.2.6. Contestualmente in questo step c'è anche un arricchimento verticale sulle dimensioni reso possibile dalle ApplyMap. In questo caso ci si appoggia alle tabelle di mapping che devono essere state definite nella sezione "Custom_Mappings", richiamata prima della "Custom". La penultima sub-routine invocata è la "Custom_Calendario" che dà origine alla dimensione temporale alla stessa maniera di quanto illustrato nel sotto-paragrafo 6.2.5. Infine, l'ultima funzione richiamata è la "Custom_LinkTables". Qui, partendo dalle tabelle temporanee generate a partire dai fatti, per ognuna delle tre link tables da procreare, si prelevano i campi adeguati e si concatenano opportunamente andando a costituire le link tables così come sono state descritte nel sotto-paragrafo 6.2.3.

Ovviamente, terminata ogni elaborazione occorre compiere due ulteriori passaggi: la rinomina dei campi usando la tabella di mapping "MapCatalogoInformazioni" creata precedentemente a partire dal QVD del catalogo informativo e lo store in QVD della

tabella considerata nella cartella *Dw_Associative* di progetto per mezzo della funzione "Fz_StoreTable". Il catalogo informativo viene caricato a partire dall'ETL 01 centralizzato ma adoperato solo in questa fase. In ogni caso serve compilarlo opportunamente fin dal principio associando ad ogni codifica tecnica l'appropriata descrizione utente che sarà visualizzata a front-end, settando ad 'S' il campo FLAG_ATTIVO (a meno che il campo non sia utile per la Section Access e non lo si vuole quindi rinominare) e impostando coerentemente il FLAG_VISIBILITA_UTENTE valorizzandolo ad 'N' nel caso si tratti di codici, chiavi, campi ordinamento, o ad 'S' altrimenti.

È bene notare che le quattro fasi dell'ETL sono comuni per "Scheda Accordi Quadro" e "Flow", cambierà solamente il front-end che preleverà dei QVD piuttosto che altri. La predisposizione del dato è però condivisa essendo unico il progetto.

Terminata questa fase, i dati sono pronti ad essere rappresentati.

CAPITOLO 8
QUINTA FASE DI SVILUPPO APP: IL FRONT-END

Fin qui è stata predisposta la nuvola dati dell'app. Senza una rappresentazione chiara e persuadente, però, il progetto si svaluterebbe poiché è proprio con gli oggetti grafici che gli utenti interagiscono. Avere una navigabilità delle informazioni agile e d'impatto è infatti il valore aggiunto della business intelligence e determina gran parte del successo del lavoro svolto. Questo capitolo si concentra su questo aspetto e, dopo un approfondimento sul concetto di data visualization, traccia un quadro sulla struttura visiva dell'app. Per motivi di privacy quanto mostrato è stato opportunamente mascherato. Nello specifico, con un intervento di transcodifica le ragioni sociali sono state mappate con descrizioni forfettarie e i valori numerici sono stati moltiplicati per un fattore casuale dato dal risultato di una formula che sfrutta la funzione rand().

8.1 La data visualization

Si dice che un'immagine valga più di mille parole. Questo è quanto di più vero quando si parla di dati. Infatti il nostro cervello ha la capacità di elaborare le immagini molto più velocemente del testo. Circa il 70% di tutti i nostri recettori sensoriali sono nei nostri occhi e così si riesce a discernere una scena in meno di un decimo di secondo e processare fino a 36000 messaggi visivi all'ora. Ovviamente queste tempistiche sono ben lontane da quelle tipicamente necessarie a comprendere le informazioni testuali. Infatti le immagini vengono elaborate 60000 volte più velocemente rispetto al testo e il 90% delle informazioni trasmesse al cervello è di tipo visivo. A ragion veduta, la raffigurazione grafica diviene la prima fonte veicolante di conoscenza.

La data visualization è proprio l'insieme di metodi che utilizzano rappresentazioni visive per dare un senso, esplorare e comunicare dati quantitativi. Il suo scopo prominente è quindi quello di trasmettere le informazioni in modo chiaro ed efficace attraverso mezzi grafici, migliorare la comprensione e quindi, in ultimo, consentire decisioni migliori favorendo azioni più coscienziose. È quindi un componente essenziale per garantire il successo di qualunque progetto di business intelligence. La data visualization dà vita alla BI. Se il data modeling può essere paragonato alle fondamenta di una casa, la data visualization è l'arredamento; una fase segue l'altra ma sono entrambe indispensabili per la riuscita dell'attività.

La potenza della data visualization è quella di fornire un approccio d'impatto per mostrare

connessioni e relazioni non scontate e invisibili nel formato testuale. Aiuta quindi l'utente a fare confronti, individuare pattern ricorrenti, notare anomalie, riconoscere aree che hanno bisogno di maggiore attenzione e su cui magari intervenire avendo un profondo impatto sui traguardi raggiunti da un'azienda. I dati diventano infatti più semplici da digerire e interpretare. Diviene così più naturale avere l'intuizione corretta e muoversi opportunamente e col giusto tempismo verso l'azione. Il valore aggiunto della rappresentazione grafica è quello di sfruttare degli oggetti grafici dinamici per poter ri-assemblare in maniera intuitiva una grande mole di dati che, se osservati in report, tabelle o fogli di lavoro tradizionali, sarebbero non navigabili. La dinamicità va a braccetto con l'interattività: così facendo l'analisi diventa più costruttiva e si è spinti a porre nuove domande stimolati dall'esplorazione.

Quando si architetta una rappresentazione grafica occorre trovare il giusto compromesso tra estetica e funzionalità. Quindi la visualizzazione dei dati non deve risultare noiosa per essere efficace o estremamente sofistica per apparire attraente. La finalità della data visualization è comunicare i dati in modo rapido e significativo garantendo al tempo stesso la massima accuratezza. Una visualizzazione deve servire per uno scopo chiaro e non tempestare gli utenti di dettagli inutili. L'idea è quindi quella di creare raffigurazioni di dati armoniose e allo stesso tempo funzionali al fine di fornire metodi intuitivi per la percezione di dati complessi. Per pianificare una visualizzazione dati è necessario innanzitutto comprendere i dati stessi, capire come sono organizzati e collegati in modo da scegliere con cognizione di causa l'oggetto grafico che offre una presentazione dei valori più adeguata. È opportuno quindi sfruttare modelli di progettazione ben noti e tenere conto di alcune regole pratiche: la coerenza tra le informazioni esposte è basilare in modo da consentire uniformità e consequenzialità nella navigazione e nella comprensione; anche l'ordine visivo è essenziale: un numero eccessivo di informazioni compromette la chiarezza; non bisogna poi esagerare con i colori: una tinta non adatta posta in un punto sbagliato potrebbe comportare confusione anziché chiarezza.

Qlik Sense offre un vasto numero di visualizzazioni per presentare i dati caricati nelle apps, semplici da aggiungere e altamente personalizzabili. Nello specifico, mette a disposizione i seguenti oggetti: grafico a barre, box plot, grafico combinato, grafico di distribuzione, casella di filtro, misuratore, istogramma, KPI, grafico lineare, mappa, grafico a torta, tabella pivot, grafico a dispersione, tabella, testo e immagine, mappa ad albero, grafico a cascata.

	Grafico a barre	Il grafico a barre visualizza una barra per ogni valore di dimensione. La lunghezza della barra corrisponde al relativo valore numerico della misura.
	Box plot	Il box plot è adatto per il confronto degli intervalli e delle distribuzioni di gruppi di dati numerici ed è rappresentato da una scatola con baffi e una linea centrale.
	Grafico combinato	Il grafico combinato riunisce le barre e le linee nello stesso grafico. Le barre e le linee presentano assi diversi che consentono di confrontare le percentuali e le somme.
	Grafico di distribuzione	Il grafico di distribuzione è adatto per il confronto degli intervalli e delle distribuzioni di gruppi di dati numerici. I dati sono tracciati come punti dei valori lungo un asse.
	Casella di filtro	La casella di filtro consente di controllare i dati mostrati nelle visualizzazioni su un foglio. Una casella di filtro consente di filtrare contemporaneamente i dati di più dimensioni.
	Misuratore	Il misuratore consente di visualizzare il valore di una singola misura, senza dimensioni.
	Istogramma	L'istogramma è adatto per visualizzare la distribuzione di dati numerici su un intervallo continuo o per un certo periodo di tempo. I dati sono divisi in contenitori.
#1	KPI	Il KPI viene utilizzato per presentare dati quantitativi fondamentali relativi alle prestazioni. È possibile aggiungere un collegamento a un foglio.
	Grafico lineare	Il grafico lineare visualizza le linee dei dati tra i valori. I grafici lineari vengono spesso utilizzati per visualizzare una tendenza nei dati su un intervallo di tempo.
	Mappa	La mappa viene utilizzata per combinare dati geospaziali e valori di misura, come le vendite relative a una regione o a un negozio.
	Grafico a torta	Il grafico a torta visualizza la relazione tra una singola dimensione e una singola misura.
	Tabella pivot	La tabella pivot mostra dimensioni e misure come righe e colonne di una tabella. Consente di analizzare i dati contemporaneamente in più dimensioni. I dati in una tabella pivot possono essere raggruppati in base a una combinazione di dimensioni e possono essere visualizzati anche somme parziali.
	Grafico a dispersione	Il grafico a dispersione presenta i valori da due misure. Si rileva utile per rappresentare i dati nei casi in cui a ciascuna istanza corrispondano due numeri, ad esempio una nazione (popolazione e relativo tasso di crescita). È possibile utilizzare una terza misura opzionale che viene riflessa nella dimensione delle bolle. Quando si visualizzano serie di dati di grandi dimensioni, per rappresentare la dimensione della misura verranno utilizzati i colori anziché la dimensione della bolla.
	Tabella	La tabella visualizza i valori sotto forma di record, in modo che ogni riga della tabella contenga i campi calcolati utilizzando le misure. In generale, una tabella contiene una dimensione e diverse misure.
A	Testo e immagine	È possibile utilizzare la visualizzazione testo e immagine per aggiungere testo, immagini, misure e collegamenti a una pagina Web.
	Mappa ad albero	La mappa ad albero mostra i dati gerarchici. Una mappa ad albero può mostrare un gran numero di valori simultaneamente all'interno di uno spazio limitato.
	Grafico a cascata	Il grafico a cascata illustra come un valore iniziale viene influenzato da valori intermedi positivi e negativi.

Figura 8.1: Visualizzazioni Qlik Sense

L'interattività è integrata in quanto Qlik Sense evidenzia automaticamente le voci associate alle selezioni eseguite e queste vengono riprodotte in tutte le visualizzazioni affiliate nell'intera applicazione. Se gli oggetti predisposti da Qlik non bastassero, o nel caso fossero ritenuti inadeguati, c'è sempre la possibilità di utilizzare visualization extensions o widgets creati sfruttando gli strumenti di Qlik Sense Dev Hub o scaricati dal Qlik Branch.

8.2 Struttura dell'app

Un'applicazione Qlik Sense può essere costituita da uno o più fogli. In questi vengono posizionati i vari filtri, grafici, tabelle, mappe utili a rappresentare i dati. Disporre di diversi fogli offre l'opportunità di strutturare quanto esposto in sezioni di approfondimento specifico. In questo modo si riesce sia a trasmettere ordine che a migliorare la comprensione di ciò che viene visualizzato. Proprio per questo, il consiglio suggerito dalle best practices Qlik al momento della progettazione della data visualization, è quello di creare innanzitutto una struttura di fogli vuoti, in cui ognuno di questi rappresenta un'idea o un obiettivo. Ciò fornisce la possibilità di ragionare sui contenuti e sulle loro rappresentazione per macro-argomenti. Per quanto riguarda l'app "Scheda Accordi Quadro", il primo contatto deve sicuramente essere quello con la selezione della holding/cliente su cui si vuole procedere nell'analisi. Compiuta questa scelta, occorre poi avere un quadro di riferimento del cliente scelto, quindi una scheda informativa che sintetizzi le informazioni di rilievo. Come riportato tra le richieste, si vogliono poi visualizzare i dati del mondo interno per cui è certamente necessario esporre un foglio sugli investimenti in Rai Pubblicità. Similmente, bisogna fornire uno sheet che mostri le quote di mercato e consenta di effettuare i confronti con la concorrenza per permettere un'analisi a tutto tondo. Infine, sono stati richiesti dei dati specifici riguardanti in particolare GRP, CPG, viewability e CPM. Dal momento che i primi due elementi riguardano il mondo TV e radio, si è optato per creare un foglio che potesse consentire un focus su questo universo aggiungendo qui tutto quanto è tipico di questi due mezzi. Viewability e CPM sono invece dati concernenti il mondo web. Per cui si è scelto di operare allo stesso modo offrendo un foglio che permetta di zoomare su questo ambito. Nel totale, sono quindi stati ideati i seguenti sei fogli: "Selezione cliente", "Scheda informativa", "Investimenti Rai Pubblicità", "Analisi mercato", "Focus TV/Radio", "Focus web". Per quanto riguarda il flow, invece, un solo foglio basta a mostrare l'andamento temporale degli investimenti per linea di prodotto.

Prima di scendere nelle descrizioni sommarie dei fogli congeniati e realizzati, occorre però fare una digressione su due concetti ampiamenti sfruttati: master library e set analysis. La prima è sostanzialmente un pannello utile a contenere le cosiddette voci principali, cioè risorse riutilizzabili quali visualizzazioni, dimensioni e misure. Proprio per questo, nel corso del concepimento dei vari fogli, sono state via via create master dimensions e master measures in modo da definire una volta sola dimensioni e misure e poterle poi ri-impiegare su oggetti differenti. Il beneficio apportato da questa centralizzazione è quello che gli aggiornamenti eseguiti per una voce principale saranno applicati a tutti gli elementi in cui questa si utilizza. La set analysis è, invece, una tecnica innovativa introdotta da Qlik che consente di determinare il livello di sensibilità alle selezioni. In generale, quando si effettua una selezione, le funzioni di aggregazione come Sum, Max, Min, Avg e Count eseguono l'aggregazione sul set di dati determinato dalle selezioni attuali. Le selezioni definiscono quindi automaticamente la serie di dati da aggregare. La set analysis consente, al contrario, di definire un gruppo indipendente da quanto si è selezionato. Ciò si rivela particolarmente utile soprattutto quando si desidera mostrare un valore particolare, indipendentemente dalle selezioni attuali, o anche quando si eseguono tipi di confronto diversi. Questa tecnica è stata utilizzata in larga misura per la generazione di tante e variegate master measures.

I fogli prodotti cercano di seguire quanto più possibile le richieste utente ma, in concomitanza, sono state sfruttate anche le proprie conoscenze e le proprie intuizioni facendo valere le proprie idee.

Per quanto riguarda l'app "Scheda Accordi Quadro", sono quindi stati realizzati sei fogli. Il primo, denominato "Selezione cliente", è quello preposto alla scelta della holding/cliente da approfondire. Nello specifico, sono stati predisposti una serie di oggetti per far sì che la scelta risulti semplificata: una barra con cinque caselle di filtro che permettono o di scegliere direttamente la holding di interesse o di limitare il set possibile alle solo holding con cui dovrebbe essere stipulato un accordo quadro o di determinare il periodo da osservare (scegliendo anno e mese) o il mezzo; un ulteriore filtro espanso utile a restringere la preferenza a un determinato settore prevalente di investimento; un bar chart mostrante le holding ordinate per fatturato totale (si dettaglia qui che per fatturato totale si intende l'importo netto lordizzato, come da specifiche utente) che consente di scorgere rapidamente i clienti che hanno investito di più e quindi, di conseguenza, quelli più interessanti. In basso, ricorrendo a una visualization extension, sono stati inseriti sei pulsanti che permettono di saltare immediatamente a un foglio specifico, nel caso si voglia scendere verticalmente su un ambito di analisi particolare, ma anche di collegarsi con l'app del flow. Infatti, come illustrato nel paragrafo 4.2, si è concordato sullo scindere l'analisi del flow dal resto dell'app ma con la promessa di fornire un ponte di contatto. Per innalzarlo si sfruttano le App Integration API messe a disposizione da Qlik che consentono di passare parametri a un'app e di aprirla quindi con una o più selezioni applicate. Ciò è realizzato tramite opportune concatenazioni al base URL identificante l'app stessa. In questo modo la navigazione utente è coerente e non risente della suddivisione.

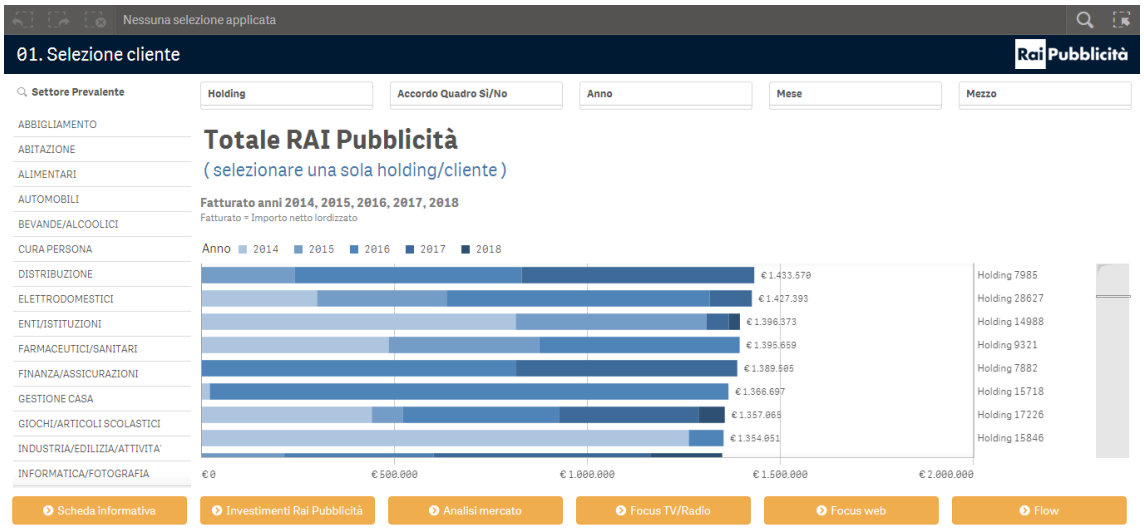

Figura 8.2: Foglio "Selezione cliente"

Il foglio successivo, "Scheda informativa", mira a dare una collocazione all'interno del mercato alla holding prescelta esibendo le informazioni che più la caratterizzano. Da qui in poi, per chiarezza, viene sempre specificata in una nota la fonte dati che, in questa

circostanza, è la Nielsen. Viene presentata la rilevanza della holding opzionata usando due ranking: uno generale che mostra quindi il suo posizionamento complessivo e uno riguardante invece il piazzamento circoscritto al gruppo dei clienti aventi lo stesso settore prevalente. Attraverso dei grafici, a barre, ad anello o a mappa, si espone poi la distribuzione degli investimenti per anno, mezzo, settore e cliente. Si fornisce quindi, tramite tabelle, un'informativa sulle agenzie che seguono la holding scelta e sui venditori clienti e agenzie che hanno l'incarico alla vendita per quel dato cliente. Infine, ricorrendo a dei KPI, si rivela il dato più considerevole: la quota Rai, cioè la percentuale di investimenti in Rai rispetto al totale, e i relativi scostamenti di questa rispetto alla quota Rai media per chi investe nello stesso settore prevalente della holding selezionata e rispetto alla quota Rai media di tutti gli investitori.

Figura 8.3: Foglio "Scheda informativa"

I fogli seguenti sono tutti tematici e consentono di addentrarsi in una sfera di analisi ben precisa. La scelta compiuta è stata quella di esporre sempre una sequenza di filtri utili a circoscrivere l'osservazione a quanto desiderato e un testo riportante la holding prescelta, il suo settore di investimento prevalente e i clienti appartenenti a quella data holding o solo quelli selezionati, nel caso si voglia limitare l'analisi ulteriormente, in modo che sia ben chiaro il riferimento ai dati esposti.

Il terzo foglio, chiamato "Investimenti Rai Pubblicità", consente di calarsi direttamente nel mondo interno, come dettaglia la postilla a fondo pagina. Tramite un grosso line chart si mostra il fatturato totale per anno dando la possibilità di scendere anche a livello mensile. Sotto, in maniera testuale, è riportato il fatturato totale dell'anno scelto o di quello corrente se nessuna selezione temporale è stata compiuta, il suo corrispettivo nell'anno immediatamente precedente e la variazione percentuale. Nel caso sia presentata l'informazione per l'anno corrente, la variazione è calcolata all'ultima data di caricamento in modo che il confronto risulti equo. Come da richiesta, questo trittico di informazioni è esposto anche per il fatturato al netto degli eventi. Sfruttando dei KPI, che forniscono un impatto sicuramente maggiore, si evidenziano alcuni dati di riguardo:

l'incidenza sul fatturato Rai, cioè quanto influisce l'investimento della holding prescelta rispetto ai guadagni Rai, il ranking generale e quello limitato al solo settore prevalente (calcolati però ovviamente a partire dal dato interno) e l'incidenza delle iniziative speciali della holding scelta, cioè quanti degli investimenti sono dedicati a prodotti pubblicitari facenti parte di questa categoria, rapportata alla media dello stesso valore calcolato tra i primi cinquanta clienti. Chiarezza maggiore su quest'ultima informazione è data da un grafico ad anello che, posto appena accanto all'ultimo KPI descritto, mostra la distribuzione del fatturato per categoria di prodotto. Infine, in forma tabellare, si espongono, suddivisi per mezzo: il fatturato totale, la percentuale sconto cliente, la percentuale sconto totale, le sanatorie, gli omaggi contrattuali, l'importo lordo.

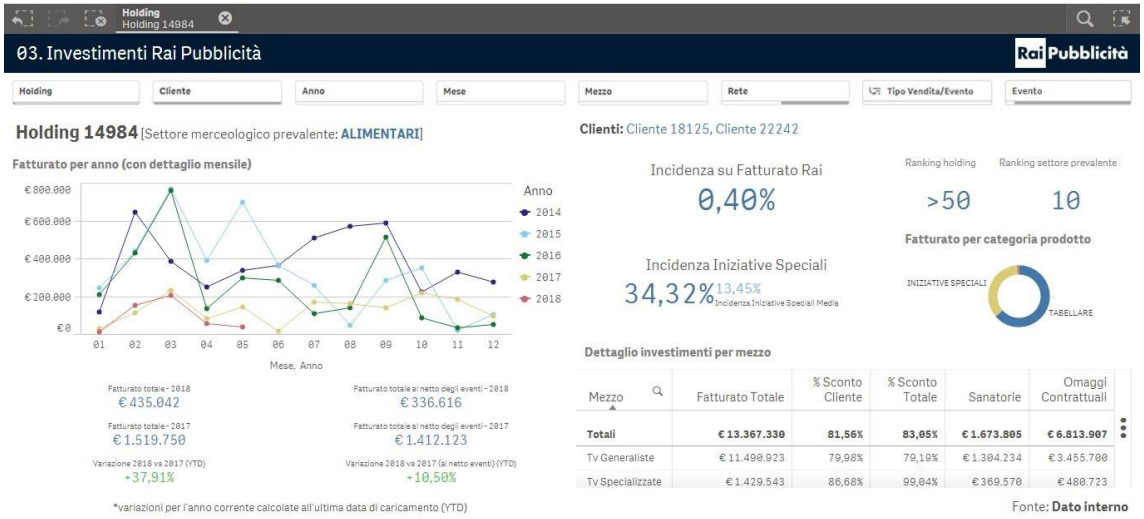

Figura 8.4: Foglio "Investimenti Rai Pubblicità"

"Analisi mercato" è il foglio che deve consentire il confronto con i competitors. Ovviamente qui la fonte è Nielsen. Tramite una tabella pivot vengono mostrati gli investimenti per concessionario ripartibili per anno. Accanto, mediante un grafico ad anello, è esposto lo stesso dato ma in forma grafica così da fornire un elemento maggiormente tangibile. La fascia in basso viene occupata da tre bar charts mostranti la distribuzione degli investimenti rispettivamente per cliente, mezzo e settore. Infine, sulla destra, si forma un quadrato con nove KPI esponenti le quote desiderate. Si creano sostanzialmente tre strisce dedicate ognuna a un dominio specifico. La prima è rivolta alla quota Rai calcolata sostituendo a quanto fornito dalla Nielsen riguardante il nostro concessionario il dato interno. La seconda è indirizzata alla quota Rai misurata solo col dato Nielsen puro. La terza è orientata alla quota Mediaset. Per ogni fascia è riportato il valore della quota cliente, quello medio del settore prevalente e quello medio su tutti i clienti. Inoltre il dato esposto si riferisce all'anno vagliato o all'anno corrente se non è stata compiuta nessuna selezione e viene mostrato, per ogni quota, anche il margine positivo o negativo con l'anno appena precedente. Nel caso si tratti dell'anno corrente, per uniformità, il confronto è eseguito solo sul periodo consolidato, lasso di tempo per cui la Nielsen garantisce che i dati siano confermati.

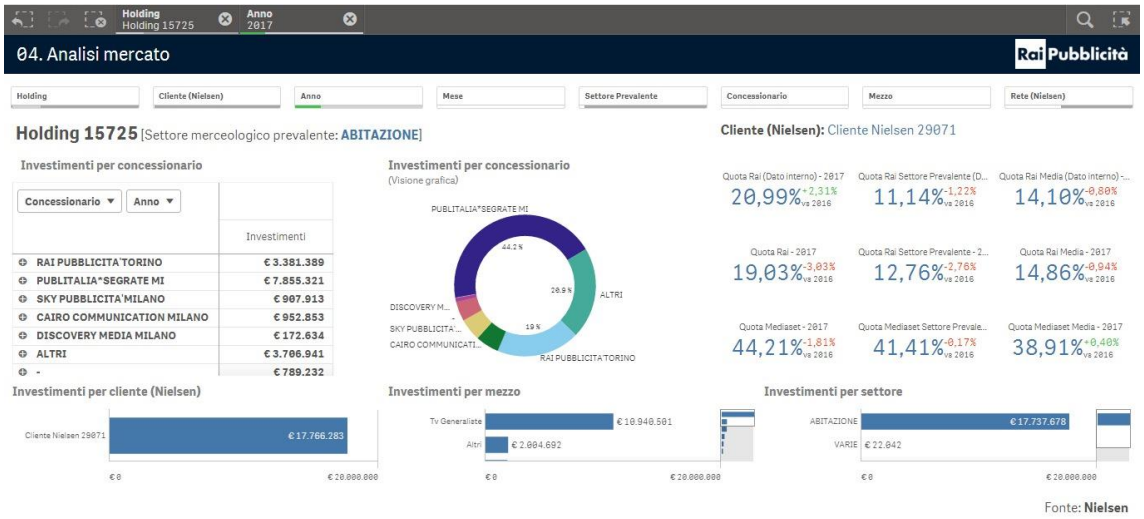

Figura 8.5: Foglio “Analisi mercato”

Il quinto foglio, “Focus TV/Radio”, permette di inquadrare i dati peculiari di un ambito ben preciso. Qui la fonte è il dato interno. Sulla sinistra un filtro consente di selezionare il target di interesse dal momento che i valori di GRP e CPG saranno visualizzati solo dopo aver scelto un singolo target, perché privi di senso altrimenti. Contrariamente agli altri fogli, per ogni holding non è mostrato il settore prevalente di investimento ma il target di riferimento, più utile in questo contesto, mentre sulla destra vengono puntualizzati i due target standard definiti. I dati su GRP e CPG sono esibiti sia in forma grafica, tramite bar chart il primo e line chart il secondo, che in forma tabellare. Selezionato uno dei due target standard, vengono inoltre visualizzati i corrispettivi valori di CPG di benchmark. La parte destra del foglio è destinata a contenere le informazioni riguardo il trasmesso. Nello specifico, c’è una tabellina che mostra, divisi per mezzo, il numero di secondi e la percentuale di spot trasmessi in posizione pregiata (prima o ultima di un break). Poi, tramite un KPI, si espone l’incidenza su secondi Rai, cioè la percentuale di secondi di messa in onda rispetto al totale Rai. Infine, avvalendosi di un bar chart, viene mostrata la redditività sia netta che lorda (cioè il rapporto tra l’importo netto lordizzato o l’importo lordo e i secondi) per anno, consentendo anche di scendere a livello mensile.

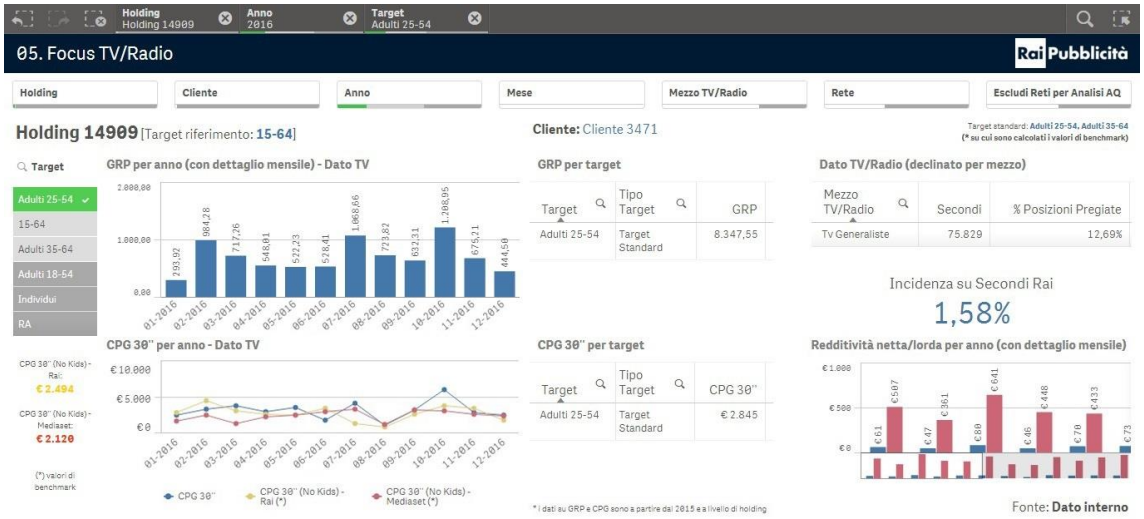

Figura 8.6: Foglio "Focus TV/Radio"

Per ultimo troviamo il foglio "Focus web". Sono esternati: mediante un bar chart, il numero di impressions vendute ed erogate per anno; usando dei KPI, il netto ricavato da questa forma di comunicazione nell'anno prescelto o nell'anno corrente se non sono state compiute selezioni rapportato all'anno precedente e la variazione tra quanto erogato e quanto venduto; per mezzo di due line charts, il CPM cliente confrontato con quello medio Rai riferito sia allo stimato che all'effettivamente erogato. La parte bassa del foglio è destinata a contenere i dati sulla viewability. Questi, come detto nel paragrafo 5.1, provengono dal mondo ComScore, per cui, a sinistra, sono riportati ragione sociale e settore merceologico secondo l'anagrafica della società che ha fornito i dati. Più a destra vengono posizionati i valori di viewability del cliente e quelli del settore cui il cliente appartiene e il loro scostamento. Questi dati sono disponibili per il solo primo semestre del 2017, per cui saranno visibili solo selezionando questo specifico intervallo.

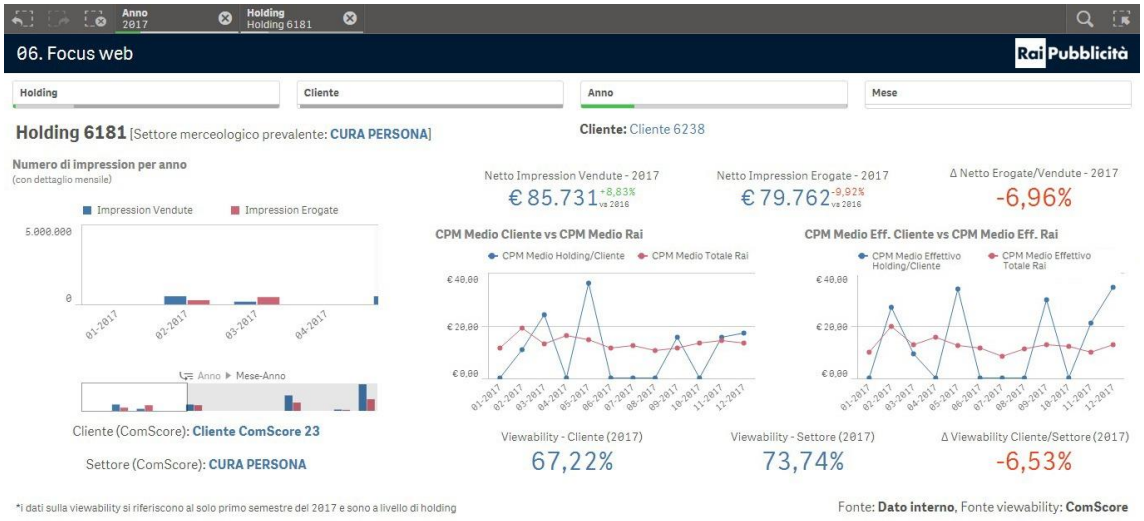

Figura 8.7: Foglio "Focus web"

Per quanto riguarda l'app "Flow" basta un solo foglio che, come desiderato, mostri l'andamento degli investimenti per linea di prodotto declinato per concessionario. Ciò viene realizzato ricorrendo a un line chart che espone tante linee quanti sono i concessionari e ognuna di queste individua l'evoluzione degli investimenti nel tempo. Come richiesto è possibile scendere fino al dettaglio settimanale. Più in basso, una tabellina puntualizza quanto mostrato: per ogni marca si visualizza l'investimento declinato per mese-anno, settimana, concessionario, mezzo e rete. Per favorire la selezione, i filtri su cliente e marca sono in forma estesa cosicché risulti più evidente l'ambito di analisi.

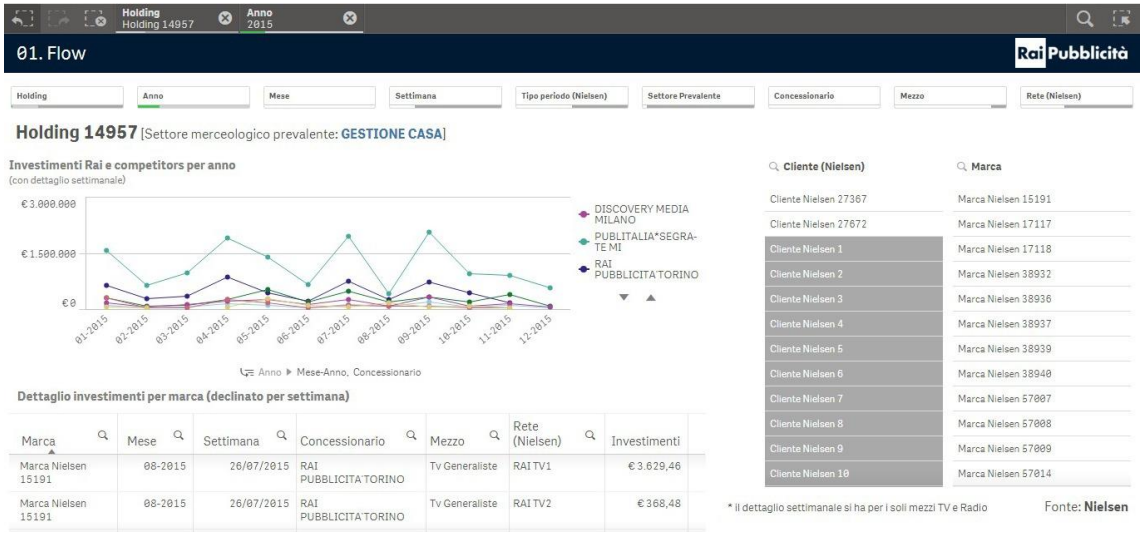

Figura 8.8: Foglio "Flow"

CAPITOLO 9

SESTA FASE DI SVILUPPO APP: INTEGRAZIONE CON R

Nelle prossime pagine è trattato il penultimo stadio di progetto. A dare origine a questo punto c'è un'ulteriore richiesta utente: l'analisi dei trend. Per riuscire a soddisfare questa esigenza si è ricorso ad R, strumento di analisi statistica che tra le tante funzioni che mette a disposizione ne include alcune per lo studio delle serie temporali. Dal momento che però si mira a fornire una user experience coerente e compatta, si è sfruttata la capacità di Qlik di integrazione con strumenti esterni in modo da consentire agli utenti di interagire con un solo software senza accorgersi della complessità che ci sta dietro. Nei paragrafi che seguono si esporrà allora l'ampliamento delle funzionalità base di Qlik e i passaggi necessari a installare e predisporre i nuovi componenti. Verrà poi fatta una digressione sulle serie storiche e, infine, si applicheranno le funzionalità appena integrate nel contesto dell'app sviluppata con l'aggiunta di un nuovo foglio che realizza quanto richiesto sfruttando i sistemi esplicitati.

9.1 Estendere Qlik Sense con funzionalità di motori di calcolo esterni

Qlik Sense è uno strumento aperto a rapportarsi con altri software di analisi. Offre infatti delle API che consentono l'integrazione e il passaggio di dati con strumenti esterni permettendo un arricchimento delle funzionalità base sfruttando la potenza dei motori di calcolo di terze parti. In questo modo si realizza quella che viene definita come AAI (Advanced Analytics Integration), concetto sempre più crescente nel mondo Qlik essenzialmente volto a consentire l'impiego di altri validi strumenti di analisi. Per concretizzare questa possibilità si sfruttano le SSE (Server Side Extensions) che sono sostanzialmente dei plugin che permettono lo scambio di dati da Qlik verso un linguaggio di scripting/programmazione esterno e viceversa avvalendosi delle remote procedure calls (RPC). Nello specifico, si utilizza gRPC: un sistema di RPC open source che si serve di HTTP/2 a livello applicativo e dei Protocol Buffers come meccanismo language-neutral e platform-neutral per la serializzazione di dati strutturati e che, inoltre, fornisce autenticazione, streaming bi-direzionale e controllo del flusso. Essenzialmente ciascun plugin SSE è un micro-servizio di comunicazione tra Qlik e dei third party engines. Ciò significa che un utente può selezionare dei dati sul front-end Qlik e questi vengono passati al di fuori di Qlik, valutati in un linguaggio esterno, elaborati e i risultati restituiti indietro all'applicazione Qlik. La potenza di ciò sta nell'opportunità di compiere un'analisi più

avanzata e approfondita rispetto a quanto potrebbe essere ottenuto utilizzando solo le funzionalità native di Qlik.

Impiegando delle SSE, cambia l'architettura base di Qlik. Nella sua versione standard, ogni grafico/oggetto nel front-end è reso da uno specifico ipercubo di dati che non è altro che la raccolta di dimensioni e misure rilevanti richieste da tale oggetto. Ogni volta che viene effettuata una selezione, il client Qlik richiede la versione aggiornata del cubo multidimensionale. Il motore QIX è responsabile del calcolo del nuovo ipercubo e dell'invio dei dati al front-end per il rendering. Questo modello è presentato schematicamente di seguito.

Figura 9.1: Architettura Qlik base[9]

Quando si utilizzano invece delle server side extensions, l'architettura si complica. Se un oggetto contiene una funzione di terze parti al suo interno, nel momento in cui viene richiesto l'ipercubo, il motore QIX calcolerà solo ciò che può utilizzando le funzioni native di Qlik. Se viene riconosciuta una funzione SSE personalizzata, il motore di Qlik passerà i dati associati al plugin SSE e richiederà la valutazione di tale funzione esterna (tramite una chiamata di procedura remota). Il plugin, dopo aver ricevuto il risultato dei calcoli compiuti dal software di terzi parti, restituirà il risultato a Qlik. Il motore QIX combinerà quanto ottenuto con ciò che è stato valutato in autonomia e riporterà l'ipercubo risultante al front-end per il rendering. L'architettura modificata è mostrata in figura 9.2.

[9] *Axis Group*, < http://www.axisgroup.com/qlik-sense-server-side-extensions-part-13-architecture-environment/ >, ultima consultazione: 20/03/2018

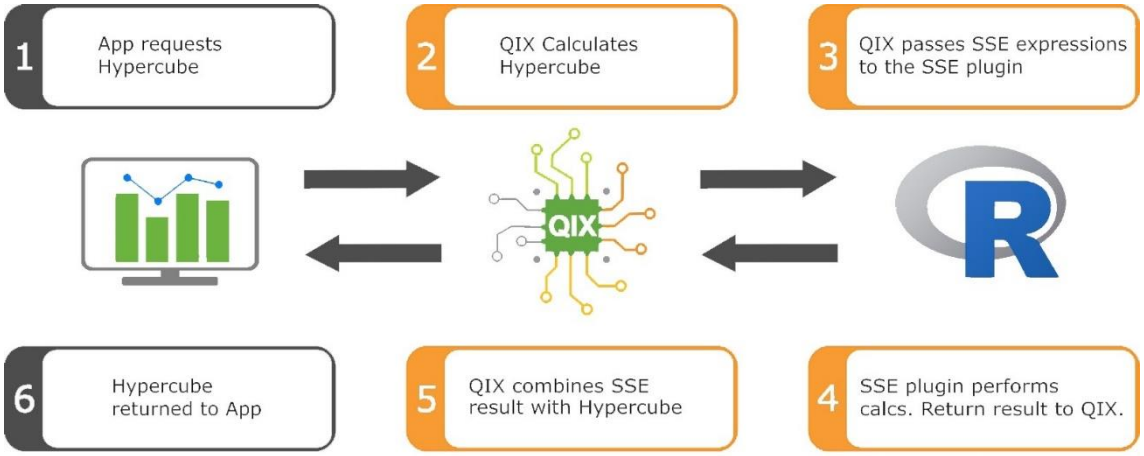

Figura 9.2: Architettura Qlik con SSE[10]

È quindi possibile scrivere funzioni personalizzate al di fuori di Qlik e poi invocarle al suo interno, richiamandole come fossero funzioni Qlik native. In questo modo l'esperienza utente rimane sostanzialmente invariata poiché lo scambio diretto di dati tra il motore QIX e i motori di calcolo di terze parti avviene in background e il risultato dei calcoli avanzati realizzati da strumenti esterni può essere visualizzato direttamente all'interno di Qlik Sense coerentemente all'esplorazione dati utente.

9.2 Installazione e configurazione

I componenti da installare e configurare sono essenzialmente due: R e il plugin SSE.

Per quanto riguarda R, essendo free, l'installer è liberamente scaricabile dal sito ufficiale. È importante collocarlo non in una user directory, in modo da consentire l'utilizzo da parte di tutti gli utenti. Realizzato questo passaggio occorre installare il package "Rserve" che sostanzialmente abilita le comunicazioni da e verso R. Rserve non è solo un pacchetto, ma un'applicazione. È fornito come libreria di R solo per comodità. Funge da socket e consente l'invio di richieste binarie ad R. In concomitanza ad Rserve sono stati installati anche altri packages ritenuti trasversalmente utili per possibili sviluppi futuri: "dplyr" che consente di lavorare in maniera agevole con i data frames; "forecast", che fornisce metodi e strumenti per l'analisi previsionistica; "rpart", contenente funzioni per il partizionamento ricorsivo utili per classificazione, regressione e alberi di decisione. Per includere una libreria precedentemente installata occorre avvalersi del comando "library(package)" che carica il namespace correlato e lo collega alla search list.

Relativamente alle server side extensions, Qlik Sense ha rilasciato un plugin open source denominato "SSEtoRserve" che realizza proprio la comunicazione tra il motore QIX e Rserve. Questo plugin è altamente configurabile. Tramite il file

[10] *Axis Group*, < http://www.axisgroup.com/qlik-sense-server-side-extensions-part-13-architecture-environment/ >, ultima consultazione: 20/03/2018

“SSEtoRserve.exe.config” possono essere infatti modificate le seguenti proprietà:

- grpcPort: è la porta che questo plugin dovrebbe aprire, cioè quella a cui il motore Qlik dovrà connettersi. Il valore di default è 50051;
- grpcHost: è l’host da cui questo plugin dovrebbe accettare richieste. Di default è valorizzato a localhost. Se Qlik è installato su un’altra macchina bisogna settare questo parametro a 0.0.0.0 in modo che SSEtoRserve sia raggiungibile da qualunque altro sistema;
- rservePort: è la porta Rserve a cui questo plugin dovrebbe connettersi. Di default assume il valore 6311;
- rserveHost: è l’host Rserve a cui questo plugin dovrebbe connettersi. Il valore pre-impostato è 127.0.0.1 (cioè localhost);
- rserveUser: nel caso in cui Rserve richieda un’autenticazione, qui occorre specificare l’username. Di default questo attributo viene lasciato vuoto;
- rservePassword: come sopra, ma per la password;
- certificateFolderFullPath: se si vuole abilitare la mutua autenticazione tra questo plugin e Qlik, qui è possibile definire il percorso alla cartella dove saranno contenuti i certificati. Nello specifico, questa directory dovrà includere tre certificati aventi i seguenti nomi: “root_cert.pem”, “sse_server_cert.pem”, “sse_server_key.pem”. Di default questo campo è vuoto, ciò significa che viene aperta una connessione insicura;
- rserveInitScript: consente di definire uno script R pre-impostato che sarà inviato in automatico non appena viene instaurata la connessione tra il plugin e Rserve. Viene utilizzato solitamente per includere librerie una tantum ed eliminare così la ridondanza del richiamare la stessa libreria per ogni espressione all’interno dell’app Qlik. Anche questo campo di base viene lasciato vuoto, quindi non è eseguita nessuna azione preliminare;
- rProcessPathToStart: se si vuole che SSEtoRserve lanci all’avvio un processo R (ad esempio Rserve), qui è possibile indicare il percorso all’exe desiderato. Se il processo muore, il plugin proverà ad eseguirlo nuovamente dopo circa 10 secondi. Di default questo attributo è non valorizzato, quindi si assume che Rserve stia già girando;
- rProcessCommandLineArgs: nel caso in cui il campo precedente sia configurato, qui possono essere specificati gli argomenti da passare al processo da avviare. Anche in questo caso di base il campo è lasciato vuoto, quindi nessun argomento è trasferito allo start dell’rProcess definito sopra;
- allowScript: è un flag (valori possibili: “true” o “false”) che consente di disabilitare il metodo “EvaluateScript” deputato all’esecuzione del codice. Di default è impostato a “true” e viene settato a “false” quando si vogliono limitare le funzionalità per questioni di sicurezza;
- functionDefinitionsFile: permette di specificare un file (ad esempio “FuncDefs.json”) in cui sono definite le proprie funzioni SSE. Di default questo campo non è valorizzato, quindi non sono aggiunte funzioni personali a quelle

base.

Riguardo quest'ultimo punto, le funzioni possono essere aggiunte senza la necessità di ricompilare il plugin, basta inserire le loro definizioni nel file JSON e queste verranno automaticamente mostrate come funzioni disponibili nel client Qlik. Un esempio di questo file è mostrato di seguito.

```
{
    "Functions" : [
        {
            "Id" : 0,
            "Name" : "ConcatStringsTensor",
            "FunctionType" : 2,
            "ReturnType": 0,
            "CacheResultInQlik" : "true",
            "FunctionRScript": "paste(q$str1,q$str2);",
            "Params" : {
                    "str1" : 0,
                    "str2" : 0
                }
        }
    ]
}
```

Figura 9.3: Struttura di un file JSON di definizione delle proprie funzioni

Come è possibile osservare, sono sette i parametri da configurare:

- Id: è un valore univoco che individua la funzione. Qlik invia questo ID a SSEtoRserve per identificare quale funzione deve essere eseguita;
- Name: è la stringa utilizzata nelle espressioni Qlik;
- FunctionType: identifica il tipo di funzione. Può assumere i valori 0, 1 e 2. 0 sta per "Scalar", cioè per ogni riga viene tornato un solo scalare; 1 indica "Aggregation", quindi più righe di dato come input e un singolo valore restituito come output; infine 2 indica "Tensor", cioè l'output avrà lo stesso numero di records dell'input;
- ReturnType: specifica il tipo del valore restituito: 0 per "String", 1 per "Numeric", 2 per "Dual", cioè sono ammessi sia stringa che valore numerico;
- CacheResultInQlik: è un flag che di default è impostato a "true" e consente di indicare se il risultato restituito deve essere cachato o meno;
- FunctionRScript: è lo script R che verrà eseguito nel momento in cui viene chiamata la funzione;
- Params: precisa il tipo dei parametri mandati da Qlik. I tipi possibili sono gli stessi definiti per il valore di ritorno.

È bene notare che le modifiche ai plugin richiedono il riavvio del motore QIX. È solo durante questa fase che Qlik tenta di contattare l'SSE chiamando il metodo

“GetCapability” il cui obiettivo è fornire le funzionalità implementate dal plugin.

È quindi possibile definire delle funzioni di plugin che saranno invocate da Qlik tramite il metodo RPC “ExecuteFunction”. Per richiamarle basterà usare la seguente formula “<EngineName>.<FunctionName>([<Parameter>...])” in cui “<EngineName>” è l’identificativo del plugin, “<FunctionName>” è la denominazione della funzione e “<Parameter>” è l’elenco dei parametri da passare alla funzione.

Di default SSEtoRserve include però anche otto script functions, cioè funzioni che accettano in ingresso direttamente il codice da eseguire scritto nel client Qlik. In questo caso il metodo RPC invocato da Qlik sarà “EvaluateScript” sempre che non sia stato disabilitato tramite l’attributo “allowScript” dal file di configurazione. Nella figura seguente sono schematizzati i tipi di queste script functions predefinite.

Function Name	Function Type	Argument Type	Return Type
ScriptEval	Scalar, Tensor	Numeric	Numeric
ScriptEvalStr	Scalar, Tensor	String	String
ScriptAggr	Aggregation	Numeric	Numeric
ScriptAggrStr	Aggregation	String	String
ScriptEvalEx	Scalar, Tensor	Dual	Numeric
ScriptEvalExStr	Scalar, Tensor	Dual	String
ScriptAggrEx	Aggregation	Dual	Numeric
ScriptAggrExStr	Aggregation	Dual	String

Figura 9.4: Script functions fornite da SSEtoRserve

Per invocare le prime quattro funzioni la chiamata deve essere di questo tipo: “<EngineName>.<FunctionName>(<Script> [,<Parameter>...])”. “<EngineName>” identifica il plugin; “<FunctionName>” indica il metodo da invocare; “<Script>” è una stringa che racchiude il codice da eseguire; “<Parameter>” include i parametri addizionali contenenti dati da Qlik (dovranno essere separati tramite virgola). Il secondo gruppo di quattro funzioni ha “Dual” come argument type, cioè significa che i parametri possono presentare differenti data types. Per questo motivo occorre chiamare queste funzioni allo stesso modo di quelle precedenti, tranne per il fatto che bisogna aggiungere un’informazione prima dello script specificante il tipo degli argomenti passati. L'espressione risultante ha così la forma “<EngineName>.<FunctionName> (<ParameterDataTypes>, <Script> [, <Parameter> ...]) in cui “<ParameterDataTypes>” è una stringa contenente i tipi di dati dei parametri, ordinati in base a come questi vengono specificati. Ad esempio, "NSD" significa che abbiamo fornito tre parametri, il primo è numerico (N), il secondo una stringa (S) e l'ultimo dual (D).

Nel nostro caso non è stata definita nessuna plugin function ma si è sfruttato solamente le script functions predefinite che di base offrono già un certo grado di libertà. Sono state mantenuti i settaggi standard tranne per quanto riguarda il parametro "rProcessPathToStart" valorizzato al percorso in cui risiede Rserve in modo che questa applicazione si avvii contestualmente al plugin.

L'ultimo passo mancante è quello della configurazione di Qlik Sense. Nel caso si parli della versione server, occorre definire una nuova analytic connection specificando il nome, cioè l'alias del plugin (lo imposteremo ad "R"), l'host, cioè l'indirizzo della macchina su cui è installato il plugin (sarà "localhost" dal momento che SSEtoRserve sta sul server Qlik), la porta su cui è in ascolto il plugin (50051, cioè quella di default) e lasciando gli altri campi con i loro valori di default. Se invece volessimo configurare il plugin per la versione desktop, bisognerebbe modificare il file "Settings.ini" aggiungendo la seguente riga "SSEPlugin=R,localhost:50051" specificante sostanzialmente gli stessi parametri settati sul server.

Infine, allo scopo di ottenere un'integrazione su tutti i fronti, tramite NSSM, che è un service helper, è stato definito un nuovo servizio denominato "SSEtoRserveService" a partire dal relativo .exe. Il Qlik Sense Engine Service è stato contestualmente editato aggiungendo alle dipendenze il servizio appena creato in modo tale che, appena il QES si avvia, parta anche il plugin.

9.3 Analisi delle serie storiche

Come chiarito nell'introduzione di questo capitolo, la volontà di incorporare R nell'architettura Qlik nasce dalla specifica finalità di analisi dei trend. Ovviamente l'uso di un motore di calcolo esterno fornisce tanti nuovi orizzonti di sviluppo ma in questo contesto muove verso una direzione ben precisa.

Lo studio dei trend rientra nell'ambito dell'analisi delle serie storiche. Per serie storica (o temporale) si intende un fenomeno che evolve nel tempo cioè la successione di dati osservati su un determinato evento (variabile y) ordinati secondo la variabile temporale t (per t = 1, 2, …, N). I dati si dispongono quindi ad intervalli più o meno regolari e il loro ordinamento non può essere alterato. Il significato del dato si arricchisce del fatto di essere osservato prima e/o dopo di un altro. In generale, se si considera una generica grandezza y che varia nel tempo t, si possono avere serie storiche continue se y è funzione continua del tempo t o serie storiche discrete se y si rileva solo in certi istanti (t = 0, 1, 2, 3, …, N). Ci sono diverse sfere di applicazione di una serie storica: fenomeni economici, aziendali, di produzione industriale, metereologici, medico-sanitari. Oltre a essere interessante in sé, capire il meccanismo generatore di una serie temporale può essere utile sia per controllare un fenomeno che per fare previsioni. Per quanto riguarda quest'ultimo caso si può fare un'ulteriore distinzione in serie storiche deterministiche e probabilistiche. Una serie storica è deterministica se conoscendo i valori passati è possibile prevederne con certezza, senza errori, i valori futuri. Questa tipologia di serie corrisponde a comportamenti teorici che nella realtà non possono verificarsi mai ma solo tutt'al più avvicinarsi al modello ideale. Si parla invece di serie storiche probabilistiche quando l'andamento temporale è caratterizzato da oscillazioni irregolari. Questi sono i processi

stocastici per cui le previsioni saranno sempre affette da errori poiché, pur conoscendo i valori passati, è possibile anticipare solo in parte l'evoluzione futura del fenomeno.

Una serie storica è costituita da quattro componenti principali:

- trend (T_t): è l'andamento nel lungo periodo della serie. Rappresenta il movimento tendenziale monotono di fondo che mette in evidenza un'evoluzione strutturale del fenomeno dovuta a cause che agiscono in modo sistematico sullo stesso;
- ciclo (C_t): è la componente congiunturale. Spiega le fluttuazioni pluriennali a periodicità irregolare, cioè gli scostamenti verso l'alto o verso il basso dei dati, le alternanze tra fasi di espansione e contrazione rispetto al trend legate solitamente all'andamento generale dell'economia;
- stagionalità (S_t): delinea il ripetersi regolare di effetti dovuto a fattori climatici (alternanza delle stagioni) e/o di organizzazione sociale che si ripetono annualmente;
- random (ε_t): mostra l'irregolarità, la casualità legata a disturbi imprevedibili di natura accidentale che determinano oscillazioni di brevi periodi.

I modelli di studio di una serie temporale sono due: composizione e decomposizione. Nel primo caso, conoscendo le componenti elementari e facendo una supposizione sul tipo di aggregazione, si desume la serie risultante. Nella seconda circostanza, invece, data una serie, si estrapolano gli andamenti costituenti per stabilirne le caratteristiche principali. I modelli di decomposizione sono i più interessanti nella pratica dal momento che, identificando le componenti di una serie, ci si può concentrare sulla tendenza di fondo e usare questa informazione a fini previsionistici. L'analisi delle serie storiche consiste così in un insieme di metodologie che permettono di scomporre l'andamento di una serie. Prima di addentrarci nella comprensione dei procedimenti atti a distinguere le varie componenti, è bene chiarire i legami che possono sussistere tra le stesse.

Esistono tre possibili modelli di relazione:

- modello additivo: presuppone l'indipendenza tra le componenti. Le fluttuazioni delle serie non variano col suo livello. Sia T_t che C_t, S_t ed ε_t sono espresse nella stessa unità di misura di y_t.

$$y_t = T_t + C_t + S_t + \varepsilon_t$$

- modello moltiplicativo: le componenti sono dipendenti tra di loro. C'è proporzionalità tra l'oscillazione della serie e il suo livello. Solo T_t (per convenzione) viene espressa nell'unità di misura di y_t mentre le altre componenti sono espresse come variazioni assolute.

$$y_t = T_t \times C_t \times S_t \times \varepsilon_t$$

  Questo modello può essere trasformato in uno additivo grazie all'uso dei logaritmi:

$$\log y_t = \log T_t + \log C_t + \log S_t + \log \varepsilon_t$$

- modello misto: presenta una parte additiva e una moltiplicativa. Nello specifico, è l'errore l'unica componente a sommarsi.

$$y_t = T_t \times C_t \times S_t + \varepsilon_t$$

La stima della componente ciclica presenta notevoli difficoltà anche perché il ciclo economico non mostra oscillazioni di carattere regolare. Pertanto, quello che solitamente si fa è considerare la componente ciclica unitamente alla componente del trend, cioè il trend-ciclo, definita anche come componente sistematica. Per cui, da questo momento in poi, quando si parlerà di trend si sottintenderà il suo legame vincolante con il ciclo.

Per la determinazione delle componenti si possono usare metodi empirici o analitici. Il più semplice metodo empirico per la valutazione del trend ma anche il più impreciso è quello della "curva ad occhio". Sostanzialmente si traccia una linea sulla serie ricorrendo al principio della conservazione delle aree: le aree al di sopra della linea devono essere uguali alle aree al di sotto della linea. Questo metodo, altamente rapido e approssimativo, può essere adatto solo per una stima formale ma non per un'analisi tecnica. Il sistema empirico effettivamente adoperato è quello delle medie mobili (MA – moving average). La media mobile è una media aritmetica (semplice o ponderata) che si sposta ad ogni nuova iterazione (ad ogni tempo t) dall'inizio verso la fine della successione dei dati. Praticamente consiste nel calcolo di una nuova serie storica in cui il termine relativo ad un determinato tempo è il risultato della media di k termini contigui della serie originaria. Se k è dispari ciascuna media mobile si riferisce al tempo centrale su cui è stata calcolata. Ad esempio, se k=3, il valore della media mobile riferito al tempo t è dato da:

$$MM3(y_t) = \frac{(y_{t-1} + y_t + y_{t+1})}{3}$$

Se k è pari è necessario calcolare la media di due medie mobili contigue per ottenere un valore centrato sui tempi della serie storica (media mobile centrata a k termini). Ad esempio, se k=4, il valore della media mobile riferito al tempo t viene ottenuto nel modo seguente:

$$MM4(y_{t-1,t}) = \frac{(y_{t-2} + y_{t-1} + y_t + y_{t+1})}{4}$$

$$MM4(y_{t,t+1}) = \frac{(y_{t-1} + y_t + y_{t+1} + y_{t+2})}{4}$$

$$MM4(y_t) = \frac{MM4(y_{t-1,t}) + MM4(y_{t,t+1})}{2}$$

L'applicazione delle medie mobili ha il vantaggio di essere elastica, adattarsi a qualunque tipo di andamento e smussare le oscillazioni di qualunque forma, lisciando la serie e riducendone la variabilità. Il limite è però quello di non ottenere stime per gli $n/2$ dati iniziali e finali. In particolare, con una serie mensile, si perdono sei termini all'inizio (T_t inizia dal 7° termine) e sei alla fine della serie. Utilizzando questa metodologia, per ricavare le varie componenti, occorre addentrarsi in una procedura iterativa che consta di sette passaggi:

1. Calcolo della prima componente di trend-ciclo: lo si fa tramite una media mobile centrata a dodici mesi. La MM12(y_t) dovrebbe eliminare le oscillazioni stagionali, e gran parte della componente erratica e quindi rappresenta una stima di prima approssimazione del trend-ciclo che possiamo identificare con $\hat{TC_t}$.
2. Calcolo preliminare della componente di stagionalità mista ad errore: la si ottiene sottraendo (nel caso di modello additivo) o dividendo (nel caso di modello

moltiplicativo) dalla serie originaria i valori calcolati al punto 1. Si parla rispettivamente di differenza lorda di stagionalità o rapporto lordo di stagionalità.

$$S_t + \varepsilon_t = y_t - \hat{TC_t} \qquad S_t \times \varepsilon_t = \frac{y_t}{\hat{TC_t}}$$

3. Calcolo della componente di stagionalità: se non si ha motivo di ritenere che nel corso del tempo la stagionalità si modifichi (stagionalità rigida o costante) allora si può procedere alla ulteriore scomposizione dei valori individuati in componente stagionale e accidentale. La stima della componente stagionale (S_t^*) si ottiene calcolando medie aritmetiche delle differenze lorde di stagionalità (nel caso di modello additivo) o dei rapporti lordi di stagionalità (nel caso di modello moltiplicativo) relativi allo stesso periodo (medie nei diversi anni del valore corrispondente allo stesso mese, trimestre, quadrimestre…). Si otterrà per ciascun periodo un coefficiente.
4. Costruzione della serie destagionalizzata: questa si ricava per differenza (o rapporto) tra la serie originaria e il risultato ottenuto al passo precedente.

$$D_t = y_t - S_t^* \qquad D_t = \frac{y_t}{S_t^*}$$

5. Stima della componente trend-ciclo della serie: se le stime dei coefficienti di stagionalità sono valide, la serie destagionalizzata non dovrebbe presentare oscillazioni stagionali. Si può allora ottenere una stima del trend-ciclo (TC_t^*) eliminando le fluttuazioni residue attraverso una media mobile con opportuno numero di termini (3, 5, 7 o più da verificare empiricamente). In questo modo si smussa la serie dagli effetti dei disturbi casuali.
6. Calcolo dell'intera parte sistematica della serie: arrivati a questo punto è possibile ricomporre la parte sistematica della serie, cioè quella costituita dalla componente trend-ciclo ottenuta allo step precedente e dagli effetti di stagionalità costanti calcolati al passo 3. I valori così ottenuti prescindono dagli errori accidentali.

$$y_t^* = TC_t^* + S_t^* \qquad y_t^* = TC_t^* \times S_t^*$$

7. Calcolo dei residui: questi sono ora ricavabili come differenza (o rapporto) tra la serie originaria e la componente sistematica calcolata al punto 6.

$$\varepsilon_t^* = y_t - y_t^* \qquad \varepsilon_t^* = \frac{y_t}{y_t^*}$$

La media mobile è un algoritmo molto semplice ed efficiente, anche se la stima del trend che si ottiene si può dimostrare essere meno accurata di quella ottenibile con il metodo analitico. Quest'ultimo procedimento viene così utilizzato specialmente quando lo scopo dell'analisi è di tipo previsionistico giacché si specifica il trend attraverso una funzione matematica della variabile temporale t. L'ipotesi alla base per l'applicazione di questa metodologia è che non compaia la componente stagionale o che la serie sia stata destagionalizzata in modo tale che la parte sistematica sia composta dal solo trend e quindi $y_t = f(t) + \varepsilon_t$ dove appunto T_t sia funzione del tempo f(t). La determinazione della f(t) può avvenire con qualsiasi funzione analitica, ovviamente dipende dall'andamento che si presume abbia la serie. Si utilizza l'analisi della regressione per trovare una curva che descriva il modello matematico con il miglior adattamento ai dati

della serie; si cerca quindi la miglior curva interpolatrice. Le forme funzionali più utilizzate sono le seguenti:

- funzione costante: $f(t) = \beta_0$

  L'andamento di fondo della serie storica è costante e la serie è quindi stazionaria (pattern orizzontale);

- funzione lineare: $f(t) = \beta_0 + \beta_1 t$

  β0 è l'intercetta e β1 è la pendenza della retta. Se β1 > 0, il trend è crescente; se β1 < 0, il trend è decrescente; se β1 = 0 esiste un pattern orizzontale (serie stazionaria, cioè si ricade nel caso precedente);

- funzione esponenziale: $f(t) = \beta_0 \beta_1^t$

  È caratterizzata da una crescita repentina e apparentemente senza limiti. Usando una trasformazione logaritmica si può rendere lineare: $\ln f(t) = \ln \beta_0 + t \ln \beta_1$;

- polinomio di secondo grado: $f(t) = \beta_0 + \beta_1 t + \beta_2 t^2$

  È un ramo di parabola, crescente o decrescente, convesso o concavo a seconda dei segni dei coefficienti.

Per compiere quest'operazione ci si avvale solitamente del metodo dei minimi quadrati che è una tecnica di ottimizzazione (o regressione) che permette di trovare una funzione, rappresentata da una curva ottima (o curva di regressione), che si avvicini il più possibile ad un insieme di dati. In particolare, la funzione trovata deve essere quella che minimizza la somma dei quadrati delle distanze tra i dati osservati e quelli della curva che rappresenta la funzione stessa.

Condotta questo studio sull'analisi delle serie storiche, siamo pronti ad applicare quanto appreso in una circostanza concreta.

9.4 Caso d'uso

Fatte le dovute considerazioni è arrivato il momento di passare alla pratica.

Ciò che è stato realizzato è l'aggiunta di un nuovo foglio denominato "Analisi dei trend" all'app del flow che consente di rispondere alle richieste utente sfruttando la potenza del motore statistico di R. In figura 9.5 è possibile osservare quanto prodotto.

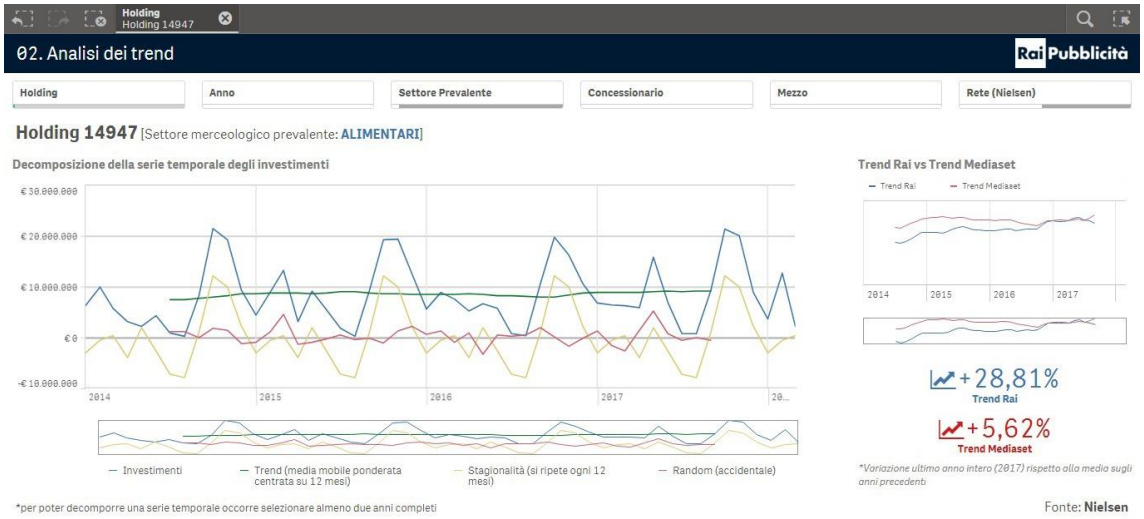

Figura 9.5: Foglio "Analisi dei trend"

Come spiegato nel paragrafo precedente, l'analisi delle serie storiche può essere compiuta per studiare il fenomeno, e si parla quindi di analisi descrittiva cioè la diagnostica che mira a tratteggiare un evento e le sue componenti per farne uno studio strutturale, o per fare previsioni, e si allude di conseguenza all'analisi predittiva, che punta invece a stimare un comportamento futuro. Nel contesto trattato, essendo l'app sviluppata destinata ad essere adoperata nella fase di interazione con i clienti, l'analisi predittiva risulterebbe fuori luogo perché significherebbe avere la presunzione di giudicare la loro condotta negli anni a venire. Per tale motivo, scendendo nello specifico, ciò che è stato reclamato è solamente la possibilità di studiare i trend degli investimenti per linea di prodotto per trarne dei pattern di azione. Per cui, oltre ad avere i soliti filtri di selezione disposti in alto e la visualizzazione "Testo e immagine" che delinea la holding scelta e il suo settore prevalente, gran parte del foglio è occupato da un grosso grafico lineare che decompone la serie storica degli investimenti. La dimensione prescelta è "Mese-Anno" dal momento che i dati sono sì per settimana, ma solo per i mezzi TV e radio; pertanto in questo contesto si è scelto di operare a livello mensile. Le linee raffigurate sono quattro rappresentanti le misure esposte: gli investimenti, il trend, la stagionalità, la parte random. Queste ultime tre riproducono le componenti costituenti la serie degli investimenti estrapolate utilizzando R. Nello specifico, il codice impiegato in ognuna di queste misure ha questa forma: "R.ScriptEval(Script [,Parameter...])".

Analizziamo le singole porzioni del codice:

- R: identifica il plugin configurato;
- ScriptEval: è la funzione di script che ammette come argomento un tipo numerico e che ritorna a sua volta righe numeriche;
- Script: qui occorre scrivere ciò che deve essere eseguito da R;
- Parameter: delinea i parametri contenenti dati Qlik da passare alla funzione.

Lo script di R che è stato settato è il seguente:

$$'newdata <- ts(q\$Investimenti, start = c(q\$StartYear, 1), frequency = 12);$$
$$decompose(newdata, type = "additive")\$component);'$$

Mentre i parametri sono:

$$Sum([Netto\ Accordo\ Quadro\ Nielsen])\ as\ Investimenti, Min(Anno)\ as\ StartYear$$

Quello che si fa è dunque creare in prima battuta l'oggetto time-series impiegando la funzione ts(). Vengono passati tre argomenti: i dati, cioè il vettore dei valori considerati; start, che indica il tempo della prima osservazione; frequency, che denota il numero di periodi per unità di tempo. Non viene invece specificato un punto di fine dal momento che si preferisce considerare tutti i valori senza imporre una data di conclusione della serie. Come è possibile notare, lo script R fa riferimento ai campi dati Qlik, passati come parametri, attraverso il q data frame. In questo modo si specifica che i dati costituenti la serie storica sono rappresentati dall'ipercubo restituito da Qlik riferente alla somma dei netti Nielsen. Allo stesso modo si precisa che il tempo della prima osservazione è rappresentato dal minimo anno alla selezione attuale. Dato che si è scelto di considerare la serie storica a livello mensile, il valore di "frequency" viene impostato a 12. Una volta originato l'oggetto time-series, il passo successivo è quello di estrapolarne le componenti. Per fare ciò si ricorre alla funzione decompose() che isola trend, stagionalità e componente irregolare e lo fa avvalendosi delle medie mobili. Nel dettaglio, il trend rappresenta la media mobile ponderata centrata su 12 mesi. Gli argomenti passati a questa funzione sono l'oggetto time-series ricavato al passo precedente e il tipo di modello (additivo o moltiplicativo). Da un'analisi svolta è emerso che la fluttuazione stagionale non varia proporzionalmente all'evoluzione del livello della serie per cui il tipo di modello è additivo. È ammesso anche un terzo argomento che non abbiamo adoperato utile a specificare un vettore di coefficienti da usare per filtrare la componente stagionale. Infine "$component", posto appena dopo la funzione decompose(), serve a specificare il sottoinsieme dell'oggetto ritornato da estrapolare, cioè la componente desiderata. Per questo motivo nelle tre misure illustrate precedentemente sarà valorizzato a "trend", "seasonal" o "random". Nello specifico, questa funzione determina innanzitutto il trend avvalendosi della media mobile (se il filtro è NULL, come nel nostro caso, viene utilizzata una finestra simmetrica con pesi uguali) e la rimuove dalle serie temporale. Viene poi calcolata la stagionalità facendo la media, per ciascuna unità di tempo (nel nostro caso per ogni mese), su tutti i periodi (cioè gli anni). La linea della componente stagionale è quindi centrata. Infine, la parte accidentale viene determinata rimuovendo il trend e la stagionalità dalle serie storica originale. Questo metodo funziona bene solo se il vettore di dati copre un numero intero di periodi completi, altrimenti non si riesce a dedurre nessuna componente stagionale. Per questo motivo in basso a sinistra viene aggiunta una nota dettagliante ciò. Questo line chart consente quindi di avere evidenza sulla tendenza degli investimenti dal punto di osservazione desiderato ma permette anche di determinare quali sono i periodi di aumento/calo ricorrenti sfruttando la stagionalità.

Sulla destra c'è poi un altro grafico lineare, stavolta più contenuto, che confronta il trend Rai con quello Mediaset, principale competitor. Questi andamenti sono ricavati ricorrendo ancora una volta ad R analogamente a quanto spiegato poc'anzi ma con la particolarità dell'uso della set analysis per impostare in maniera statica il concessionario. In questo modo si percepisce chiaramente l'andamento in Rai e in Mediaset e come la

forchetta tra i due nel corso degli anni si sia allargata o ristretta.

Infine ci sono due KPI che mostrano la percentuale di incremento/decremento del trend in Rai e Mediaset. In particolare, questi rappresentano la variazione negli investimenti tra l'ultimo anno intero (2017 allo stato attuale) e la media degli anni precedenti, facendo emergere altre indicazioni interessanti.

CAPITOLO 10

SETTIMA FASE DI SVILUPPO APP: PASSAGGIO IN PRODUZIONE

Sebbene sia necessaria una continua opera di manutenzione e ritocco, si giunge a un punto in cui lo sviluppo dell'applicazione è ritenuto ultimato e si procede allora con la fase di testing prima e con quella di confronto con l'utenza finale poi. Nonostante questo stadio sembri marginale rispetto agli altri, assume invece un'importanza capitale in quanto senza i necessari controlli e il placet del committente il lavoro è vacuo. Al compimento di questa fase, l'app può essere rilasciata in ambiente di produzione. Questo capitolo conclude l'implementazione del progetto.

10.1 Processo di sviluppo, test e rilascio

Come descritto nel settimo capitolo, l'app è stata implementata in ambiente di collaudo. Questo è la norma dal momento che, disponendo di due server, come spiegato in 2.1, è corretto operare in collaudo per sviluppare e verificare mentre si passa in produzione solo quando l'app è pronta a essere consegnata e quindi pubblicata. In precedenza è stato precisato come al progetto realizzato sia stato assegnato il PROJECTID 1002, denotante per l'appunto un lavoro ancora in corso d'opera.

Terminato lo sviluppo è doveroso procedere con i test per verificarne correttezza, completezza e affidabilità. Queste caratteristiche devono riguardare sia Qlik in sé avendo cura di ispezionare le reazioni alle varie selezioni, per appurare che il comportamento riscontrato sia quello atteso, che la nuvola dati esposta in modo da accertarsi che quanto mostrato sia valido. In questo stadio è necessario appoggiarsi a SQL Developer, eseguire a campione una serie di interrogazioni e confrontare i dati risultanti con quelli presentati dall'app. Se la comparazione dà esito positivo vuol dire che si è operato in maniera corretta, altrimenti significa che sono stati commessi degli errori in qualche passaggio riguardante la modellazione o la creazione delle misure e degli oggetti a front-end ed è quindi indispensabile tornare a uno degli steps precedenti per risolvere il problema.

Conclusa questa fase di revisione e manutenzione, l'app è pronta per essere sottoposta all'utenza committente. Occorre allora spostarla dallo stream "Lavoro", cui hanno accesso solo gli sviluppatori, ad "App da Rilasciare" cui invece sono abilitati tutti gli utenti con limitazioni di visibilità per app settate adoperando le custom properties. Così facendo l'utenza finale può proporre modifiche, adeguamenti e suggerire quant'altro si pensi possa migliorare la fruibilità e la chiarezza dell'informazione esposta. Nel nostro

caso sono state apportate solo piccole variazioni grafiche.

Compiuto quest'ultimo passaggio, l'app è pronta a essere portata in ambiente di produzione. Per fare ciò occorre assegnare un nuovo PROJECTID con un numero compreso tra 101 e 999 atto a palesare che il progetto è ora propenso a essere pubblicato. È stato attribuito il PROJECTID 102 e, contestualmente, sono state predisposte le tabelle del framework nello schema QLIKADM dell'istanza DWH sul server di produzione. Concluse tutte queste operazioni l'app può essere navigata sul nuovo ambiente. In prima battuta viene sempre posta su "Lavoro" in modo da essere ancora rivedibile e aggiornabile e per compiere ulteriori test sfruttando i dati reali. Svolti gli ultimi controlli, occorre pubblicare l'app e renderla visibile a chi di competenza. Produzione è pensato per avere più streams dedicati con controllo degli accessi specifico. In questo momento, però, c'è un solo stream ufficiale e la limitazione di visibilità è gestita per app, per cui il progetto è stato pubblicato sotto quest'unico flusso. Nella figura sottostante è possibile osservare il flow chart adottato come best practice per la gestione del processo di rilascio di nuove applicazioni o modifiche/evoluzioni di quest'ultime.

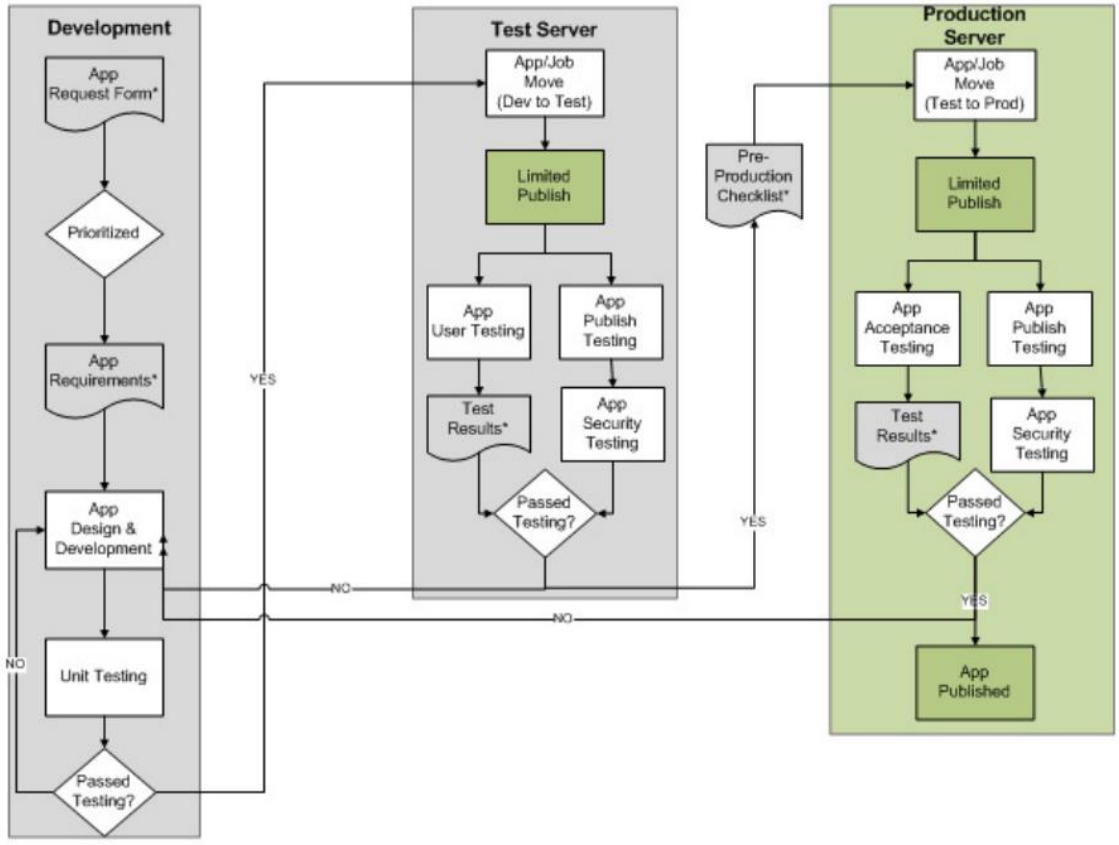

Figura 10.1: Flow chart descrivente il passaggio in produzione

CONCLUSIONE

Il lavoro realizzato ha consentito di acquisire nuove competenze teoriche e pratiche in ambito advertising, data warehouse e business intelligence. L'ambiente aziendale in cui è stata svolta l'attività ha permesso di toccare con mano numerose nozioni e paradigmi solo studiati in università. Ciò è stato particolarmente fruttuoso sia per accrescere le competenze personali sia a comprendere le dinamiche della cooperazione. I concetti di collaborazione, interazione tra colleghi con background differenti, rispetto delle scadenze, obiettivi da portare a termine, ravvisati solo in parte nei progetti universitari, sono stati affrontati con ancora più vigore e in un ambiente tutto nuovo. Si è operato in un contesto maturo e valido in cui c'è stato spazio sia per apprendere che per proporre e fornire il proprio contributo e punto di vista.

È stato possibile tastare la consistenza di un vero data warehouse, partecipare alla modellazione dati e redigere il catalogo informativo di quanto contenuto ed esponibile. È stato studiato il framework di ETL aziendale, analizzandone aspetti positivi e negativi, trovando falle e apportando modifiche al fine di ottimizzarne le prestazioni. Si è quindi compresa l'importanza che riveste il processo di estrazione, pulizia/trasformazione e caricamento dei dati, specialmente quando si tratta una grosse mole di informazioni, e l'attenzione che bisogna prestare ai carichi di lavoro e alle performance. Realizzando da zero un'app Qlik Sense, sono stati compiuti tutti i passi necessari a portare a termine un progetto: dall'interazione con l'utenza committente per comprendere le specifiche e le richieste al reperimento dei dati, dalla creazione degli oggetti necessari alla costruzione del modello fino alla realizzazione dell'interfaccia grafica e al passaggio che ha condotto dall'ambiente di collaudo a quello di produzione. Inoltre un approccio proattivo è sfociato nell'integrazione di R, dotando così il software aziendale di un nuovo tool utile a compiere analisi più approfondite orientate alla costituzione di modelli predittivi. La soddisfazione dell'utenza finale in merito a quanto prodotto è stata l'espressione finale di quanto di buono fatto.

Sviluppi futuri

L'app implementata realizza ampiamente quanto richiesto. Ovviamente ha bisogno di opportuna periodica manutenzione e testing per verificare che sia tutto corretto e coerente. È però anche ampliabile e migliorabile. L'idea base è quella di potenziare l'interfaccia grafica usando delle visualization extensions che la rendano più accattivante e appetibile. Quella più suggestiva prevede invece di rendere l'app un hub che consenta di accedere ad ambiti di studio specifici. Dal momento che l'analisi degli investimenti della clientela può includere una vasta gamma di informazioni, l'estensione della nuvola dati e la creazione di altre sfere di approfondimento è infatti un'ipotesi da vagliare.

Chiaramente, a seconda delle esigenze, possono sempre nascere nuove apps. In quest'ottica è evidente la necessità di partire dalla modellazione e strutturazione del data lake comune magari riorganizzandolo in base alle opportunità che via via si manifestano.

Riguardo il framework di ETL, invece, questo potrebbe essere ancora affinato introducendo nuove utilità. Ad esempio, si potrebbe gestire tramite le tabelle di settings anche il caricamento delle tabelle di mapping che allo stato attuale avviene via codice nella sezione "Custom". Il progetto più ambizioso è però quello di implementare un livello 0 di ETL, definito ETL di polling, che, monitorando periodicamente i cambiamenti, avvii automaticamente la catena dei reloads.

Infine la parte conclusiva del progetto è stata dedicata all'integrazione di R. L'utilizzo che se n'è fatto è stato limitato dal dominio di applicazione. R offre però una moltitudine di possibilità e quindi gli scenari di impiego che si parano davanti sono tanti e mirano principalmente all'analisi predittiva, prossimo ambito verso cui spingersi.

GLOSSARIO[11]

Campagna pubblicitaria: È un insieme coordinato di azioni pubblicitarie sostenute da una comune idea creativa che vengono condotte entro un arco temporale prestabilito su uno o più media, al fine di raggiungere degli obiettivi di comunicazione.

Concessionaria pubblicitaria: È una società che svolge l'attività di vendita degli spazi pubblicitari per conto degli editori, a fronte della quale viene remunerata con una percentuale del fatturato realizzato. La concessionaria di pubblicità opera come intermediario tra editori e agenzie: acquisisce in via esclusiva gli spazi pubblicitari dai proprietari dei mezzi di comunicazione e li vende ai centri media o, più raramente, direttamente agli inserzionisti. La gestione dei rapporti commerciali avviene attraverso una rete di agenti distribuiti a livello territoriale. L'attività di concessionaria può essere svolta sia da una società separata dall'editore che da una divisione interna dell'editore stesso; tale attività, inoltre, può essere svolta per conto di un solo editore (concessionaria diretta o monomandataria) o per più editori (concessionaria plurimandataria).

CPG (Cost Per GRP): È un indicatore utilizzato per valutare l'economicità di un piano media o di un singolo avviso. Esprime il costo di ogni GRP sviluppato sul target e si ottiene dividendo l'investimento totale di un piano mezzi (o il costo di un singolo spazio pubblicitario) per i GRP sviluppati dallo stesso.

CPM (Cost Per Mille): È una modalità di pianificazione e acquisto della pubblicità online. Esprime il costo per migliaia di impressions, laddove l'impression è un'unità di misura dell'esposizione pubblicitaria in rete. Il cost per mille o CPM può essere definito come il costo monetario corrisposto dall'inserzionista per ottenere un migliaio di impressions attraverso una campagna di display advertising. In origine cost per thousand, tale indicatore è oggi noto col nome latinizzato di cost per mille (CPM); il CPM ha soppiantato il CPI (cost per impression).

GRP (Gross Rating Point): È l'unità di misura della pressione pubblicitaria. Viene utilizzato per quantificare la pressione esercitata da una campagna pubblicitaria sul suo target group. Rappresenta, come suggerisce il nome, una misura grezza, poiché esprime la percentuale di target audience raggiunta dalla campagna pubblicitaria in termini lordi, cioè includendo le duplicazioni. Ciò nonostante questo indicatore si rivela particolarmente utile quando occorre valutare e confrontare le performance di piani o

[11] *Glossario Marketing*, < http://www.glossariomarketing.it >, ultima consultazione: 29/03/2018

veicoli diversi rispetto a un determinato target group. Il GRP, o indice della pressione pubblicitaria, è dato dal rapporto percentuale tra il numero di contatti lordi realizzati dal veicolo o piano con riferimento a un dato target e l'entità stessa del target. Più di frequente, viene determinato in termini assoluti moltiplicando la copertura o reach (il numero di individui appartenenti al target group che sono stati esposti almeno una volta alla comunicazione pubblicitaria) del piano mezzi per la sua frequenza o frequency (il numero di volte in cui il pubblico è esposto, o potenzialmente esposto, al messaggio pubblicitario); la formula da seguire, in tal caso, è pertanto la seguente:

$$GRP = Reach \times Frequency \times 100$$

Impression: È un'unità di misura dell'esposizione pubblicitaria in rete; esprime il numero di visualizzazioni di un annuncio pubblicitario servite da un advertising server a un utente in un dato intervallo temporale. È utilizzato come parametro base (ormai superato dall'utilizzo di parametri più raffinati e precisi) per la pianificazione di campagne di display advertising; la campagna viene pagata in base al numero di volte in cui l'annuncio è stato mostrato all'utente su uno o più siti web, indipendentemente dai clic sull'annuncio stesso.

Target: Letteralmente significa bersaglio ed è l'obiettivo che ci si ripromette di raggiungere con un prodotto, con una comunicazione o con uno o più mezzi di comunicazione. Il termine target group è usato per indicare un gruppo di consumatori (segmenti) accomunati da caratteristiche simili (età, reddito, stili di vita, bisogni, ecc…) al quale l'impresa si rivolge con una strategia di marketing e un marketing mix specifici.

Viewability[12]: È una metrica di pubblicità online utilizzata per misurare il tasso di impressions visualizzabili come definito dall'Interactive Advertising Bureau (IAB) e la Media Ratings Council (MRC). Attualmente perché un annuncio sia considerato visualizzabile, deve avere almeno il 50% dei suoi pixel in vista per 1 secondo (visualizzazione standard) o 2 secondi (video).

[12] *Programmatic RTB*, < http://www.programmatic-rtb.com/cose-definizione-viewability/ >, ultima consultazione: 29/03/2018

BIBLIOGRAFIA

De Toni A. F., Fornasier A., *La guida del Sole 24 Ore al Knowledge Management*, Gruppo 24 Ore, Milano, 2012

Laudon K. C., Laudon J. P., *Management dei Sistemi Informativi*, ed. italiana a cura di Pennarola F., Morabito V., Pearson, Milano, 2006

Howson C., *Successful Business Intelligence. Unlock the Value of BI & Big Data*, McGraw-Hill, Milano, 2013

Kimball R., Ross M., *The Data Warehouse Toolkit. The Definitive Guide to Dimensional Modeling*, Wiley, Indianapolis, Indiana, 2013

Golfarelli M., Rizzi S., *Data warehouse. Teoria e pratica della progettazione*, McGraw-Hill, Milano, 2006

Codice dei contratti pubblici, d.lgs. n.50/2016

Massari A., Montalti M., Oliveri A. P., *L'accordo quadro negli appalti pubblici. Analisi teorico-normativa e modelli operativi*, a cura di Massari A., Maggioli Editore, Rimini, 2013

SITOGRAFIA

Wikipedia: Italia, < https://it.wikipedia.org/wiki/Business_intelligence >, ultima consultazione: 14/01/2017

Oracle, < http://www.oracle.com/technetwork/developer-tools/sql-developer/what-is-sqldev-093866.html >, ultima consultazione: 19/01/2018

R Project, < https://www.r-project.org/about.html >, ultima consultazione: 21/01/2018

Wikipedia: Italia, < https://it.wikipedia.org/wiki/R_(software) >, ultima consultazione 21/01/2018

Qlik: Help, < http://help.qlik.com/ >, ultima consultazione: 28/03/2018

Oracle: Help Center, < https://docs.oracle.com/en/ >, ultima consultazione: 20/01/2018

DataBase and Data Mining Group: Politecnico di Torino, < http://dbdmg.polito.it/wordpress/teaching/ >, ultima consultazione: 04/03/2018

Wikipedia: Italia, < https://it.wikipedia.org/wiki/Data_warehouse >, ultima consultazione: 23/01/2018

HTML.it, < http://www.html.it/pag/16910/l-architettura-di-oracle/ >, ultima consultazione: 23/01/2018

Database Master, < http://databasemaster.it/dbms-oltp-dbms-olap/ >, ultima consultazione: 24/01/2018

DataBase Master, < http://databasemaster.it/utenti-e-schema-in-oracle/ >, ultima consultazione: 24/01/2018

Wikipedia: Italia, < https://it.wikipedia.org/wiki/Extract,_transform,_load >, ultima consultazione: 27/01/2018

Blog @ Applied Informatics, < http://blog.appliedinformaticsinc.com/etl-extract-transform-and-load-process-concept/ >, ultima consultazione: 27/01/2018

Eccellere – Business Community, < http://www.eccellere.com/Rubriche/Tecnologia/etl.asp >, ultima consultazione: 27/01/2018

MIT – Massachusetts Institute of Technology, < https://web.mit.edu/kerberos/ >, ultima consultazione: 12/02/2018

Wikipedia: Italia, < https://it.wikipedia.org/wiki/NTLM >, ultima consultazione: 12/02/2018

MSDN – Microsoft Developer Network, < https://msdn.microsoft.com/en-us/library/cc236627.aspx >, ultima consultazione: 12/02/2018

Nielsen: What People Watch, Listen To and Buy, < http://www.nielsen.com/it/it.html >, ultima consultazione: 19/02/2018

Wikipedia: Italia, < https://it.wikipedia.org/wiki/Nielsen_(azienda) >, ultima consultazione: 19/02/2018

comScore, Inc., < https://www.comscore.com/ita >, ultima consultazione: 19/02/2018

Wikipedia: Italia, < https://it.wikipedia.org/wiki/ComScore >, ultima consultazione: 19/02/2018

English Wikipedia, < https://en.wikipedia.org/wiki/Data_visualization >, ultima consultazione: 14/03/2018

Silvon – Driving the intelligent enterprise, < https://www.silvon.com/blog/data-visualization-brings-bi-life/ >, ultima consultazione: 14/03/2018

Highcharts, < https://www.highcharts.com/blog/post/role-data-visualization-business-intelligence/ >, ultima consultazione: 14/03/2018

CRAN: The Comprehensive R Archive Network, < https://cran.r-project.org/ >, ultima consultazione: 20/03/2018

GitHub Inc., < https://github.com/qlik-oss/server-side-extension >, ultima consultazione: 21/03/2018

GitHub Inc., < https://github.com/qlik-oss/sse-r-plugin >, ultima consultazione: 21/03/2018

Axis Group, < http://www.axisgroup.com/qlik-sense-server-side-extensions-part-13-architecture-environment/ >, ultima consultazione: 20/03/2018

Axis Group, < http://www.axisgroup.com/qlik-sense-server-side-extensions-part-23-simple-example-plugin/ >, ultima consultazione: 20/03/2018

Qlik | Community,
< https://community.qlik.com/blogs/qlikviewdesignblog/2017/07/07/automatically-start-the-r-plugin >, ultima consultazione: 20/03/2018

GRPC, < https://grpc.io/ >, ultima consultazione: 20/03/2018

Wikipedia: Italia, < https://en.wikipedia.org/wiki/GRPC >, ultima consultazione: 20/03/2018

NSSM – the Non-Sucking Service Manager, < https://nssm.cc/ >, ultima consultazione: 21/03/2018

Federica Web Learning: UNINA, < http://www.federica.unina.it/economia/statistica-per-le-decisioni-impresa/modelli-lineari-analisi-serie-storiche/ >, ultima consultazione: 24/03/2018

DISIA: UNIFI, < http://local.disia.unifi.it/marliani/stat_eco_A/lucidi_SS.pdf >, ultima consultazione: 24/03/2018

DIS: UNIROMA1, < http://www.dis.uniroma1.it/~desantis/NOTE/timeseries.pdf >, ultima consultazione: 24/03/2018

LVPROJECT,
< http://www.lvproject.com/images/ANALISI%20SERIE%20STORICHE.pdf >, ultima consultazione: 24/03/2018

Didattica: UNIROMA2,
< http://didattica.uniroma2.it/assets/uploads/corsi/34414/Lucidi_serie_storiche.pdf >, ultima consultazione: 24/03/2018

Wikipedia: Italia, < https://it.wikipedia.org/wiki/Metodo_dei_minimi_quadrati >, ultima consultazione: 25/03/2018

Glossario Marketing, < http://www.glossariomarketing.it >, ultima consultazione: 29/03/2018

Programmatic RTB, < http://www.programmatic-rtb.com/cose-definizione-viewability/ >, ultima consultazione: 29/03/2018

ELENCO FIGURE

RINGRAZIAMENTI

Concluso il lavoro, giunge il tempo della gratitudine.

In primo luogo desidero ringraziare la professoressa Baralis, relatrice universitario e punto di riferimento accademico, per la sua disponibilità e cortesia. Assoluta riconoscenza va poi a Rai Pubblicità, che mi ha accolto e mi ha offerto la possibilità di addentrarmi in un mondo nuovo. Grazie al mio tutor aziendale Mauro Sandri, per avermi seguito, consigliato, guidato. E grazie anche a tutti i dipendenti dell'ufficio IT e ai consulenti per il loro costante apporto. Non si sono mai tirati indietro dal dispensare suggerimenti, spiegare e aiutarmi, mi hanno insegnato tanto.

La tesi rappresenta l'ultimo pezzo di un puzzle cominciato nel settembre 2015, fatto di tanti piccoli e grandi quadratini. Come si sa, per assemblare al meglio un puzzle, bisogna partire dagli angoli. Io parto dalla mia famiglia, a cui va il mio ringraziamento più grande per esserci, sempre. Mamma e papà fin da piccolo mi hanno insegnato il valore del sacrificio, della volontà e della meritocrazia, mi hanno supportato ed elogiato. Mio fratello Fabio, invece, è da sempre il mio punto di riferimento. È lui a cui miro per il successo che riesce ad ottenere in qualunque ambito di cui si occupa. E per ultimo mio nonno, che non c'è più, ma che con le sue parole mi ha costantemente spronato a fare meglio, spinto ad avventurarmi, a provare, a non lasciarmi sopraffare dalle difficoltà. È anche merito suo se ho scelto di trasferirmi a Torino. So che adesso da lassù è orgoglioso di me.

Tra i tanti pezzi del puzzle ce n'è uno più grande degli altri. È la famiglia del collegio Einaudi, che mi ha ospitato, cresciuto e maturato. Gli scherzi, le risate, i pranzi, le cene, le notti, le feste, le grigliate, i litigi, le amicizie, le passioni. C'è stato un po' di tutto. Chi più chi meno, chi di passaggio, chi saldamente, ha rappresentato parte di questi anni. Cito Elisa, supporto morale, che mi è stata vicino in questi mesi, ha creduto in me e mi ha dato la forza per raggiungere questo traguardo. Ci sono poi i cheesecakers, che hanno un nome buffo ma anche un cuore grande. Eva, Antonio, Salvo, Elena, Sofia, Angela, Claudia, grazie per i momenti passati insieme. E infine tutti gli altri. Siete troppi e anche se non vi nomino un pensiero va ad ognuno di voi.

Non posso poi non ringraziare i miei colleghi, con cui ho condiviso le giornate in università, lo studio, gli esami. Ma anche le serate, il calcetto, le vacanze. Grazie a Emanuele, Marco, Biagio, Leandro, Dario, Carmelo, Mario, Tano, Peppe e tutti gli altri. Molti di loro sono diventati come fratelli, adesso probabilmente le nostre strade si divideranno ma sono consapevole del fatto che la nostra amicizia è forte e sana e quindi ci sentiremo e vedremo comunque.

E infine grazie agli amici nuovi, Ada e Vale, che in poco mi hanno già trasmesso tanto, e a quelli di sempre, Angelo, Ciccio, Alberto che mi fanno ancora respirare l'odore di Sicilia quando ritorno a casa.