Università degli Studi di Modena e Reggio Emilia

Facoltà di Ingegneria di Modena

Corso di Laurea Specialistica in Ingegneria Informatica

PROGETTO E REALIZZAZIONE DI TECNICHE DI BUSINESS INTELLIGENCE PER L'ANALISI AMBIENTALE

Relatore:
[Chiar.mo] Prof.ssa Sonia Bergamaschi

Correlatore:
[Dott.] Ing. Simone Smerieri

Candidato:
Abdul Rahman Dannaoui

Anno Accademico 2008/2009

INDICE

1. INTRODUZIONE

La presente tesi e stata svolta durante un periodo di stage presso l'azienda "Quix Srl" di Soliera.
Lo stage si è concentrato in generale sull'utilizzo di tecnologie di *Business Intelligence*. In particolare, e` stato affrontato: lo studio del modello multidimensionale OLAP; il processo di definizione e creazione di un *data warehouse* e dei cubi contenuti al suo interno; lo studio di PDI (*Pentaho Data Integration*) che è uno strumento ETL(*Extraction, Transformation, Load*) *open source*. Si e` inoltre affrontato lo studio di BART, che è uno strumento di *Business Intelligence* sviluppato dall'azienda, in grado di definire cubi di dati OLAP, basati su strutture dati relazionali e di interrogarli per realizzare *report*. Tali *report* forniscono un supporto all'analisi di dati raccolti dalla base di dati di SIAM, che è un gestionale *web-based* sviluppato sempre nell'azienda ed usato dalle provincie italiane per la gestione delle pratiche ambientali presentate da cittadini italiani presso gli uffici provinciali.

1.1 Il progetto di stage

L'azienda Quix aveva già sviluppato quattro cubi che riguardano: gli atti, le attività, le pratiche e le tempistiche usando BART e un pacchetto di query SQL per l'alimentazione periodica di questi quattro cubi sul dominio dei rifiuti e dell'acqua.
Inoltre, l'azienda usava le strutture dei cubi con quattro diversi RDBMS: **MySql, Postgres, Oracle e SqlServer** e quindi aveva tenuto quattro diversi file per l'alimentazione periodica delle strutture dati e per la loro creazione durante l'installazione di SIAM (ognuno di questi file corrisponde a un dialetto dei RDBMS utilizzati) .
Si è quindi utilizzato PDI per realizzare un'unica trasformazione che si interfaccia con i diversi RDBMS utilizzati, sfruttando la capacità di PDI di interfacciarsi con RDBMS usando i driver JDBC.
Inoltre, sono state affrontate le diverse fasi:analisi, progettazione, alimentazione periodica, interrogazione e formulazione di nuovi *report* per un nuovo cubo che riguarda il dominio dell'aria.

L'articolazione della tesi e` la seguente.
Nel capitolo 2, saranno introdotti i concetti base del *data warehouse* e saranno spiegati i diversi passi per la creazione di una nuova struttura per il supporto all'analisi dei dati, che è il cosiddetto *star schema* di un *data warehouse*, partendo dalla definizione della struttura fino alla realizzazione dei *report* di analisi.

Nel capitolo 3, saranno analizzate le tecnologie utilizzate per creare la trasformazione dei dati e il nuovo cubo relativo all'aria. In particolare, nella prima parte, verrà spiegato lo strumento di ETL e come esso viene utilizzato per la creazione dei flussi di caricamento dei dati nelle tabelle dei cubi, spiegando in dettaglio le impostazioni e le opzioni più importanti per l'utilizzo di questo strumento; verrà introdotto un esempio di un flusso a scopo illustrativo.
Nella seconda parte del capitolo, verrà spiegato BART (lo strumento della reportistica): come è realizzato, quali sono i moduli principali e le tecnologie che usa questo strumento (sviluppato nella tesi di Mattia Bonacorsi [5]). In particolare, nel secondo paragrafo verranno brevemente illustrati: *Mondrian*, che è il server OLAP e *JPIVOT*, che è una libreria che permette di visualizzare il risultato delle interrogazioni. Nel terzo paragrafo, verrà illustrato brevemente il linguaggio MDX che è il linguaggio dell'interrogazione del cubo multidimensionale e, infine, nell'ultimo paragrafo sarà spiegato come utilizzare BART attraverso un esempio completo.

Nel capitolo 4 sarà descritta l'attività svolta durante lo stage. Questo capitolo è diviso in 2 parti; la prima spiega la trasformazione fatta per risolvere il problema dell'eterogeneità dei dialetti utilizzati dei diversi database e descrive la differenza tra il vecchio metodo utilizzato per il caricamento delle tabelle. Viene inoltre mostrato il vantaggio di utilizzare PDI come strumento di ETL.

Nella seconda parte, saranno spiegati i passi della creazione del nuovo cubo. In particolare, vengono spiegati i requisiti del cliente, poi viene mostrato il procedimento dell'analisi del cubo e la sua progettazione. Infine, sarà spiegato come vengono realizzati i flussi di caricamento, come viene interrogato il cubo e come viene visualizzato il risultato.

Nel quinto capitolo, saranno presentate le conclusioni e i possibili sviluppi futuri, mentre nel sesto capitolo verrà introdotto un glossario con i termini utilizzati in questa tesi e nel settimo capitolo sarà presentata la bibliografia utilizzata.

2. GENERALITÀ SUI *DATA WAREHOUSE (*DWH*)*

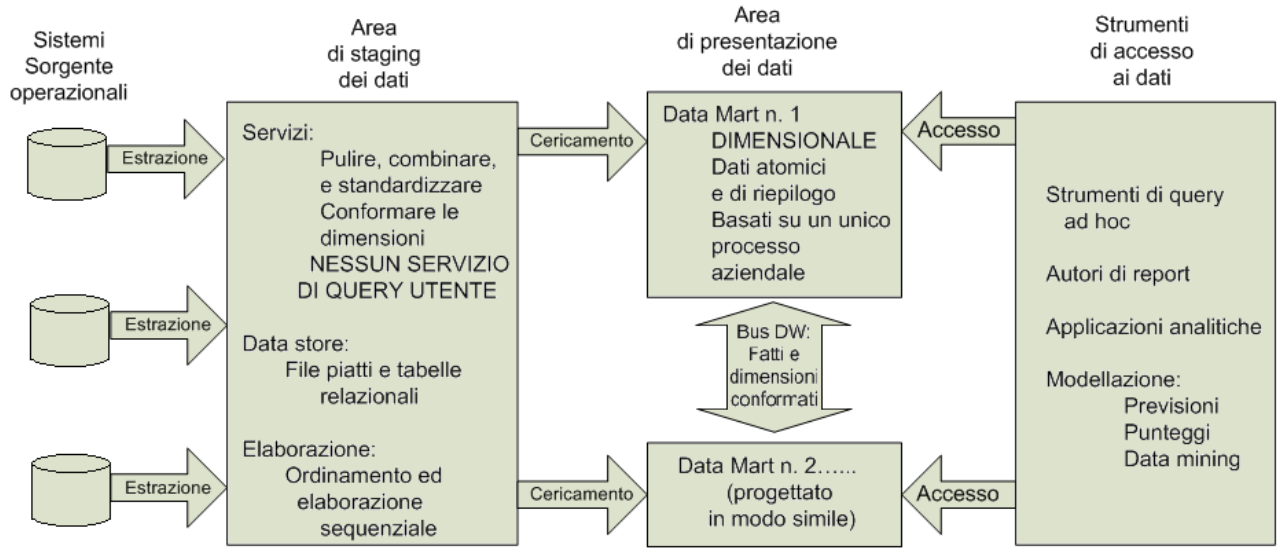
Prima di iniziare a trattare le tecnologie utilizzate durante lo svolgimento del progetto dello stage, è opportuno iniziare con l'introduzione di alcuni concetti molto importanti che servono a capire cos'è un *data warehouse*, a cosa serve e quali sono le fasi nella progettazione di un *data warehouse*, premesse la progettazione, parte degli obiettivi e delle definizioni, con diverse fasi porta ad ottenere il risultato finale, i *report*, che offrono le informazioni analitiche che servono all'azienda.

In un mercato molto competitivo, come quello odierno, è di notevole importanza prendere decisioni in tempi brevi. Per migliorare i processi decisionali si devono analizzare tutti i dati disponibili dell'azienda. In particolare, è necessario analizzare dinamicamente il mercato, in modo da capirne i meccanismi e prevederne gli andamenti. Per rendere di facile consultazione i dati di svariati sistemi e provenienti da differenti applicazioni realizzate per scopi diversi, si rende necessaria la realizzazione di un *data warehouse*.

Il *data warehouse* rappresenta un nuovo approccio per fornire accesso alle informazioni dell'impresa con lo scopo di trovare risposta alle richieste degli utenti del livello decisionale. L'approccio tradizionale di analisi dei dati si basa sull'uso di strumenti semplici, basati su un linguaggio naturale o formale come SQL, per effettuare interrogazioni (query). Questo approccio tuttavia diventa inefficiente su grandi quantità di dati. Un approccio più moderno è quello denominato *On Line Analytical Processing* (OLAP), che si basa sulla predisposizione di una vasta gamma di query che sintetizzano i dati in base alle regole aziendali. OLAP è più rapido perché si basa su dati precedentemente aggregati e pertanto più vicini alle richieste degli utenti. Il rischio è quello di scartare dati di dettaglio di eventuale interesse. Per evitare questo rischio è possibile utilizzare appostiti *tool* di *Data Mining*, che consentono analisi più approfondite, sfruttando tecniche sviluppate nei campi della statistica e delle *machine learning*, per esempio le reti neurali.

2.1 Componenti di un *DATA WAREHOUSE*

La figura che segue mostra i diversi componenti di un *data warehouse*, che costruiscono i suoi tre livelli, a prescindere dell'architettura che viene utilizzata:



Il **primo livello** è composto dai sistemi transazionali che memorizzano i dati di base che possono essere dei sistemi di supporto operativo. Tra questi, i più diffusi sono gli ERP (*Enterprise Resource Planning)*, sistemi che, alla gestione della produzione e della distribuzione, integrano le applicazioni informatiche per i processi di supporto, quali amministrazione, gestione delle risorse umane e contabilità). CRM (*Customer Relationship Managment)*, sistemi per la gestione delle relazioni con i clienti. Tali sistemi sfruttano basi di dati transazionali, memorizzano cioè singole transazioni riferite a eventi gestionali; si pensi a tutte le transazioni che vengono memorizzate giornalmente da una società che gestisce una catena di 100 ipermercati dislocati in tutta Europa, ciascuno dei quali vende migliaia di prodotti ogni giorno.

Questo livello potrebbe essere in realtà pensato come un livello esterno al *data warehouse*, visto che non abbiamo nessun controllo sul formato dei dati e sulla loro qualità.
Le interrogazioni per questo livello sono di solito piuttosto standardizzate e coinvolgono pochi record alla volta e spesso le sorgenti non sono integrate tra loro, cioè ogni applicazione ha il suo *database*, senza condivisione di dati comuni.

Tra il **primo e il secondo livello** si trova la *data staging area* in cui si trovano alcuni strumenti software specialistici di ETL (*Extraction – Trasfomation- Load*) che sono degli strumenti dedicati alla mappatura, "pulizia" e trasferimento dei dati nei *database* fisici del secondo livello.

Il **secondo livello ovvero la *data presentation Area*** che è costituito dalle basi dati direzionali, realizzate con approcci logici di *data warehousing* o di *data marting*: questi basi dati integrano fonti diverse di dati e separano l'ambiente operativo e transazionale del sistema informativo aziendale dall'ambiente di analisi e produzione delle informazioni manageriali.

Un ***data warehouse*** è un archivio di informazioni, raccolte da numerosi database operativi, in grado di supportare le attività di analisi aziendale e i processi decisionali. Inmon (1996) è stato il primo a parlare di *data warehouse* definendoli come una raccolta di dati integrata, orientata ai dati e focalizzata su un soggetto di *business*, con un ampia memoria storica, in grado di fornire supporto alle decisioni. I *data warerhouse* vengono spesso percepiti come strumenti con dimensioni proporzionali all'azienda da cui sono utilizzati. Tuttavia per alcuni utenti è necessario accedere solo ad una parte di tali informazioni e per questo motivo un'organizzazione può ricorrere alla creazione di uno o più ***data mart***. Quest'ultimo è un sottoinsieme di un *data warehouse*. La Land's End, azienda produttrice di capi di abbigliamento è stata fra le prime a creare un *data warehouse* utilizzabile da tutti i dipendenti dell'azienda. Questo sistema non fu praticamente usato dai dipendenti per il semplice motivo che la dimensione dell'archivio era sovradimensionata rispetto alle necessità degli individui, che incontravano notevoli difficoltà nel trovare informazioni elementari. L'azienda ha deciso di creare *data mart* settoriali come quello per il *merchandising* che contiene solo le informazioni relative alla vendita.

Attualmente, l'approccio più usato nella realizzazione del *data warehouse* è quello di **Ralph kimBall** che parla di un *data warehouse* fatto partendo dai singoli *data mart*, tra di loro indipendenti e tali che ognuno di essi tratta un argomento e fatti diversi dall'altro; l'insieme di essi costruisce il *data warehouse.* Nei paragrafi successivi, quando si parla di progettazione di *data warehouse,* si farà riferimento alla progettazione del singolo *data mart*. *Data warehouse e data mart*, a loro volta, alimentano il terzo livello.

Il **terzo livello** è quello dei sistemi di *business intelligence*, che comprendono diversi strumenti software che interrogano un cubo OLAP (che sarà discusso nei paragrafi successivi) che viene generato partendo dallo schema base definito nella fase di progettazione del *data warehouse*.

2.2 Caratteristiche di un *DATA WAREHOUSE*

Generalmente in un'azienda la struttura dei dati è orientata ad ottimizzare i processi aziendali (l'emissione di un nuovo ordine o di una fattura, il carico e lo scarico del magazzino ecc), mentre il *data warehouse* è orientato ad un concetto aziendale (le vendite o gli acquisti) e contiene tutte le informazioni correlate al concetto, raccolte da vari sistemi di elaborazione. Una delle caratteristiche di un *data warehouse* è l'integrazione. Essa nasce dalla necessità di dare coerenza alle diverse rappresentazioni dei dati provenienti da applicazioni progettate per scopi diversi. Ci troviamo quindi ad affrontare il problema di rendere i diversi dati disponibili in azienda, accessibili ed omogenei in un unico ambiente, ma questo pone difficoltà, alcune delle quali sono analizzate nel seguito:

• Gli attributi che si riferiscono allo stesso argomento possono essere definiti in modo diverso, ad esempio il codice di un cliente può essere definito come una stringa oppure come un numero, quindi bisogna scegliere il tipo più adatto e ricondurre tutti gli attributi codice allo stesso tipo;

• Applicazioni che usano sistemi diversi di grandezze in base alle quali vengono effettuati i calcoli, ad esempio, ci possiamo trovare di fronte al totale dell'ordine espresso in lire in un'applicazione ed espresso in euro in altre;

Un *data warehouse* deve essere non-volatile. Inoltre, l'utente finale non deve poter cambiare i dati in esso contenuti, poiché il *data warehouse* viene usato per fare indagini e non per inserire o modificare operazioni. Nel *data warehouse* non si deve andare a modificare l'indirizzo di un cliente, anche perché in tal caso si perderebbe ogni riferimento storico al fatto che il cliente ha cambiato indirizzo. I dati vengono caricati solitamente in massa con una certa periodicità (solitamente di sera), per non appesantire il sistema e successivamente possono essere consultati dall'utente finale.

Una caratteristica importante di un *data warehouse* è la dipendenza dal tempo. In un *database* aziendale le operazioni accessibili, di solito, sono quelle dell'ultimo anno, in un *data warehouse* l'intervallo temporale si allarga fino ad arrivare a coprire più anni contemporaneamente. In ambiente gestionale, il *database* contiene il valore corrente di un dato (ad esempio l'ultimo numero di telefono di un cliente) e questo dato può essere modificato perdendo ogni riferimento al dato precedente, mentre in un *data warehouse* i dati possono essere visti come delle foto istantanee (*snapshot*) fatte in determinati momenti, perciò tengono conto anche della storia dei soggetti. In un sistema gestionale si può o meno fare riferimento ad elementi temporali (giorno della settimana, anno, ora, ...), mentre in un *data warehouse* si deve sempre fare riferimento a qualche elemento di tempo.
I dati presenti in un *data warehouse* devono essere consistenti. Questo significa che se due persone interrogano l'archivio in momenti diversi per conoscere le vendite avvenute nel mese di Aprile, devono ottenere lo stesso risultato; inoltre se i dati di un determinato periodo, per qualche motivo, non sono stati caricati completamente, l'utente che li richiede deve essere avvisato dell'incompletezza dei dati.

2.3 I Metadati

In un *data warehouse* i metadati, ovvero i dati sui dati, giocano un ruolo di primo piano, la possibilità di condividere e riutilizzare i metadati riduce di molto il costo e la complessità di sviluppo nonché la gestione e l'utilizzo dei *data warehouse*.
I metadati sono utili allo sviluppatore come prima documentazione della struttura del *data warehouse* e dei processi di trasformazione che subiscono i dati, all'analista servono a capire come sono stati ottenuti i dati salvati nel *data warehouse* e quindi a formulare in modo più preciso le sue interrogazioni. Tipicamente i metadati tengono conto di:

- Struttura dei dati transazionali,
- Trasformazioni dei dati transazionali,
- Sorgente dei dati transazionali,
- Modello dei dati transazionali e multidimensionali,
- Routine utilizzate per accedere ai dati multidimensionali.

2.4 Progettazione del *DATA WAREHOUSE*

il primo passo da fare durante la realizzazione di un *data warehouse* è la progettazione della struttura delle sue tabelle e per fare ciò, si devono considerare alcuni aspetti importanti:

• La granularità, cioè il livello di dettaglio dei dati salvati nel *data warehouse*. Più alto è il livello di dettaglio e più bassa è la granularità e viceversa. La granularità è direttamente legata al volume di dati salvato e, di conseguenza, alle prestazioni del sistema. Ovviamente bisogna scegliere il giusto livello di granularità per evitare di memorizzare informazioni che non verranno mai prese in considerazione e per evitare di non registrare informazioni importanti;

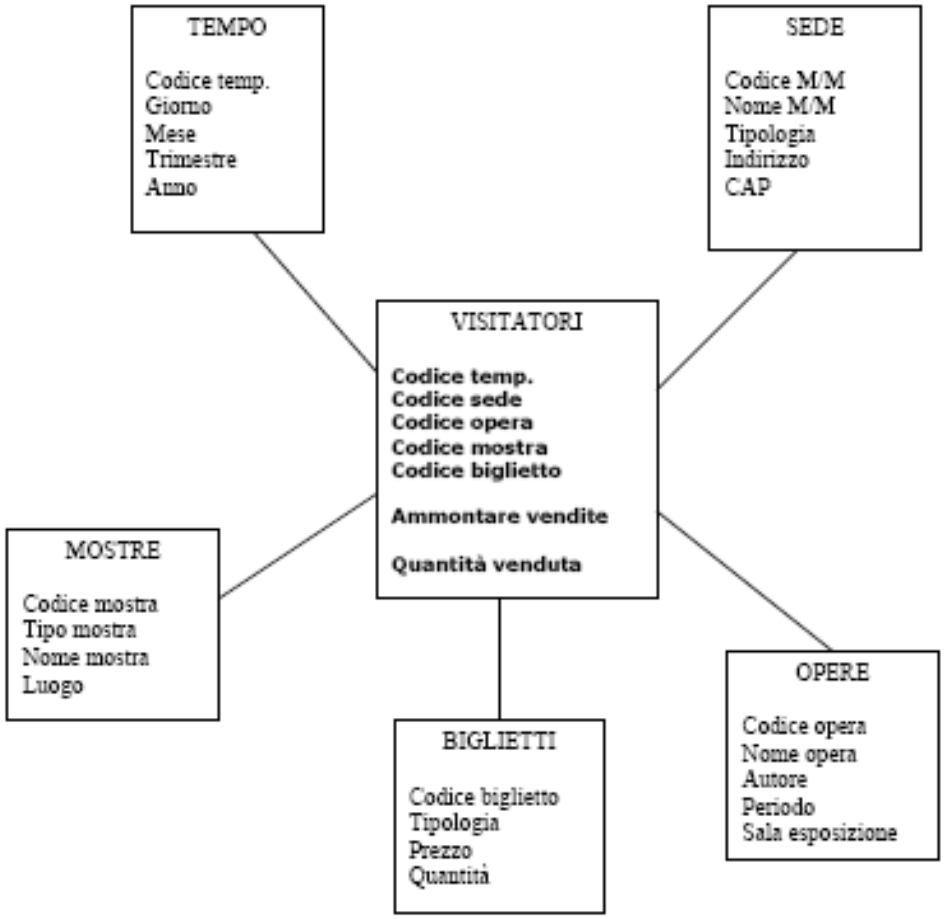
• Il tempo di aggiornamento del *data warehouse*, cioè il periodo che intercorre tra due successivi aggiornamenti dei dati contenuti nel *data warehouse*;

• Il partizionamento dei dati. Il partizionamento dei dati si ha quando i dati contenuti in una stessa struttura logica vengono divisi in più di una unità fisica ed inoltre un dato appartiene ad una ed una sola partizione;

Nel *data warehouse* è importante stabilire come partizionare i dati in modo che ciascuna unità fisica di dati possa essere manipolata indipendentemente dalle altre. Il partizionamento presenta alcuni vantaggi: maggior facilità di creare indici, riorganizzare, recuperare i dati e monitorare le operazioni degli utenti. La fase di sviluppo del processo di *data warehousing* inizia spesso dalla creazione di un modello aziendale dimensionale che descrive i valori metrici e le dimensioni più importanti dell'area di argomenti selezionata in base ai requisiti degli utenti.
A differenza dei sistemi di elaborazione delle transazioni in linea (OLTP) che organizzano i dati con metodologie di normalizzazione, i dati nel *data warehouse* sono organizzati in modo non normalizzato per migliorare le prestazioni delle query.

I dati di un *data warehouse* sono organizzati secondo una semplice struttura, detta schema multidimensionale o più semplicemente schema a stella (*Star Schema*).
Il primo termine mette in luce la presenza di molteplici dimensioni di analisi; il secondo la struttura "stellare" dello schema, una volta che esso venga interpretato con il modello Entità-Relazione classico.

La principale caratteristica dello schema a stella è la sua struttura regolare e indipendente dal problema considerato.
In ambiente informativo è indispensabile ristrutturare i dati e configurare correttamente il sistema in modo da soddisfare le necessità degli utenti finali ai quali ci si rivolge. In tal modo si consente al DBMS di elaborare le query (le interrogazioni) che si rivolgono al sistema, tenendo bene a mente che il segreto del successo verso tale comunità è molto spesso la semplicità della struttura dati con la quale questo deve interfacciarsi. Il modello dimensionale va proprio incontro a questo obiettivo di semplicità.
la figura seguente mostra un semplice esempio di schema a stella per l'analisi dei dati di un museo, al cui centro è stata posta la tabella dei fatti (*fact table*), mentre le entità satelliti più piccole sono dette delle dimensioni (*dimension table*).



Al contrario degli schemi Entità-Relazione tradizionali, lo schema a stella è evidentemente asimmetrico: c'è una tabella dominante di grandi dimensioni al centro del diagramma, ed è l'unica a possedere collegamenti multipli, che si concretizzano attraverso identificatori univoci (campi chiavi) in essa contenuti, con le altre tabelle; queste ultime hanno tutte un solo collegamento (join) alla tabella centrale, con lo scopo di minimizzare il numero di join richiesti per ciascuna query.
A volte durante la fase di progettazione del *data warehouse*, non sempre è chiaro se un campo estratto dal database operazionale sia un attributo dimensionale piuttosto che un fatto; spesso il dubbio può essere risolto osservando se il campo ha la caratteristica di variare il proprio valore con continuità (o meglio, ogni volta che viene rilevato), proponendosi come una potenziale misura dell'attività da inserire nella *fact table*; oppure se è più ragionevole considerarlo come la descrizione di qualcosa che è più o meno costante e quindi inserirlo come attributo dimensionale.
Un ruolo fondamentale degli attributi delle tabelle dimensionali è quello di essere usate all'interno dei vincoli nelle interrogazioni sul *data warehouse* o come intestazioni delle colonne negli output per gli utenti finali .

Nello schema a stella della figura precedente, la *fact table* contiene i fatti: ammontare vendite, e quantità venduta, mentre le *dimensional tables* ospitano ognuna la descrizione funzionale della dimensione del *business* presa in considerazione nella *fact table*, dettagliando i parametri rilevanti ai fini dell'analisi.
Per estendere l'esempio fatto precedentemente si potrebbero aggregare le vendite rispetto alle mostre per cercare di capire se le varie mostre hanno successo oppure no, oppure si potrebbero aggregare i visitatori per tipologia di biglietto per verificare quali sono le classi di utenti (giovani, adulti, ecc, ….) che maggiormente frequentano il museo per poi prendere, di conseguenza, decisioni appropriate.
Per la visualizzazione di dati multidimensionali, la soluzione ottimale è la tecnologia OLAP, alla lettera "*On Line Analytical Processing*".

2.5 Strumenti ETL

Il secondo passo nella realizzazione di un *data warehouse* consiste nel caricare le tabelle della struttura già creata prendendo i dati transazionali e ciò viene fatto usando degli strumenti ETL.
Per popolare un *data warehouse,* il primo passo è quindi di ricondurre i dati ad un unico formato passando attraverso la conversione e decodifica dei dati. Il processo di trasformazione dei dati in un formato integrato e uniforme deve essere inoltre automatizzato, in modo da poter essere eseguito periodicamente, con la frequenza necessaria per soddisfare i requisiti aziendali del *data warehouse.*
Durante il processo di alimentazione del *data warehouse*, la riconciliazione avviene in due occasioni: quando il *data warehouse* viene popolato per la prima volta, e periodicamente quando il *data warehouse* viene aggiornato.
Gli strumenti ETL fanno le seguenti operazioni:

Estrazione: I dati rilevanti vengono estratti dalle sorgenti e ci sono due tipi di estrazioni:
L'estrazione **statica** viene effettuata quando il *data warehouse* deve essere popolato per la prima volta e consiste concettualmente in una fotografia dei dati operazionali, mentre l'estrazione **incrementale** viene usata per l'aggiornamento periodico del *data warehouse*, e cattura solamente i cambiamenti avvenuti nelle sorgenti dall'ultima estrazione, e comunque la scelta dei dati da estrarre avviene in base alla loro qualità,

Pulitura: Si incarica di migliorare la qualità dei dati delle sorgenti dati duplicati, ad esempio: inconsistenza tra valori logicamente associati, dati mancanti, uso non previsto di un campo, valori impossibili o errati, valori inconsistenti per la stessa entità dovuti a differenti convenzioni, valori inconsistenti per la stessa entità dovuti a errori di battitura, ecc, …….

Trasformazione: Converte i dati dal formato operazionale sorgente a quello del *data warehouse*. La corrispondenza con il livello sorgente è complicata dalla presenza di fonti distinte eterogenee, che richiede una complessa fase di integrazione ad esempio: presenza di **testi liberi** che nascondono informazioni importanti oppure utilizzo di **formati differenti** per lo stesso dato.
Durante questa fase si applicano anche gli algoritmi che calcolano le misure della tabella dei fatti; le misure sono infatti attributi calcolati, quindi, non basta solo estrarre i dati dalla sorgente dati ma bisogna anche usare i dati estratti per calcolare queste misure.

Caricamento: il caricamento del *data warehouse* può essere fatto in due modi:

- **a.*Refresh*:** I dati del *data warehouse* vengono riscritti integralmente, sostituendo quelli precedenti (tecnica normalmente utilizzata solo per popolare inizialmente il *data warehouse*)

b.*Update*: I soli cambiamenti occorsi nei dati sorgente vengono aggiunti nel *data warehouse* (tecnica normalmente utilizzata per l'aggiornamento periodico del *data warehouse*).

Quando vengono creati i flussi ETL per popolare il *data warehouse*, essi vengono schedulati per essere eseguiti quando le tabelle del *data warehouse* non sono in uso per evitare problemi ed errori dovuti alla inconsistenza dei dati; di solito vengono schedulati di notte quando i dati sono tanti e i flussi ETL impiegano molto tempo per caricare le tabelle del *data warehouse*.

2.6 OLAP e formulazione dei *report*

Ora che il *data warehouse* è stato creato ed i dati sono caricati, si passa alla fase di analisi e quindi alla formulazione dei *report*, si necessita quindi di un metodo per visualizzare questi dati interrogandoli in modo efficace e veloce.
Per la visualizzazione di dati multidimensionali, la soluzione ottimale è la tecnologia OLAP, che fornisce gli strumenti per accedere, esplorare e analizzare dati multidimensionali in un modo semplice e "naturale" e fornisce una vista multidimensionale dei dati.

Gli schemi a stella sono approssimazioni relazionali del modello di dati OLAP e possono rappresentare un punto di partenza eccellente per la creazione del concetto chiave dell'OLAP, il cubo multidimensionale.
In un modello di dati OLAP, le informazioni vengono gestite concettualmente come cubi, composti da categorie descrittive (dimensioni) e valori quantitativi (misure). Per l'esempio precedente, le dimensioni del cubo saranno:tempo, sede, mostre, biglietti e opere, che corrispondono alle dimensioni dello *star schema* e le misure saranno: Ammontare vendite e quantità venduta, che corrispondono ai fatti della TF. All'interno di ogni dimensione di un modello di dati OLAP, i dati sono organizzati in una gerarchia che rappresenta i livelli di dettaglio dei dati. Per esempio, nell'ambito della dimensione Tempo esisteranno i livelli: Giorno, Mese, Anno, Trimestre. Una determinata istanza del modello OLAP includerebbe i valori specifici per ogni livello della gerarchia. Un utente dei dati OLAP si sposterà verso l'alto o il basso (*Roll UP, Drill Down*) di tale gerarchia per visualizzare i vari livelli dei dati e scegliere il livello di dettaglio delle informazioni desiderato. I cubi, le dimensioni, le gerarchie e le misure sono gli elementi essenziali dell'esplorazione multidimensionale OLAP. Grazie a questo modello di descrizione e presentazione dei dati, gli utenti hanno la possibilità di esplorare in modo semplice e intuitivo un *set* di dati complesso.

Il risultato che viene visualizzato nei *report* è l'analisi dei dati caricati nel cubo che dà le informazioni che servono ai manager per prendere le decisioni circa le strategie dell'azienda. Sono quindi il frutto del *data warehouse* realizzato e qui c'è da dire che più il *data warehouse* è progettato bene, più il risultato visualizzato nei *report* è significativo e corretto.

3. LE TECNOLOGIE UTILIZZATE NEL PROGETTO

3.1 PDI- Lo strumento di ETL

Durante lo svolgimento dello stage, è stato utilizzato *Pentaho Data Integration* versione 3.2.0 (detto anche *KETTLE* o anche *Spoon*) che è uno strumento *open source* della piattaforma *pentaho*. *Kettle* è la componente di *pentaho* responsabile della *Extraction, Transformation and Load* (ETL) dei processi.

Sebbene gli ETL siano gli strumenti più frequentemente utilizzati in ambienti di *data warehouse*, *Kettle* può essere utilizzato anche per altri scopi:

-Migrazione di dati tra applicazioni o *database*
-Esportazione di dati da *database* a file flat
-Caricamento massiccio di dati nei *database*
-Pulizia di dati
-Integrazione di applicazioni

Kettle è facile da usare. Ogni processo è creato con uno strumento grafico in cui si specifica che cosa fare, senza la scrittura di codice per indicare il modo di farlo; a causa di questo, si potrebbe dire che *Kettle* è orientato ai metadati.

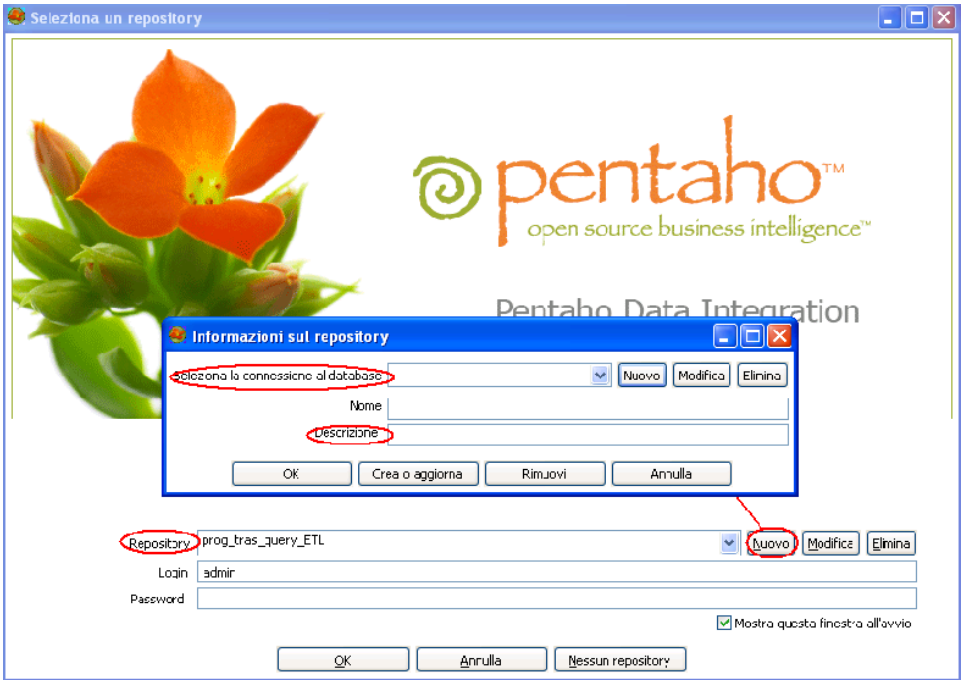
Kettle può essere utilizzato come applicazione *stand-alone*, oppure può essere utilizzato come parte di una *Pentaho Suite* più vasta. Come uno strumento di ETL, è il più popolare *tool open source* disponibile. *Kettle* supporta una vasta gamma di formati di ingresso e di uscita, compresi file di testo, file Excel, e *database* commerciali oppure *free*. Inoltre, la capacità delle trasformazione di *Kettle,* permette di manipolare i dati con pochissime limitazioni.

Kettle è un *grafical user interface* che permette di progettare delle trasformazioni e di JOB che sono in grado di descrivere se stessi utilizzando un file XML o possono essere messi nel *database* di *Kettle*. In sintesi, *Pentaho Data Integration* rende più facile la costruzione, l'aggiornamento e la manutenzione del *data warehouse*.

Kettle sopporta le seguenti piattaforme:

* Microsoft Windows, vista incluso
* Linux GTK
* Apple OSX
* Solaris: utilizzando una interfaccia Motif
* AIX: utilizzando una interfaccia Motif
* HP-UX: utilizzando una interfaccia Motif
* FreeBSD: supporto preliminare su i386, non ancora su x86_64

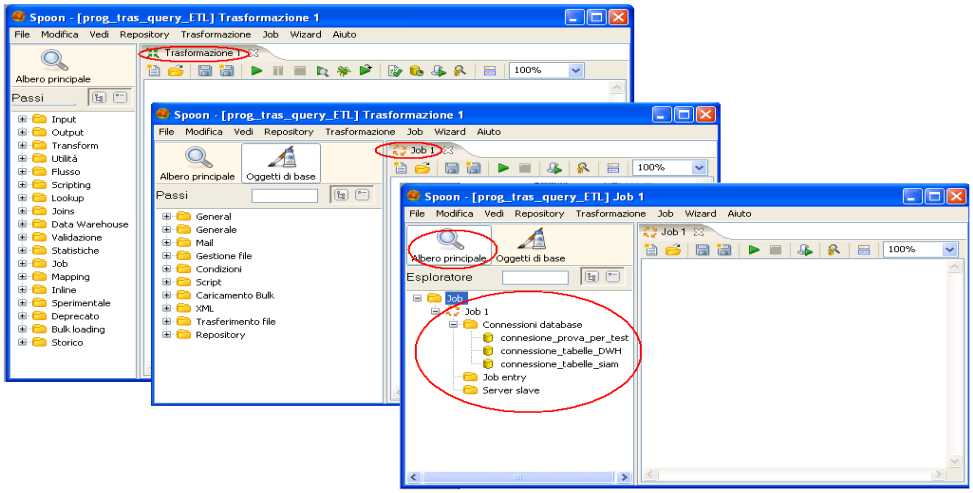
Per concludere questa introduzione su *Kettle*, rimane da ricordare che *Kettle* permette di memorizzare i file di trasformazione ed i JOB nel *file system* locale o nel *repository* di *Kettle,* che può essere creato in qualsiasi *database* relazionale comune. Per caricare una trasformazione da un *database repository,* si deve collegare a questo *repository*.



3.1.1 Interfaccia grafica

Quando viene lanciato *Kettle*, l'interfaccia grafica che viene visualizzata è la seguente:

E quindi dall'albero che si trova a sinistra si decide di scegliere di creare una trasformazione oppure un JOB. E secondo la scelta fatta, viene visualizzata una tela per disegnare una trasformazione o un JOB.



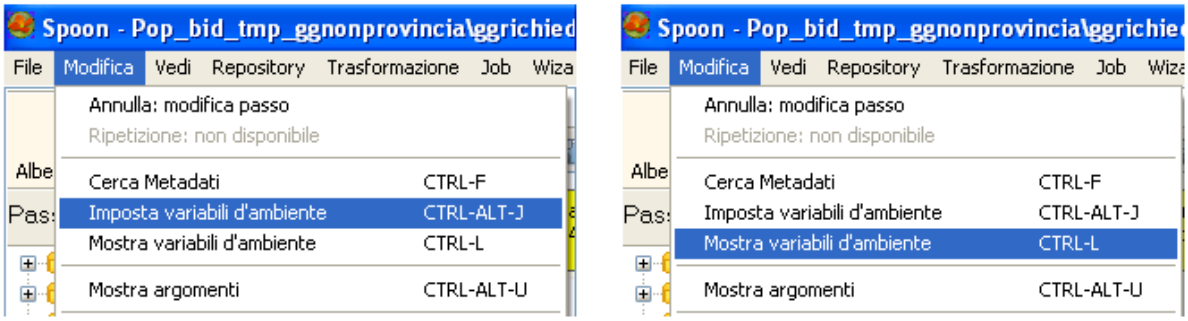
L'albero principale in alto a sinistra del pannello di *Spoon* permette di esplorare le connessioni associate con i JOB e le trasformazioni che si sono aperte. Nel disegnare una trasformazione, il *Core Objects*, in basso a sinistra del pannello, contiene le misure disponibili per costruire la trasformazione, incluso *input*, *output*, *Lookup*, *Transform*, *scripting* e di più. Durante la progettazione di un JOB, il *core objects* contiene voci di JOB disponibili. La barra del *Core Objects* contiene una varietà di tipi di *JOB entry*.

3.1.2 Le icone della barra degli strumenti

le icone della barra degli strumenti sono da sinistra a destra

Icona	Descrizione
	Crea un nuovo JOB o trasformazione
	Apri trasformazione / JOB da un file nel caso non si sia connessi a un *repository* oppure da un *repository* se si è connessi ad uno
	Salva una trasformazione / JOB in un file o in un *repository*
	Salva una trasformazione / JOB con un nome diverso o un *filename* diverso
	Apri la *window* della stampante
	Esegui trasformazione/ JOB. Esegui la trasformazione corrente da un file XML oppure dal *repository*
	Anteprima della trasformazione: esegue la trasformazione corrente dalla memoria senza salvarla. Si può vedere l'anteprima delle righe prodotte selezionando questo pulsante
	Esegui la trasformazione in *debug mode*, che permette di risolvere i problemi degli errori dell'esecuzione
	Riesegui la trasformazione in una certa data o ora
	Verifica la trasformazione: esegue dei controlli per ogni passo per vedere se tutto funziona correttamente
	Esegue un analisi d'impatto: quale sarà l'impatto della trasformazione sul *database* utilizzato
	Genera lo script SQL che serve per fare le stesse modifiche che fa la trasformazione
	Esplora il *database*, permette di vedere i dati in anteprima, eseguire query SQL sulla tabella, generare DDL e altro

3.1.3 Variabili d'ambiente



A volte si ha bisogno di utilizzare delle variabili che vengono cambiate dinamicamente, oppure anche per sfruttare la trasformazione o il JOB in un altro progetto, quindi realizzare la modularità richiesta nei progetti informatici. Per fare ciò si usa la funzione "Imposta variabile d'ambiente" che permette di creare e passare dei valori ai parametri definiti nelle trasformazioni o nei JOB.
Questa funzione è accessibile selezionando **Modifica | Imposta variabile d'ambiente** dalla barra dei menu, come si vede nell'immagine sopra riportata. Ed è comunque visualizzata quando si esegue una trasformazione che utilizza delle variabili indefinite e ciò permette di impostarle prima di eseguirla.
Nell'immagine riportata sopra, viene visualizzata anche la funzione, "Mostra Variabili D'ambiente" che permette di visualizzare l'elenco di tutte le variabili d'ambiente ed i loro valori. Essa è accessibile selezionando **Modifica | Mostra variabili d'ambiente** dalla barra del menu.

3.1.4 Creazione di una trasformazione / JOB

È possibile creare una nuova trasformazione in uno dei tre modi:
1. Cliccando file--> Nuovo --> trasformazione sulla barra degli strumenti Principale
2. Cliccando Nuovo, quindi Trasformazione
3. Utilizzando il tasto CTRL-N

Una qualsiasi di queste azioni apre una nuova vista grafica per iniziare a designare la trasformazione.

È possibile creare un nuovo JOB in uno dei tre modi:
1. Cliccando il pulsante file --> Nuovo --> JOB sulla barra degli strumenti principale
2. Cliccando Nuovo, quindi JOB
3. Utilizzando il tasto CTRL-ALT-N

Una qualsiasi di queste azioni apre una nuova scheda per iniziare a designare un nuovo JOB.

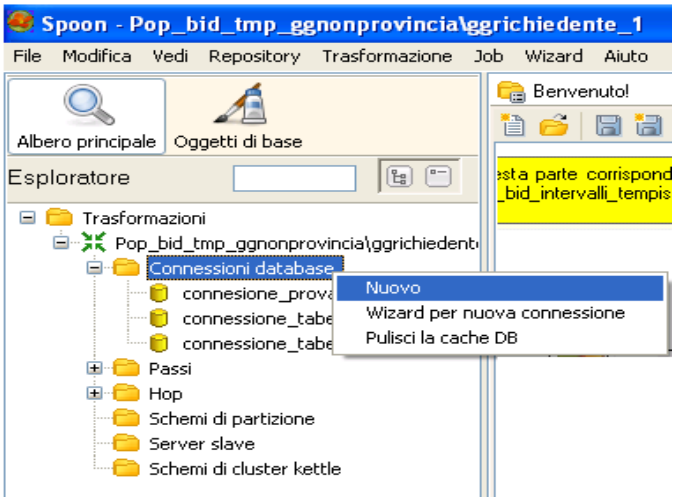
Note aggiuntive :
Le note aggiuntive permettono di aggiungere delle note di testo descrittivo per il JOB o per la trasformazione e si può fare come segue:
* Per aggiungere una nota nella vista grafica, si fa right - click nella tela e si seleziona aggiungi nota
* Per modificare una nota, si fa doppio click sulla nota
* Per eliminare una nota, si fa right - click sulla nota e si seleziona **Elimina nota**.

3.1.5 Connessioni *database*

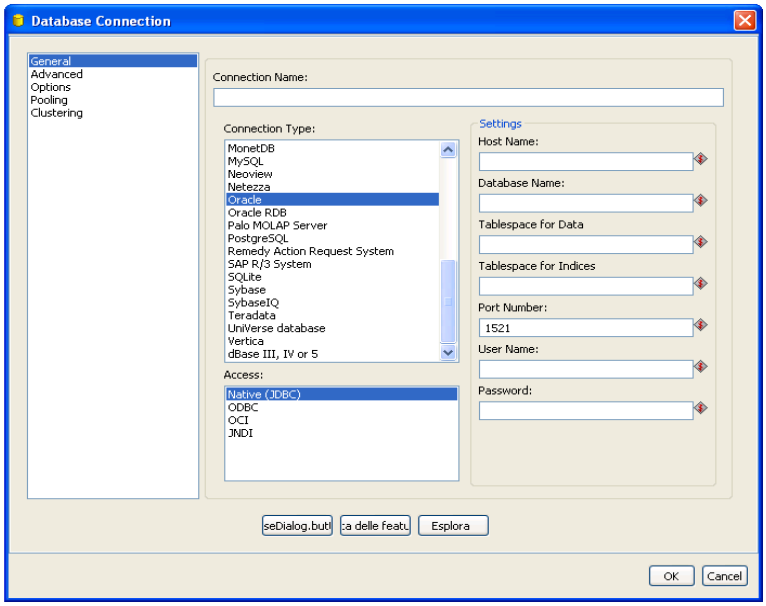
Una connessione *database* descrive il metodo con cui *KETTLE* si connette a un *database*. È possibile creare collegamenti specifici per un JOB o per una trasformazione o di memorizzarli nel *repository* di *KETTLE* per il riutilizzo in più trasformazioni o JOB.

3.1.5.1 Creazione di un nuovo collegamento

Per creare una nuova connessione, si fa right - click su **Connessioni Database** nell'albero principale a sinistra e si seleziona **Nuovo** o **Wizard per nuova connessione**.
È anche possibile fare doppio click su **database connection**, o premere **F3**.



la *dialog box* ***Database Connection*** appare. Di seguito verranno spiegate i tab di questa *dialog box.*



General: In questa tab si possono impostare le informazioni di base sulla connessione, come il nome della connessione, il tipo, il metodo di accesso, il nome del server e le credenziali per accedere. La tabella che segue fornisce una descrizione dettagliata delle opzioni disponibili nella tab **General**:

Funzione	Descrizione
Connection Name	Identifica univocamente una connessione tra le trasformazioni e i JOB
Connection Type	Tipo di *database* a cui si connette (per esempio, MySQL, Oracle, e così via)
Access	Questo sarà uno Native (JDBC), ODBC, o OCI. I Tipi di accesso disponibili variano a seconda del tipo di *database* a cui ci si connette
host name	Definisce il nome *host* del server su cui risiede il *database*. È inoltre possibile specificare *l'host IP-address*
Database name	Identifica il nome del *database* a cui si desidera connettersi. In caso di ODBC, viene specificato qui il nome DSN
Port number	Qui viene impostato il numero di porta
Username	Facoltativo, specifica il nome utente per la connessione al *database*
Password	Facoltativo, specifica la *password* per la connessione al *database*

***Pooling* :** Prima di spiegare questa tab facciamo una breve definizione della **connection pool**. La *connection pool* è una cache di connessioni al *database* che viene utilizzata per migliorare le performance di esecuzione dei comandi su un *database*. Una volta che una connessione è stata creata, viene messa nella *connection pool* e viene riutilizzata più volte in modo tale da non dover ricreare una connessione ad ogni richiesta di dati.

Questa tab permette di configurare la connessione al *database* per utilizzare una *connection pool* e definire delle opzioni collegate alla *connection pool* come: la dimensione iniziale, la dimensione massima e i parametri della *connection pool*. La tabella che segue fornisce una descrizione più dettagliata delle opzioni disponibili nella *pooling* tab:

Funzione	Descrizione
Use a connection pool	Abilita la *connection pool*
The initial pool size	Imposta la dimensione iniziale della *connection pool*
The maximum pool size	Imposta il numero Massimo delle connessioni nella *connection pool*
Parameter Table	Permette di definire ulteriori parametri personalizzati nella *connection pool*

Clustering : Questa tab consente di attivare la creazione di cluster per la connessione al *database* e creare connessioni alle partizioni dei dati. Per consentire la creazione di cluster per la connessione, si attiva la opzione 'ENABLE CLUSTER'.
Per creare una nuova partizione di dati, si mette un *ID e host name*, la porta del *database*, nome utente e la *password* per il collegamento alla partizione.

Advanced : Questa tab consente di configurare le seguenti proprietà per la connessione al sistema SAP:

Funzione	Descrizione
Quote all in database	Specifica la lingua da utilizzare quando si collega a SAP
Force all to lower case	Specifica il numero dei sistemi SAP a cui si vuol connettere
Force all to upper case	Specifica il numero di clienti (di tre cifre) per la connessione

Il pulsante Esplora : Il *Database Explorer* consente di esplorare interattivamente il *database* di destinazione, visualizzare in anteprima i dati, generare DDL e molto altro ancora. Per aprire il *Database Explorer* per una connessione esistente, si clicca il pulsante 'Esplora' o si fa right - click sulla connessione nell'albero principale e si seleziona 'Esplora'.

Il pulsante Feature list: La *feature list* espone l'elenco JDBC URL, la classe, e le varie impostazioni per la connessione database come ad esempio l'elenco delle parole riservate.

3.1.5.2 Operazioni con la connessione

Modifica di una connessione :Per modificare una connessione esistente, si fa doppio click sul nome della connessione nell'albero principale oppure right - click sul nome della connessione e si sceglie 'Modifica'

Duplicazione di una connessione: Per duplicare una connessione esistente, si fa right - click sul nome della connessione e si seleziona 'Duplica'.

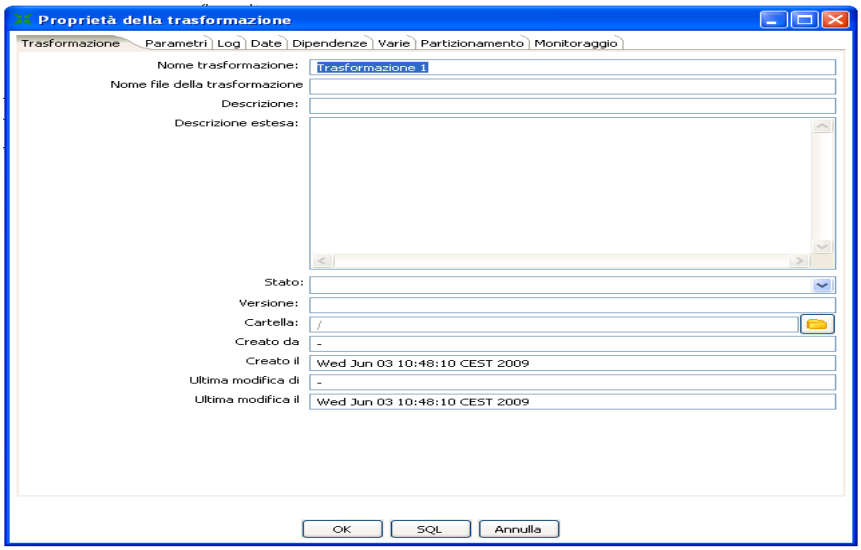
Eliminazione di una connessione : Per eliminare una connessione al database, si fa right - click sul nome della connessione nell'albero principale e si seleziona 'Elimina'.

Svuotare la cache del *database*: Questa opzione viene usata per accelerare le connessioni *Spoon* che utilizzano un cache di *database*. Quando le informazioni nella cache non rappresenta più la struttura del *database*, si fa right click sulla connessione nell'albero principale delle connessione e si seleziona 'Pulisci la cache DB della connessione ...'. Questo comando è usato comunemente, quando le tabelle del *database* vengono modificate, create o eliminate.

3.1.6 Le impostazioni di una trasformazione

Le Impostazioni di una trasformazione sono un insieme di proprietà che descrivono la trasformazione e configurano il suo comportamento. Si può Accedere alle impostazioni della Trasformazione dal menu principale sotto 'Trasformazione | parametri' oppure cliccando <CTRL+T>.

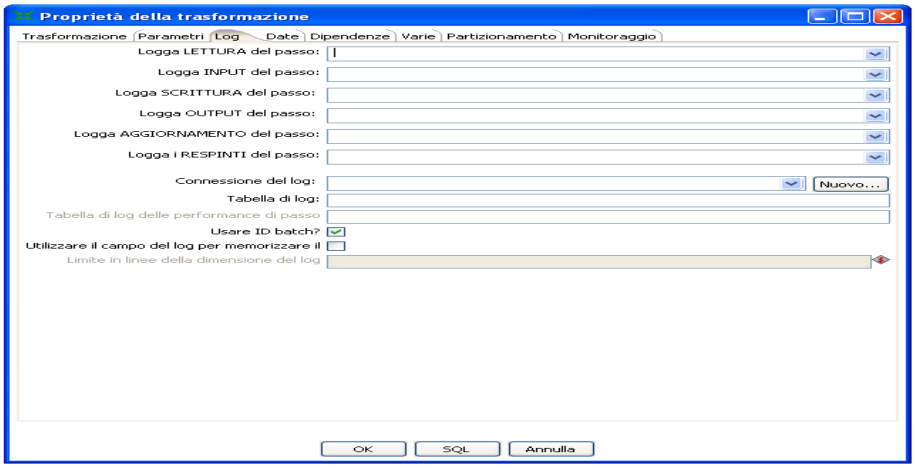
3.1.6.1 Tab trasformazione



La tab trasformazione consente di specificare le caratteristiche generali circa la trasformazione, tra cui:

Opzione	Descrizione
Nome trasformazione	Il nome della trasformazione. Obbligatorio nel caso volessimo salvarla in un *repository*
Descrizione	Una breve descrizione della trasformazione che viene visualizzata quando si esplora la *repository*
Descrizione estesa	Una descrizione estesa della trasformazione
Stato	Bozza oppure in produzione
Versione	Descrizione della versione
Cartella	La cartella nella *repository* dove è salvata la trasformazione
Creato da	Visualizza il nome dello sviluppatore che ha creato la trasformazione
Creato il	Visualizza la data e l'orario in cui è stata create la trasformazione
Ultima modifica di	Visualizza il nome dell'ultimo utente che ha modificato la trasformazione
Ultima modifica il	Visualizza la data e l'orario dell'ultima modifica della trasformazione

3.1.6.2 Tab log

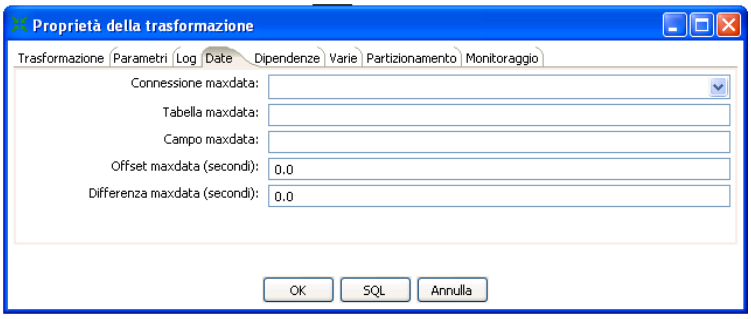


La tab di log consente di configurare come e dove vengono acquisite le informazioni di log. Le impostazioni includono:

Setting	Descrizione
Logga lettura del passo	Utilizza il numero delle righe da leggere in questo passo per scrivere nella tabella di log
Logga INPUT del passo	Utilizza il numero delle righe di input in questo passo per scrivere nella tabella di log
Logga scrittura del passo	Utilizza il numero delle righe scritte da questo passo per scrivere nella tabella di log
Logga OUTPUT del passo	Utilizza il numero delle righe di output da questo passo per scrivere nella tabella di log

Logga aggiornamento del passo	Utilizza il numero delle righe aggiornate da questo passo per scrivere nella tabella di log
Logga i respinti del passo	Utilizza il numero delle righe respinte da questo passo per scrivere nella tabella di log
Connessione del log	La connessione utilizzata per scrivere nella tabella di log
Tabella di log	Specifica il nome della tabella di log (per esempio L_ETL)
Usare Batch-ID?	Viene abilitato se si vuole avere un batch ID nel file L_ETL Viene disabilitato con *Spoon*\ Pan con una versione < 2.0

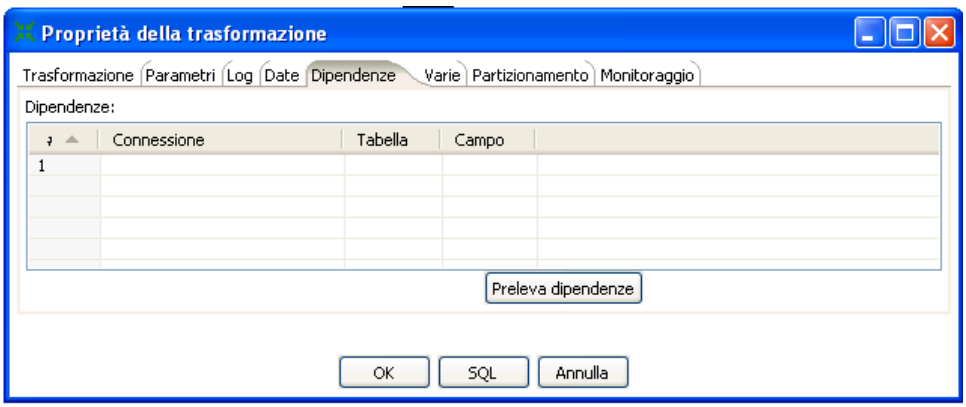
3.1.6.3 Tab *date*



Le tab *date* permette di configurare le seguenti impostazioni relative alla data:

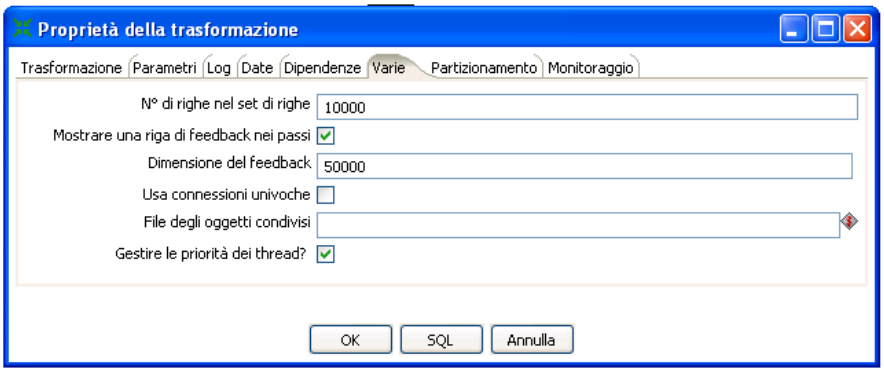
Setting	Descrizione
Connessione Maxdata	Prende il limite superiore per un intervallo di data su questa connessione
Tabella Maxdata	Prende il limite superiore per un intervallo di data su questa tabella
Campo Maxdata	Prende il limite superiore per un intervallo di data in questo campo
Offset Maxdata	Aumenta il limite superiore della data con questo valore Esempio: se trovassimo che il campo DATE_LAST_UPD ha un valore massimo di 2004-05-29 23:00:00 e sappiamo che il valore dell'ultimo momento non è corretto. In questo caso, basta mettere il valore dell'offset -60
Differenza maxdata	Imposta la differenza del Massimo data nell'intervallo dato

3.1.6.4 Tab dipendenze



La tab dipendenze consente di inserire tutte le dipendenze per una trasformazione. Ad esempio, se una dimensione dipende da tre tabelle di *lookup*, bisogna assicurarsi che le tabelle di *lookup* non sono cambiate. Se i valori di tali tabelle di *lookup* sono state modificate, bisogna estendere l'intervallo di date per forzare un aggiornamento completo della dimensione.

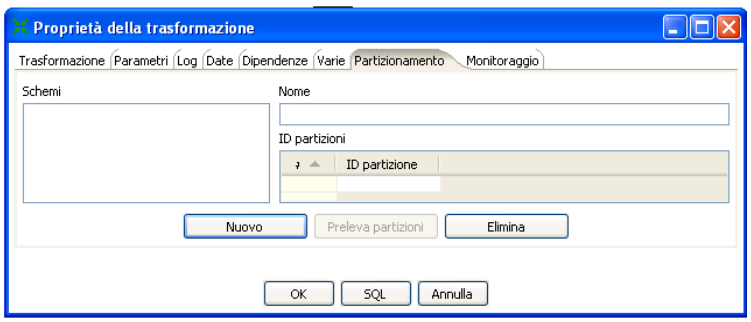
3.1.6.5 Tab varie



La tab Varie consente di configurare le seguenti impostazioni:

Setting	Description
N° di righe nel set di righe	Permette di modificare la dimensione del buffer tra i passi connessi in una trasformazione
Mostrare una riga di *feedback* nei passi	Controlla o meno di aggiungere una *feedback* nel file di log quando la trasformazione si sta eseguendo. Per default, questo campo è attivato e configurato per visualizzare una *feedback* ogni 5000 righe
Dimensione del *feedback.*	Imposta il numero di righe da processare prima di inserire un record di *feedback* nel log
Usa connessioni univoche	Permette di aprire una sola connessione per *database* definiti e usata nella trasformazione
File degli oggetti condivisi	Specifica dove si trova il file XML usato per salvare gli oggetti condivisi come le connessioni database, schemi di *clustring*, …..ecc
Gestire le priorità dei *thread*	Permette di attivare o disattivare la logica interna per cambiare la priorità dei *thread* java basato sul numero delle righe di input e degli output nel buffer di set di righe

3.1.6.6 Tab partizionamento



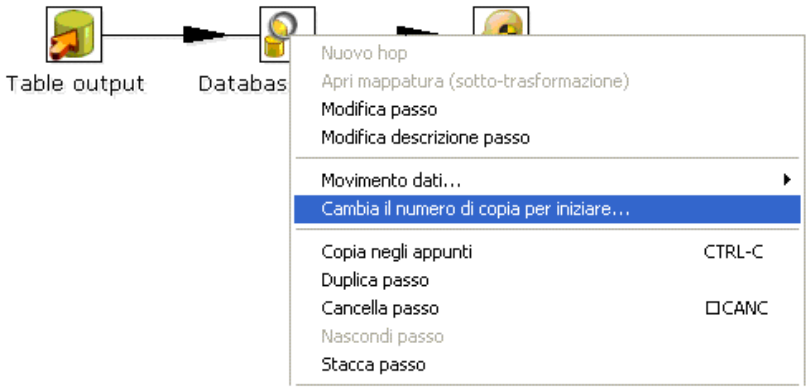
La tab partizionamento fornisce un elenco delle partizioni di database disponibili. Per creare una nuova partizione basta fare click su Nuovo. Il pulsante "preleva partizioni" recupera un elenco delle partizioni disponibili che sono stati definiti per la connessione.

3.1.7 Passi di una trasformazione

Un passo è una parte di una trasformazione. I Passi offrono una vasta gamma di funzionalità che vanno dalla lettura di file di testo fino all'implementazione di *slowly changing dimensions.*

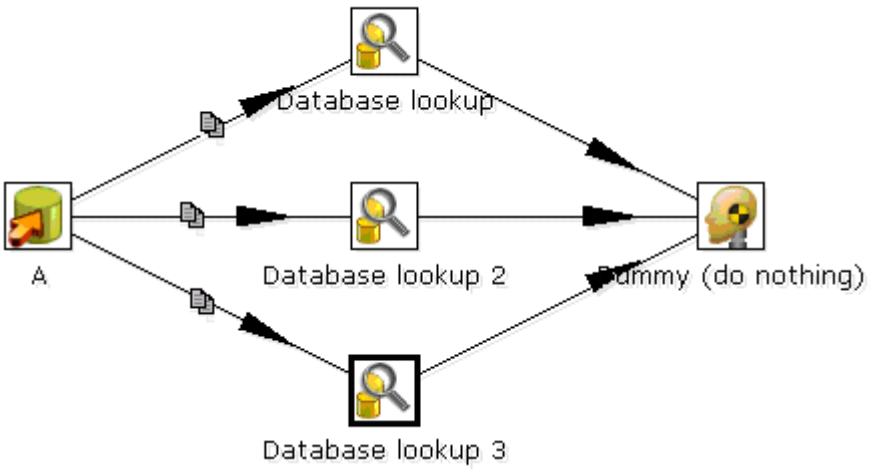
3.1.7.1 Avvio di più copie di un passo

Per motivi di prestazioni, (come ad esempio ridurre la latenza), il lancio di un passo di *lookup* tre volte o più. L'avvio dello stesso passo più volte tiene occupato il *database* su diversi collegamenti, in modo efficace riducendo la latenza. Per avviare più copie di un passo in una trasformazione, si fa *right click* su un passo nella vista grafica e si sceglie **Change number of copies to start...**:



E quindi appare una finestra che chiede di mettere il numero di volte. Se viene inserito 3 questo sarà visualizzato come segue:

Un modo equivalente di questo è fare come segue;

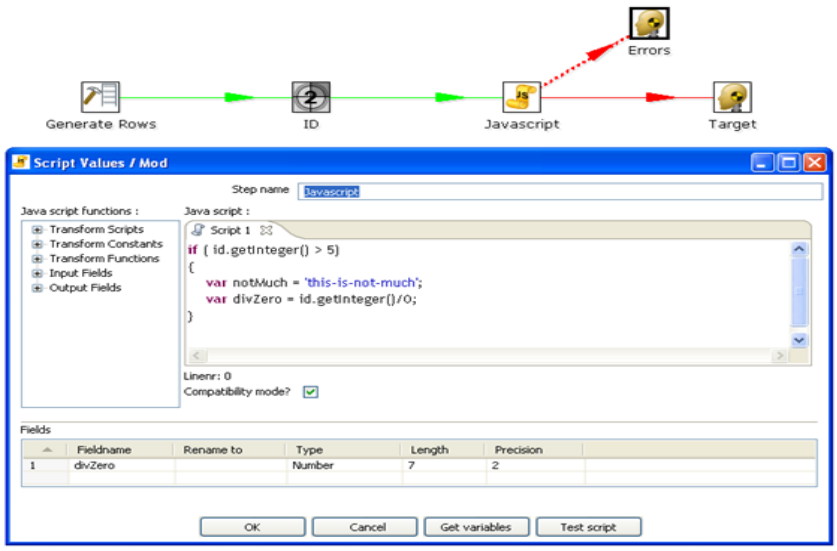


3.1.7.2 Distribuire o copia?

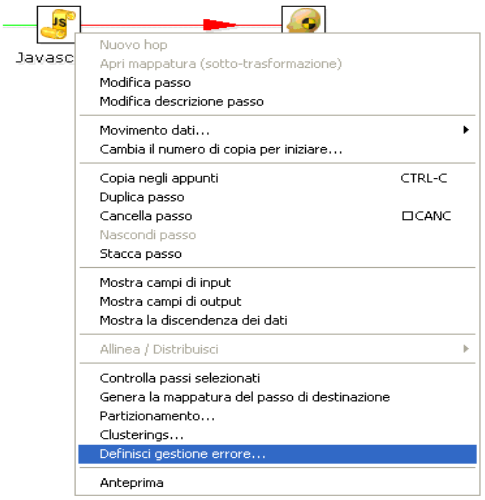
Nell'esempio precedente, ci sono linee con il simbolo che è stato visualizzato perché ci sono più uscite da un passo verso più passi *target* e quando si ha questo caso apre un *dialog box* per chiedere se distribuire o copiare i record tra i passi.
Distribuire i record indica che le righe sono distribuite tra i passi *target*. In questo modo, la prima riga va dal punto A al punto 1 del *database lookup*, la seconda al *database lookup* 2, la terza alla *databse lookup* 3, la quarta torna alla *database lookup* 1, e così via. Se viene scelto di copiare i record tra i passi, tutte le righe dal passo A vengono copiate a tutti e tre i passi target. Il passo B ottiene tre copie di tutti i file che A ha inviato.

3.1.7.3 Gestione degli errori

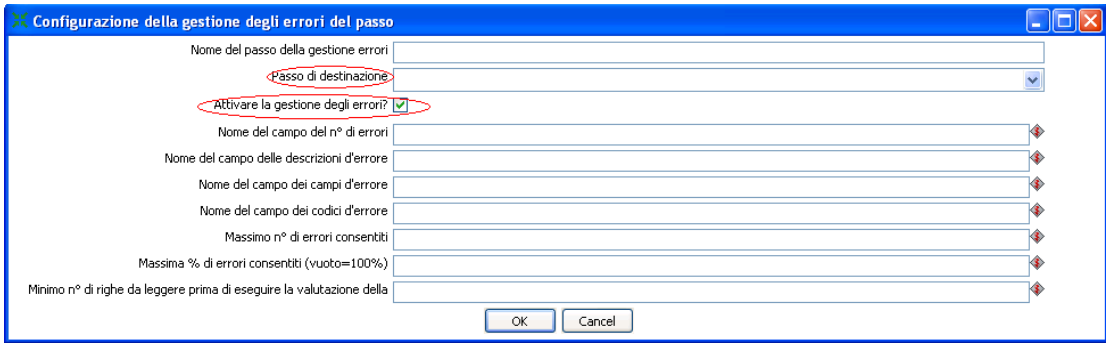
Il Passo della gestione degli errori consente di configurare un passo in modo che invece di arrestare una trasformazione quando si verifica un errore, i record che hanno causato un errore vengano passati ad un altro passo.
Nel seguente esempio, viene generato un errore nel passo *Script Values* quando un ID è superiore a cinque.



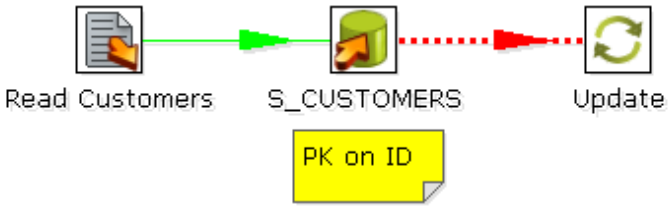
Per configurare la gestione degli errori si fa right - click sul passo e poi **Definisci gestione errore.**



Come minimo, è necessario impostare un passo obiettivo per l'errore nel flusso di dati e scegliere "attivare la gestione degli errori".



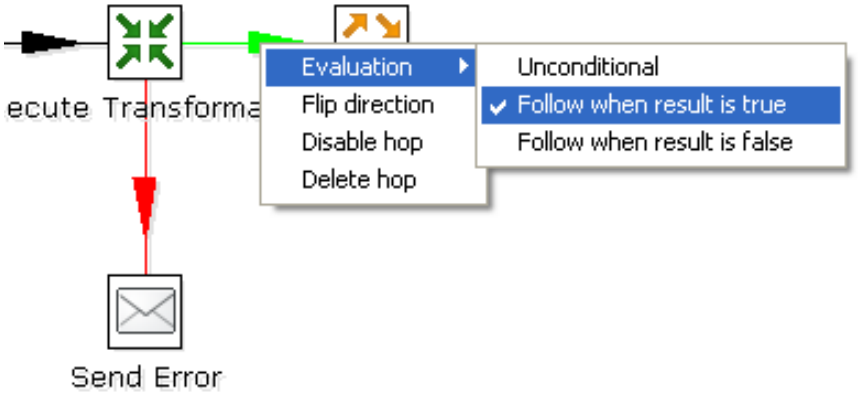
In questo modo, si può facilmente definire nuovi flussi di dati nelle trasformazioni. Il tipico caso d'uso è un modo alternativo di fare un Upsert (Inserisci / Aggiorna):



Questa trasformazione svolge un *insert* a prescindere dal contenuto della tabella. Se si mette una chiave primaria duplicata sul ID (in questo caso l'ID cliente) *l'insert* nella tabella provoca un errore. A causa della gestione degli errori si può passare la riga di errore per il passo di aggiornamento. Test preliminari hanno dimostrato che questa strategia di esecuzione di *upsert* è tre volte più veloce in alcune situazioni (con un basso rapporto *update* *insert*).

3.1.8 Hop

Un hop collega un passo di una trasformazione o di un JOB con un altro passo. La direzione del flusso di dati è indicata con una freccia sul riquadro di visualizzazione grafica. Un hop può essere attivato o disattivato (a fini di prova, per esempio).



3.1.8.1 Hop di trasformazione

Quando un hop è disattivato in una trasformazione, i passi che seguono questo hop vengono tagliati fuori da tutti i dati che scorrono a monte di esso. Questo può portare a risultati imprevisti quando si modificano le passi a valle. Ad esempio, se un particolare tipo di passo offre un pulsante "*Get row from result",* cliccando il pulsante non può rivelare nessuno dei campi in arrivo finché hop è ancora disattivata.

3.1.8.2 Hop di JOB

Oltre all'ordine d'esecuzione, un hop specifica anche la condizione in cui la *entry* successiva del JOB sarà eseguita. È possibile specificare la modalità di valutazione facendo right - click sull'hop e scegliendo:

1**."Incondizionale"** specifica che la prossima *entry* del JOB sarà eseguita, indipendentemente dal risultato del *JOB entry* originario
2**."Seguire quando il risultato è vero"** specifica che il prossimo *JOB entry* non sarà eseguita quando il risultato del *JOB entry* originario non è vero, vale a dire con successo
3**." Seguire quando il risultato è falso"** specifica che il prossimo *JOB entry* sarà eseguita solo quando il risultato del *JOB entry* originario è falso, cioè nel caso di una esecuzione infrattuosa esempio: il file non trovato, tabella non trovato, si è verificato un errore, la valutazione è falsa , ...

3.1.8.3 Creazione di un hop

Per creare un nuovo hop tra due passaggi si utilizza una delle seguenti opzioni:
* Si trascina la vista grafica tra due passi, tenendo premuto il tasto centrale del *mouse*
* Si trascina la vista grafica tra due passi, premendo il tasto <Maiusc> e utilizzando il tasto sinistro del *mouse*
* Utilizzando <Ctrl> + click-sinistro per selezionare due passi, e poi si fa right click sul passo e si sceglie **Nuovo Hop**

3.1.8.4 Dividere un hop

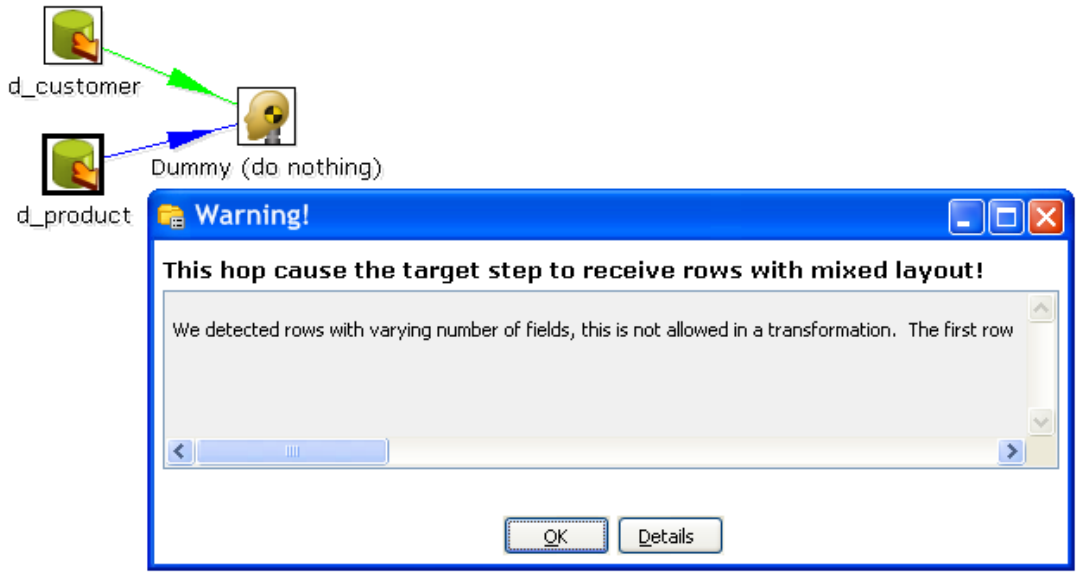
Per dividere un Hop si inserisce un nuovo passo in un nuovo hop tra due passi trascinando il passo (in Vista grafica) su un hop quindi verrà chiesto se si desidera suddividere il hop e quindi si sceglie "si" per dividere l'hop.

3.1.8.5 Cicli

I Cicli non sono ammessi nelle trasformazioni perché *KETTLE* dipende dai passi precedenti per determinare il valore dei campi che sono stati passati da un passo all'altro. Consentire i *loop* nelle trasformazioni può risultare in un ciclo infinito e altri problemi.
I *loop* sono inoltre ammessi nei JOB perché *Spoon* esegue i passi di JOB in sequenza. Però bisogna assicurarsi di non costruire un ciclo senza fine.

3.1.8.6 Rilevazione dei flussi con campi diversi

Combinare le righe con un *layout* diverso non è consentito in una trasformazione. Facendolo causa il fallimento del passo perché i campi non possono essere trovati. Nel caso in cui venga fatta una miscelazione dei *layout* con numero o nomi di campi diversi allora vengono forniti avvisi:



In questo caso, il messaggio d'errore dice:
Abbiamo rilevato delle righe con numero variabile di campi, questo non è consentito in una trasformazione. La prima riga contiene 13 campi, l'altra 16:

[ENG:customer_tk=0, version=0, date_from=, date_to=, CUSTOMERNR=0, NAME=, FIRSTNAME=, LANGUAGE=, GENDER=, STREET=, HOUSNR=, BUSNR=, ZIPCODE=, LOCATION=, COUNTRY=, DATE_OF_BIRTH=]

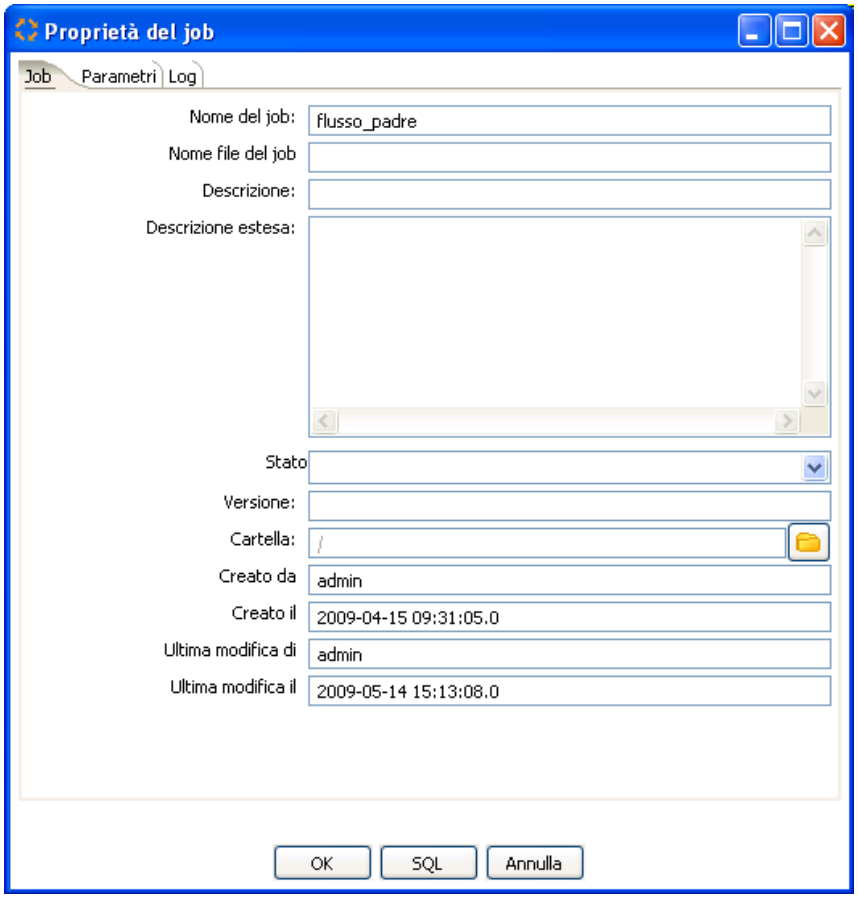
3.1.8.7 Colori di hop

Nelle trasformazioni/ JOB vengono visualizzati in vari colori basati sulla proprietà e lo stato dell'hop. La tabella seguente descrive il significato del colore di un hop in una trasformazione/JOB:

colore dell'hop	significato
Verde	i record andranno nel passo successivo solo se soddisfano la condizione
Rosso	i record andranno nel passo successivo solo se non soddisfano la condizione
Grigio	il hop è disabilitato
Nero	la hop è incondizionale

3.1.9 Impostazioni di un JOB

Le impostazione di un JOB sono delle opzioni che determinano il comportamento di un JOB e come deve archiviare ciò che fa. Per scegliere le impostazioni di un JOB, si seleziona dal menu principale **JOB---Parametri** e quindi viene visualizzata la schermata che segue:

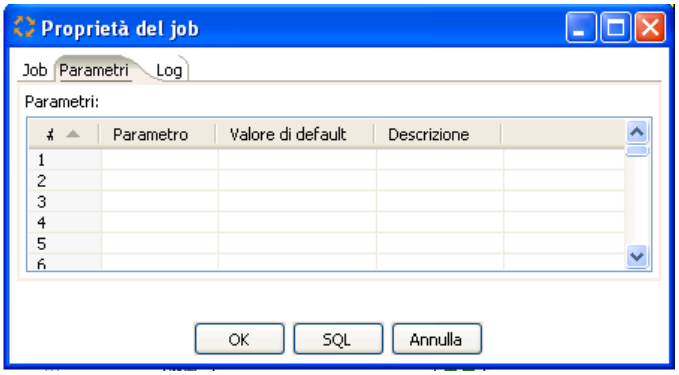


3.1.9.1 Tab JOB

La tabella seguente descrive tutte le impostazioni che si trovano nella tab JOB:

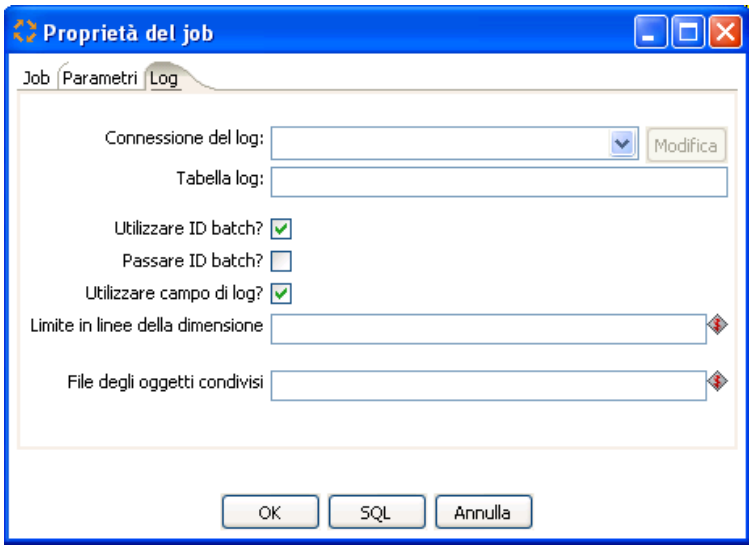
Opzione	Descrizione
Nome del JOB	Il Nome del JOB
Descrizione	La descrizione del JOB che viene visualizzata quando viene esplorato il *repository*
Descrizione estesa	La descrizione estesa del JOB
Stato	Lo stato del JOB
Versione	Descrizione della versione
cartella	La cartella dov'è salvato il JOB
Creato da	Visualizza il nome dello sviluppatore che ha creato il JOB
Creato il	Visualizza la data della creazione del JOB
Ultima modifica di	Visualizza il nome dell'ultimo sviluppatore che ha modificato il JOB
Ultima modifica il	Visualizza la data dell'ultima modifica

3.1.9.2 Tab parametri



Questa tab contiene la lista dei parametri che devono essere passati al JOB nel caso volessimo creare dei JOB parametrizzati che vengono usate più volte con parametri diversi.
Parametro: è il nome del parametro.
Valore di *default*: il valore di *default* del parametro nel caso non viene passato al JOB.
Descrizione: descrizione del parametro e del suo uso.

3.1.9.3 Tab Log



La tabella seguente descrive tutte le caratteristiche generali che vengono visualizzate nella tab LOG:

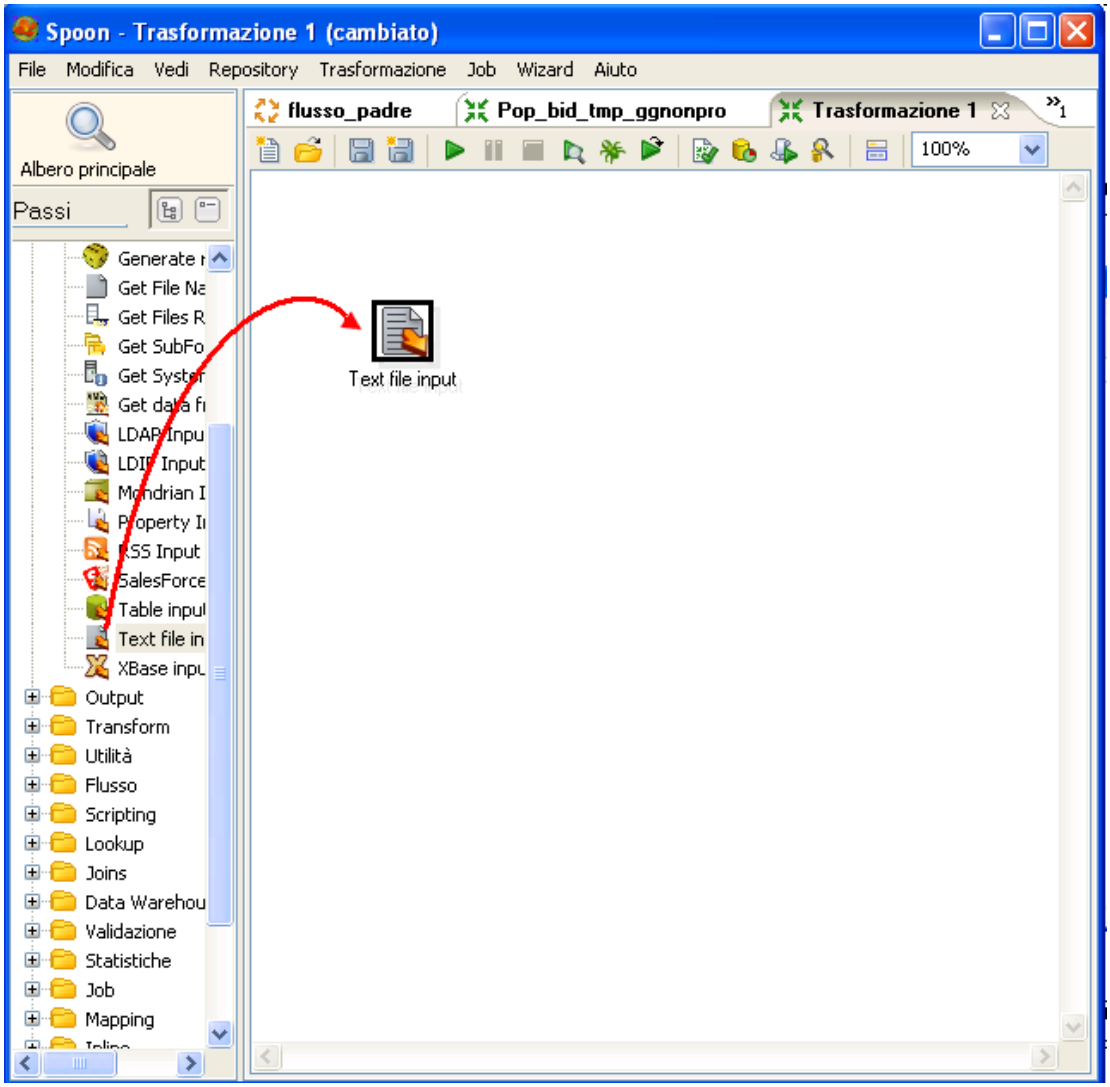
Opzione	Descrizione
Connessione del log	Utilizza questa connessione per scrivere su un file di log
Tabella log	Specifica il nome della tabella di log (per esempio L_ETL)
Utilizzare ID batch?	Utilizza un batch ID nella tabella di log
Passare ID batch?	Serve a passare l'ID batch generato a una trasformazione nel JOB oppure no
Utilizare campo di log?	Serve a scrivere il risultato di log nella tabella di log come un campo di testo
Pulsante SQL	Generare la query SQL che serve a creare la tabella di log

3.1.10 Vista grafica

La vista grafica contiene la tela su cui le trasformazioni e i JOB sono disegnati. Ci sarà una scheda separata per ogni JOB e / o trasformazione attualmente aperta con un'icona che indica il tipo di file. I tab della vista grafica forniscono una rappresentazione più facile del lavoro che dovrebbe essere fatto con il flusso di dati.

3.1.10.1 Aggiungere un passo

Per aggiungere un passo in una trasformazione (o in un JOB) basta selezionare un passo dall'albero che si trova a sinistra e trascinarlo nella tela come è descritto dalla figura seguente:



Al cursore del mouse verrà visualizzato un quadrato che rappresenta la posizione del passo quando il pulsante verrà lasciato e quindi il passo diventa una parte della trasformazione. Inoltre è possibile aggiungere un passo per una trasformazione facendo right - click dentro la tela e selezionando *Nuovo passo* e poi scegliere il tipo del passo.

Nascondere un passo: per nascondere un passo che si trova nella tela di una trasformazione, basta fare right - click sul passo e selezionare NASCONDI PASSO e quindi il passo verrà nascosto ma non eliminato.

3.1.10.2 Opzioni di un passo di una trasformazione

Questa sezione descrive il menu che viene visualizzato quando si fa un right - click sul passo in una trasformazione:
Modifica passo: questa opzione permette di modificare le impostazioni di un passo.
Modifica descrizione passo: permette di cambiare la descrizione testuale di un passo.
Movimento dati : permette di scegliere tra copiare i dati del passo nei passi successivi oppure distribuirli tra di essi.

Cambia il numero di copie per iniziare: permette l'avvio di più copie di un passo per velocizzare la trasformazione.
Copia negli appunti : Questa opzione consente di copiare il codice XML che definisce il passo al *clipboard*. Quindi è possibile incollare il passo in un altra trasformazione.

Duplica Passo: permette di creare una copia del passo che viene posizionata un po' a destra del passo.
Cancella passo: permette di rimuovere definitivamente il passo dalla trasformazione.
Nascondi Passo: nasconde il passo nella Vista grafica, ma non lo rimuove dalla trasformazione.
Mostra campi di input: Questa opzione determina tutti i campi e la loro origine rintracciando all'indietro il flusso in ingresso verso la sua fonte.
Mostra campi di output: Questa opzione aggiunge i campi del passo corrente ai campi di input e mostra il risultato.

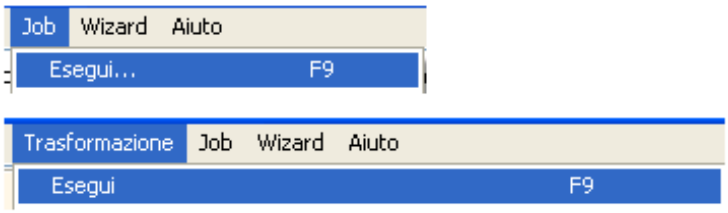
3.1.10.3 Opzioni di un *JOB entry*

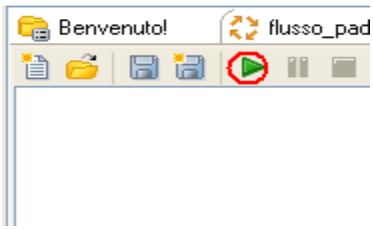
Apri Trasformazione / JOB: apre una nuova scheda visualizzando la trasformazione o il JOB selezionato.
Modifica *JOB entry*: consente di cambiare le impostazioni di un *JOB entry* .
Modifica la descrizione della *JOB entry*: apre una finestra di dialogo che permette di inserire una descrizione testuale della *JOB entry*.
Crea una copia shadow di questa *JOB entry*: Questa opzione crea una copia, in posizione un po' più basso a destra della voce originale della *JOB entry*.
Copia la *JOB entry* negli appunti: Copia il codice XML che descrive la *JOB entry* negli appunti.
Allineare / distribuisci : Questa opzione consente di mantenere pulito il grafico allineando le *JOB entry* gli uni con gli altri.
Stacca entry: stacca la *JOB entry* dal Hop che la collega ad un 'altro passo.
Cancella tutte le copie di questa entry:Elimina tutte le copie di questo *JOB entry*, non solo questa.

Per aggiungere un Hop, il modo più facile e veloce di farlo è quello di trascinare con il *mouse* da un passo all'altro utilizzando il pulsante centrale del *mouse*.

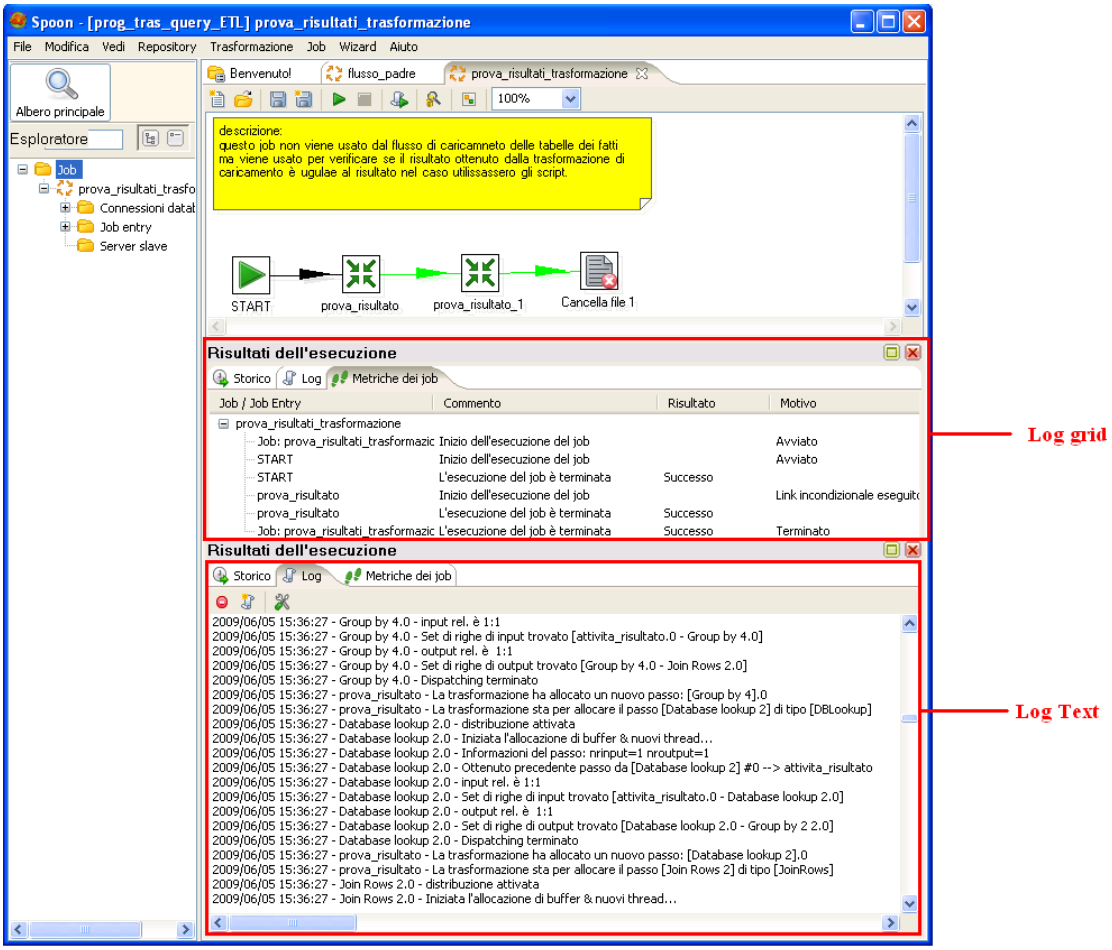
3.1.11 Esecuzione di una trasformazione/ JOB

Quando si finisce di designare una trasformazione/ JOB, è possibile eseguirla cliccando su **trasformazione/JOB** e poi **Esegui** dal menu principale della barra degli strumenti , premendo F9 oppure premendo il pulsante **start**.





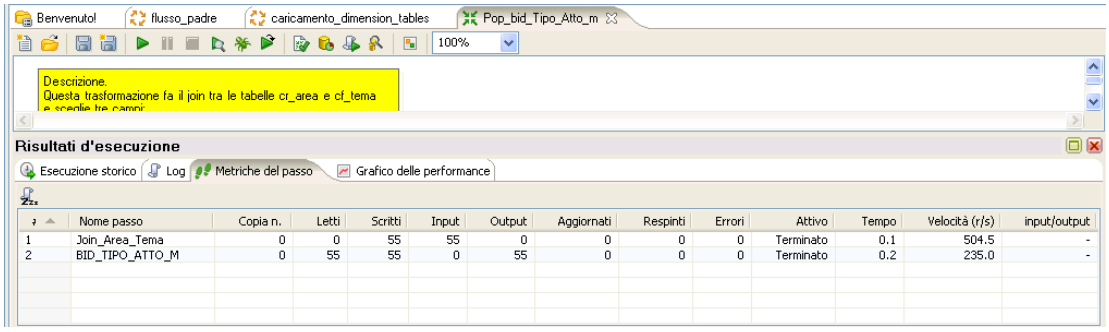
Logging



Una scheda di log viene visualizzata automaticamente ogni volta che viene eseguita una trasformazione o un JOB. La *log grid* mostra una serie di passi di una trasformazione o di un *JOB entry* dell'esecuzione in corso. Il *log text* mostra informazione di log in base al livello di esecuzione.

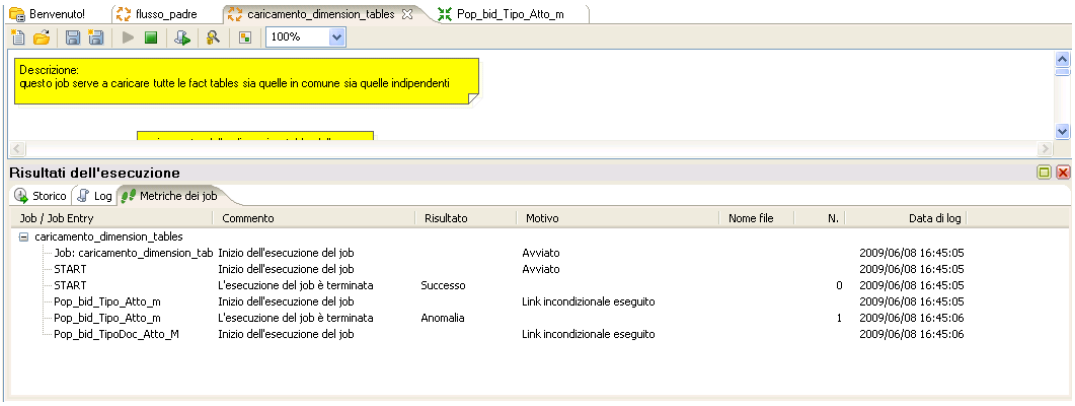
Log Grid: Un *log grid* è un albero che offre una vista gerarchica dell'esecuzione di una trasformazione o di un JOB. I passi di una Trasformazione o le *JOB entry* saranno evidenziate in rosso, se falliscono durante l'esecuzione a causa di un errore.

Dettagli di una log Grid di una trasformazione: la *log grid* mostra i seguenti dettagli per ogni passo di una trasformazione in esecuzione:



Opzione	Descrizione
Nome passo	Il nome del passo
Copia n	Il numero delle copie del passo
letti	Il numero delle righe letti dal flusso in ingresso
scritti	Il numero delle righe scritti nel flusso in uscita
Input	Il numero delle righe letti da un file o un *database*
Output	Il numero delle righe scritti in un file o in un *database*
aggiornati	Il numero delle righe aggiornati in un *database*
Respinti	Numero degli scarti
Errori	Numero degli errori
Attivo	Lo stato del passo: in esecuzione, terminato, fermato
Tempo	Il numero di secondi durante i quali il passo era in esecuzione
velocità	La velocità in righe con cui il passo ha processato i record

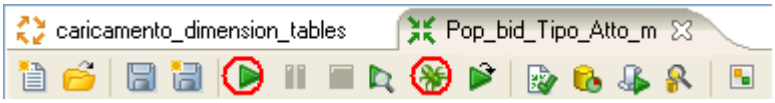
Dettagli di un log Grid di un *JOB Entry*:la *log grid* di un *JOB entry* mostra i seguenti dettagli:



Opzione	Descrizione
JOB/*JOB Entry*	Il nome del JOB\ *JOB entry*
Commento	Commento sullo stato dell'esecuzione del *JOB entry*
Risultato	Il risultato dell'esecuzione
Motivo	Il motivo per cui la *JOB entry* è iniziata
Data di log	La data di inizio della *JOB entry*

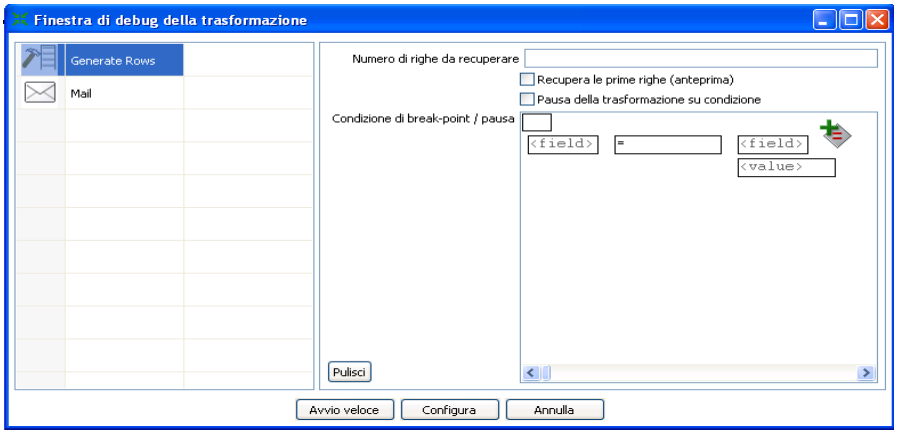
3.1.12 Pulsanti

3.1.12.1 Pulsanti di una trasformazione



Start: questo pulsante lancia la trasformazione ed è necessario che la trasformazione sia salvata prima di essere lanciata ed il risultato è visualizzato nella parte del *log Text*.

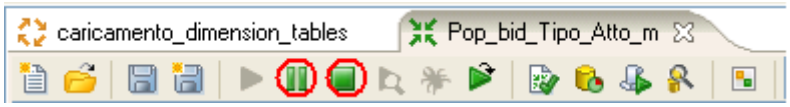
Anteprima (debug):



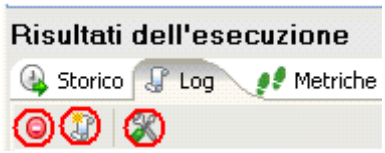
Questo pulsante avvia la finestra di dialogo *debug* della trasformazione che consente di specificare il numero di righe da visualizzare in anteprima e definire i *breakpoint* condizionali per l'esecuzione in anteprima.
Dopo aver configurato la informazioni di *debug*, si clicca sul pulsante 'avvio veloce' per avviare l'esecuzione in anteprima del passo selezionato. L'output di esecuzione viene visualizzato nella parte di *log Text*.

Opzioni di Debug: La seguente tabella fornisce una descrizione dettagliata delle opzioni di *debug*:

Opzione	Descrizione
Lista dei passi	La lista dei passi a sinistra mostra una lista di passi disponibili per la trasformazione corrente. Si seleziona un passo e poi viene configurato come ad esempio il numero di righe e *i break - points*
Numero di righe da recuperare	Si inserisce il numero di righe da visualizzare in anteprima per il passo selezionato. Dopo che le righe richieste vengano ottenuti dai diversi passi, la trasformazione finisce e il risultato viene visualizzato
Recupera le prime righe (anteprima)	Bisogna selezionarlo nel caso si ha bisogno di restringere il numero di righe da visualizzare al numero scelto sopra
Pausa della trasformazione su condizione	Si attiva per fermare la trasformazione se si verifica la condizione di un *Break-point* durante l'esecuzione
Condizione di *Break-Point/ Pausa*	La condizione che deve fermare la trasformazione nel caso in cui si verifica



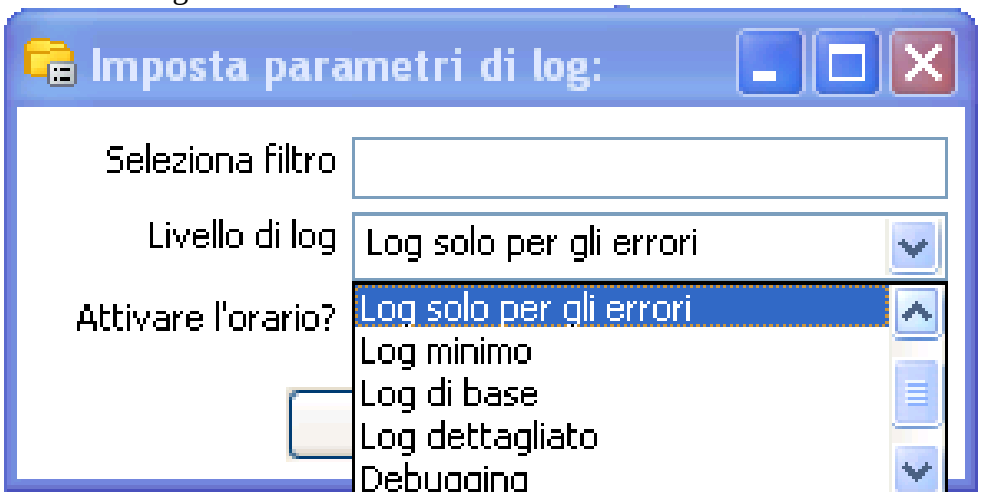
Pausa: Permette di sospendere temporaneamente l'esecuzione della trasformazione.
Stop: Permette di fermare definitivamente l'esecuzione della trasformazione.



Mostra linee d'errore: Questo pulsante visualizza tutte le linee di registro testo che contengono la parola ERRORE (maiuscole o minuscole). È quindi possibile scegliere di modificare il passo della fonte di errore.

Pulisci log: Questo pulsante cancella il testo della log text.

Parametri log: Se si inserisce un testo nel campo di filtro, solo le righe che contengono questo testo verranno visualizzati nella finestra di *log text*. L'opzione "livello di log" consente di selezionare il livello di registrazione.



È possibile scegliere uno di questi:

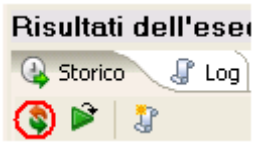
* Niente: Non mostra alcun output
* Log solo per gli errori: Mostra solo gli errori
* Log minimo: Solo l'uso minimo di registrazione
* Log di base: Questo è il livello di registrazione di base di default
* Log dettagliato: Lascia l'output della registrazione dettagliata
* Debugging: Per scopi di *debug*, molto dettagliato di uscita.
* Basso livello (molto dettagliato): Accesso a livello di riga, questo è in grado di generare una grande quantità di dati.

Se la voce "Attivare l'orario?" è attivata, tutte le linee della registrazione sarà preceduta dal momento della giornata.

3.1.12.2 Pulsanti di un JOB

Inizio JOB: Questo pulsante inizia l'esecuzione del JOB attuale. Si noti che *Spoon* lancia i tentativi di avviare il JOB da un file XML o dal *repository* di *KETTLE*. È pertanto necessario che il JOB viene salvato. L'output dell'esecuzione viene visualizzato nel *log text* nella vista di log..

Stop JOB : Questo pulsante interrompe l'esecuzione di un JOB.



Aggiorna: Aggiorna la finestra del registro di log.

Questi tre pulsanti sono uguali a quelli delle trasformazioni che sono stati discussi sopra.

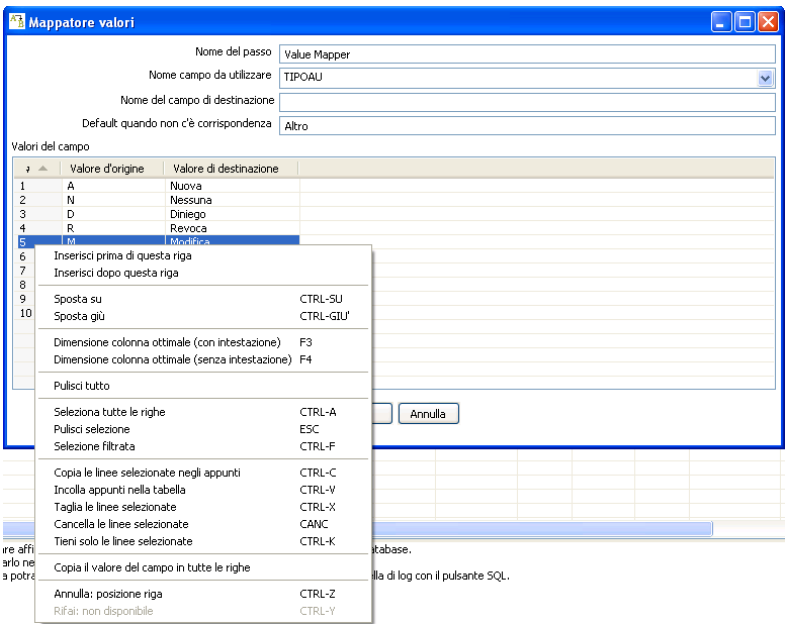
3.1.13 Grid

Descrizione

I *Grid* (tabelle) sono utilizzati nelle interfacce di *Kettle* per entrare o di visualizzare delle informazioni. Questa sezione descrive le funzioni comuni disponibili quando si lavora con una *grid*.

Utilizzo

si fa un right - click su un campo per iniziare a modificare le sue impostazioni e quindi viene visualizzata la schermata seguente:



La seguente tabella descrive le funzioni disponibili quando si fa right - click su una cella nella *grid*:

Opzione	Descrizione
Inserisci prima di questa riga	Inserisce una riga vuota prima della riga che è stata cliccata
Inserisci dopo questa riga	Inserisce una riga vuota dopo la riga che è stata cliccata
Sposta su	Sposta la riga cliccata verso l'alto
Sposta giù	Sposta la riga cliccata verso il basso
Dimensione colonna ottimale (con intestazione)	Ridimensiona tutte le colonne, quindi visualizza totalmente tutti i valori, l'*header* incluso
Dimensione colonna ottimale (senza intestazione)	Ridimensiona tutte le colonne, quindi visualizza totalmente tutti i valori, l'*header* escluso
Pulisci tutto	Cancella tutte le informazioni nella *grid*. Viene visualizzato una schermata di conferma
Selezione tutte le righe	Selezione tutte le righe nella *grid*
Pulisci selezione	Cancella le righe selezionate
Copia le righe selezionate negli appunti.	Copia le righe selezionate negli appunti con una rappresentazione testuale
Incolla appunti nella tabella	Incolla le righe che ci sono negli appunti nella *grid*
Taglia le linee selezionate	Taglia le righe selezionate dalla *grid*
Cancella le linee selezionate	Cancella tutte le righe selezionate dalla *grid*
Tieni solo le linee selezionate	Lascia nella *grid* solo le linee selezionate
Copia il valore del campo in tutte le righe	Se tutte le righe nella *grid* devono avere lo stesso valore allora si può selezionare questa opzione per fare ciò
Annulla	Annulla l'ultima operazione fatta sulla *grid*
Rifai	Rifai l'ultima operazione fatta sulla *grid*

3.2 BART- Lo strumento di *Business Intelligence*

In questo paragrafo verrà spiegato lo strumento di *Business intelligence* BART, che è stato utilizzato durante lo stage con particolare riferimento alle sezioni della creazione del cubo OLAP, della sua interrogazione, e la visualizzazione del risultato via Web.

3.2.1 Introduzione

BART è l'acronimo di *Business Application and Reportistic Tool*, ed è un'applicazione Web di *business intelligence* che è stata sviluppata nell'azienda Quix , che offre un metodo per la creazione del cubo OLAP, l'interrogazione di esso e la visualizzazione del risultato (*Report*) usando solo il browser senza alcuna necessità di installare altre applicazioni (mentre i componenti di BART vengono installati su uno o più server).

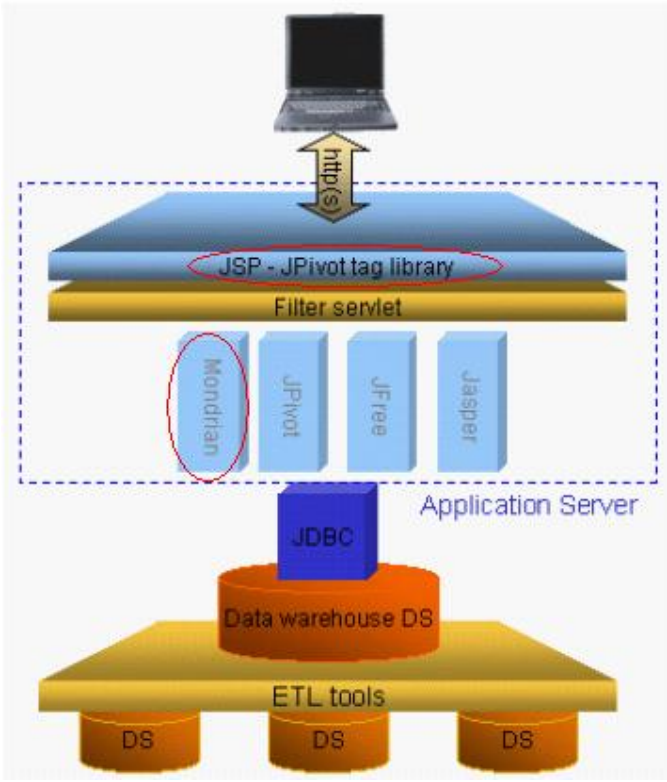
BART utilizza il server OLAP *Mondrian* come motore per risolvere le interrogazioni multidimensionali, MDX che permette di interrogare i dati multidimensionali e JPIVOT come *front-end* per visualizzare e navigare i risultati delle interrogazioni, il *framework* Jakarta Struts per lo sviluppo di un'applicazione Web conforme al pattern MVC e Java come linguaggio di programmazione.

Di seguito verrà spiegato come vengono utilizzati le parti di BART che corrispondono alle operazioni necessarie per lo sviluppo di un *Report* e che sono anche le parti che sono state utilizzate durante lo stage, cioè la parte della creazione del cubo OLAP (che utilizza il server *Mondrian* di *pentaho*), MDX, e JPIVOT.

3.2.2 Mondrian - JPIVOT

Mondrian è un server OLAP che consente di gestire modelli multidimensionali interfacciandosi a *database* relazionali; la connessione alla base di dati di *data warehouse* avviene via JDBC, il che rende indipendente *Mondrian* dal particolare RDBMS utilizzato.
Lo schema multidimensionale della base dati può essere sia *star* che *snowflake*, e la sua descrizione viene fornita al motore sotto forma di file XML.
Mondrian è progettato per delegare all'RDBMS tutte le funzionalità che questo è in grado di eseguire al meglio, in particolare l'aggregazione e l'utilizzo, ove consentito, di viste materializzate per ottimizzare la velocità di risposta.
Le raffinate strategie di caching consentono buone prestazioni in termini di velocità di esecuzione.

JPIVOT è una libreria di tag JSP personalizzati che leggono una tabella OLAP e permettono agli utenti di effettuare tipiche operazioni di navigazioni degli strumenti OLAP come SLICE, DICE, DRILL- DOWN e ROLL-UP.
JPIVOT consente all'utente la navigazione all'interno di una tabella OLAP elaborando dinamicamente query MDX. Utilizza *Mondrian* come motore OLAP, ma può interagire anche con sorgenti dati XMLA (XML for Analisys). JPIVOT si basa sulla libreria WCF (*Web Component Framework*) per il *rendering* degli oggetti grafici dell'interfaccia utente e sul noto pacchetto *JFreeChart* per il tracciamento di grafici.



Caratteristiche dell'architettura:

- Facilità di navigazione
- Visualizzazione grafici sincronizzata con il percorso di navigazione
- Estensibilità interfaccia utente
- Semplicità di integrazione
- Elevata scalabilità: le prestazioni dipendono dall'RDBMS e *dall'Application Server* adottati
- Flessibilità di installazione: RDBMS, motore OLAP e strato di presentazione possono anche essere installati su macchine diverse
- Indipendenza dal sistema operativo
- Indipendenza dal client (*web browser*) utilizzato

La configurazione dell'architettura si sviluppa principalmente nei seguenti passi:

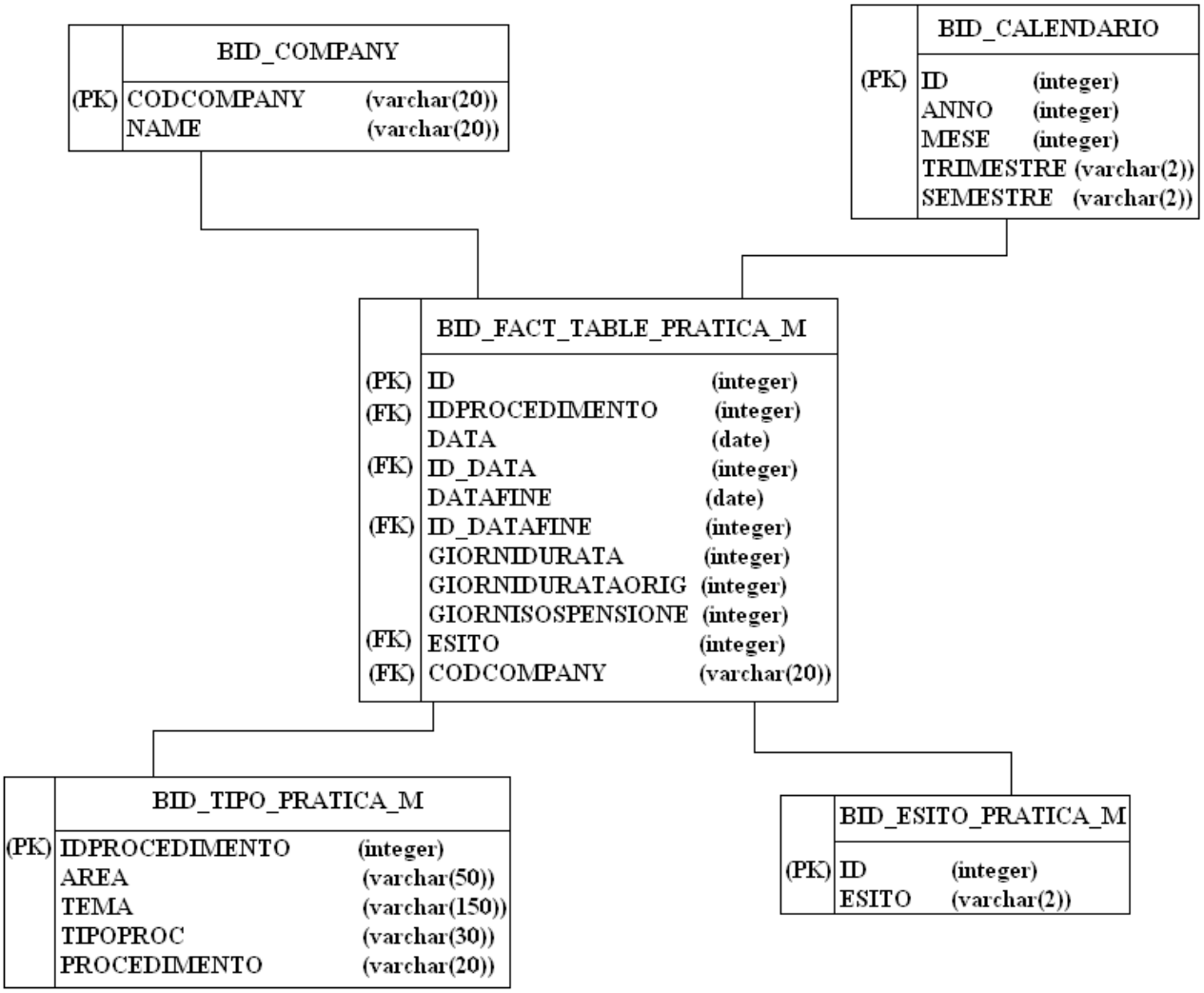
- Progettazione e creazione del *database*
- Editing del file XML per la definizione dello schema del *data warehouse*
- Definizione delle query predefinite (istruzioni MDX)
- Testare la connessione al *database* ed esecuzione delle query

Mondrian-JPIVOT rappresenta un vero e proprio strumento analitico che oltre ad offrire funzionalità di *reporting* implementa importanti risorse di analisi a livello *OLAP*.

E' fondamentale sapere che *Mondrian* utilizza un file XML per descrivere la struttura del *data warehouse* su cui operare: attraverso questo file è possibile identificare **Cubi**, **Dimensioni**, **Misure**, **Gerarchie**, **Livelli** ecc. Ciò permette di esprimere dei riferimenti da utilizzare nella scrittura delle istruzioni MDX e consente a JPIVOT di poter realizzare la navigazione all'interno della tabella OLAP (in pratica lo stato della tabella OLAP non è altro che la rappresentazione grafica del risultato di una query MDX); il modello XML esprime lo schema del *database* multi-dimensionale.

L'editing del file XML è una delle operazioni più delicate da affrontare. Il file deve essere scritto in maniera coerente e sintatticamente corretta. Naturalmente l'elaborazione del file deve essere un'operazione graduale: inizialmente si definiscono semplici relazioni, successivamente si arricchisce la struttura dello schema in modo da raggiungere livelli di complessità sempre più elevata.

La fase di scrittura dello schema XML per la descrizione del modello del *data warehouse* rappresenta un passo cruciale da affrontare per rendere operativa l'architettura *Mondrian-JPIVOT;* la definizione dello schema deve essere fatta con molta cura in modo da rispettare le esigenze di analisi previste ed in modo da rispettare la conformità con il modello relazionale precedentemente pianificato. Si noti che il file XML deve essere scritto dal programmatore.
Uno schema definisce un *database* multidimensionale; contiene il modello logico (costituito da cubi, gerarchie e membri) e le relazioni con il modello fisico.
Di seguito viene riportato un esempio, partendo dallo schema a stella, viene scritto il file XML di *Mondrian*.



Questa schema a stella è composto da una tabelle di fatti BID_FACT_TABLE_PRATICA_M e quattro tabelle dimensionali che sono: BID_CALENDARIO, BID_COMPANY, BID_TIPO_PRATICA_M, BID_ESITO_PRATICA_M.

Vicino a ogni attributo di ciascuna tabella è riportato il tipo di dato di quell'attributo e le chiavi primarie sono indicate con la presenza del simbolo (PK) a sinistra, mentre le *FOREIGN KEY* sono indicate con il simbolo (FK) che sono quattro ed ognuna di loro fa riferimento alla chiave primaria di una tabelle dimensionale.

Questo schema a stella dovrebbe essere mappato in un file XML, usando dei tag XML specifici, in modo che *Mondrian* riesca a capirlo per interrogarlo e eseguire le operazioni fondamentali per la visualizzazione multidimensionali (ROLL-UP, DRILL-DOWN, ecc ……) quindi viene mappato in xml come segue:

```
<?xml version="1.0" encoding="UTF-8"?>
<Schema name="Pratica">
  <Cube name="Pratica">
    <Table name="bid_fact_table_pratica_m"/>
    <Dimension foreignKey="codcompany" name="Company">
      <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="codcompany">
        <Table name="bid_company"/>
        <Level column="name" name="COMPANY" type="String" uniqueMembers="true"/>
      </Hierarchy>
    </Dimension>
    <Dimension foreignKey="idprocedimento" name="Tipo">
      <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="idprocedimento">
        <Table name="bid_tipo_pratica_m"/>
        <Level column="area" name="AREA" type="String" uniqueMembers="true"/>
        <Level column="tema" name="TEMA" type="String" uniqueMembers="false"/>
        <Level column="tipoproc" name="TIPOPROC" type="String" uniqueMembers="false"/>
        <Level column="procedimento" name="PROCEDIMENTO" type="String" uniqueMembers="false"/>
      </Hierarchy>
    </Dimension>
    <Dimension foreignKey="id_data" name="Tempo inizio">
      <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="id">
        <Table name="bid_calendario"/>
        <Level column="anno" name="ANNO_INIZIO" type="Integer" uniqueMembers="true"/>
        <Level column="semestre" name="SEMESTRE_INIZIO" type="String" uniqueMembers="false"/>
        <Level column="trimestre" name="TRIMESTRE_INIZIO" type="String" uniqueMembers="false"/>
        <Level column="mese" name="MESE_INIZIO" type="Integer" uniqueMembers="false"/>
      </Hierarchy>
    </Dimension>
    <Dimension foreignKey="id_datafine" name="Tempo fine">
      <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="id">
        <Table name="bid_calendario"/>
        <Level column="anno" name="ANNO_FINE" ordinalColumn="" type="Integer" uniqueMembers="true"/>
        <Level column="semestre" name="SEMESTRE_FINE" type="String" uniqueMembers="false"/>
        <Level column="trimestre" name="TRIMESTRE_FINE" type="String" uniqueMembers="false"/>
        <Level column="mese" name="MESE_FINE" type="Integer" uniqueMembers="false"/>
      </Hierarchy>
    </Dimension>
    <Dimension foreignKey="esito" name="Esito">
      <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="id">
        <Table name="bid_esito_pratica_m"/>
        <Level column="esito" name="ESITO" uniqueMembers="true"/>
      </Hierarchy>
    </Dimension>
    <Measure aggregator="count" column="id" formatString="Standard" name="N_PRATICHE"/>
    <Measure aggregator="avg" column="giornidurata" formatString="#,###.00" name="DURATA_MEDIA (gg)"/>
    <Measure aggregator="max" column="giornidurata" formatString="Standard" name="DURATA_MAX (gg)"/>
  </Cube>
</Schema>
```

Come osservabile dallo schema riportato, il modello XML si basa sui seguenti tag:

- **Cube** = collezione di misure e dimensioni
- **Table** = definisce le relazioni con le tabelle del modello fisico
- **Dimension** = specifica ogni dimensione di riferimento, le dimensioni possono essere associate ad un unico cubo oppure condivise (utilizzate da più cubi)
- **Hierarchy** = definisce le gerarchie legate ad una dimensione
- **Level** = definisce un livello per una data gerarchia
- **Measures** = esprime le misure di interesse; ogni misura ha un nome che identifica la colonna nella *fact table* ed un aggregatore.

L'aggregatore può essere:

- *Sum* (somma)
- *Avg* (media)
- *Max* (massimo)
- *Min* (minimo)
- *Count* (conteggio)
- *Distinct Count* (conteggio distinto)

La strategia di aggregazione usata da *Mondrian* è la seguente:

• La *fact table* è memorizzata nel DBMS

• Per aggregare i dati viene utilizzata la clausola *group by* di SQL

• Se l'RDMBS supporta viste materializzate e l'amministratore del *database* sceglie di creare viste materializzate per particolari aggregazioni, allora *Mondrian* usa esse per semplicità.

Partendo dallo schema multidimensionale possiamo sviluppare le seguenti definizioni:

- Definiamo membro un punto in cui le dimensioni sono determinate da un preciso insieme di valori degli attributi
- Una gerarchia è un insieme di membri organizzati in una struttura per convenienza di analisi
- Un livello è una collezione di membri che hanno la stessa distanza dalla radice della gerarchia
- Una dimensione è una collezione di gerarchie che vengono discriminate dallo stesso attributo della *fact table.*

Come è possibile osservare, per ragioni di uniformità, le *misure* sono trattate come un membro di una speciale dimensione chiamata "Measures"; l'attributo opzionale *formatString* specifica come il valore dell'attributo deve essere visualizzato.

Di seguito viene fornita una spiegazione dettagliata dello schema xml scritto:

Il tag <schema> che si vede nella seconda riga serve a definire un *database* multidimensionale che contiene un modello logico, costituito da cubi, gerarchie, e dei loro membri, e di una mappatura di questo modello su un modello fisico che è la fonte dei dati che viene presentata attraverso il modello logico. Nel nostro caso lo schema a stella si chiama PRATICA ed è costituito da un cubo PRATICA come viene indicato nella tag < Cube> della terza riga.

La tag <Table> nella quarta riga indica che il cubo **Pratica** viene mappato alla tabella dei fatti di nome BID_FACT_TABLE_PRATICA_M.
La tabella dei fatti è collegata a cinque tabelle dimensionale e ciò viene indicato attraverso i tag <dimension> che si trovano dopo il tag <Table>. La tabella dei fatti ha inoltre 3 misure, e ciò viene indicato usando il tag <Measure> nelle righe 44, 45 e 46..

La prima tabella dimensionale viene descritta usando il tab <dimension> che si trova nella quinta riga. l'attributo *foreign key* di questa tag sta ad indicare che il campo **codcompany** è una *foreign key* che fa riferimento alla dimensione di nome **Company**.
Le righe da 6 a 9 definiscono la dimensione **Company**. In particolare, la tag < Hierarchy> indica che la dimensione è composta da livelli. L'attributo allMemberName="Tutte" indica che quando voglio scegliere di visualizzare tutti i livelli di questa dimensione, devo formulare la query MDX scrivendo **Tutte** come livello della dimensione (sarà più chiaro quando verrà spiegato MDX), mentre l'attributo *primary key*, indica che il *primary key* di questa dimensione si chiama

codcompany e quindi la *foreign key* della tabella dei fatti (**codcompany**) fa riferimento alla chiave primaria (**codcompany**) di questa dimensione.
Nella settima riga si vede il tag <table> che indica che la tabella mappata a questa dimensione è la tabella di nome **BID_COMPANY** che ha un livello, indicato con la tag <level>, che ha un nome **company** (ciò è indicato con l'attributo *name*) e di tipo stringa (perché come possiamo vedere nello schema a stella, nella tabella BID_COMPANY, il *primary key* **codcompany** è di tipo varchar(20)).
Dopodiché si vedono le due tag </hierarchy> e </dimension> che indicano la fine della definizione della gerarchia e della dimensione.

Per quanto riguarda UNIQUEMEMBER, esso viene utilizzato per ottimizzare la generazione di SQL. Se sappiamo che i valori di un determinato livello nella tabella dimensionale sono unici in tutti gli altri valori nel livello superiore, quindi si imposta uniqueMembers = "true", altrimenti, si imposta a "false". Ad esempio, una dimensione tempo [anno]. [Mese] avrà uniqueMembers = "false" per mese, siccome lo stesso mese appare in diversi anni. D'altro canto, se si ha una gerarchia [classe di prodotto]. [Nome Prodotto], e sono sicuro che [nome prodotto], è unico, quindi è possibile impostare uniqueMembers = "true". Se non siamo sicuri, allora bisogna sempre impostare uniqueMembers = "false".
Nel livello più alto, esso sarà sempre uniqueMembers = "true", in quanto non vi è alcun livello più alto.

Per quanto riguarda le altre dimensioni, le cose sono uguali, però ci sono alcune noti importanti da indicare per evitare gli errori dovuti alla mal definizione dello schema:

1.bisogna rispettare la gerarchia dei tag, cioè non posso mettere <cubo> e poi <schema>, Oppure <table> e poi <cubo> ma bisogna sempre seguire la gerarchia seguente:

```
<Schema>
  <Cube>
    <Table>
    <Dimension>
      <Hierarchy>
        <Table>
        <Level>
      </Hierarchy>
    </Dimension>
    <Measure>
  </Cube>
</Schema>
```

2. bisogna rispettare la gerarchia dei livelli con cui vorremmo vedere i risultati, cioè se vogliamo Vedere che il semestre è composto da trimestri allora devo definire il livello del semestre prima e poi il livello del trimestre. Se viene fatta la definizione del trimestre prima e poi del semestre, ciò causerà degli errori nel risultato del *report* generato

3. un'ultima nota riguarda il numero delle dimensioni definiti; Come possiamo vedere nello schema XML definito, ci sono 5 dimensioni, mentre nello schema a stella, ci sono solo quattro e Ciò è dovuto al fatto che la tabella dei fatti ha due *foreign key* diversi ID_DATA e ID_DATAFINE che fanno riferimento alla stessa dimensione e ciò rende necessario la definizione di due dimensioni quando viene definito lo schema XML perché solo cosi si può fare il raggruppamento di ID_DATA o di ID_DATAFINE basandosi sulla dimensione del tempo.

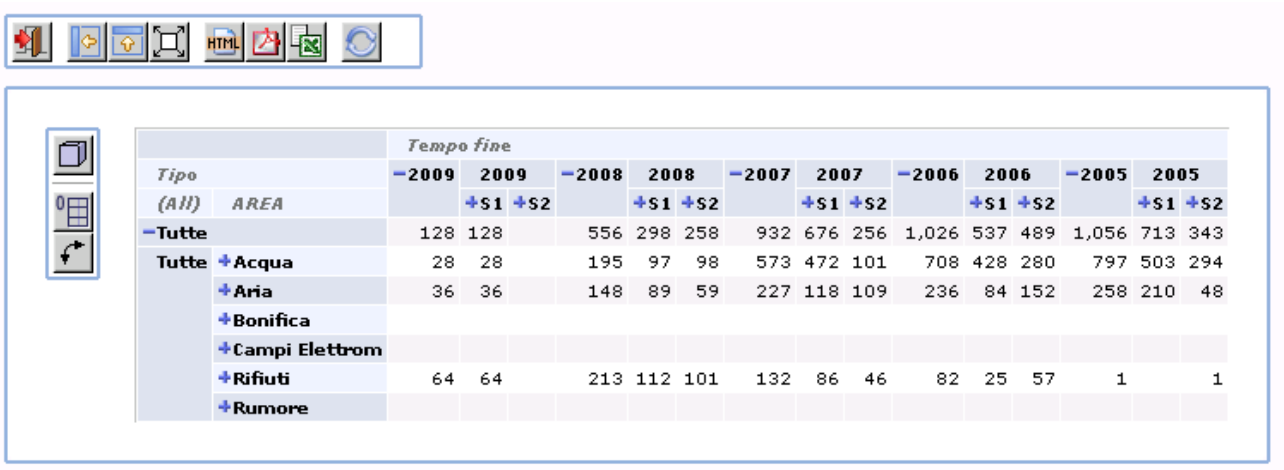
Dopo la definizione delle tabelle dimensionale si passa alla definizione delle misure della tabelle dei fatti e come si può vedere nello schema XML, ci sono 3 misure che sono N_pratiche, DURATA_MEDIA (gg), DURATA_MAX(gg).

La prima misura non fa altro che il conteggio del numero delle pratiche basandosi sull'ID delle pratiche che si trova nella tabella dei fatti.
La seconda fa la media in giorni del campo **giornidurata** (della tabella dei fatti) delle pratiche, mentre la terza sceglie la durata massima di una pratica scegliendo il massimo del campo **giornidurata** della tabella dei fatti.

Dopo aver definito lo schema XML, l'ultimo passo da affrontare consiste nell'impostare una query di default da utilizzare come istruzione MDX (nel paragrafo successivo verrà spiegato in dettaglio il linguaggio MDX) di base per inizializzare la tabella OLAP visualizzata tramite JPIVOT.
Un 'esempio di una query MDX potrebbe essere il seguente:

```
select DrilldownLevel(
            Order ({Parameter("annocorrente", [Tempo fine],
[Tempo fine].[Tutte].[2008]).prevMember.prevMember.prevMember.prevMember:ParamRef("annocorrente")}
      , [Tempo fine].currentMember.Name,DESC)) ON columns,
  UNION ( {[Tipo].[Tutte]},[Tipo].[Tutte].children ) ON rows
from [Pratica]
where
[Measures].[N_PRATICHE]
```

A questo punto si può avviare JPIVOT e testare il corretto funzionamento dell'applicazione eseguendo operazioni di navigazione sulla tabella (ogni operazione di navigazione corrisponde ad una determinata query MDX).
Avviando la *JPIVOT Table* sarà richiamata l'esecuzione della query definita e il risultato dell'elaborazione si concretizzerà nella seguente visualizzazione:



		Tempo fine														
Tipo		−2009	2009		−2008	2008		−2007	2007		−2006	2006		−2005	2005	
(All)	AREA		+S1	+S2		+S1	+S2		+S1	+S2		+S1	+S2		+S1	+S2
−Tutte		128	128		556	298	258	932	676	256	1,026	537	489	1,056	713	343
Tutte	+Acqua	28	28		195	97	98	573	472	101	708	428	280	797	503	294
	+Aria	36	36		148	89	59	227	118	109	236	84	152	258	210	48
	+Bonifica															
	+Campi Elettrom															
	+Rifiuti	64	64		213	112	101	132	86	46	82	25	57	1		1
	+Rumore															

Partendo da questa visualizzazione iniziale si può procedere diramando e modificando l'albero di navigazione in modo da suddividere, aggregare e visualizzare i dati secondo le proprie esigenze.

3.2.3 MDX (*Multidimensional Expression*):

MDX è un linguaggio di query per *database* OLAP (cubi multidimensionali) e utilizza sintassi simile a SQL (che è un linguaggio standard per interrogare le tabelle relazionali).
MDX fornisce una sintassi specializzata per interrogare e manipolare i dati memorizzati in cubi OLAP multidimensionali ed è adottato dalla maggior parte dei venditori OLAP ed è diventato lo standard di fatto per i sistemi OLAP.

Nota: MDX non ha la capacità di DDL (*Data Definition Language*) ma ha la capacità di DML (*Data Manipulation Language*) cioè con MDX non si possono creare delle strutture dati ma si può solo interrogarle.

Prima di entrare nel merito di spiegare MDX, occorre introdurre alcuni concetti base della struttura del cubo da interrogare.

Come abbiamo visto prima, un cubo è un sottoinsieme di dati organizzato e riassunti in una struttura multidimensionale ed è ciò che fornisce il meccanismo che consente tempi di risposta rapidi e uniformi per delle query complesse.

I concetti fondamentali del cubo da capire sono le dimensioni e le misure:

* le dimensioni forniscono la descrizione categorica da cui le misure sono separate per l'analisi
* le misure identificano i valori numerici riassunti per l'analisi come per esempio: **giornidurata, prezzo, costo,** ecc …..

Ogni dimensione del cubo può contenere una gerarchia di livelli per specificare la ripartizione categorica a disposizione degli utenti. Ad esempio, una dimensione negozio potrebbe includere la gerarchia di livelli seguente: Paese, Stato, Città, e Nome negozio. Ogni livello di una dimensione ha una granularità più fine rispetto al suo genitore. Allo stesso modo, la gerarchia di una dimensione tempo può comprendere i livelli: anno, trimestre, e mese.

Una dimensione può essere creata per essere utilizzata in un cubo o in più cubi. Una dimensione per un singolo cubo si chiama ***private dimension***, mentre una dimensione che può essere utilizzata da più cubi è chiamata ***Shared Dimension***.

Un ultimo elemento importante da notare è il concetto di ***Member***. Un membro non è altro che un oggetto in una dimensione o una misura. Un ***Calculated Member*** è un ***Member*** di una dimensione il cui valore è calcolato in fase di esecuzione utilizzando una determinata espressione. ***Calculated Members*** possono anche essere definiti come misure e solo le loro definizioni vengono memorizzate mentre i loro valori sono calcolati quando servono per rispondere a una query. Inoltre consentono di aggiungere membri e misure ad un cubo senza aumentare le sue dimensioni.
Anche se i ***Calculated Members*** devono essere calcolati sulla base dei dati in un cubo esistente, è possibile creare espressioni complesse combinando questi dati con degli operatori aritmetici, numeri, e una varietà di funzioni.
Sebbene il termine "cubo" suggerisce tre dimensioni, un cubo può avere fino a 64 dimensioni, comprese le misure di dimensione.

Formulazione delle query MDX

Per capire come funziona MDX partiamo a spiegare la struttura base di una query MDX e poi facciamo un esempio semplice di interrogazione di un cubo utilizzando la struttura del cubo **pratica** che è stato definito sopra. Faremo poi un altro esempio aumentando un po' la difficoltà della interrogazione.

La struttura base di una query MDX è la seguente:

```
SELECT axis [,axis]
FROM cube
WHERE slicer [,slicer]
```

La clausola SELECT viene utilizzata per definire le dimensioni sulle assi.

La clausola WHERE è usata per fare il SLICE delle dimensioni.

La clausola FROM è utilizzata per specificare il cubo da cui i dati vengono prelevati e non può essere lasciata vuota.

Il contenuto delle assi potrebbe essere:

. member – un membro è un elemento di una dimensione e corrisponde a un pezzo specifico dei dati. Ad esempio:

[Tempo].[1997]
[Tempo fine].[Tutte].[2008]
[Tipo].[Tutte].[Acqua]

La sua espressione è usata per recuperare una serie di membri enumerati da una dimensione, gerarchia, o livello. Ad esempio:

dimension.Members
hierarchy.Members
level. Members

.tuple – una tupla è una collezione di **members** da diversi dimensioni. Esempio:
([Tempo].[1997], [Tipo].[Tutte].[Acqua])
(1997, Acqua)
(1997, [Tempo fine].[Tutte].[2008])

.set – un set è una collezione di tuple. Esempio:
{[Tempo].[1997], [Tempo].[1998], [Tempo].[1999]}
{1997, 1998, 1999}
{(1997, Acqua), (1998, Acqua)}

Funzioni e espressioni di MDX:

le funzioni si riferiscono a una determinata operazione che dovrebbe essere fatta su un insieme di dati (Sum(),TopCount()).

Le espressioni descrivono la sintassi con cui la funzione viene usata ([1997].children, [Tipo].DefaultMember).

La funzione CrossJoin():

CrossJoin () viene usata per generare il prodotto cartesiano di due insiemi in ingresso. Se i due insiemi esistono in due dimensioni indipendenti, l'operatore del ***CrossJoin*** crea un nuovo insieme costituito da tutte le combinazioni dei membri delle due dimensioni e vengono utilizzate come segue: ***crossjoin*** (set1, set2)

Le funzioni TopCount() e BottomCount():
Queste espressioni ordinano un insieme di dati basandosi su una espressione numerica e scelgono l'elemento con l'indice più alto in base a graduatoria stabilita e vengono utilizzate come segue:
TopCount (set, index, espressione numerica)
BottomCount (set, index, espressione numerica)

La funzione Filter():
Questa funzione è utilizzata per filtrare un insieme in base a una condizione particolare:
Filter (set, search condition)

La funzione Order():
Questa funzione fornisce la capacità di ordinamento all'interno del linguaggio MDX e viene utilizzata come segue:

Order (set, string expression [, ASC | DESC | BASC | BDESC])
or
Order (set, numeric expression [, ASC | DESC | BASC | BDESC])

Query# 1.1

```
SELECT Measures.MEMBERS ON COLUMNS,
[company].MEMBERS ON ROWS
FROM [pratica]
```

Questa query soddisfa la richiesta di interrogare le misure registrate nella tabella dei fatti per ogni **company** insieme a una sintesi per ogni livello definito nella gerarchia della dimensione Company.
In alternativa, visualizza le misure per la gerarchia di **company**. Eseguendo questa query, si vede una riga denominata "All Company". Un membro "Tutte" viene generato per default e diventa un membro predefinito per la dimensione.

La definizione delle assi potrebbe essere racchiusa tra parentesi graffe, che sono utilizzate per indicare l'insieme. Oltre a prendere le membri di una dimensione, un solo membro di una dimensione potrebbe essere selezionato

Query# 1.2

```
select
DrilldownLevel(Order ({Parameter("annocorrente",[Tempo fine].
                                    [Tempo fine].[Tutte].[2008]).prevMember.prevMember.prevMember.prevMember
                                    :ParamRef("annocorrente")}, [Tempo fine].currentMember.Name,DESC)) ON columns,
{[Tipo].[Tutte].Children} ON rows
from [Pratica]
```

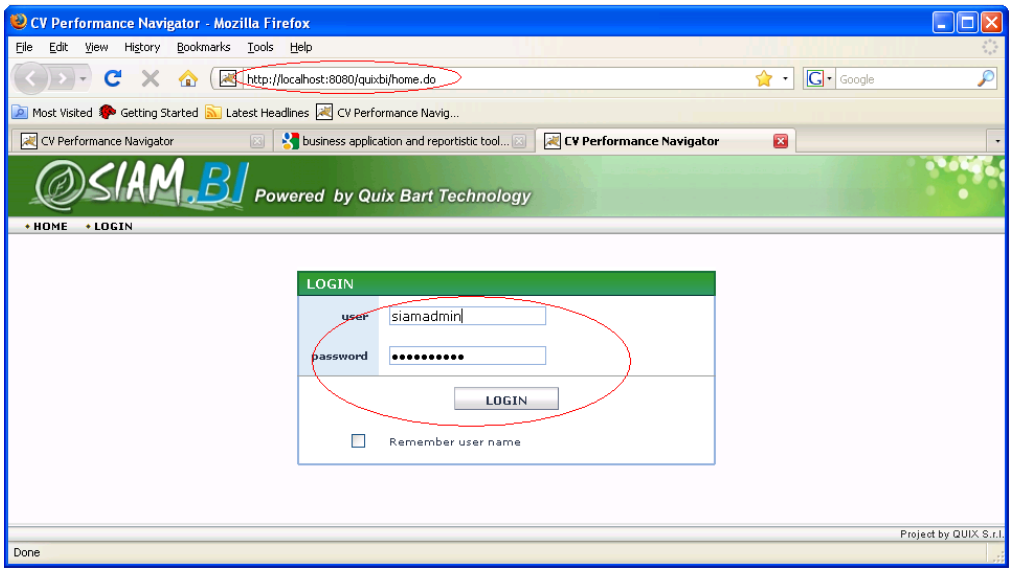
Questa query visualizza quattro colonne che sono relativi ai quattro anni a partire dall'anno corrente (che è un parametro che si chiama **annocorrente**) andando all'indietro come viene indicato con **.prevMember** (come si vede ci sono 4 .**prevMember**) e gli anni vengono visualizzati in ordine discendente attraverso l'utilizzo della funzione ORDER e anche a causa della presenza di DESC.

Sui dati delle colonne si possono fare le operazioni di Drill-down e quindi anche di Roll-Up per ciascun livello grazie alla presenza della funzione DRILLDOWNLIVELL.
Inoltre vengono visualizzati sulle righe tutti i valori del primo livello della dimensione TIPO con la possibilità di visualizzare anche i discendenti di questo livello e ciò viene fatto attraverso l'utilizzo di **.children**.
I dati visualizzati nella risposta alla query sono presi dal cubo PRATICA come viene indicato dalla clausola FROM.

Si può ottenere delle informazioni dettagliate su MDX, come viene creata la query , la lista di tutte le funzioni che possono essere usate in MDX e le espressioni di tutte le funzioni visitando il sito http://mondrian.pentaho.org/documentation/mdx.php.

3.2.4. Utilizzo di BART

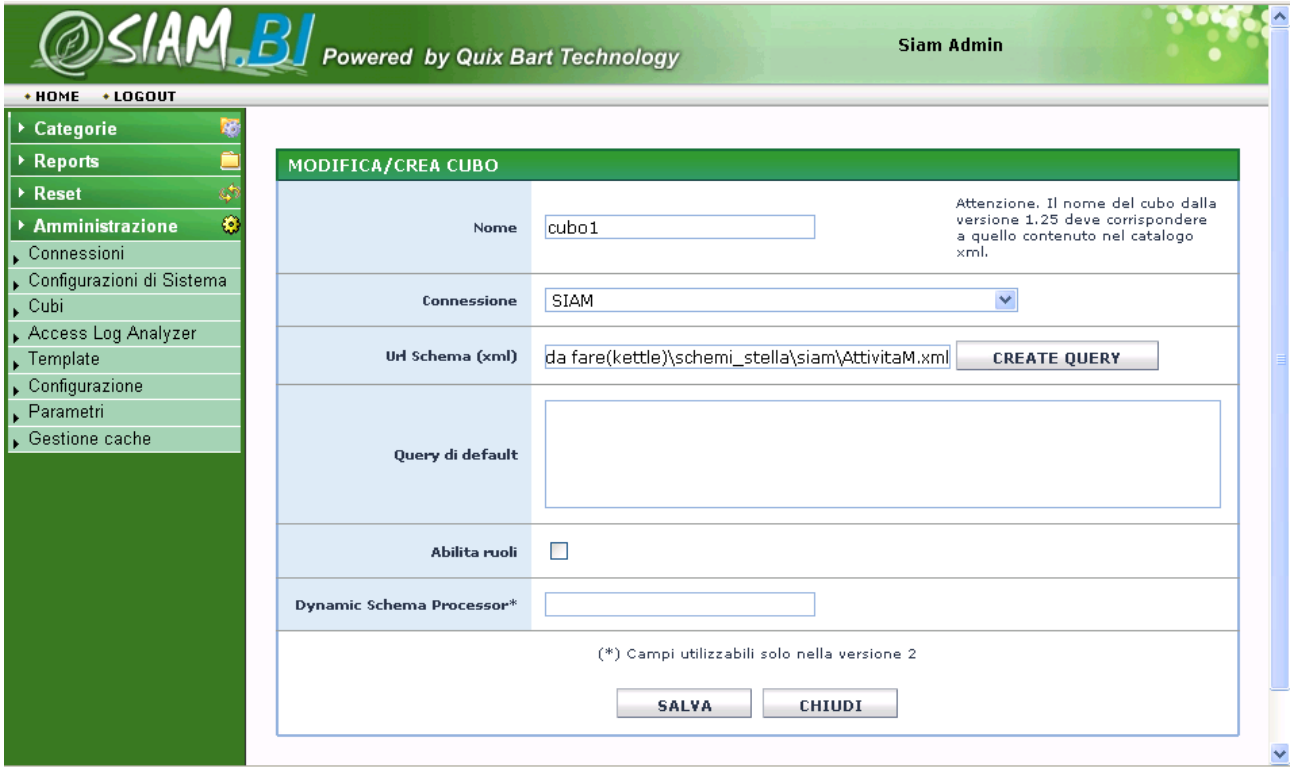
La figura seguente mostra la schermata che viene visualizzata quando si vuole accedere a BART e chiede l'autenticazione per loggare:



Quando si fa il *login* in BART viene visualizzata la schermata seguente:

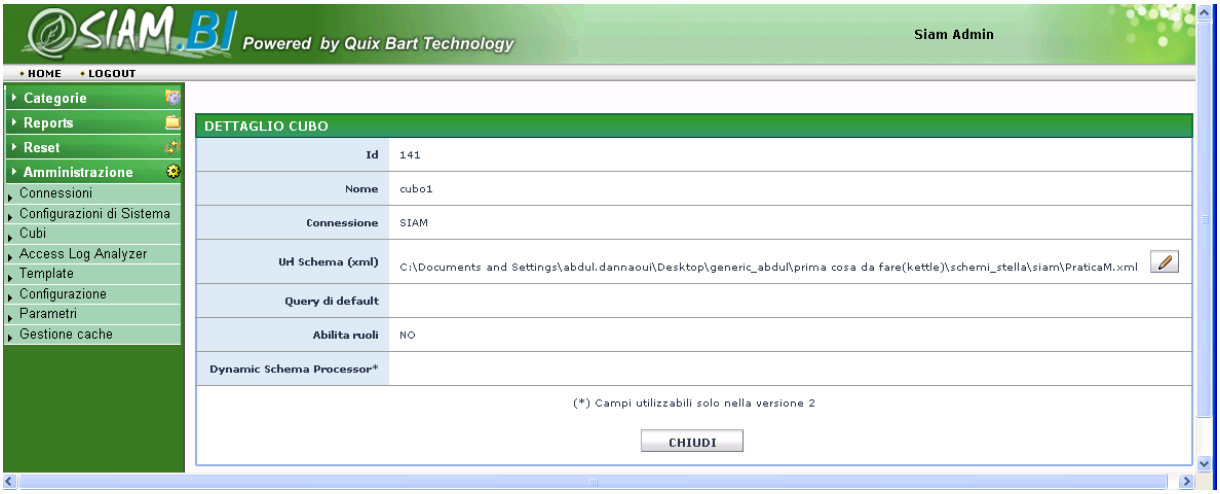
E quindi si sceglie la voce Amministrazione, poi la voce Cubi e allora viene visualizzata la seguente schermata:

E quindi si clicca il pulsante nuovo che fa apparire la seguente schermata:

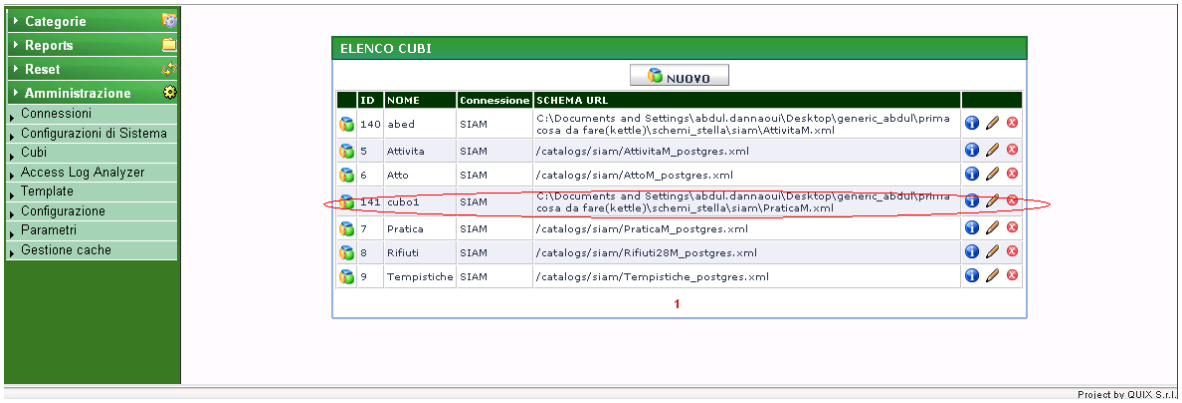


NOME: Serve a dare un nome al nuovo cubo che si intende creare.
CONNESSIONE: Specifica il nome della database a cui si intende fare la connessione.
URL SCHEMA (xml): Specifica il *path* dove si trova il file xml che definisce il cubo.
Query di default: specifica la query che viene lanciata per default. Se non viene specificato niente, allora BART crea una query di default con tutte le misure sulle colonne.

E poi si fa salva, quindi si vede la schermata seguente:

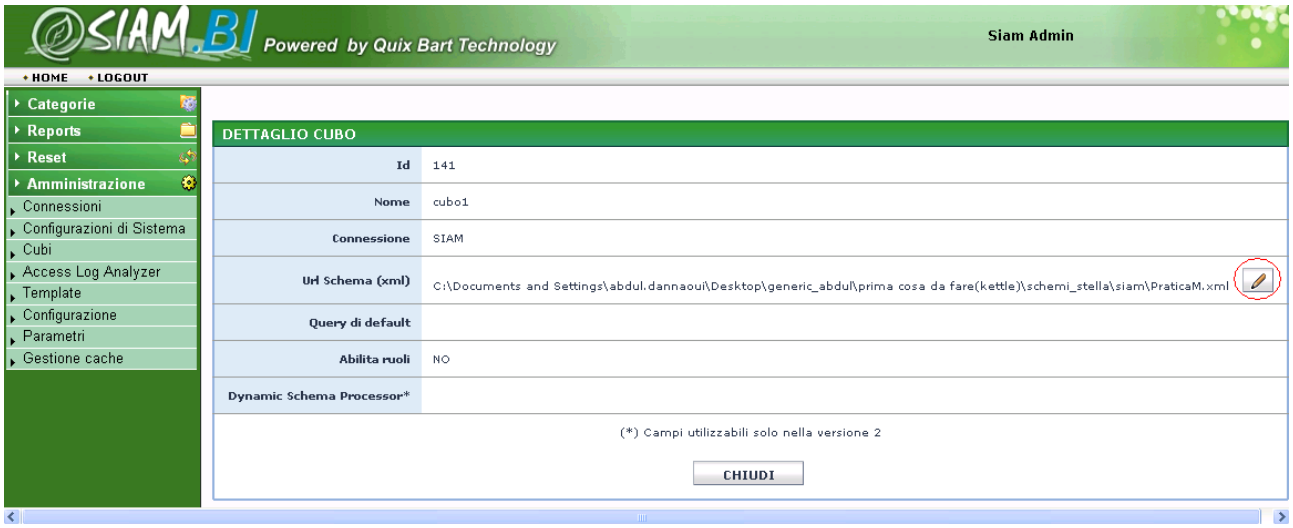


E cliccando chiudi viene visualizzata la schermata seguente:

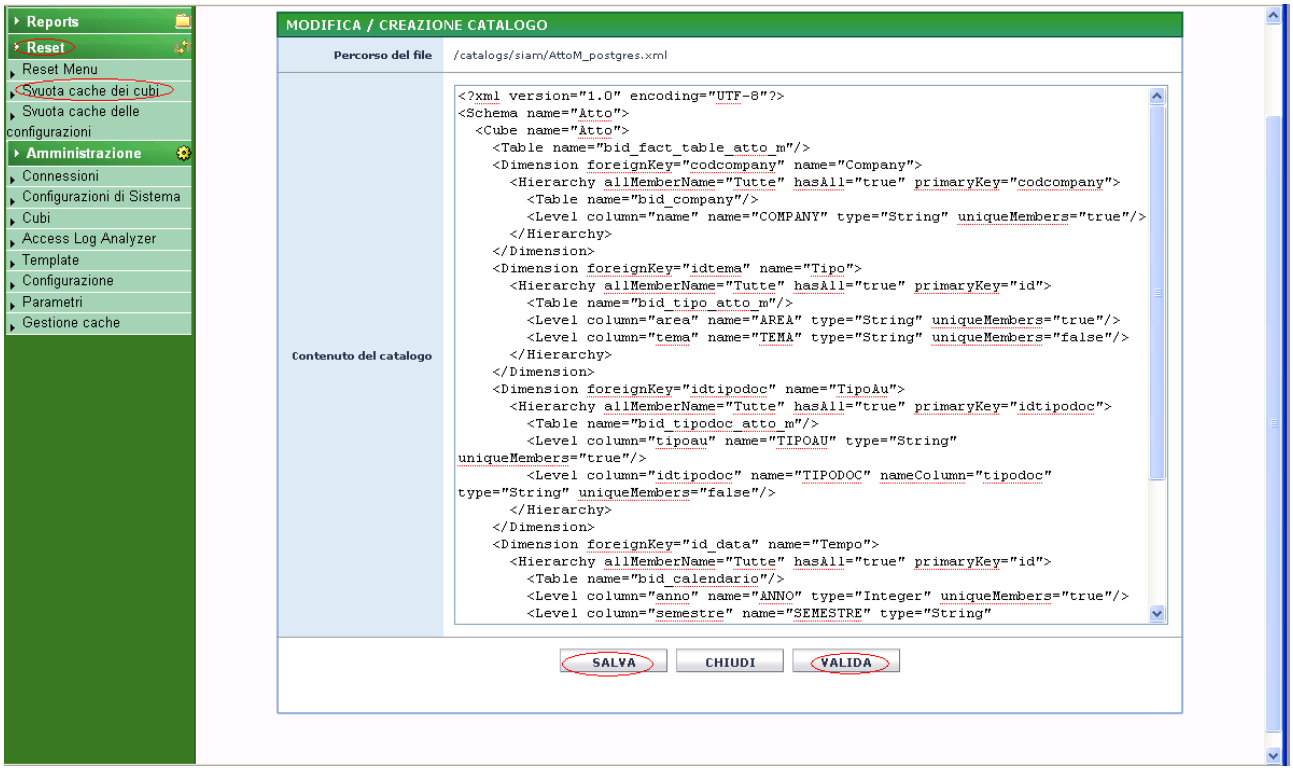


Che indica che il nostro cubo è stato creato come si vede indicato nella figura sopra.
Quando si vuole cambiare la struttura del cubo, si può fare in due modi:
1. Cancellare il cubo cliccando sull'icona in rosso che si trova a destra come si vede nell'immagine e ricrearlo di nuovo seguendo lo stesso procedimento che abbiamo visto prima

2. cliccando sul cubo e poi cliccando il pulsante indicato nell'immagine che segue:

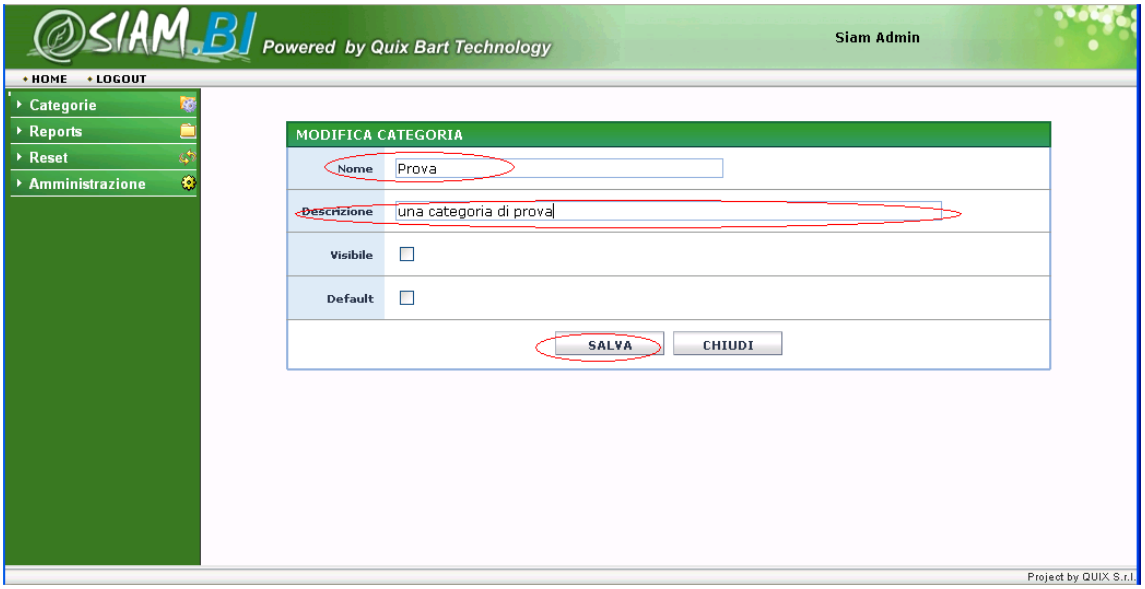


Quindi si vede la schermata in cui viene modifica la struttura:



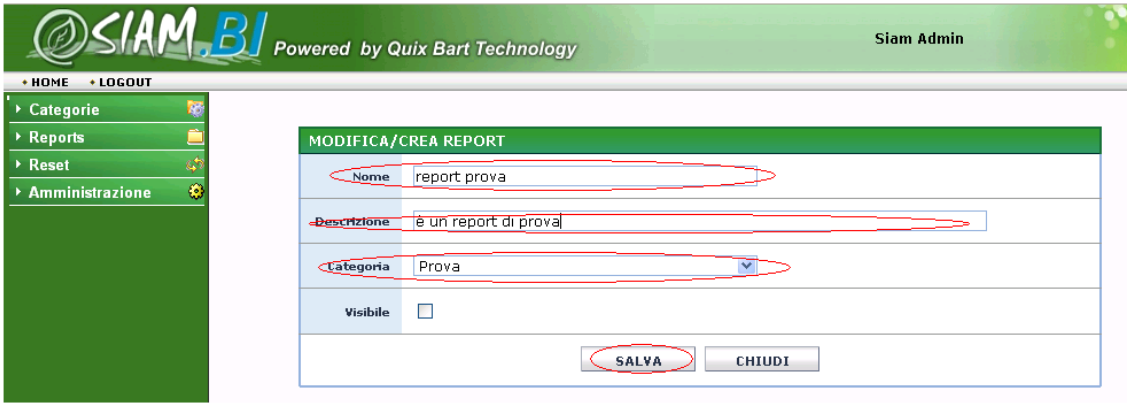
Quindi si può verificare se le modifiche sono giuste cliccando su Valida oppure salvare le modifiche cliccando salva.
Dopodiché bisogna aggiornare la struttura nella cache, perché come abbiamo detto *Mondrian* usa la cache per fare le operazioni, quindi si clicca **Reset** nel menu principale a sinistra e poi **svuota cache dei cubi**.

Adesso che abbiamo definito la struttura del cubo creando il file XML, passiamo alla creazione del *report* formulando una query MDX.
Prima di creare il *report*, bisogna creare una nuova categoria quindi si clicca **categorie, Nuova** e quindi viene visualizzata la seguente schermata:



Mettiamo il nome della categoria e la sua descrizione e poi salviamo cliccando il pulsante SALVA e quindi la categoria viene creata:

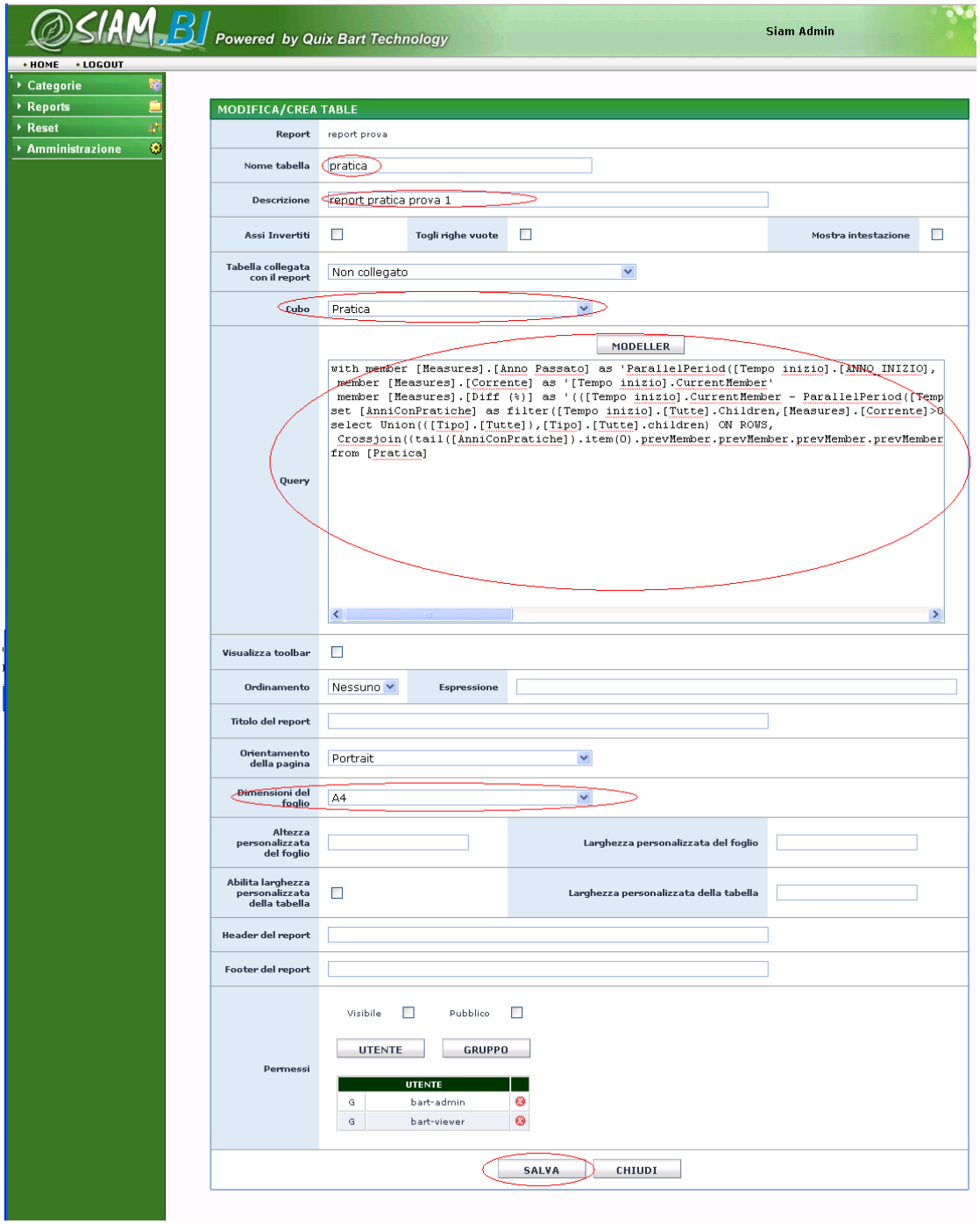
Adesso si passa a creare il *report* quindi si clicca ***Report***, **Nuova** e si vede la schermata seguente:



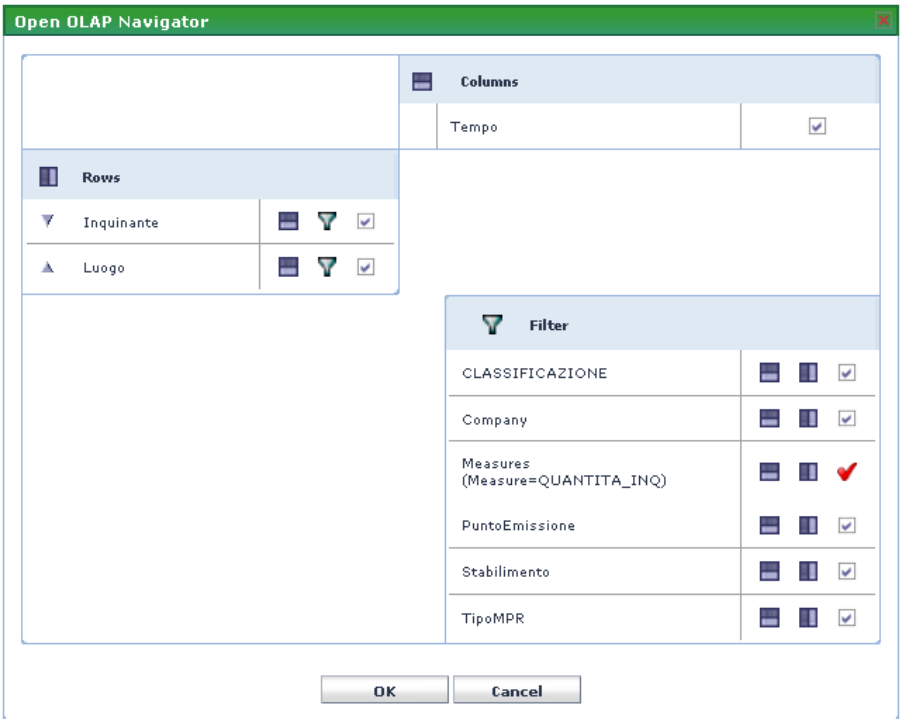
Quindi mettiamo il nome del *report*, la sua descrizione e a che categoria appartiene e poi salviamo cliccando il pulsante **Salva**, e quindi viene visualizzata la schermata seguente:

Adesso per creare il *report* si clicca **report prova** che mostra la seguente schermata:

E quindi si sceglie di creare una tabella, un grafico, una nuova pagina JSP, ecc ……. Quindi per il nostro esempio facciamo **nuova tabella** e quindi viene visualizzata la schermata seguente:

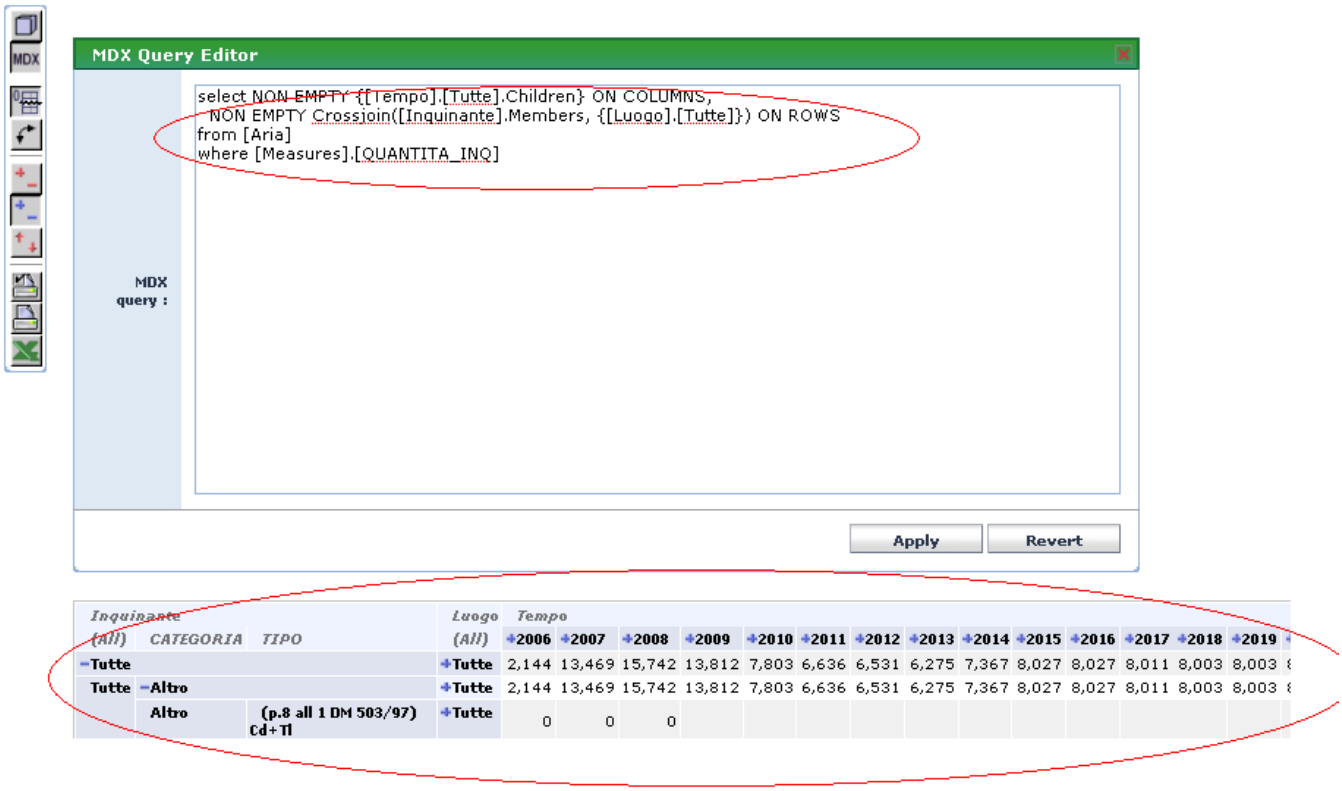


A questo punto, si mettono le informazioni necessarie per la creazione della nuova tabella.
Nome: il nome della tabella.
Descrizione: la descrizione della nuova tabella e a cosa serve.
Cubo: il nome del cubo di cui prendere i dati.
Query: è la query MDX da eseguire.
Inoltre, per la creazione della query MDX si può utilizzare il *Modeller* cliccando il pulsante **Modeller** che si trova sopra il campo della query MDX e quindi viene visualizzata la schermata seguente:



Questa schermata permette la selezione delle dimensioni da vedere sulle righe e sulle colonne e permette anche le operazioni di SLICE delle dimensioni fissando una dimensione a un valore determinato.

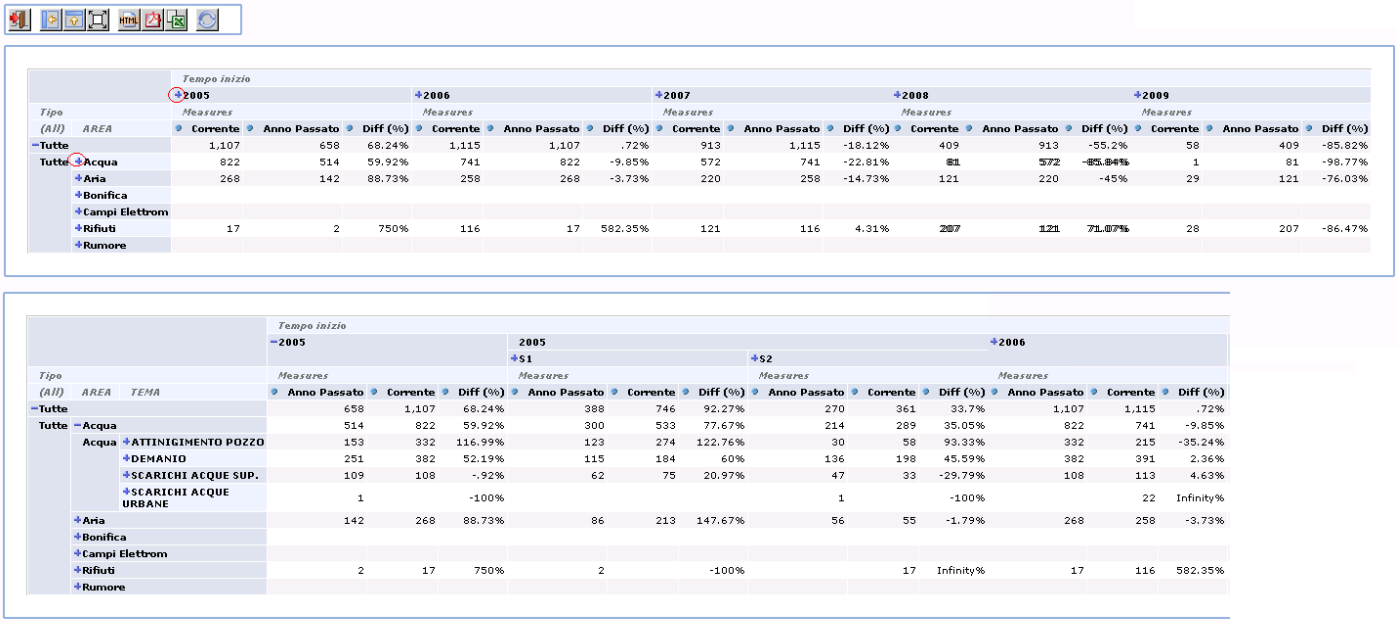
Quando viene fatta la scelta delle dimensioni e le condizioni di filtraggio, si clicca OK e quindi viene generata la query MDX che viene visualizzata con il risultato sotto come segue:



Se il risultato visualizzato va bene allora si copia la query MDX e si incolla nel campo MDX query. Dopodiché viene salvato il *report* cliccando su **salva**.

Per visualizzare il risultato del *report* creato, si clicca REPORT, REPORT PROVA, e poi si sceglie PRATICA:

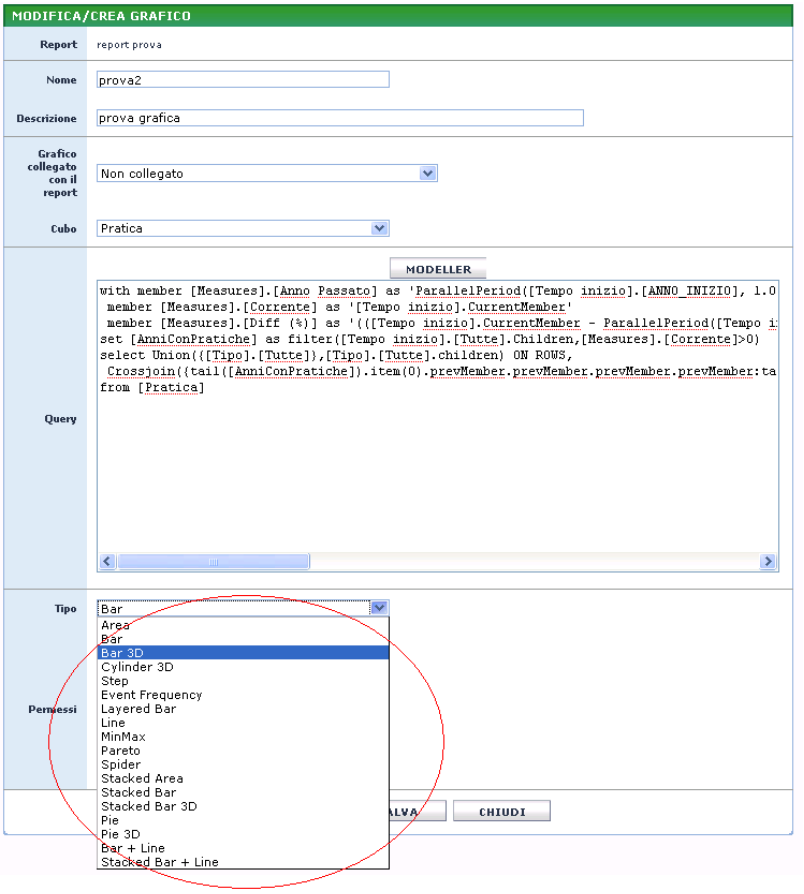
E quindi viene visualizzata la tabella su cui si possono visualizzare i dati facendo operazioni di ROLL-UP, DRILL-DOWN, ecc …..



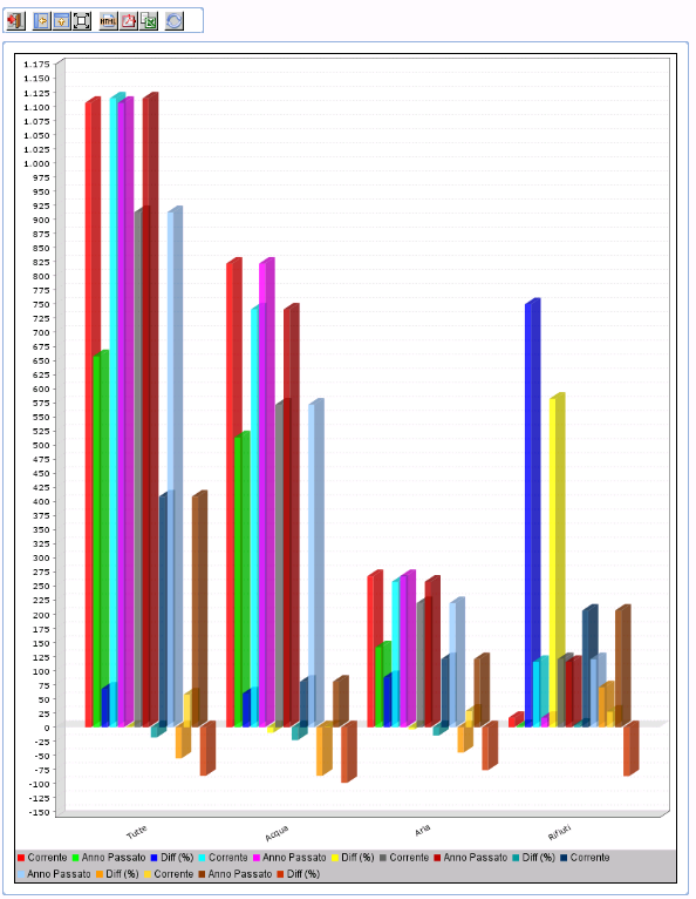
Tempo inizio	2005			2006			2007			2008			2009		
Tipo	Measures			Measures			Measures			Measures			Measures		
(All) AREA	Corrente	Anno Passato	Diff (%)	Corrente	Anno Passato	Diff (%)	Corrente	Anno Passato	Diff (%)	Corrente	Anno Passato	Diff (%)	Corrente	Anno Passato	Diff (%)
Tutte	1,107	658	68.24%	1,115	1,107	.72%	913	1,115	-18.12%	409	913	-55.2%	58	409	-85.82%
Acqua	822	514	59.92%	741	822	-9.85%	572	741	-22.81%	81	572	-85.84%	1	81	-98.77%
Aria	268	142	88.73%	258	268	-3.73%	220	258	-14.73%	121	220	-45%	29	121	-76.03%
Bonifica															
Campi Elettrom															
Rifiuti	17	2	750%	116	17	582.35%	121	116	4.31%	207	121	71.07%	28	207	-86.47%
Rumore															

Tempo inizio	2005			2005						2006		
				S1			S2					
Tipo	Measures			Measures			Measures			Measures		
(All) AREA TEMA	Anno Passato	Corrente	Diff (%)	Anno Passato	Corrente	Diff (%)	Anno Passato	Corrente	Diff (%)	Anno Passato	Corrente	Diff (%)
Tutte	658	1,107	68.24%	388	746	92.27%	270	361	33.7%	1,107	1,115	.72%
Acqua	514	822	59.92%	300	533	77.67%	214	289	35.05%	822	741	-9.85%
ATTINGIMENTO POZZO	153	332	116.99%	123	274	122.76%	30	58	93.33%	332	215	-35.24%
DEMANIO	251	382	52.19%	115	184	60%	136	198	45.59%	382	391	2.36%
SCARICHI ACQUE SUP.	109	108	-.92%	62	75	20.97%	47	33	-29.79%	108	113	4.63%
SCARICHI ACQUE URBANE	1		-100%				1		-100%		22	Infinity%
Aria	142	268	88.73%	86	213	147.67%	56	55	-1.79%	268	258	-3.73%
Bonifica												
Campi Elettrom												
Rifiuti	2	17	750%	2		-100%		17	Infinity%	17	116	582.35%
Rumore												

Se vogliamo fare un esempio di un grafico allora si fanno gli stessi passi e si sceglie nuovo grafico e quindi viene visualizzata una schermata come quella prima ma con altre opzioni. Uno delle opzioni è il tipo del grafico da visualizzare.

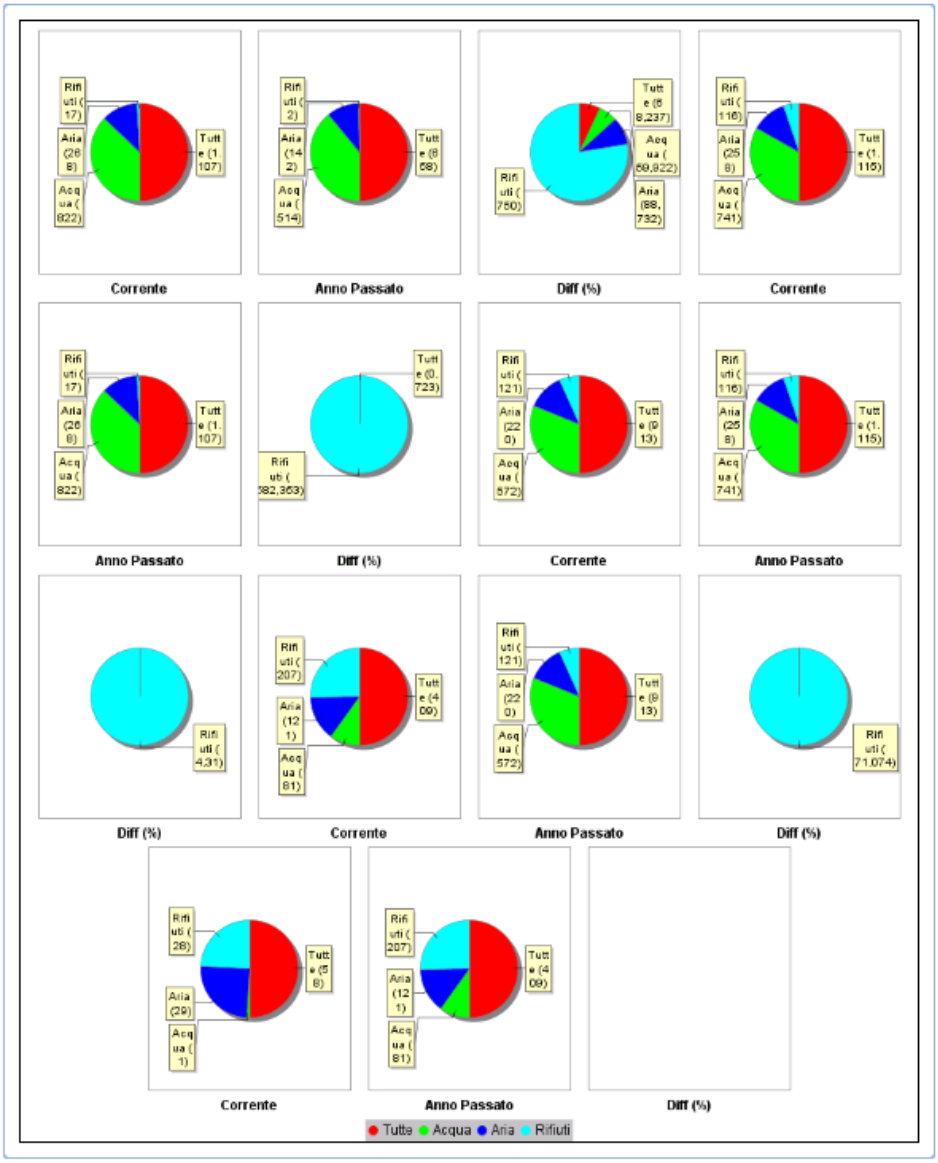


Se scegliamo bar 3D viene visualizzato il risultato della query MDX in formato grafico come segue:



Per modificare un *report*, si sceglie la cartella dove si trova e poi si clicca il seguente pulsante :

E quindi si riapre la pagina della modifica, in cui si possono introdurre le modifiche che vogliamo fare e salvarle. Ad esempio, facciamo la modifica del tipo di grafico che dovrebbe essere visualizzato e scegliamo un grafico a torta, quindi il nuovo risultato del *report* sarà come segue:



4. REALIZZAIONE DEL PROGETTO DI STAGE

Come è stato accennato nell'introduzione, il lavoro dello stage è stato suddiviso in due parti principali, la creazione della trasformazione dei dati e la creazione del cubo.
Nel resto del capitolo sarà dettagliato il lavoro fatto per la creazione della trasformazione e le diversi fasi della creazione del cubo, però, prima di entrare nel merito di ciascuna delle due parti, è opportuno descrivere il gestionale che contiene i dati transazionali, nel nostro caso è il SIAM, da cui vengono estratti utilizzando *KETTLE* per caricare i cubi OLAP .

SIAM è un software *web-based* che soddisfa le esigenze delle provincie di avere un livello operativo in materia di rilascio delle autorizzazioni e di controllo operativo nell'area ambientale e garantisce agli operatori del settore tutte le informazioni e gli strumenti per agevolare l'esecuzione delle proprie mansioni. Inoltre, SIAM è uno strumento di supporto, che agevola il coordinamento, la condivisione e la coerenza dei dati e delle procedure operative, fornisce adeguate e specifiche soluzioni per la rendicontazione, il monitoraggio e per tutte le numerosissime attività operative.

SIAM è un sistema informativo integrato in grado di raccogliere, organizzare, e uniformare tutte le pratiche, le attività, i controlli e gli atti rilasciati dai diversi uffici del settore ambiente dell'ente. Attraverso SIAM è possibile:

- Recepire le richieste di imprese e cittadini,
- Avviare nuovi processi autorizzativi o di controllo in materia ambientale,
- Raccogliere e catalogare tutte le informazioni e documenti inerenti il processo attivato,
- Pianificare e monitorare tutte le attività necessarie al completamento del processo attivato,
- Raccogliere ed eventualmente trasmettere i dati dell'istruttoria tecnica ad altri soggetti interessati/coinvolti nel procedimento (interni o esterni all'ente: altri uffici, ARPA, Comune, ecc.),
- Generare automaticamente bozze di documenti e atti,
- Vigilare sul rispetto delle normative ambientali vigenti (attraverso controlli programmati e prescrizioni),
- Verificare lo stato di avanzamento di ciascun procedimento,
- Monitorare i carichi di lavoro di ciascun settore/operatore.

I dati contenuti nel *database* SIAM costituiscono il punto di partenza per il caricamento dei cubi OLAP**.**

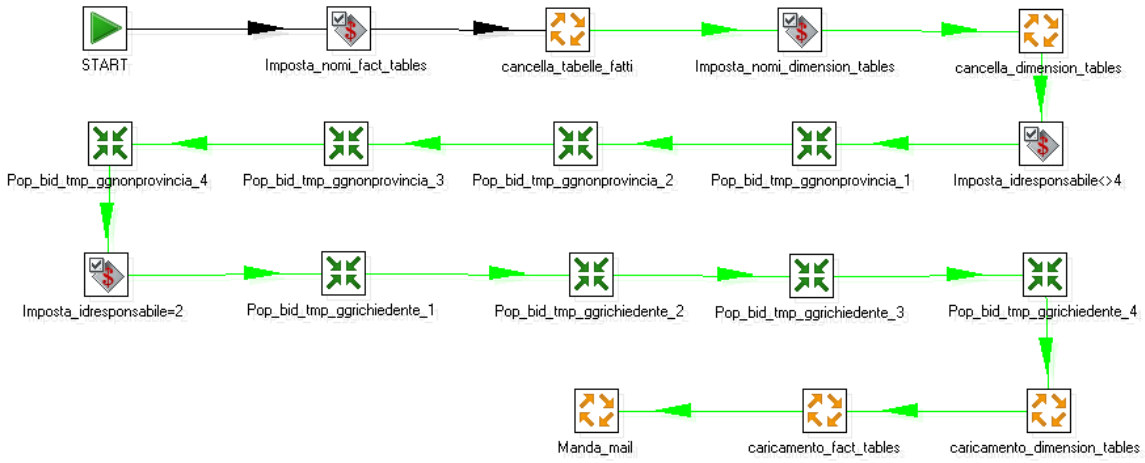
4.1 Creazione della trasformazione dei dati

Appena iniziato lo stage, l'azienda aveva realizzato quattro cubi per la creazione dei *report* per l'analisi di dati.
I processi di caricamento di questi cubi erano realizzati utilizzando delle query normali che selezionano i dati, li trasformano, li puliscono, fanno il calcolo delle misure e poi li caricano nelle tabelle dei cubi senza l'utilizzo di uno strumento ETL.
Il problema è nato quando l'azienda ha iniziato a vendere il prodotto SIAMBI (BART) a diversi provincie che usavano RDBMS diversi. Quindi, l'azienda si è trovata a creare uno script con le query di caricamento per ogni RDBMS. Il problema grave veniva incontrato quando si aveva la necessità di modificare le query di caricamento. Poiché i principali RDBMS utilizzati dalle provincie che hanno acquistato SIAM sono quattro:MYSQL, ORACLE, POSTGRES e SQLSERVER ci si trovava davanti al problema di cambiare quattro diversi file di query, e più le query diventavano complesse, più il lavoro richiedeva tempo e sforzo.
Inoltre, i flussi di caricamento dei cubi avevano delle condizioni diverse tra un ente e l'altro e quindi l'azienda si trovava davanti a delle modifiche da fare ogni volta che l'applicazione doveva essere installata presso un cliente.

Come soluzione al primo problema, è stato pensato di utilizzare uno strumento di ETL che si interfaccia con i diversi database esistenti sul mercato. Dopo aver provato vari strumenti si è deciso di utilizzare *KETTLE* che è uno strumento Open – source.
Per il secondo problema, si è pensato di passare dei parametri che vengono definiti nella fase dell'esecuzione del flusso e quindi vengono passati una volta sola con gli altri parametri di connessione alla base di dati secondo le esigenze del cliente.

Con *KETTLE* è stato realizzato un flusso generale che include parametri di connessione alla base di dati e, quindi, permette di risolvere il problema dell'eterogeneità dei basi di dati utilizzati e altri per impostare le condizioni secondo le esigenze del cliente.

Il flusso generale è il seguente:



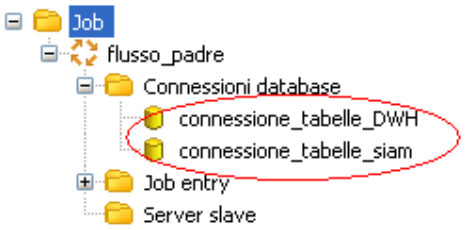
Nel flusso creato ci sono tanti punti che differiscono con il modo usuale del caricamento delle tabelle di *data warehouse* e che sono state fatte secondo la scelta dell'azienda.
Il flusso inizia con l'impostazione dei nomi delle tabelle dei fatti e poi si cancellano i dati contenuti in essi. Qua si incontra la prima differenza, perché eliminare le tabelle e poi caricarle era una scelta

dell'azienda perché di solito il caricamento nelle tabelle viene in modo incrementale oppure differenziale mantenendo i dati già esistenti nelle tabella senza eliminarli.
Questa scelta è stata fatta perché il volume dei dati da caricare è basso e quindi eliminarli e poi caricarli non comporta tanta perdita di tempo e semplifica il flusso perché non ci sarà più bisogno di creare delle trasformazioni che fanno la differenza tra i record esistenti e i record nuovi per poi fare il caricamento differenziale nelle tabelle.
Dopo la cancellazione delle tabelle di fatti vengono passati i nomi delle tabelle dimensionali da eliminare e caricare di nuovo (ciò dovrebbe avvenire dopo lo svuotamento delle tabelle dei fatti se non viene visualizzato un errore a causa del riferimento del *Foreign key* delle tabelle dei fatti alle chiavi primarie delle tabelle dimensionali). Dopodiché, viene passato un parametro da utilizzare in una condizione di una query utilizzata nella trasformazione che è suddivisa in quattro parti per caricare una tabella temporanea e poi viene passato un altro parametro alla stessa trasformazione per caricare un'altra tabella temporanea e queste due tabelle vengono utilizzate per il caricamento della tabella dei fatti **tempistiche** e alla fine vengono caricate le tabelle dimensionali e poi le tabelle dei fatti e viene mandata una mail che contiene i record scartati a causa di un errore di inserimento o a causa di fallimento del *lookup* in una tabelle dimensionale durante il caricamento delle tabelle dei fatti.
Di seguito verranno spiegati tutti i passaggi che sono stati fatti per la creazione di questa trasformazione.

La lista dei parametri da passare al flusso generale è la seguente:

#	Nome	Valore
1	connection_url_dwh	jdbc:oracle:thin:@192.168.100.22:1521:srvorcl10
2	connection_url_siam	jdbc:oracle:thin:@192.168.100.22:1521:srvorcl10
3	driver_class_name_dwh	oracle.jdbc.OracleDriver
4	driver_class_name_siam	oracle.jdbc.OracleDriver
5	password_dwh	siam_prov_pc
6	password_siam	siam_prov_pc
7	user_name_dwh	siam_prov_pc
8	user_name_siam	siam_prov_pc
9		
10	soglia	715
11	limite_DurataGiorniOrig	900
12		
13	ind_destinazione	abdul.dannaoui@quix.it
14	nome_ind_risp	prova@quix.it
15	server_smtp	mail.quix.it
16	Porta	25

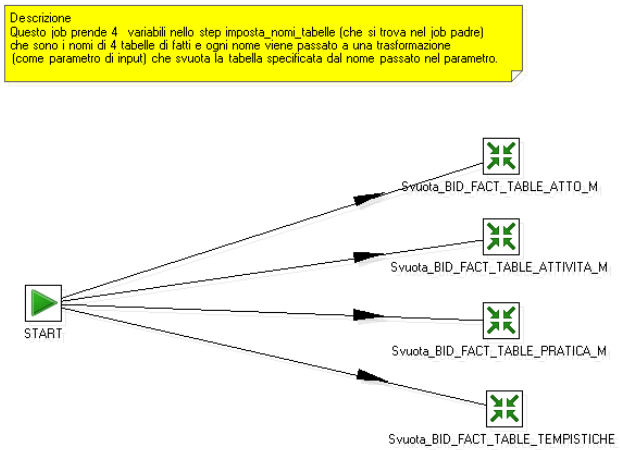
I primi otto parametri sono quelli che determinano la connessione alla base di dati e vengono ripetuti due volte perché nella trasformazione vengono definite due connessioni, una per la connessione alle tabelle del SIAM e l'altra per la connessione alle tabelle del *data warehouse* come si vede nella figura che segue:



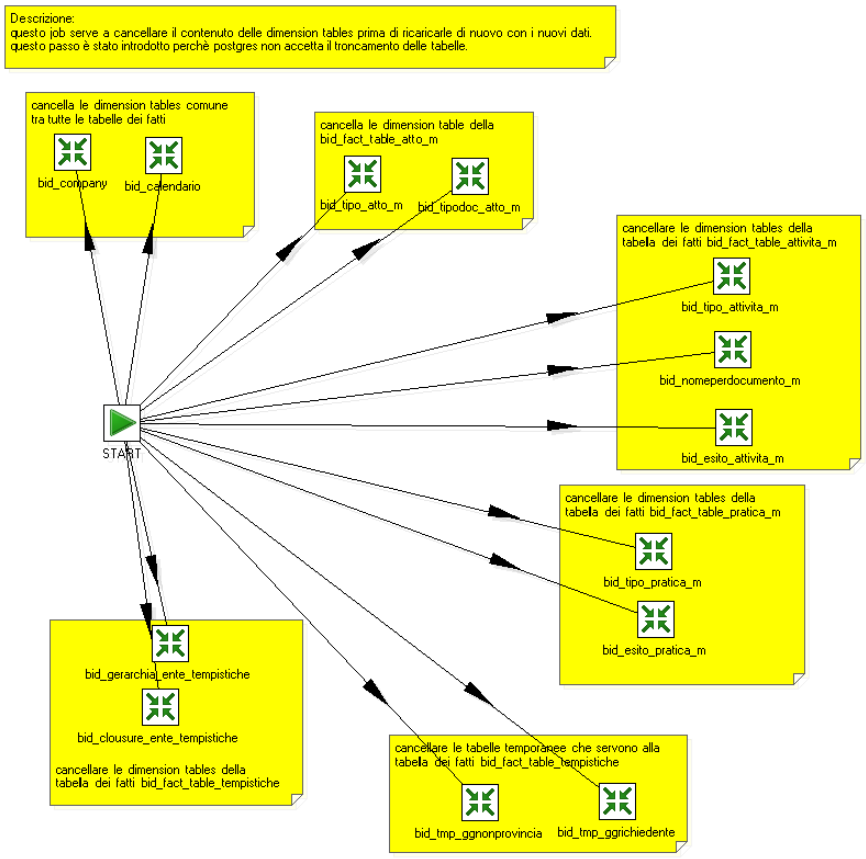
Anche se queste due connessioni sono uguali, l'azienda ha scelto di definire due connessione per eventuali situazioni future che potrebbero richiedere due installazione dei due *database* su due diversi RDBMS. Inoltre le due connessioni sono uguali perché le tabelle del *data warehouse* e del SIAM usano lo stesso RDBMS per soddisfare le esigenze dei clienti di avere un solo RDBMS.

I due parametri che seguono sono delle parametri che sono diversi tra un ente e l'altro e quindi vengono impostati secondo le esigenze del cliente.
I parametri che restano sono i parametri che determinano l'indirizzo a cui la mail degli scarti dovrebbe essere inviata.
Di seguito vengono riportate le immagini di ciascun JOB e viene descritto il suo ruolo nel flusso globale.

Il primo JOB che discutiamo è il JOB che cancella le tabelle dei fatti; come si vede sono quattro cubi che saranno cinque una volta terminata la progettazione del nuovo cubo.
Ciascuna delle trasformazioni (in verde) elimina una tabella.

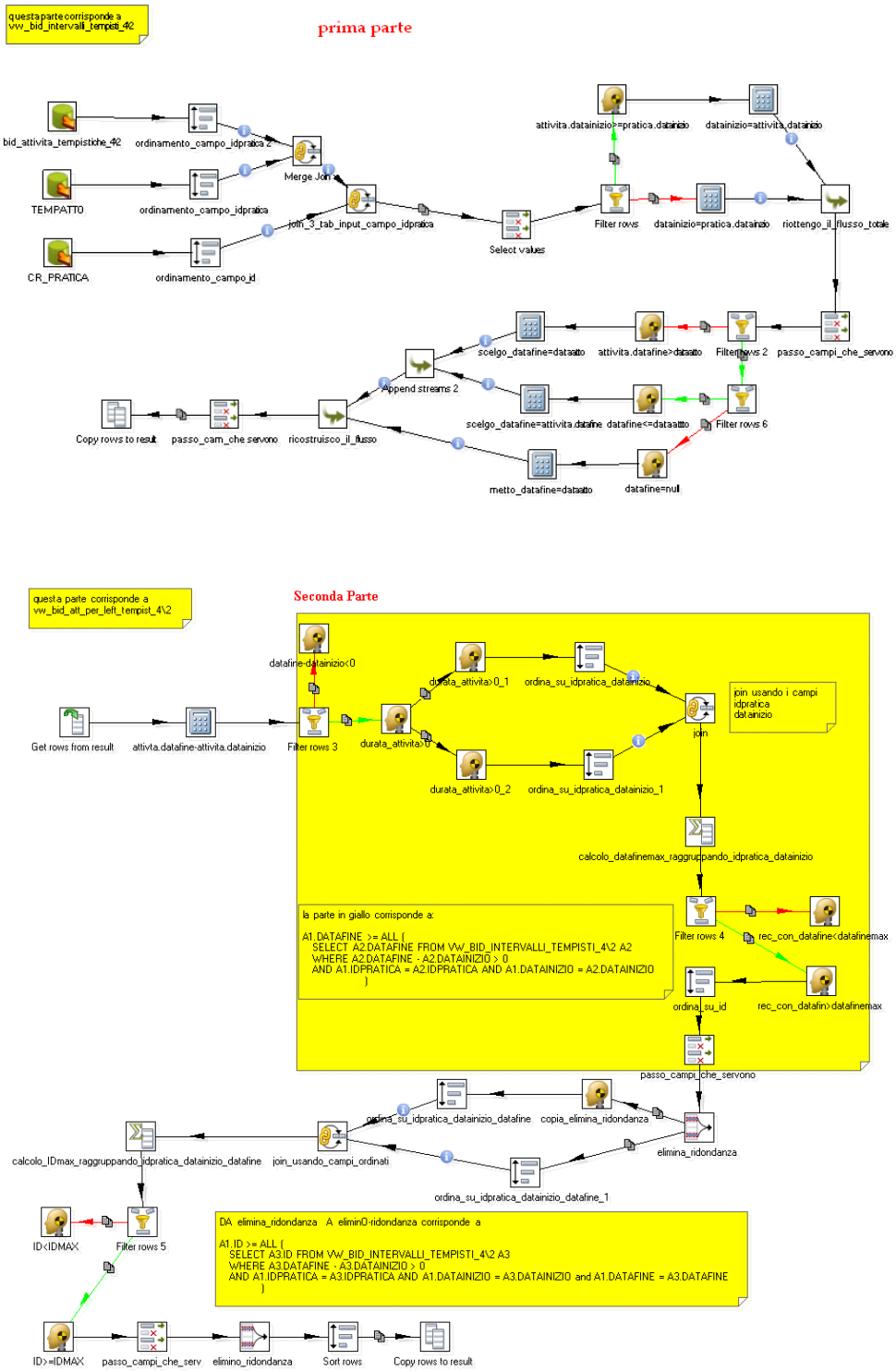


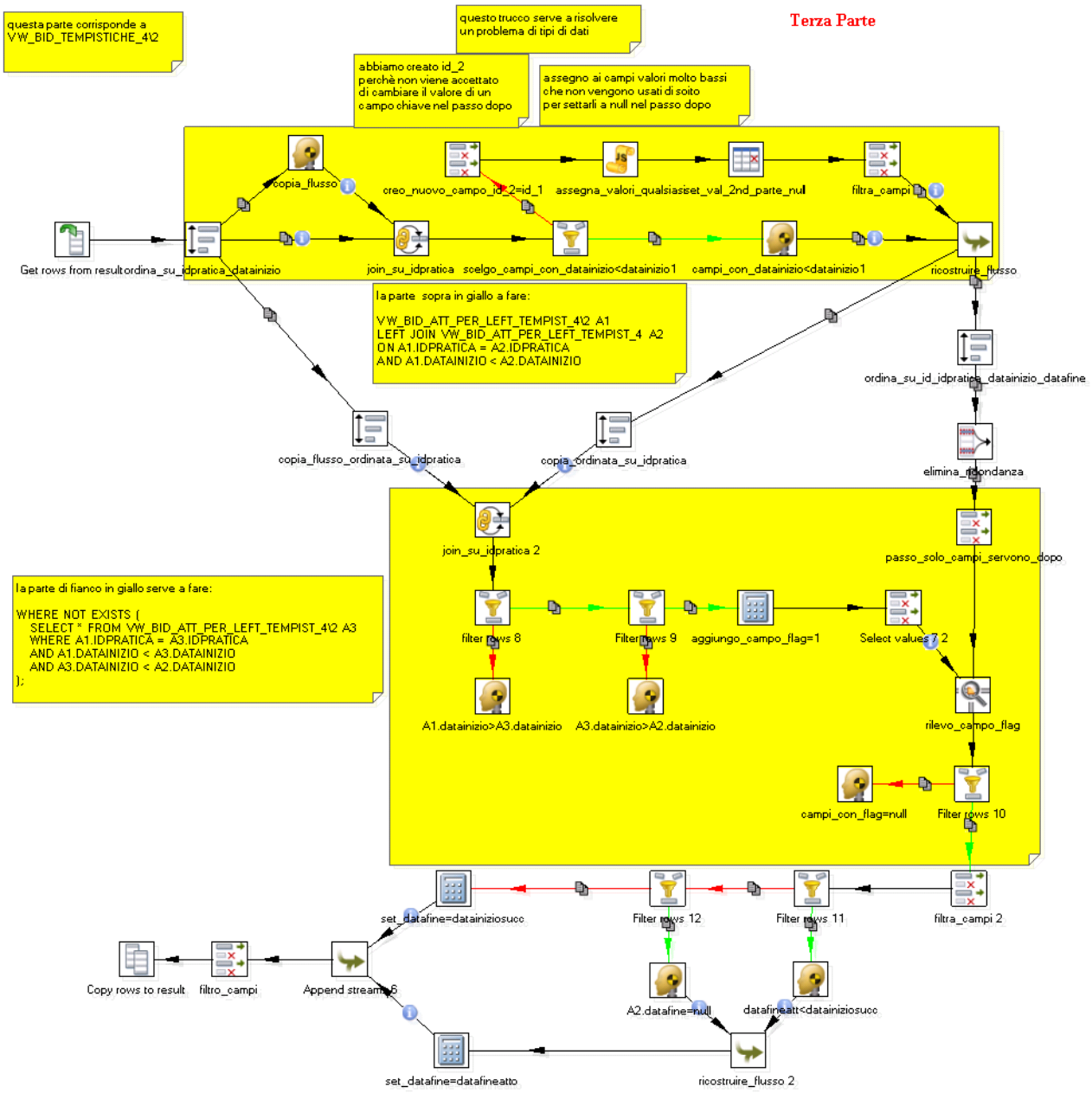
Il secondo JOB è uguale al precedente però cancella le tabelle dimensionali e la cancellazione viene eseguita in modo parallelo su tutte le tabelle che vengono visualizzate nel JOB come si può vedere nell'immagine che segue:

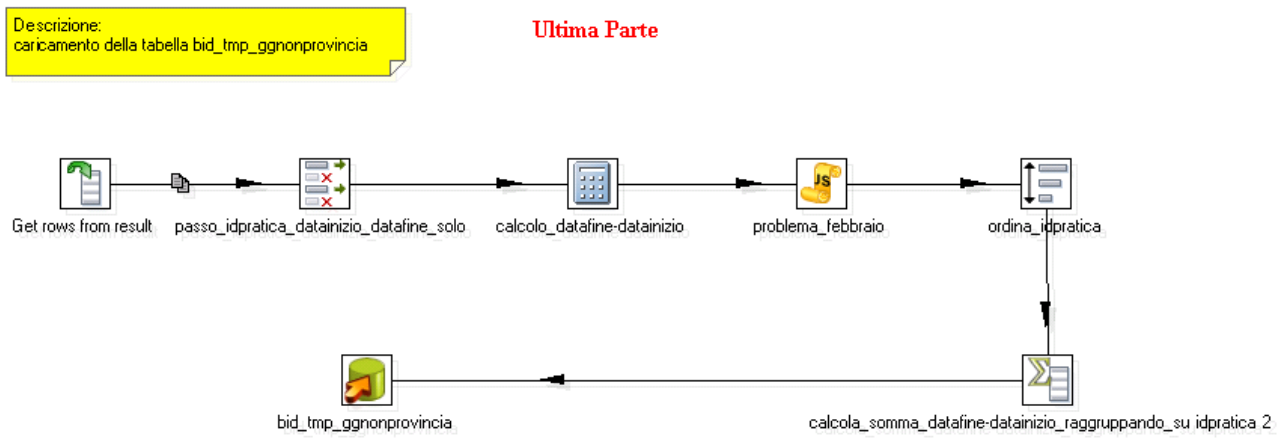


Prima di introdurre il JOB della cancellazione delle tabelle dimensionale era stata fatta la scelta del troncamento della tabella quando si arriva a caricarla e quindi il JOB precedente non serviva, però durante la prova del flusso con POSTGRES ci siamo accorti che non accetta il troncamento e ci siamo trovati davanti a questa unica scelta che alla fine abbiamo adottato.

La trasformazione che segue è una delle trasformazioni più pesanti del flusso perché è necessario leggere tante tabelle e fare tante elaborazioni dei dati prima di caricarli nelle tabelle temporanee che servono a caricare la tabella dei fatti TEMPISTICHE.
Di seguito vengono visualizzate le diverse parti della trasformazione :

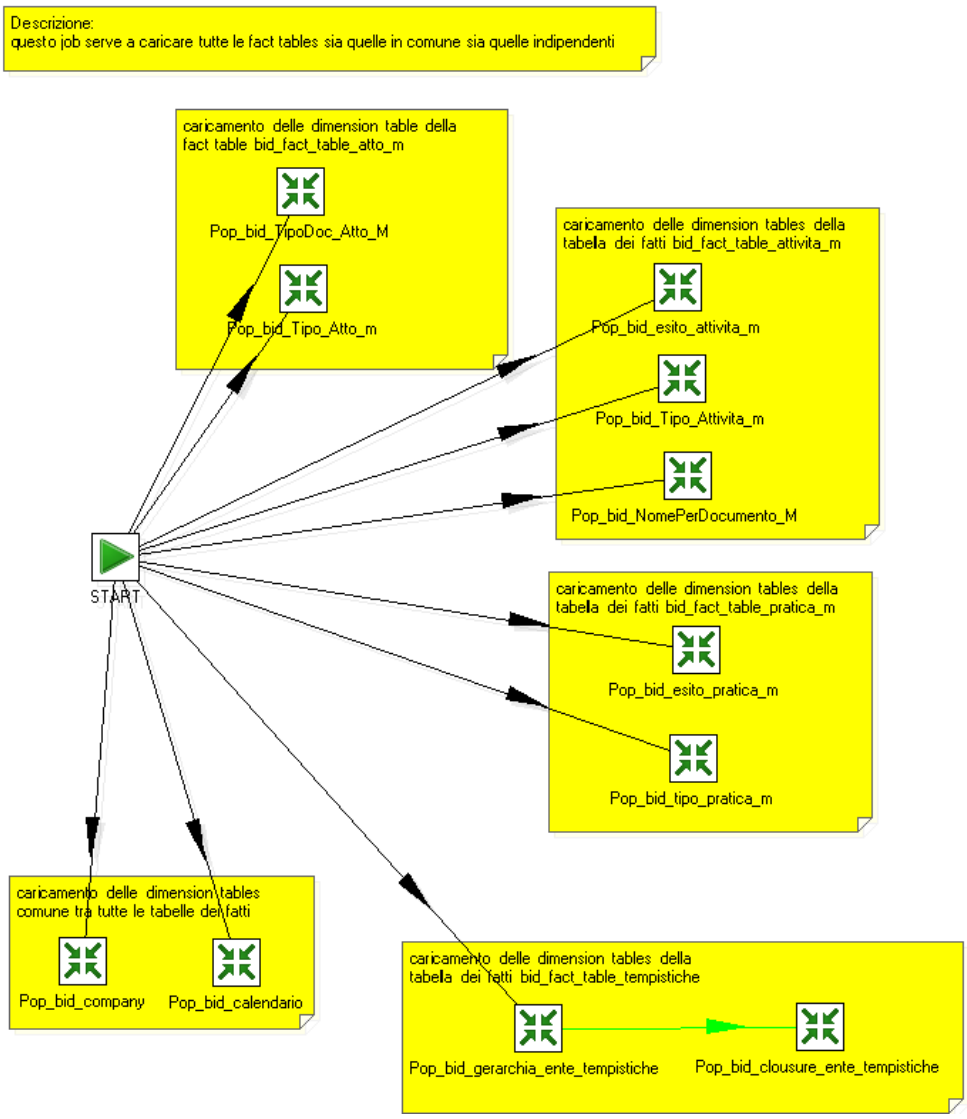



questa parte corrisponde a
VW_BID_TEMPISTICHE_4\2
questo trucco serve a risolvere
un problema di tipi di dati
Terza Parte
abbiamo creato id_2
perchè non viene accettato
di cambiare il valore di un
campo chiave nel passo dopo
assegno ai campi valori molto bassi
che non vengono usati di soito
per settarli a null nel passo dopo
copia_flusso
creo_nuovo_campo_id_2=id_1
assegna_valori_qualsiasiset_val_2nd_parte_null
filtra_campi
Get rows from result
ordina_su_idpratica_datainizio
join_su_idpratica
scelgo_campi_con_datainizio<datainizio1
campi_con_datainizio<datainizio1
ricostruire_flusso
la parte sopra in giallo a fare:
VW_BID_ATT_PER_LEFT_TEMPIST_4\2 A1
LEFT JOIN VW_BID_ATT_PER_LEFT_TEMPIST_4 A2
ON A1.IDPRATICA = A2.IDPRATICA
AND A1.DATAINIZIO < A2.DATAINIZIO
ordina_su_id_idpratica_datainizio_datafine
copia_flusso_ordinata_su_idpratica
copia_ordinata_su_idpratica
elimina_ridondanza
join_su_idpratica 2
passo_solo_campi_servono_dopo
la parte di fianco in giallo serve a fare:
WHERE NOT EXISTS (
SELECT * FROM VW_BID_ATT_PER_LEFT_TEMPIST_4\2 A3
WHERE A1.IDPRATICA = A3.IDPRATICA
AND A1.DATAINIZIO < A3.DATAINIZIO
AND A3.DATAINIZIO < A2.DATAINIZIO
);
filter rows 8
Filter rows 9
aggiungo_campo_flag=1
Select values 7 2
A1.datainizio>A3.datainizio
A3.datainizio>A2.datainizio
rilevo_campo_flag
campi_con_flag=null
Filter rows 10
set_datafine=datainiziosucc
Filter rows 12
Filter rows 11
filtra_campi 2
Copy rows to result
filtro_campi
Append streams 8
A2.datafine=null
datafineatt<datainiziosucc
set_datafine=datafineatto
ricostruire_flusso 2

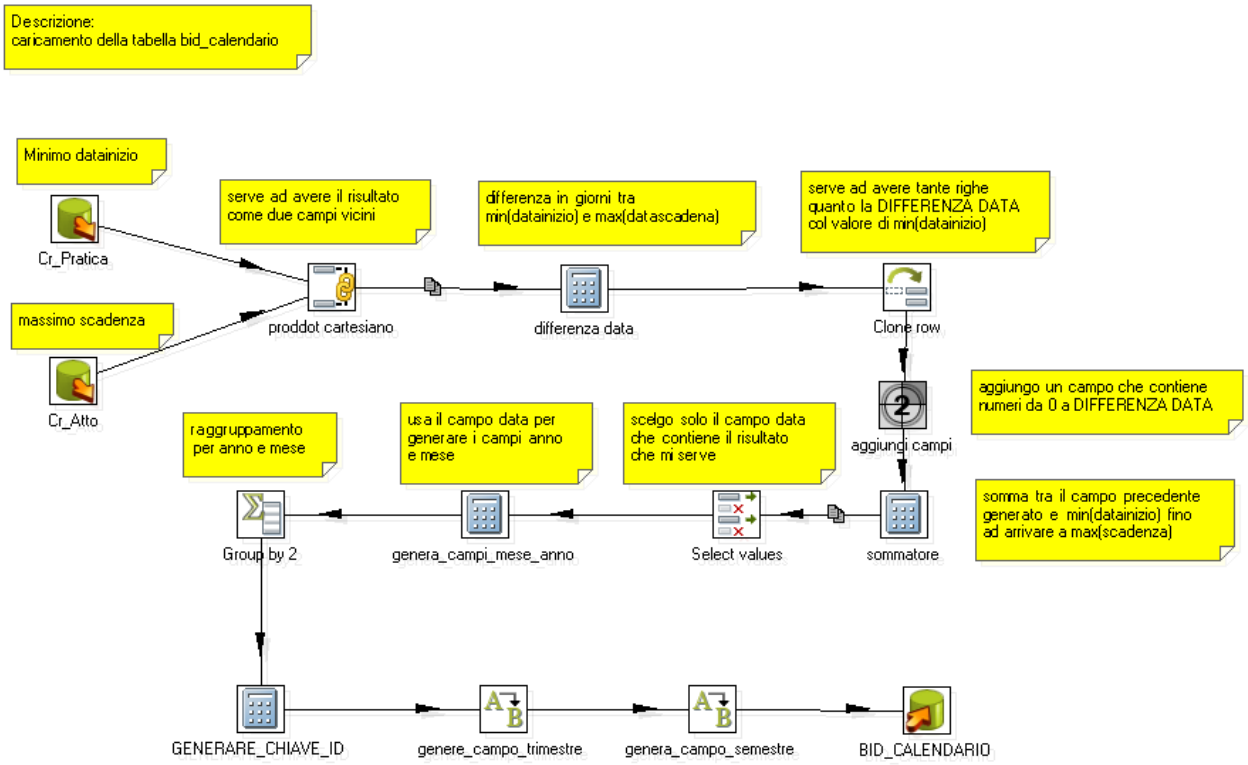

Descrizione:
caricamento della tabella bid_tmp_ggnonprovincia
Ultima Parte
Get rows from result
passo_idpratica_datainizio_datafine_solo
calcolo_datafine-datainizio
problema_febbraio
ordina_idpratica
bid_tmp_ggnonprovincia
calcola_somma_datafine-datainizio_raggruppando_su idpratica 2

Questa trasformazione legge 3 diverse tabelle e poi fa il join tra il risultato ottenuto usando le apposite chiavi, poi fa delle elaborazioni sulle date per calcolare le durate delle attività eliminando la durata tra l'inizio di un'attività e la fine di un'altra quando esse si sovrappongono per alla fine dare le tempistiche di ciascun'attività.
A questa trasformazione viene passato la prima volta un parametro con idresponsabile=2 e quindi calcola le tempistiche delle attività che sono state svolte da un soggetto diverso dal comune.
Inoltre, alla stessa trasformazione viene passato il parametro idresponsabile <> 4 che identifica che il soggetto che ha svolto l'attività è una provincia e quindi calcola le tempistiche per le attività svolte dalla provincia.

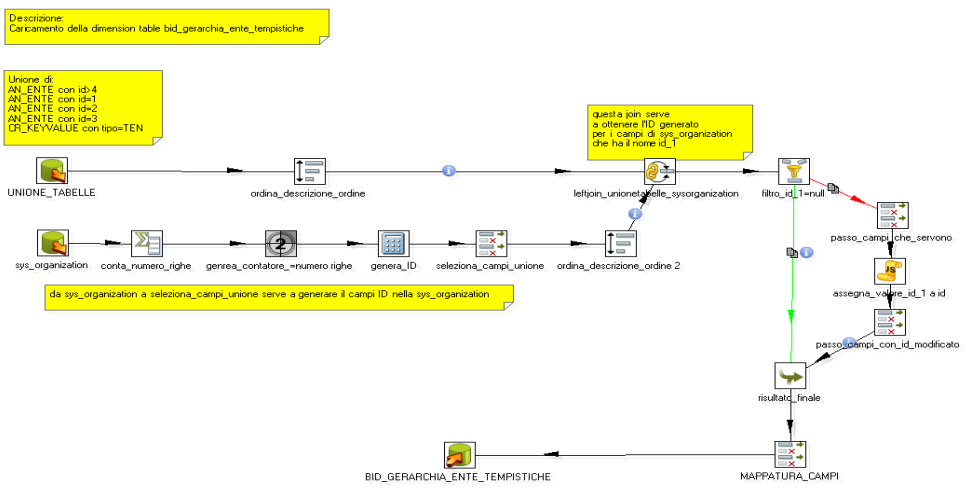
Dopodiché vengono caricate le tabelle dimensionali e di seguito viene riportato il JOB che fa questo lavoro:



Come si può vedere nell'immagine, ognuno dei quadratini gialli identifica le tabelle dimensionali di un cubo e poi c'è un quadrato che identifica le tabelle dimensionali comuni tra questi cubi che sono due, una per le provincie e l'altra per la dimensione tempo che viene utilizzata in tutti i cubi.
Le trasformazione del caricamento delle tabelle dimensionali sono semplici rispetto alle trasformazioni del caricamento delle tabelle dei fatti; di seguito riporto le immagini di alcune trasformazioni per il caricamento delle tabelle dimensionali per poi confrontarle con quelle delle tabelle dei fatti.

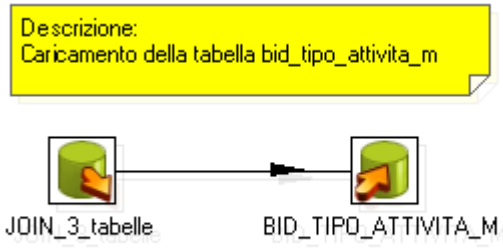


Questa trasformazione carica la tabella dimensionale BID_CALENDARIO che rappresenta la dimensione tempo in tutte le strutture realizzate dall'azienda, essa prende il massimo tra le date di scadenza degli atti e il minimo tra le date di inizio delle pratiche e genera dei record di cinque campi come segue: ID, anno, semestre, trimestre, mese.
ID è la chiave primaria della dimensione ed è composta dall'anno e dal mese. Ad esempio, se abbiamo una data 2009-07-16 allora ID sarà 200907, anno contiene l'anno cioè 2009, mese contiene il mese e poi semestre potrebbe essere S1 o S2 secondo il mese mentre il trimestre potrebbe essere T1,T2,T3,T4 secondo il mese nella data.
I record vengono generati partendo dal minimo della data di inizio delle pratiche e aggiungendo un mese alla volta fino ad arrivare al massimo della data scadenza degli atti.
Questi record servono a fare il raggruppamento dei risultati selezionati dalla tabella dei fatti e quindi vedere il risultato per anno, per mese, per semestre, per trimestre e quindi fare le operazioni di ROLL-UP e di DRILL-DOWN.
La seconda trasformazione di caricamento delle tabelle dimensionali che andremo a vedere è la seguente:



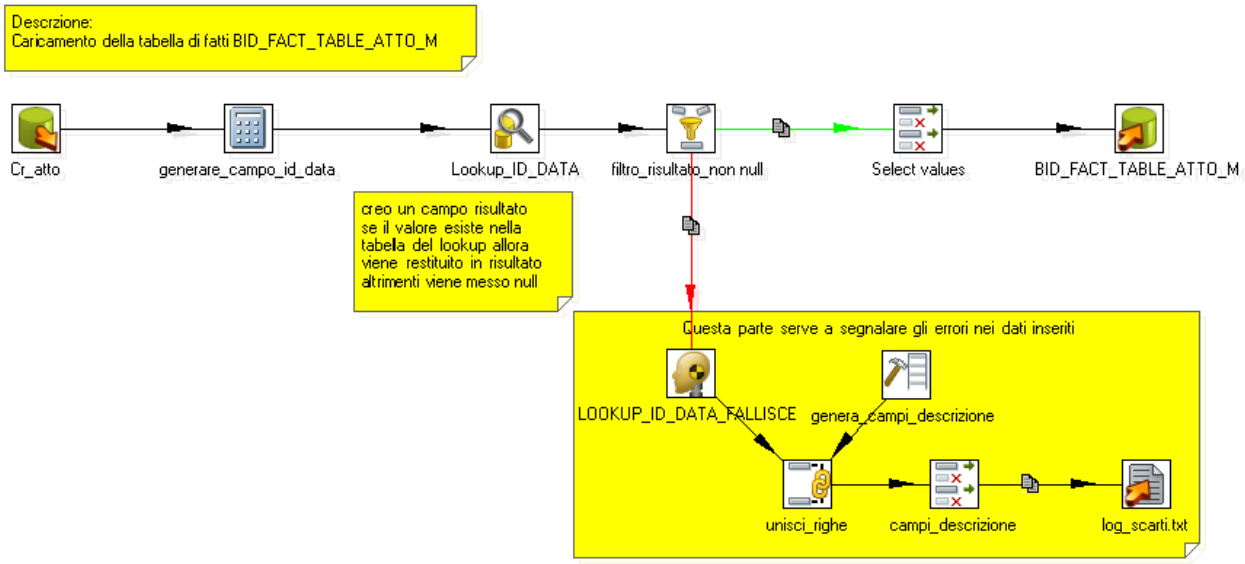
Questa trasformazione carica la dimensione GERARCHIA del cubo TEMPISTICHE e elabora i dati per generare ID modificato per i record secondo il valore del campo **id** originale di ognuno di essi.

L'ultima trasformazione di caricamento delle tabelle dimensionali che vedremo sarà il seguente:



Questa trasformazione carica la dimensione tipo attività del cubo attività e come si vede non fa altro che leggere da una tabella della base di dati transazionale del SIAM per caricarli in una tabella dimensionale del cubo di analisi del *data warehouse* e questo è il tipo delle trasformazioni più semplice che possono essere fatte.

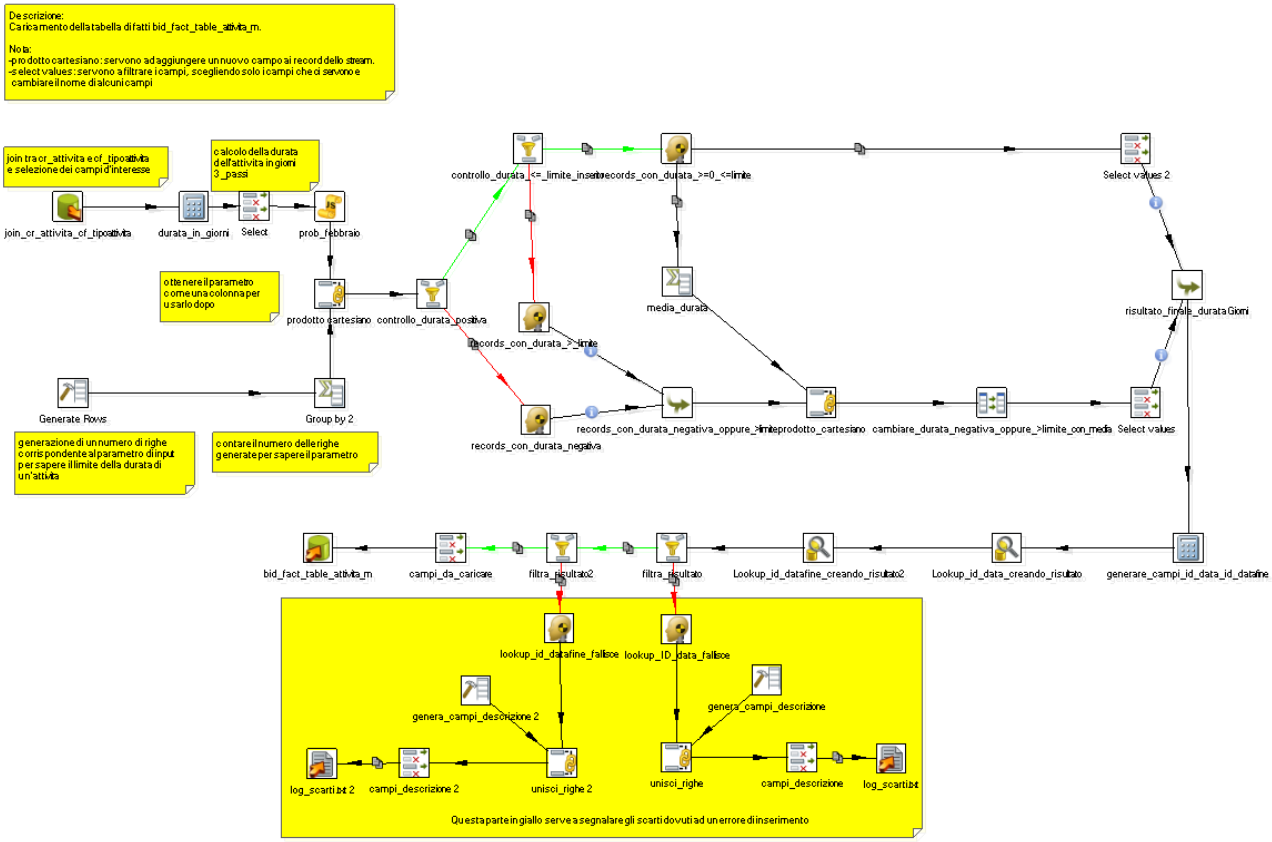
A questo punto possiamo dire che abbiamo visto tre diversi livelli di complessità per fare le trasformazioni di caricamento delle tabelle dimensionali e quindi possiamo passare alle trasformazioni delle tabelle dei fatti; cominciamo con quello degli atti.



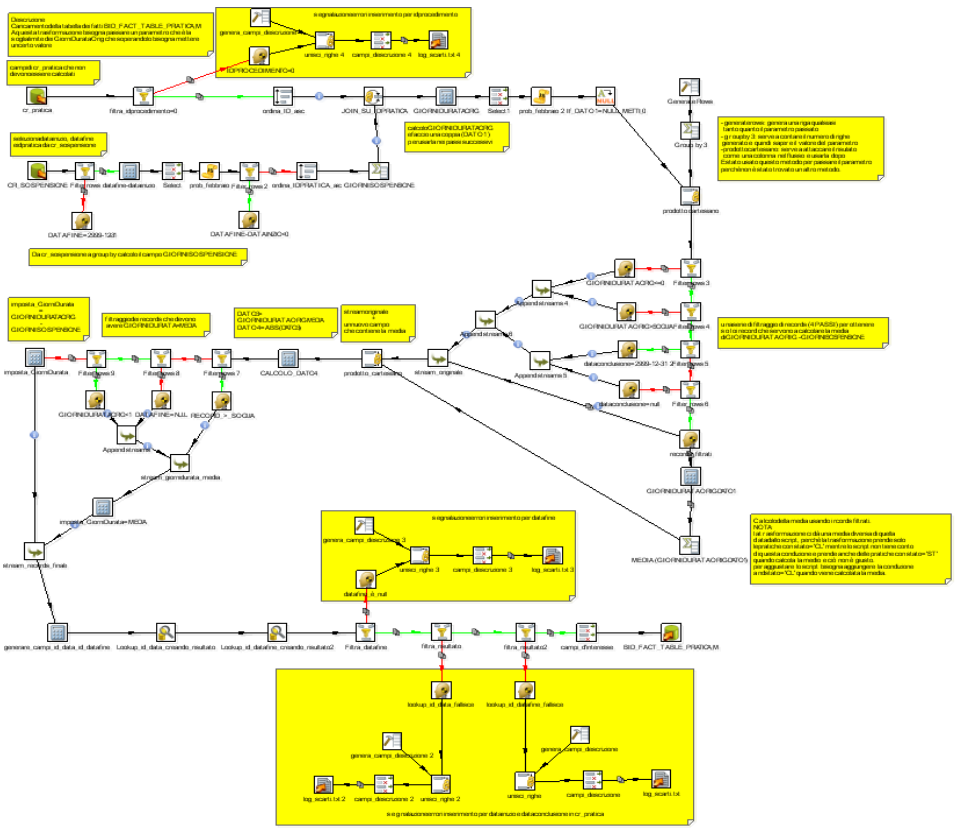
Questa trasformazione legge i dati da un tabella e genera il campo IDCALENDARIO e prima di caricare i dati nel cubo fa il *lookUp* nella dimensione tempo per verificare se tutti gli IDCALENDARIO si trovano nella dimensione tempo, altrimenti verrà scartato il record e verrà scritto in un file di scarti. Tale record viene segnalato via mail, assieme al nome del cubo, la descrizione dell'errore, la tabella sorgente da cui è stato letto il record e la chiave di ricerca nella tabella per trovarlo. Va osservato che questo tipo di errore succede spesso, a causa di un errore di inserimento delle date durante la fase di compilazione delle pratiche degli utenti del SIAM, e, di conseguenza è stato deciso che questo tipo di controllo è indispensabile sia per avere un risultato significativo e corretto dei *report* prodotti sia per aggiustare le date malamente inseriti nella base di dati transazionale del SIAM.

Le altre trasformazioni di caricamento dei cubi sono i seguenti:

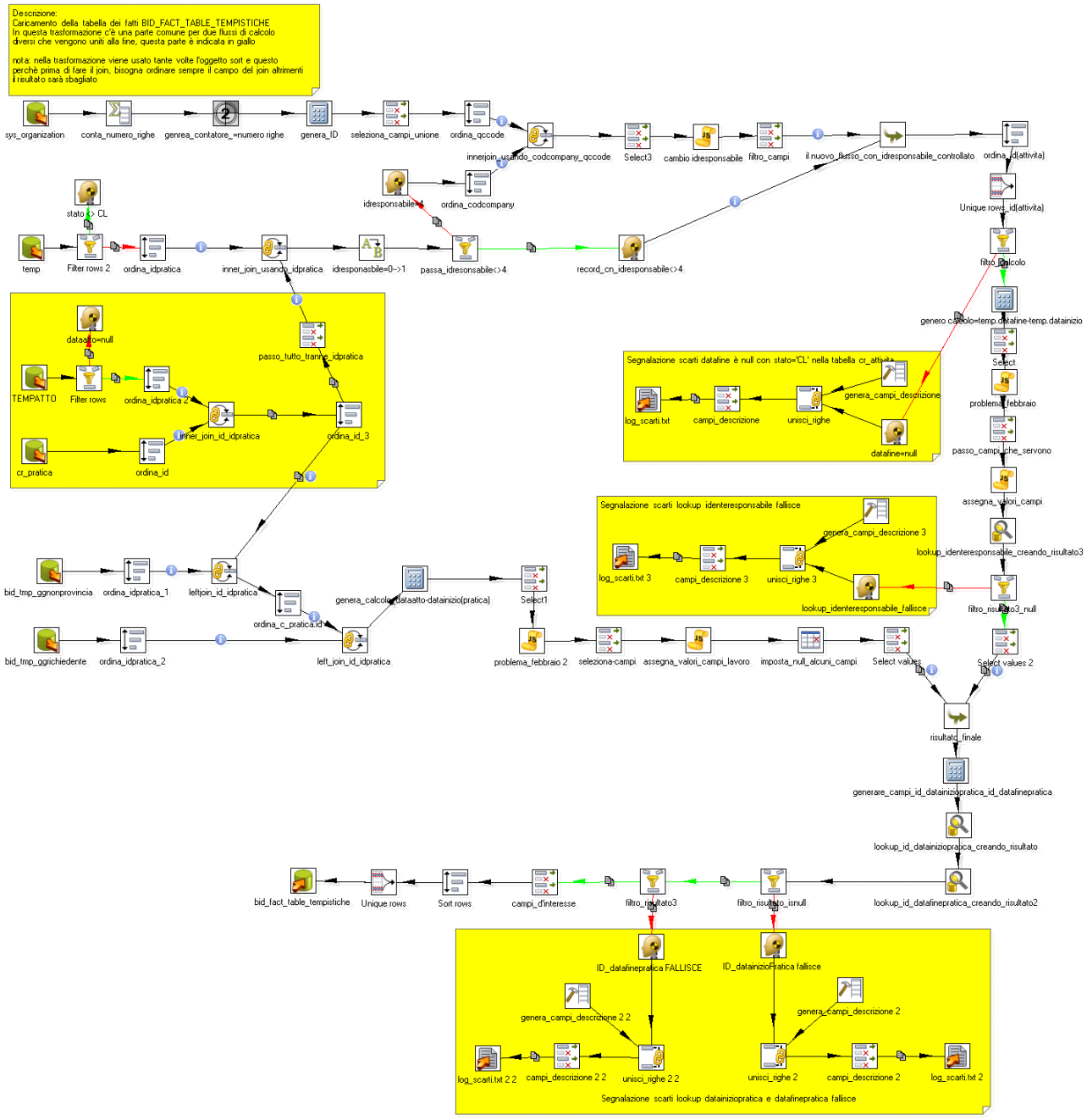
a. Per il cubo dell'attività:



b. Per il cubo delle pratiche:



c. Per il cubo delle tempistiche:



Quest'ultima è quella più complesse perché ci sono tanti elaborazioni e tanti controlli da fare con il segnalamento degli eventuali errori di inserimento.
Alla fine del caricamento delle tabelle dei fatti viene mandata la mail di notifica di fine elaborazione del flusso e degli eventuali scarti con gli eventuali errori.

4.2 Realizzazione del nuovo cubo

Dopo avere creato il flusso generale discusso nel paragrafo precedente, è stato definito insieme al cliente dell'azienda un nuovo progetto per la definizione di nuovi cubi in materia ambientale, per monitorare e controllare la "salute" dell'ambiente sul territorio e in particolare l'aria.
In questo modo è nato il progetto dello stage che verrà dettagliato in tutte le sue parti nei paragrafi successivi.

4.2.1 Requisiti del cliente

Per iniziare questa fase, ci sono state delle riunioni con il cliente per definire gli obbiettivi da raggiungere i *report* da creare, quindi il cliente ha chiesto di produrre i *report* seguenti:

1. Un *report* che visualizza per ogni **tipo inquinante la quantità autorizzata** per essere emessa in ogni **comune** per **anno**
2. Un *report* che visualizza per ogni **tipo materia prima la quantità autorizzata** per essere utilizzata nel processo di produzione in ogni **comune** per **anno**
3. Un *report* che visualizza per ciascun **stabilimento** di ogni **comune** la quantità degli inquinanti **autorizzati** per essere emessi di esso al **mese**
4. Un *report* che visualizza per ciascun **stabilimento**, **il quantitativo delle materie prime autorizzate** per essere utilizzate al **mese** da questo stabilimento
5. Un *report* che fa il confronto per un **anno** scelto tra **la concentrazione dell'inquinante** emesso rispetto alla **concentrazione autorizzata** di esso e rispetto alla **concentrazione emessa dichiarata dall'autocontrollo** periodico fatto dall'ente stesso

4.2.2 Analisi dei requisiti

Una volta che i requisiti del cliente sono stati definiti, si è passati alla fase di analisi di essi per definire la struttura del *data warehouse* da creare.

Dal primo *report* richiesto appare la necessità di avere quattro dimensioni e una misura per la tabella dei fatti. Questi dimensioni sono: **comune, inquinante, tipo autorizzazione e tempo**, mentre la misura è **Quantita_inquinante**. Inoltre, il primo *report* evidenzia una granularità della dimensione **tempo** fino al livello dell'anno.

Il secondo requisito evidenzia la necessità di avere una dimensione e una misura in più. La dimensione è **materia prima** mentre la misura è la **quantità delle materie prime** e la granularità della dimensione **tempo** è uguale a ciò che è stato definito prima.

Il terzo requisito evidenzia la necessità di una nuova dimensione da aggiungere che è **stabilimento**. Inoltre, questo *report* aggiunge anche una granularità più fine per la dimensione **tempo** che arriva fino al livello del mese.

Il quarto *report* richiesto non aggiunge né dimensioni né misure perché per realizzarlo basta utilizzare le dimensioni e le misure già definite però, è opportuno introdurre una nuova dimensione che è **punto emissione**. Infatti, ogni stabilimento potrebbe avere più di un punto d'emissione e, quindi, bisogna aggiungere questa dimensione per eventuale richieste di realizzare dei *report* che visualizzano anche i punti emissione, richiesta molto probabile in futuro.

Il quinto requisito aggiunge una nuova misura che è la **durata**. Questa misura calcola la durata dell'accensione del punto di emissione in un mese; la durata serve a calcolare la concentrazione dell'inquinante che è la media pesata delle emissioni in tale durata.
Inoltre, siccome le informazioni del DWH possono essere riferite a diversi enti, per una migliore visione dei dati, è stato necessario creare una nuova dimensione che è **company** che determina il nome dell'ente a cui fa capo il dato.

Quindi dopo l'analisi dei requisiti sono arrivato alla decisione di creare le seguenti dimensioni:

1. Inquinante
2. Tempo
3. Comune
4. Stabilimento
5. Company
6. Tipo Autorizzazione
7. PuntoEmissione
8. MateriaPrima

E le seguenti misure:

1. Durata
2. Quantita_Inquinante
3. Quantita_MateriaPrima

4.2.3 Progettazione e realizzazione dello schema a stella

Dopo avere definito le dimensione e le misure che deve avere il nuovo cubo, si passa alla fase della realizzazione di esso definendo gli attributi di ciascuna dimensione e della tabella dei fatti. Gli attributi risultanti sono i seguenti:

1. **Inquinante:** per questa dimensione sono stati definiti due livelli: **Categoria e Tipo** più la chiave primaria della tabella che è l'identificativo univoco di ciascun inquinante. Questi due livelli sono stati introdotti per poter raggruppare la dimensione dell'inquinate per categoria o per tipo d'inquinante e quindi fare le operazioni di ROLL-UP e di DRILL-DOWN utilizzando questi due livelli
2. **Tempo:** per la dimensione del tempo sono stati definiti quattro livelli che sono: **anno, semestre, trimestre e mese** più una chiave univoca che è composta dalla concatenazione dell'anno e del mese. I *report* richiesti hanno bisogno solo di due livelli, mentre questa dimensione ne ha quattro e questo perché è stato deciso di utilizzare una dimensione comune con gli altri cubi che erano stati realizzati prima del cubo in oggetto. Ciò ha il vantaggio di soddisfare altre esigenze future che hanno il requisito di raggruppare per semestre o per trimestre senza nessun bisogno di introdurre delle modifiche alle strutture delle dimensioni del nuovo cubo
3. **Comune:** questa dimensione è composta da due livelli che sono: **provincia e comune** più la chiave univoca che è composta da **codice provincia** concatenato con il **codice del comune** e che ha una lunghezza fissa di 6 cifre
   Anche questa tabella è una tabella che viene utilizzata da più cubi e per questo che serve il livello di provincia che viene utilizzato per la realizzazione dei *report* per gli altri cubi.
4. **Stabilimento:** questa dimensione ha due livelli che sono: **descrizione e descrizione_completa** più una chiave univoca che è l'identificativo dello stabilimento
5. **Company:** questa dimensione ha solo un livello: **name** che contiene il nome dell' azienda più l'identificativo univoco che è l'identificativo della stessa. Inoltre, questa dimensione è una dimensione che viene utilizzata anche dagli altri cubi

6. **TipoAutorizzazione:** questa dimensione ha un solo livello che è il **Tipo** che descrive il tipo di dato raccolto nel gestionale. Il tipo è riferito alla fase autorizzativa, fase di controllo, piuttosto che fase di autocontrollo. Nella tabella della dimensione viene riportata la descrizione della fase
7. **PuntoEmissione:** la seguente dimensione ha due livelli: **descrizione e descrizione_completa** più la chiave univoca che identifica il punto Emissione e che è composta dall'identificativo dello stabilimento concatenato con il numero del punto d'emissione
8. **MateriaPrima:** questa dimensione ha un livello che è il **Tipo** più la chiave univoca che identifica la materia prima

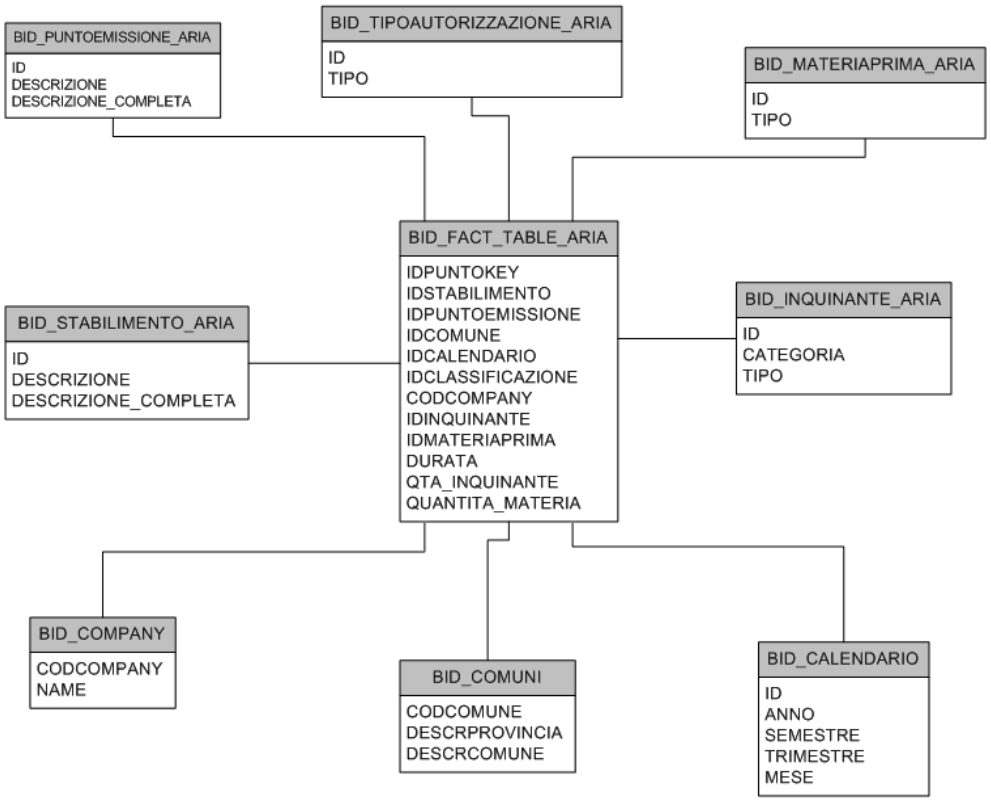
Per quanto riguarda la tabella dei fatti, essa dovrebbe contenere 8 attributi che sono:

1. **IDINQUINANTE:** cha fa riferimento alla chiave della dimensione **INQUINANTE**
2. **IDCALENDARIO.** Che fa riferimento alla dimensione **TEMPO**
3. **IDCOMUNE:** che fa riferimento alla dimensione **COMUNI**
4. **IDSTABILIMENTO:** che fa riferimento alla dimensione **STABILIMENTO**
5. **CODCOMPANY:** che fa riferimento alla dimensione **COMPANY**
6. **IDCLASSIFICAZIONE:** che fa riferimento alla dimensione **TipoAutorizzazione**
7. **IDPUNTOEMISSIONE:** che fa riferimento alla dimensione **PUNTOEMISSIONE**
8. **IDMATERIAPRIMA:** che fa riferimento alla dimensione **MATERIAPRIMA**
9. **IDPUNTOKEY:** non fa riferimento a nessuna tabella dimensionale, ma è necessario per risalire al dato del gestionale elaborato nella trasformazione. Quindi, nel caso ci sia un errore di inserimento, questo attributo facilita la ricerca dell'errore e quindi aggiustarlo nella base di dati transazionale del SIAM.

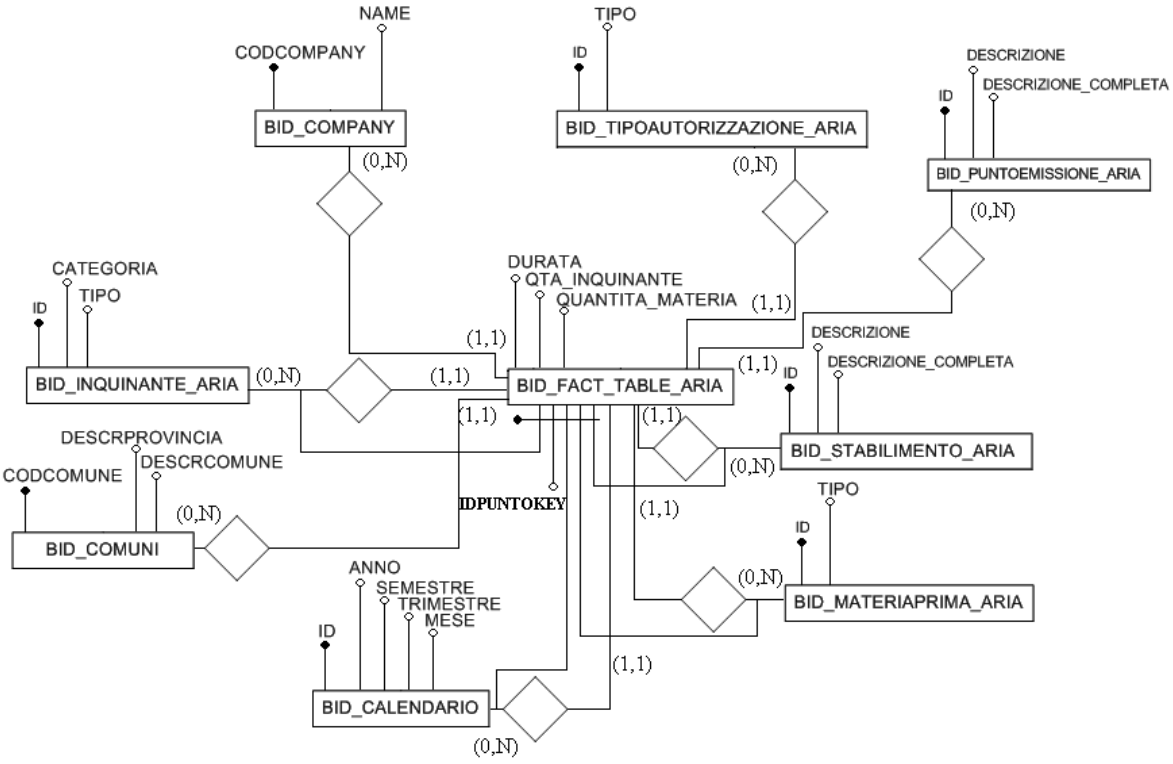
Questi attributi sono le chiavi primarie delle tabelle dimensionali già definite prima per permettere il collegamento tra le tabelle dimensionali e la tabella dei fatti che dovrebbe far riferimento sempre ai dati contenuti nelle tabelle dimensionale e ciò permette di evitare l'inserimento dei dati che non servono oppure dati sbagliati. Ad esempio, se abbiamo un record con **IDCALENDARIO= 299912** da inserire nella tabella dei fatti e questo ID non c'è nella dimensione **tempo** allora c'è stato un errore nell'inserimento della data e quindi bisogna segnalare questo errore.

Inoltre, la tabella dei fatti dovrebbe contenere tre misure come è stato detto prima per soddisfare le richieste del cliente di realizzare questi *report*. Se, nel futuro, il cliente dovesse chiedere dei *report* che non possano essere soddisfatte dalle dimensioni e dalle misure introdotte , allora bisogna modificare la struttura del cubo introducendo delle nuove dimensioni o delle misure o anche entrambe.

L'analisi fatta conduce allo schema a stella seguente:



Come possiamo vedere nella foto sopra riportata ogni dimensione (tabella dimensionale) contiene gli attributi che definiscono i suoi livelli e la chiave primaria che viene utilizzata come una *foreign key* per collegare la dimensione alla tabella dei fatti.
Nella tabella dei fatti vediamo le *foreign key* e le misure che sono state aggiunti per riuscire a creare i *report* definiti dal cliente.
Lo schema ER di questo schema a stella è il seguente:

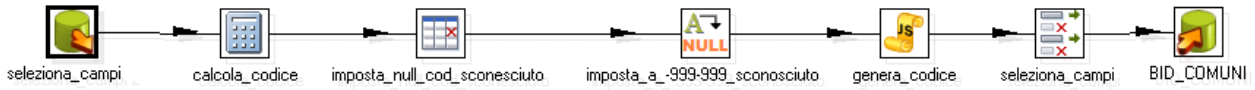


Siccome questo cubo dovrebbe essere installato sulle quattro basi di dati che abbiamo citato prima, allora sono stati create quattro versioni di script per la creazione di questa struttura dove ognuno di essi corrisponde a un RDBMS.

4.2.4 Alimentazione dei dati

Adesso che abbiamo la struttura del cubo con le tabelle create si passa a caricarle prendendo i dati dalla base di dati del SIAM che contiene i dati transazionali usando *KETTLE*. Di seguito verranno discusse tutte le trasformazioni realizzate per il caricamento.
Nella fase di alimentazione di dati si comincia con il caricamento delle tabelle dimensionali e poi si procede con il caricamento della tabella dei fatti e questo a causa dell'impossibilità di caricare la *fact table* prima e poi le tabelle dimensionali perché lo strumento darà un errore a causa del riferimento dei campi della tabella dei fatti alle chiavi delle tabelle dimensionali che saranno vuote.

Caricamento della BID_COMUNI:

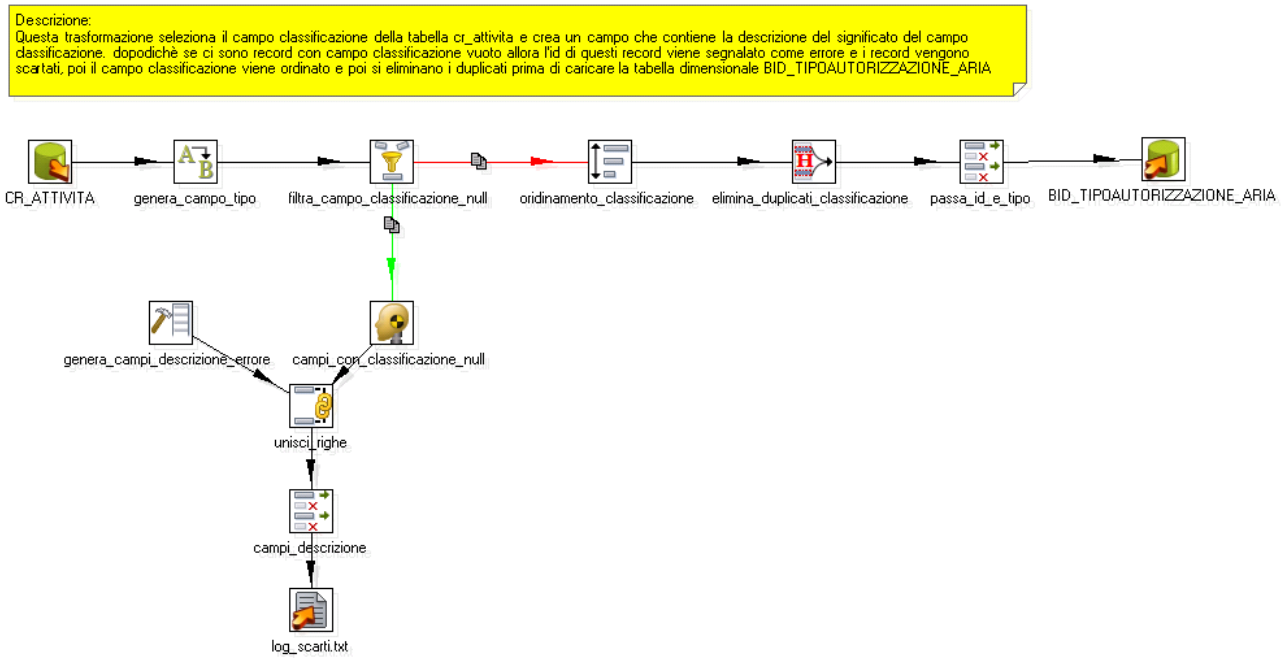


Questa trasformazione carica la tabella dimensionale BID_COMUNI facendo l'unione di tre tabelle e selezionando il **codice provincia** con la sua descrizione e il **codice comune** con la sua descrizione dal risultato del join di queste tre tabelle, quindi si procede a calcolare il primo campo della tabella che è anche la chiave. Questo campo è **codcomune** che dovrebbe essere fatto dalla concatenazione del codice provincia e del codice comune e dovrebbe essere di lunghezza fissa di sei cifre, se è conosciuto, quindi se la concatenazione dei due dà un risultato meno di sei cifre, vengono aggiunti degli zeri per arrivare al codice. Inoltre, se la provincia e il comune sono sconosciute allora il codice sarà -999-999.
Per quanto riguarda gli altri due campi della tabella, il campo **descrprovincia** sarà semplicemente uguale al campo denominazione della provincia scelta prima mentre il campo **descrcomune** sarà il campo denominazione del comune.
Dopo che il **codcomune** è stato calcolato e i campi sono pronti, essi vengono caricati nella tabella.

Caricamento della BID_PUNTOEMISSIONE_ARIA:

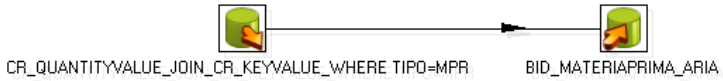
Per caricare questa tabella vengono selezionati i campi necessari dalle tabelle **aria_puntoemissione**, **an_stabilimento, comuni e cr_keyvalue** del SIAM e poi vengono costruiti i campi della tabella BID_PUNTOEMISSIONE_ARIA. Il campo chiave ID sarà costruito dalla concatenazione dei campi IDSTABILIMENTO e numero della tabella **aria_PuntoEmissione**. Il campo descrizione sarà uguale al campo numero della tabella precedente mentre l'ultimo campo sarà ottenuto dalla concatenazione dei campi restanti e poi dal risultato vengono eliminati i record doppi e quindi si procede al caricamento della tabella.

Caricamento della BID_TIPOAUTORIZZAZIONE_ARIA:



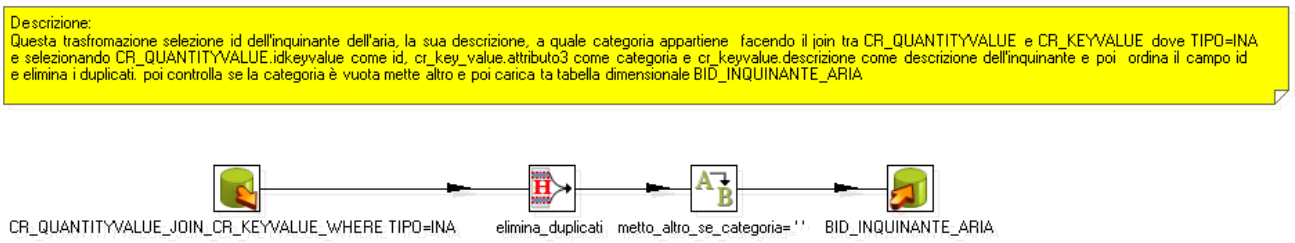
Questa trasformazione sceglie i campo ID e classificazione della tabella **CR_attivita** e poi secondo il campo dì classificazione genera il campo TIPO.
Se si trovano record con campo classificazione vuoto allora questi record vengono scartati e viene segnalato l'errore di inserimento con l'ID che viene utilizzato per la ricerca rapida del record.
Dopodiché vengono eliminati i record duplicati e quindi vengono caricati nella tabella.

Caricamento della BID_MATERIAPRIMA_ARIA:



Questa trasformazione fa il join tra le tabelle CR_QUANTITYVALUE e CR_KEYVALUE del SIAM dove TIPO = 'MPR', seleziona i campi CR_QUANTITYVALUE.idkeyvalue come ID e cr_keyvalue.descrizione come TIPO e li carica nella tabella.

Caricamento della BID_INQUINANTE_ARIA:

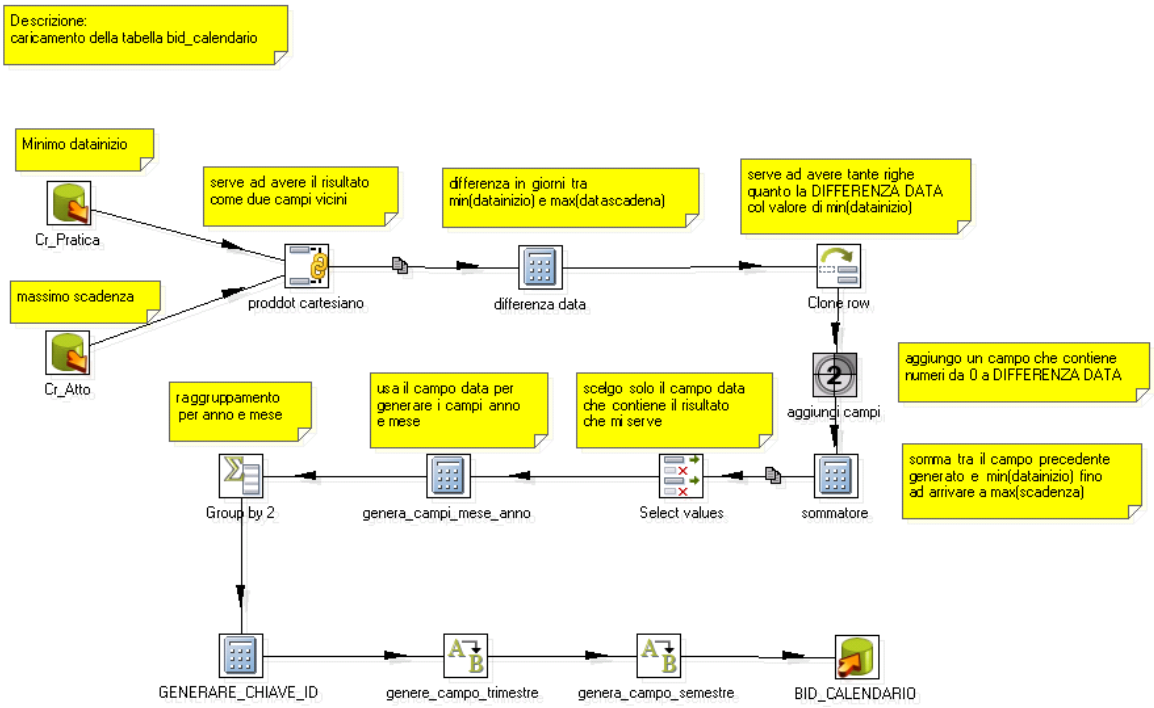


Questa trasformazione è uguale alla trasformazione precedente con il TIPO = 'INQ' e sceglie un campo in più che il campo CR_keyvalue.attributo3 che sarà il campo categoria della tabella dimensionale. Dopodiché elimina i record duplicati e mette il valore **altro** se trova un valore del campo categoria vuoto e quindi li carica nella tabella.

Caricamento della BID_STABILIMENTO_ARIA:

Questa trasformazione carica la tabella dimensionale BID_STABILIMENTO_ARIA che è fatta da tre campi che sono:
id = idstabilimento, Descrizione = comune + via, Descrizione_completa = ragione sociale + forma giuridica + comune + indirizzo (tipovia+nomevia+numero_civico) quindi abbiamo bisogno di diversi campi che si trovano in 7 tabelle.
si parte a selezionare i campi che servono facendo il join tra le tabelle che contengono questi campi e il risultato viene ordinato sull' **idstabilimento** e **datainizio dell'oggetto impresa** in ordine discendente, poi si fa il concatenamento dei diversi campi per ottenere il formato desiderato dei 3 campi del risultato e poi vengono eliminati gli ID duplicati, e siccome il risultato era ordinato su ID e **datainizio** discendente, allora rimane solo l'ultima ragione sociale delle società che hanno cambiato la ragione con il tempo e quindi il risultato viene caricato nella tabella dimensionale.

Caricamento della BID_CALENDARIO:

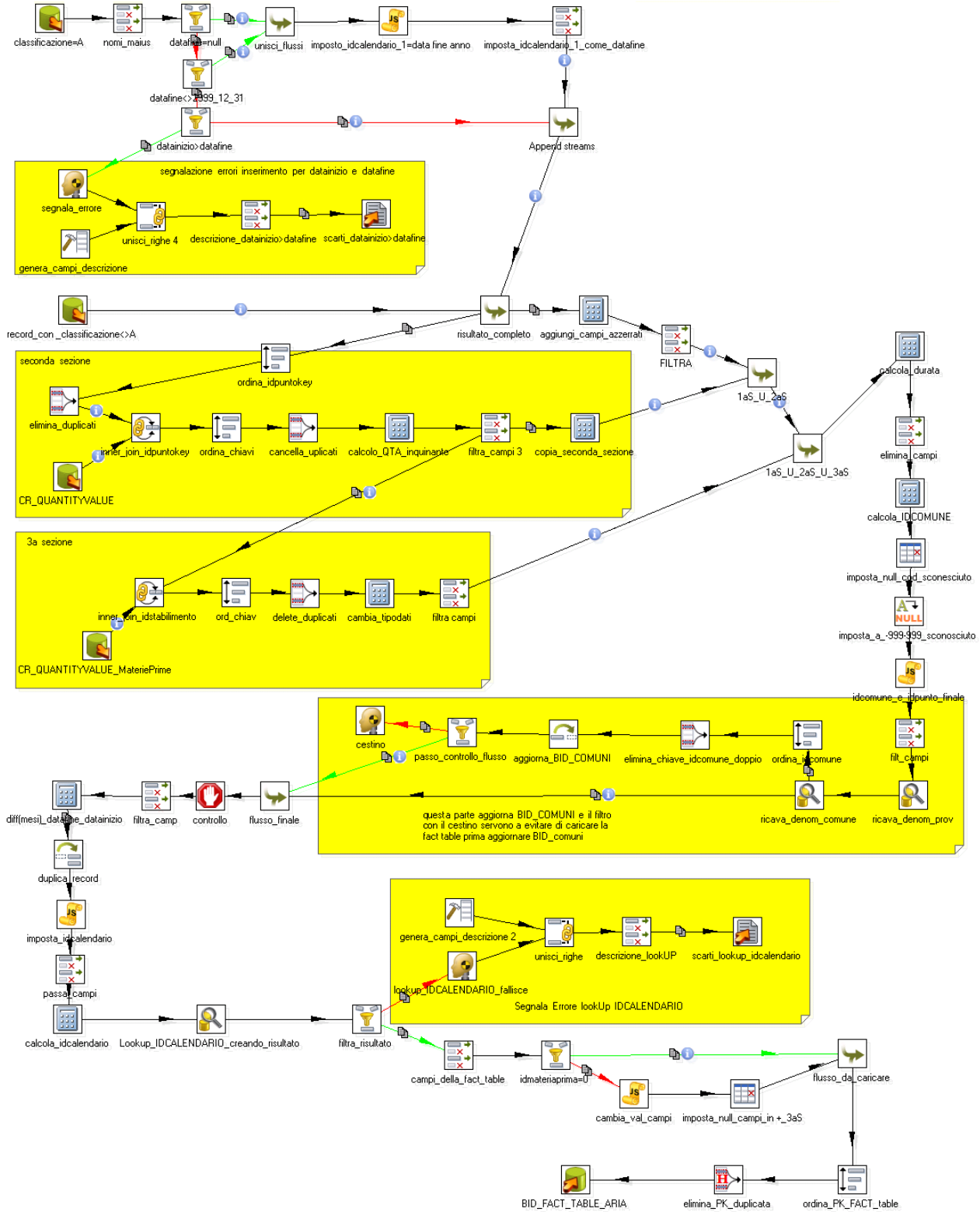


Questa trasformazione è stata discussa nel paragrafo **4.1 Creazione della trasformazione.**

Caricamento della BID_COMPANY:

Questa trasformazione consiste due campi **qccode** e **qcname** dalla tabella **sys_organization** del SIAM, cambiare il loro nome e caricali nella tabella dimensionale BID_COMPANY.

Caricamento della tabella dei fatti BID_FACT_TABLE_ARIA:



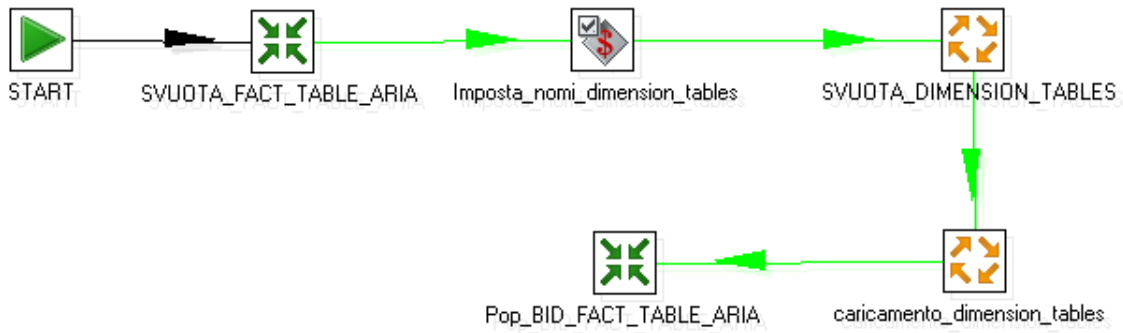
Alla fine si arriva al caricamento della tabella dei fatti che è composta da tre sezione. La prima sezione contiene tutti i campi definiti nella struttura del cubo con IDINQUINANTE, IDMATERIAPRIMA, QUANTITA_MATERIA, QTA_INQUINANTE posti a **null** perché questa sezione dovrebbe rispondere ai *report* che riguardano i punti emissione.

La seconda sezione contiene tutti i campi pieni tranne i due campi IDMATERIAPRIMA e QUANTITA_MATERIA perché questa sezione sarà utilizzata per rispondere ai *report* che riguardano solo gli inquinanti, mentre la terza sezione avrà i campi IDPUNTOEMISSIONE, IDINQUINANTE, QTA_INQUINANTE, IDCLASSIFICAZIONE, e DURATA posti a NULL perché questa sezione deve rispondere ai *report* che riguardano solo la materia prima.
Comunque è stato scelto di utilizzare questo metodo perché è facile fare i calcoli dei campi e preparali con lo strumento ETL che farlo quando si formulano le query MDX.

La trasformazione parte con il calcolo dalla prima sezione, poi si prende una copia e si fa il join con i dati necessari presi dalla tabella CR_QUANTITYVALUE che sono IDPUNTOKEY, IDINQUINANTE e FLUSSO e ciò serve a fare il join tra la sezione già calcolata e questi dati per ottenere i dati riguarda l'inquinante e calcolare la quantità e per evitare di calcolare altri campi che sono stati calcolati nella prima sezione, quindi avremo una seconda sezione uguale alla prima con la differenza che **idinquinante** e **qta_inquinante** non sono vuoti (NULL). Il risultato ottenuto per la seconda sezione viene copiato e poi si fa uno procedimento identico a quello di prima per calcolare la terza sezione.
Il risultato delle tre sezioni viene unito e vengono generati gli altri campi utilizzando i campi del risultato. Si comincia con il calcolo della **durata** e poi IDCOMUNE e poi IDPUNTOEMISSIONE e poi si ricava la denominazione della provincia e del comune utilizzando i campi ST_CODPROV e ST_CODCOMUNE. Se si trova un IDCOMUNE che non c'è nella tabelle dimensionale ma c'è nella tabelle dei fatti, allora, la *dimension table* BID_COMUNI viene aggiornata inserendo il record che manca. Dopodiché, si fa il *lookup* della IDCALENDARIO e viene segnalato l'errore di *lookUp* se c'è. IDCALENDARIO viene generato partendo da DATAINIZIO e aggiungendo un mese alla volta fino ad arrivare alla DATAFINE.
Nella trasformazione si trovano anche dei blocchi per la segnalazione dell'errore come era già fatto nelle altre trasformazioni che caricano le altre tabelle dei fatti degli altri cubi.

Il flusso completo del caricamento delle tabelle del cubo ARIA:

Adesso che tutte le trasformazioni sono pronte, esse vengono integrate tutte insieme per gestire la dipendenza tra di loro cioè lo svuotamento della tabella dei fatti prima e poi delle tabelle dimensionali e il caricamento delle tabelle dei fatti e poi della tabella dimensionale; è quindi risultato il flusso seguente:



Questo flusso carica solo il cubo ARIA ma dopo viene integrato con il flusso generale creato in precedenza per fare il caricamento di tutti i cubi insieme gestendo la dipendenza tra di loro in un unico flusso.

4.2.5 Creazione del cubo

A questo punto la struttura del cubo è stata creata e i dati sono stati caricati, dopo aver fatto tutte le operazione dell'estrazione dalla base di dati del SIAM, la trasformazione dei dati generando dati calcolati e il caricamento di essi nella struttura, passiamo alla mappatura dello schema a stella nel server OLAP *Mondrian* per creare il cubo ARIA.
Di seguito viene riportato il file XML che genera il cubo ARIA facendo la mappatura della struttura relazionale creata ovvero lo schema a stella:

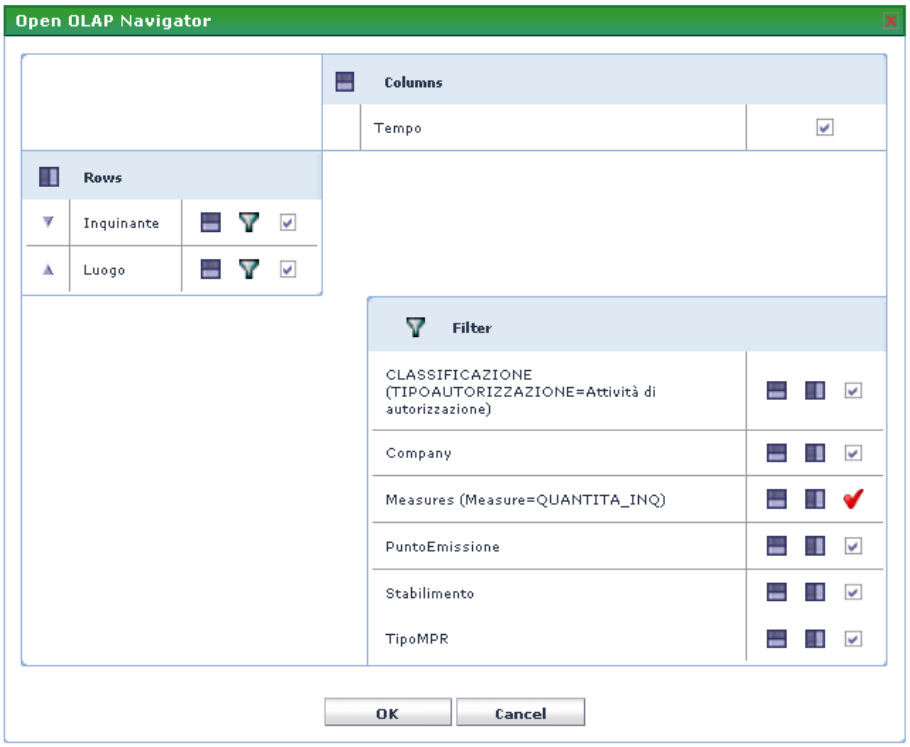
```
<?xml version="1.0" encoding="UTF-8" ?>
- <Schema name="Aria">
  - <Cube name="Aria">
      <Table name="BID_FACT_TABLE_ARIA" />
    - <Dimension foreignKey="CODCOMPANY" name="Company">
      - <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="CODCOMPANY">
          <Table name="BID_COMPANY" />
          <Level column="NAME" name="COMPANY" type="String" uniqueMembers="true" />
        </Hierarchy>
      </Dimension>
    - <Dimension foreignKey="IDCALENDARIO" name="Tempo">
      - <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="ID">
          <Table name="BID_CALENDARIO" />
          <Level column="ANNO" name="ANNO" type="Numeric" uniqueMembers="true" />
          <Level column="SEMESTRE" name="SEMESTRE" type="String" uniqueMembers="false" />
          <Level column="TRIMESTRE" name="TRIMESTRE" type="String" uniqueMembers="false" />
          <Level column="MESE" name="MESE" type="Numeric" uniqueMembers="false" />
        </Hierarchy>
      </Dimension>
    - <Dimension foreignKey="IDCOMUNE" name="Luogo">
      - <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="CODCOMUNE">
          <Table name="BID_COMUNI" />
          <Level column="PROVINCIA" name="PROVINCIA" type="String" uniqueMembers="true" />
          <Level column="COMUNE" name="COMUNE" type="String" uniqueMembers="false" />
        </Hierarchy>
      </Dimension>
    - <Dimension foreignKey="IDPUNTOEMISSIONE" name="PuntoEmissione">
      - <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="ID">
          <Table name="BID_PUNTOEMISSIONE_ARIA" />
          <Level column="DESCRIZIONE" name="BREVE_DESCRIZIONE" type="String" uniqueMembers="true" />
          <Level column="DESCRIZIONE_COMPLETA" name="ESTESA_DESCRIZIONE" type="String" uniqueMembers="false" />
        </Hierarchy>
      </Dimension>
    - <Dimension foreignKey="IDCLASSIFICAZIONE" name="CLASSIFICAZIONE">
      - <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="ID">
          <Table name="BID_TIPOAUTORIZZAZIONE_ARIA" />
          <Level column="TIPO" name="TIPOAUTORIZZAZIONE" type="String" uniqueMembers="true" />
        </Hierarchy>
      </Dimension>
    - <Dimension foreignKey="IDMATERIAPRIMA" name="TipoMPR">
      - <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="ID">
          <Table name="BID_MATERIAPRIMA_ARIA" />
          <Level column="TIPO" name="TIPOMATERIAPRIMA" type="String" uniqueMembers="true" />
        </Hierarchy>
      </Dimension>
    - <Dimension foreignKey="IDINQUINANTE" name="Inquinante">
      - <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="ID">
          <Table name="BID_INQUINANTE_ARIA" />
          <Level column="CATEGORIA" name="CATEGORIA" type="String" uniqueMembers="true" />
          <Level column="TIPO" name="TIPO" type="String" uniqueMembers="false" />
        </Hierarchy>
      </Dimension>
    - <Dimension foreignKey="IDSTABILIMENTO" name="Stabilimento">
      - <Hierarchy allMemberName="Tutte" hasAll="true" primaryKey="ID">
          <Table name="BID_STABILIMENTO_ARIA" />
          <Level column="DESCRIZIONE" name="BREVE_DESCRIZIONE" type="String" uniqueMembers="true" />
          <Level column="DESCRIZIONE_COMPLETA" name="ESTESA_DESCRIZIONE" type="String" uniqueMembers="false" />
        </Hierarchy>
      </Dimension>
      <measure aggregator="SUM" column="DURATA" formatString="Standard" name="DURATA" />
      <Measure aggregator="SUM" column="QTA_INQUINANTE" formatString="Standard" name="QUANTITA_INQ" />
      <Measure aggregator="SUM" column="QUANTITA_MATERIA" formatString="Standard" name="QUANTITA_MPR" />
    - <CalculatedMember name="CONCETRAZIONE_ORARIA_INQ" dimension="Measures" visible="true">
        <Formula>[Measures].[QUANTITA_INQ] / [Measures].[DURATA] * 3600</Formula>
        <CalculatedMemberProperty name="FORMAT_STRING" value="##0.00" />
      </CalculatedMember>
  </Cube>
</Schema>
```

Nella figura riportata sopra, il nome del cubo viene dichiarato e viene mappato con la tabella dei fatti dello schema a stella come è indicato nei quadrati verdi.
Dopodiché le dimensioni vengono dichiarate facendo il collegamento tra la tabella dei fatti e la dimensione utilizzando la *foreign key* nella *fact table* e la chiave primaria nella *dimension table* e poi, per ogni dimensione, vengono dichiarati i livelli che determinano il livello della granularità dei dati facendo la mappatura col campo della tabella dimensionale, e dandogli il nome che dovrebbe essere utilizzato perché sia chiamato nelle query MDX.
A questo punto si passa alla dichiarazione delle misure, e ciò viene fatto come è evidenziato nei quadrati neri.
Per ogni misura, viene dichiarato l'aggregatore e viene fatta la mappatura con un campo della tabella dei fatti di cui bisogna prendere i dati che rispondono a questa misura e, alla fine, le viene dato il nome che dovrebbe essere utilizzato nelle chiamate di questa misura nelle query MDX.
L'ultima misura, dentro il quadrato in viola, viene dichiarata come un membro calcolato e le viene dato un nome che è CONCENTRAZIONE_ORARIA_INQ che non è altro che il risultato tra la divisione della quantità dell'inquinante e la durata trasformata in ore anziché in secondi.
Per i dettagli su come viene fatta la mappatura dello schema a stella nel server OLAP *Mondrian* generando il file XML e su come viene utilizzato BART per la dichiarazione del cubo si rimanda ai paragrafi **3.2.2. Mondrian-JPIVOT. e 3.2.4 Utilizzo di BART.**

4.2.6 Interrogazione del cubo e *report* di analisi

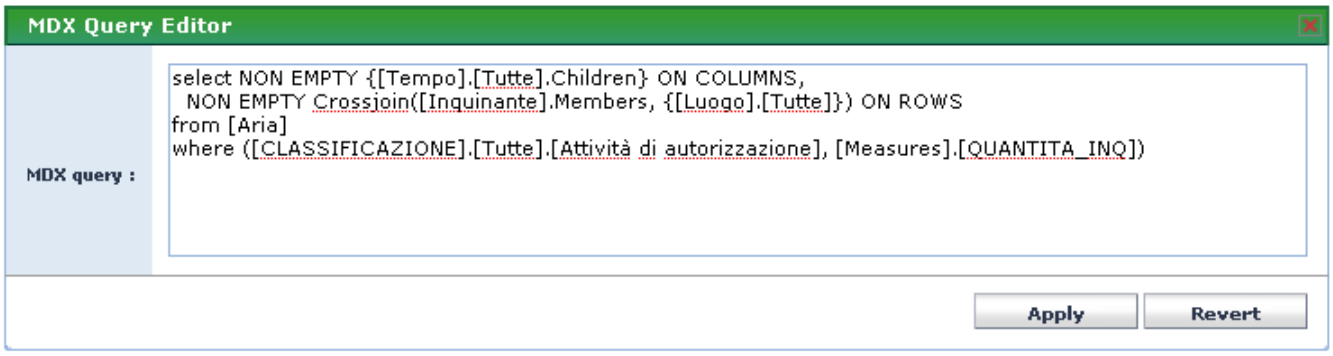
A questo punto, il cubo è pronto per essere interrogato e quindi viene utilizzato il linguaggio MDX per formulare le query di interrogazione che servono a visualizzare i *report* che ha chiesto il cliente.
Di seguito vengono riportati le query fatte e il *report* prodotto da essa e viene spiegato come i *report* rispondono ai requisiti del cliente.
Per realizzare i *report*, è stato utilizzato il *Modeller* di BART come è stato spiegato nel Paragrafo **3.2.4 UTILIZZO DI BART** e il risultato è il seguente:

Il primo *report*: per creare questo *report* viene utilizzato il *Modeller* di QUIXBI e dopo aver fatto le scelte viene visualizzata l'immagine seguente:



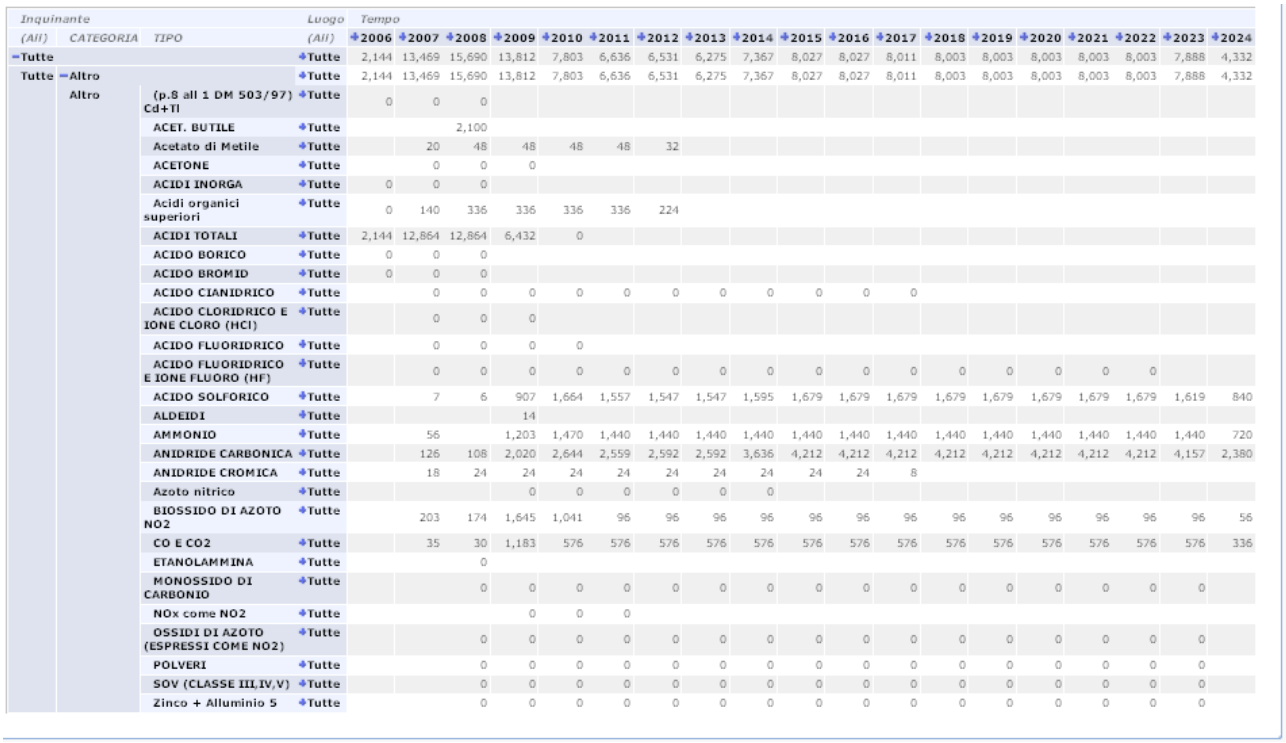
Il primo *report* ha l'obiettivo di visualizzare la quantità di ogni inquinante autorizzata per essere emessa per ogni **comune** per **anno**. Quindi, come si può vedere nell'immagine sopra riportata, sulle righe è stato scelto di visualizzare gli inquinanti e i comuni mentre sulle colonne è stato scelto di visualizzare la dimensione del tempo. Inoltre, è stato scelto di visualizzare la misura QUANTITA_INQ; per la classificazione viene scelto solo il **tipo = attività di autorizzazione** che identifica la quantità autorizzata dell'inquinante.

Utilizzando il *Modeller* e facendo le scelte descritte sopra la query MDX generata è la seguente:



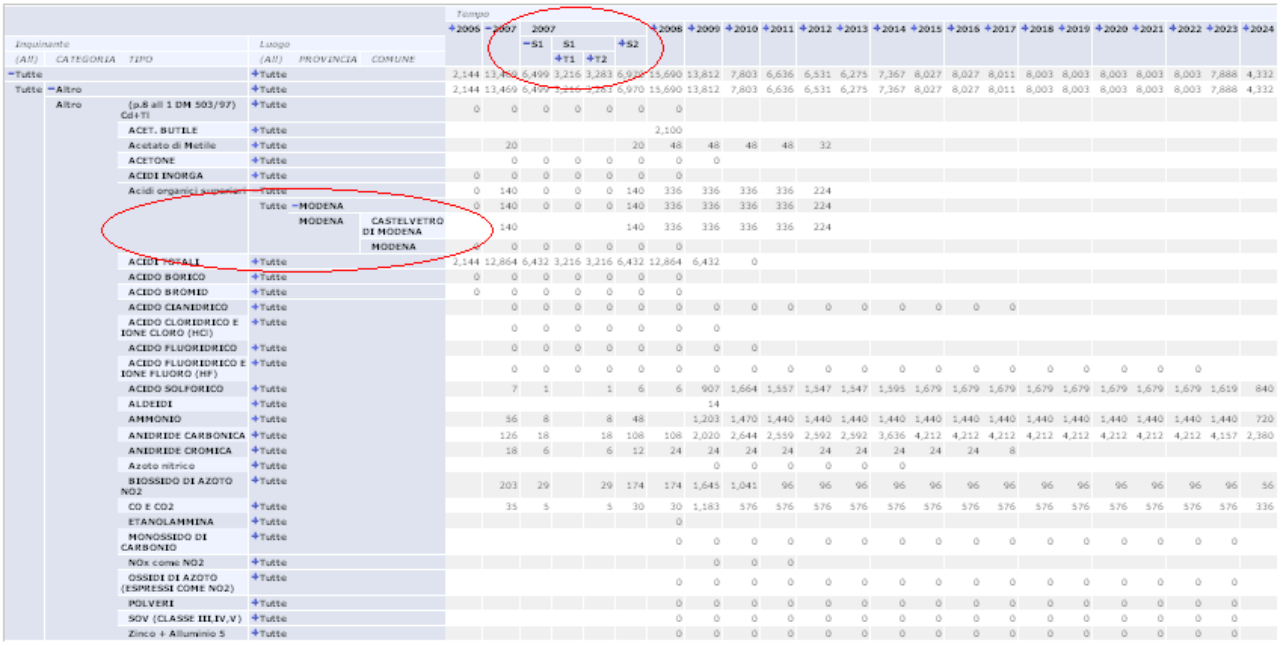
Come si può vedere, viene selezionata la dimensione del tempo sulle colonne con tutti i suoi livelli per permettere di fare le operazioni di ROLL-UP e DRILL-DOWN quando viene visualizzato il *report* e inoltre vengono selezionate solo le colonne che hanno dati eliminando le colonne non significative.
Sulle righe invece, vengono selezionate le dimensioni **inquinante** e **luogo** facendo il cross join tra di loro mentre nella condizione viene selezionata solo la misura QUANTITA_INQ e la classificazione di tipo attività di autorizzazione per filtrare il risultato visualizzando solo la quantità degli inquinanti autorizzata realizzando cosi l'operazione di SLICE.

Infine, la query descritta sopra fa vedere il seguente risultato:



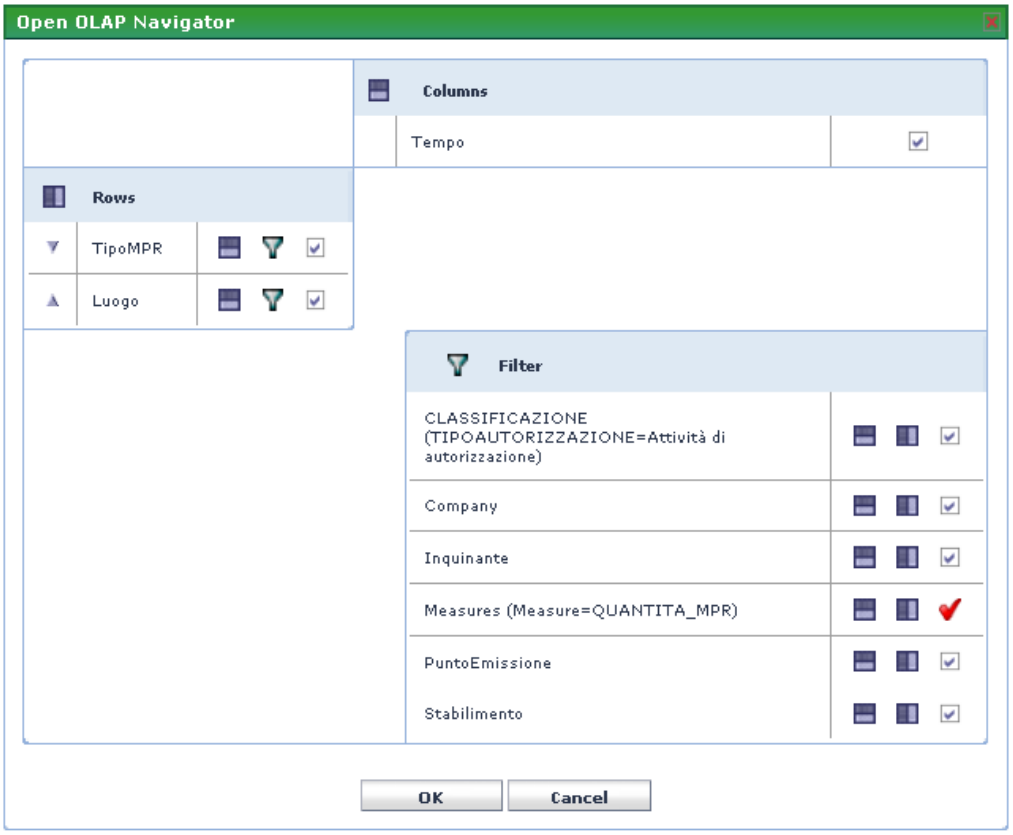
Inquinante			Luogo	Tempo																			
(All)	CATEGORIA	TIPO	(All)	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024
Tutte			Tutte	2,144	13,469	15,690	13,812	7,803	6,636	6,531	6,275	7,367	8,027	8,027	8,011	8,003	8,003	8,003	8,003	8,003	7,888	4,332
Tutte	Altro		Tutte	2,144	13,469	15,690	13,812	7,803	6,636	6,531	6,275	7,367	8,027	8,027	8,011	8,003	8,003	8,003	8,003	8,003	7,888	4,332
	Altro	(p.8 all 1 DM 503/97) Cd+Tl	Tutte	0	0	0																
		ACET. BUTILE	Tutte			2,100																
		Acetato di Metile	Tutte		20	48	48	48	48	32												
		ACETONE	Tutte		0	0	0															
		ACIDI INORGA	Tutte	0	0	0																
		Acidi organici superiori	Tutte	0	140	336	336	336	336	224												
		ACIDI TOTALI	Tutte	2,144	12,864	12,864	6,432	0														
		ACIDO BORICO	Tutte	0	0	0																
		ACIDO BROMID	Tutte	0	0	0																
		ACIDO CIANIDRICO	Tutte		0	0	0	0	0	0	0	0	0	0	0							
		ACIDO CLORIDRICO E IONE CLORO (HCl)	Tutte		0	0	0															
		ACIDO FLUORIDRICO	Tutte		0	0	0	0														
		ACIDO FLUORIDRICO E IONE FLUORO (HF)	Tutte		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
		ACIDO SOLFORICO	Tutte		7	6	907	1,664	1,557	1,547	1,547	1,595	1,679	1,679	1,679	1,679	1,679	1,679	1,679	1,679	1,619	840
		ALDEIDI	Tutte				14															
		AMMONIO	Tutte		56		1,203	1,470	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	720
		ANIDRIDE CARBONICA	Tutte		126	108	2,020	2,644	2,559	2,592	2,592	3,636	4,212	4,212	4,212	4,212	4,212	4,212	4,212	4,212	4,157	2,380
		ANIDRIDE CROMICA	Tutte		18	24	24	24	24	24	24	24	24	24	8							
		Azoto nitrico	Tutte				0	0	0	0	0	0										
		BIOSSIDO DI AZOTO NO2	Tutte		203	174	1,645	1,041	96	96	96	96	96	96	96	96	96	96	96	96	96	56
		CO E CO2	Tutte		35	30	1,183	576	576	576	576	576	576	576	576	576	576	576	576	576	576	336
		ETANOLAMMINA	Tutte			0																
		MONOSSIDO DI CARBONIO	Tutte			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
		NOx come NO2	Tutte				0	0	0													
		OSSIDI DI AZOTO (ESPRESSI COME NO2)	Tutte			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
		POLVERI	Tutte			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
		SOV (CLASSE III,IV,V)	Tutte			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
		Zinco + Alluminio 5	Tutte			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

E come si può notare, nell'immagine vengono visualizzate i segni + che permettono di fare le operazione di ROLL-UP e di DRILL-DOWN. Come esempio faccio il DRILL-DOWN dell'anno 2007 e il DRILL- DOWN di un comune e quindi viene visualizzato il risultato seguente:



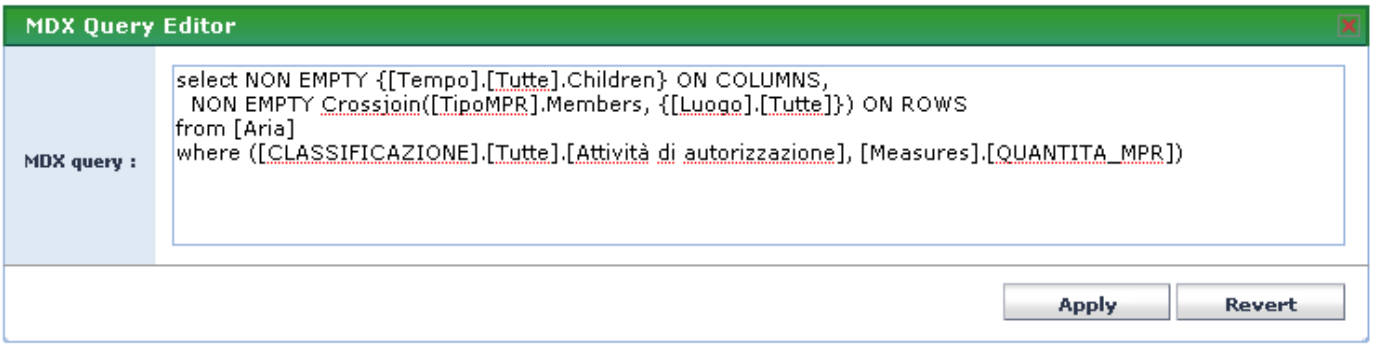
Inquinante			Luogo			Tempo																				
						2006	2007	S1	2007 S1	T1	2007 S1 T2	S2	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024
(All)	CATEGORIA	TIPO	(All)	PROVINCIA	COMUNE																							
Tutte			Tutte			2,144	13,469	6,499	3,216	3,283	6,970	15,690	13,812	7,803	6,636	6,531	6,275	7,367	8,027	8,027	8,011	8,003	8,003	8,003	8,003	8,003	7,888	4,332
Tutte	Altro		Tutte			2,144	13,469	6,499	3,216	3,283	6,970	15,690	13,812	7,803	6,636	6,531	6,275	7,367	8,027	8,027	8,011	8,003	8,003	8,003	8,003	8,003	7,888	4,332
	Altro	(p.8 all 1 DM 503/97) Cd+Tl	Tutte			0	0	0	0	0	0	0																
		ACET. BUTILE	Tutte									2,100																
		Acetato di Metile	Tutte				20				20	48	48	48	48	32												
		ACETONE	Tutte				0	0	0	0	0	0	0															
		ACIDI INORGA	Tutte			0	0	0	0	0	0	0																
		Acidi organici superiori	Tutte			0	140	0	0	0	140	336	336	336	336	224												
			Tutte	MODENA		0	140	0	0	0	140	336	336	336	336	224												
			MODENA		CASTELVETRO DI MODENA		140				140	336	336	336	336	224												
					MODENA	0	0	0	0	0	0	0																
		ACIDI TOTALI	Tutte			2,144	12,864	6,432	3,216	3,216	6,432	12,864	6,432	0														
		ACIDO BORICO	Tutte			0	0	0	0	0	0	0																
		ACIDO BROMID	Tutte			0	0	0	0	0	0	0																
		ACIDO CIANIDRICO	Tutte				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
		ACIDO CLORIDRICO E IONE CLORO (HCl)	Tutte				0	0	0	0	0	0	0															
		ACIDO FLUORIDRICO	Tutte				0	0	0	0	0	0	0	0	0													
		ACIDO FLUORIDRICO E IONE FLUORO (HF)	Tutte				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
		ACIDO SOLFORICO	Tutte				7	1		1	6	6	907	1,664	1,557	1,547	1,547	1,595	1,679	1,679	1,679	1,679	1,679	1,679	1,679	1,619	840
		ALDEIDI	Tutte										14															
		AMMONIO	Tutte				56	8		8	48		1,203	1,470	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	1,440	720
		ANIDRIDE CARBONICA	Tutte				126	18		18	108	108	2,020	2,644	2,559	2,592	2,592	3,636	4,212	4,212	4,212	4,212	4,212	4,212	4,212	4,212	4,157	2,380
		ANIDRIDE CROMICA	Tutte				18	6		6	12	24	24	24	24	24	24	24	24	24	8							
		Azoto nitrico	Tutte										0	0	0	0	0	0										
		BIOSSIDO DI AZOTO NO2	Tutte				203	29		29	174	174	1,645	1,041	96	96	96	96	96	96	96	96	96	96	96	96	96	56
		CO E CO2	Tutte				35	5		5	30	30	1,183	576	576	576	576	576	576	576	576	576	576	576	576	576	576	336
		ETANOLAMMINA	Tutte									0																
		MONOSSIDO DI CARBONIO	Tutte									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		NOx come NO2	Tutte										0	0	0													
		OSSIDI DI AZOTO (ESPRESSI COME NO2)	Tutte									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		POLVERI	Tutte									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		SOV (CLASSE III,IV,V)	Tutte									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		Zinco + Alluminio 5	Tutte									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Il secondo *report*: per creare questo *report* viene utilizzato il *Modeller* di QUIXBI e dopo aver fatto le scelte viene visualizzata l'immagine seguente:

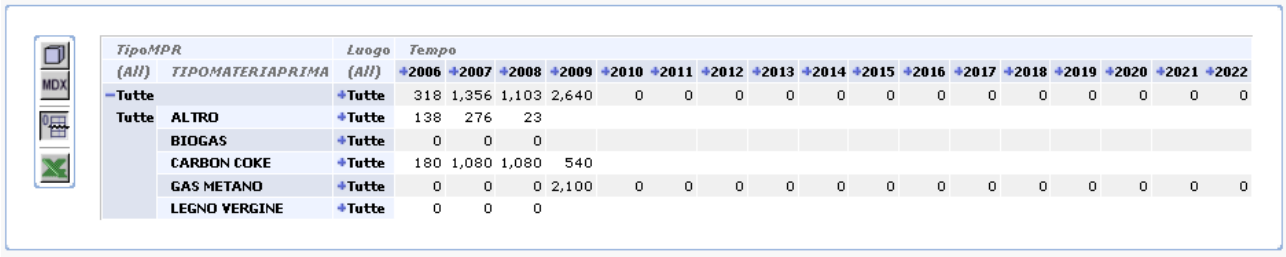


Il secondo *report* è uguale al primo ma con l'obiettivo di far vedere la quantità delle materie prima anziché la quantità degli inquinanti quindi sulle righe è stato scelto di visualizzare le materie prima e i comuni mentre sulle colonne è stato scelto di visualizzare la dimensione del tempo. Inoltre, è stato scelto di visualizzare la misura QUANTITA_MPR e la dimensione classificazione è stata affettata per riportare solo le materie prime con classificazione uguale **attività di autorizzazione**.

Utilizzando il *Modeller* e facendo le scelte descritte sopra la query MDX generata è il seguente:

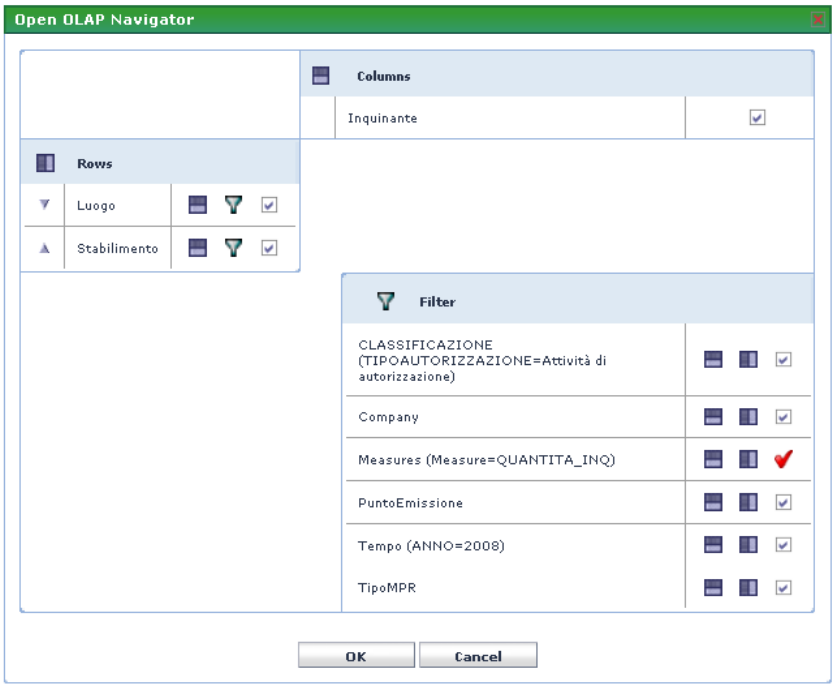


Come si può vedere, è uguale a quella di prima con l'unica differenza che è stata scelta la dimensione delle materie prime e la misura QUANTITA_MPR anziché la dimensione inquinante e la misura QUANTITA_INQ e quindi la query dà il seguente risultato, sempre con la possibilità di espansione e di raggruppamento del risultato:



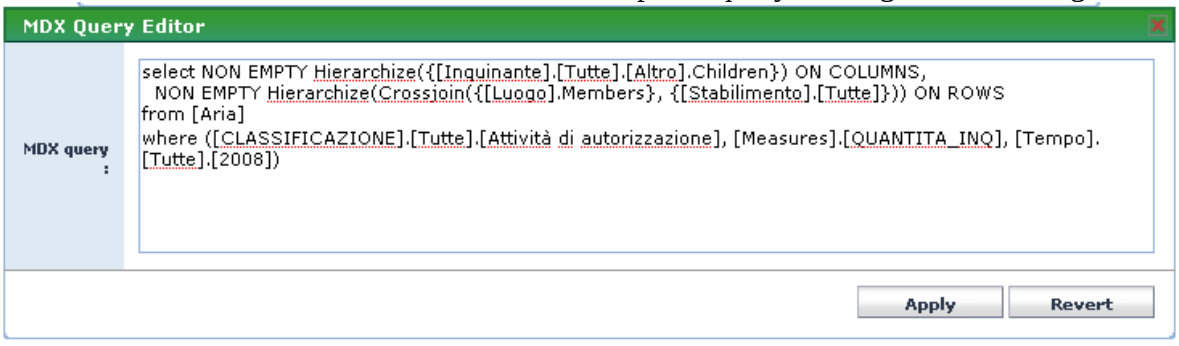
TipoMPR (All)	TIPOMATERIAPRIMA	Luogo (All)	Tempo +2006	+2007	+2008	+2009	+2010	+2011	+2012	+2013	+2014	+2015	+2016	+2017	+2018	+2019	+2020	+2021	+2022
−Tutte		+Tutte	318	1,356	1,103	2,640	0	0	0	0	0	0	0	0	0	0	0	0	0
Tutte	ALTRO	+Tutte	138	276	23														
	BIOGAS	+Tutte	0	0	0														
	CARBON COKE	+Tutte	180	1,080	1,080	540													
	GAS METANO	+Tutte	0	0	0	2,100	0	0	0	0	0	0	0	0	0	0	0	0	0
	LEGNO VERGINE	+Tutte	0	0	0														

Il terzo *report*: per creare questo *report* viene utilizzato il *Modeller* di QUIXBI e dopo aver fatto le scelte viene visualizzata l'immagine seguente:



Il terzo *report* ha l'obiettivo di visualizzare la quantità autorizzata degli inquinanti per ogni stabilimento di ogni comune per l'anno in corso. Quindi, come si può vedere nell'immagine sopra riportata, sulle righe è stato scelto di visualizzare i comuni e gli stabilimenti mentre sulle colonne è stato scelto di visualizzare la dimensione inquinante ed è stato filtrato **per classificazione = attività di autorizzazione** per precisare che si vuole solo la quantità autorizzata; per la dimensione del tempo è stato scelto di vedere solo l'anno 2008 mentre per le misure è stato scelto di visualizzare solo la QUANTITA_INQ. Quindi, il *report* visualizza la quantità autorizzata di ogni inquinante per ogni stabilimento di ogni comune per l'anno 2008 e ciò che è stato richiesto dal cliente.

Utilizzando il *Modeller* e facendo le scelte descritte sopra la query MDX generata è il seguente:

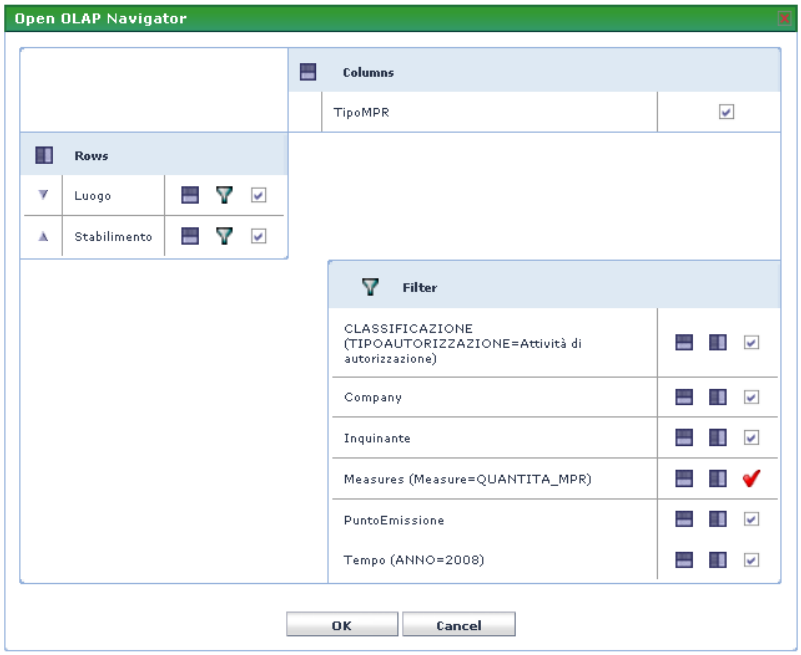


Come si può vedere viene selezionata la dimensione inquinante sulle colonne con tutti i suoi livelli già espansi permettendo di visualizzare subito tutti i tipi di inquinanti quando viene visualizzato il *report* e inoltre vengono selezionate solo le colonne che hanno dati, eliminando le colonne non significative.
Sulle righe invece, vengono selezionate le dimensioni luogo e stabilimento facendo il cross join tra di loro mentre nella condizione viene selezionata la misura QUANTITA_INQ, la dimensione **tempo = 2008** per filtrare il risultato visualizzando solo quello dell'anno 2008 e la dimensione **classificazione = attività di autorizzazione** per visualizzare solo la quantità degli inquinanti autorizzati realizzando cosi l'operazione di SLICE delle dimensioni.

la query descritta sopra fa vedere il risultato seguente (di seguito riportato solo in parte perché è troppo lungo a causa della presenza di tanti tipi di inquinanti):

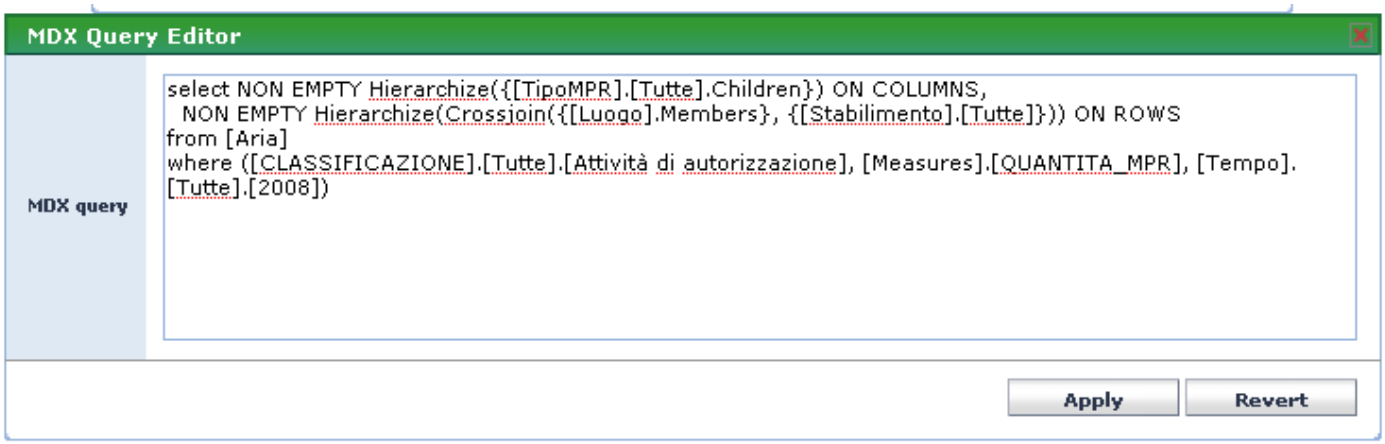
				Inquinante							
Luogo			Stabilimento	Altro							
(All)	PROVINCIA	COMUNE	(All)	(p.8 all 1 DM 503/97) Cd+Tl	ACET. BUTILE	Acetato di Metile	ACETONE	ACIDI INORGA	Acidi organici superiori	ACIDI TOTALI	ACIDO
−Tutte			+Tutte	0	2,100	48	0	0	336	12,864	
Tutte	−BOLOGNA		+Tutte							12,864	
	BOLOGNA	CRESPELLANO	+Tutte							12,864	
	−GROSSETO		+Tutte								
	GROSSETO	FOLLONICA	+Tutte								
		GROSSETO	+Tutte								
	−MODENA		+Tutte	0	2,100	48	0	0	336	0	
	MODENA	CASTELVETRO DI MODENA	+Tutte			48			336		
		MODENA	+Tutte						0	0	
		SAN CESARIO SUL PANARO	+Tutte				0				
		SASSUOLO	+Tutte					0			
		SOLIERA	+Tutte	0	2,100						
		ZOCCA	+Tutte								
	−REGGIO NELL'EMILIA		+Tutte								
	REGGIO NELL'EMILIA	CORREGGIO	+Tutte								
	−VERONA		+Tutte								
	VERONA	VERONA	+Tutte								

Il quarto *report*: per creare questo *report* viene utilizzato il *Modeller* di QUIXBI e dopo aver fatto le scelte viene visualizzata l'immagine seguente:

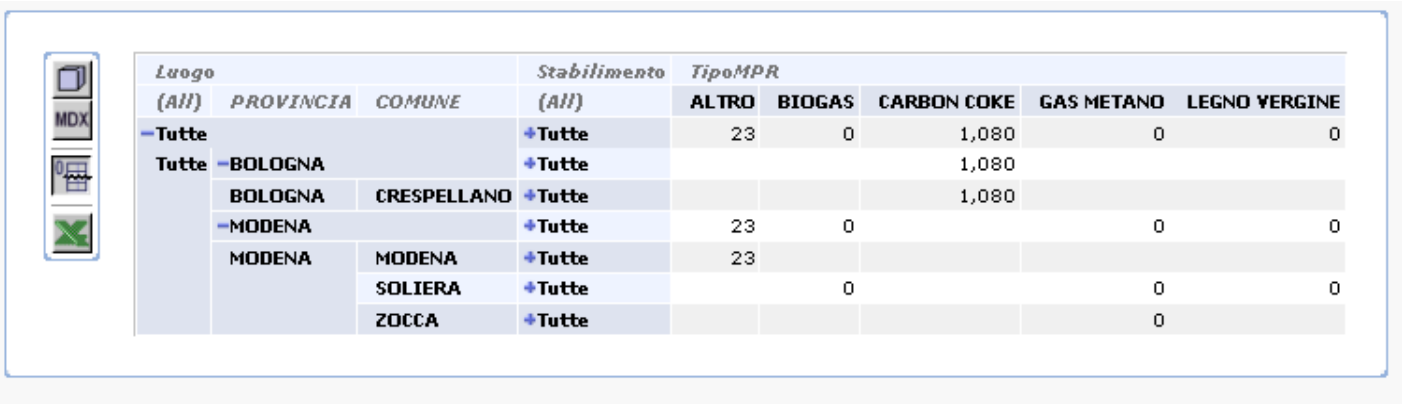


Il Quarto *report* è uguale al terzo ma con l'obiettivo di far vedere la quantità delle materie prima anziché la quantità degli inquinanti quindi sulle colonne è stato scelto di visualizzare le materie prima mentre sulle righe sono rimaste le dimensioni del comune e dello stabilimento. Inoltre, è stato scelto di visualizzare la misura QUANTITA_MPR anziché QUANTITA_INQ mentre gli altri due filtri sono rimasti gli stessi.

Utilizzando il *Modeller* e facendo le scelte descritte sopra la query MDX generata è il seguente:

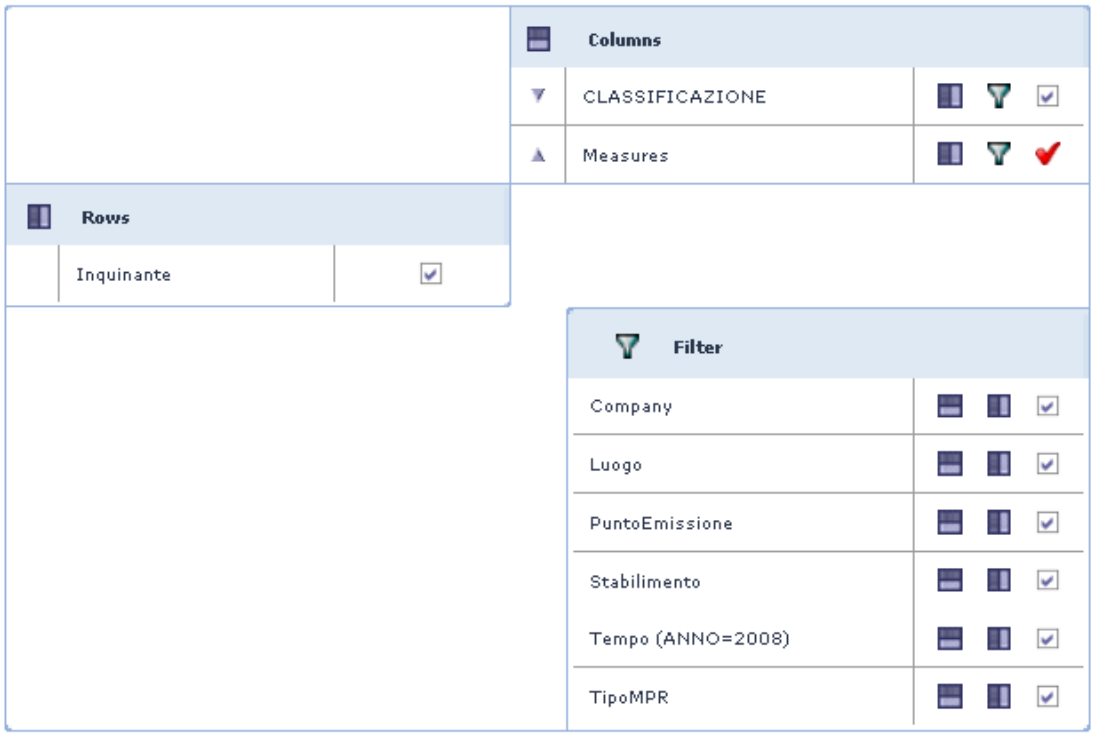


Come si può vedere, la query MDX è uguale a quella di prima con l'unica differenza che è stata scelta la dimensione delle materie prime e la misura QUNATITA_MPR anziché la dimensione inquinante e la misura QUANTITA_INQ e quindi la query dà il seguente risultato, sempre con la possibilità di espansione e di raggruppamento del risultato:



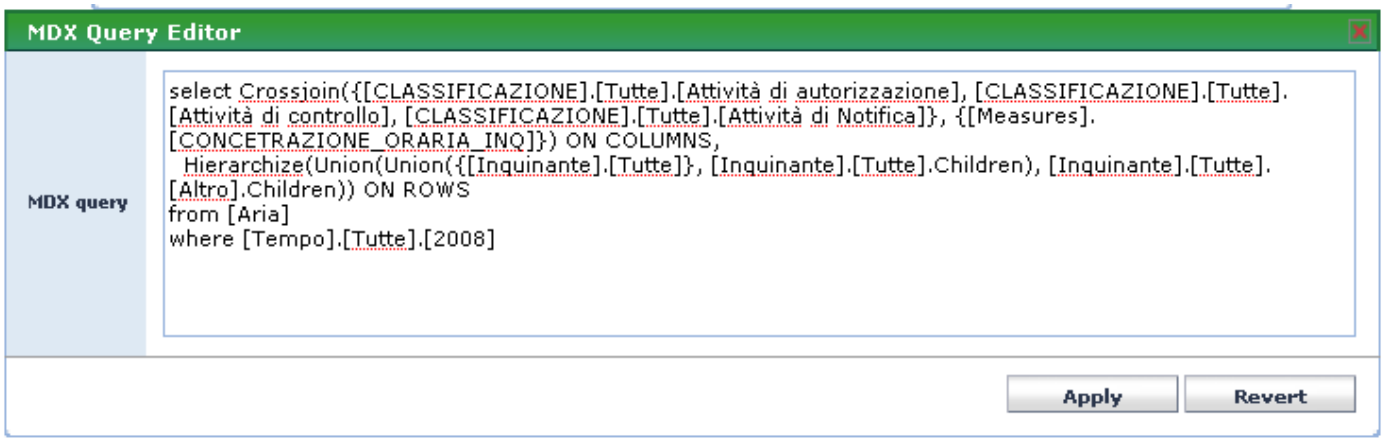
Luogo			Stabilimento	TipoMPR				
(All)	PROVINCIA	COMUNE	(All)	ALTRO	BIOGAS	CARBON COKE	GAS METANO	LEGNO VERGINE
−Tutte			+Tutte	23	0	1,080	0	0
Tutte	−BOLOGNA		+Tutte			1,080		
	BOLOGNA	CRESPELLANO	+Tutte			1,080		
	−MODENA		+Tutte	23	0		0	0
	MODENA	MODENA	+Tutte	23				
		SOLIERA	+Tutte		0		0	0
		ZOCCA	+Tutte				0	

Il quinto *report*: per creare questo *report* viene utilizzato il *Modeller* di QUIXBI e dopo aver fatto le scelte viene visualizzata l'immagine seguente:



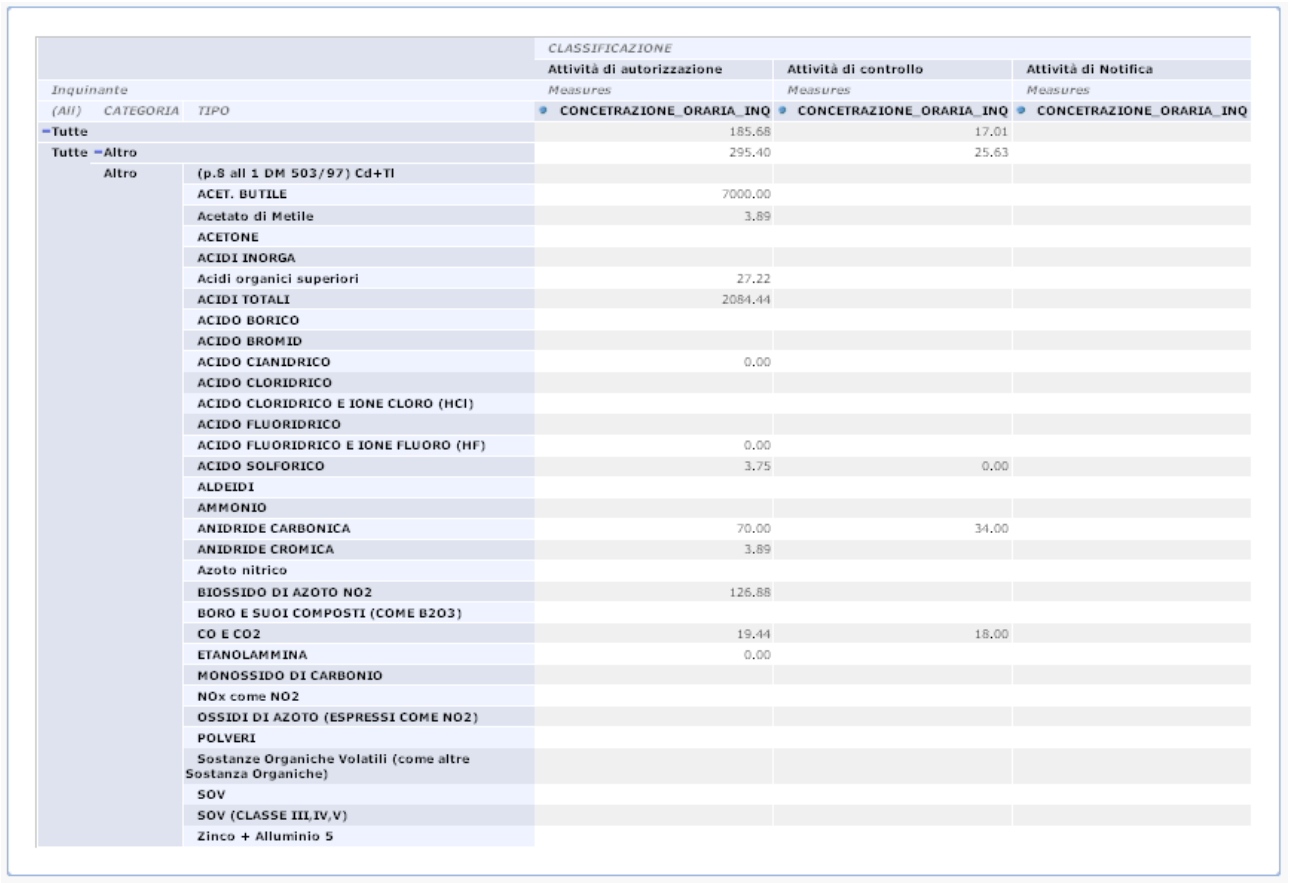
Il quinto *report* ha l'obiettivo di fare il confronto tra la concentrazione degli inquinanti emessi, la concentrazione degli inquinanti autorizzati per essere emessi e per la concentrazione degli inquinanti dichiarati nell'attività di autocontrollo fatta periodicamente dall'ente stessa per l'anno in corso e quindi per questo *report* è stato scelto l'anno 2008. Quindi, come si può vedere nell'immagine sopra riportata, sulle righe è stato scelto di visualizzare gli inquinanti mentre sulle colonne è stato scelto di visualizzare la dimensione classificazione con solo i tre tipi di autorizzazione citati sopra e la misura CONCENTRAZIONE_ORARIA_INQ. Inoltre, è stato scelto di filtrare la dimensione del tempo per visualizzare solo l'anno 2008.

Utilizzando il *Modeller* e facendo le scelte descritte sopra la query MDX generata è il seguente:



Come si può vedere, viene selezionata la classificazione con i tre tipi desiderati sulle colonne e la concentrazione degli inquinanti. Sulle righe invece, viene selezionata la dimensione inquinante mentre nella condizione viene selezionata solo la dimensione Tempo con anno =2008 per filtrare il risultato visualizzando solo la concentrazione dell'anno 2008.

Infine, la query descritta sopra fa vedere il seguente risultato:



			CLASSIFICAZIONE		
			Attività di autorizzazione	Attività di controllo	Attività di Notifica
Inquinante			Measures	Measures	Measures
(All)	CATEGORIA	TIPO	CONCETRAZIONE_ORARIA_INQ	CONCETRAZIONE_ORARIA_INQ	CONCETRAZIONE_ORARIA_INQ
− Tutte			185.68	17.01	
Tutte	− Altro		295.40	25.63	
	Altro	(p.8 all 1 DM 503/97) Cd+Tl			
		ACET. BUTILE	7000.00		
		Acetato di Metile	3.89		
		ACETONE			
		ACIDI INORGA			
		Acidi organici superiori	27.22		
		ACIDI TOTALI	2084.44		
		ACIDO BORICO			
		ACIDO BROMID			
		ACIDO CIANIDRICO	0.00		
		ACIDO CLORIDRICO			
		ACIDO CLORIDRICO E IONE CLORO (HCl)			
		ACIDO FLUORIDRICO			
		ACIDO FLUORIDRICO E IONE FLUORO (HF)	0.00		
		ACIDO SOLFORICO	3.75	0.00	
		ALDEIDI			
		AMMONIO			
		ANIDRIDE CARBONICA	70.00	34.00	
		ANIDRIDE CROMICA	3.89		
		Azoto nitrico			
		BIOSSIDO DI AZOTO NO2	126.88		
		BORO E SUOI COMPOSTI (COME B2O3)			
		CO E CO2	19.44	18.00	
		ETANOLAMMINA	0.00		
		MONOSSIDO DI CARBONIO			
		NOx come NO2			
		OSSIDI DI AZOTO (ESPRESSI COME NO2)			
		POLVERI			
		Sostanze Organiche Volatili (come altre Sostanza Organiche)			
		SOV			
		SOV (CLASSE III,IV,V)			
		Zinco + Alluminio 5			

Quindi, dopo la realizzazione del quinto *report* si può dire che la struttura realizzata è stata adatta per soddisfare la richiesta del cliente.

5. CONCLUSIONI E LAVORO FUTURO

Lo stage svolto aveva due obiettivi: il primo era quello di risolvere un problema attuale dell'azienda che era quello della dipendenza dei flussi di caricamento delle strutture dei cubi creati con il RDBMS utilizzato ed ottimizzare i tempi di caricamento di esse.
Il secondo obiettivo era quello di creare una nuova struttura OLAP che si occupasse delle pratiche dell'aria e che rispondesse alle esigenze del cliente.

Per il primo obiettivo sono stati definiti i seguenti sotto-obiettivi da raggiungere:

1. Utilizzare un nuovo strumento ETL *open-source*, per creare i flussi di caricamento, per avere dei costi il più possibile contenuti
2. Rendere i flussi di caricamento indipendenti del RDBMS utilizzato
3. Ottimizzare il caricamento delle tabelle diminuendo i tempi di essi
4. Rendere flessibile la trasformazione per rispondere alla esigenza dell'azienda di scegliere i cubi da caricare, in modo da permettere di installare la trasformazione da qualsiasi cliente

Il punto 1. e` stato sicuramente raggiunto, perché, è stato utilizzato PDI che è uno strumento *Open - source* che non richiede costi di licenza.
Anche il punto 2. è stato raggiunto utilizzando i parametri per la definizione della connessione alla base di dati e quindi permettendo di utilizzare qualsiasi RDBMS che si interfaccia con SPOON.
Per il caricamento dei dati (punto 3.) c'è stato un intervento riguardo la struttura della tabella BID_CALENDARIO, che è una dimensione comune tra tutti i cubi, che ha permesso di ottimizzare molto i tempi di caricamento di questa tabella e, di conseguenza, i tempi di caricamento di tutte le strutture create; quindi anche questo sotto obiettivo è stato raggiunto.
L'ultimo sotto obiettivo è stato raggiunto, utilizzando dei parametri che permettono di scegliere se caricare il cubo o meno; se si vuole caricare il cubo, questi parametri vengono passati, altrimenti no. Ciò ha permesso di rendere la trasformazione molto flessibile e quindi è stato raggiunto anche questo sotto obiettivo.

Per il secondo obiettivo sono stati invece definiti i seguenti sotto obiettivi:

1. Creazione di una nuova struttura per l'aria
2. Progettazione e realizzazione di un flusso per il suo caricamento
3. Ottimizzazione del flusso di caricamento
4. Creazione dei *report* per rispondere alle esigenze del Cliente.

La nuova struttura è stata realizzata partendo dai requisiti del cliente e facendo un'analisi molto approfondita per renderla facilmente estendibile per eventuali esigenze future per altri *report* di analisi; quindi possiamo dire che il primo sotto obiettivo è stato pienamente raggiunto.
Dopo la creazione della struttura è stato creato il flusso di caricamento delle tabelle utilizzando sempre *SPOON*. Poiché nella prima parte dello stage si sono incontrati dei problemi prestazionali nel caricamento delle tabelle, è stato definito un sotto obiettivo che include i punti 2. e 3.; si e` fatta un'analisi approfondita del flusso creato per cercare di ottimizzarlo e per evitare problemi prestazionali riguardo al caricamento della tabella dei fatti a causa dei join tra le tabelle.
Questo problema è stato risolto cambiando l'ordine delle operazioni da fare e ottenendo pienamente il risultato misurato in termini di diminuzione notevole nei tempi di caricamento.
Per quanto riguarda l'ultimo sotto-obiettivo, è stata utilizzata l'applicazione QUIXBI sviluppata da QUIX per realizzare i *report* che hanno soddisfatto i requisiti del cliente quindi anche questo obiettivo è stato raggiunto.

Complessivamente il lavoro prodotto sembra essere un buon compromesso tra il mantenimento di costi bassi e l'ottenimento di prestazioni accettabili; infatti, durante lo stage è stato verificato che SPOON presentava dei limiti prestazionali rispetto ad alcuni prodotti a pagamento di costo però notevolmente superiore.
Si noti che, la prima trasformazione realizzata in *KETTLE* per risolvere i problemi della dipendenza con i RBDMS, è stata fatta anche con l'obiettivo di imparare ad usare questo strumento, quindi all'inizio mancava l'adeguata esperienza per utilizzarlo. Come lavoro futuro, si possono fare delle ottimizzazioni per questi flussi cambiando sia la logica della realizzazione delle trasformazione sia i blocchi utilizzati con altri blocchi.
Inoltre, si possono definire nuovi *report* da realizzare utilizzando tutte le strutture realizzate introducendo delle modifiche su di essi per aggiungere nuovi dimensioni e nuove misure per soddisfare le nuove richieste.
Infine, riguarda il quinto *report*, la definizione della concentrazione voluta non è stata molto chiara quindi questo *report* potrebbe essere modificato introducendo la formula giusta nel file di mappatura di *Mondrian* quando si arriva alla definizione giusta facendo altre riunioni con il cliente per discutere questo requisito.

Per finire, l'esperienza di lavoro è stata particolarmente gratificante in quanto mi ha dato l'opportunità di conoscere più da vicino la complessità sistemica della vita all'interno dell'azienda; ho potuto inoltre migliorare la mia preparazione nella realizzazione di *data warehouse* con un caso di mondo reale, che presenta sempre delle difficoltà impreviste e capire meglio i meccanismi di gestione di un'azienda.

6. GLOSSARIO

Aggregazione: Nel modello multidimensionale, meccanismo per la sintesi dei dati che permette di raggruppare un insieme di eventi in un unico evento che li riassume.

Alimentazione: procedura attraverso la quale i dati presenti nelle sorgenti operazionali vengono integrati, resi consistenti e caricati nel *data warehouse*.

Attributo: Nel modello relazionale, nome dato ad una colonna di una relazione con un dominio di valori finito. Nel modello multidimensionale, è un campo con dominio finito (tipicamente alfanumerico) che descrive una dimensione.

Business Intelligence: Qualsiasi processo usato per estrarre ed analizzare dati aziendali. Questo termine sta avendo un notevole successo e sta soppiantando il suo sinonimo *Decision Support System.*

Campo: La rappresentazione di un attributo in una base di dati, in un file.

Caricamento: Fase del processo di *data warehousing* durante la quale i dati operazionali ripuliti e trasformati vengono caricati nel *data warehouse.*

Cubo: Metafora adottata per rappresentare i dati nel modello multidimensionale. Ogni evento accaduto viene visto come una cella di un cubo i cui spigoli rappresentano le dimensioni di analisi.

Data warehouse: Una collezione di dati di supporto per il processo decisionale orientata ai soggetti di interesse, integrata e consistente, rappresentativa dell'evoluzione temporale e non volatile.

Data mart (DM): Un sottoinsieme o un'aggregazione dei dati presenti nel *data warehouse*, contenente l'insieme delle informazioni rilevanti per una particolare area del *business*, una particolare divisione dell'azienda, una particolare categoria di soggetti.

Data warehousing: Metodi, tecnologie e strumenti di ausilio per condurre analisi dei dati finalizzate all'attuazione di processi decisionali e al miglioramento del patrimonio informativo aziendale.

Database: La combinazione dello schema di *database* e dei dati memorizzati.

Database Managment System (DBMS): un *database management system* è un software, o un gruppo di software, attraverso il quale è possibile definire, creare e gestire una base di dati e regolarne l'accesso alle informazioni. La maggior parte dei DBMS utilizzati nel *Data warehousing* sono basati sulla tecnologia relazionale mentre alcuni dei prodotti più recenti sfruttano direttamente una memorizzazione di tipo multidimensionale.

Database operazionale: Contiene i dati generati da operazioni, principalmente di carattere amministrativo, svolte all'interno dei processi gestionali.

Dimension table: nello schema a stella, tabella relazionale denormalizzata contenente una chiave surrogata e l'insieme degli attributi facenti parte di una gerarchia.

Dimensione: è un attributo strutturale di un Array Multidimensionale formato da una lista di membri simili dal punto di vista dell'utente finale; ad esempio giorni, mesi, trimestri e anni fanno parte della dimensione temporale mentre comuni, provincie e regioni possono far parte di una dimensione geografica. Una dimensione può essere considerata un indice in base al quale identificare i valori in un Array Multidimensionale e permette di avere un modo conciso e intuitivo di organizzare e selezionare i dati da analizzare.

Drill-Down: Operatore OLAP che diminuisce l'aggregazione dei dati in un cubo introducendo un ulteriore livello di dettaglio.

Drill-through: operatore OLAP che permette il passaggio dai dati aggregati multidimensionali di un cubo ai dati operazionali presenti nelle sorgenti o nel livello riconciliato.

Estrazione: Fase del processo di *data warehousing* durante la quale i dati rilevanti vengono estratti dalle sorgenti.

ETL – Extraction, Trasformation and Loading: Si tratta di strumenti che permettono di integrare schemi eterogenei, nonché di estrarre, trasformare, ripulire, validare, filtrare e caricare i dati dalle sorgenti nel *data warehouse*.
Nelle architetture a più livelli le operazioni svolte vengono spesso globalmente indicate con il termine riconciliazione.

Fact Table (Tabella dei fatti): nello schema a stella, tabella relazionale contenente tutte le misure di un fatto. La *fact table* importa le chiavi di tutte le tabelle dimensionali, che insieme formano la chiave primaria.

Fatto: concetto di interesse per il processo decisionale. Tipicamente modella un insieme di eventi che accadono nell'impresa.

Integrazione: procedimento attraverso il quale più schemi operazionali sorgenti possono essere combinati in un unico schema consistente.

Join : date due relazioni e un predicato booleano che coinvolge attributi di entrambe, l'operazione di join collega tutte le coppie di tuple prese dalle due relazioni per le quali il predicato è soddisfatto.

Modello Multidimensionale: Modello alla base dei sistemi di *data warehousing*, vede i dati come punti in uno spazio le cui dimensioni corrispondono ad altrettante possibili dimensioni di analisi; ciascun punto, rappresentativo di un evento accaduto nell'azienda, viene descritto tramite un insieme di misure di interesse per il processo decisionale.

OLAP: *On-line Analytical Processing*, elaborazione interattiva dei dati orientata all'analisi dinamica e multidimensionale.

Pulitura: Fase del processo di *data warehousing* durante la quale viene migliorata la qualità dei dati.

Report: rapporto riassuntivo di tipo statico destinato a utenti che hanno necessità di accedere periodicamente a informazioni strutturate in modo pressoché invariabile.

Roll-Up: operatore OLAP che aggrega un insieme di eventi in un cubo diminuendo il livello di dettaglio.

Schema a stella: Schema base per la rappresentazione di dati multidimensionali attraverso il modello relazionale. Consiste di una tabella dei fatti e di un insieme di tabelle dimensionali a esse collegate.

Slice: Una slice è composta da un sotto-insieme, o fetta, di un array multidimensionale corrispondente a uno o più elementi di una o più dimensioni. Le dimensioni lungo le quali si effettua la selezione degli elementi non saranno presenti nel sottoinsieme stesso. Dal punto di vista dell'utente, il termine Slice si riferisce a una vista bidimensionale creata dal cubo multidimensionale.

Slice and Dice: Il processo di navigazione dei dati da parte dell'utente finale, è effettuato per mezzo di visualizzazioni interattive prodotte da rotazioni e selezioni o da operazioni di Drill-Down/ Roll-Up.

Trasformazione: Fase del processo di *data warehousing* durante la quale i dati vengono convertiti dal formato operazionale sorgente a quello utilizzato nel livello riconciliato.

7. BIBLIOGRAFIA

[1] Wikipedia, it.wikipedia.org

[2] L. Cabibbo - "Il Modello Dimensionale", Documento, Università Degli Studi Roma Tre, 2000

[3] Pentaho Analysis Services, mondrian.pentaho.org

[4] Miriade Technology - "Mondrian-JPIVOT, Business Intelligence con Strumenti Software Open Source", www.miriade.it

[5] M. Bonacorsi - "BART: Uno Strumento di Analisi di Business e Reportistica", Università Degli Studi Di Modena e Reggio Emilia, 2006

[6] Pentaho Analysis Services – Mondrian Developer Guide, mondrian.pentaho.org/documentation

[7] Strumenti ETL, "Data Warehouse – teoria e pratica della progettazione" M. Golfarelli, S. Rizzi

[8] Fasi della progettazione concettuale, "Data Warehouse – teoria e pratica della progettazione" M. Golfarelli, S. Rizzi

[9] Kimball R., Ross M., The Data Warehouse Toolkit – The Complete Guide to Dimensional Modeling. Wiley, New York, USA, 2002.

[10] Kimball R., Caserta J., The Data Warehouse ETL Toolkit.Wiley, Indianapolis, USA, 2004.

[11] http://www.olap.it/

[12] http://JPIVOT.sourceforge.net/

[13] http://mondrian.sourceforge.net/

[14] http://www.pentaho.org/

[15] http://kellte.pentaho.com/products/data_integration/

[16] http://ragnoworld.splinder.com

[17] http://www.mdxtutorials.net/

[18] http://kettle.pentaho.org/

8. RINGRAZIAMENTI

*Dedico questa tesi a **Meryem** e a tutta la mia famiglia: **papà, mamma, i miei fratelli e le mie sorelle.***
In tutta la mia vita da studente non mi hanno mai fatto mancare supporto, affetto e fiducia, mettendomi nelle condizioni di serenità grazie alle quali ho raggiunto la fine del mio percorso di studi.
*Vorrei ringraziare di tutto il mio cuore, mio fratello **Maher** che è stato un vero fratello e che mi ha dato sempre un supporto sia economico sia morale per continuare i miei studi, per affrontare i problemi della vita e per crescere e continuare a crescere fino a diventare ciò che sono.*
*Inoltre un grazie particolare alla mia fidanzata **Meryem** che mi ha dato tutto il supporto durante il periodo dello stage anche se a volte mi faceva arrabbiare.*

Per la realizzazione della tesi ringrazio:

*- La **prof.ssa Sonia Bergamaschi** per la fiducia dimostratami e per i preziosi consigli su come organizzare il mio lavoro ;*

*- Il **prof. Flavio Bonfatti** per avermi aiutato quando ho avuto i problemi con la prima azienda dove ho iniziato il mio stage.*

*- **L'azienda QUIX** per avermi dato l'opportunità per svolgere questo lavoro e per imparare tutte le tecnologie che ho imparato.*

*- L'ing. **Mattia Bonacorsi** che mi ha seguito per tutto il periodo dello stage dandomi tutte le informazioni che servivano per affrontare i problemi dello stage.*

*- L'ing. **Simone Smerieri** per la pazienza che ha avuto quando mi spiegava il lavoro da fare.*
*- L'ing. **Claudio Masini** per la sua disponibilità e per il suo aiuto.*

*- **Tutti gli ingegneri dell'azienda Quix** per avermi supportato e per avermi sempre fatto sentire come se fossi a casa mia.*

Ho inoltre il dovere di ringraziare:

*- **Luca Carafoli e Alessandro Fronteddu**, i miei cari compagni di studio che mi davano sempre la voglia di studiare e di passare gli esami studiano e discutendo gli argomenti delle lezioni con me e portandomi gli appunti quando non riuscivo a seguire le lezioni.*

*- **Sawzar Rashid** per tutti i passaggi notturni che mi dava anche per quelli diurni perché senza di lui avrei camminato troppo.*

*Vorrei infine ringraziare anche **tutte le persone che mi hanno dato sempre problemi** e mi davano sempre fastidio e cercavano di ostacolare il mio percorso perché un percorso sempre tranquillo non è bello e anche perché grazie ai problemi che mi hanno dato sono diventato la persona che sono adesso.*