UNIVERSITÀ DEGLI STUDI DI PADOVA

Facoltà di Scienze Statistiche

Corso di laurea triennale in Statistica e Tecnologie Informatiche

Tesi di laurea

ESPERIENZA DI STAGE IN OMICRON TECHNOLOGIES: PROGETTO SUL DATABASE FNOMCEO E DISCUSSIONE SULL'IMPORT/EXPORT DEI DATI CON XML

INTERNSHIP EXPERIENCE IN OMICRON TECHNOLOGIES: WORK ON FNOMCEO'S DATABASE AND DISCUSSION ABOUT DATA IMPORT/EXPORT WITH XML

Relatore: Prof. Nicola Ferro

Laureando: Leonardo Manfrotto
Matricola n. 557344

Anno accademico 2008-2009

Indice

Capitolo 1

Introduzione

L'intenzione principale del progetto di stage da seguire era finalizzata alla progettazione e all'implementazione di una base di dati per la business intelligence, presso la "Omicron Technologies S.R.L.". L'azienda, con varie sedi in Italia, occupa nel campo della produzione e della consulenza informatica, realizzando applicazioni e servizi informatici per i propri clienti. Attualmente, i mercati principali di riferimento riguardano lo sviluppo di applicazioni Web e l'implementazione di soluzioni sistemistiche, rivolte al mondo internet, per le aziende e la pubblica amministrazione.

Uno di questi clienti è la FNOMCeO - Federazione Nazionale degli Ordini dei Medici Chirurghi e Odontoiatri - la quale ha richiesto una base di dati (d'ora in poi DB, ovvero database) per contenere tutti i dati relativi ai professionisti della FNOMCeO e alle loro attività. Il DB è a libera consultazione degli utenti nel sito http://www.fnomceo.it, per risalire ad alcuni dati di un particolare professionista, mentre necessita di username e password per avere informazioni più dettagliate, ad esempio,

sui corsi di aggiornamento seguiti e il numero di crediti accumulati dai professionisti.

Ed è su questo DB che verterà la relazione. Verranno descritti i procedimenti necessari alla realizzazione di alcuni rapporti (report) necessari alla Federazione, partendo dal DB denominato FNOMCEO e scrivendo le interrogazioni in linguaggio SQL (query) necessarie. Successivamente, verrà spiegato anche il procedimento di importazione nel DB di alcuni dati relativi alle partecipazioni dei professionisti ai corsi di aggiornamento nel 2008. Negli argomenti appena citati verranno inclusi gli aspetti teorici riguardanti le basi di dati, il fulcro teorico centrale di questo progetto.

Prima di iniziare a descriverli, però, sarà necessario dedicare uno spazio all'analisi approfondita del DB Fnomceo, per capirne motivazioni, architettura e funzionamento. Alla fine della parte operativa, invece, verranno fatte alcune considerazioni di natura prettamente statistica sulla funzionalità dei report per la Federazione e se questi possono essere resi più comprensibili con grafici adatti o addirittura migliorabili. Infine, un ulteriore capitolo sarà dedicato all'uso di XML nelle basi di dati, sia dal punto di vista teorico che dal punto di vista pratico, anche se solo in un accenno.

Il tutto verrà esposto nel completo rispetto della privacy dei professionisti presenti nel DB aziendale, e quindi senza alcun riferimento all'interno della relazione.

Capitolo 2

Analisi del Progetto

2.1 Il ciclo di vita di un sistema informativo

La progettazione del DB Fnomceo verrà spiegata secondo una metodologia tipica nel campo dei sistemi informativi e che riguarda, più ampiamente, il ciclo di vita di un SI (Sistema Informativo).

Ecco illustrate qui di seguito le tappe del ciclo con una breve descrizione:

1. Studio di fattibilità: studio preliminare sui costi delle varie alternative possibili e decisione delle priorità di realizzazione delle varie componenti del sistema.

2. Raccolta e analisi dei requisiti: individuazione e studio delle proprietà e delle funzionalità che il sistema informativo deve avere. Tramite il rapporto con i futuri utenti del sistema, viene prodotta una descrizione completa dei dati coinvolti e di quali operazioni fare su di essi.

3. Progettazione: si divide in progettazione dei dati e in progettazione delle applicazioni. Nel primo caso si tratta dell'individuazione della struttura e dell'organizzazione dei dati, mentre nel secondo della definizione delle caratteristiche dei programmi applicativi. I due procedimenti possono essere sviluppati equivalentemente in sequenza o in cascata, a seconda delle esigenze. Le descrizioni dei dati e delle applicazioni sono formali e fanno riferimento a dei modelli specifici.

4. Implementazione: realizzazione del sistema informativo secondo quanto definito nella fase di progettazione.

5. Validazione e collaudo: verifica del funzionamento e della qualità del sistema informativo implementato.

6. Funzionamento: il sistema informativo diventa operativo ed esegue i compiti per il quale è stato progettato. Questa fase richiede anche processi di gestione, controllo e manutenzione.

Nel complesso, questo procedimento non è mai strettamente sequenziale. Ad ogni tappa bisogna rivedere le decisioni prese precedentemente, anche a costo di cambiarle per le necessità del progetto, come si può vedere dalla figura.

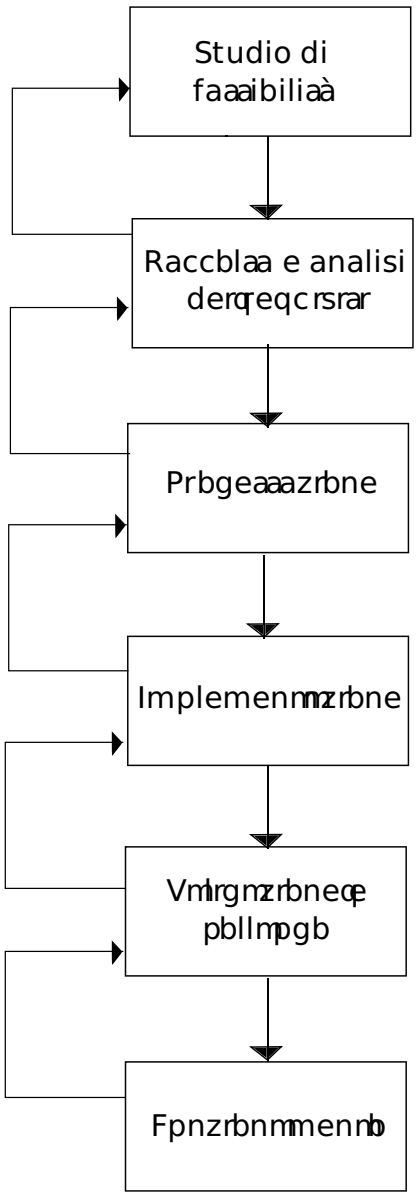


Figura 2.1: Ciclo di vita dei Sistemi Informativi

Il ciclo prevede numerose e complesse attività, che occupano sia risorse umane che economiche. Quindi è importante per un'azienda che vuole implementare un SI, seguire passo passo ogni tappa dell'evoluzione del progetto, instaurando un rapporto di reciproca collaborazione con i tecnici che si occuperanno dell'implementazione e, in seguito, della gestione e manutenzione del SI. Altrimenti, il rischio per l'azienda è di avere importanti perdite sia dal punto di vista economico che da quello delle risorse umane nello sviluppo di una cosa che non risponde agli interessi dell'azienda stessa.

Nell'analisi esplorativa del progetto Fnomceo, siamo particolarmente interessati ai punti 2 e 3 del ciclo appena presentato: analisi dei requisiti e progettazione del DB. In particolare, per la progettazione è necessario approfondire quali sono i vari stadi: nel campo delle basi di dati abbiamo una progettazione concettuale, una progettazione logica ed infine una progettazione fisica.

La prima raccoglie le informazioni dell'analisi dei requisiti e le usa per elaborare una descrizione ad alto livello semantico dei dati da memorizzare, nella maggior parte dei casi si tratta di un modello Entità-Associazione (Entity-Relationship o ER). La seconda implica la trasformazione dello schema ER in uno schema logico, in cui sono ben definite le tabelle che compongono il nostro DB e ne viene verificata la qualità (normalizzazione). L'ultima, infine, aggiorna lo schema logico con le specifiche dei parametri fisici di memorizzazione dei dati, organizzando file e indici. Nella descrizione della progettazione di

FNOMCEO, verranno descritte soprattutto la progettazione concettuale e la progettazione logica[1].

2.2 Analisi dei requisiti

La prima domanda a cui bisogna rispondere è: che cos'è la FNOMCeO? Si tratta di una federazione nata per raccogliere sotto lo stesso organo ministeriale tutti gli Ordini dei Medici Chirurghi e Odontoiatri d'Italia. Storicamente la divisione dell'Ordine MCeO era (ed è tuttora) su regime provinciale, in cui ogni Ordine si occupava di gestire i singoli professionisti iscritti in una determinata provincia. Col tempo si cercò di trovare un sistema che svincolasse i professionisti dall'esercizio nella singola provincia e che fungesse anche da struttura di riferimento per essi.

Una volta nata, la Federazione dovette raccogliere un gran numero di informazioni non organizzate provenienti da ogni singolo Ordine Provinciale relative ai dati anagrafici dei professionisti iscritti, ai corsi di aggiornamento e alle eventuali sanzioni disciplinari. Con la successiva informatizzazione e lo sviluppo di archivi sempre più capienti, la quantità di dati che arrivò nelle mani della Federazione aumentò esponenzialmente rendendo macchinoso e difficile qualsiasi ricerca, anche sul monitoraggio dell'operato di un singolo professionista.

Per questo motivo nacque l'esigenza di definire un DB a livello federale, che semplificasse i compiti di: archiviazione dei dati da par-

[1] per ulteriori approfondimenti teorici, si vedano i riferimenti bibliografici [a], [c] ed [e]

te della stessa Federazione, ricerca di informazioni e dati relativi alla storia dei professionisti iscritti da parte di ogni singolo cittadino interessato alla consultazione, rielaborazioni statistiche e monitoraggio dei professionisti da parte dei vari Ordini Provinciali.

Inoltre, grazie alle possibilità offerte da internet, è nata in tempi abbastanza recenti l'idea di affiancare un'applicazione Web al DB, presente nel sito della Federazione [2], per facilitare la consultazione dei dati. L'applicazione doveva essere suddivisa in due parti: una parte pubblica, che permetteva a qualsiasi utente della rete Internet di accedervi per ottenere informazioni base sui professionisti, quali: nome, cognome, data e luogo di nascita, percorso di studi e abilitazioni; e una parte privata, rivolta solo a chi è in possesso di username e password per dare la possibilità di consultare anche le reportistiche sui professionisti richieste dagli organi della Federazione.

La FNOMCeO ha affidato il progetto della costruzione del DB alla Omicron Technologies nel 2004. Oltre alla costruzione, l'azienda si doveva occupare anche dello sviluppo dell'applicazione Web e, successivmente, della gestione e della manuntenzione del DB e dell'elaborazione dei report sopra citati richiesti dalla Federazione.

L'azienda ha utilizzato un DBMS (DataBase Managment System) basato su Oracle, uno dei DB commerciali più usati. Infatti garantisce un'ottima organizzazione per la sicurezza e la manutenzione dei dati e per la gestione dei privilegi degli utenti. Inoltre, per quanto non sia un

[2] http://www.fnomceo.it

sistema affidabile e potente per la creazione e la gestione delle interrogazioni come MySQL, Oracle è uno dei migliori sistemi per la gestione di grandi quantità di dati e di relazioni. All'interno dell'azienda inoltre, è considerato il sistema più adatto per le operazioni che i tecnici devono eseguire[3].

Omicron Technologies lavora da diversi anni con Oracle, nonostante questo comporti il pagamento di una licenza, installando anche un Server Oracle che contiene tutti i progetti di basi di dati sviluppati fino ad oggi, compreso FNOMCEO.

2.3 Progettazione concettuale del DB

Per tradurre l'analisi dei requisiti in un contesto più sintetico considerando anche tutte le tipologie di dati raccolti dai vari Ordini Provinciali, utilizzeremo come modello di riferimento il modello ER[4]. Il compito sarà la ricostruzione del procedimento che ha portato alla base di dati attuale, chiamato *reverse engineering*, in cui cercheremo di inserire le considerazioni viste precedentemente.

Il database è sviluppato attorno ad un'entità centrale, chiamata *PROFESSIONISTA*, in cui sono presenti alcune informazioni di tipo anagrafico. Collegata a questa si possono individuare dei gruppi di entità che rimandano a degli insiemi comuni:

[3] si veda [4] e [5] per ulteriori approfondimenti su Oracle
[4] Peter Chen, 1976. Si veda anche [a]

- STUDIO: l'insieme raccoglie tutte le informazioni relative al percorso di studi di un professionista, alla laurea, alle specializzazioni e alle abilitazioni;
- REPERIBILITA': l'insieme fa riferimento ai modi in cui è possibile contattare un professionista, tramite indirizzo, cittadinanza, ecc...
- ORDINE: l'insieme raccoglie le informazioni sull'iscrizione di un professionista ad un Ordine, al relativo albo e sugli eventuali provvedimenti e sanzioni presi con le rispettive motivazioni;
- EDUCAZIONE: insieme che riguarda tutti i dati sui corsi di aggiornamento facenti parte del progetto ECM (Educazione Continua in Medicina), non obbligatori, ai quali un professionista partecipa, come il tipo di evento o il numero di crediti, ecc...
- UTENTE: l'insieme fa riferimento ai dati relativi all'applicazione Web e alla traccia delle operazioni effettuate dai professionisti;
- ALTRO: insieme di tutte le entità che non rientrano in nessuno dei precedenti gruppi.

Data la complessità dello schema, è praticamente impossibile disegnare un modello ER con tutte le entità, le relazioni e le cardinalità. Ma per facilitarne la comprensione verrà disegnata solo una parte significativa:

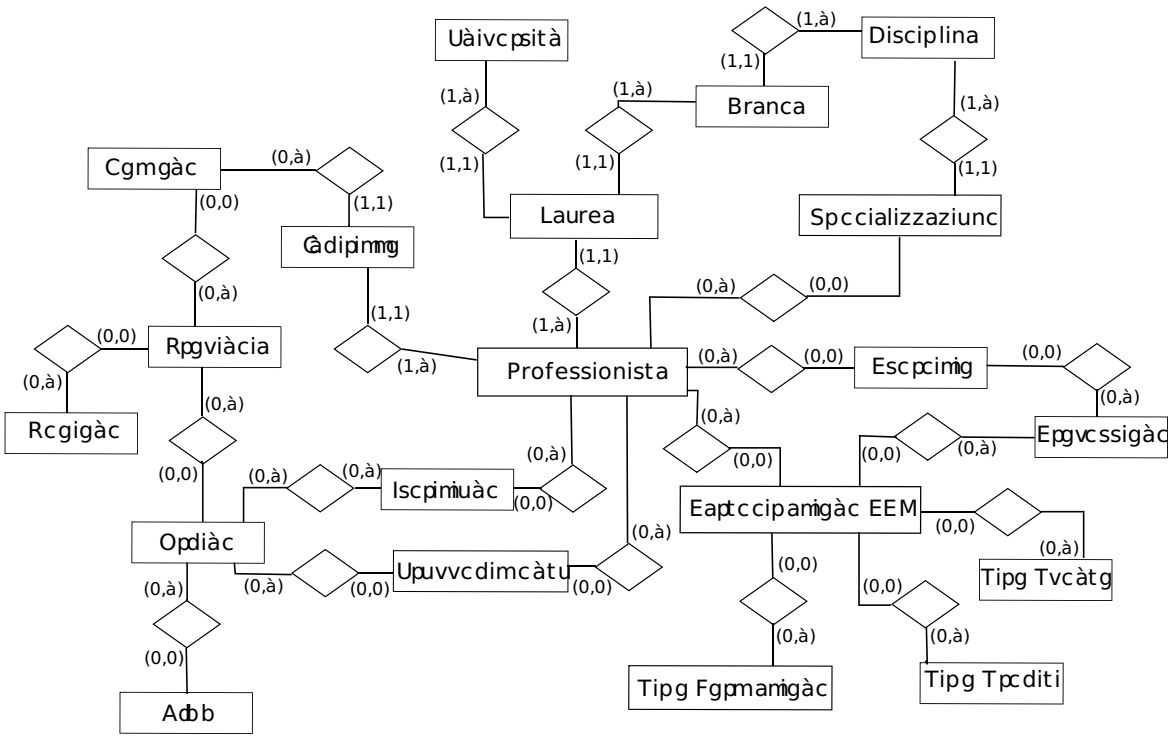


Figura 2.2: Porzione di schema ER del database FNOMCEO

2.4 Progettazione logica del DB

Il prossimo passo da seguire comprende la ristrutturazione del modello ER e la sua traduzione verso il modello relazionale. L'approccio classico si basa innanzitutto sull'analisi delle ridondanze e quindi sull'analisi delle prestazioni del modello ER appena disegnato. L'azienda, però, si è resa conto che spogliare i dati per eliminare eventuali ridondanze consisteva in un notevole spreco di risorse organizzative, economiche e di tempo, considerando il modo scrupoloso con cui i dati venivano (e vengono tuttora) raccolti e anche la probabile perdita di informazioni qualora l'analisi fosse stata applicata.

Ma se da un lato rimanevano integre tutte le informazioni raccolte conservando semplicità e comprensibilità dello schema, dall'altro rimaneva ancora il problema della grande quantità di memoria occupata e, di conseguenza, del tempo impiegato per eseguire qualsiasi operazione, con un equivalente spreco di risorse.

La soluzione adottata a livello organizzativo da parte dell'azienda permette di mantenere tutti i dati e allo stesso tempo risparmiando memoria e tempi di esecuzione delle operazioni, il tutto a discapito però della comprensibilità dello schema. Si tratta della costruzione di alcune tabelle aggiuntive che originariamente non erano comprese nello schema ER di FNOMCEO che rendono più veloci le operazioni eseguite più spesso, come interrogazioni, importazioni e gestioni dei flussi.

Si possono dividere le tabelle aggiuntive in due gruppi, che verranno approfondite meglio più avanti: il primo gruppo raccoglie le tabelle che

si affiancano a delle entità già presenti nello schema. Hanno lo scopo di memorizzare dati storici che difficilmente verranno rielaborati, ma che vengono comunque raccolti per la completezza delle informazioni rendendo di fatto le entità principali più snelle. Queste tabelle vengono definite come tabelle storiche.

Il secondo gruppo, invece, raccoglie le tabelle che non rientrano concettualmente nello schema ER. Esse fungono da supporto per le operazioni viste poco fa e per monitorare le richieste provenienti dall'applicazione Web. Queste tabelle vengono chiamate, appunto, tabelle di supporto.

2.4.1 Tabelle storiche

Come già accennato, le tabelle storiche sono state implementate per raccogliere i dati di una tabella del DB che non vengono più soggetti a operazioni o interrogazioni con una certa frequenza. Tali tabelle sono identiche nel nome con la tabella originale (fatta eccezione per il suffisso *_HISTORY*) e negli attributi, con l'aggiunta eventualmente di un indice. La tabella storica, quindi, conterrà la maggior parte dei dati della tabella originale che non verrano quasi mai richiamati rendendo di fatto la tabella molto pesante. Al contrario, la tabella originale sarà una tabella con poche righe, facile e veloce da consultare.

Comunque, per quanto ci siano due tabelle fisicamente diverse, l'entità rimane unica concettualmente.

2.4.2 Tabelle di supporto

Il sistema, nella sua completezza (ovvero applicazione Web e DB), oltre a memorizzare effettivamente i dati sul DB si occupa anche di gestire determinati processi su di essi. Per controllare meglio i percorsi che fanno i dati all'interno del DB o quando questi vengono richiamati dall'applicazione Web, abbiamo bisogno di ulteriori informazioni relative alle operazioni che eseguiamo nel DB. Le tabelle di supporto, quindi, si occupano del controllo dei flussi di dati e del controllo degli errori interni al sistema.

Le tabelle, memorizzate in FNOMCEO, possono essere divise in tre categorie:

- Tabelle temporanee per l'inserimento dei dati;
- Tabelle temporanee per l'eliminazione dei dati;
- Tabelle per l'esportazione.

Tabelle temporanee per l'inserimento dei dati

Queste tabelle sono indispensabili per poter controllare e, in caso, modificare eventuali informazioni prima di poterle inserire nel DB.

I dati non hanno un metodo di inserimento automatico poiché il modo di raccoglierli varia a seconda dell'organizzazione incaricata, che può essere un Ordine Provinciale come il Ministero della Salute. Successivamente le organizzazioni inviano i dati alla Federazione, che a sua volta li invia all'azienda con una richiesta di inserimento nel DB.

L'azienda ha il compito di verificare che non ci siano errori tramite un tecnico od un sistema automatico, ripulendo da varie incongruenze e riadattando i dati per il loro inserimento [5]. Questa operazione viene eseguita su una tabella temporanea all'interno di un database diverso da FNOMCEO.

Se i dati non presentano errori vengono tranquillamente inseriti nel DB, altrimenti vengono copiati in una tabella temporanea, in cui viene indicata la tipologia di errore, in attesa di un riscontro da parte della Federazione.

Le tabelle temporanee di inserimento possiedono lo stesso nome e gli stessi attributi della relativa tabella originale di riferimento, con l'aggiunta di un suffisso *_TMP* nel nome.

Tabelle temporanee per l'eliminazione dei dati

Quando si ha a che fare con l'eliminazione di una tupla da una relazione bisogna sempre fare determinate considerazioni. La più importante riguarda come quella tupla interagisca con il resto del DB, se possiede qualche chiave esterna e se eliminandola non si provochi qualche reazione a catena che comprometterebbe la stabilità del DB. Inoltre, l'eliminazione di dati da un database informativo come FNOMCEO non è mai consigliabile per la perdita in qualità dell'informazione, ma a volte risulta necessaria. Ad esempio il DB contiene i dati relativi ai professionisti attivi nella tabella *PROFESSIONISTA*, cioè che esercitano la professio-

[5] per un esempio approfondito si veda il Capitolo 3

ne. Se uno di questi cessa la sua attività dev'essere rimosso dalla tabella il record con i suoi dati.

Per garantire una maggiore sicurezza durante l'eliminazione di una tupla, si procede in due passi. Il primo consiste nello spostamento della tupla in una tabella temporanea per l'eliminazione. La tabella è identica in tutto e per tutto alla tabella madre, tranne per il nome, in cui viene aggiunto il suffisso *_TO_DELETE*.

Il secondo passo consiste nell'effettiva eliminazione della tupla da parte di un tecnico o automaticamente da parte del sistema.

Tuttavia, nelle tabelle storiche il dato eliminato rimane per una sicurezza maggiore, per quanto nella tabella principale non ci sia più.

Tabelle per l'esportazione

I dati presenti nel DB possono essere soggetti ad esportazioni per dei motivi di manutenzione dei Server che ospitano i DB aziendali. La procedura di esportazione è automatica. Per questo risulta molto utile tenere traccia degli spostamenti che fanno i dati e degli errori che possono verificarsi. In FNOMCEO i dati vengono esportati tramite file XML, per via della compatibilità di questo formato con qualsiasi tipo di sistema di destinazione, come definito dal W3C [6].

Nel caso di un'operazione di esportazione tramite XML possiamo tenere traccia del nome del file contenente i dati, del tempo impiegato

[6] World Wide Web Consortium

per l'elaborazione, degli errori che eventualmente vengono commessi e dell'esito dell'operazione al termine.

Inoltre, XML (eXtensible Markup Language) è un metalinguaggio usato per definire un linguaggio per descrivere dei documenti strutturati, quali possono essere proprio le tabelle di un database. Per questa ragione è ampiamente utilizzato dalle aziende che operano in questo settore per l'esportazione dei dati[7].

2.5 Riepilogo delle tabelle presenti nel DB

Una volta terminate tutte le fasi della progettazione, siamo in possesso delle varie tabelle che definiscono il DB e che vengono popolate dei dati inviati dalla Federazione. Le tabelle sono state ordinate in una tabella per poter vedere alcune delle loro caratteristiche, come le loro funzioni, le strategie e le frequenze di aggiornamento e le operazioni fondamentali su ognuna di esse.

Le funzioni svolte da una tabella all'interno del DB possono essere di tre tipi: *archiviazione*, quando abbiamo a che fare con la memorizzazione di nuovi dati, *data transfer*, quando abbiamo informazioni su dinamiche di trasferimento dei dati e *informazione data*, quando abbiamo delle informazioni che non sono soggette a modifiche.

A prescindere da quali sono le funzioni principali di una tabella ci sono diverse strategie di aggiornamento per le tabelle: *insert*, quando i record vengono inseriti nella tabella oppure modificati o cancellati,

[7] si veda [e] e soprattutto il capitolo 6 per approfondimenti su XML

insert only, quando i record possono solo essere inseriti, *reload*, quando i dati in una tabella vengono ricaricati e aggiornati, *truncate*, quando c'è la necessità di definire una tabella storica, e *no update*, quando la tabella non viene mai aggiornata.

NOME TABELLA	STRATEGIA UPDATE	FREQUENZA DI AGGIORNAMENTO	FUNZIONE TABELLA	EVENTO DI CRESCITA
ABILITAZIONE	insert/ reload/ truncate	mensile	archiviazione	nuova abilitazione
ABILITAZIONE_HISTORY	insert only	annuale	archiviazione	
ACCREDITATORE	insert/ reload	su richiesta	informazione data	nuovo accreditatore
ALBO	no update	su richiesta	informazione data	
ALTRO	insert/ reload	mensile	archiviazione	nuova convenzione SSN [8]
ALTRO_HISTORY	insert only	annuale	archiviazione	
ATTIVITA_PROFESSIONALE	insert/ reload	mensile	archiviazione	nuova attività
BRANCA	no update		informazione data	
CAUSALE_CANCELLAZIONE	insert/ reload	mensile	archiviazione	cancellazione professionista
CAUSALE_PROVVEDIMENTO	insert/ reload	mensile	archiviazione	nuovo provvedimento
CAUSALE_RICONOSCIMENTO	insert/ reload	mensile	archiviazione	nuovo riconoscimento
CITTADINANZA	insert/ reload/ truncate	mensile	archiviazione	nuovo professionista
CITTADINANZA_HISTORY	insert only	annuale	archiviazione	
CODICE_DEONTOLOGICO	no update		informazione data	
COMUNE	insert only	su richiesta	informazione data	nuovo comune
COMUNE_CAP	insert only	su richiesta	informazione data	nuovo comune
CONTATTO	insert / reload/ truncate	mensile	archiviazione	nuovo contatto
CONTATTO_HISTORY	insert only	annuale	archiviazione	
DISCIPLINA	insert only	su richiesta	informazione data	nuova disciplina
DISCIPLINA_ECM	no update		informazione data	
ELENCO_SPECIALE	insert/ delete	mensile	archiviazione	nuovo riferimento legislativo
ESENZIONE	insert/ reload/ truncate	mensile	archiviazione	nuova esenzione
ESENZIONE_HISTORY	insert only	annuale	archiviazione	

ESENZIONE_TIPO	insert/ reload	mensile	archiviazione	nuovo tipo di esenzione
ESERCIZIO	insert/ reload	mensile	archiviazione	nuovo esercizio
ESERCIZIO_HISTORY	insert only	annuale	archiviazione	
EXPORT_XML	insert/ delete	su richiesta	data transfer	nuova esportazione
FILE_DATA	insert/ delete	su richiesta	data transfer	caricamento file
IMPORT_CODICE_FISCALE	insert/ delete	su richiesta	data transfer	
INDIRIZZO	insert/ reload	mensile	archiviazione	nuovo professionista
INDIRIZZO_HISTORY	insert only	annuale	archiviazione	
ISCRIZIONE	insert/ reload/ truncate	mensile	archiviazione	nuova iscrizione
ISCRIZIONE_HISTORY	insert only	annuale	archiviazione	
ISTITUTO_PSICOTERAPIA	insert only	su richiesta	informazione data	nuovo istituto
LAUREA	insert/ reload/ truncate	mensile	archiviazione	nuova laurea
LAUREA_HISTORY	insert only	annuale	archiviazione	
LOG_FILE	insert/ delete	su richiesta	data transfer	caricamento file
LOG_PROF_ERROR	insert/ delete	su richiesta	data transfer	nuova professionista
ORDINE	insert/ reload	su richiesta	informazione data	nuovo ordine
PARTECIPAZIONE_ECM	insert/ reload/ truncate	annuale	archiviazione	nuova partecipazione
PARTECIPAZIONE_ECM_HISTORY	insert only	annuale	archiviazione	
POSTA_MASSIVA	insert/ reload	mensile	archiviazione	nuova posta massiva
PRIVACY	insert/ reload/ truncate	mensile	archiviazione	nuovo professionista
PRIVACY_HISTORY	insert only	annuale	archiviazione	
PROFESSIONE	insert only	su richiesta	informazione data	nuova professione
PROFESSIONISTA	insert/ delete/ truncate	mensile	archiviazione	nuovo professionista
PROFESSIONISTA_ELIMINATO	insert only	su richiesta	archiviazione	eliminazione professionista
PROFESSIONISTA_HISTORY	insert only	annuale	archiviazione	
PROFESSIONISTA_TO_DELETE	insert/ delete	su richiesta	data transfer	professionista da eliminare
PROFESSIONISTA_VARIATO	insert/ delete	su richiesta	data transfer	modifica professionista
PROFESSIONISTA_VARIATO_BUG	insert/ delete	su richiesta	data transfer	modifica professionista
PROFILO	insert/ reload	mensile	archiviazione	nuovo professionista
PROF_ES_ISCR	insert/ reload/ truncate	mensile	archiviazione	nuova iscrizione esercizio

PROF_ES_ISCR_HISTORY	insert only	annuale	archiviazione	
PROVINCIA	no update		informazione data	
PROVVEDIMENTO	insert/ reload/ truncate	mensile	archiviazione	nuovo provvedimento
PROVVEDIMENTO_HISTORY	insert only	annuale	archiviazione	
PROVV_COD_DEONT	insert/ reload/ truncate	mensile	archiviazione	nuovo provvedimento deontologico
PROVV_COD_DEONT_HISTORY	insert only	annuale	archiviazione	
REGIONE	no update		informazione data	
SPECIALIZZAZIONE	insert/ reload/ truncate	mensile	archiviazione	nuova specilizzazione
SPECIALIZZAZIONE_HISTORY	insert only	annuale	archiviazione	
STATO	insert only	su richiesta	informazione data	nuovo stato
TIPO_CREDITI	insert/ reload	su richiesta	informazione data	nuovo tipo di crediti
TIPO_EVENTO	insert/ reload	su richiesta	informazione data	nuovo tipo di evento
TIPO_FORMAZIONE	insert/ reload	su richiesta	informazione data	nuovo tipo di formazione
UNIVERSITA	insert only	su richiesta	informazione data	nuova università
UTENTE	insert/ delete	mensile	archiviazione	nuovo utente
UTENTE_PROFILO	insert/ delete	mensile	archiviazione	nuovo utente
ZONA_GEOGRAFICA	no update		informazione data	

Tabella 2.1: Database FNOMCEO

[7]Servizio Sanitario Nazionale

Capitolo 3

Creazione di report su FNOMCEO

Uno dei compiti affidati all'azienda, oltre alla gestione e alla manutenzione del DB, è l'impostazione e la creazione di vari report secondo le indicazioni fornite dalla Federazione.

Gli scopi dei report sono essenzialmente due: il primo è di fornire informazioni di riepilogo utili alla Federazione per vedere l'evoluzione della situazione dei professionisti, mentre il secondo è di essere disponibili alla visualizzazione on-line da parte degli utenti iscritti al sito Web della FNOMCeO e che ne fanno richiesta sul Web-Server.

Però una delle cose che restano da capire è cosa siano di preciso i report.

3.1 Definizioni su report e viste

I report possono essere definiti come delle viste ad alto livello, che occupano poco spazio di memoria, in cui gli utenti hanno poche informazioni molto chiare.

Prendendo in esame un qualsiasi DB, si può definire anche cos'è una vista: si tratta di una tabella le cui righe non sono esplicitamente memorizzate nella base di dati, ma sono calcolati da essa quando sono necessarie, in base ad una definizione di vista. Ovvero può essere pensata come una tabella contenente il risultato di una query di interrogazione.

Inoltre, la vista può essere memorizzata nel DB come tabella aggiuntiva e può essere usata a sua volta per definire nuove interrogazioni e nuove viste.

Oltre ad una facilitazione nell'impostazione di query di interrogazione complesse, le viste sono utili per la gestione del modo in cui gli utenti (che siano tecnici di basso livello o utenti on-line) vedono i dati. Nel contesto della sicurezza, infatti, possono essere definite viste che danno ad un gruppo di utenti accesso solo alle informazioni che è permesso loro vedere. Essi non devono preoccuparsi della distinzione tra viste e tabelle di base, ma solo dei dati che gli si presentano davanti agli occhi.

Le distinzioni, però, sono necessarie quando si parla degli aggiornamenti sulle viste e sulle tabelle del DB. Una modifica, o peggio, l'eliminazione di righe all'interno di una vista può provocare importanti cambiamenti all'interno del DB, fino ad un suo collasso concettuale[1].

Negli anni si è cercato di definire degli standard SQL anche per la definizione e gli aggiornamenti delle viste[2]. Nel dettaglio:

- SQL-92: aggiornamenti solo su viste ricavate da singole tabelle, usando gli operatori di selezione e proiezione dell'algebra relazio-

[1] ulteriori informazione sulle viste su [a],[b] ed [e]

[2] gli standard sono disponibili (a pagamento) sul sito www.iso.org

nale (estrapolazione di righe e colonne da una tabella);

- SQL-99: implementazione dell'aggiornamento su viste ricavate da più tabelle a patto che nella vista siano presenti le chiavi primarie delle tabelle di base;

- SQL-2003: definizione e creazione di viste da altre viste, ma non è ancora possibile utilizzare l'operatore ORDER BY.

3.2 Un compromesso: il Flat-Database

Come accennato precedentemente, uno degli utilizzi dei report è la loro reperibilità on-line.

La memorizzazione di dati o l'esecuzione di una query nel DB sono processi che impiegano molto tempo ad essere completati: si passa da operazioni di circa un minuto fino a quelle che ci impiegano mezz'ora! Un tempo di caricamento così eccessivo per una pagina Web dopo una richiesta ad un Web-Server non è sicuramente ammissibile.

Inoltre si può assumere che la velocità di caricamento della pagina richiesta dipenda esclusivamente dalla velocità di evasione delle richieste da parte del Web-Server: i dati richiesti al sistema, in questo caso, da parte del Client Web sono di tipo testuale e le velocità di connessione con le tecnologie attuali (come ADSL, fibre ottiche, ecc...) possono rendere trascurabile il tempo di trasferimento attraverso i canali di comunicazione.

Per ovviare a questo problema, gli archivi informatici di grosse dimensioni collegati ad applicazioni Web, come nel caso di FNOMCEO, si appoggiano su delle strutture ausiliarie. Si tratta di basi di dati che non prevedono connessioni tra più relazioni né presenza di dati multidimensionali, ma bensì di DB composti da una sola tabella, contenente i campi d'interesse e i rispettivi record. Il nome di questa struttura è Flat-Database (Flat-DB) e deriva dall'appiattimento del DB per via della bidimensionalità.

I dati sono memorizzati nella tabella così come dovranno poi essere mostrati agli utenti che ne fanno richiesta. Ed è per questo che all'interno del Web Server saranno presenti più Flat-DB, per rispondere alle diverse richieste giunte all'applicazione e più velocemente.

Le tabelle all'interno dei Flat-DB non saranno altro che i report come richiesti dalla Federazione e inseriti in una forma strutturata.

Il Flat-DB è quindi un ottimo compresso tra la velocità richiesta dai Web-Server e dalle applicazioni Web e la completezza di informazioni tipica dei grandi DB come FNOMCEO[3].

Per la creazione e la gestione dei Flat-DB viene utilizzato MySOL poichè ha una marcia in più rispetto ai concorrenti che lo rende adatto a questo tipo di operazioni: la query-cache. Si tratta di una cache di memoria che permette di annullare i tempi di richiesta di una query nel caso in cui essa venga ripetutamente richiesta. La query viene memorizzata all'interno della cache e se corrisponde con la query appena richiesta

[3]definizioni e informazioni sui Flat-DB sono state trovate su [2] e, in generale, sul Web

presenta subito il risultato, senza doverlo ricalcolare.

3.3 Creazione dei report

La creazione di un report non è sempre una cosa facile e strutturata da fare. Il numero di relazioni coinvolte in un singolo report può essere molto alto e quindi l'impostazione della query può risultare molto difficile. Un altro problema può essere organizzare l'output della query richiesta esattamente come vuole la Federazione oppure come deve essere inserita nel Flat-DB. Infine il tempo di esecuzione di una query può essere molto alto, se questa è stata costruita facendo riferimento a molte relazioni in una volta sola.

Il metodo migliore per affrontare il problema è quello di impostare la query in mini-query più semplici a carico computazionale basso, in modo da poter arrivare in breve tempo ad una sola query complicata che fornisce i risultati richiesti. Per fare tutto questo oltre alle relazioni presenti nel DB sarà neccessario far uso di viste intermedie create ad hoc per raggiungere lo scopo.

Di seguito verranno mostrati alcuni report finali creati, divisi per categoria di report, con il rispettivo codice implementato per arrivare alla tabella risultante.

3.3.1 Esempio di report per i Flat-DB

REPORT_SPECIALIZZAZIONE

Il report ritorna il numero di specializzazioni dei professionisti divisi per Codice CEE e per provincia. Viene estrapolato da numerose relazioni: PROFESSIONISTA, ISCRIZIONE, SPECIALIZZAZIONE, REGIONE, PROVINCIA, BRANCA. Il risultato finale del report non verrà visualizzato in quanto la tabella ha una dimensione eccessiva.

```
CREATE OR REPLACE VIEW V_PROF_ATTIVI (ID, SESSO) AS
SELECT ID, SESSO
FROM PROFESSIONISTA
WHERE ATTIVO = '1';
COMMENT ON TABLE V_PROF_ATTIVI IS 'Ritorna l''elenco dei professionisti attivi';

CREATE OR REPLACE VIEW V_ISCRITTI_X_SESSO (PROVINCIA_ID,
PROFESSIONISTA_ID, SESSO) AS
SELECT ISCRIZIONE.PROVINCIA_ID, ISCRIZIONE.PROFESSIONISTA_ID,
V_PROF_ATTIVI.SESSO
FROM ISCRIZIONE INNER JOIN V_PROF_ATTIVI
ON V_PROF_ATTIVI.ID = ISCRIZIONE.PROFESSIONISTA_ID;

CREATE OR REPLACE VIEW V_SPECIALIZZAZIONI (PROVINCIA_ID,
BRANCA_ID, QUANTE) AS
SELECT V_ISCRITTI_X_SESSO.PROVINCIA_ID,
SPECIALIZZAZIONE.BRANCA_ID, COUNT(*) AS QUANTE
FROM V_ISCRITTI_X_SESSO JOIN SPECIALIZZAZIONE
ON V_ISCRITTI_X_SESSO.PROFESSIONISTA_ID
= SPECIALIZZAZIONE.PROFESSIONISTA_ID
GROUP BY BRANCA_ID, PROVINCIA_ID;

CREATE OR REPLACE VIEW V_SPECIALIZZAZIONI_PROVINCIA
(BRANCA_ID, QUANTE, DESCRIZIONE, ZONA_GEOGRAFICA_DESCRIZIONE,
ZONA_GEOGRAFICA_ID, PROVINCIA) AS
SELECT V_SPECIALIZZAZIONI.BRANCA_ID, V_SPECIALIZZAZIONI.QUANTE,
REGIONE.DESCRIZIONE, REGIONE.ZONA_GEOGRAFICA_DESCRIZIONE,
REGIONE.ZONA_GEOGRAFICA_ID, PROVINCIA.DESCRIZIONE AS PROVINCIA
FROM PROVINCIA, REGIONE, V_SPECIALIZZAZIONI
WHERE REGIONE.ID = PROVINCIA.REGIONE_ID
AND PROVINCIA.ID = V_SPECIALIZZAZIONI.PROVINCIA_ID;

CREATE OR REPLACE VIEW V_REPORT_SPECIALIZZAZIONI_ALL
(QUANTE, REGIONE, ZONA_GEOGRAFICA_DESCRIZIONE, ZONA_GEOGRAFICA_ID,
PROVINCIA, CODICE, DESCRIZIONE, CODICE_CEE) AS
SELECT V_SPECIALIZZAZIONI_PROVINCIA.QUANTE,
V_SPECIALIZZAZIONI_PROVINCIA.DESCRIZIONE AS REGIONE,
```

```
V_SPECIALIZZAZIONI_PROVINCIA.ZONA_GEOGRAFICA_DESCRIZIONE,
V_SPECIALIZZAZIONI_PROVINCIA.ZONA_GEOGRAFICA_ID,
V_SPECIALIZZAZIONI_PROVINCIA.PROVINCIA,
BRANCA.CODICE, BRANCA.DESCRIZIONE, BRANCA.IDPADRE
FROM V_SPECIALIZZAZIONI_PROVINCIA, BRANCA
WHERE BRANCA.ID = V_SPECIALIZZAZIONI_PROVINCIA.BRANCA_ID;

CREATE OR REPLACE VIEW V_SPECIALIZZAZIONI_CEE
(QUANTE, REGIONE, ZONA_GEOGRAFICA_DESCRIZIONE,
 ZONA_GEOGRAFICA_ID, PROVINCIA, CODICE_CEE) AS
SELECT SUM(QUANTE) AS QUANTE, REGIONE,
ZONA_GEOGRAFICA_DESCRIZIONE, ZONA_GEOGRAFICA_ID,
PROVINCIA, CODICE_CEE
FROM V_REPORT_SPECIALIZZAZIONI_ALL
GROUP BY REGIONE, ZONA_GEOGRAFICA_DESCRIZIONE,
ZONA_GEOGRAFICA_ID, PROVINCIA, CODICE_CEE;

CREATE OR REPLACE VIEW REPORT_SPECIALIZZAZIONI
(QUANTE, REGIONE, ZONA_GEOGRAFICA, ZONA_GEOGRAFICA_ID,
 PROVINCIA, CODICE_CEE, DESCRIZIONE) AS
SELECT QUANTE, REGIONE, ZONA_GEOGRAFICA_DESCRIZIONE,
ZONA_GEOGRAFICA_ID, PROVINCIA,
V_SPECIALIZZAZIONI_CEE.CODICE_CEE,
BRANCA.DESCRIZIONE
FROM V_SPECIALIZZAZIONI_CEE, BRANCA
WHERE V_SPECIALIZZAZIONI_CEE.CODICE_CEE = BRANCA.CODICE;
COMMENT ON TABLE REPORT_SPECIALIZZAZIONI
IS 'Ritorna␣il␣numero␣di␣specializzazioni␣per␣CODICE_CEE␣e␣per␣PROVINCIA';
```

3.3.2 Esempio di report di riepilogo per la Federazione

EVENTI_PROFESSIONE_NEW

Il report mostra il numero di eventi e progetti a cui hanno presenziato i professionisti, ordinati per professione. Viene estrapolato dalle relazioni PROFESSIONE e PARTECIPAZIONE_ECM.

```
CREATE OR REPLACE VIEW V_PART_EVENTI (PROFESSIONE_ID, NUMERO_EVENTI)
AS SELECT PROFESSIONE_ID, COUNT(TIPO_EVENTO_VALORE)
FROM PARTECIPAZIONE_ECM
WHERE TIPO_EVENTO_VALORE = 'E'
GROUP BY PROFESSIONE_ID
ORDER BY PROFESSIONE_ID

CREATE OR REPLACE VIEW V_PART_PROG (PROFESSIONE_ID, NUMERO_PROGETTI)
AS SELECT PROFESSIONE_ID, COUNT(TIPO_EVENTO_VALORE)
FROM PARTECIPAZIONE_ECM
WHERE TIPO_EVENTO_VALORE = 'P'
GROUP BY PROFESSIONE_ID
ORDER BY PROFESSIONE_ID

SELECT PROFESSIONE.DESCRIZIONE AS PROFESSIONE,NUMERO_EVENTI, NUMERO_PROGETTI
FROM V_PART_EVENTI INNER JOIN V_PART_PROG
ON V_PART_PROG.PROFESSIONE_ID = V_PART_EVENTI.PROFESSIONE_ID
INNER JOIN PROFESSIONE ON PROFESSIONE.ID = V_PART_EVENTI.PROFESSIONE_ID
ORDER BY PROFESSIONE.DESCRIZIONE
```

PROFESSIONISTI_30CREDITI_2008

Il report mostra il numero di professionisti che hanno superato la quota di 30 crediti nel 2008 divisi per professione. Inoltre mostra anche qual è la percentuale di professionisti sul totale di quella professione ad aver superato la quota. Viene estrapolato dalle relazioni PROFESSIONE e PARTECIPAZIONE_ECM.

```
CREATE OR REPLACE VIEW PROFESSIONISTA_30CREDITI_2008 AS
SELECT PROFESSIONE .DESCRIZIONE , NUM_30_CREDITI ,
ROUND( NUM_30_CREDITI / NUMERO_PROFESSIONISTI_TOT , 2) * 100 AS PERCENTUALE
FROM (
SELECT PROFESSIONE_ID , COUNT( PROFESSIONISTA_CODICE_FISCALE )
AS NUM_30_CREDITI , NUMERO_PROFESSIONISTI_TOT
```

```
FROM (
SELECT PROFESSIONISTA_CODICE_FISCALE , PROFESSIONE_ID , SUM( NUMERO_CREDITI )
AS CREDITI_ACCUMULATI
FROM PARTECIPAZIONE_ECM
WHERE PARTECIPAZIONE_ECM .CORSO_ECM LIKE '2008% '
GROUP BY PROFESSIONISTA_CODICE_FISCALE , PROFESSIONE_ID
HAVING SUM( NUMERO_CREDITI ) > 30) JOIN (
SELECT PROFESSIONE_ID AS PROFESSIONE , COUNT( PROFESSIONISTA_CODICE_FISCALE )
AS NUMERO_PROFESSIONISTI_TOT
FROM (
SELECT PROFESSIONISTA_CODICE_FISCALE , PROFESSIONE_ID , SUM( NUMERO_CREDITI )
AS CREDITI_ACCUMULATI
FROM PARTECIPAZIONE_ECM
WHERE PARTECIPAZIONE_ECM .CORSO_ECM LIKE '2008% '
GROUP BY PROFESSIONISTA_CODICE_FISCALE , PROFESSIONE_ID)
GROUP BY PROFESSIONE_ID
) ON PROFESSIONE = PROFESSIONE_ID
GROUP BY PROFESSIONE_ID , NUMERO_PROFESSIONISTI_TOT
ORDER BY PROFESSIONE_ID ASC
) INNER JOIN PROFESSIONE ON PROFESSIONE .ID = PROFESSIONE_ID
ORDER BY PROFESSIONE .DESCRIZIONE ASC;
```

Tabelle con i risultati

Tabella 3.1: *EVENTI_PROFESSIONE_NEW*

PROFESSIONE	**NUMERO EVENTI**	**NUMERO PROGETTI**
Assistente sanitario	1.353	2.213
Biologo	7.748	3.825
Chimico	1.989	873
Dietista	677	255
Educatore professionale	502	308
Farmacista	7.368	1.202
Fisico	88	79
Fisioterapista	6.579	2.865
Igienista dentale	318	39
Infermiere	20.612	17.749
Infermiere pediatrico	1.421	2.629
Logopedista	1.386	578
Medico chirurgo	98.793	18.716
Odontoiatra	8.680	185
Odontotecnico	416	6
Ortottista/Assistente di oftalmologia	550	241
Ostetrica/o	2.205	3.205
Ottico	854	7
Podologo	198	4
Psicologo	8.307	3.592
Tecnico audiometrista	248	80
Tecnico audioprotesista	316	8
Tecnico della fisiopatologia cardiocircolatoria e perfusione cardiovascolare	121	76
Tecnico della prevenzione nell'ambiente e nei luoghi di lavoro	627	278
Tecnico di neurofisiopatologia	106	39
Tecnico educazione e riabilitazione psichiatrica e psicosociale	114	40
Tecnico ortopedico	514	6
Tecnico sanitario di radiologia medica	2.003	3.597
Tecnico sanitario laboratorio biomedico	2.050	1.634
Terapista della neuro e psicomotricità dell'età evolutiva	615	131
Terapista occupazionale	147	46
Tutte le professioni	5.370	2.389
Veterinario	2.591	954

Tabella 3.2: *PROFESSIONISTI_30CREDITI_2008*

PROFESSIONE	**PROFESSIONISTI CON PIU' DI 30 CREDITI**	**PERCENTUALE**
Assistente sanitario	38	5,35
Biologo	2535	24,77
Chimico	52	15,29
Dietista	35	10,61
Educatore professionale	9	14,06
Farmacista	3812	17,02
Fisico	2	10,53
Fisioterapista	1907	37,50
Igienista dentale	22	3,93
Infermiere	993	9,50
Infermiere pediatrico	98	5,05
Logopedista	183	28,96
Medico chirurgo	57223	26,25
Odontoiatra	6930	25,17
Odontotecnico	13	12,15
Ortottista/Assistente di oftalmologia	33	10,38
Ostetrica/o	332	12,02
Ottico	30	54,55
Podologo	23	13,07
Psicologo	1386	19,79
Tecnico audiometrista	5	6,25
Tecnico audioprotesista	82	27,42
Tecnico della fisiopatologia cardiocircolatoria e perfusione cardiovascolare	1	2,08
Tecnico della prevenzione nell'ambiente e nei luoghi di lavoro	8	11,11
Tecnico di neurofisiopatologia	2	7,69
Tecnico educazione e riabilitazione psichiatrica e psicosociale	5	20,83
Tecnico ortopedico	82	23,63
Tecnico sanitario di radiologia medica	611	10,66
Tecnico sanitario laboratorio biomedico	40	7,84
Terapista della neuro e psicomotricità dell'età evolutiva	115	36,51
Terapista occupazionale	8	24,24
Veterinario	654	14,35

3.4 Creazione del Flat-DB

Una volta creati i report è necessario inserirli nelle tabelle che andranno a formare i vari Flat-DB per l'applicazione Web. I nomi delle tabelle e degli attributi saranno gli stessi dei nomi dei report e delle rispettive colonne per comodità e per la facilità di comprensione. Inoltre è opportuno fare attenzione ai vincoli sui datatype (campi): la definizione dei campi e le interrogazioni per formare i report su essi vengono eseguite in un database Oracle, mentre nei nuovi Flat-DB vengono definiti dei campi in ambiente MySQL.

Invece i vincoli sulle chiavi primarie e sulla presenza di valori NULL non sono necessari nella definizione dei Flat-DB. Gli output ottenuti dai report sono già delle elaborazioni di dati corretti, puliti e ordinati quando questi sono stati inseriti in FNOMCEO.

Per quanto riguarda il popolamento del Flat-DB, gli unici problemi riguardano lo spostamento dei dati ottenuti dai report da un database all'altro, soprattutto perchè è l'unica operazione che verrà ripetuta più volte, a seconda delle indicazioni della Federazione per gli aggiornamenti.

Il sistema scelto in azienda prevede l'utilizzo di file di testo .CSV [4], un formato usato per l'importazione e l'esportazione di tabelle di dati. In questo modo è possibile compattare le tabelle dei report in unico file contenente solo i dati ottenuti separati dal carattere ',' e dal carattere di

[4]Comma-Separated Values

file linea 'CRLF' [5].

Successivamente, con l'ausilio di un programma di Client-SQL con un'interfaccia grafica, è possibile gestire in automatico il popolamento delle tabelle tramite i file .CSV.

[5] codifica UTF-16: Unicode Transformation Format, 16 bit

Capitolo 4

Importazione dati in FNOMCEO: corsi di aggiornamento

L'inserimento di nuovi dati all'interno del DB FNOMCEO copre più o meno tutti i vari settori individuati nell'analisi dei requisiti (si veda il Capitolo 1). Per questo motivo la mole di dati che l'azienda ha l'obbligo di trattare per l'inserimento è immensa e purtroppo, come verrà spiegato in questo capitolo, non automatizzata nella maggior parte dei casi.

Durante il periodo di stage, è stato chiesto all'azienda di supervisionare l'importazione dei dati in una specifica tabella, *PARTECIPAZIONE_ECM*, che verrà presa da esempio per spiegare come si svolge questo procedimento.

4.1 ECM: Educazione Continua in Medicina

I professionisti hanno la facoltà di seguire dei corsi di aggiornamento professionale, siano essi semplici eventi come le conferenze o progetti

di ricerca. Inoltre, hanno completa libertà nella scelta dei corsi.

Per strutturare e verificare l'utilità dei corsi, il Ministero della Salute ha istituito nel 1992 il programma ECM - Educazione Continua in Medicina.

Il progetto comprende l'insieme organizzato e controllato di tutte le attività formative con lo scopo di mantenere elevata e al passo dei tempi la professionalità degli Operatori della Sanità, ovvero dei professionisti che partecipano alle attività. Dall'introduzione del programma, i professionisti devono maturare un minimo di crediti formativi annuali, pur continuando ad avere libera scelta sui corsi da seguire. La Commissione a cui è stata affidata l'elaborazione del programma ECM deve definire questi crediti, definiti come una misura del tempo e dell'impiego che ogni Operatore della Sanità ha dedicato annualmente all'aggiornamento[1].

La tabella *PARTECIPAZIONE_ECM* contiene tutti i dati relativi alle partecipazioni dei professionisti a queste attività, oltre ad essere la tabella di destinazione dell'importazione. Ecco qui di seguito i vari attributi che compongono la tabella:

[1] ulteriori informazioni sul progetto su [6]

Tabella 4.1: *PARTECIPAZIONE_ECM*

Nome Attributo	**Descrizione**
ID	chiave primaria, è un numero progressivo generato automaticamente
CORSO_ECM	sigla del corso
NUMERO_CREDITI	numero dei crediti del corso
DATA_INIZIO	data di inizio del corso
DATA_FINE	data di fine del corso
CODICE	codice del corso assegnato dal Ministero
CODICE_EDIZIONE	numero corrispondente all'edizione del corso
CODICE_ORGANIZZATORE	codice dell'ente organizzatore del corso
PROFESSIONISTA_CODICE_FISCALE	codice fiscale del partecipante al corso, chiave esterna su *PROFESSIONISTA*
DISCIPLINA_ECM_ID	codice della disciplina, chiave esterna su *DISCIPLINA_ECM*
PROFESSIONE_ID	codice della professione, chiave esterna su *PROFESSIONE*
ACCREDITATORE_ID	identificatore dell'accreditatore al corso
TIPO_EVENTO_VALORE	tipo di evento ('E' o 'P'), chiave esterna su *TIPO_EVENTO*
TIPO_CREDITI_VALORE	tipo di crediti assegnati ('P','D','T','R'), chiave esterna su *TIPO_CREDITI*
TIPO_FORMAZIONE_VALORE	tipo di formazione (default 'FOR'), chiave esterna su *TIPO_FORMAZIONE*
FORZATA	attributo che indica l'eventuale forzatura nell'immissione del record

4.2 Procedimento di importazione

Ancora una volta la quantità dei dati in arrivo dal Ministero della Salute è enorme. E nonostante i dati siano strutturati all'incirca come verranno inseriti nella tabella *PARTECIPAZIONE_ECM*, il processo non può essere completamente automatizzato.

Infatti, gli operatori che hanno lo scopo di raccogliere i dati durante gli eventi formativi, e che devono passarli al Ministero della Salute per aggiornare il numero dei crediti, non hanno nessuno schema organizzato da seguire. I dati, quindi, possono risultare incompleti o sbagliati. Tra gli errori più comuni ci sono i codici fiscali sbagliati o mancanti, oppure nomi e cognomi dei professionisti invertiti.

Per questo, le varie fasi da completare nel processo di inserimento devono sempre avere una supervisione tecnica umana.

Le fasi, in dettaglio, sono:

1. Rielaborazione dei dati inviati dal Ministero: pulizia, completamento e uniformazione;

2. Riconoscimento e codifica degli eventuali errori trovati al loro interno;

3. Inserimento dei dati nel DB FNOMCEO.

Nel caso in cui i dati non superino i primi due punti essi vengono lasciati momentaneamente da parte o cancellati a seconda dei casi.

Inoltre, per motivi di sicurezza, i dati da manipolare vengono copiati e memorizzati in un DB secondario, denominato COGEAPS_TMP[2]. Se ci dovessero essere errori durante la manipolazione dei dati, c'è la possibilità di recuperare gli originali in qualsiasi momento. Invece, se le operazioni sono andate a buon fine e si è raggiunto il risultato atteso, i dati elaborati vengono inseriti in FNOMCEO e prontamente cancellati da COGEAPS_TMP.

Per ogni fase, comunque, c'è una determinata documentazione da seguire. La documentazione è un concetto importante per qualsiasi sistema informativo. Essa descrive i processi standard da seguire, ovvero l'applicazione dell'algoritmo di pulizia dei dati, in questo caso, in un linguaggio comprensibile e interpretabile univocamente da un qualsiasi tecnico che deve lavorare sul DB[3].

Omicron Technologies ha salvato la documentazione relativa a FNOMCEO direttamente sul Web-Server aziendale e reperibile tramite un qualsiasi browser Web all'intero di un'apposita area riservata.

4.3 Analisi dei dati ricevuti

Qui di seguito vengono presentate alcune considerazioni riguardo ai dati ricevuti in azienda.

I dati spediti dal Ministero arrivavano fino a circa un anno fa in due file separati. Il primo file conteneva e contiene tuttora le partecipazioni

[2]Il CoGeAPS - Consorzio Gestione Anagrafica Professioni Sanitarie è un organismo con lo scopo di gestire e certificare i crediti formativi del progetto ECM. Si veda anche [8]

[3]si veda [b] per approfondimenti sull'inserimento dei dati

agli Eventi e ai Progetti Formativi Aziendali (PFA) in una struttura comune e in formato .MDB (Microsoft Database Access©) e contengono le seguenti informazioni:

Tabella 4.2: Dati Ricevuti

Nome Attributo	**Descrizione**
CODICE_EVENTO	numero identificativo del corso assegnato dal Ministero
EDIZIONE	edizione del corso
CODICE_FISCALE	codice fiscale del professionista
RUOLO	ruolo del professionista nell'evento o progetto
COGNOME	cognome del professionista
NOME	nome del professionista
PROFESSIONE	professione del professionista quando si iscrive ad un evento o ad un progetto
DISCIPLINA	disciplina del corso
altro	altri campi non necessari all'inserimento

Il secondo file veniva spedito all'azienda fino all'anno scorso. Esso conteneva in una tabella Excel[4] tutte le informazioni sui singoli corsi. Da questo anno, quindi, è disponibile solamente il codice relativo al corso, mentre le altre informazioni come la data di inizio e di fine e il numero di crediti assegnati al corso sono mancanti.

I dati trattati durante lo stage fanno riferimento alle partecipazioni ai corsi durante il 2008.

4.4 Importazione

Ecco una visione dettagliata delle operazioni da portare a termine per l'importazione.

[4]Microsoft Excel©, formato .XLS

4.4.1 Azioni pre-operative

Codifica dei dati

Come precedentemente detto i dati vengono raccolti in molti modi diversi. Come primo passo è necessario lavorare sul modo in cui i dati vengono presentati. Le operazioni da eseguire sono le seguenti:

- Eliminare gli spazi iniziali e finali dalle stringhe;
- Convertire tutte le stringhe in caratteri maiuscoli;
- Verificare il tipo di codifica dei caratteri per poi, successivamente, riuscire ad aggirare eventuali problemi legati alla presenza di caratteri speciali;
- Eseguire un controllo sui tipi di dati che ci si dovrebbe aspettare da ogni attributo.

Campi non popolati e obbligatori

Prendendo in considerazione i campi obbligatori della tabella *PARTECIPAZIONE_ECM*, si verifica se essi siano tutti popolati. Nel caso non sia così si intraprendono le seguenti azioni:

- Campo RUOLO (TIPO_CREDITI_VALORI): non popolato oppure con valore diverso da P,D,T,R; valorizzato a 'P' (Partecipante);
- Campo PROFESSIONE: non popolato; valorizzato a 99 (codice per Tutte le professioni);

- Campo DISCIPLINA: non popolato; valorizzato a 0 (codice per Area interdisciplinare).

Campi non presenti

Nei file inviati dal Ministero della Salute non sono presenti le informazioni su alcuni campi. Ecco quali sono e quali azioni devo essere intraprese:

- Campo CORSO_ECM: valorizzato a [#anno_partecipazione] oppure [#anno_partecipazione]_PFA nel caso in cui si tratti di un Progetto Formativo;

- Campo ACCREDITATORE_ID: valorizzato a 1 (codice per Ministero);

- Campo TIPO_FORMAZIONE_VALORE: valorizzato a 'FOR' (codice per Formazione Residenziale);

- Campo TIPO_EVENTO_VALORE: valorizzato a 'E' se il record riguarda un evento, 'P' se riguarda un Progetto Formativo.

Codice Fiscale

Se durante il controllo sui codici fiscali risultano delle incongruenze, si può eseguire un controllo incrociato con i nomi e cognomi dei professionisti nel file di riferimento e con quelli della tabella *PROFESSIONISTA* di FNOMCEO. Se si trovano dei record con codici fiscali sufficiente-

mente simili o con uguali nomi e cognomi allora i codici fiscali vengono corretti.

Questa operazione può essere evitata se dal controllo risultano un numero troppo elevato di incongruenze.

Crediti

Data la non disponibilità del file contenente ulteriori informazioni sui singoli corsi compreso il numero dei crediti assegnati, la decisione presa è quella di assegnare ad ogni corso, qualora fossero presenti diversi valori in NUMERO_CREDITI per gli stessi valori di corso e edizione, il numero di crediti minore trovato.

4.4.2 inserimento dei dati nelle tabelle

Dati temporanei

Se il record da inserire ha il campo ERRORE_IMPORT valorizzato a 'TMP', esso verrà memorizzato nella tabella temporanea *PARTECIPAZIONE_ECM_TMP*. Quando saranno disponibili i nuovi aggiornamenti sui professionisti da parte della Federazione i record verranno modificati di conseguenza e si tenterà nuovamente di importarli nella tabella di destinazione.

Dati completi e corretti

Se il campo ERRORE_IMPORT non viene valorizzato (NULL) si può procedere con l'importazione nella tabella di destinazione *PARTECIPA-*

ZIONE_ECM.

Eliminazione dei dati

I record con il campo ERRORE_IMPORT valorizzato numericamente vengono segnalati al Ministero. Se entro un termine temporale stabilito non vengono re-inviati all'azienda corretti, si procede con la loro eliminazione dal DB.

Inoltre, una volta terminata l'importazione nel DB dei dati corretti, le loro copie nel DB COGEAPS_TMP vengono eliminate, dopo un ulteriore controllo in *PARTECIPAZIONE_ECM*.

4.5 Esempio di codice

L'applicazione di quanto detto fino ad ora, viene mostrata tramite il codice SQL usato per portare a termine l'importazione dei dati. Il codice sarà diviso in vari gruppi (pulizia, controlli e inserimento) e farà riferimento solo alle partecipazioni dei professionisti ai Progetti Formativi. La tabella di riferimento si chiama NEW_PFA ed è ovviamente presente nel DB COGEAPS_TMP. Le operazioni relative all'inserimento delle partecipazioni agli Eventi sono molto simili, e pertanto non verranno riportate.

```
-- DB selezionato: COGEAPS_TMP

-- controllo degli spazi e dei caratteri
UPDATE NEW_PFA
SET EVENTO = TRIM(EVENTO);
UPDATE NEW_PFA
SET EDIZIONE = TRIM(EDIZIONE);
UPDATE NEW_PFA
```

```
SET CODICE_FISCALE = TRIM(UPPER(CODICE_FISCALE));
UPDATE NEW_PFA
SET COGNOME = TRIM(UPPER(COGNOME));
UPDATE NEW_PFA
SET NOME = TRIM(UPPER(NOME));
UPDATE NEW_PFA
SET PROFESSIONE = TRIM(UPPER(PROFESSIONE));
UPDATE NEW_PFA
SET CORSO_ECM = TRIM(UPPER(CORSO_ECM));

-- crediti
UPDATE NEW_PFA
SET CREDITI = (SELECT MIN(CREDITI) FROM NEW_PFA
WHERE EVENTO = NEW_EVENTI_1.EVENTO)
WHERE EVENTO IN (
SELECT EVENTO FROM (
SELECT COUNT(*) AS CREDITI_DIVERSI, MIN(CREDITI) AS MINIMO, EVENTO FROM (
SELECT DISTINCT CREDITI, EVENTO FROM NEW_EVENTI_1
ORDER BY EVENTO ASC)
GROUP BY EVENTO)
WHERE CREDITI_DIVERSI > 1
);

-- check sul codice fiscale
CREATE VIEW VISTA PROVA AS
SELECT CODICE_FISCALE, NOME, COGNOME FROM (
SELECT LENGTH(CODICE_FISCALE) AS LUNGHEZZA,
CODICE_FISCALE, NOME, COGNOME FROM NEW_PFA)
WHERE LUNGHEZZA <> 16;
SELECT COD_FALSO, CODICE_FISCALE, NAME, NOME, FAMILYNAME, COGNOME
FROM FNOMCEO.PROFESSIONISTA INNER JOIN VISTA_PROVA
ON NAME = NOME AND COGNOME = FAMILYNAME;
-- esempio update codice fiscale
UPDATE NEW_PFA
SET CODICE_FISCALE = (
SELECT CODICE_FISCALE
FROM FNOMCEO.PROFESSIONISTA INNER JOIN VISTA_PROVA
ON NAME=NOME AND COGNOME= FAMILYNAME)
WHERE NOME = 'EMANUELA' AND COGNOME = 'RANALDI';

-- nomi e cognomi
CREATE VIEW V1 AS
SELECT FNOMCEO.PROFESSIONISTA.CODICE_FISCALE, NEW_PFA.NOME AS NN,
NEW_PFA.COGNOME AS NC, FNOMCEO.PROFESSIONISTA.NOME,
FNOMCEO.PROFESSIONISTA.COGNOME
FROM NEW_PFA INNER JOIN FNOMCEO.PROFESSIONISTA
ON NEW_PFA.CODICE_FISCALE = PROFESSIONISTA.CODICE_FISCALE
WHERE PROFESSIONISTA.COGNOME <> NEW_PFA.COGNOME
OR NEW_PFA.NOME <> PROFESSIONISTA.NOME;
UPDATE NEW_PFA
SET NOME = (
SELECT NOME FROM FNOMCEO.PROFESSIONISTA
WHERE NEW_PFA.CODICE_FISCALE = PROFESSIONISTA.CODICE_FISCALE),
COGNOME = (SELECT COGNOME FROM PROFESSIONISTA
```

```
WHERE NEW_PFA.CODICE_FISCALE = PROFESSIONISTA.CODICE_FISCALE)
WHERE NEW_PFA.CODICE_FISCALE IN (SELECT CODICE_FISCALE FROM V1);
SELECT COUNT(*) FROM NEW_PFA
WHERE (COGNOME, NOME) IN (SELECT NOME, COGNOME FROM NEW_PFA);
UPDATE NEW_PFA
SET (NOME, COGNOME) = (SELECT NOME, COGNOME FROM FNOMCEO.PROFESSIONISTA
WHERE PROFESSIONISTA.CODICE_FISCALE = NEW_PFA.CODICE_FISCALE)
WHERE NEW_PFA.CODICE_FISCALE IN (SELECT CODICE_FISCALE FROM NEW_PFA
WHERE (COGNOME, NOME) IN (SELECT NOME, COGNOME FROM NEW_PFA));

-- controllo codice fiscale in professionista
SELECT COUNT(*) FROM NEW_PFA
WHERE NEW_PFA.CODICE_FISCALE NOT IN (SELECT CODICE_FISCALE FROM
FNOMCEO.PROFESSIONISTA);

-- professione
UPDATE NEW_PFA
SET PROFESSIONE = (SELECT ID FROM PROFESSIONE
WHERE NEW_PFA.PROFESSIONE = UPPER(PROFESSIONE.DESCRIZIONE))
WHERE PROFESSIONE IN (SELECT UPPER(DESCRIZIONE) FROM PROFESSIONE);
UPDATE NEW_PFA
SET PROFESSIONE = (
SELECT ID FROM PROFESSIONE WHERE
DESCRIZIONE = 'Terapista della neuro e psicomotricità dell''età evolutiva')
WHERE PROFESSIONE LIKE 'TERAPISTA DELLA NEURO E PSICOMOTRICIT%';
UPDATE NEW_PFA
SET PROFESSIONE = (
SELECT ID FROM PROFESSIONE WHERE DESCRIZIONE =
'Tecnico educazione e riabilitazione psichiatrica e psicosociale')
WHERE PROFESSIONE LIKE 'TECNICO DELLA RIABILITAZIONE PSICHIATRICA';

-- campi non presenti
ALTER TABLE COGEAPS_STAGE.NEW_PFA ADD ACCREDITATORE_ID decimal(22);
UPDATE NEW_PFA
SET ACCREDITATORE_ID = 1;
ALTER TABLE COGEAPS_STAGE.NEW_PFA ADD TIPO_FORMAZIONE_VALORE char(3) DEFAULT 'FOR';
UPDATE NEW_PFA
SET TIPO_FORMAZIONE_VALORE = 'FOR';
ALTER TABLE COGEAPS_STAGE.NEW_PFA ADD TIPO_EVENTO_VALORE char(1);
UPDATE NEW_PFA
SET TIPO_EVENTO_VALORE = 'P';
ALTER TABLE COGEAPS_STAGE.NEW_PFA ADD DATA_INIZIO date;
UPDATE NEW_PFA
SET DATA_INIZIO = TO_DATE(CONCAT(SUBSTR(CORSO_ECM, 1, 4), '-01-01'), 'YYYY-MM-DD');
ALTER TABLE COGEAPS_STAGE.NEW_PFA ADD DATA_FINE date;
UPDATE NEW_PFA
SET DATA_FINE = TO_DATE(CONCAT(SUBSTR(CORSO_ECM, 1, 4), '-12-31'), 'YYYY-MM-DD');

--campi non popolati
ALTER TABLE COGEAPS_STAGE.NEW_PFA ADD RUOLO char(1);
UPDATE NEW_PFA
SET RUOLO = 'P';
ALTER TABLE COGEAPS_STAGE.NEW_PFA ADD PROFESSIONE decimal(22);
UPDATE NEW_PFA
```

```
SET PROFESSIONE = 99
WHERE PROFESSIONE IS NULL;
ALTER TABLE COGEAPS_STAGE.NEW_PFA ADD DISCIPLINA decimal(22);
UPDATE NEW_PFA
SET DISCIPLINA = 0;

-- errore
UPDATE NEW_PFA
SET ERRORE_IMPORT = 'TMP'
WHERE NEW_PFA.CODICE_FISCALE NOT IN
(SELECT CODICE_FISCALE FROM FNOMCEO.PROFESSIONISTA);
UPDATE NEW_PFA
SET ERRORE_IMPORT = 2
WHERE CODICE_FISCALE IN (
SELECT CODICE_FISCALE FROM (
SELECT LENGTH(CODICE_FISCALE) AS LUNGHEZZA, CODICE_FISCALE FROM NEW_PFA)
WHERE LUNGHEZZA <> 16
);
SELECT * FROM (
SELECT COUNT(*) AS CONTA, EVENTO, EDIZIONE CODICE_FISCALE,
RUOLO, PROFESSIONE, DISCIPLINA FROM NEW_PFA
GROUP BY EVENTO, EDIZIONE, CODICE_FISCALE, RUOLO, PROFESSIONE, DISCIPLINA
ORDER BY CONTA DESC
)
WHERE CONTA > 1;

-- DB selezionato: FNOMCEO

-- importazione dati
INSERT INTO FNOMCEO.PARTECIPAZIONE_ECM
(ID, CORSO_ECM, NUMERO_CREDITI, DATA_INIZIO, DATA_FINE, CODICE,
CODICE_EDIZIONE, CODICE_ORGANIZZATORE, PROFESSIONISTA_CODICE_FISCALE,
DISCIPLINA_ECM_ID, PROFESSIONE_ID, ACCREDITATORE_ID, TIPO_EVENTO_VALORE,
TIPO_CREDITI_VALORE, TIPO_FORMAZIONE_VALORE, FORZATA)
SELECT PARTECIPAZIONE_ECM_SEQ.NEXTVAL, CORSO_ECM, CREDITI, DATA_INIZIO,
DATA_FINE, EVENTO, EDIZIONE, 999, CODICE_FISCALE, DISCIPLINA,
PROFESSIONE, ACCREDITATORE_ID, TIPO_EVENTO_VALORE, RUOLO,
TIPO_FORMAZIONE_VALORE, NULL
FROM COGEAPS_TMP.NEW_PFA
WHERE (ERRORE_IMPORT IS NULL);
INSERT INTO FNOMCEO.PARTECIPAZIONE_ECM_TMP
(ID, CORSO_ECM, NUMERO_CREDITI, DATA_INIZIO, DATA_FINE, CODICE,
CODICE_EDIZIONE, CODICE_ORGANIZZATORE, PROFESSIONISTA_CODICE_FISCALE,
DISCIPLINA_ECM_ID, PROFESSIONE_ID, ACCREDITATORE_ID, TIPO_EVENTO_VALORE,
TIPO_CREDITI_VALORE, TIPO_FORMAZIONE_VALORE, COGNOME, NOME)
SELECT PARTECIPAZIONE_ECM_TMP_SEQ.NEXTVAL, CORSO_ECM, CREDITI, DATA_INIZIO,
DATA_FINE, EVENTO, EDIZIONE, 999, CODICE_FISCALE, DISCIPLINA,
PROFESSIONE, ACCREDITATORE_ID, TIPO_EVENTO_VALORE, RUOLO,
TIPO_FORMAZIONE_VALORE, COGNOME, NOME
FROM COGEAPS_TMP.NEW_PFA
WHERE (ERRORE_IMPORT = 'TMP');
```

Capitolo 5

Alcune considerazioni statistiche e non

Il lavoro di gestione in una base di dati non si limita a quanto visto nei precedenti capitoli. Mentre si è discusso in modo ampio di costruzione di rapporti di sintesi e di pulizia ed inserimento di dati, non sono stati approfonditi, infatti, argomenti come la modifica e l'eliminazione di dati[1] o di tabelle dal database o la gestione dei dati.

Inoltre potrebbe essere utile fare alcune considerazioni sulla funzionalità dei report descritti precedentemente e sulla loro comprensibilità.

5.1 Modifiche all'interno del DB

Le modifiche all'interno del database possono riguardare i dati o le tabelle. Per quanto riguarda i dati, le modifiche avvengono assai raramente.

Come visto nel Capitolo 2, all'interno del DB FNOMCEO è stata implementata una tabella dove allocare temporaneamente le tuple della relazione PROFESSIONISTA, in attesa che queste vengano eliminate.

[1] altre informazioni dettagliate su [b]

Si potrebbe pensare ad un modo identico anche per la semplice modifica dei dati, ma in realtà non è così. Visto che è utile a fini di completezza dell'informazione mantenere tutte gli aggiornamenti sui dati, la modifica di una tupla di dati è, in realtà, l'inserimento di un nuovo record nella tabella e lo spostamento del vecchio record nella rispettiva tabella storica. Ad esempio, nella tabella *INDIRIZZO* nel caso in cui un professionista cambi cittadinanza viene tenuta traccia della vecchia appartenenza nella tabella storica, e inserito in *INDIRIZZO* il nuovo record aggiornato che fa sempre riferimento a quel professionista.

Le modifiche della struttura del database, invece, avvengono ancora più raramente. Può accadere su richiesta, nel caso in cui sono stati effettuati dei cambiamenti di tipo concettuale in un determinato campo, quale può essere l'ordine, le generalità di un provvedimento, ecc...

Comunque sia, prima di effettuare qualsiasi modifica sul database viene creato un backup di sicurezza per mantenere intatta l'integrità dei dati ed evitare il collasso del DB.

5.2 Integrazione sui report

I report chiesti dalla Federazione riguardano specialmente la partecipazione dei professionisti agli eventi e ai progetti del progetto di Educazione Continua e del numero di crediti acquistati da coloro che vi hanno partecipato (si veda gli esempi di report al Capitolo 2).

Il risultato finale dei report sono delle tabelle in cui vengono presentati i dati senza però dare, molto spesso, un indicazione globale di quello

per cui sono stati costruiti. Verrà proposto qui di seguito un esempio di come si potrebbe realizzare un riepilogo diviso per professione (vedi figura 3.1).

Dalle tabelle e dai grafici per la professione di assistente sanitario si possono fare alcune considerazioni:

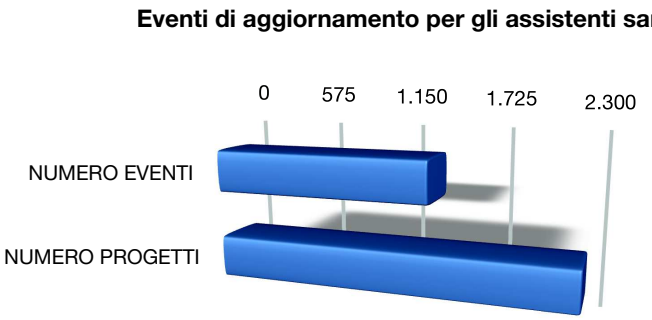
- il grafico a torta presenta quanto scritto nella tabella sovrastante, ovvero quante sono state le partecipazioni ai corsi di aggiornamento per i professionisti e quanti di questi hanno partecipato a degli eventi o a dei progetti;

- l'istogramma indica a quanti eventi o progetti hanno partecipato i professionisti; si noti che c'è molta differenza con le partecipazioni, in quanto le partecipazioni possono fare riferimento allo stesso evento per più professionisti fino a raggiungere il totale di eventi a cui i professionisti hanno presenziato;

- l'ultima tabella presenta invece alcune banali statistiche di riepilogo che possono fornire un'idea sulla necessità di aumentare (o diminuire) in futuro il numero degli eventi o dei progetti a seconda di quanti professionisti mediamente vi partecipano, oppure se per gli eventi è necessario aumentare (o diminuire) il numero dei crediti che i professionisti acquisiscono mediamente nel caso in cui, ad esempio, le regolamentazioni dell'ECM dovessero cambiare.

PROFESSIONE	PARTECIPAZIONI EVENTI	PARTECIPAZIONI PROGETTI	TOTALE
Assistente sanitario	6.444	7.192	13.636

PROFESSIONE	NUMERO EVENTI	NUMERO PROGETTI	TOTALE CREDITI
Assistente sanitario	1.353	2.213	141.374



STATISTICHE DI RIEPILOGO	
NUMERO DI PROFESSIONISTI PER EVENTO	4,36
NUMERO DI PROFESSIONISTI PER PROGETTO	3,25
NUMERO DI CREDITI PER PROFESSIONISTA	10,37

Figura 5.1: Riepilogo sulla professione di assistente sanitario

5.3 Oltre ai report: il data mining

Il riepilogo presentato è solo un abbozzo di quello che si potrebbe ricavare dai dati dei report. Per un'analisi più approfondita si possono utilizzare strumenti statistici utili a ricavare delle informazioni utili non solo dai dati presenti nei rapporti, ma anche dalla stessa base di dati. Questo modo di operare rientra nell'ambito del data mining (letteralmente estrazione di dati).

Per dare una definizione più rigorosa si può dire che il data mining rappresenta l'attività di elaborazione in forma grafica o numerica di grandi raccolte di dati o di flussi continui di dati (per l'appunto le miniere di dati) con lo scopo di estrarre informazioni utili a chi detiene i dati stessi.

In certi casi non si sa nemmeno cosa sia l'informazione utile, poiché molto spesso non è definito a priori quale sia l'oggetto di interesse, ma lo si scopre analizzando i dati. In generale, comunque, è possibile distinguere due situazioni chiave in cui il data mining risulta utile, partendo dai dati disponibili:

1. la costruzione di un modello globale che spieghi nel complesso il fenomeno in esame;

2. l'individuazione di particolarità nell'andamento dei dati, ovvero identificare le situazioni al di fuori del comportamento standard nel fenomeno.

Da queste due situazioni possiamo ottenere delle soluzioni da utilizzare nei tempi e nei modi giusti, a seconda di qual è il problema da affrontare, come la previsione di una variabile quantitativa nel tempo o la classificazione o il raggruppamento dei dati rispetto ad una certa caratteristica.

Per effettuare queste analisi statistiche è necessario l'intervento di un tecnico specializzato in operazioni di data mining, quale può essere uno statistico con nozioni di informatica che abbia la possibilità di mettersi in contatto con la Federazione per ricevere le istruzioni e gli obiettivi da perseguire[2].

[2]ulteriori informazioni sul data mining su [d]

Capitolo 6

Trasferimento dati con XML: import ed export

Come è stato ripetuto più volte, FNOMCEO è un DB complesso sotto molti punti di vista, dall'ideazione alla gestione. Inoltre, esso è un DB inserito in un contesto di molteplici interazioni con il mondo esterno al database stesso, dovuto anche alla definizione e all'analisi dei requisiti per i quali è stato creato.

L'inserimento dei dati nel DB è solo una piccola parte delle interazioni che vengono effettuate: c'è la necessità di mettere i dati a disposizione degli utenti del Web Server, fornire le elaborazioni richieste dai vari Enti e Federazioni e avere la possibilità di trasferire dei backup dei dati su altre macchine nel caso di operazioni di manutenzione del server; tutte operazioni già viste nei capitoli precedenti.

Soffermandosi sulle singole operazioni, però, si nota come ognuna di esse sia in rapporto con un formato di input o output diverso dai formati delle altre operazioni. Per citare un esempio si pensi ai formati usati

per la creazione dei Flat-DB citati nel capitolo 3 o ai formati dei documenti che arrivano dalla Federazione per l'inserimento in FNOMCEO del capitolo 4: nel primo caso i dati vengono esportati in formato .CSV (Comma Separated Values), mentre nel secondo vengono elaborati dati .MDB (Microsoft Database Access®) e fino all'anno scorso anche dati .XLS (Microsoft Excel®).

Lo svantaggio principale nell'uso di tutti questi formati è la necessità di usare diversi programmi in grado di leggerli, soprattutto nel caso dei software proprietari. Questo comporta la non comunicabilità tra i dati da esportare o da importare e, a volte, la creazione ripetuta di gruppi di dati per tutti i tipi di formati necessari.

Una soluzione a questi problemi può essere l'uso di un formato standard che possa essere letto da tutte le applicazioni e che possa essere applicato al contesto dei DB. I documenti XML vengono incontro a questa esigenza.

6.1 Il linguaggio XML

Ma qual è il punto di collegamento tra il linguaggio XML e i database? Prima di rispondere a questa domanda è necessario presentare brevemente XML[1].

Il linguaggio XML (eXtensible Markup Language) è un metalinguaggio di markup, ovvero un linguaggio marcatore che definisce altri linguaggi marcatori. Per essere precisi più che un linguaggio si può trattare

[1] Il discorso su XML è stato impostato su approcci e definizioni di [f], [1] e, in minor parte [[e]]

come una notazione specifica. XML può infatti definire il markup specifico per un qualsiasi oggetto o informazione che può essere strutturata. La sua definizione non dice nulla riguardo la semantica dei tag, ma si occupa soltanto di definire con quale sintassi dev'essere organizzato il linguaggio.

XML è l'evoluzione del primo metalinguaggio di marcatura creato nei laboratori IBM a partire dagli anni '70: SGML. Il linguaggio aveva lo scopo di definire la struttura di un documento, tramite una marcatura di tipo descrittivo, ma data la complessità del vocabolario SGML si è optato per una semplificazione e una modifica delle sue caratteristiche. Nel 1996 è iniziata la lavorazione di quello che ora è il linguaggio XML, mentre nel 1998 è stata raggiunta la raccomandazione dal W3C (http://www.w3c.org).

Teoricamente, si potrebbe pensare che HTML sia soltanto una versione del linguaggio XML, in quanto anch'esso è un linguaggio di marcatura, ma per alcune differenze di sintassi questo non si adatta perfettamente a XML. Allo stesso modo un browser Web non è in grado di leggere un documento XML, poichè è programmato per leggere determinati tag nei marcatori, mentre i tag in XML possono essere i più svariati. Per risolvere questi problemi sono stati definiti il linguaggio XHTML e i fogli di stile XSLT.

Le applicazioni di XML si possono classificare in quattro grandi gruppi:

- Linguaggi orientati ai dati;

- Linguaggi orientati ai documenti;
- Protocolli e linguaggi di programmazione;
- Linguaggi ibridi.

Tra questi gruppi, il più interessante è senza dubbio il primo: i linguaggi XML orientati ai dati vengono usati per descrivere informazioni che sarebbero altrimenti memorizzate in una base di dati.

Anche il gruppo dei linguaggi ibridi, però, si presta a considerazioni sui database: sono linguaggi che presentano caratteristiche sia dei linguaggi orientati ai dati che di quelli orientati ai documenti, ovvero quei linguaggi rivolti alla strutturazione logica di un documento o un file di testo scritto in linguaggio naturale. L'insieme dei due campi provocherà un tipo di linguaggio complesso per basi di dati non convenzionali.

Requisito indispensabile di ogni documento XML è il suo essere *ben formato*, ovvero che la sua forma testuale rispetti l'apertura e la chiusura dei tag e il loro annidamento. Inoltre, un documento XML può essere anche *valido* nel caso ci sia un altro documento che definisce uno schema sintattico a cui il documento XML deve far riferimento e che viene rispettato. La definizione formale dello schema a cui deve sottostare il documento XML rappresenta una fonte di riferimento precisa per il documento stesso, ma anche leggibile e comprensibile dagli esseri umani. Si cerca di *standardizzare* (usando un espressione leggermente esagerata) il documento, in modo tale che altri documenti dello stesso tipo e con lo stesso linguaggio XML personalizzato possano essere scritti e

letti da altri utenti. Quindi un linguaggio schema, per essere utile deve soddisfare tre requisiti:

- dev'essere *espressivo*, cioè dev'essere in grado di poter formalizzare la maggior parte dei requisiti sintattici;
- deve rendere possibile l'implementazione di processori di schemi *efficienti*, cioè i requisiti spaziali e temporali per il controllo della validità di un documento dovrebbero crescere, idealmente, in modo lineare con le sue dimensioni.
- infine, dev'essere *comprensibile*, cioè, come si diceva poco sopra, anche gli autori esterni devono essere in grado di capire il linguaggio per poter scrivere ulteriori documenti validi.

Tra i linguaggi che fungono a questo scopo, si possono citare DTD (Document Type Definition) e XML Schema.

6.2 Alberi XML e tabelle relazionali

Un documento XML è costituito da una struttura gerarchica che viene generalmente chiamata albero XML, poichè consiste in un diverso numero di nodi di vari tipi organizzati ad albero. I nodi possono rappresentare un tag, degli attributi di un tag o un'informazione. Dall'esempio si può capire come un documento XML possa essere trasformato in un albero:

```
<collezione>
  <descrizione>Film Videoteca Padova</descrizione>
```

```
  <film id="057">
    <titolo>Eyes Wide Shut</titolo>
    <regia>Stanley Kubrick</regia>
    <genere>Drammatico</genere>
    <data>1999</data>

    <attore ruolo="Dr. Bill Harford">Tom Cruise</attore>
    <attore ruolo="Alice Harford">Nicole Kidman</attore>
    ...

    <location>New York</location>
    <crew>
      <sceneggiatore>Stanley Kubrick</sceneggiatore>
      ...
    </crew>

    <supporto>VHS</supporto>
   ...

  </film>
</collezione>
```

Considerando i vari tipi di dati che formano il documento XML possiamo ragionevolmente trasformare il documento nel seguente albero:

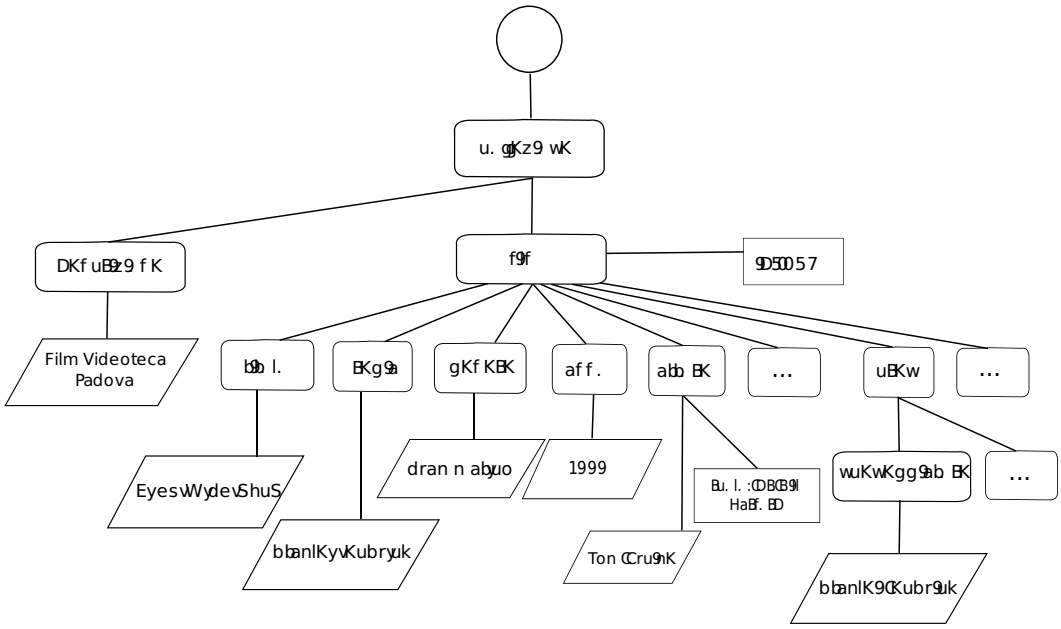


Figura 6.1: Albero XML

I vari nodi possono essere divisi sotto queste categorie:

- Nodo di testo: corrisponde a un frammento di informazione rappresentata nel documento XML ed è sempre una foglia, ovvero un nodo terminale dell'albero. Viene segnalato tramite un parallelogramma. Un esempio può essere "Tom Cruise";

- Nodo elemento: definisce un raggruppamento logico dell'informazione rappresentata dai discendenti (ovvero dai nodi che stanno sotto e che sono collegati ad esso). Viene segnalato tramite un rettangolo dagli angoli smussati ed un esempio può essere "film";

- Nodo attributo: viene sempre associato ad un nodo elemento, in modo tale da dare ulteriori informazioni all'elemento stesso. Viene segnalato da un rettangolo ed un esempio può essere "ruolo: Dr. Bill Harford";

- Nodo commento: è un tipo particolare di foglia, il cui scopo è analogo a quello dei commenti nei linguaggi di programmazione, ovvero esprimere meta-informazioni in modo informale. In questo caso non ci sono nodi commento;

- Nodo di istruzione per l'elaborazione: è composto da un obiettivo (o target) e un valore, e può essere utilizzato per trasmettere meta-informazioni specializzate agli strumenti che operano sul codice XML;

- Nodo radice: si tratta del nodo che sta a capo di tutto il documento, può avere come figlio un solo nodo elemento, chiamato elemento radice. Viene rappresentato dal cerchio.

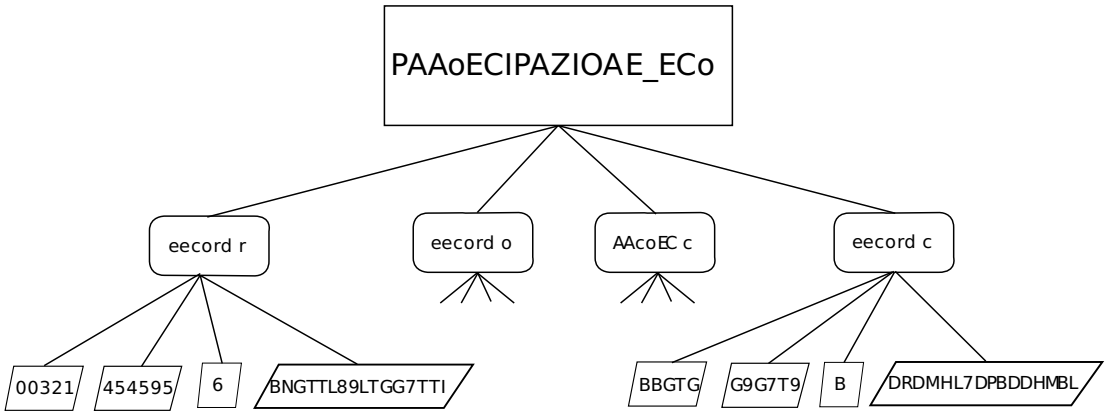
Inoltre va fatta una precisazione sull'ordine in cui sono stati inseriti gli oggetti nell'albero: proseguendo da sinistra verso destra rispettiamo l'ordinamento del documento originale.

L'esempio appena riportato mostra in generale com'è organizzato un albero XML. In realtà, nel caso delle tabelle relazionali di un database le cose si fanno ancora più semplici: prendendo in esame la tabella PARTECIPAZIONE_ECM vista nel capitolo 4 e inserendo alcuni dati (semplificando la tabella per comodità) vediamo come il confine con gli alberi XML sia molto sottile:

Tabella 6.1: *PARTECIPAZIONE_ECM* (Semplificata)

ID	**CORSO_ECM**	**NUMERO_CREDITI**	**PROFESSIONISTA_CODICE_FISCALE**
000321	324592	6	BNGTTL81L23G752I
000322	304582	10	VNCCRB72T58D568H
000323	405234	6	PLMTTI63D10F637K
000324	394751	6	GRDMHL73P04G920L

La versione semplificata di *PARTECIPAZIONE_ECM* a quattro campi comprende quattro record inventati ad hoc. Le tabelle relazionali sono spesso rappresentate nella forma tabulare, ma è altrettanto facile disegnarle in forma di albero:


PAAoECIPAZIOAE_ECo
eecord r
eecord o
AAcoEC c
eecord c
00321
454595
6
BNGTTL89LTGG7TTI
BBGTG
G9G7T9
B
DRDMHL7DPBDDHMBL

in un documento XML è possibile raggruppare informazioni relative ad una entità centrale in un unico documento.

Infine, tramite la definizione del tipo di documento con XML (con XML Schema ad esempio) si può organizzare il documento in modo tale che rispetti le indicazioni fornite.

6.3 XML Database

Da quanto visto è quindi possibile, ipoteticamente, avere a che fare con una base di dati che possa elaborare documenti XML. Il problema principale sta nella definizione di un linguaggio di interrogazione simile a SQL per inserire o estrapolare dati dal DB.

Tornando alle applicazioni dei documenti XML classificate prima si possono immaginare diversi scenari d'uso per un linguaggio di interrogazione applicabile su questi tipi di documenti.

Per i linguaggi orientati ai dati lo scopo è quello di usare query che potrebbero essere usate anche in un modello relazionale. Quindi, è necessario trasformare dati in nuove rappresentazioni XML e integrare informazioni provenienti da più fonti di diversi tipi.

Per i linguaggi ibridi, invece, si ripresenta la dualità tra i linguaggi orientati ai dati appena visti e i linguaggi orientati ai documenti. Le considerazioni su quest'ultimo tipo di linguaggi possono prevedere la realizzazione di query che estrapolino alcune parti del testo, che eseguano ricerche contestuali nel documento o, in generale, che comprendano tutte le attività studiate nel campo del recupero delle informazioni (*in-*

formation retrieval). L'impostazione di un linguaggio di interrogazione nell'ambito dei linguaggi ibridi prevede quindi un miscuglio degli scopi e delle operazioni viste sugli altri due tipi di linguaggio, esaminando documenti che abbiano in parte informazioni organizzate in dati e in parte informazioni racchiuse in documenti non assimilabili in forma tabulare, ma di facile comprensione alla lettura di un essere umano.

Tutto questo porta quindi all'ideazione di un tipo di DB che sia collegato all'uso di documenti XML e con princìpi simili alle basi dei modelli relazionali: questo tipo di DB è l'XML database.

La definizione generica di XML database (o XML-DB) è la seguente: sistema software di dati persistente che permette la memorizzazione di dati in formato XML. Questi dati potranno essere successivamente integrati, esportati o serializzati nel formato desiderato. Persistente indica che la struttura dati mantiene sempre la versione precedente di se stessa quando questa viene modificata.

Gli XML-DB possono essere divisi in due classi:

- Native XML: la modellazione interna della struttura dei database è basata su XML e i documenti XML sono la principale unità di riferimento dei dati. In questi tipi di DB è necessario usare un linguaggio di interrogazione ad hoc come XQuery [2];
- XML-Enabled: la base di partenza è un normale DB relazionale in cui viene integrata la possibilità di leggere in input documenti XML e di scriverli in output. Questo implica che la conversione

[2] http://www.xquery.com

dei dati nei documenti XML e viceversa venga fatta all'interno del database stesso.

La seconda classe è la strada preferibile per permettere a FNOMCEO di elaborare documenti XML. Infatti, già una tabella del DB prevedeva l'esportazione dei dati in XML, ma solo per alcuni casi specifici come la manutenzione del server che ospita il DB. Inoltre il processo di trasformazione di dati in documenti XML viene sempre fatta a parte e non all'interno di FNOMCEO.

6.4 Il linguaggio SQL/XML

Il metodo più intuitivo per il supporto di XML nella base di dati già esistente consiste nella pubblicazione di viste XML dei dati relazionali. SI vuole quindi trasformare una tabella contenente i dati di interesse in un documento XML che rispecchia esattamente le caratteristiche della tabella stessa. Questa operazione si chiama *mappatura* e per realizzarla si usa il linguaggio SQL/XML, un'estensione del linguaggio XML.

La mappatura permette di avere un documento XML basato su una strategia scelta dall'utente. Nell'esempio di *PARTECIPAZIONE_ECM* è possibile ottenere questo risultato:

```
<PARTECIPAZIONE_ECM>
  <record id="000321" corso_ecm="324592" numero_crediti="6"
          professionista_codice_fiscale="BNGTTL81L23G752I" />
  <record id="000322" corso_ecm="304582" numero_crediti="10"
          professionista_codice_fiscale="VNCCRB72T58D568H" />
  <record id="000323" corso_ecm="405234" numero_crediti="6"
          professionista_codice_fiscale="PLMTTI63D10F637K" />
  <record id="000324" corso_ecm="394751" numero_crediti="6"
          professionista_codice_fiscale="GRDMHL73P04G920L" />
</PARTECIPAZIONE_ECM>
```

o quest'altro:

```
<PARTECIPAZIONE_ECM>
  <record>
    <id>000321</id>
    <corso_ecm>324592</corso_ecm>
    <numero_crediti>6</numero_crediti>
    <professionista_codice_fiscale>
        BNGTTL81L23G752I</professionista_codice_fiscale>
  </record>
  <record>
    <id>000322</id>
    <corso_ecm>304582</corso_ecm>
    <numero_crediti>10</numero_crediti>
    <professionista_codice_fiscale>
        VNCCRB72T58D568H</professionista_codice_fiscale>
  </record>
  <record>
    <id>000323</id>
    <corso_ecm>405234</corso_ecm>
    <numero_crediti>6</numero_crediti>
    <professionista_codice_fiscale>
        PLMTTI63D10F637K </professionista_codice_fiscale>
  </record>
  <record>
    <id>000324</id>
    <corso_ecm>394751</corso_ecm>
    <numero_crediti>6</numero_crediti>
    <professionista_codice_fiscale>
        GRDMHL73P04G920L </professionista_codice_fiscale>
  </record>
</PARTECIPAZIONE_ECM>
```

o, in alternativa, è possibile usare altre strategie.

Come si evince dagli esempi, è possibile delineare delle caratteristiche comuni (e valide per tutte le strategie adoperate) dai documenti appena ricavati: innanzitutto il nome della tabella diventa il nome del documento, cioè quello che possiamo definire elemento radice, mentre ogni record viene identificato con un elemento a sè stante, come mostrato nell'albero della tabella relazionale.

Oltre a queste caratteristiche esistono vari operatori che permettono di definire il documento XML secondo le proprie esigenze. Nel primo esempio è stato utilizzato l'operatore *xmlattributes*, mentre nel secondo

l'operatore *xmlforest*; in entrambi è stato usato l'operatore *xmlelement* per definire il documento e i record[3]. Ecco come si presenta la prima query:

```
xmlelement(name, "PARTECIPAZIONE_ECM",
  SELECT xmlelement(name, "record", xmlattributes(id, corso_ecm,
              numero_crediti, professionista_codice_fiscale))
  FROM PARTECIPAZIONE_ECM
)
```

ed ecco la seconda:

```
xmlelement(name, "PARTECIPAZIONE_ECM",
  SELECT xmlelement(name, "record", xmlforest(id, corso_ecm,
              numero_crediti, professionista_codice_fiscale))
  FROM PARTECIPAZIONE_ECM
)
```

Ottenendo molte viste in XML in cui sono racchiusi, a seconda dei dati che vengono trattati, elementi strutturati ed elementi non-strutturati (quali sono i documenti) è ora possibile sfruttare appieno le potenzialità di un linguaggio di interrogazione che opera sui documenti XML, come XQuery. Questo linguaggio è stato progettato per generalizzare in qualche modo quelle che sono le operazioni di SQL, nello stesso modo in cui XML generalizza la tabella di un database.

Il procedimento inverso, invece, ovvero passare da un documento XML a una tabella relazionale è decisamente più complicato, in quanto non esiste una mappatura standard come per la trasformazione da tabella relazionale a XML. La motivazione principale sta nel fatto che il linguaggio XML è molto più espressivo del modello relazionale e quindi è difficile creare una struttura rigida che contenga le informazioni di quei tipi di documento.

[3]Informazioni più specifiche su SQL/XML su [10]

Per cercare di ovviare al problema, le aziende che commerciano DBMS hanno considerato l'idea di utilizzare delle tecniche ad hoc.

Oracle è una delle aziende che ha pensato alla soluzione di questo problema di mappatura. E' stata presa in esame poiché il DB FNOMCEO è un DB Oracle, e quindi è utile alla causa. Essa usa XSU (XML/-SQL Utility), un'interfaccia per Java e PL/SQL, ovvero il linguaggio di interrogazione proprietario di Oracle.

Tramite XSU, documenti XML con una struttura regolare possono essere tranquillamente inseriti in una o più tabelle. Inoltre, la tecnologia *object/relational* di Oracle; in cui una tabella può contenere altre tabelle innestate permette la mappatura diretta delle nidificazioni presenti in documenti XML[4].

[4] Approfondimenti su XSU in [g]

Bibliografia

[a] Paolo Atzeni, Stefano Ceri, Stefano Paraboschi e Stefano Torlone (2002), Basi di dati: Modelli e linguaggi di interrogazione, McGraw-Hill, Milano.

[b] Ralph Kimball e Joe Caserta (2004), The Data Warehouse ETL Toolkit, Wiley, Indianapolis.

[c] Elmasri Ramez A. e Navathe Shamkant B (2007), Sistemi di basi di dati. Fondamenti, Pearson Education Italia, Milano.

[d] Bruno Scarpa e Adelchi Azzalini (2004), Analisi dei dati e data mining, Springer Verlag, Milano.

[e] Ramakrishnan Raghu e Gehrke Johannes (2004), Sistemi di basi di dati, McGraw-Hill, Milano.

[f] Moller Anders e Schwartzbach Michael (2007), Introduzione a XML, Pearson Education-Addison Wesley, Milano.

[g] Chaudhri Akmal B., Rashid Awais, Zicari Roberto (2003) XML Data Managment, Pearson Education, Boston.

Siti consultati

[1] http://www.wikipedia.org

[2] http://www.britannica.com

[3] http://www.iso.org

[4] http://www.oracle.com

[5] http://www.xenialab.it/meo/web/white/oracle/

[6] http://www.ministerodellasalute.it/ecm/ecm.jsp

[7] http://www.fnomceo.it

[8] http://www.cogeaps.it

[9] http://www.omitech.it

[10] http://www1.cs.unicam.it/insegnamenti/bdmulti/default.aspx