UNIVERSITÀ DEGLI STUDI DI PADOVA

Facoltà di Scienze Statistiche

Corso di laurea in
Statistica e Tecnologie Informatiche

Tesi di laurea

ESTRAZIONE DATI E PREPARAZIONE DATABASE DELLA FNOMCEO PER REPORT

EXTRACTING DATA AND PREPARING FNOMCEO'S DATABASE FOR REPORTING

Relatore: Chiar.mo Prof Graziano Deambrosis

Laureando: Giancarlo Simion
Matricola N. 555590 – STI

Anno accademico 2008-2009

Indice

Introduzione

L'esperienza di stage condotta presso la ditta "Omicron Technologies S.R.L." di Padova, azienda che opera nel settore informatico occupandosi principalmente della gestione di basi di dati e dello sviluppo di applicazioni Web, mi ha dato la possibilità di venire direttamente a contatto con l'uso di database informatici e di apprendere alcune tecniche e conoscenze necessarie per lavorare su di essi.
Ho partecipato ad un progetto su un DB (database) denominato *FNOMCEO* contenente dati relativi ai professionisti della FNOMCeO[1] e alla loro attività. Tale progetto comprendeva la realizzazione di particolari reportistiche necessarie alla Federazione a fini di analisi descrittive. Per portare a termine il progetto mi è stato chiesto di lavorare direttamente su *FNOMCEO* per produrre le query.
Successivamente ho avuto anche modo di prendere parte al processo di importazione nel DB dei dati relativi alle partecipazioni dei professionisti a determinati corsi di aggiornamento nell'anno 2008.

Questa relazione descrive il lavoro svolto durante lo stage, includen-

[1] FNOMCeO: La Federazione Nazionale Ordini Medici Chirurghi e Odontoiatri è l'organo che raccoglie tutti gli Ordini dei Medici Chirurghi e Odontoiatri delle provincie italiane e ha sede a Roma. In Italia, contrariamente a quanto succede in altri paesi, l'Ordine dei MCeO è a regime provinciale, retaggio storico della frammentazione territoriale del nostro paese.

do alcuni aspetti teorici necessari a facilitarne la comprensione. Di fondamentale importanza risulta infatti avere le idee chiare sia per quanto riguarda l'universo dei database, in particolare dei DBMS (Database Management System), che per alcune tecniche informatiche necessarie operativamente a svolgere il lavoro oggetto della presente analisi.
Ulteriori approfondimenti sulle tematiche trattate sono inoltre consultabili nelle Appendici.

Data la complessità del DB in questione, prima di descrivere la parte operativa (comprendente anche alcuni frammenti di codice utilizzato), sarà dedicato uno spazio ad un'analisi esplorativa di *FNOMCEO* per comprenderne il funzionamento e soprattutto la struttura, con le sue tabelle.

Premetto infine che, per motivi di privacy, molte delle informazioni riguardanti i professionisti presenti nel database aziendale non potranno essere riportate all'interno della relazione.

Capitolo 1

Analisi esplorativa di FNOMCEO

1.1 Nascita del progetto

La FNOMCeO è nata grazie all'esigenza di centralizzare gli ordini dei Medici Chirurgi ed Odontoiatri; in Italia infatti, per motivi storici, essi vengono gestiti autonomamente da ciascuna provincia e, prima della nascita della Federazione, non c'era alcuna struttura centrale di riferimento per i singoli professionisti.

La quantità di informazioni che gli Ordini devono raccogliere a livello di provincia e archiviare è enorme. Qualsiasi utente interessato a visualizzare i dati relativi al proprio medico deve poter consultare tali informazioni, inoltre i dati memorizzati servono all'Ordine per eventuali rielaborazioni statistiche o, semplicemente, per attività di monitoraggio dell'operato dei professionisti iscritti.

Per questo motivo è nata l'esigenza di creare, a livello federale, un DB che potesse in qualche modo facilitare questi compiti e soprattutto che fosse in grado di raccogliere tutti i dati in un'unica struttura. Per faci-

litare le operazioni sui dati, parallelamente sistema di memorizzazione, andava creata anche un'applicazione Web collegata direttamente al DB.

Il progetto della costruzione del DB per la gestione nazionale dei professionisti degli ordini provinciali nasce nel 2004 su commissione da parte della FNOMCeO presso la Omicron Technologies di Padova. Da allora è stato sviluppato dall'azienda l'intero DB, utilizzando una tecnologia DBMS Oracle, e l'applicazione Web disponibile sul sito della Federazione (`http://www.fnomceo.it`). Lo scopo principale di questo DB, oltre a quello di raccogliere e memorizzare sistematicamente tutti i dati provenienti dagli ordini provinciali, è sostanzialmente quello di permettere alla Federazione di monitorare quantitativamente ma soprattutto qualitativamente, tramite un'opportuna attività di reportistica di sintesi, l'operato dei professionisti su scala nazionale. Per questo motivo viene spesso richiesto all'azienda di compiere particolari elaborazioni dei dati in modo tale da poterli presentare opportunamente e consultare.

1.1.1 Professionisti presenti nel DB

Dato il legame che sussiste, dal momento in cui operano nello stesso settore medico-sanitario, tra odontoiatri, chirurgi e altre figure professionali mediche, il database *FNOMCEO* non raccoglie solamente dati relativi ai primi, ma anche quelli degli altri professionisti che operano sul territorio nazionale in ambito medico. I dati riguardanti professionisti esterni alla Federazione vengono archiviati esclusivamente per motivi statistici. In dettaglio i professionisti archiviati in *FNOMCEO* esercitano, secondo la

classificazione del *Ministero della Salute*[1], le seguenti professioni:

- Assistente sanitario
- Biologo
- Chimico
- Dietista
- Educatore professionale
- Farmacista
- Fisico
- Fisioterapista
- Igienista dentale
- Infermiere
- Infermiere pediatrico
- Logopedista
- Medico chirurgo
- Odontoiatra
- Odontotecnico
- Ortottista/Assistente di oftalmologia
- Ostetrica/o
- Ottico
- Podologo
- Psicologo
- Tecnico audiometrista
- Tecnico audioprotesista

[1] Sito del Ministero: `http://www.ministerosalute.it`

- Tecnico della fisiopatologia cardiocircolatoria e perfusione cardiovascolare
- Tecnico della prevenzione nell'ambiente e nei luoghi di lavoro
- Tecnico di neurofisiopatologia
- Tecnico educazione e riabilitazione psichiatrica e psicosociale
- Tecnico ortopedico
- Tecnico sanitario di radiologia medica
- Tecnico sanitario laboratorio biomedico
- Terapista della neuro e psicomotricità dell'età evolutiva
- Terapista occupazionale
- Veterinario

1.1.2 Applicazione Web

L'applicazione Web sviluppata da Omicron Technologies si divide in due parti: una pubblica e una privata. La parte pubblica[2] è accessibile da qualsiasi Web-User e permette, tramite l'ausilio di un motore di ricerca, l'individuazione e la consultazione di dati pubblici di ciascun professionista registrato nel database; dati come cognome e nome, iscrizioni, lauree e abilitazioni sono consultabili da qualsiasi terminale connesso alla rete Internet. La parte privata, differentemente, è accessibile solo da utenti provvisti di Username e Password e rende possibile la visualizzazione di rielaborazioni dei dati e di particolari reportistiche ad hoc richieste dalla Federazione.

[2]Parte pubblica: `http://application.fnomceo.it/Fnomceo/public/dettagliProfessionista.ot`

1.2 Perché Oracle?

Oracle è attualmente il DBMS più diffuso in ambito commerciale. Questo perché costituisce un sistema particolarmente sicuro sia per quanto riguarda la gestione dei privilegi degli utenti, che per la manutenzione dei dati. Sebbene a fini esclusivamente interrogativi (tramite l'uso di query) non costituisca uno strumento potente quanto il suo principale antagonista MySQL, Oracle è decisamente migliore per la gestione di grandi quantità di dati e, oltre un certo numero di relazioni presenti all'interno di un database, come specificato dai tecnici in azienda, Oracle è il sistema in assoluto più efficiente in tutti i campi. Questo è il principale motivo per cui l'azienda padovana ha costruito il DB utilizzando una tecnologia Oracle.

La Omicron Technologies lavora da anni utilizzando tecnologia Oracle, sebbene esso a differenza di MySQL comporti l'acquisto di una licenza e quindi un utilizzo di risorse economiche aziendali. Omicron Technologies possiede nella sua sede a Padova un Server Oracle sul quale sono allocati gran parte delle basi di dati dei progetti ad oggi fatti, compreso *FNOMCEO*.

Per ulteriori approfondimenti tecnici su Oracle si veda l'Appendice A.

1.3 Analisi qualitativa del DB

Prendendo il modello ER (Entity Relationship) come modello di astrazione di riferimento su cui svolgere un'analisi qualitativa, il database è sviluppato attorno ad un'entità centrale, chiamata *PROFESSIONISTA*, che raccoglie alcune informazioni riguardanti il singolo professionista. Il resto delle entità presenti è sostanzialmente suddiviso in cinque gruppi, ciascuno dei quali è collegato direttamente al professionista formando una configurazione che ricorda la topologia "a stella":

- l'insieme *LAUREA*: raccoglie tutte le informazioni relative ai titoli di studio conseguiti dal professionista, come l'università che ha rilasciato l'attestato di diploma, l'albo professionale, la specializzazione, ...
- l'insieme *ECM*[3] (Educazione Continua in Medicina): raggruppa tutti i dati relativi ai corsi di aggiornamento non obbligatori a cui il professionista partecipa, come la partecipazione, le caratteristiche del corso, l'accreditatore, i crediti assegnati, ...
- l'insieme *INFORMAZIONI PROFESSIONE*: include tutte le informazioni sull'esercizio del professionista, come quelle sulla sua attività, sui provvedimenti legali, sulle abilitazioni, ...
- l'insieme *PROFESSIONISTA ESTESO*: raccoglie tutte le informazioni estese, non presenti nell'entità principale, del professionista; ne fanno parte la residenza, il domicilio, la cittadinanza, ...

[3]ECM: `http://www.ministerosalute.it/ecm/ecm.jsp`

- l'insieme *ISCRIZIONE*: contiene le informazioni relative all'iscrizione del professionista agli ordini e le modifiche alle iscrizioni (come le cancellazioni e le loro causali);
- l'insieme *ALTRO*: raggruppa altre entità, per lo più isolate, come *UTENTE* (che contiene le informazioni per accedere all'area privata dell'applicazione Web).

Per vedere una parte significativa dello schema ER del database *FNOMCEO* si veda la Figura 1.1.

1.4 Analisi tecnica del DB

Quando il progetto è iniziato, nel 2004, l'azienda ha progettato la base di dati in maniera molto scrupolosa considerando la quantità di dati in entrata che era destinata a memorizzare. Per questo motivo lo schema finale risulta particolarmente complesso dato che le procedure per l'ottimizzazione delle strutture (come la normalizzazione e la ristrutturazione dello schema) sono state portate a termine in modo tale da permettere un risparmio considerevole della memoria del sistema nonché un taglio nei tempi impiegati per eseguire qualsiasi processo, a discapito della sua semplicità e comprensibilità.

A livello organizzativo sono state adottate alcune interessanti convenzioni, analizzate in seguito, che costituiscono la chiave principale per un lavoro da parte di più tecnici sul DB; inutile dire infatti che senza queste convenzioni lavorare su *FNOMCEO* risulterebbe estremamente

più complicato, facendo perdere anche importanti risorse economiche all'azienda.

Per rendere più veloci e sicure le operazioni del DBMS sul DB (come interrogazioni, gestioni dei flussi, estrazioni di dati, importazioni, ...) sono state costruite nello schema alcune tabelle aggiuntive, non previste nello schema concettuale ristrutturato. Queste tabelle si possono raccogliere sostanzialmente in due gruppi: il primo raccoglie quelle che hanno strettamente a che fare con le entità progettate inizialmente, il secondo invece include quelle create con lo scopo di controllo di alcuni processi che agiscono sul sistema. Le tabelle di quest'ultimo gruppo, differente da quelle del primo, non sono create per la memorizzazione dei dati in entrata.

Per quanto riguarda il primo gruppo, le tabelle incluse in esso fanno concettualmente parte di un'entità dello schema principale e il loro compito è quello di raccogliere dati storici che difficilmente verranno rielaborati, ma che devono essere comunque memorizzati per completezza. Tali tabelle vengono identificate come *tabelle storiche*.

Le tabelle del secondo gruppo invece, essendo concettualmente esterne al contesto del professionista, servono per regolare i flussi dei dati e per monitorare le richieste dell'applicazione Web. Esse vengono denominate *tabelle di supporto*.

1.4.1 Convenzioni generali adottate

I nomi delle tabelle sono stati codificati in caratteri *ASCII Unicode*, omettendo quindi i caratteri conosciuti come "speciali"; essi corrispondono ad una breve descrizione del contenuto. Al posto del carattere di spazio, nei nomi si utilizza il carattere *underscore* ('_') perché il DBMS non ammette nomi con spazi interni. Anche i nomi dei campi seguono le stesse regole utilizzate per nominare le tabelle. Le chiavi esterne sono nominate usando la convenzione *[nome tabella a cui si riferiscono]_[nome campo]*. Quest'ultima regola usata per le chiavi esterne risulta particolarmente comoda per quanto riguarda l'elaborazione di query su più tabelle che usano l'operatore di giunzione.

Le viste sono largamente utilizzate all'interno del database; il loro compito è quello di supporto all'elaborazione dei report. Questo perché Oracle non "digerisce" le giunzioni annidate tra tabelle e quindi lo strumento della vista per realizzare report viene praticamente sempre utilizzato, anche al fine di arrivare a report più complessi tramite una procedura "step by step" partendo dalla query più elementare.

Dal momento in cui Oracle è un sistema case-insensitive, ossia che non riconosce la differenza fra caratteri maiuscoli e minuscoli, nei paragrafi successivi i nomi delle tabelle e dei loro campi, come quello del DB, saranno indicati in lettere maiuscole.

1.4.2 Tabelle storiche

Qualora una tabella del DB, soggetta ad operazioni corpose di popolamento, contenga dati non soggetti a operazioni (rielaborazioni o query), la tecnica utilizzata è quella di creare una tabella secondaria con lo stesso nome e gli stessi attributi (o, al più, con l'aggiunta di un indice) seguita dal suffisso *_HISTORY*. Tale tabella raccoglie tutti i dati presenti nella tabella iniziale che non sono più richiesti con frequenza dal sistema, ma che devono comunque, nell'eventualità, essere disponibili. Questo crea tecnicamente un troncamento della tabella di partenza e fa sì che i dati storici possano essere archiviati su una struttura molto pesante che difficilmente verrà interrogata o sarà soggetta a richieste particolari data la sua costituzione. Tale processo alleggerisce la tabella principale ottimizzando quindi i tempi di esecuzione di qualsiasi tipo di operazione su di essa.
Il contenuto della relazione originale viene quindi spezzato in due parti, ma solo a scopo operazionale; concettualmente bisogna sottolineare che, sebbene le tabelle siano fisicamente due, l'entità rimane unica.

1.4.3 Tabelle di supporto

Il sistema (applicazione Web, DB e DBMS), oltre che memorizzare effettivamente i dati sul DB, gestisce particolari processi su di essi. Per controllare meglio queste dinamiche, vengono utilizzate delle tabelle di supporto che raccolgono informazioni relative soprattutto alle operazioni di trasferimento dati. Queste tabelle sono memorizzate all'interno di

FNOMCEO e svolgono un compito cruciale in quanto permettono all'azienda, tramite l'analisi dei loro contenuti, il monitoraggio del flusso di dati e il controllo degli errori interni al sistema.

Specificatamente queste tabelle appartengono a quattro distinte categorie:

1. Tabelle di LOG[4].
2. Tabelle temporanee di eliminazione.
3. Tabelle temporanee di inserimento.
4. Tabelle per l'esportazione.

Tabelle di LOG

Queste tabelle raccolgono i LOG del Web. Gli utenti infatti hanno la possibilità di caricare dei dati nel DB e di consultarli dall'applicazione, generando flussi di informazioni; per questo motivo vengono raccolte le informazioni di flusso relative alla sessione di trasferimento Client-Server.

Per tenere traccia degli errori riscontrati durante il trasferimento di dati dal Web ci si appoggia alle convenzioni dettate da W3C (World Wide Web Consortium) in modo tale da rendere i dati contenuti in queste tabelle sufficientemente conformi e preservare tutte le informazioni utili.

[4]LOG: con questo termine si intende la registrazione cronologica delle operazioni man mano che vengono eseguite; nel caso di *FNOMCEO* per LOG si intende solamente la registrazione delle operazioni provenienti dal Web

Il W3C elenca le informazioni seguenti come quelle che una tabella di Web-LOG dovrebbe necessariamente contenere:

Date, Time (informazioni temporali della connessione)

Clien IP, Server IP (indirizzi IP del Client e del Server)

Service Name (client)

Client-Server method (GET, POST)

Status (errore come errno)

Client Browser (Firefox, Safari, IE, Opera, . . .)

Risorse a cui il Client stava accedendo (hmtl, asp)

Server-Client: Server Byte (byte inviati dal server)

Referrer (sito di provenienza)

Cookie (cookie HTTP)

Server Name (nome del server che genera il log)

Protocol Version (protocollo di comunicazione Client-Server)

Time Taken (tempo di connessione)

Server Port (porta di connessione lato Server)

Byte received/sended (byte ricevuti e inviati dal Server)

Client-Server Username (solo se il metodo è POST)

Tabelle temporanee di eliminazione

Dal momento in cui l'eliminazione di record dal DB è un processo molto delicato, l'operazione di rimozione viene suddivisa in due passi, questo per garantire una maggiore sicurezza al sistema. Nel primo passo si memorizza il record da cancellare in una tabella chiamata con il nome

della tabella contenente il record seguito dal suffisso *_TO_DELETE*. Nel passo successivo la richiesta viene analizzata (dal sistema o direttamente da un tecnico) e il record corrispondente nella tabella principale e in quella che raccoglie i record in attesa di eliminazione viene eliminato.

Solitamente si cerca di eliminare meno dati possibili da una base di dati, questo per motivi informativi; infatti quando un dato viene cancellato non è più possibile reperirlo. Nel DB in questione il professionista deve essere cancellato dall'entità centrale qualora non eserciti più effettivamente l'attività perché il DB raccoglie informazioni riguardanti i professionisti attivi. Tuttavia, nelle tabelle storiche, le informazioni sull'attività di un professionista non più presente nella relazione *PROFESSIONISTA* vengono mantenute.

Tabelle temporanee di inserimento

L'inserimento di dati funziona in maniera molto simile alla loro eliminazione. In questo caso però non tutte le richieste di inserimento vengono salvate su una tabella a parte prima di procedere all'effettiva memorizzazione nella tabella principale. Questo perché, se l'eliminazione è un processo che coinvolge solitamente qualche record isolato, l'inserimento ne coinvolge un numero assai più significativo rendendo un'eventuale operazione analoga a quella vista per l'eliminazione troppo pesante.
Una volta che i dati da inserire sono resi disponibili dagli Ordini e dal Ministero (per approfondire si veda il Capitolo 3), essi vengono ripuliti e riadattati prima di essere effettivamente memorizzati nelle tabelle

principali del DB. Questo viene fatto perché, appena inviati, i dati non risultano ancora sufficientemente conformi dato che provengono da fonti diverse e che i requisiti per la loro immissione non è detto che siano verificati.
Per svolgere le operazioni di pulizia e controllo dei dati in ingresso, essi vengono salvati in una tabella temporanea allocata all'interno di un altro database Oracle.

Qualora, una volta che i dati sono stati controllati e rielaborati, sia presente comunque qualche anomalia (valori nulli non reperibili o presenza di chiavi esterne senza riferimento), i dati vengono comunque tutti memorizzati in *FNOMCEO*, ma su tabelle diverse. I dati completi vengono salvati nella tabella principale di destinazione, quelli non completi invece in una tabella temporanea, che raccoglie tutti quei record che sono in attesa di un'ulteriore elaborazione e che non possono ancora essere direttamente salvati nella tabella principale. Tale tabella, contenente i record temporanei, ha lo stesso nome di quella dove sono stati memorizzati i dati completi seguito dal suffisso *_TMP*.

Tabelle per l'esportazione

I dati presenti nel DB sono soggetti a esportazioni dovute soprattutto a motivi di manutenzione del Server che ospita le basi di dati aziendali. Quando i dati vengono esportati, dato che si tratta di una procedura automatica, risulta estremamente utile tenere traccia delle operazioni e del loro esito. Nel DB *FNOMCEO* spesso i dati vengono esportati su file

.XML, data la compatibilità che ha questo formato con qualsiasi tipo di sistema di destinazione.
Nel caso ad esempio di un'operazione di esportazione mediante XML viene tenuta traccia del nome del file contenente i dati, del tempo di elaborazione, degli errori, dei codici degli errori e dell'esito dell'esportazione su un'apposita tabella.

XML (Extended-HTML) è il linguaggio più diffuso per salvare dati strutturati su file di testo. Esso è simile al linguaggio HTML (Hyper Text Markup Language) e definisce i metadati[5] tramite TAG salvandoli in file con estensione .DTD (Document Type Definition). Gli oggetti (strutture) sono compresi tra i tags <!ELEMENT... > e </ELEMENT>, tra i quali c'è praticamente il contenuto del comando `CREATE TABLE` del linguaggio SQL.
XML ha inoltre la capacità di poter decifrare facilmente le strutture gerarchiche e costituisce un mezzo perfetto per trasferire dati tra sistemi incompatibili. Per queste ragioni è ampiamente utilizzato dalle aziende per l'esportazione di dati.

1.5 Analisi delle tabelle presenti

Questa sezione contiene un'analisi delle tabelle presenti nel Database *FNOMCEO*. Per rendere più comprensibile lo schema, le relazioni del DB sono state ordinate in una tabella che indica la loro funzione all'interno del sistema, le strategie di aggiornamento, la frequenza con cui i

[5]Metadata: dati che descrivono dati, necessari per dare un senso a qualsiasi schema di dati

dati vengono caricati o aggiornati e l'evento caratteristico che determina la memorizzazione di un nuovo record.

Le strategie di aggiornamento (o di update) della tabella possono essere di cinque tipi diversi: *no update* (la tabella non viene mai aggiornata), *insert* (la tabella è soggetta all'inserimento di dati e alla modifica di singoli record), *reload* (i dati contenuti vengono interamente ricaricati e aggiornati), *truncate* (viene creata una tabella storica dei contenuti della tabella, spezzando i dati) e *insert only* (i dati possono essere solamente inseriti, non modificati o cancellati). I tipi di strategie di update *insert* e *insert only* non possono coesistere nella stessa tabella perché il primo indica anche che i dati possono essere modificati mentre il secondo, tipico delle tabelle storiche, no. Nello schema sono presenti tabelle per ogni diversa strategia di update: alcune di esse infatti hanno lo scopo di archiviazione e memorizzazione continua di nuovi dati, altre servono solo per contenere dati storici e altre ancora contengono esclusivamente informazioni conosciute a priori, note in fase di progettazione del DB.
Le funzioni che svolge la tabella all'interno dello schema possono essere di tre tipi, a seconda del contenuto informativo presente: *data transfer* (informazioni su dinamiche di trasferimento dati), *archiviazione* (memorizzazione di informazioni), *informazione data* (informazioni conosciute a priori e non soggette solitamente a modifiche: provincie, regioni, . . .).

NOME TABELLA	STRATEGIA UPDATE	FREQUENZA DI CARICO	FUNZIONE TABELLA	EVENTO CRESCITA
ABILITAZIONE	insert / reload / truncate	mensile	archiviazione	nuova abilitazione
ABILITAZIONE_HISTORY	insert only	mensile	archiviazione	
ACCREDITATORE	insert / reload	su richiesta	informazione data	nuovo accreditatore
ALBO	no update	su richiesta	informazione data	nuovo albo
ALTRO	insert / reload	mensile	archiviazione	nuova convenzione SSN
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	mensile	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 / reload	mensile	archiviazione	
ESENZIONE_TIPO	insert / reload	mensile	archiviazione	nuovo tipo esenzione
ESERCIZIO	insert / reload	mensile	archiviazione	nuovo esercizio
ESERCIZIO_HISTORY	insert only	annuale	archiviazione	
EXPORT_XML	insert / delete	su richiesta	data tranfer	nuova esportazione
FILE_DATA	insert / delete	su richiesta	data tranfer	caricamento file
IMPORT_CODICE_FISCALE	insert / delete	su richiesta	data tranfer	
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 tranfer	caricamento file
LOG_PROF_ERROR	insert / delete	su richiesta	data tranfer	nuovo 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	mensile	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	su richiesta	archiviazione	
PROFESSIONISTA_TO_DELETE	insert / delete	su richiesta	data tranfer	professionista da eliminare
PROFESSIONISTA_VARIATO	insert / delete	su richiesta	data tranfer	modifica professionista
PROFESSIONISTA_VARIATO_BUG	insert / delete	su richiesta	data tranfer	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 specializzazione
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 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	

1.6 Schema ER

Lo schema ER è un ottimo strumento per capire il funzionamento di un database. In questo caso il suo ultimo stadio (quello ottenuto dopo la ristrutturazione dello schema iniziale, contenente tutte le entità) non è graficamente rappresentabile su una pagina per motivi di complessità, tuttavia una porzione significativa di esso può facilitarne la comprensione.

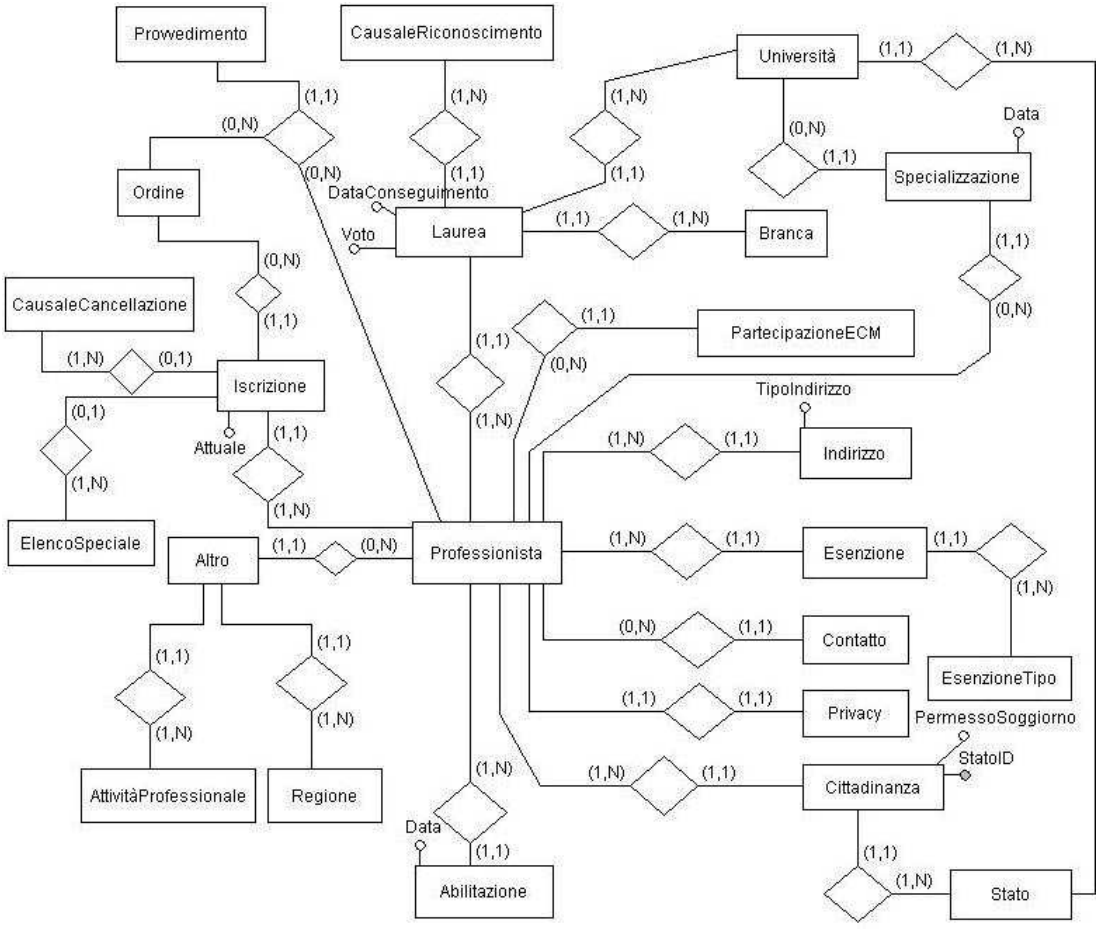

Figura 1.1: Porzione dello schema ER del database FNOMCeO

Note sullo schema

1. Le entità *Cittadinanza*, *Specializzazione*, *Laurea* e *Abilitazione* si riferiscono all'istanza e possono essere interpretate come attributi del singolo professionista separati dall'entità *Professionista*.
2. L'entità *Indirizzo* contiene gli indirizzi di residenza, domicilio e studio commerciale con identificazione ('R', 'D', 'C') tramite l'attributo *TipoIndirizzo*.
3. Un professionista può avere più abilitazioni e più cittadinanze. Nell'entità *Cittadinanza* vengono anche memorizzati eventuali permessi di soggiorno e gli status di rifugiato politico.
4. L'attributo *Attuale* nell'entità *Iscrizione* è un attributo booleano che indica se l'iscrizione è da considerarsi valida in un certo istante temporale.
5. Quando un'iscrizione viene cancellata, la chiave esterna di *Iscrizione* su *CausaleCancellazione* non ha più valore 'NULL' (associato di default nel momento dell'iscrizione) e automaticamente l'attributo *Attuale* viene settato sul valore 'FALSE'.
6. Un Ordine ha la facoltà di prendere un determinato provvedimento disciplinare nei confronti di un professionista, mandandolo a giudizio.
7. L'entità *Altro* contiene i dati relativi alle convenzioni sui professionisti da parte del Servizio Sanitario Nazionale (SSN).

Capitolo 2

Creazione di Report su FNOMCEO

2.1 Introduzione

Oggetti importanti di studio per un sistema di questo tipo sono le dinamiche delle richieste al DB provenienti dal Web. Se da un lato infatti la memorizzazione di blocchi di dati in entrata non richiede particolari velocità di esecuzione permettendo che il processo impieghi parecchi minuti per essere portato a termine, le richieste provenienti dal Web esigono tempi di risposta molto più rapidi, solitamente nell'ordine di qualche secondo.

Si pensi, a titolo di esempio, a quanto possa essere inutile un caricamento di pagina da un comunissimo Web-Browser (come Firefox, Explorer, Safari o Opera) che, per essere completato, richieda alcuni minuti.

2.2 Tra Web-Server e database: il Flat-Database

La velocità di evasione da parte di un Web-Server delle richieste avanzate a dati contenuti in un DB interno solitamente non è legata alla banda di connessione. I dati che vengono richiesti al sistema da parte del Client Web infatti non sono usualmente di tipo multimediale ma testuale e dunque, arrivati alle tecnologie attuali di connessione (ADSL, fibre ottiche, Wi-MAX, ...), il loro tempo di trasferimento attraverso il canale di comunicazione è, in molti casi, trascurabile. Premesso questo e sottolineando il fatto che esistono casi in cui il "collo di bottiglia" è dato dalla connessione, i costi temporali sono quindi dovuti principalmente alla fase di estrazione dei dati. Il tempo impiegato per l'evasione è quindi in questo caso approssimativamente equiparabile al tempo di esecuzione da parte del DBMS della query.

Questo è il motivo fondamentale per cui molti archivi informatici di una certa dimensione collegati ad applicazioni Web si appoggiano su basi di dati ausiliarie, chiamate anche "piatte" (Flat-DB) in quanto non contengono particolari strutture o informazioni multidimensionali. I Flat-DB sono progettati unicamente per garantire un'alta efficienza nelle esecuzioni di particolari query. Al loro interno vengono spesso memorizzati direttamente i risultati stessi così come devono essere consegnati al pre-processore di ipertesti[1] del Server.

È ovvio che il tempo impiegato da un DBMS per fornire i dati conte-

[1]Per elaborare i dati necessari a formare i file (HTML) che vengono mandati in formato HTTP, spesso vengono utilizzati dei pre-processi, come possono essere quelli un Server Apache (PHP) o di un Asp.Net (ASP).

nuti in una singola tabella è assolutamente minore rispetto a quello per dati contenuti su più tabelle che necessitano anche eventualmente di più controlli.

In questo caso, la FNOMCeO ha richiesto che si possano visualizzare dall'area privata del suo sito Internet particolari elaborazioni dei dati contenuti nel DB. Queste richieste, alcune delle quali verranno trattate in dettaglio in questo capitolo, sono tutte traducibili in query Oracle-SQL più o meno complesse dal momento in cui i dati sono tutti memorizzati nel DB Oracle *FNOMCEO*.

Dopo aver scritto le query in Oracle-SQL, si dovrà creare un Flat-DB dentro il quale saranno definite le tabelle dove verranno salvati i risultati ottenuti dalle query.

Il sistema utilizzato per la creazione e la gestione del Flat-DB è MySQL e le tabelle interne adottano tutte il motore MyISAM (per approfondimenti si veda l'Appendice A).

La scelta del DBMS è motivata principalmente da motivi economici. MySQL infatti, a differenza di Oracle, come già detto in precedenza, non necessita di alcun contratto di licenza. Oltre ai motivi economici, particolarmente importante è il fatto che MySQL si integra con facilità con il Web, utilizzando soprattutto minori risorse in questo caso rispetto ad un ipotetico DB Oracle contenente gli stessi dati e relazioni, ma soprattutto che MySQL gestisce una query-cache che permette di rendere nulli i tempi di elaborazione di una query, nel caso in cui essa venga richiesta più volte.

La scelta del motore invece è dovuta al fatto che, dal momento in cui esso non considera informazioni relative a vincoli di integrità referenziale (omettendo in elaborazione quelli che sono i controlli incrociati fra le chiavi), con MyISAM il DBMS è molto più veloce nell'elaborare le query rispetto altri Engine ad oggi disponibili.

2.3 Il flusso di dati

Una volta creato il DB piatto, un ulteriore problema è quello di capire come far arrivare i dati al suo interno e soprattutto con quale frequenza. Dato che l'aggiornamento della base di dati *FNOMCEO* soggetta ai report viene eseguito con frequenza annuale per le tabelle più importanti, come sottolineato dall'analisi svolta nel Capitolo 1, non ha alcun senso aumentare tale frequenza per il popolamento delle tabelle del database ausiliario MySQL-MyISAM perché i dati contenuti in esso non subirebbero alcuna modifica effettiva. Il Flat-DB verrà popolato quindi ogni anno, al termine dell'aggiornamento delle tabelle in questione presenti in *FNOMCEO*.

Proprio a causa della frequenza di aggiornamento dei dati è possibile che l'utente che ha accesso all'area privata dal sito non possa visualizzare il risultato delle query fatte sui dati aggiornati, perché egli avrà solamente la possibilità di vedere quelli dei dati caricati durante l'ultimo trasferimento dal DB Oracle a quello MySQL.

Sebbene nella maggior parte dei casi riguardanti problemi di flusso sia preferibile l'utilizzo di un programma automatico batch che invia i

dati una volta che essi sono stati aggiornati, a fini pratici, data la bassa frequenza dell'operazione e il fatto che il processo di immissione nel DB principale va controllato e monitorato richiedendo tempi diversi, il trasferimento viene eseguito da un tecnico direttamente dall'azienda.

2.4 Creazione dei report

In questa sessione verranno elencati alcuni report significativi creati e verrà riportato il codice usato per eseguirli sul DB, oltre che alcuni risultati sotto forma relazionale dell'output Oracle.

La difficoltà di questi report non sta tanto nell'elevato numero di relazioni coinvolte, quanto nell'organizzare l'output nello specifico modo richiesto dalla Federazione. Dal momento in cui è stato deciso infatti di memorizzare sul Flat-DB direttamente i risultati delle richieste possibili provenienti dal'applicazione Web, i dati vanno registrati esattamente come devono essere pre-processati dal Web-Server nell'istante che precede immediatamente il loro invio al Client.

Siccome i sistemi Oracle non accettano, per motivi computazionali, operazioni di giunzione (JOIN) annidate all'interno della stessa query, per riuscire a produrre la reportistica richiesta dalla FNOMCeO, è necessario ricorrere alla creazione di viste in modo tale da produrre, procedendo dall'interrogazione più elementare, secondo un approccio "bottom-up", il report ricercato.

2.4.1 REPORT_DOPPI

Descrizione Elenco del numero di professionisti iscritti ad entrambi gli albi della Federazione (Medici Chirurgi ed Odontoiatri) ordinati per regione, provincia e zona geografica.

Relazioni coinvolte PROFESSIONISTA, ISCRIZIONE, REGIONE, PROVINCIA.

Oracle-SQL

```
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
        (PROVINCIA_ID, PROFESSIONISTA_ID, NUMEROISCRIZIONI, ALBO, SESSO) AS
SELECT
        ISCRIZIONE.PROVINCIA_ID,
        ISCRIZIONE.PROFESSIONISTA_ID,
        COUNT(ISCRIZIONE.PROFESSIONISTA_ID) AS NUMEROISCRIZIONI,
        MIN(ISCRIZIONE.ALBO_TIPO) AS ALBO,
        SESSO
FROM
        ISCRIZIONE INNER JOIN V_PROF_ATTIVI ON V_PROF_ATTIVI.ID = ISCRIZIONE.PROFESSIONISTA_ID
GROUP BY
        ISCRIZIONE.PROFESSIONISTA_ID, SESSO, PROVINCIA_ID;
COMMENT ON TABLE V_ISCRITTI IS
        'Ritorna␣l''elenco␣delle␣iscrizioni␣attuali␣dei␣professionisti␣attivi';
-------------------------------------------------------
CREATE OR REPLACE VIEW V_DOPPI AS
SELECT
        PROVINCIA_ID,
        COUNT(*) AS ISCRITTI
FROM
        V_ISCRITTI
WHERE
        NUMEROISCRIZIONI = 2
GROUP BY
        PROVINCIA_ID;
COMMENT ON TABLE V_DOPPI IS 'Ritorna␣il␣numero␣degli␣iscritti␣doppi␣per␣id␣di␣provincia';
-------------------------------------------------------
CREATE OR REPLACE VIEW REPORT_DOPPI
        (ISCRITTI, PROVINCIA, REGIONE, ZONA_GEOGRAFICA, ZONA_GEOGRAFICA_ID) AS
SELECT
        V_DOPPI.ISCRITTI,
        PROVINCIA.DESCRIZIONE AS PROVINCIA,
        REGIONE.DESCRIZIONE AS REGIONE,
        REGIONE.ZONA_GEOGRAFICA_DESCRIZIONE AS ZONA_GEOGRAFICA,
        REGIONE.ZONA_GEOGRAFICA_ID AS ZONA_GEOGRAFICA_ID
```

```
FROM
        V_DOPPI, REGIONE, PROVINCIA
WHERE
        V_DOPPI.PROVINCIA_ID = PROVINCIA.ID AND
        PROVINCIA.REGIONE_ID = REGIONE.ID;
COMMENT ON TABLE REPORT_DOPPI IS 'Ritorna␣il␣numero␣dei␣doppi␣iscritti␣per␣provincia';
```

2.4.2 REPORT_PROFESSIONISTA_NASCITA

Descrizione Elenco ordinato per fascia d'età del numero di professionisti presenti in *FNOMCEO*.

Relazioni coinvolte PROFESSIONISTA, ORDINE.

Oracle-SQL

```
CREATE OR REPLACE VIEW V_PROFESSIONISTA_NASCITA
        (ATTIVO, DATA_NASCITA, ANNO_NASCITA, ANNO_CORRENTE, PROVINCIA) AS
SELECT
        ATTIVO,
        DATA_NASCITA,
        TO_CHAR(DATA_NASCITA, 'YyyY') AS ANNO_NASCITA,
        TO_CHAR(SYSDATE, 'YyyY') AS ANNO_CORRENTE,
        ORDINE.PROVINCIA_ID
FROM
        PROFESSIONISTA, ORDINE
WHERE
        ATTIVO = '1' AND
        PROFESSIONISTA.ORDINE_ATTUALE_ID = ORDINE.ID
ORDER BY
        DATA_NASCITA DESC;
-------------------------------------------------------
CREATE OR REPLACE VIEW REPORT_PROFESSIONISTA_NASCITA (QUANTI, COSA, ORDINE) AS
SELECT COUNT(*) AS QUANTI, 'MINORE␣di␣24' AS COSA, '01' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 24 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣24␣e␣i␣29' AS COSA, '02' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 29 AND ANNO_CORRENTE-ANNO_NASCITA >= 24 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣29␣e␣i␣34' AS COSA, '03' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 34 AND ANNO_CORRENTE-ANNO_NASCITA >= 29 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣34␣e␣i␣39' AS COSA, '04' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 39 AND ANNO_CORRENTE-ANNO_NASCITA >= 34 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣39␣e␣i␣44' AS COSA, '05' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 44 AND ANNO_CORRENTE-ANNO_NASCITA >= 39 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣44␣e␣i␣49' AS COSA, '06' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 49 AND ANNO_CORRENTE-ANNO_NASCITA >= 44 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣49␣e␣i␣54' AS COSA, '07' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 54 AND ANNO_CORRENTE-ANNO_NASCITA >= 49 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣54␣e␣i␣59' AS COSA, '08' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 59 AND ANNO_CORRENTE-ANNO_NASCITA >= 54 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣59␣e␣i␣64' AS COSA, '09' AS ORDINE
```

```
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 64 AND ANNO_CORRENTE-ANNO_NASCITA >= 59 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣64␣e␣i␣69' AS COSA, '10' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA < 69 AND ANNO_CORRENTE-ANNO_NASCITA >= 64 UNION
SELECT COUNT(*) AS QUANTI, 'TRA␣i␣69␣e␣i␣74' AS COSA, '11' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA <= 74 AND ANNO_CORRENTE-ANNO_NASCITA >= 69 UNION
SELECT COUNT(*) AS QUANTI, 'MAGGIORE␣di␣75' AS COSA, '12' AS ORDINE
FROM V_PROFESSIONISTA_NASCITA
WHERE ANNO_CORRENTE-ANNO_NASCITA >= 75
ORDER BY ORDINE ASC;
```

2.4.3 PROFESSIONISTI_30CREDITI_2008

Descrizione Elenco ordinato, per albo a cui sono iscritti i professionisti, del numero di professionisti che hanno partecipato durante il 2008 a corsi ECM ottenendo un numero di crediti maggiore di 30, elencando anche l'ammontare relativo rispetto al totale dei professionisti iscritti al loro stesso albo (per informazioni sulle professioni si veda il Capitolo 3).

Relazioni coinvolte PARTECIPAZIONE_ECM, PROFESSIONE.

Oracle-SQL

```
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;
```

Output

DESCRIZIONE	*NUM_30_CREDITI*	*PERCENTUALE*
Assistente sanitario	52	8
Biologo	2.717	29
Chimico	55	17
Dietista	35	11
Educatore professionale	11	18
Farmacista	3.857	17
Fisico	2	11
Fisioterapista	1.980	41
Igienista dentale	23	4
Infermiere	12.575	14
Infermiere pediatrico	122	7
Logopedista	194	34
Medico chirurgo	60.813	31
Odontoiatra	6.934	25
Odontotecnico	13	12
Ortottista/Assistente di oftalmologia	39	13
Ostetrica/o	390	16
Ottico	30	55
Podologo	23	13
Psicologo	1.521	24
Tecnico audiometrista	6	8
Tecnico audioprotesista	82	28
Tecnico della fisiopatologia cardiocircolatoria e perfusione cardiovascolare	1	2
Tecnico della prevenzione nell'ambiente e nei luoghi di lavoro	9	13
Tecnico di neurofisiopatologia	2	8
Tecnico educazione e riabilitazione psichiatrica e psicosociale	5	22
Tecnico ortopedico	82	24

Tecnico sanitario di radiologia medica	749	15
Tecnico sanitario laboratorio biomedico	48	11
Terapista della neuro e psicomotricità dell'età evolutiva	115	37
Terapista occupazionale	8	25
Veterinario	688	16

2.5 Creazione del Flat-Database

La procedura di creazione di un DB MySQL è abbastanza semplice. In questo caso c'è solamente bisogno di un Server MySQL che abbia sufficienti risorse per il nostro scopo; Server già presente in azienda.

Una volta creato il DB di destinazione utilizzando un software con interfaccia grafica opportuno (come phpMyAdmin se si tratta di un Server Apache) oppure la classica sintassi SQL `CREATE DATABASE` da terminale, si creano le tabelle che verranno popolate con i risultati ottenuti dai report elaborati, descritti in parte nella sezione precedente.

2.5.1 Creazione tabelle

Le tabelle di destinazione presenti nel DB MySQL vengono denominate convenzionalmente con lo stesso nome dei report e i loro campi, per ovvie ragioni, dal momento in cui esse devono contenere i risultati di questi ultimi, devono essere gli stessi.

In questa fase di creazione delle tabelle all'interno del Flat-DB è opportuno porre attenzione ai vincoli sui campi (datatype). Nella definizione, durante la creazione di una nuova tabella, del tipo di un campo infatti Oracle e MySQL hanno alcune importanti differenze. Il tipo definito sul

DB di partenza è un datatype Oracle, mentre quello di destinazione è MySQL; bisogna quindi utilizzare con scelte ponderate le equivalenze tra datatype dei due DBMS (Appendice A).

Per quanto riguarda i vincoli di chiave primaria o quelli relativi alla presenza di valori *NULL*, non è necessario definirli sul Flat-DB. Infatti gli output ottenuti dai report sono stati elaborati su dati già inseriti correttamente nel DB Oracle, ripuliti e completati in fase di inserimento (per un esempio di pulizia dati e completamento in inserimento si veda il Capitolo 3).

2.5.2 Codice

Vengono ora riportate, a titolo esemplificativo, le creazioni delle tabelle che conterranno i risultati ottenuti dai tre report descritti precedentemente. Dal momento in cui queste tabelle vengono create nella base di dati MySQL, il codice utilizzato è MySQL-SQL. Nella creazione della tabella MySQL è indicato anche l'Engine utilizzato, sebbene MyISAM sia il motore di default e quindi non sia necessario specificarlo.

Per evidenziare le differenze tra i datatype viene inoltre mostrata la sintassi di creazione delle stesse tabelle in un DB Oracle.

MySQL

```
---- MySQL
CREATE TABLE REPORT_DOPPI
(
        ISCRITTI NUMBER,
        PROVINCIA VARCHAR(50),
        REGIONE VARCHAR(40),
        ZONA_GEOGRAFICA VARCHAR(20),
        ZONA_GEOGRAFICA_ID VARCHAR(2)
) type=MyISAM;
CREATE TABLE REPORT_PROFESSIONISTA_NASCITA
```

```
(
   QUANTI BIGINT,
   COSA VARCHAR(15),
   ORDINE VARCHAR(2)
) type=MyISAM;
CREATE TABLE PROFESSIONISTA_30CREDITI_2008
(
   DESCRIZIONE VARCHAR(150),
   NUM_30_CREDITI BIGINT,
   PERCENTUALE BIGINT
) type=MyISAM;
```

Oracle-SQL

```
---- Oracle-SQL
CREATE TABLE REPORT_DOPPI
(
        ISCRITTI NUMBER,
        PROVINCIA VARCHAR2(50 BYTE),
        REGIONE VARCHAR2(40 BYTE),
        ZONA_GEOGRAFICA VARCHAR2(20 BYTE),
        ZONA_GEOGRAFICA_ID CHAR(2 BYTE)
);
CREATE TABLE REPORT_PROFESSIONISTA_NASCITA
(
   QUANTI DECIMAL(22),
   COSA VARCHAR2(15),
   ORDINE CHAR(2)
);
CREATE TABLE PROFESSIONISTA_30CREDITI_2008
(
   DESCRIZIONE VARCHAR2(150),
   NUM_30_CREDITI DECIMAL(22),
   PERCENTUALE DECIMAL(22)
);
```

2.5.3 Popolamento del Flat-Database

Una volta creato il Flat-DB e le tabelle su cui salvare i dati, l'ultima operazione da fare è quella della popolazione delle tabelle con i risultati dei report. Quest'operazione sarà l'unica tra quelle viste che verrà ripetuta. Annualmente infatti, una volta ricevuti i dati sui professionisti da Federazione e Ministero, il Flat-DB verrà ripopolato con i contenuti dei report aggiornati.

Il problema è quello di gestire un ponte di comunicazione tra un DBMS e l'altro in modo da trasferire i dati al DB MySQL. Il metodo

utilizzato è stato quello di creare un file di testo .CSV[2] (formato estremamente portabile e perfettamente compatibile anche con Microsoft Excel©) contenente i risultati delle query eseguite sul database Oracle. Questo file viene letto direttamente da MySQL in fase di importazione.

Grazie a programmi di Client-SQL come SQuirreL[3] che permettono di operare su DBMS basati sul linguaggio SQL tramite un'interfaccia grafica, le operazioni di esportazione ed importazione dei dati utilizzando file .CSV sono gestite in automatico.

I file .CSV creati sono privi di header e non contengono altro che l'elenco dei dati formattati per essere inseriti nel DB di destinazione; in dettaglio, i singoli record sono separati dal carattere di fine linea 'CRLF', le colonne dal carattere ',' e la codifica dei dati è UTF-16.

[2]CSV (Comma-Separated Values): è un formato di file basato su file di testo utilizzato per l'importazione ed esportazione, ad esempio da fogli elettronici o database, di una tabella di dati.

[3]SQuirrel-SQL: `http://squirrel-sql.sourceforge.net/`

Capitolo 3

Importazione Dati in FNOMCEO

3.1 Introduzione

3.1.1 Partecipazioni ai corsi

I professionisti, durante il loro esercizio, svolgono corsi di aggiornamento professionale. Nel caso dei professionisti a cui il database afferisce, si tratta di attività formative certificate dal Ministero della Salute. L'ente ministeriale svolge solitamente anche la funzione di accreditatore, assegnando a ciascun corso di aggiornamento professionale un determinato numero di crediti. Ogni professionista è libero di provvedere al proprio aggiornamento professionale in piena autonomia, senza nessun obbligo di partecipazione.

Questi corsi ricadono all'interno dei programmi di Educazione Continua in Medicina (E.C.M.).

Nel DB è presente una tabella denominata *PARTECIPAZIONE_ECM* che raccoglie tutti i dati relativi alle partecipazioni dei professionisti ai corsi di aggiornamento, divisi nelle due categorie di Eventi e Progetti

Formativi Aziendali (PFA).

3.1.2 Come funziona l'inserimento dei dati

I dati relativi alle partecipazioni dei professionisti ai corsi sono raccolti dal Ministero della Salute e successivamente inviati in azienda per l'inserimento nel database. Il loro inserimento, per quanto possa sembrare una procedura banale e facilmente automatizzabile, richiede in realtà una serie di fasi che, per la loro natura, necessitano di una particolare competenza tecnica e hanno bisogno di una supervisione tecnica umana.

In dettaglio le fasi sono:

1. Rielaborazione dei dati inviati dal Ministero, rendendoli puliti, completi e uniformi.
2. Codifica degli eventuali errori trovati al loro interno.
3. Memorizzazione sul database *FNOMCEO*.

La fase finale di memorizzazione dei record può essere fatta solamente una volta che i dati sono pronti per entrare e quindi rispondono a determinati requisiti di conformità.

Per procedere con le varie fasi necessarie per l'elaborazione dei dati in ingresso, in questo caso viene fatta una copia dei dati ricevuti in un database Oracle secondario (denominato *COGEAPS_TMP*) in modo tale da poter recuperare gli originali in qualsiasi momento, nel caso si riscontrassero degli errori in fase di manipolazione. Una volta terminate le operazioni sulla copia dei dati e controllato che sia stato raggiunto

il risultato atteso, ossia che i dati siano pronti ad essere archiviati in *FNOMCEO*, si procede ad inserire direttamente la copia rielaborata dei dati e a cancellare quelli presenti in *COGEAPS_TMP*.

3.2 Documentazione: un concetto importante

Un sistema di archiviazione di dati ben progettato deve possedere anche una documentazione. Con documentazione si intende il "mettere per iscritto" la descrizione dei processi standard, ossia la conversione degli algoritmi che agiscono nel sistema in documenti testuali comprensibili in modo che essi siano leggibili in maniera chiara e interpretabili univocamente da qualsiasi tecnico che dovrà lavorare sul database. Un buon sistema informativo necessita di una propria documentazione per garantire la sua adattabilità. Il concetto di documentazione è strettamente collegato alla durata temporale di una base di dati.

Per quanto riguarda il database in questione *FNOMCEO*, la documentazione è stata salvata direttamente sul Web-Server aziendale e resa disponibile con l'accesso ad un'area riservata tramite protocollo HTTP (da un qualsiasi browser Web è quindi possibile consultarla)[1].

[1]Il file di testo contenente la documentazione per l'operazione in oggetto si chiama "IMPORT_PARTECIPAZIONI_ECM.RTF" e descrive le convenzioni utilizzate per l'importazione delle partecipazioni.

3.3 Caratteristiche della tabella di destinazione

La tabella di destinazione dei dati in entrata si chiama *PARTECIPAZIONE_ECM* e contiene i seguenti campi:

Nome Campo	Proprietà	Descrizione
ID	decimal(22), PK	chiave primaria, è un numero progressivo generato da una SEQUENCE Oracle.
CORSO_ECM	varchar2(10)	sigla del corso.
NUMERO_CREDITI	decimal(22), NN	numero dei crediti del corso.
DATA_INIZIO	date NN	data di inizio del corso.
DATA_FINE	date NN	data del termine del corso.
CODICE	varchar2(50), NN	codice del corso, è un numero standard dato dal Ministero.
CODICE_EDIZIONE	varchar2(50), NN	numero corrispondente all'edizione del corso.
CODICE_ORGANIZZATORE	varchar2(10), NN	codice dell'ente organizzatore del corso.
PROFESSIONISTA_CODICE_FISCALE	varchar2(16), NN FK	codice fiscale del partecipante al corso, chiave esterna su *PROFESSIONISTA*.
DISCIPLINA_ECM_ID	decimal(22), NN	codice della disciplina, chiave esterna su *DISCIPLINA_ECM*.
PROFESSIONE_ID	decimal(22), NN	professione con cui si è iscritto il professionista, chiave esterna su *PROFESSIONE*.
ACCREDITATORE_ID	decimal(22), NN	identificativo dell'accreditatore del corso.
TIPO_EVENTO_VALORE	char(1), NN FK	tipo di evento (E, P), chiave esterna su *TIPO_EVENTO*.
TIPO_CREDITI_VALORE	char(1), NN FK	tipo dei crediti assegnati (P, D, T, R), chiave esterna su *TIPO_CREDITI*.
TIPO_FORMAZIONE_VALORE	char(3), NN FK	tipo della formazione (di default FOR).
FORZATA	char(1)	attributo che indica l'eventuale forzatura nell'immissione di un record.

[2]Notazione sulla tabella: FK (Foreign Key) indica una chiave esterna, NN (Not Null) un campo che non deve mai essere vuoto e PK (Primary Key) la chiave primaria.

3.4 Caratteristiche dei dati ricevuti

Alcune importanti osservazioni vanno fatte in merito ai dati ricevuti.

Riguardo ai corsi, nei singoli record sorgenti vengono salvati solamente i codici dei corsi a cui i professionisti hanno partecipato, senza ulteriori informazioni (come data di inizio e della fine del corso, organizzatore e numero dei crediti assegnati).

Negli anni precedenti veniva inviato all'azienda, successivamente ai file delle partecipazioni, un file in formato Microsoft Excel© (.XLS) contenente tutte le informazioni sui singoli corsi. Per questo motivo nella tabella *PARTECIPAZIONE_ECM* erano previsti i campi destinati alle informazioni contenute nel file Excel. Da quest'anno però tale file non viene più inviato e quindi è disponibile solamente il codice del corso.

Riguardo ai dati, invece, va detto che i file contenenti le partecipazioni ricevuti hanno tutti una struttura comune. Ogni anno l'azienda riceve infatti un certo numero di file dal Ministero della Salute contenenti tutte le partecipazioni in formato standard ad Eventi e Progetti Formativi (PFA) in formato .MDB (Database Access©), che contengono le seguenti informazioni:

Nome Campo	**Descrizione**
CODICE_EVENTO	codice del corso come numero dato dal Ministero
EDIZIONE	edizione del corso
CODICE_FISCALE	codice fiscale del partecipante al corso
RUOLO	ruolo del professionista iscritto (partecipante, ...)

COGNOME	cognome del professionista
NOME	nome del professionista
PROFESSIONE	professione con cui il professionista si è iscritto all'evento o al progetto formativo
DISCIPLINA	disciplina del corso
altro	Campi non necessari per l'import dati

In questo caso i dati importati in *FNOMCEO* saranno tutti relativi alle partecipazioni ai corsi avvenute durante l'anno 2008.

3.5 Preparazione all'importazione

3.5.1 Considerazioni pre-operative

Codifica dei dati

Quando si ha a che fare con dati di questo tipo, siccome non si può mai avere la certezza della loro natura e del modo in cui sono stati raccolti, bisogna cercare di non commettere errori che si possono classificare come banali. Per questo motivo, prima di procedere, è opportuno svolgere un insieme di operazioni sui dati:

- Eliminare gli spazi iniziali e finali dalle stringhe.
- Convertire tutte le stringhe in caratteri maiuscoli (in modo tale da riuscire a confrontare stringhe con le stesse informazioni).
- Verificare il tipo di codifica dei caratteri (Unicode, ISO-646, ISO 8859-1, ...) per poi, successivamente, riuscire ad aggirare eventuali problemi legati alla presenza di caratteri speciali.

- Accertarsi che i campi siano effettivamente popolati da numeri dove ci si aspetta numeri e da stringhe dove ci si aspetta stringhe, ossia eseguire un controllo sui datatype.

3.5.2 Azioni sui dati

Campi non popolati e obbligatori

Prima di mettere mano ai dati sono stati presi in considerazione i campi obbligatori della tabella di destinazione (*PARTECIPAZIONE_ECM*) e, relativamente a campi non popolati presenti tra i dati ricevuti, sono state intraprese le seguenti azioni:

Nome Campo	Condizione	Azione
RUOLO	Non popolato oppure con valori diversi da P, D, T, R	'P' (partecipante)
PROFESSIONE	non popolato	99 (tutte le professioni)
DISCIPLINA	non popolato	0 (area interdisciplinare)

Campi non presenti

Nei file inviati dal Ministero non sono presenti le informazioni riguardanti i seguenti campi presenti nella tabella di destinazione: *CORSO_ECM*, *ACCREDITATORE_ID*, *TIPO_FORMAZIONE_VALORE*. Per ognuno di essi vengono elencate le azioni intraprese:

1. CORSO: viene valorizzato a `[#anno_partecipazione]<_PFA (nel caso in cui si tratti di un Progetto Formativo)>` (dove il contenuto delle parentesi quadre è obbligatorio e quello tra i simboli minore e maggiore opzionale);

2. ACCREDITATORE_ID: viene impostato a 1 (Ministero);

3. TIPO_FORMAZIONE_VALORE: si presuppone si tratti di Formazione Residenziale, quindi viene inizializzato a 'FOR';

4. TIPO_EVENTO_VALORE: valorizzato a 'E' se il file riguarda gli Eventi, 'P' se riguarda i Progetti Formativi.

Codice fiscale

Il controllo del codice fiscale va effettuato manualmente controllando i dati e incrociando i nomi e i cognomi dei professionisti. Se si trovano codici fiscali sufficientemente simili nella tabella *PROFESSIONISTA* dove nome e cognome corrispondono ai dati di import, allora il codice fiscale viene corretto.

È utile in questa fase aggiungere un accorgimento: i nomi e cognomi della tabella *PROFESSIONISTA* sono salvati completi, ossia in caso di secondo nome il campo *NOME* conterrà il nome completo; nella tabella di importazione questa completezza non è detto che sia presente.

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.

3.5.3 Identificazione del tipo di errore

Dal momento in cui è possibile riscontrare qualche errore che potrebbe bloccare l'operazione di importazione, è utile "mappare" in anticipo i record che darebbero problemi durante il trasferimento dei dati. La convenzione adottata è quella di aggiungere una colonna alla tabella di import (in *COGEAPS_TMP*) di nome *ERRORE* che conterrà i codici degli errori così come sono descritti nella documentazione. Di seguito vengono elencate le anomalie e viene specificata l'azione intrapresa (ossia il valore settato nel campo *ERRORE*).

Anomalia	**ERRORE**
Il codice professionista non è corretto alla luce delle elaborazioni fatte sui dati.	2
Il nome, il cognome o entrambi sono nulli, ma il codice fiscale c'è.	4
Presenza di partecipazioni uguali in seguito a un raggruppamento per codice dell'evento, edizione, codice fiscale, ruolo, professione e disciplina.	3
Il codice fiscale, corretto (lunghezza di 16 caratteri), non è presente in *PROFESSIONISTA*.	'TMP'

3.6 Importazione in PARTECIPAZIONE_ECM

3.6.1 Dati temporanei

Se il codice fiscale del professionista non viene trovato tra le anagrafiche attualmente presenti (ossia se l'errore risulta uguale a 'TMP'), non verrà creato un nuovo professionista a partire dalla partecipazione, ma il record di partecipazione verrà memorizzato nella tabella temporanea *PARTECIPAZIONE_ECM_TMP*. In questo modo, successiva-

mente, se verranno inviati aggiornamenti sull'anagrafica dei professionisti da parte della Federazione, si proverà nuovamente ad importare la partecipazione.

3.6.2 Dati completi

Se il codice di errore sui dati elaborati e puliti risulta nullo, si può procedere con l'importazione nella tabella di destinazione *PARTECIPAZIONE_ECM* presente nel database Oracle.

3.6.3 Eliminazione dati

I dati che risultano con codice di errore diverso dal valore nullo o da 'TMP' vengono segnalati al Ministero e, qualora non venissero re-inviati corretti dallo stesso ente entro un periodo temporale stabilito (di solito semestrale), successivamente eliminati dal sistema. Una volta terminata l'importazione nel DB, infine, i dati corretti (codice di errore nullo) presenti nella tabella di import vengono rimossi, dopo un ulteriore controllo che tutti siano stati memorizzati in *PARTECIPAZIONI_ECM* su *FNOMCEO*.

3.7 Esempio di codice

In seguito viene riportato il codice SQL-Oracle eseguito per portare a termine l'importazione dei dati. Viene riportato il codice completo riguardante l'elaborazione di un file relativamente pulito contenente parte-

cipazioni di professionisti a Progetti Formativi. La tabella di riferimento si chiama *NEW_PFA* allocata nel DB *COGEAPS_TMP*. Successivamente si sono svolte elaborazioni più corpose su file relativi alle partecipazioni a Eventi ma, per motivi di lunghezza del codice e per il fatto che tali elaborazioni risultano essere molto simili a quelle esaminate nel presente lavoro, esse non vengono 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');

--campo 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');
```

Appendice A

Oracle e MySQL

A.1 Introduzione

MySQL è il più diffuso tra i DBMS relazionali Open Source, Oracle invece è il più diffuso tra gli RDBMS (Relational Data Base Management System) commerciali; per questo motivo, dato che non è così improbabile dover passare dall'uno all'altro, può essere particolarmente interessante focalizzare l'attenzione sulle differenze che ci sono tra questi due DBMS (o RDBMS dato che si basano entrambi sul modello logico relazionale). Sia per Oracle che per MySQL la documentazione presente in rete è sterminata, anche se si consiglia di fare riferimento ai siti ufficiali[1]. Si tenga presente che, aspetti tecnici a parte, la differenza principale sta nel fatto che MySQL è Open Source e quindi gratuito, mentre Oracle no.

[1] Siti ufficiali: MySQL (`http://www.mysql.com`), Oracle (`http://www.oracle.com`)

A.2 Elementi comuni

Oracle e MYSQL sono due ottimi database relazionali basati sul linguaggio SQL. SQL è un linguaggio case-insensitive, il che significa che non riconosce la differenza tra le lettere maiuscole e minuscole. Dato che il linguaggio su cui si basano è lo stesso, in realtà sono più gli elementi in comune che le differenze.

L'SQL è identico al 95% su tutti e due i sistemi; entrambi fanno riferimento alla documentazione *ANSI* (American National Standard Institute) *standard SQL: 2003* anche se sia MySQL che Oracle hanno estensioni personalizzabili. I tipi di applicazioni supportate e supportabili, le piattaforme disponibili su cui essi girano e il linguaggio con cui sono stati scritti (il C) sono gli stessi.

Se fino a qualche anno fa aveva senso confrontare i limiti che avevano i diversi DB (massimo numero di colonne, dimensione massima di un DB, di un BLOB, di colonne in un indice, il massimo e il minimo numero rappresentabili, il numero di cifre significative di un numero, . . .), con le attuali versioni non lo è più in quanto tali limiti sono talmente ampi su entrambi i DBMS da non risultare rilevanti a fini pratici. Anche l'assenza del supporto per transazioni (sviluppato con Transact-MySQL), la gestione delle subquery, la mancanza di viste, spesso riportati come difetti di MySQL, sono relativi a versioni particolarmente vecchie del prodotto.

A.3 Tipi di dati (datatype)

I Datatype supportati da entrambi i DBMS sono molteplici e vi sono parecchie differenze, anche grazie alla presenza in MySQL di Engine con funzionalità diverse e specifiche. Sebbene MySQL consenta di risparmiare spazio rispetto a Oracle proprio sull'ottimizzazione dei tipi di dati, Oracle integra un numero più ampio di datatype. In MySQL infatti, ad esempio, le strutture TYPE, dimension, bitmap index, global partitional index e materialized view non hanno nessuna equivalenza.

Per i datatype numerici le differenze principali stanno nella scala di precisione. MySQL risulta più selettivo rispetto a Oracle e consente di scegliere un datatype che richiede meno spazio. In MySQL non esistono le SEQUENCE, sostituite dalla definizione di una colonna come SERIAL che corrisponde ad una dichiarazione di BIGINT UNSIGNED NOT NULL AUTO_INCREMENT.

Per i datatype stringa un'importante differenza è che MySQL consente di specificare, colonna per colonna, il "character set" e la "collation", mentre Oracle no. Con i campi CHAR, VARCHAR e TEXT su Engine MyISAM è possibile definire indici FULLTEXT (ossia indici che consentono la ricerca all'interno dell'intero campo testuale con la clausola MATCH).

Naturalmente entrambi i DBMS consentono di indicare attributi particolari sulle colonne di una relazione (eg. NOT NULL) con delle differenze minime. Oracle dispone di più capacità di definizione di vincoli (CONSTRAINT) rispetto a MySQL, che consente la dichiarazione di

chiavi esterne (FOREIGN KEY) fornendo un ovvio supporto completo solamente con l'utilizzo dell'Engine InnoDB. In un certo senso si può dire che MySQL si avvicina molto ad Oracle se si utilizza l'Engine InnoDB comportandosi in maniera molto simile.

In seguito vengono elencati i principali datatype di MySQL con le strutture equivalenti di Oracle. Ovviamente l'equivalenza può essere letta solamente da un lato, diversi infatti risultano i datatype equivalenti in MySQL se si parte da Oracle.

MySQL Data Type	**Oracle Data Type**
BIGINT	NUMBER(19, 0)
BIT	RAW
BLOB	BLOB, RAW
CHAR	CHAR
DATE	DATE
DATETIME	DATE
DECIMAL	FLOAT (24)
DOUBLE	FLOAT (24)
DOUBLE PRECISION	FLOAT (24)
ENUM	VARCHAR2
FLOAT	FLOAT
INT	NUMBER(10, 0)
INTEGER	NUMBER(10, 0)
LONGBLOB	BLOB, RAW
LONGTEXT	CLOB, RAW
MEDIUMBLOB	BLOB, RAW
MEDIUMINT	NUMBER(7, 0)
MEDIUMTEXT	CLOB, RAW
NUMERIC	NUMBER
REAL	FLOAT (24)
SET	VARCHAR2

SMALLINT	NUMBER(5, 0)
TEXT	VARCHAR2, CLOB
TIME	DATE
TIMESTAMP	DATE
TINYBLOB	RAW
TINYINT	NUMBER(3, 0)
TINYTEXT	VARCHAR2
VARCHAR	VARCHAR2, CLOB
YEAR	NUMBER

A.4 Prestazioni, ottimizzazione e configurazione

Le differenze tra le prestazioni dei due RDBMS costituiscono il punto di partenza per la progettazione di un sistema informatico di archiviazione. MySQL è molto più leggero rispetto ad Oracle nelle connessioni e nei semplici accessi ai dati. Questo fatto lo rende decisamente più efficiente in molti scenari applicativi (soprattutto nelle applicazioni Web). Oracle, invece, per la gestione delle transazioni, con un numero elevato di utenti e accessi concorrenti, si comporta molto meglio di MySQL.

Le differenze sull'ottimizzazione sono parecchie. Sebbene MySQL non abbia tutte le strutture dati che Orale prevede e supporta, Oracle non ha un equivalente degli Storage Engine che offre una certa flessibilità e permette di scegliere la struttura ottimale per ogni oggetto presente nel DB. MySQL inoltre non ha una LIBRARY CACHE, ma possiede una QUERY CACHE molto efficace; nel caso in cui infatti una query venga eseguita ripetutamente, il DB non viene interrogato ma si arrangia il DBMS a fornire i risultati alla richiesta. Oracle non possiede nessuna QUERY CACHE.

La configurazione di MySQL è più semplice di quella di Oracle ma, se si sfruttano in modo pesante le possibilità date dagli Engine diversi, può richiedere parecchia attenzione ed esperienza.

A.5 Programmi di supporto

Oracle fornisce un'ampia collezione di tool per le più comuni attività di amministrazione dei dati. Anche MySQL fornisce strumenti simili. Per rendere pratico il confronto la lista seguente riporta i più conosciuti programmi Oracle e come ottenere funzionalità simili con MySQL:

- SQLPLUS: interprete SQL ben sostituito dall'analogo interprete mysql di MySQL.
- EXP/IMP: tool per esportare ed importare dati con Oracle. L'analogo MySQL è mysqldump che è ricco di funzionalità. Una caratteristica molto utile di mysqldump è che esso genera un file di dump in SQL, risultando quindi molto semplice da utilizzare per conversioni/migrazioni/...
- SQL*LOADER: programma di caricamento massivo di dati di Oracle. MySQL fornisce funzionalità simili con lo statement SQL LOAD DATA INFILE o con il tool mysqlimport. Oracle non implementa un comando SQL per importare direttamente i dati.
- Enterprise Manager Web Interface di amministrazione del DB. Distribuito come Open Sorce vi è il diffusissimo phpMyAdmin che

consente l'amministrazione dei DB MySQL in modo molto semplice e veloce.

A.6 Sicurezza

A.6.1 Una panoramica sulla sicurezza

Oracle e MySQL hanno entrambi un sistema interno per provvedere alla sicurezza dei dati nel DB. MySQL usa una serie di tabelle per tenere traccia degli utenti e dei privilegi che possono avere e utilizza queste tabelle per controllare le connessioni al DB.

Differentemente da Oracle, che come molti altri DB utilizza solo il nome utente e la password per autenticare l'accesso, MySQL utilizza una clausola addizionale chiamata "location parameter". Questo parametro è solitamente composto dal nome dell'Host, dall'indirizzo IP o da una Wildcard. Con questo parametro aggiuntivo MySQL può ulteriormente restringere un accesso di utenti al DB e vincolarlo solamente ad alcuni Host o domini.

A.6.2 Privilegi

I privilegi supportati da MySQL si possono dividere in due tipi: amministrativi e privilegi per singoli oggetti. I primi sono globali e riguardano anche il lato server di MySQL, i secondi invece riguardano esclusivamente gli oggetti interni al database come tabelle, colonne, indici e pro-

cedure di memorizzazione. I privilegi in MySQL sono di tipo gerarchico e precisamente, in ordine:

Global (accesso globale, super-amministratore): permette di avere tutti i privilegi consentiti dal sistema.

Per-host basis : permette di avere solamente privilegi che partono dalla connessione al DBMS.

Database-level : permette privilegi esclusivamente interni al database, senza poter modificare utenti o connessioni.

Table-specific (singole tabelle): consente l'accesso o la modifica solamente a singole tabelle dello schema.

Column-specific (su singole colonne di singole tabelle): consente l'accesso esclusivamente a colonne di determinate tabelle.

Ciascun livello ha una tabella corrispondente nel database.

In Oracle, differentemente, non sussiste il concetto di ruolo gerarchico come in MySQL perciò, per garantire ad un gruppo di utenti gli stessi privilegi, deve essere dichiarato separatamente per ciascun utente ogni tipo di privilegio. Questo rende una struttura più personalizzabile ma meno banale da gestire.

A.7 Architettura

MySQL è un processo multithreaded che fa tutto: ascolta un socket, lancia i thread necessari alle sessioni utente, effettua il parsing e l'ottimizzazione degli statement SQL. In MySQL, per ogni sessione utente viene

lanciato un processo diverso (come in Oracle) che si occupa del parsing dell'SQL e dell'accesso ai dati. In MySQL non esistono processi di background distinti come in Oracle. Entrambi i RDBMS, per funzionare in maniera ottimale ed efficiente, hanno bisgono di una buona quantità di memoria disponibile.

Tipicamente con MySQL applicazioni differenti utilizzano database differenti (che corrispondono agli schemi di Oracle); questo perché in MySQL non esiste il concetto di istanza e non c'è la presenza di un listener separato. Ogni database in MySQL corrisponde ad una directory nel FILE SYSTEM riservata ai dati. Alla creazione di ogni tabella viene creato un file FRM all'interno della directory contenente la descrizione e, a seconda degli Engine, eventuali altri file che mantengono dati e indici.

Con Oracle la parte di accesso ai dati è una componente fondamentale del motore, mentre con MySQL è separata rendendo quindi possibile sfruttare le caratteristiche dei diversi Engine disponibili. La tabella seguente riporta le caratteristiche degli Engine più interessanti.

MyISAM	Engine di default. Molto veloce e leggero. Consente la creazione di indici FULLTEXT. Non ha il supporto delle transazioni.
InnoDB	Consente la gestione completa delle trasazioni; è adatto a tutte le applicazioni che richiedono una forte trasazionalità e sicurezza dei dati.

Memory	Molto utile per dati temporanei ed elaborazioni in sequenza, ha la particolaritò rispetto agli altri motori che memorizzano i dati su memoria secondaria che li mantiene in memoria principale.
Archive	Viene utilizzato per la gestione di grosse moli di dati o per la storicizzazione: i dati sono mantenuti in forma compressa sul file system.
CSV	Engine molto semplice ma anche comodo per la conversione dei dati. I dati sono mantenuti su file in formato testo (CSV: Comma Separated Values).
Federated	Consente di accedere a tabelle presenti su un sistema remoto. Si tratta di un accesso simile a quello presente in Oracle con i database link. I dati in questo caso sono mantenuti sul sistema remoto.

A.8 Aprile 2009: Oracle acquista Sun

Sun Microsystem, azienda che si occupa della produzione di software come Java, ha acquistato durante il 2008 il software DBMS MySQL per 1 miliardo di dollari. Nel mese di Aprile 2009 la stessa Sun è stata acquistata da Oracle per 7,4 miliardi di dollari.

Si è subito pensato al fatto che Oracle con questo acquisto potesse in qualche modo “uccidere” il software OpenSource MySQL, principale concorrente nel mondo dei DBMS. Ciò, alla luce di alcune osservazioni,

è molto improbabile.

MySQL registra infatti 70mila download giornalieri e costituisce un importantissimo strumento per i giovani sviluppatori, a differenza di Oracle che lo è quasi esclusivamente per le aziende. Ulteriormente c'è da dire che lo sviluppo di questi due DBMS è storicamente molto differente: MySQL infatti è stato sviluppato in maniera molto più selvaggia rispetto a Oracle che invece è cresciuto gradualmente concentrando le sue potenzialità sulle esigenze aziendali.

Le prospettive future per MySQL sono presumibilmente due, dal momento in cui sembra che Oracle non possa abbandonare MySQL per i motivi precedentemente descritti:

1. Oracle investirà nello sviluppo di MySQL, trasferendo alcune sue conoscenze;

2. Oracle implementerà un sistema misto.

La prima sembra ad oggi la più probabile.

A.9 Oracle o MySQL?

Alla luce di quanto descritto e analizzato, non è possibile privilegiare una scelta dell'uno o dell'altro DBMS senza conoscere prima le esigenze ed i problemi che si intendono risolvere. Oracle infatti è ottimizzato per un sistema di grosse dimensioni che non richieda elaborazioni continue di query, mentre MySQL funziona meglio rispetto al suo antago-

nista nel caso in cui si tratti di archivi non troppo complessi soggetti a frequenti richieste.

Quindi la risposta più corretta alla domanda è: dipende

Bibliografia

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

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

[c] Paulraj Ponniah (2001), *Data Warehousing*, Wiley, New York.

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

[e] Ben-Gan Itzik and Moreau Tom (2006), *Transact SQL. Guida pratica*, Mondadori Informatica, Milano.

[f] Giuseppe Callegarin (1996), *Nuovo corso di informatica. Basi di dati e sistemi informativi. Per le scuole superiori (3)*, CEDAM, Padova.

[g] Ramez A. Elmasri and Shamkant B. Navathe (2004) *Sistemi di basi di dati*, Pearson Education, Milano.

Siti consultati

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

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

[3] http://squirrel-sql.sourceforge.net/

[4] http://manuals.sybase.com/onlinebooks/

[5] http://www.1keydata.com/datawarehousing/

[6] http://www.kimballgroup.com/

[7] http://www.forbes.com/

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

[9] http://www.mysql.com/

[10] http://www.levysoft.it/

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