Laboratorio di Basi di dati 1
II esercitazione 18 marzo 2003

Utilizzando l'Enterprise Manager e il Query Analyzer di SQL Server e *tenendo presenti i suggerimenti che seguono il testo dell'esercitazione*, svolgere le seguenti operazioni:

1. - **Per i gruppi che hanno creato, durante la precedente esercitazione, il database per la gestione dei corsi:**
     (a) selezionare, se non fosse già selezionato, il database per la gestione dei corsi.
     (b) dal Query Analyzer di SQL server, scrivere il comando SQL per cancellare dal database le tabelle `Corsi` e `Docenti` precedentemente create. Verificare che le tabelle non siano più nel database;
   - **Per i gruppi che non hanno creato, durante la precedente esercitazione, il database per la gestione dei corsi:**
     (a) dall'Enterprise Manager di SQL Server o dal Query Analyzer di SQL Server, creare un nuovo database con nome `GestioneCorsi`N, dove N va sostituito con il numero del gruppo di appartenenza (se il vostro gruppo è il numero 5, il nome del Database da creare è `GestioneCorsi5`);
     (b) lanciare SQL Query Analyzer e selezionare, se non fosse già selezionato, il database appena creato.

2. Data la relazione con schema:

   `Professori(`<u>`id`</u>`, Cognome, Nome)`

   che memorizza le informazioni relative ai professori dell'università di Genova, dove:

   - <u>`id`</u> rappresenta l'identificativo, numerico di 5 cifre, univoco, di ogni professore, ed è chiave primaria della relazione;
   - `Cognome` e `Nome` rappresentano il cognome e il nome di ogni professore e devono essere obbligatoriamente presenti;

   scrivere il comando SQL che crei la tabella per questa relazione (scriverlo prima su carta, per verificare anticipatamente di aver impostato tutti vincoli necessari), utilizzando i tipi di dato `decimal` e `varchar`, ragionando sulla dimensione, in numero di cifre e caratteri, da utilizzare per la definizione di ciascun campo.

   Verificare che la tabella sia effettivamente presente nel database `GestioneCorsi`N e verificare che abbia lo schema che si voleva ottenere. Se lo schema non fosse corretto, utilizzare SQL per cancellare la tabella e ricrearla.

   Salvare il comando SQL eseguito (o provato ad eseguire) per la creazione della tabella `Professori`.

3. Di seguito:

   (a) Scrivere il comando SQL che permetta di modificare la tabella `Professori`, aggiungendo alla tabella le colonne `Stipendio`, che rappresenta lo stipendio annuale in Euro di ogni professore (utilizzare 8 cifre, due delle quali decimali), con valore predefinito 15000 Euro, e la colonna `InCongedo`, a valori booleani, che specifica se un professore è in congedo, con valore predefinito false. Ovviamente lo stipendio non può assumere valori negativi.

   (b) Dall'Enterprise Manager, inserire due tuple nella tabella, inserendo dei valori di stipendio con il numero massimo di cifre non decimali. Provare a violare i vincoli definiti per verificarne la correttezza.

   (c) Scrivere il comando SQL che modifichi la tabella `Professori`, modificando la colonna `Stipendio`, in modo che possa contenere dati con 7 cifre non decimali e due decimali. Osservare il messaggio dato da SQL Server.

   (d) Scrivere il comando SQL che modifichi la tabella `Professori`, modificando la colonna `Stipendio`, in modo che possa contenere dati con 5 cifre non decimali e due decimali. Osservare il messaggio dato da SQL Server. Il comando va a buon fine? Perché?

   (e) Inserire, dall'Enterprise Manager, 10 tuple per la tabella `Professori`, inserendo dei valori per la colonna `InCongedo`.

(f) Scrivere il comando SQL che modifichi la tabella `Professori`, cancellando la colonna `InCongedo`. Il comando va a buon fine?

(g) Dall'Enterprise Manager modificare la tabella `Professori` e togliere il vincolo di Default sul campo `InCongedo`, quindi rieseguire il comando SQL che modifichi la tabella `Professori`, cancellando la colonna `InCongedo`. Il comando va a buon fine?

(h) Salvare la sequenza di comandi SQL eseguita (con i tentativi falliti) per la modifica della tabella `Professori`.

4. Di seguito:

   (a) sia data la relazione con schema:
   `Corsi(id, CorsoDiLaurea, Denominazione, Professore, Attivato)`
   che mantiene le informazioni sui corsi dei vari corsi di laurea dell'università degli Studi di Genova, dove:
   - `id` rappresenta l'identificativo alfanumerico di 10 caratteri, univoco, di ogni corso, ed è chiave primaria della relazione;
   - `CorsoDiLaurea` rappresenta il corso di laurea (es. ingegneria, informatica) in cui il corso è tenuto;
   - `Denominazione` è il nome del corso (es. Basi di Dati 1);
   - `Professore` è l'identificativo del professore titolare del corso, ed è chiave esterna della relazione;
   - `Attivato` è un attributo booleano, con valore predefinito false, che indica se il corso è attivato nell'anno accademico corrente oppure no.

   La denominazione del corso e il corso di laurea sono sempre indicate e la coppia Corso di laurea e Denominaizone è unica all'interno della relazione.

   L'integrità referenziale rispetto alla chiave esterna `Professore` deve essere gestita in modo che
   - se viene modificato l'identificativo di professore per una tupla di `Professori`, la modifica sia propagata alla tabella `Corsi`;
   - non sia permesso di cancellare un Professore se esistono corsi associati a quel professore.

   Scrivere il comando SQL che crei la tabella per questa relazione, tenendo presenti i vincoli descritti e la descrizione data per gli attributi. Una volta scritto il comando corretto, salvarlo.

   (b) Inserire 15 tuple nella tabella `Corsi`, relative a corsi di Informatica, Ingegneria Elettronica, Ingegneria Meccanica, Fisica, Matematica. Tenere presenti le tuple di `Professori` inserite al punto precedente. Inserire anche corsi non attivati e senza professore (si gestisca manualmente il fatto che un corso attivato deve avere un professore titolare del corso).

   (c) Effettuare la modifica e la cancellazione di una tupla dalla tabella `Professori`, così da verificare la correttezza dei vincoli di integrità.

5. Data la relazione con schema:

   `Studenti(Matricola, Cognome, Nome, CorsoDiLaurea, Iscrizione, Relatore)`

   che mantiene le informazioni relative agli studenti dell'università degli Studi di Genova dove:

   - `Matricola` rappresenta la matricola di ogni studente, alfanumerica, ed è chiave primaria della relazione;
   - `Cognome` e `Nome` rappresentano il cognome e il nome di ogni studente;
   - `CorsoDiLaurea` rappresenta il corso di laurea (es. ingegneria, informatica) a cui lo studente è iscritto;
   - `Iscrizione` rappresenta l'anno accademico (es. 2000/2001) della prima iscrizione dello studente al corso di laurea;
   - `Relatore` è il professore (si ipotizzi che sia uno solo) che segue il lavoro di tesi dello studente e che viene assegnato allo studente durante il corso dei suoi studi.

   scrivere il comando SQL che crei la tabella per questa relazione, ragionando sui vincoli da utilizzare per mantenere la consistenza dei dati memorizzati e sui tipi di dati più adatti per implementare gli attributi. Si noti che `Relatore` è chiave esterna per la relazione; la semantica dell'attributo richiede che una modifica ai dati del relatore abbia effetto anche sui dati degli studenti che segue durante la tesi di laurea, mentre che non sia possibile cancellare i dati di un professore se esistono studenti di cui è stato relatore.

   Inserire una decina di tuple nella tabella `Studenti`, verificando la correttezza dei vincoli impostati.

6. Scrivere in SQL ed eseguire utilizzando il Query Analyzer di SQL Server (osservare i risultati ottenuti, di volta in volta, riportati nella scheda Grids), i comandi che permettano di ottenere:

   (a) tutti i dati di tutti gli studenti dell'università di Genova;

   (b) i corsi attivati durante l'anno accademico in corso per tutti corsi di laurea dell'università di Genova;

   (c) cognome, nome e anno di iscrizione degli studenti di informatica dell'università di Genova, in ordine alfabetico;

   (d) l'elenco degli studenti dei corsi di laurea di informatica che non hanno assegnato alcun relatore;

   (e) i nomi, in ordine alfabetico inverso e senza duplicati, dei corsi tenuti in tutti i corsi di laurea dell'università di Genova;

   (f) la matricola, il cognome e il nome degli studenti dell'università di Genova iscritti ai corsi di laurea di informatica e di matematica;

   (g) la matricola e il corso di laurea degli studenti iscritti a fisica nell'anno accademico 1998/1999 o che hanno il cognome che comincia per 'P';

   (h) cognome e nome degli studenti di informatica dell'università di Genova iscritti dopo l'anno accademico 1999/2000, il cui nome sia compreso, in senso alfabetico, fra 'Beatrice' e 'Giovanni';

   (i) i professori relatori di studenti iscritti ai corsi di laurea di ingegneria che guadagnano più di 10000 Euro e meno di 15000 Euro l'anno (prima senza utilizzare il `JOIN` e poi utilizzandolo);

   (j) l'elenco, in ordine alfabetico, dei professori che insegnano in corsi (indicare quali) attivati presso i corsi di laurea di ingegneria elettronica e/o informatica dell'università di Genova (prima senza utilizzare il `JOIN` e poi utilizzandolo);

   (k) l'elenco dei corsi, in ordine in base al codice identificativo, tenuti da professori che sono relatori di tesi di informatica (prima senza utilizzare il `JOIN` e poi utilizzandolo);

   Salvare la sequenza di comandi SQL eseguita.