Basi di Dati

Allievi Informatici e Telecomunicazionisti

Proff. Stefano Ceri e Maristella Matera

PRIMA PROVA IN ITINERE – 17 NOVEMBRE 2003

N.B. La soluzione proposta riflette un'interpretazione NON UNIVOCA delle specifiche

UTENTE (IdUsr, Nome, Password, Categoria)
PAGINA (URL, Titolo, DimInByte, Tipo, Proprietario)
ACCESSO (IdUsr, Data, Ora, UrlPag, PaginaProvenienza)
COLLEGAMENTO (PagFrom, PagTo)

Lo schema si riferisce agli accessi alle pagine di un sito Web da parte di utenti registrati. Memorizza inoltre le informazioni relative al collegamento tra le pagine (attraverso i link) e per ogni accesso registra la pagina di provenienza. Ogni pagina ha un utente – rappresentato tramite il suo identificatore - come "proprietario". Per la pagina iniziale di ogni navigazione, la pagina di provenienza è posta a **Null**.

A. DDL (2 punti)

Scrivere i comandi SQL per creare le tabelle PAGINA e COLLEGAMENTO, effettuando opportune e ragionevoli ipotesi sui domini, sui vincoli e sulle reazioni ai cambiamenti.

```
create table PAGINA (
        URL             varchar(255) primary key,
        Titolo          varchar(255),
        DimInByte       integer > 0,
        Tipo            varchar(20) default "personal",
        Proprietario    varchar(30) references UTENTE(IdUsr)
                                    on update cascade
                                    on delete set null
)
```

Il valore null a seguito della cancellazione permette di identificare le pagine che nessuno più mantiene (o di cui comunque non si hanno informazioni certe sul proprietario)

```
create table COLLEGAMENTO (
        PagFrom   varchar(255) references PAGINA(URL)
                               on update cascade
                               on delete cascade,
        PagTo     varchar(255) references PAGINA(URL)
                               on update cascade
                               on delete no action,
        primary key (PagFrom, PagTo)
)
```

In caso di aggiornamento del link, la scelta è di aggiornare la tabella collegamento di conseguenza. In caso di cancellazione di una delle pagine, la politica è differenziata:

- *il cascade su PageFrom garantisce che la tabella COLLEGAMENTO venga automaticamente aggiornata (e mantenuta "minima")*

- *il no action su PageTo garantisce che non si possano cancellare pagine quando il sito ancora contiene collegamenti attivi a quelle pagine – garanzia di integrità.*

Per completezza ecco la definizione delle tabelle NON RICHIESTE:

```
create table UTENTE (
        IdUsr           varchar(30) primary key,
        Nome            varchar(255),
        Password        varchar(20),
        Categoria       varchar(20) default "guest"
)

create table ACCESSO (
        IdUsr varchar(30) references UTENTE(idusr),
        Data date,
        Ora time,
        UrlPag varchar(255) references PAGINA(URL) on update cascade on delete cascade,
        PaginaProvenienza varchar(255),
        primary key (IdUsr, Data, Ora, UrlPag)
)
```

Una integrità referenziale tra la coppia (UrlPag, PaginaProvenienza) e COLLEGAMENTO(PagTo, PagFrom) potrebbe avere un senso (la navigazione non può che avvenire seguendo link validi), ma non è applicabile a causa della scelta di usare il Null per caratterizzare le pagine iniziali di ogni percorso di navigazione. Infatti PagFrom è parte della chiave di Collegamento, e non può ovviamente assumere valori nulli.

B. LINGUAGGI FORMALI (7 punti)

1. Esprimere in Algebra Relazionale ottimizzata, Calcolo Relazionale e Datalog la seguente interrogazione **(3 punti)**:
   Trovare le pagine "flop", cioè quelle che sono state visitate soltanto dai loro proprietari.

Π_{URL} (ACCESSO – (ACCESSO $\triangleright\!<_{UrlPag=URL \wedge IdUsr<>Proprietario}$ PAGINA))

Da ottimizzare col push delle proiezioni…

$$\{ t \mid \exists t_A \in ACCESSO$$
$$(t[UrlPag] = t_A[UrlPag] \wedge$$
$$\neg (\exists t_{A2} \in ACCESSO, \exists t_P \in PAGINA$$
$$t_{A2}[UrlPag] = t_A[UrlPag] \wedge$$
$$t_{A2}[UrlPag] = t_P[UrlPag] \wedge$$
$$t_{A2}[IdUsr] <> t_P[Proprietario])$$
$$)$$
$$\}$$

```
AccDaNonProprietario(Url) :-    ACCESSO(IdU, _, _, Url, _),
                                PAGINA(Url, _, _, _, IdP), IdP <> IdU
AccSoloProprietario(UrlFlop) :- ACCESSO(_, _, _, UrlFlop, _),
                                ¬ AccDaNonProprietario(UrlFlop)
? – AccedutaSoloProprietario(X)
```

2. Esprimere in algebra relazionale ottimizzata la seguente interrogazione (**1.5 punti**):
Trovare le pagine per cui l'ultimo utente che le ha consultate è il loro proprietario.
[Si presti attenzione al fatto che l'istante di accesso è contraddistinto da data e ora].

$$\text{PAGINA} \rhd\!\lhd_{\text{URL=UrlPag} \wedge \text{Proprietario=IdUsr}}$$
$$(\ (\text{ACCESSO} - (\text{ACCESSO} \rhd\!\lhd_{(\text{UrlPag=UrlPag}) \wedge (\text{Data<Data})} \text{ACCESSO})) -$$
$$(\text{ACCESSO} \rhd\!\lhd_{(\text{UrlPag=UrlPag}) \wedge (\text{Data=Data}) \wedge (\text{Ora<Ora})} \text{ACCESSO}) \quad)$$

Da ottimizzare col push delle proiezioni…

3. Usando la ricorsione, esprimere la seguente interrogazione (**1.5 punti**):
Sapendo che l'utente "Alex" ha visitato la pagina "p001" in data "03-09-03" alle ore "13:20:05", estrarre tutte le pagine cui l'utente ha fatto accesso successivamente.

```
Raggiunge("p001") :-   ACCESSO ("Alex", "03-09-03", "13:20:05" "p001",_).
Raggiunge(X) :-        ACCESSO ("Alex", Data, Ora,X,Y), raggiunge(Y),
                       Data>="03-09-03", Ora>"13:20:05"
```

Facoltativo (1 punto): Discutere come realizzare un insieme di regole che traccino il percorso seguito da Alex dopo l'accesso alla pagina "p001" in data "03-09-03" alle ore "13:20:05", cioe' che estraggano il numero progressivo di accesso, l'ora e la pagina acceduta, limitandosi ad osservare gli accessi del giorno corrente (cioe` in data "03-09-03").

```
Percorso(P, O, 1) :- ACCESSO ("Alex", "03-09-03", O,P,_), O="13:20:05", P="p001".
Percorso(P, O, N):- Next(P,O,N), ¬Futuro(P,O,N).
Next(Y,O,N):-       ACCESSO ("Alex", "03-09-03", O, Y, X),
                    Percorso(X, O2, N-1), O>O2.
Futuro(P,O,N):-     Next(P, O, N), Next(P, O2, N), O>O2
```

C. Interrogazioni in SQL (8 PUNTI)

1. *Estrarre la dimensione totale in KByte delle pagine scaricate dagli "studenti" nel luglio 2003* ***(1 punto)***

```
SELECT sum(DimInByte)/1024
FROM (ACCESSO A join UTENTE U on A.IdUsr=U.IdUsr) join PAGINA on UrlPag=URL
WHERE Categoria="Studente" AND Data between 01-07-03 and 31-07-03
```

2. *Trovare i titoli delle pagine che non sono state visitate dai loro proprietari nell'ottobre 2003 (1.5 punti)*

```
SELECT distinct Titolo
FROM PAGINA P
WHERE URL NOT IN
  ( SELECT UrlPag
    FROM ACCESSO
    WHERE IdUsr=P.Proprietario AND Data between 1/10/03 and 31/10/03 )
```

3. *Determinare la categoria di utenti (diversa da "studenti") che ha effettuato il maggior numero di accessi a pagine di tipo "didattica"* ***(2.5 punti).***

```
SELECT Categoria
FROM (ACCESSO A join UTENTE U on A.IdUsr=U.IdUsr) join PAGINA on UrlPag=URL
WHERE Categoria <> "Studenti" AND Tipo = "Didattica"
GROUP BY Categoria
HAVING count(*) >= ALL
 (SELECT count(*)
  FROM (ACCESSO A join UTENTE U on A.IdUsr=U.IdUsr) join PAGINA on UrlPag=URL
  WHERE Categoria <> "Studenti" AND Tipo = "Didattica"
  GROUP BY Categoria)
```

Oppure

```
CREATE VIEW SOMMA-ACCESSI(Cat, Num) AS
SELECT Categoria, Count(*)
FROM (ACCESSO A join UTENTE U on A.IdUsr=U.IdUsr) join PAGINA on UrlPag=URL
WHERE Categoria <> "Studenti" AND Tipo = "Didattica"
GROUP BY Categoria

SELECT Categoria
FROM SOMMA-ACCESSI
WHERE Num = ( SELECT max(Num) FROM SOMMA-ACCESSI)
```

4. *Trovare gli utenti che hanno visitato almeno 10 pagine diverse e non hanno mai iniziato le loro navigazioni prima delle 13:00 [l'inizio di navigazione viene caratterizzato da un accesso privo di pagina di provenienza]* ***(3 punti).***

```
SELECT IdUsr, Nome
FROM UTENTI
WHERE IdUsr IN   (     SELECT IdUsr FROM ACCESSO
                       GROUP BY IdUsr
                       HAVING count(distinct UrlPag) >= 10     )  AND
    IdUsr NOT IN   (     SELECT IdUsr FROM ACCESSO
                       WHERE Ora < 13:00:00 and PaginaProvenienza IS NULL )
```