PROPRIETA’ DELLE TRANSAZIONI (ACID)

Atomicità=una transazione è un’unità atomica di elaborazione da eseguire o completamente o per niente

Consistency preserving (preservazione della consistenza)=una transazione deve far passare il database da uno stato consistente ad un altro

Isolation (isolamento)= una transazione non deve rendere visibili i suoi aggiornamenti ad altre transazioni finchè non è committed

Durability= se una transazione cambia e il cambiamento è committed queste modifiche non devono essere perse a causa di fallimenti successivi

MODELLO ER

[ENTITA’](#ENTITA)

Le entità corrispondono a classi di oggetti del mondo reale che hanno proprietà omogenee, comuni ed esistenza “autonoma” ai fini dell’applicazione di interesse.

Ogni entità ha un nome che la identificano univocamente nello schema e viene rappresentata graficamente con un rettangolo con il nome dell’entità all’interno

Ad ogni entità corrispondono degli attributi (compresa di chiave) e le opportune relazioni

[ATTRIBUTI](#ATTRIBUTI)

Gli attributi sono delle proprietà che caratterizzano le entità ed ogni attributo ha un nome che lo identifica. Possono essere semplici e composti, single-valued e multivalued, memorizzati e derivati. Gli attributi semplici sono quelli per cui ogni entità ha un valore singolo per tale attributo. Gli attributi composti sono quelli che possono essere divisi in sottoparti. Vale lo stesso per gli attributi single-valued e multi-valued. Gli attributi memorizzati sono quegli attributi che sono specificati mentre i derivati sono quelli che si ottengono attraverso le informazioni di altri attributi. Un attributo può essere chiave e questo deve essere un attributo che deve avere un valore unico per ogni entità

[DOMINIO DI UN ATTRIBUTO](#DOMINIODIUNATTRIBUTO)

Il dominio di un attributo è un range di valori che può assumere quell’attributo, come ad esempio “età” può avere come dominio “16-40”

Per un attributo composto il dominio è il prodotto cartesiano di ogni dominio degli attributi da cui è composto

[RELAZIONE](#RELAZIONE)

Una relazione è un’associazione tra n tipi di entità E1,E2,En. Le occorrenze o istanze di relazione associano n entità dei tipi di relazione richiesti. Ogni tipo di entità partecipa al tipo di relazione e il grado di un tipo di relazione è il numero di entità che partecipano ad esse. (Se il grado è 2 la relazione è detta binaria)

I tipi di relazione sono di due tipi: quelle normali che associano due entità diverse e le relazioni ricorsive che associano un’entità a sé stessa ma avendo due ruoli diversi

[RELAZIONI RICORSIVE](#RELAZIONIRICORSIVE)

le relazioni ricorsive si hanno quando lo stesso tipo di entità partecipa ad un tipo di relazione più di una volta con ruoli diversi, il nome del ruolo in questo caso diventa essenziale per distinguere il significato di ciascuna partecipazione

[VINCOLI](#VINCOLI)

Esistono due tipi di vincoli: 1)rapporto di cardinalità che specifica il numero di istanze di relazione in cui un’entità può partecipare e 2) il vincolo di partecipazione e abbiamo la partecipazione totale o parziale

[RAPPORTO DI CARDINALITÀ](#RAPPORTODICARDINALITA)

Il rapporto di cardinalità non è altro che l’inserimento della partecipazione minima e massima ad una determinata relazione, nella maggior parte dei casi si utilizzano solamente 3 valori: 0, 1 ed N.

Il valore 0 per la cardinalità minima indica una partecipazione opzionale del tipo di entità alla relazione

ENTITA’ DEBOLI

Sono tipi di entità che possono non avere attributi chiave propri. Sono relati a entità specifiche da un altro tipo di entità in combinazioni con alcuni valori dei suoi attributi.

Normalmente ha una chiave parziale che è l’insieme di attributi che identifica univocamente due entità relate alla stessa entità possessore

Un tipo di entità debole ha sempre un vincolo di partecipazione totale. I tipi di entità deboli possono anche essere rappresentati come attributi composti multivalued

MODELLO EER

SOTTOCLASSE

Una sottoclasse è un raggruppamento che dipende da particolari caratteristiche. Ci sono delle situazioni dove è necessario rappresentare questi raggruppamenti come ad esempio per memorizzare delle informazioni specifiche. Ad esempio un’entità impiegato potrebbe essere raggruppata in “segretaria, ingegnere, tecnico ecc ecc”. questo legame è definito come relazione classe/sottoclasse

Siccome una sottoclasse è membro di una superclasse deve avere valori per tutti gli attributi della superclasse ed eredita anche tutti i tipi di relazione in cui partecipa la superclasse

SPECIALIZZAZIONE

È il processo di definire un insieme di sottoclassi sulla base di particolari caratteristiche. Se tutte le classi in una specializzazione hanno la condizione di appartenenza sullo stesso attributo la specializzazione è detta attribute-defined

SOTTOCLASSI USER-DEFINED

Se non abbiamo alcuna condizione per determinare l’appartenenza ad una sottoclasse, la sottoclasse è detta user-defined. In questo caso l’appartenenza è specificata individualmente per ogni entità dall’utente e non da una condizione valutabile automaticamente

VINCOLO DI DISGIUNZIONE

Un’entità può essere membro di al di più una sottoclasse e si denota con d. se non vale tale vincolo, le entità possono avere un overlap. La stessa entità può essere membro di più sottoclasse e si denota con o.

VINCOLO DI COMPLETEZZA

Un vincolo di specializzazione totale specifica che ogni entità deve appartenere ad una sottoclasse e si denota con una linea doppia che collega la superclasse al cerchio. Una specializzazione parziale permette a qualche entità di non appartenere a nessuna sottoclasse. Si denota con una linea singola che collega la superclasse al cerchio

COMBINAZIONI DI VINCOLI

I due vincoli sono indipendenti, pertanto sono ammesse tutte e quattro le combinazioni: -disgiunto totale, -disgiunto parziale, -overlap totale, -overlap parziale

VINCOLI, CON INSERIMENTO E CANCELLAZIONI

Cancellare un’entità da una superclasse implica che sia cancellata automaticamente da tutte le sottoclassi. Vale lo stesso per l’inserimento. Inserire un’entità in una superclasse di una specializzazione totale implica che l’entità sia inserita obbligatoriamente in almeno una sottoclasse

SOTTOCLASSI SHARED

Sono delle sottoclassi che hanno più di una superclasse e questo porta al concetto di ereditarietà multipla poiché una classe shared eredita gli attributi da più superclassi ma non sempre è consentita perché può creare dei conflitti con gli attributi

TIPI UNIONE E CATEGORIE

Quando più entità condividono una relazione, per unirla creiamo una sottoclasse che è l’unione di più tipi di entità e questa sottoclasse è detta di tipo unione o categoria

GRADO RELAZIONI

Il grado di una relazione si vede in base a quante entità sono collegate ad essa

MODELLO RELAZIONALE

DOMINI

Un dominio è un insieme di valori atomici. Un metodo per specificare un dominio è specificare un tipo di dato da cui sono presi i dati che formano il dominio

VINCOLI DI DOMINIO

Il valore di ciascun attributo di A deve essere un valore atomico appartenente al dominio

VINCOLI DI CHIAVE

Una relazione è definita come un insieme di tuple. Tutti gli elementi di un insieme sono distinti. Formalmente una chiave è una superchiave tale che rimuovendo uno dei suoi attributi non è più una superchiave, in questo caso la chiave è detta minimale. In una relazione possono esistere più chiavi che sono definite chiavi candidate e in tal caso se ne sceglie una come chiave primaria. Una chiave deve godere anche delle proprietà di invarianza nel tempo (non possono esserci copie di chiavi e non possono avere valori null). Ci sono anche le chiavi esterne che sono paragonate ai puntatori in C, perché si riferiscono ad altre chiavi

VINCOLI E OPERAZIONE INSERT

INSERT può violare tutte e 4 i tipi di vincoli:

dominio: un valore di un attributo può non apparire nel corrispondente dominio

chiave: il valore della chiave nella tupla già esistente nella relazione

integrità di entità: la chiave primaria inserita è null

integrità referenziale: il valore di una chiave esterna riferisce ad una tupla che non esiste nella relazione referenziata

per gestire la violazione bisogna: forzare l’inserimento completo o rifiutare l’inserimento

VINCOLI OPERAZIONE DELETE

La delete può violare solo l’integrità referenziale. Per gestire le violazioni bisogna:

rigettare la cancellazione, tentare di propagare la cancellazione, modificare i valori dell’attributo referenziante

o una combinazione delle 3

RISTRUTTURAZIONE

Per la ristrutturazione bisogna valutare gli indici di prestazioni, il costo di un’operazione e l’occupazione di memoria

Per studiare questi parametri è necessario conoscere il volume dei dati:

con le occorrenze di entità e relazioni e le dimensioni di ciascun attributo

le caratteristiche delle operazioni: tipo (interattiva o batch), frequenza e dati coinvolti

TAVOLA DEI VOLUMI

Concetto, tipo, volume. Il numero delle occorrenze delle associazioni dipende da due parametri:

1)numero delle occorrenze delle entità coinvolte nelle associazioni

2)numero medio di partecipazioni di un’occorrenza di entità alle occorrenze di associazioni

TAVOLA DELLE OPERAZIONI

Operazioni, tipo, frequenza

Viene riportata per ogni operazione la frequenza prevista e il tipo (interattiva o batch)

TAVOLE DI ANALISI

Concetto costrutto accessi tipo

Viene riportata per ogni operazione il numero di accessi ai concetti coinvolti ed il tipo di accesso (lettura o scrittura)

Le operazioni di scrittura sono più onerose di quelle di lettura

COSTO DI UN’OPERAZIONE

Il costo di un’operazione si ottiene contando il numero degli accessi delle occorrenze di entità e relazioni

RIDONDANZE

Una ridondanza è un attributo derivabile

Vantaggi= semplificazione delle interrogazioni

Svantaggi= appesantimento degli aggiornamenti e maggiore occupazione di spazio

La decisione su se mantenere o meno il dato ridondante si valuta attraverso il costo delle operazioni, per capire tra le due qual è la più vantaggiosa

ELIMINAZIONI DELLE GENERALIZZAZIONI

1. Accorpamento delle figlie della generalizzazione nel padre
2. Accorpamento del padre della generalizzazione nelle figlie
3. Sostituzione della generalizzazione con associazioni

COME SCEGLIERE TRA LE DIVERSE ALTERNATIVE

1. Conviene quando le operazioni non fanno molta distinzione tra le occorrenze e gli attributi delle entità figlie con l’entità padre. Introduciamo valori nulli ma abbiamo un minor numero di accessi
2. È applicabile quando la generalizzazione è totale
3. Quando la generalizzazione non è totale e ci sono operazioni che fanno distinzioni tra entità padre e entità figlie (è quella meno conveniente perché richiede un sacco di accessi)

**PER LE GENERALIZZAZIONI A PIU’ LIVELLI SI PUO’ PROCEDERE ANALIZZANDO UNA GENERALIZZAZIONE ALLA VOLTA A PARTIRE DAL FONDO DELLA GERARCHIA**

/\*PARTIZIONAMENTO/ACCORPAMENTO DI CONCETTI

1)Partizionamento di entità:

decomposizione verticale(attributi)

decomposizione orizzontale(entità)

2) eliminazione di attributi multivalore

3) accorpamento di entità\*/

IDENTIFICATORI

I criteri generali per scegliere una chiave primaria sono:

-escludere valori con attributi nulli

-numero minimo di attributi

-identificatore coinvolto in molte operazioni

-velocità di accesso all’indice

IL MODELLO RELAZIONALE NON CONSENTE I MULTIVALORI, PER CUI VANNO GESTITI ATTRAVERSO DELLE RELAZIONI

ALGEBRA RELAZIONALE

Dominio: relazioni

Operatori: operazioni su insiemi: unione, intersezione, differenza e prodotto cartesiano e operazioni per database relazionali: select, project e join

SELECT

È usato per selezionare un sottoinsieme di tuple in una relazione che soddisfa una condizione di selezione

La condizione di selezione è un’espressione booleana formata da clausole della forma:

<nome\_attributo> op\_confronto <valore costante>

Oppure

<nome\_attributo> op\_confronto <nome\_attributo>

Op\_confronto è uno degli operatori {=, !=, <, >, <=, >=}

La condizione di selezione viene eseguita per ogni tupla individualmente e se è vera, la tupla è inserita nella relazione risultante

Il grado della relazione è uguale a quello di partenza

Il numero di tuple risultanti è minore o uguale a quello di partenza

L’operatore select è unario o commutativo (unario=ha un unico argomento)

PROJECT

È usato per selezionare un sottoinsieme delle colonne di una relazione (in pratica gli attributi)

La relazione risultante ha gli attributi specificati nella <attribute\_list> nello stesso ordine in cui appaiono nella lista, motivo per cui non è un’operazione commutativa

Il grado risultante della project è uguale al numero di attributi specificati nella <attribue\_list>

Se nella project non è presente una chiave candidata si potrebbero avere delle tuple duplicate ma le rimuove implicitamente

Il numero di tuple risultanti è minore o uguale al numero di tuple di partenza

RENAME

È un operatore che serve a rinominare gli attributi in una relazione che risulta dall’algebra relazionale. È un operatore unario

UNION COMPATIBILITY

Per poter applicare un’operazione insiemistica a due relazioni queste devono essere union compatibility

Due relazioni sono union compatibility se hanno lo stesso grado e stesso dominio

Solo così possiamo effettuare l’unione, intersezione e differenza

Per il prodotto cartesiano la union compatibility non serve e nel caso in cui due attributi abbiano lo stesso nome nelle due relazioni, si deve effettuare il rename di uno dei due

JOIN

è un’opeazione abbastanza frequente, RxS, ed è per questo che è stata creata quest’operazione speciale JOIN

è usata per combinare tuple relate in una sola tupla

i diversi join: theta join, equijoin e natural join

il theta join correla i dati in due relazioni diverse sulla base di una condizione booleana

il natural join richiede che i due attributi di join abbiano lo stesso nome (se nessuna combinazione di tuple soddisfa la condizione di join, la relazione risultante avrà zero tuple)

l’equijoin è come il tetha join ma sua condizione è implicita ed è quella di uguaglianza

JOIN ESTERNO

Il join esterno estende, con valori nulli, le n-ple che verrebbero tagliate fuori da un join(interno)

Esistono tre versioni:

-sinistro(left)

-destro(right)

-completo(full)

SINISTRO= mantiene tutte le n-ple del primo operando, estendendole con valori nulli, se necessario

DESTRO= mantiene tutte le n-ple del secondo operando, estendendole con valori nulli, se necessario

FULL=mantiene tutte le n-ple di entrambi gli operatori, estendendole con valori nulli, se necessario

EQUIVALENZA DI ESPRESSIONI

Due espressioni sono equivalenti se producono lo stesso risultato qualunque sia l’istanza attuale della base di dati

NORMALIZZAZIONE

Esistono delle misure informali di qualità per il disegno di schemi di relazione:

1 semantica degli attributi

2 riduzione dei valori ridondanti nelle tuple

3 riduzione dei valori null nelle tuple

4 non consentire tuple spurie

SEMANTICA DEGLI ATTRIBUTI IN UNA RELAZIONE

La semantica specifica come interpretare i valori degli attributi in una relazione

UPDATE ANOMALIES

Le update anomalies sono delle anomalie che possono sorgere a causa di un database non correttamente progettato e si dividono in:

-insertion anomalies

-deletion anomalies

-modification anomalies

INSERTION ANOMALIES

È l’impossibilità di aggiungere dati al database a causa dell’assenza di altri dati

DELETION ANOMALIES

È quando non è possibile eliminare i dati dalla tabella senza dover eliminare l’intero record

MODIFICATION ANOMALIES

Se cambiamo il valore di uno degli attributi di una particolare entità che dipende da un’altra, dobbiamo aggiornare tutte le tuple di quell’entità

DIPENDENZE FUNZIONALI

È un vincolo tra due insiemi di attributi del database

RIDUZIONE DEI VALORI NULL NELLE TUPLE

in alcuni disegni di schemi possiamo raggruppare molti attributi in una relazione grossa ma se molti degli attributi non applicano a tutte le tuple possiamo avere molti valori null che oltre a sprecare spazio crea problemi con COUNT e SUM

INTERPRETAZIONE DI NULL

Il valore null può avere diverse interpretazioni:

-l’attributo non si applica a questa tupla

-il valore dell’attributo per questa tupla non è noto

-il valore dell’attributo è noto ma assente, cioè non è stato ancora registrato

DIPENDENZA FUNZIONALE

Una dipendenza funzionale è un vincolo tra due insiemi di attributi del database

Un vincolo stabilisce che se x->y allora per ogni t1 e t2 in r tali che t1[x]=t2[x] deve valere t1[y]=t2[y]

Ciò significa che i valori della componente y di una tupla di r dipendono da i valori della componente x

I valori della componente x di una tupla determinano univocamente i valori della componente y

Cioè esiste una dipendenza funzionale da x a y dove:

y è funzionalmente dipendente da x

x è la parte sinistra della dipendenza funzionale

y è la parte destra della dipendenza funzionale

REGOLE DI INTERFERENZA FUNZIONALE

R1 regola riflessiva

Se x è un sottoinsieme di y allora y dipende da x

R2 regola argomentativa

Se y dipende funzionalmente da x allora yz dipende funzionalmente da xz

R3 regola transitiva

Se Y è dipendente da X e Z è dipendente da X allora Z è dipendente da X

R4 regola di decomposizione

Se yz dipendono da x allora z dipende da x

R5 regola di unione

Se y dipende da x e z dipende da x allora yz dipende da x

R6 regola pseudotransitiva

Se y dipende da x e z dipede da wy, allora z dipende da wx

Immagine che contiene testo, screenshot, computer, monitor

Descrizione generata automaticamente

Immagine che contiene testo, screenshot, elettronico, schermo

Descrizione generata automaticamente

Immagine che contiene testo, screenshot, monitor, computer

Descrizione generata automaticamente

Immagine che contiene testo, screenshot, monitor, computer

Descrizione generata automaticamente

Immagine che contiene testo, screenshot, monitor, computer

Descrizione generata automaticamente

AImmagine che contiene testo, screenshot, monitor, computer

Descrizione generata automaticamente

LE REGOLE DI INTERFERENZA DI ARMSTRONG COMPRENDONO LA R1,R2 ED R3

NORMALIZZAZIONE DEI DATI

La normalizzazione dei dati può essere vista come un processo che consente di decomporre schemi di relazione non ottimali in schemi più piccoli tali da garantire la mancanza di update anomalies

Definizioni varie

Una chiave k è una superchiave con la proprietà aggiuntiva che la rimozione da k di un qualche attributo fa perdere a k la proprietà di superchiave

Tra tutte le chiavi candidate una di esse viene scelta arbitrariamente ed è detta chiave primaria

FORME NORMALI

1NF

La 1NF è stata definita per non consentire attributi multivalued, composti e loro combinazioni

Gli unici valori consentiti da 1NF sono valori atomici

Per normalizzare bisogna rimuovere gli attributi della relazione annidata in primaria in essa

2NF

È basata sul concetto di dipendenza funzionale piena

Una dipendenza funzionale x->y è piena se la rimozione di qualche attributo A da X implica che la dipendenza non vale più

Mentre invece una dipendenza funzionale è parziale se qualche attributo di X viene rimosso e la dipendenza vale ancora

Uno schema è in 2NF se:

-R è in prima forma normale

-ogni attributo non primo A in R è pienamente funzionalmente dipendente dalla chiave primaria di R

3NF

È basata sul concetto di dipendenza transitiva

Una dipendenza funzionale x->y in uno schema R è una dipendenza transitiva se esiste un insieme di attributi Z che non è sottoinsieme di alcuna chiave di R e valgono X->Z e Z->Y

Uno schema è in 3NF se:

-è in 2F

-nessun attributo non primo di R è transitivamente dipendente dalla chiave primaria

SQL

Sql è un linguaggio dichiarativo, basato in parte sull’algebra relazionale ed in parte sul calcolo relazionale

L’utente specifica quale risultato deve essere raggiunto mentre il DBMS decide l’rdine delle operazioni

TERMINOLOGIA SQL (cambiamenti)

Relazione diventa tabella

Tupla diventa riga

Attributo diventa colonna

COSA CONTIENE UNO SCHEMA

Tabelle

Domini (il dominio corrisponde ad un tipo di dato)

Viste (sono delle tabelle virtuali che includono la colonna che scegliamo noi di mostrare)

Altri costrutti come permessi di autorizzazione ecc ecc

CREARE UNO SCHEMA

CREATE SCHEMA nome\_schema

AUTHORIZATION nome\_utente

CREARE UNA TABELLA

Si usa la CREATE TABLE ed è usata per specificare una nuova relazione assegnandole un nome ed un insieme di attributi e vincoli

Gli attributi sono specificati da un nome, un tipo di dato per definire il dominio dei valori ed eventuali vincoli

TIPI DI DATI E DOMINI

Numerici:

-interi (int, smallint)

-reali(flloat, real, double precision)

-numerici formattati(DECIMAL(i,j), DEC(i,j), NUMERIC(i,j))

Dove i indica il numero di cifre decimali

J indicia il numero di cifre dopo la virgola

Stringhe di caratteri:

-a lunghezza fissa(CHAR(n))  
-a lunghezza variabile(VARCHAR(n))

Per default n vale 1

Stringhe di bit:

-a lunghezza fissa(BIT(n))

-a lunghezza variabile(BIT VARYING(n))

Date:

ha 10 posizioni, con componenti YEAR, MONTH e DAY

il suo formato è: YYYY-MM-DD

Time:

ha almeno 8 posizioni con componenti HOUR, MINUTE e SECOND

il suo formato è: HH:MM:SS

VALORI NULL E DEFAULT

In SQL è possibile inserire dei valori di tipo null, per cui se si vuole impedire ciò ad alcuni attributi bisogna usare il vincolo NOT NULL

Questo vincolo va sempre specificata per la chiave primaria

È anche possibile specificare un valore di default per un attributo attraverso la clausola DEFAULT<value> dopo la dichiarazione dell’attributo

Senza la clausola il valore di default di un attributo è null

CHIAVI

PRIMARY KEY specifica una o più attributi che faranno da chiave primaria

UNIQUE specifica una chiave alternativa

FOREIGN KEY specifica l’integrità referenziale (è una chiave esterna)

TRIGGERED

L’azione referenziale triggered può essere specificata nella clausola FOREIGN KEY

Delle possibili azioni sono SET NULL, CASCADE e SET DEFAULT

RELAZIONI BASE

Le relazioni create con CREATE TABLE sono dette tabelle base o relazioni base e significa che sono create con e memorizzate come file dal DBMS

Le relazioni base sono distinte dalle relazioni virtuali, create con CREATE VIEW, cui può o meno corrispondere un file fisico

DROP SCHEMA

Se uno schema non è più necessario si usa il comando DROP SCHEMA con due possibili opzioni: CASCADE e RESTRICT

Se si usa CASCADE lo schema viene rimosso con tutte le tabelle, domini ed altri elementi

Se si usa RESTRICT lo schema è eliminato solo se non contiene elementi

ALTER TABLE

La definizione di una tabella base può essere cambiata usando il comando ALTER TABLE

Le modifiche che possono essere svolte sono:

-aggiunta o rimozione di attributi

-cambio di definizione di una colonna

-aggiunta o rimozione di un vincolo

SELECT

Il comando select è l’istruzione di base per recuperare informazioni da un database

RENAMING

Lo stesso nome può essere usato per più attributi solo se questi appartengono a relazioni diverse

Se una query coinvolge queste relazioni bisogna qualificare il nome dell’attributo con il nome della relazione per evitare ambiguità

Per rinominare qualsiasi attributo si usa AS

MANCANZA DEL WHERE

Omettere la where equivale a WHERE TRUE, cioè tutte le tuple della relazione nella FROM fanno parte del risultato

Se più di una relazione è coinvolta nella FROM allora il risultato sarà il prodotto cartesiano delle relazioni

CARATTERE JOLLY

per recuperare tutti gli attributi delle tuple selezionate si usa il carattere jolly \*

DUPLICAZIONI DI TUPLE IN SQL

Sql non tratta relazioni come insiemi: tuple duplicate possono apparire più di una volta

Sql non elimina le dupliazioni perché:

-è un’operazione costosa

-l’utente può essere interessato alle duplicazioni

-con funzioni di aggregazione siamo interessati a non eliminarle

DISTINCT

Se le duplicazioni non sono volute lo si specifica con la clausola DISTINCT

OPERAZIONI INSIEMISTICHE

Sql incorpora le seguenti operazioni insiemistiche:

-union

-except

-intersect

EXCEPT restituisce tutti i valori distinti della query a sinistra dell’operando non presenti nella query a destra

Usando queste operazioni le tuple duplicate sono eliminate

CONFRONTO TRA SOTTOSTRINGHE

Per il confronto tra stringhe si usa l’operatore LIKE

% rimpiazza qualsiasi numero di caratteri

\_ rimpiazza un singolo carattere

OPERATORE IN

L’operatore IN confronta un valore con un insieme di tuple union-compatibili

L’operatore IN permette di specificare valori multipli nella clausola WHERE

ANY

Confronta un singolo valore v con un multiset restituendo TRUE se v è uguale a qualche valore in V

ANY è l’equivalente dell’operatore IN

ALL

Anche ALL può essere combinato con questi operatori

ALL sarà TRUE se v che è combinato con il multiset sarà vero per ogni dato

EXISTS e NOT EXISTS

Servono per verificare se il risultato di una query annidata correlata è vuota

TIPI DI JOIN USATI IN SQL

NATURAL JOIN

INNER JOIN

LEFT OUTER JOIN/ RIGHT OUTER JOIN

FULL OUTER JOIN

FUNZIONI IN SQL

COUNT= conteggio multiple

COUNT(DISTINCT)= conteggio di tuple senza duplicazioni

SUM= somma dei valori di un attributo in una tabella

MAX= valore massimo tra gli attributi di una tabella

MIN= valore minimo tra gli attributi di una tabella

AVG= valore medio tra gli attributi di una tabella

STD= deviazione standard tra gli attributi di una tabella

ORDINAMENTO TUPLE

Per ordinare le tuple nel risultato della query si usa ORDER BY

L’ordine di default è crescente

ASC per un ordine crescente

DESC per un ordine decrescente

INSERT INTO

Inserisce nuove righe in una relazione

È possibile non assegnare valore a tutti gli attributi, in questo caso questi avranno il valore di default o null

AUTO-INCREMENT

Può essere usato per generare un identificatore unico per le nuove righe

DELETE

Rimuove una o più tuple da una relazione

UPDATE

Permette di modificare valori in una relazione

VISTE IN SQL

Le viste sono tabelle virtuali derivate da tabelle esistenti nel db

Posson essere definite per nascondere dei dati da alcune tabelle

INDICI IN SQL

Per creare indici si usa il comando CREARE INDEX

MYSQL

DARE DIRITTI AD UN UTENTE

Per dare dei diritti si usa:

GRANT <lista privilegi> ON <nome database>.<nome tabella> TO <lista utenti>

ALTER consente l’uso del comando ALTER TABLE

CREATE consente l’uso di CREATE TABLE

DELETE consente l’uso di DELETE

DROP consente l’uso di DROP TABLE

INSERT consente l’uso di INSERT

SELECT consente l’uso di SELECT

UPDATE consente l’uso di UPDATE

REFERENCES consente l’accesso alle chiavi esterne

INDEX consente la creazione ed eliminazione di indici con il comando ALTER TABLE

ALL o ALL PRIVILEGES concede tutti i privilegi

USAGE concede solo il diritto di collegarsi al server

GRANT OPTION

Possiamo consentire gli utenti a concedere questi diritti ad altri utenti

Basta aggiungere WITH GRANT OPTION

CREARE UTENTE MYSQL

CREATE USER <id-utente> IDENTIFIED BY <password>

CANCELLARE UTENTE MYSQL

DROP USER <id-utente>

CAMBIARE PASSWORD

SET PASSWORD FOR ‘nome’@’localhost’=’password’

Oppure

ALTER USER ‘nome’@’localhost’ IDENTIFIED BY ‘password’

ALCUNI COMANDI UTILI

SELECT version(); restituisce il numero di versione del dbms

SELECT now(), current\_date(); restituisce data-ora e data corrente

SELECT now(),current\_date() FROM dual;

SHOW databases; mostra i database presenti nell’installazione corrente di mysql

COMANDI MYSQL

USE <nome\_bd> usa quel database

SELECT database(); mostra il database corrente

SHOW tables[from <nome\_db>]; mostra le tabelle che fanno parte di un db

QUIT; uscire dal client

MODIFICARE I DATI DI UNA TABELLA

UPDATE <nome\_tabella>

SET nome\_colonna=espressione[nome\_colonna2=espressione2]

WHERE …

CANCELLARE DATI DA UNA TABELLA

DELETE FROM <nome\_tabella>

WHERE

SHOW CREATE TABLE<nome\_tabella>

Si può:

-rivisualizzare il comando usato per creare una tabella scelta

-copiarlo su un editor di testi

-eliminare la “spazzatura di contorno”

-ricopiarla sul monitor mysql

TRIGGER

I trigger sono un meccanismo attraverso il quale è possibile automatizzare le operazioni INSERT, UPDATE o DELETE al verificarsi di eventi riguardanti i dati

Quando definiamo un trigger in mysql dobbiamo stabilire se questo debba innerscarsi prima o dopo un certo evento

CREATE TRIGGER nome\_trigger

Tempistica evento ON nome\_tabella

FOR EACH ROW

BEGIN

…

END;

SHOW TRIGGERS; elenca i trigger

DROP TRIGGER nome\_trigger; cancella un trigger

DROP TRIGGER IF EXISTS nome\_trigger; cancella se esiste

REGOLE DI MAPPING

Per le entità e relazioni M:N la relazione chiede la chiave primaria di entrambe le entità

Per le relazioni 1:M la relazione non viene scritta ma l’entità principale (non debole) prende eventualmente gli attributi della relazione se ci sono

Per le relazioni 1:M con partecipazione opzionale la relazione non viene scritta ma l’entità principale vuole la chiave dell’entità debole

Per la relazione 1:1 la relazione non viene scritta ma una delle due entità prende la chiave dell’altro

Nel caso di partecipazione opzionale la relazione non viene scritta, l’entità (1,1) prende la chiave dell’altra entità

Nel caso di partecipazione opzionale per entrambi la relazione viene scritta e prende la chiave di entrambe le entità

Per le relazioni ricorsive si scrive la relazione che prende entrambe le chiavi delle entità nonostante siano un’entità solo anche se diverse

PROJECT IN SQL/ALGEBRA RELAZIONALE

Nell’algebra relazionale è usato per sottolineare un sottoinsieme delle colonne in una relazione

La relazione risultante ha gli attributi specificati nella lista di attributi, nello stesso ordine in cui appaiono nella lista, motivo per cui non è un’operazione commutativa

Il grado della relazione risultante è uguale al numero di elementi definiti nella lista di attributi

Se nella lista di attributi non è presente una chiave candidata si potrebbe avere delle tuple duplicate ma la project le rimuove implicitamente

Il numero di tuple risultanti è minore o uguale al numero di tuple di partenza

In SQL la proiezione è un operatore che ha in input una tabella e fornisce in output un’altra tabella con grado minore o uguale a quella di partenza, la cardinalità minore o uguale a quella di partenza

Sarà minore quando la tabella risultato contiene righe uguali che si riducono ad uno in quanto nel modello relazionale le righe delle tabelle sono tutte distinte

OPERATORE SELECT

È usato per selezionare un sottoinsieme di tuple in una relazione che soddisfa una condizione di selezione (tipo come la where in sql)

La condizione di selezione è un’espressione booleana formata da clausole della forma:

<nome\_attributo> op\_confronto <valore costante> oppure

<nome\_attributo> op\_confronto <nome\_attributo>

Gli operatori di confronto sono =,!=,<,>,<=,>=

La condizione di selezione è valutata per ogni tupla individualmente: se è vera, la tupla è inserita nella relazione risultante

Il grado della relazione risultante dopo un’operazione di select è uguale a quello della relazione di partenza

L’operatore select è unario e commutativo

NATURAL JOIN

Il join è fatto su tutte le colonne con lo stesso nome e rimuove le colonne duplicate del risultato a differenza di tutti gli altri join

TETHA JOIN

È il join più comune in quanti permette di specificare una condizione

EQUIJOIN

È come il tetha join ma la sua condizione è implicita ed è quella di uguaglianza

CROSS JOIN

Questo tipo di join combina tutte le righe della tabella a sinistra con tutte le righe della tabella a destra e serve quando si desidera selezionare tutte le possibili combinazioni di righe e colonne da entrambe le tabelle

INNER JOIN

Questo tipo di JOIN restituisce tutte le righe delle tabelle coinvolte in cui esista una corrispondenza tra i valori delle colonne presenti nella clausola

OUTER JOIN

A sua volta l'OUTER JOIN può essere di tre tipi:

* LEFT OUTER JOIN
* RIGHT OUTER JOIN
* FULL OUTER JOIN

Questo tipo di JOIN quando si desidera selezionare tutte le righe della tabella che si trova a sinistra (LEFT) o a destra (RIGHT) della clausola di JOIN o quando si desidera selezionare tutte le righe di entrambe (FULL), indipendentemente dalla corrispondenza tra i campi presenti nella clausola