

DBMS

DataBase Management System

L. Vigliano

DBMS

- Un DataBase Management System è un sistema di gestione il cui obiettivo generale è mantenere le informazioni (ovvero qualsiasi cosa sia ritenuta interessante da un individuo o da una organizzazione a cui il sistema è rivolto) e renderle disponibili su richiesta.

J. Date

L. Vigliano

DBMS

Caratteristiche

- Un DBMS deve garantire :
 - Condivisione dei dati
 - Database persistenti
 - Affidabilità
 - Privatezza
 - Efficienza
 - Efficacia

DBMS (2)

- **MYSQL** sistema relazionale di gestione per basi di dati (DBMS)
- **SQL** linguaggio per l'interrogazione e la manipolazione dei dati

MYSQL

- Basato su routine ISAM, scritto in C e C++
- Interfacciabile da C, C++, Perl, PHP, Eiffel, Java, Python, TC1API
- Gestisce fino a 60.000 tabelle e fino a 50 milioni di record
- **DDL** Data Definition Language
- **DML** Data Manipulation Language
- **SQL** Structured Query Language

Primi Comandi

- Show databases ;
- Use <nomedb>;
- Create database <nomedb>;
- Exit;

-

ATTENZIONE AL

;

Creare e cancellare database (con molta attenzione !!!)

- Create database <nomedb>;
- Drop database <nomedb>;

Usare un database

- Use <nomedb>;

Tipi di Dati

Numerici Interi

- **Tinyint** piccolissimo intero
da -128 a 127 **unsigned** 0-255
- **Smallint** piccolo intero
da -32768 a 32767 **unsigned** 0-65535
- **Int** intero
da -2.147.483.648 a 2.147.483.647
unsigned 0-4.294.967.295
..... più altri intermedi
opzione **AUTO_INCREMENT**

Tipi di dati (2)

Numerici decimali

- **Float(M,D)** dove **M** numero cifre parte intera
e **D** numero cifre parte decimale

Double(M,D) più ampio (doppia precisione)

Tipi di dati (3)

Alfanumerici

- **Char(x)** Stringa di max 255 char di lunghezza fissa
- **Varchar(x)** Stringa di max 255 char di lunghezza variabile
- **Text** file 65535 byte
- **Blob** file o immagine (Binary Large Object)

..... più altri intermedi

NON CREARE MAI INDICI SUGLI ULTIMI DUE !

Tipi di dati (3)

Temporal

TIPO

FORMATO

- Datetime aaaa-mm-gg hh:mm:ss
- Date aaaa-mm-gg
- Time hh:mm:ss
- Year aaaa
- Timestamp(x) Variabile a seconda di x che varia da 2 a 14

DATABASE IMPIEGATI

emp

Empno	Ename	Job	Date	Sal	deptno
7369	Smith	Clerck	1980-12-17	800	20
7499	Allen	Salesman	1981-02-20	1600	30
7521	Ward	Salesman	1981-02-22	1250	30
7566	Jones	Manager	1981-04-02	2950	20
7654	Martin	Salesman	1981-09-28	1250	30
7698	Blake	Manager	1981-05-01	2850	30
7782	Clarck	Manager	1981-06-09	2450	10
7788	Scott	Analyst	1981-11-09	3000	20
7839	King	President	1981-11-17	5000	10
7844	Turner	Salesman	1981-09-08	1500	30
7876	Adams	Clerck	1981-09-23	1100	20
7900	James	Clerck	1981-12-03	950	30
7902	Ford	Analyst	1981-12.03	3000	20
7934	Miller	Clerck	1982-01-23	1300	10

dept

Dptno	Dname	Loc
10	Accounting	New York
20	Reseaech	Dallas
30	Sales	Chicago
40	Operations	Boston

Creare e Cancellare Tabelle

- Create table <nometabella>(
 <nome attributo> tipodato opzioni,
 <nome attributo> tipodato opzioni

....)opzioni;

Drop table <nometabella>;

Creare e Cancellare Tabelle

- Create table emp(
 empno int not null primary key
 auto_increment,
 ename varchar(30) not null,

 sal float(4,2)
 );

Drop table emp;

Creare Tabelle con Reference

- Create table emp(
 empno int not null primary key
 auto_increment,
 deptno char(2) not null,
 foreign key (deptno) references dept
 (deptno) on update cascade on delete no action,
 ename varchar(30) not null,

 sal float(4,2)
....) Engine=InnoDB;

Gestione Tabelle

```
Create table empvip(ename,sal) as  
select ename, sal from emp  
where job = 'manager' ;
```

Modificare una tabella già esistente

```
alter table emp add projno int;
```

Rinominare una tabella

```
alter table emp rename impiegati;
```

Creare tabella solo se non esiste già

```
create table if not exist emp(.....);
```

Indici

Create index *indexname*
on *tablename(columnnodelist)*;

Esempio

create index indnome
on emp(ename);

Drop index *indexname*
on *tablename*;

DML

Data Manipulation Language

L. Vigliano

Insert

```
Insert into <nometabella> (<nome  
attributo>, <nome attributo>,...)  
values(<valore>,<valore>,.....);
```

Insert

Insert into emp

```
values(7954,' Carter' , ' clerk' ,7698,  
      '1984-04-07' ,1000,NULL,30);
```

```
Insert into emp(empno,ename,deptno)  
values(7955,' Wilson' ,30);
```

Insert esterno

Load data infile “*nomefile.txt*”
into table *tablename*
(*campo1, campo2, ..., campon*);

Update

```
Update <nome tabella>  
  set <nome attributo> = <valore>,  
      <nome attributo>=<valore>,...  
  where <condizioni>;
```

Update

Update emp

```
set job = "salesman", sal=1,1* sal  
where ename = "Wilson";
```

Update emp

```
set sal = (select 2*avg(sal) from emp  
           where job= 'salesman' )  
where job= 'salesman' ;
```


Delete

Delete from <nome tabella>
where <condizioni>;

Delete

```
Delete from emp  
      where ename = "Wilson";
```

```
Delete from emp  
      where job in  
            (select .....);
```

SQL

Structured Query Language

L. Vigliano

Select

Select *nomicampi*
from *tabelle*
where *condizioni* ;

Esempio

select job from emp; (quali compaiono)
select * from emp;

Select

Opzione Distinct

`select distinct job from emp;` (quali sono)

`select distinct autore from arau;`

Select Opzione Alias

```
select empno codice  
from emp;
```

Operatori per la clausola WHERE

= uguale

!= diverso

<> diverso

> maggiore

>= maggiore o uguale

< minore

<= minore o uguale

Operatori per la clausola WHERE (2)

Between ...	compreso tra
and	e
in	appartenente a
like	del tipo ...
is NULL	è un valore assente

NOT	Negazione
AND	Congiunzione
OR	Disgiunzione

Select con clausola Where esempi

- Confronto

```
select * from emp where job = 'clerk';
```

- Appartenenza

```
select ename, job, sal from emp  
where sal between 1200 and 5000;
```

- Somiglianza ortografica

```
select autore, qualifc from au  
where autore like 'A%';
```

Select con Join (Theta-style)

```
Select ename, loc  
from emp, dept  
where ename = 'Allen' and  
emp.deptno=dept.deptno;
```

Dove lavora Allen ?

Select con Join (Ansi-style)

```
Select ename, loc  
from emp join dept  
on emp.deptno=dept.deptno and  
   ename = 'Allen';
```

Dove lavora Allen ?

Select con Funzioni

Funzioni di insieme

AVG media

COUNT contatore

MAX massimo

MIN minimo

SUM sommatoria

Funzioni aritmetiche

+ - * / ABS ROUND TRUNC

Select con Funzioni esempi

Conta tutti gli articoli di Moscarini

```
select count(*) from arau  
where autore = 'Moscarini' ;
```

```
select max(sal), min(sal),  
       max(sal)-min(sal)  
from emp;
```

Select Aggregate opzioni

ORDER BY

GROUP BY

.... HAVING

Select Aggregate esempi

```
select sal, job, ename from emp  
  where deptno = 30  
  order by sal desc;
```

```
select deptno, 12*avg(sal) from emp  
  where job not in ( 'manager' , ' president' )  
  group by deptno;
```

```
select job, count(*), avg(sal) from emp  
  group by job having count(*) > 1 ;
```

Select Nidificate

- Selezionare tutti gli impiegati che lavorano nello stesso dipartimento di 'Allen'

Select **ename** from **emp**

where **deptno**=

(select **deptno** from **emp**

where **ename** = 'Allen');

Select Nidificate

versione con alias

- Selezionare tutti gli impiegati che lavorano nello stesso dipartimento di 'Allen'

```
Select x.ename from emp x, emp y  
where x.deptno=y.deptno  
and y.ename = 'Allen' ;
```

Algebra su Select

- Si possono effettuare

UNION

INTERSECT

MINUS

di select

Viste o Tabelle Virtuali

L. Vigliano

View o Tabelle Virtuali

- Le view non memorizzano alcun dato
- Table, e View da essa derivata, condividono lo stesso spazio
- Per questo se aggiorni una view, aggiorni anche la tabella
 - A meno che la view sia frutto di una funzione aggregata
- Se usi l'algoritmo 'Temptable' non è aggiornabile

View

```
Create view <nome vista> as  
  select <nomi attributi>  
    from <nome tabella>  
  where <condizioni>;
```

View esempi

Create view emp10 as

```
select empno, ename, job from emp  
where deptno = 10;
```

Select * from emp10;

ATTENZIONE ALLE MODIFICHE !!!!!

View in MySql

PREVISTE SOLO DALLA 5.0 in poi...

View o Tabelle Virtuali

- Vantaggi
 - non occupano memoria
 - sicurezza (si possono non far vedere certi campi)
 - convenienza : si creano view anche per eseguire query più semplici, spesso con ottimizzazione dei tempi di risposta

View o Tabelle Virtuali

- Svantaggi
 - update e delete sulle view diventano azioni pericolose per le tabelle.
 - Possibilità di inconsistenza tabelle e DB

L'opzione '**check**' controlla la consistenza del database.

View

opzione 'check'

Es. :

```
create view Oceania as  
select * from Country  
where continent="Australia"  
with check option;
```

L'opzione 'check' non permetterà di aggiornare il campo 'continent'.

Merge Table

- Tabelle MySQL che sono tabelle MyIsam combinate in una singola tabella virtuale
- Contenitore di tabelle definite in modo simile
- ...solo con Merge Storage Engine

Merge Table

Es. :

```
create table Autore_Libro(nome varchar(30), ....)  
Engine=MyISAM;
```

```
create table Autore_Articolo(nome varchar(30), ....)  
Engine=MyISAM;
```

```
insert .....;
```

```
create table mergeAutori(nome varchar(30),.....)  
Engine=Merge union(Autore_Libro, Autore_Articolo)  
insert_method=LAST;
```

Partitioned Table

- Tabelle MySQL con speciali istruzioni che indicano dove fisicamente sono memorizzate le righe.
- Basate su una Partitioning Function
- ...ancora sperimentale sulla versione 5.1

Partitioned Table

Es. :

```
alter table Vendite  
partition by range(year(day))(  
    partition p_2008 values less than(2009),  
    partition p_2009 values less than(2010),  
    partition p_catchall values less than maxvalue);
```

Sicurezza

L. Vigliano

Sicurezza in MySQL

- Sicurezza di tipo non standard
- MySQL si basa su un sistema di privilegi e permessi
- MySQL controlla i privilegi prima di ogni accesso a qualsiasi oggetto
- Privilegi globali = shutdown, show, flush,...
- Privilegi su un oggetto = DB, table,....

Sicurezza in MySQL

Sistema di privilegi e permessi

- **Autenticazione**
 - **Autorizzazione**
 - **Controllo di accesso**
- Chi sei ?
 - Cosa sei autorizzato a fare ?
 - Quali dati puoi vedere o gestire ?

Sicurezza in Mysql

Sistema di privilegi e permessi

- I privilegi sono garantiti dalle **Grant Tables**
- I comandi principali sono **Grant, Revoke e Drop user**

Sicurezza in Mysql

Grant Tables

I privilegi sono controllati gerarchicamente nel seguente ordine :

- User
- Db
- Host
- Table_priv
- Columns_priv
- Procs_priv

Sicurezza in Mysql

Grant

Grant tipo privilegio
on oggetto(DB,tabelle)
to nome_utente
identified by password;

Dove tipo privilegio può essere :

Select, Insert, Update, Delete, Alter, Index

Sicurezza in Mysql

Grant (esempio)

```
Grant select  
on DB10.emp  
to DB10;
```

```
Grant all privileges  
on *.*  
to DB...;
```

Sicurezza in Mysql

Revoke

Revoke tipo privilegio
on oggetto
from nome utente;

Esempio:

Revoke all privileges
on DB10.*
from DB1@localhost;

Sicurezza in Mysql

Drop user

Drop user nome utente;

Sicurezza in Mysql

Esempi per figure aziendali

System Administrator

```
Grant all privileges on *.* to 'root' @'localhost'  
identified by 'password' with grant option;
```

Database Administrator

```
Grant all privileges on *.* to 'john' @'localhost'  
identified by 'p4ssword' with grant option;
```

Pre-employee

```
Grant insert,update privileges on magazzino.orders  
to 'nino' @'magazzino.example.com' identified by  
'password' ;
```

L. Vigliano

Sicurezza in Mysql

In generale.....

- E' meglio costruire delle view con solo le colonne che possono essere viste e dare accesso a queste, piuttosto che dare accesso alle singole colonne delle tabelle.
- Troppi privilegi, o troppi privilegi a grana fine, o troppi privilegi separati sulle colonne possono creare problemi di performance.

Sicurezza in Mysql

Crittografia

- MySQL supporta la tecnologia **SSL** (Secure Socket Layer) tramite la libreria OpenSSL.
- MySQL mette a disposizione funzioni per criptare dati : **encrypt()**, **sha1()** e **MD5()**.