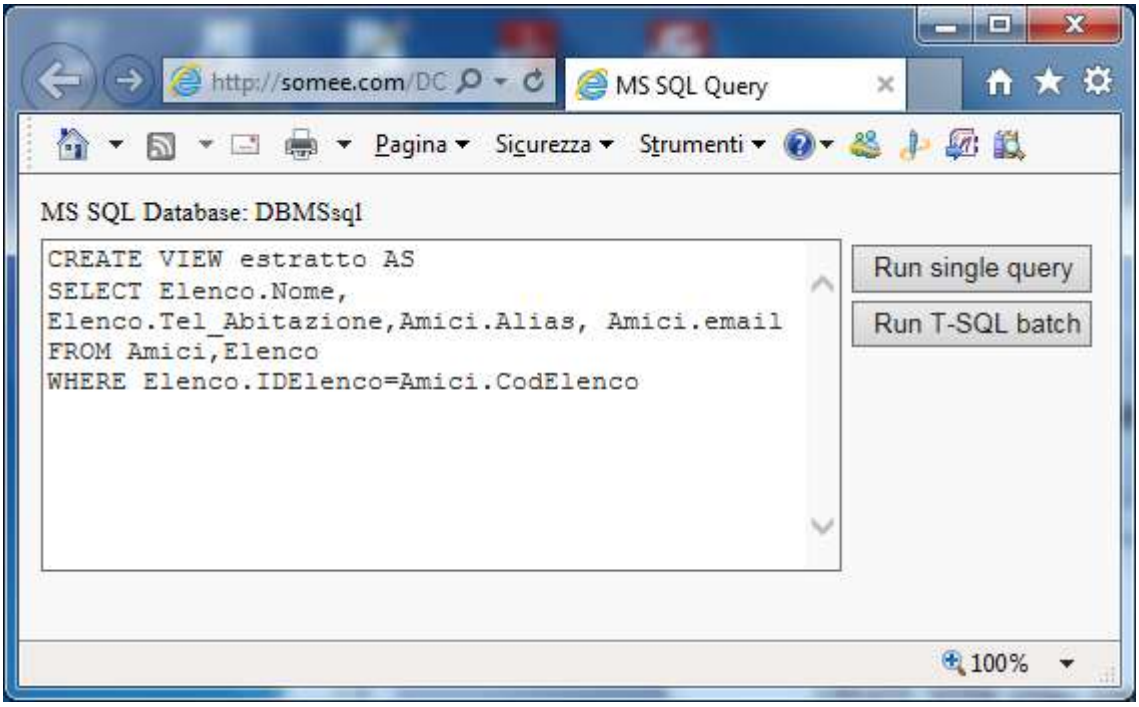
Il concetto di “vista” in una base di dati

Tra i livelli di astrazione di una base di dati c’è il **livello esterno**, o **vista**: quello più vicino all'utente 'finale' del database, che corrisponde al **modo di vedere i dati** da parte dell'utente stesso.

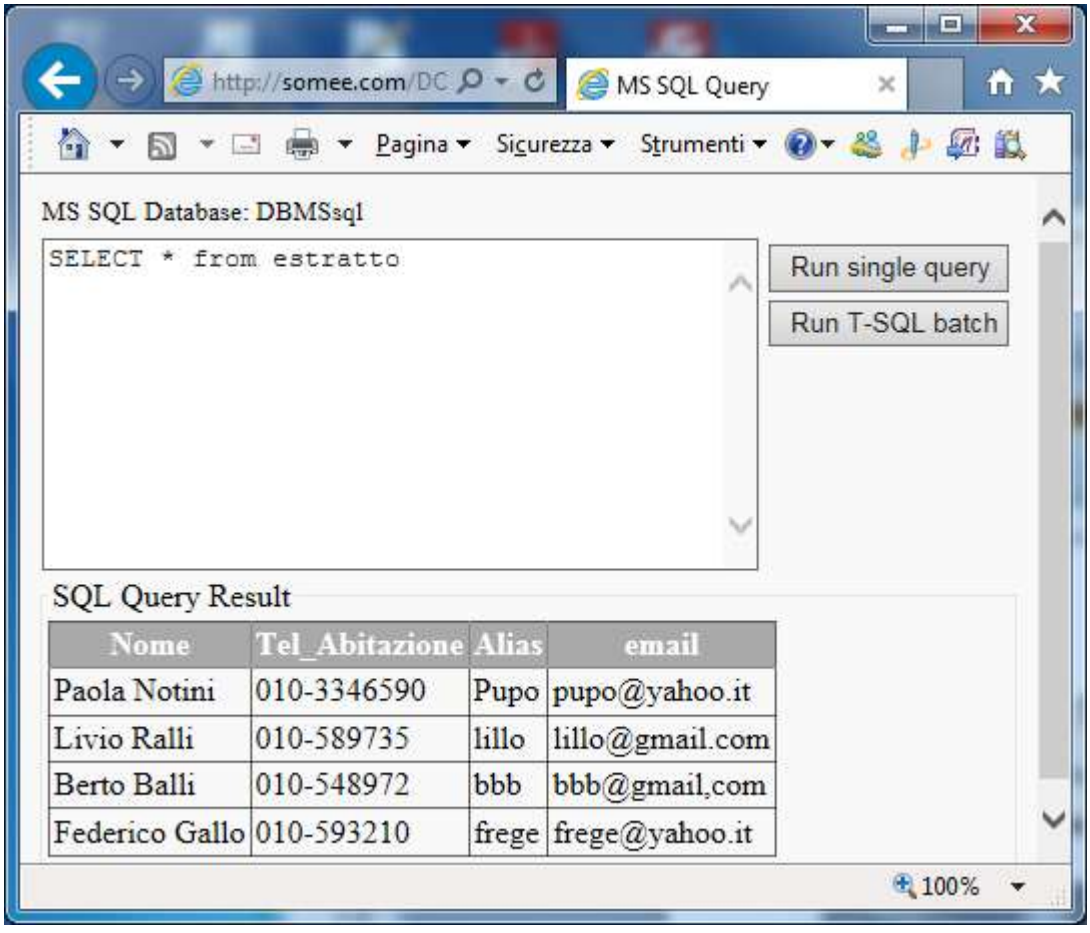
Per creare una **VIEW**

```
CREATE VIEW estratto AS
SELECT Elenco.Nome, Elenco.Tel_Abitazione,Amici.Alias, Amici.email
FROM Amici,Elenco
WHERE Elenco.IDElenco=Amici.CodElenco
```

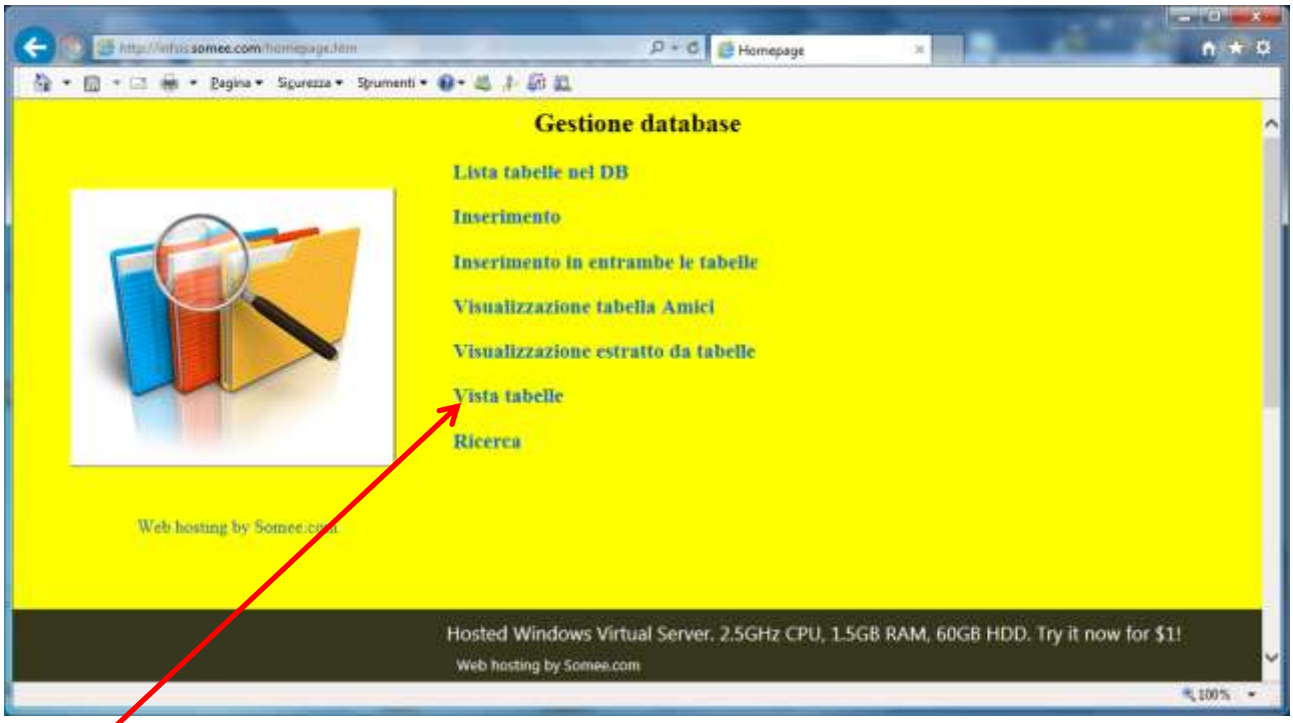


Per visualizzare la VISTA precedente con uso di Box-Editor Somee

query “estratto” memorizzata in modo permanente ma non salvata come tabella



Oppure con pagina ASP *lanciata in esecuzione* da form (*selezionabile da homepage*)

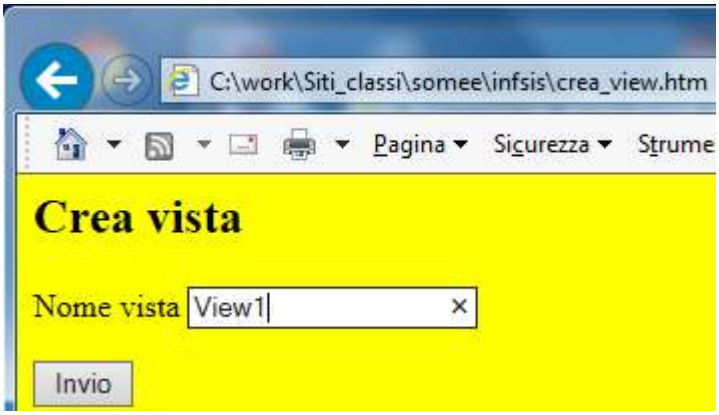


http://infsis.somee.com/view.htm *visualizza* ***view*** *già creata*

crea nuova ***view*** *con nome digitato*
e la visualizza

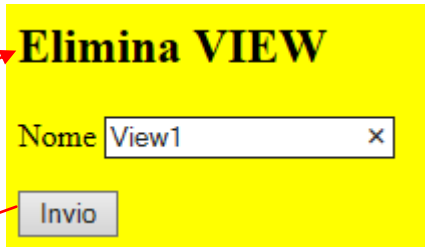
Nominativo	e-mail
Paola Notini	pupo@yahoo.it
Livio Ralli	lillo@gmail.com
Berto Balli	bbb@gmail.com
Federico Gallo	frege@yahoo.it

home



Nb: *si noti la tipica querystring*
nella barra dellURL: http://infsis.somee.com/crea_view.asp?Nome=View1

Per eliminare una ***view*** *con nome digitato*



Eliminazione realizzata

home

rem recupero **nome** digitato
nome = request.querystring("Nome")

rem sintassi per **eliminare** view
sSQL ="**Drop VIEW** "& nome

da altra[1] *home*

[1] Se nome composto con spazi bianchi, nella sintassi si usano [] es: CREATE VIEW [Amici attuali] AS *query_tipo_SELECT*

Le **viste** sono un elemento utilizzato dalla maggior parte dei DBMS.

Una vista è rappresentata da una query (SELECT), il cui risultato può essere utilizzato come se fosse una tabella. Permettono di analizzare, semplificare e personalizzare la visualizzazione del database per ogni utente.

Da un punto di vista fisico esistono diversi modi di fare questo.

Generalmente i DBMS rielaborano le query sulle viste in modo che agiscano sulle tabelle che fanno parte della vista stessa. Ad esempio possono farlo trattando la SELECT che compone la vista come se fosse una *subquery* delle query eseguite su di essa (questo è il modo più semplice, ma essendo poco prestante i DBMS dovrebbero farlo solo in casi particolari).

Una vista può essere composta da una o più tabelle; alcuni DBMS, come MySQL, consentono anche di basare la vista su un'espressione SQL che non coinvolge alcuna tabella (es: SELECT 1+1 AS somma).

Glossario

DEFINIZIONE
È una **relazione derivata**. Si specifica l'espressione che genera il suo contenuto. Esso quindi dipende dalle relazioni che compaiono nell'espressione.

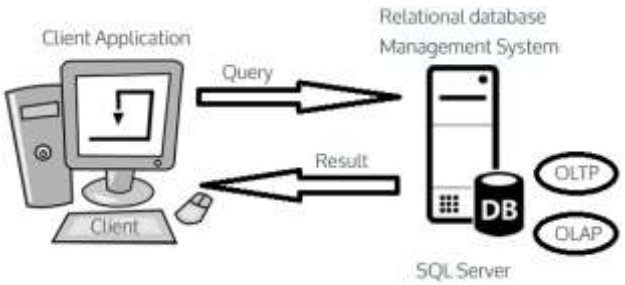
OSSERVAZIONE
Una vista può essere funzione di altre viste purché esista un ordinamento in grado di guidare il calcolo delle relazioni derivate.

Le viste possono essere **aggiornabili**, cioè è possibile eseguire su di esse comandi DML come INSERT, UPDATE e DELETE.
Non tutti i DBMS supportano questa possibilità. Poiché una vista non è altro che un'interfaccia su una o più tabelle, questi comandi andranno a modificare le tabelle sottostanti. Non tutte le viste sono aggiornabili e scrivibili. Si parla di viste "*updatable*" (sulle quali cioè si può eseguire UPDATE e DELETE) solo quando il DBMS è in grado di stabilire una mappatura inversa tra i record presenti nella vista e quelli nelle tabelle. Si parla di viste "*insertable*" (sulle quali si può eseguire INSERT) quando il DBMS è in grado di inserire il record nella tabella corretta. Ad esempio non è "*insertable*" una vista che mostra i valori massimi di una certa tabella o che raggruppa i record con una clausola GROUP BY, perché tali dati non sono fisicamente scritti su una qualche tabella, ma rielaborati tramite una query. Se anche tutti i dati contenuti nella vista sono scritti fisicamente nelle tabelle, qualora la vista coinvolga diverse tabelle è necessario che tra esse vi sia una relazione uno a uno. Inoltre generalmente non è modificabile una vista basata su una UNION.

VISTA MATERIALIZZATA
Una vista si dice "materializzata" se viene calcolata e memorizzata esplicitamente nella base di dati.

Conviene materializzare quando:

- L'aggiornamento è raro
- L'interrogazione che genera la vista è complessa



Se la vista viene materializzate è necessario ricalcolarne il contenuto ogni volta che le relazioni di cui dipende vengono aggiornate. (I dati ivi contenuti vengono aggiornati automaticamente a intervalli regolari dal DBMS. Usate di solito per applicazioni di datawarehousing.)

VISTA VIRTUALE
Una vista si dice “virtuale” se viene calcolata ogni volta che serve.

Conviene quando:

- L’aggiornamento è frequente
- L’interrogazione che genera la vista è semplice

UTILITÀ DELLE VISTE:

- Consentono di **personalizzare l’interfaccia utente** (e semplificare le query)
- Facilitano la gestione della **privatezza** dei dati e potenziano la gestione dei permessi. (Ad esempio si potrebbe creare una query che legge solo alcuni dati da una tabella - tramite la clausola WHERE- per poi assegnare il permesso in lettura ad un certo utente sulla vista, ma non sulla tabella di base. In questo modo l'utente non vedrà i dati che non vengono estratti dalla vista.)
- Permettono di **memorizzare** nel DBMS **interrogazioni complesse condivise**
- Sono utili per rendere l’interfaccia delle applicazioni indipendente dallo schema logico (**INDIPENDENZA LOGICA**) e compatibile con versioni precedenti con cui emulare una tabella il cui schema è stato modificato

ALTERNATIVE ALLE VISTE

Le viste servono a risolvere problemi che, spesso, possono essere risolti anche in altri modi.

Colonne virtuali e indici funzionali

A volte si creano delle viste solo per **aggiungere** a una tabella **una colonna**, calcolata sulle altre, senza che questa sia registrata fisicamente nel database. Le query sulla vista potranno quindi leggere la colonna aggiuntiva senza dover includere la formula necessaria per calcolarne il valore.

> Se il DBMS supporta le colonne virtuali (SQL Server, Oracle...) o gli indici funzionali (PostgreSQL), è sufficiente aggiungere alla tabella uno di questi oggetti.

Trigger

A volte le viste vengono usate oltre che per aggiungere una colonna anche per **modificare un valore al momento della visualizzazione** (per esempio, rendendo tutti i caratteri di un campo minuscoli). Un altro modo per ottenere lo stesso risultato sono i trigger. Usando i trigger è possibile scrivere su disco il valore desiderato, rendendo più veloce la lettura (ma occupando più spazio).

Tabelle riassuntive

A volte si desidera **raggruppare i dati** non solo logicamente, ma anche **fisicamente**. Una tecnica consiste nel creare delle tabelle *denormalizzate*, contenenti dati ricavati da altre tabelle. Queste tabelle riassuntive devono essere aggiornate periodicamente, preferibilmente in un momento in cui il database non viene utilizzato (di solito di notte). Questa tecnica permette prestazioni migliori nelle query di tipo OLAP (**O**n-**L**ine **A**nalytical **P**rocessing - componente tecnologica base del data warehouse: insieme di tecniche software per l'analisi interattiva e veloce di grandi quantità di dati con l’ obiettivo di performance nella ricerca e il raggiungimento di interrogazioni quanto più articolate sia possibile. Ad esempio strumenti che servono alle aziende per analizzare i risultati delle vendite, l'andamento dei costi di acquisto merci, al marketing per misurare il successo di una campagna pubblicitaria, a una università per organizzare i dati di un sondaggio ed altri casi simili)