Laboratorio di Basi di Dati e Web

Docente: Alberto Belussi

Lezione 3

Interrogazioni SQL

Le interrogazioni SQL hanno la seguente struttura:

```
SELECT <Target List>
FROM <Tables list>
[WHERE <Conditions list>]
```

L'esecuzione dell'interrogazione produce il prodotto cartesiano delle tabelle elencate nella clausola **FROM**, seleziona solo le tuple che soddisfano le condizioni della clausola **WHERE** e per ogni tupla selezionata proietta sugli attributi della clausola **SELECT.**

Sintassi

SELECT AttrEspr [[AS] Alias]
{, AttrEspr [[AS] Alias] }
FROM Tabella [[AS] Alias]
{, Tabella [[AS] Alias] }
[WHERE Condizione]

SQL: Clausola SELECT

- Il comando base del linguaggio SQL:

Lista Attributi di cui si vuole conoscere il valore (Target List)

```
SELECT Attributo {,Attributo}
FROM Tabella {,Tabella}
[WHERE Condizione]
```

Base di Dati usata negli esempi

Studente

Matricola	Nome	Indirizzo	Città	CAP	Sesso

Insegnamento

Codice	Nome_ins	Numero_crediti

Docente

Matricola	Nome	Telefono	Stipendio

Selezionare tutti gli attributi di una tabella

Nella target list può apparire il carattere speciale *, che rappresenta un'abbreviazione della lista di tutti gli attributi delle tabelle indicate nella clausola FROM

```
SELECT *
FROM Tabella
```

Esempio 1

- Visualizzare tutto il contenuto della tabella Insegnamento

```
SELECT *
FROM Insegnamento;
```

Codice	Nome_ins	Numero_crediti
INF01	Lab Basi Dati	2
INF02	Analisi I	3
INF03	Fisica I	3

Selezionare solo alcuni attributi di una tabella

Nella target list può apparire la lista esplicita degli attributi di cui si vuole conoscere il valore

```
SELECT Attributo1, Attributo2
FROM Tabella
```

Esempio 2

- Visualizzare la matricola e il nome di tutti gli studenti

```
SELECT Matricola, Nome
FROM Studente;
```

Matricola	Nome	Indirizzo	Città	CAP	Sesso

Matricola	Nome
IN000001	Rossi Marco
IN000002	Verdi Paolo
IN000003	Bianchi Luca

Espressioni generiche nella Target List

Nella target list possono comparire espressioni generiche applicate ai valori degli attributi di ciascuna tupla selezionata:

```
SELECT AttrExpr AS
       AliasAttributo
FROM Tabella
WHERE Condizione
```

Esempio 3

- Visualizzare il nome e lo stipendio settimanale degli insegnanti

```
SELECT Nome, Stipendio/4 AS StipSettimanale
FROM Docente;
```

Matricola	Nome	Telefono	Stipendio
DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000

Nome	StipSettimanale
A. A.	625
B. B.	475
C. C.	750

SQL: Clausola WHERE

- Nella clausola WHERE viene specificata la condizione che le tuple della tabella risultato devono soddisfare:

```
SELECT ListaAttributi
FROM Tabella
WHERE Condizione
```

- Condizioni atomiche della clausola WHERE:

WHERE Nome Colonna θ Valore

= <> > < >= >=

Dipende dal Tipo di Colonna

Esempio 4

- Visualizzare il codice e il nome degli insegnamenti da 3 crediti

```
SELECT Codice, Nome_ins
FROM Insegnamento
WHERE Numero_crediti = 3;
```

Codice	Nome_ins	Numero_crediti
INF01	Lab Basi Dati	2
INF02	Analisi I	3
INF03	Fisica I	3

Codice	Nome_ins
INF02	Analisi I
INF03	Fisica I

Esempio 5

- Visualizzare il nome e lo stipendio dei docenti che guadagnano più di 2000 euro

```
SELECT Nome, Stipendio
FROM Docente
WHERE Stipendio > 2000;
```

Matricola	Nome	Telefono	Stipendio
DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000

Nome	Stipendio
A. A.	2500
C. C.	3000

Combinare più predicati: AND, OR, NOT

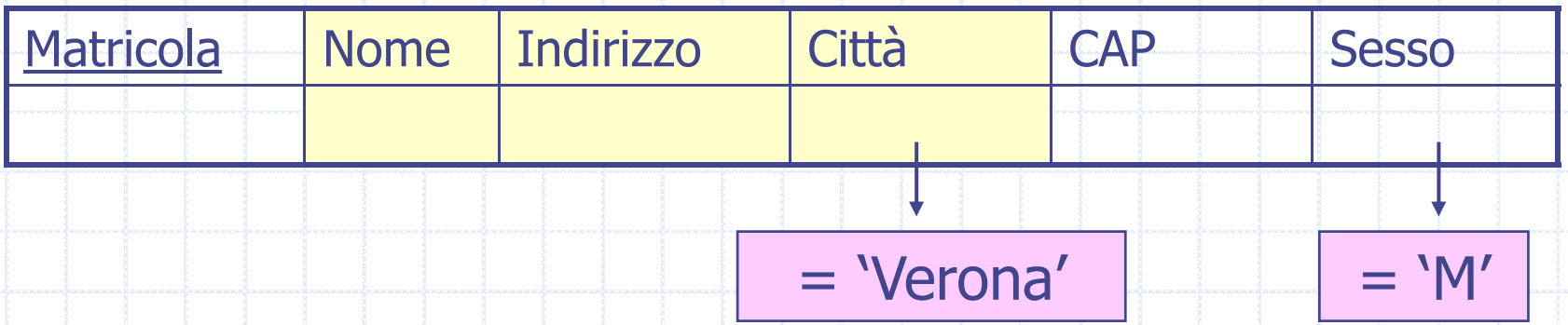
La condizione di selezione della clausola WHERE può essere composta da più predicati combinati mediante i connettivi logici AND, OR, NOT.

```
SELECT ListaAttributi
FROM Tabella
WHERE Condizione1
OPER_LOGICO Condizione2
```

Esempio 6

- Visualizzare il nome, l'indirizzo e la città di tutti gli studenti maschi che abitano a Verona

```
SELECT Nome, Indirizzo, Città
FROM Studente
WHERE Città= 'Verona'
AND Sesso= 'M';
```



Esempio 6: risultato

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Nome	Indirizzo	Città
Rossi Marco	Via X	Verona
Verdi Paolo	Via Y	Verona
Bianchi Luca	Via Z	Verona

Esempio 7

- Visualizzare il nome, l'indirizzo e la città di tutti gli studenti maschi che abitano a Verona o a Padova

```
SELECT Nome, Indirizzo, Città
FROM Studente
WHERE (Città= 'Verona' OR Città= 'Padova')
AND Sesso= 'M';
```

Matricola	Nome	Indirizzo	Città	CAP	Sesso

= 'Verona' o 'Padova'

= 'M'

Esempio 7: risultato

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Nome	Indirizzo	Città
Rossi Marco	Via X	Verona
Verdi Paolo	Via Y	Verona
Gialli Mario	Via K	Padova
Bianchi Luca	Via Z	Verona

Esempio 8

- Visualizzare matricola e nome di tutti gli studenti che non abitano a Verona

```
SELECT Matricola, Nome
FROM Studente
WHERE NOT (Città = 'Verona' )
```

Matricola	Nome	Indirizzo	Città	CAP	Sesso

NOT = 'Verona'

Esempio 8: risultato

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Matricola	Nome
VR0004	Gialli Mario

Operatore LIKE (e NOT LIKE)

Nella clausola WHERE può apparire l'operatore LIKE per il confronto di stringhe. LIKE si comporta come un operatore di "pattern matching" e consente di specificare pattern usando i caratteri speciali "_" e "%".

- "_" rappresenta un carattere arbitrario
- "%" rappresenta un numero arbitrario (anche 0) di caratteri

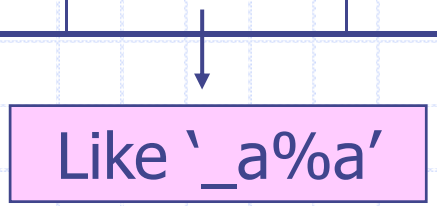
```
SELECT ListaAttributi
FROM Tabella
WHERE Attributo1 LIKE '...'
```

Esempio 9

- Visualizzare la matricola e il nome di tutti gli studenti che abitano in una città che ha una 'a' in seconda posizione e finisce per 'a'

```
SELECT Matricola, Nome
FROM Studente
WHERE Città LIKE '_a%a'
```

Matricola	Nome	Indirizzo	Città	CAP	Sesso



Esempio 9: risultato

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Matricola	Nome
VR0004	Gialli Mario

Esempio 10

- Visualizzare la matricola e il nome di tutti gli studenti che abitano in una città che non inizia per 'P'

```
SELECT Matricola, Nome
FROM Studente
WHERE Città NOT LIKE 'P%'
```

Matricola	Nome	Indirizzo	Città	CAP	Sesso

NOT LIKE `P%'

Esempio 10: risultato

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Matricola	Nome
VR0001	Rossi Marco
VR0002	Neri Maria
VR0003	Verdi Paolo
VR0005	Bianchi Luca

BETWEEN e NOT BETWEEN

- Visualizzare il nome e lo stipendio degli insegnanti che guadagnano tra i 2000 e i 3000 euro

```
SELECT Nome, Stipendio
FROM Docente
WHERE Stipendio BETWEEN 2000 AND 3000
```

Matricola	Nome	Telefono	Stipendio
DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000

Nome	Stipendio
A. A.	2500
C. C.	3000

IN e NOT IN

- Un altro modo per selezionare le righe che si vogliono considerare

```
SELECT NomeAttributo
FROM Tabella
WHERE NomeAttributo IN [NOT IN] (Valori)
```

Valori separati da ,

- Visualizzare la matricola, il nome e la città degli studenti che vivono a Verona, Venezia o Padova.

```
SELECT Matricola, Nome, Città
FROM Studente
WHERE Città IN ('Verona', 'Venezia', 'Padova');
```

IS NULL e IS NOT NULL

- Per selezionare le righe che hanno un attributo NULL (o NOT NULL)

```
SELECT NomeAttributo
FROM Tabella
WHERE NomeAttributo IS NULL (IS NOT NULL)
```

- Visualizzare tutte le informazioni degli insegnamenti che hanno un valore NULL per il numero di crediti

```
SELECT *
FROM Insegnamento
WHERE Numero_crediti IS NULL
```

Per visualizzare i dati stabilendo un ordine

- SQL permette di specificare un eventuale ordinamento delle righe del risultato di una interrogazione. Tale ordinamento è specificato tramite la clausola ORDER BY, con la quale si chiude l'interrogazione.

Default

ORDER BY AttrDiOrdinamento [DESC | ASC]
{ ,AttriDiOrdinamento [DESC | ASC] }

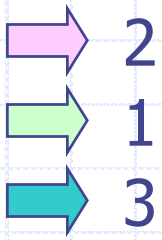
Esempio 11

- Visualizzare, ordinati per stipendio, il nome e il numero di telefono dei docenti

```
SELECT Nome, Telefono
FROM Docente
ORDER BY Stipendio
```

Esempio 11: risultato

Matricola	Nome	Telefono	Stipendio
DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000



Nome	Telefono
B. B.	045 …
A. A.	045 …
C. C.	045 …

Esempio 12

- Visualizzare in ordine alfabetico il nome degli studenti che abitano a Verona

```
SELECT Nome
FROM Studente
WHERE Città='Verona'
ORDER BY Nome
```

Nome
Bianchi Luca
Neri Maria
Rossi Marco
Verdi Paolo

Esempio 12

- Visualizzare il nome e la città degli studenti Maschi ordinati per città e per nome

```
SELECT Nome, Città
FROM Studente
WHERE Sesso = 'M'
ORDER BY Città, Nome
```

Nome	Città
Gialli Mario	Padova
Bianchi Luca	Verona
Rossi Marco	Verona
Verdi Paolo	Verona

Operatori di Aggregazione

- Vengono applicati ad un insieme di tuple.
- Prima viene normalmente eseguita l'interrogazione, considerando solo le parti FROM e WHERE. L'operatore aggregato viene poi applicato alla tabella contenente il risultato dell'interrogazione.
- Due gruppi:
  - COUNT
  - MAX, MIN, AVG, SUM
- Quando come argomento della SELECT compaiono delle funzioni aggregate, allora non possono comparire espressioni che usano i valori presenti nelle tuple singole. Ad esempio se conto le tuple con COUNT, non è poi possibile riportare nella SELECT il valore di un attributo insieme al risultato del COUNT.

COUNT

- Permette di contare il numero di tuple

COUNT (<* | [DISTINCT | ALL] ListaAttributi>)

Restituisce il numero di righe

Restituisce il numero di combinazioni **diverse** considerando i valori degli attributi in ListaAttributi

Restituisce il numero di righe che possiedono valori diversi da NULL per gli attributi in ListaAttributi

Esempio 13

- Quanti insegnamenti ci sono nella tabella Insegnamento?

```
SELECT COUNT(*)
FROM Insegnamento
```

Considera i valori NULL

```
SELECT COUNT(Codice)
FROM Insegnamento
```

Codice	Nome_ins	Numero_crediti
INF01	Lab Basi Dati	2
INF02	Analisi I	3
INF03	Fisica I	3

COUNT(*)
3

Esempio 14

- Da quante città diverse provengono gli studenti?

```
SELECT COUNT(DISTINCT Città)
FROM Studente
```

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

COUNT(DISTINCT Città)
2

Esempio 15

- Visualizzare le città di provenienza degli studenti

```
SELECT DISTINCT Città
FROM Studente
```

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Città
Padova
Verona

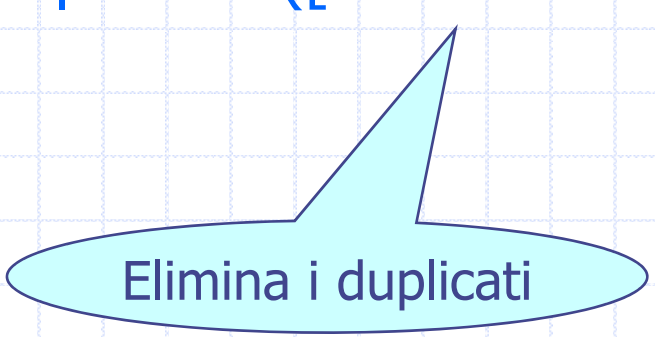
SUM, MAX, MIN, AVG

- SUM restituisce la somma dei valori posseduti dall'attributo su tutte le righe
- MAX e MIN restituiscono rispettivamente il massimo e il minimo valore tra quelli di ciascuna riga (DISTINCT e ALL non hanno effetto)
- AVG restituisce la media dei valori dell'attributo

SUM, MAX, MIN, AVG

- Ammettono come argomento un attributo o un'espressione, eventualmente preceduta dalla parola chiave DISTINCT o ALL.

< SUM | MAX | MIN | AVG >([DISTINCT | ALL] AttrEspr)



Esempio 16

- Trovare lo stipendio massimo degli insegnanti

```
SELECT MAX(Stipendio)
FROM Docente
```

Matricola	Nome	Telefono	Stipendio
DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000

MAX(Stipendio)
3000

Esempio 17

- Trovare lo stipendio minimo degli insegnanti

```
SELECT MIN(Stipendio) AS StipMinimo
FROM Docente
```

Matricola	Nome	Telefono	Stipendio
DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000

StipMinimo
1900

Esempio 18

- Trovare la somma degli stipendi di tutti gli insegnanti che guadagnano più di 2000 euro

```
SELECT SUM(Stipendio)
FROM Docente
WHERE Stipendio > 2000
```

Matricola	Nome	Telefono	Stipendio
DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000

SUM(Stipendio)
5500

Esempio 19

- Trovare la media degli stipendi di tutti gli insegnanti che guadagnano più di 2000 euro

```
SELECT AVG(Stipendio)
FROM Docente
WHERE Stipendio > 2000
```

DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000

AVG(Stipendio)
2750

Interrogazioni con raggruppamento

- Molto spesso sorge l'esigenza di applicare l'operatore aggregato a sottoinsiemi di tuple.
- La clausola GROUP BY consente di specificare come dividere le tabelle in sottoinsiemi (gruppi), raggruppando le tuple che possiedono gli stessi valori in un insieme di attributi assegnato.
- SQL impone che in una interrogazione che fa uso della GROUP BY, possano comparire come argomento della SELECT solamente un sottoinsieme degli attributi utilizzati per il raggruppamento delle righe insieme a funzioni aggregate valutate sugli altri attributi.

GROUP BY

- Utilizzato per organizzare i dati in "gruppi"
- Gli operatori di aggregazione vengono computati sui diversi gruppi

```
SELECT ListaAttributi
FROM Tabella
WHERE Condizione
GROUP BY Attributo1
```

Esempio 20

- Visualizzare il numero di studenti che provengono da città diverse (visualizzando anche la città)

```
SELECT Città, COUNT(*)
FROM Studente
GROUP BY Città
```

Esempio 20: risultato

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Città	COUNT(*)
Verona	4
Padova	1

Esempio 21

- Visualizzare il numero di studenti maschi e femmine

```
SELECT Sesso, COUNT(*)
FROM Studente
GROUP BY Sesso
```

Esempio 21: risultato

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Sesso	COUNT(*)
M	4
F	1

PREDICATI SU GRUPPI

- La clausola HAVING consente di descrivere le condizioni che si devono applicare al termine dell'esecuzione di una interrogazione che fa uso della GROUP BY per selezionare i gruppi che andranno nel risultato dell'interrogazione.
- Ogni gruppo G costruito dalla GROUP BY fa parte del risultato dell'interrogazione solo se G soddisfa il predicato della clausola HAVING.

PREDICATI SU GRUPPI: sintassi

```
SELECT Attributo1
FROM Tabella
[WHERE Condizione]
GROUP BY Attributo1
HAVING Predicato
ORDER BY Attributo1
```

Esempio 22

- Visualizzare le città in cui abitano almeno 2 studenti e il numero di studenti relativo

```
SELECT Città, COUNT(*)
FROM Studente
GROUP BY Città
HAVING COUNT(*)>=2
```

Esempio 22: risultato

Matricola	Nome	Indirizzo	Città	CAP	Sesso
VR0001	Rossi Marco	Via X	Verona	37129	M
VR0002	Neri Maria	Via W	Verona	37132	F
VR0003	Verdi Paolo	Via Y	Verona	37121	M
VR0004	Gialli Mario	Via K	Padova	52100	M
VR0005	Bianchi Luca	Via Z	Verona	37135	M

Città	COUNT(*)
Verona	4

Sintassi: sommario

SELECT ListaAttributiOEspressioni

FROM ListaTabelle

[WHERE CondizioniSemplici]

[GROUP BY ListaAttributiDiRaggruppamento]

[HAVING CondizioniAggregate]

[ORDER BY ListaAttributiDiOrdinamento]

Join

- Per selezionare informazioni da due o più tabelle
- I nomi degli attributi devono essere specificati in modo non ambiguo (NomeTabella.NomeAttributo)

```
SELECT ListaAttributi
FROM Tabella1, Tabella2
WHERE Tabella1.Attributo1 = Tabella2.Attributo2
```

Base di Dati usata negli esempi

Studente

Matricola	Nome	Indirizzo	Città	CAP	Sesso

Insegnamento

Codice	Nome_ins	Numero_crediti

Docente

Matricola	Nome	Telefono	Stipendio

Esame

Codice_ins	Anno_accademico	Studente	Voto

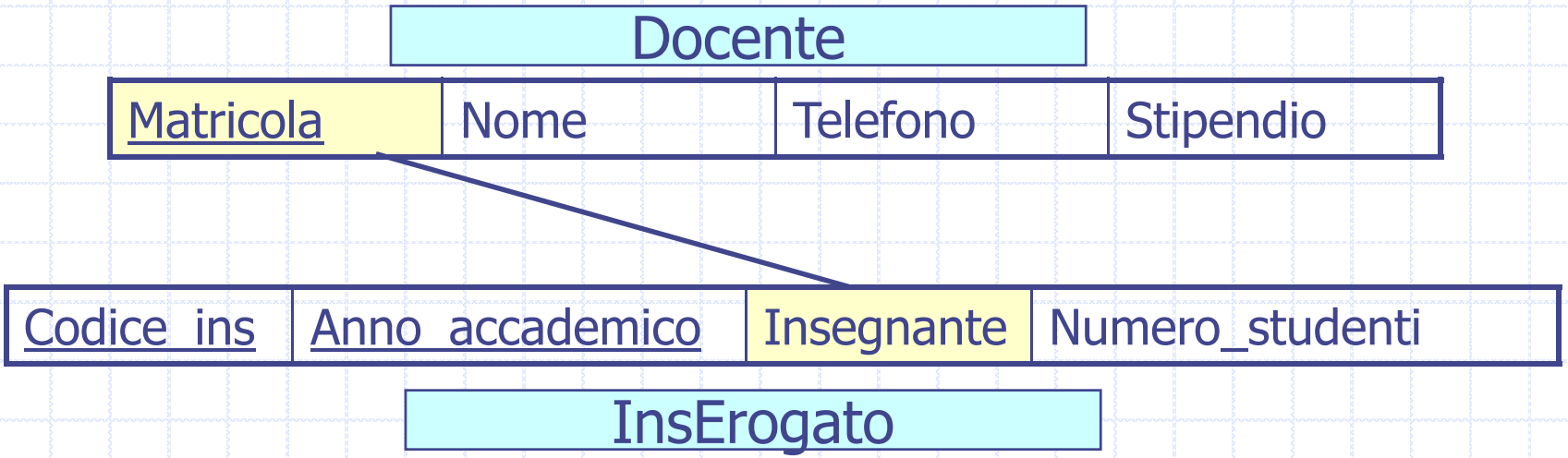
InsErogato

Codice_ins	Anno_accademico	Insegnante	Numero_studenti

Esempio 23

- Trovare gli insegnamenti tenuti da ogni docente, riportando il codice dell'insegnamento e il nome del docente

```
SELECT Docente.Nome AS NomeDocente,
       InsErogato.Codice_ins
FROM Docente, InsErogato
WHERE Docente.Matricola = InsErogato.Insegnante
```



Esempio 23: risultato

Matricola	Nome	Telefono	Stipendio
DIN001	A. A.	045 …	2500
DIN002	B. B.	045 …	1900
DIN003	C. C.	045 …	3000

Codice_ins	Anno_accademico	Insegnante	Numero_studenti
INF01	2005/2006	DIN002	10
INF02	2006/2007	DIN003	80
INF03	2006/2007	DIN001	100

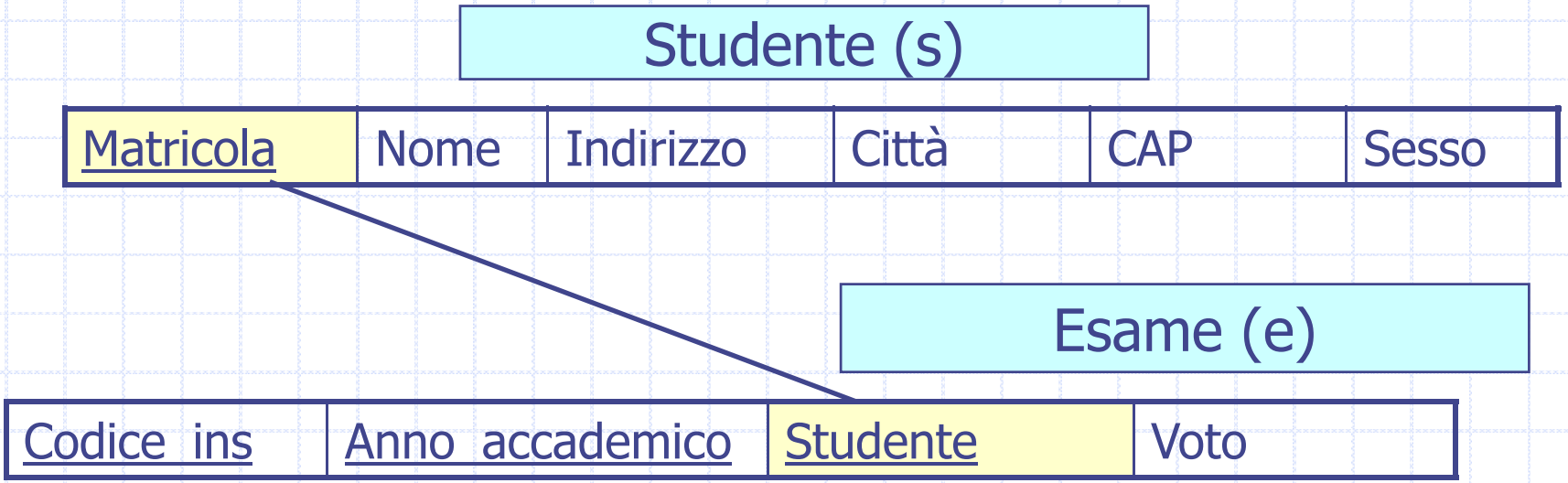
Risultato ➔

NomeDocente	Codice_ins
A. A.	INFO3
B. B.	INF01
C. C.	INF02

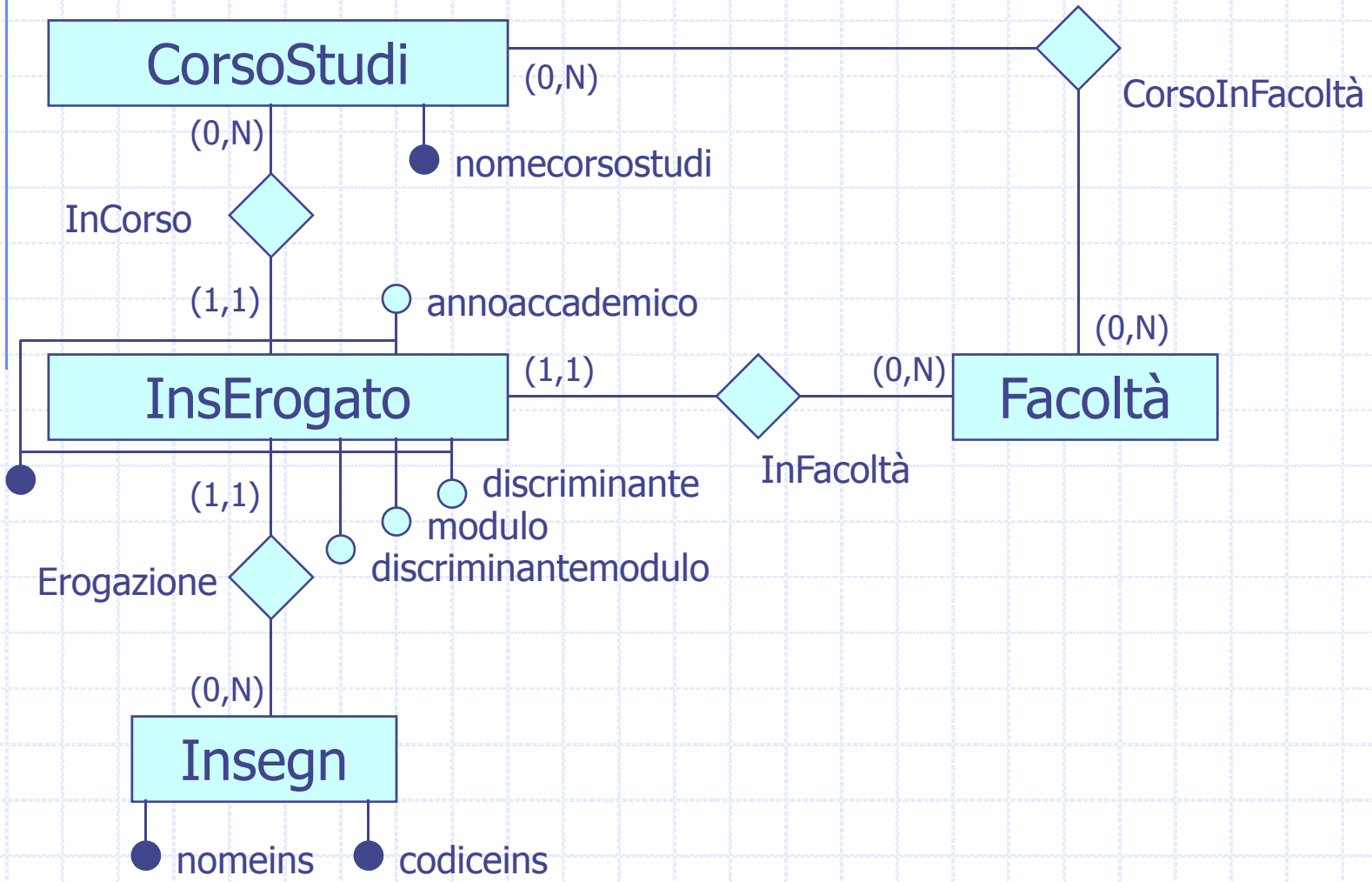
Esempio 25

- Visualizzare i voti presi dagli studenti di Verona

```
SELECT Nome, Codice_ins, Voto
FROM Studente s, Esame e
WHERE s.Matricola = e.Studente AND
      s.Città='Verona'
```



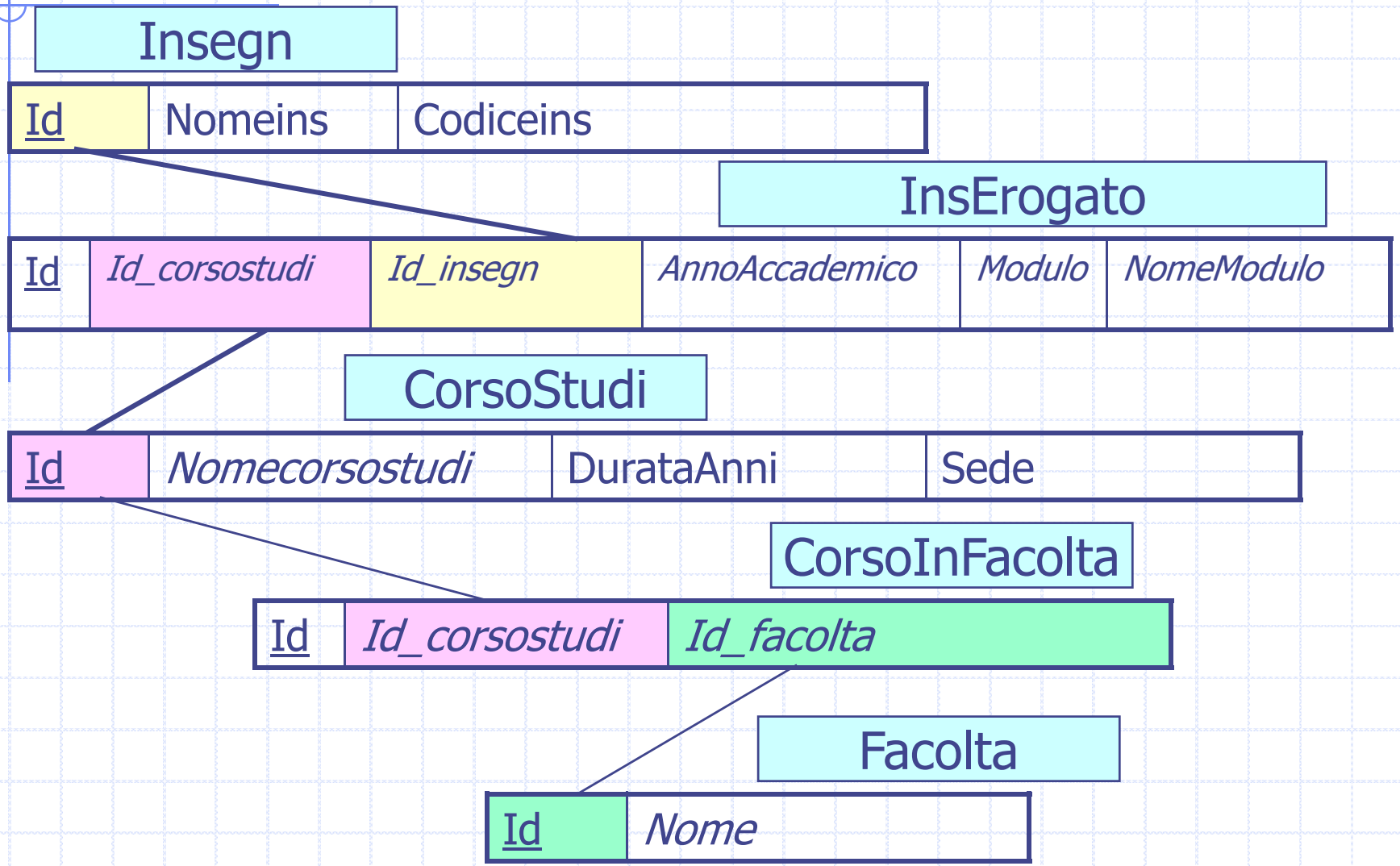
Base di Dati usata negli esercizi (DB dell'applicazione web di ateneo)



DB dell'applicazione web di ateneo

- L'applicazione per alimentare la base di dati è stata generata da uno strumento messo a punto dal Dipartimento di Informatica di Milano (ERW: Entity and Relationships on the Web, informazioni al sito http://erw.dsi.unimi.it/)
- Questo strumento genera in automatica da una descrizione XML dello schema concettuale in ER il codice che genera la base di dati in SQL standard.
- In tale traduzione il sistema sostituisce tutte le chiavi primarie con id numerici.

DB dell'applicazione web di ateneo (in corsivo gli attributi in vincolo UNIQUE)



DB dell'applicazione web di ateneo

Schema relazionale delle relazioni usate in questa esercitazione:

InsErogato(**id**, id_insegn, id_corsostudi, annoaccademico, modulo, nomemodulo*, crediti, programma*, id_facolta)

Insegn(**id**, nomeins, codiceins)

CorsoStudi(**id**, nomecorsostudi, sede, durataAnni)

CorsoInFacolta(**id_corsostudi, id_facolta**)

Facolta(**id**, nome)

Esempio contenuto del DB

Id	Anno accademico	Id_Corsostudi	Id_Insegn	Modulo	Nome modulo	Crediti
2	2006/2007	1	34	0		10
11	2006/2007	1	34	1	Teoria	8
12	2006/2007	1	34	2	Laboratorio	2

Id	NomeCorsostudi	Sede	Durata
1	Informatica	Verona	3
2	Bioinformatica	Verona	3

Id	NomeIns	CodiceIns
34	Basi di dati e Web	SCI-332
98	Programmazione	SCI-123

DB dell'applicazione web di ateneo

psql –h sqlserver –d didattica userlabXX

Consegne

Modalità di consegna:

- Invio di email all'indirizzo: alberto.belussi@univr.it
- Oggetto:

  <Matricola> (<utente laboratorio>) – esercitazione <x>
- Contenuto:

  <Matricola> - <Cognome> <Nome>
- Allegato:

  File di testo con il codice SQL.

Prima consegna:

- File SQL contenente il risultato della seconda esercitazione.
- Scadenza 15 maggio 2009 (ore 23.00).