

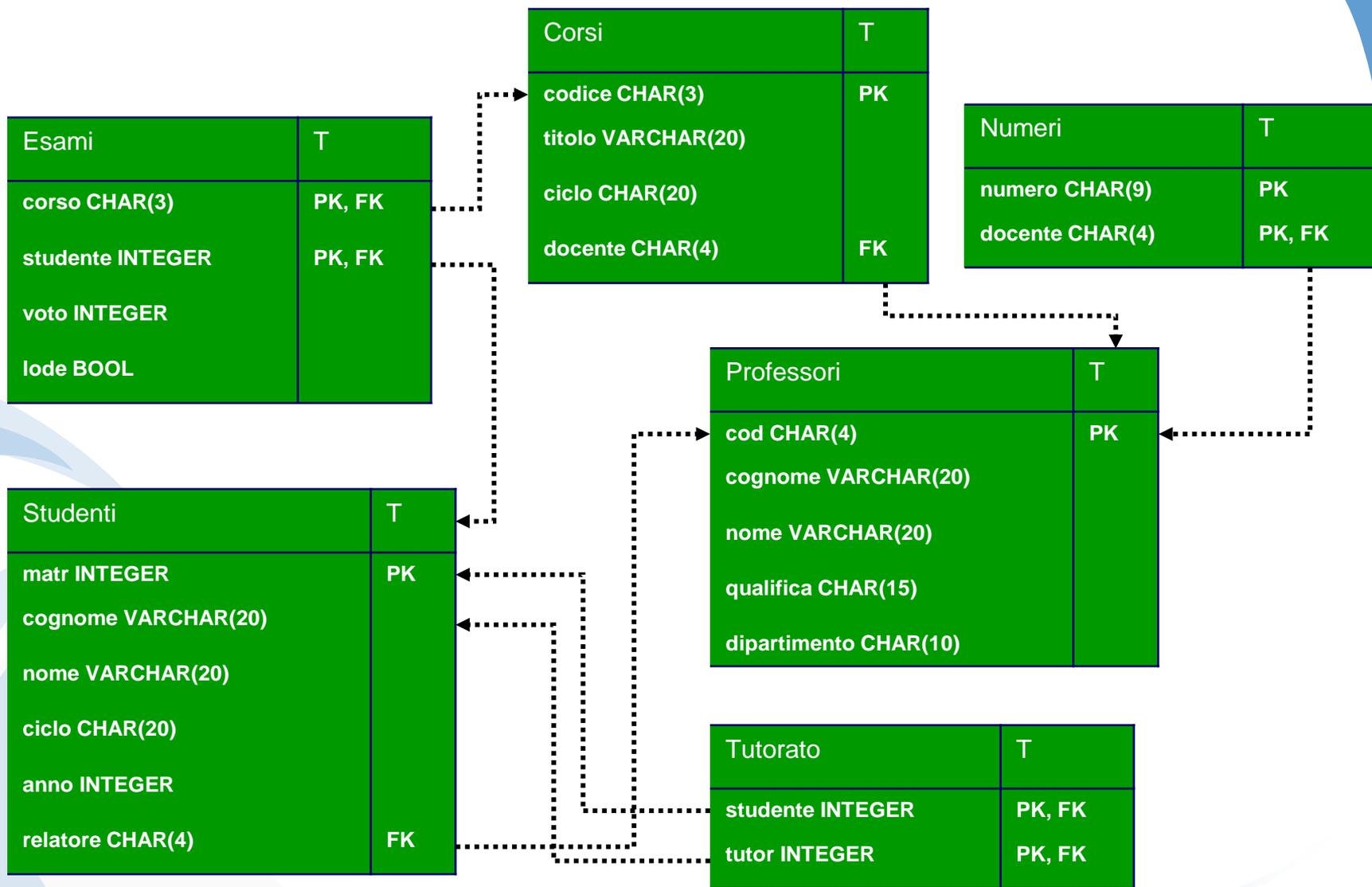
BASE DI DATI

Concetti trattati:

- Funzioni aggregate
- Raggruppamenti
- Nidificazione
- Viste

Informatica Umanistica

Università di Pisa



Professori

<u>cod</u>	cognome	nome	qualifica	Dipartimento
FT	Totti	Francesco	ordinario	Ingegneria
CV	Vieri	Christian	associato	Informatica
ADP	Del Piero	Alessandro	supplente	null

Studenti

<u>matr</u>	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
333	Rossi	Maria	laurea tr.	1	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

Corsi

<u>cod</u>	titolo	ciclo	docente
PR1	Programmazione I	laurea tr.	FT
ASD	Algoritmi e Str. Dati	laurea tr.	CV
INFT	Informatica Teorica	laurea sp.	ADP

Tutorato

<u>studente</u>	<u>tutor</u>
111	77777
222	77777
333	88888
444	88888

Numeri

<u>professore</u>	<u>numero</u>
FT	0971205145
FT	347123456
VC	0971205227
ADP	0971205363
ADP	338123456

Esami

<u>studente</u>	<u>corso</u>	<u>voto</u>	<u>lode</u>
111	PR1	27	false
222	ASD	30	true
111	INFT	24	false
77777	PR1	21	false
77777	ASD	20	false
88888	ASD	28	false
88888	PR1	30	false
88888	INFT	30	true

Concetti avanzati

- Raggruppamenti
 - Clausole GROUP BY e HAVING
 - Forma Generale della SELECT
- Nidificazione
 - Uso nel DML e DDL
 - Nidificazione, Viste e Potere Espressivo
- Esecuzione di una Query SQL

Interrogazioni con Raggruppamenti

- Nucleo della SELECT (già studiate)
 - SELECT, FROM, [WHERE]
- Clausola aggiuntiva (già studiate)
 - [ORDER BY]
- Ulteriori clausole aggiuntive
 - [GROUP BY]
 - [HAVING]

Forma generale della SELECT

- Forma generale della SELECT

```
SELECT [DISTINCT] <lista degli attributi>  
FROM <join o prodotti cartesiani>  
[WHERE <condizioni>]  
[GROUP BY <attributi di raggruppamento>]  
[HAVING <condizioni sui gruppi>]  
[ORDER BY <attributi di ordinamento>]
```

FUNZIONI AGGREGATIVE

Lista degli attributi

```
SELECT <lista degli attributi>  
FROM <join o prodotti cartesiani>
```

...

- Cosa possiamo scrivere nella lista degli attributi?
 - Finora nel risultato abbiamo considerato solo valori presenti nella base di dati iniziale
- In alcuni casi è utile avere valori calcolati
 - somme e medie di attributi numerici;
es: media dei voti degli studenti
 - conteggi;
es: numero di corsi della laurea triennale
 - massimi e minimi
 - espressioni aritmetiche
 -

Espressioni

Nella SELECT possono essere presenti espressioni contenenti:

- Operandi
 - valori degli attributi
- Operatori (non standard)
 - operatori aritmetici +, -, *, % ed altri
 - funzioni matematiche log, exp, sin, ...
 - funzioni su stringhe length, substring, ...
 - funzioni su date e tempi

Espressioni

- Esempio: Visualizzare nome e reddito in lire

Persone

Nome	Eta	Reddito
Andrea	27	30000
Aldo	25	NULL
Maria	55	36000
Anna	50	36000

```
SELECT nome, reddito*1936.27  
FROM persone;
```

Funzioni Aggregative

- Funzione aggregativa
 - argomento: attributo di una tabella
 - calcolata esaminando i valori di un attributo appartenenti ad ennuple diverse
- Tipicamente:
 - SUM (somma), COUNT (conteggio), AVG (media), MIN (minimo), MAX (massimo)

Funzioni Aggregative

- Sintassi
 - si utilizzano nella proiezione
- Esempio:
 - calcolo del voto medio degli esami
 - funzione AVG() applicata all'attributo voto della tabella Esami

Risultato = $\pi_{AVG(voto)}$ (Esami)

Funzioni Aggregative

- Domanda:
 - perché la proiezione?
 - non sarebbe più semplice non includere la proiezione π :

~~Risultato = AVG(voto) (Esami)~~

- Risposta:
 - il risultato non può essere un numero
 - deve essere una tabella (di una ennupla e una colonna)

Funzioni Aggregative

- Semantica:
 - viene calcolato il risultato della proiezione sugli attributi utilizzati come argomenti
 - viene applicata la funzione aggregativa ai valori dell'attributo
 - il risultato è una tabella con una singola ennupla
 - una colonna per ciascuna funzione aggregativa utilizzata nella proiezione

Funzioni Aggregative

- Esempio:

$$\text{Risultato} = \pi_{\text{AVG(voto)}} (\text{Esami})$$

- I passo: $\pi_{\text{voto}} (\text{Esami})$
- Il passo: calcolo della media dei valori dell'attributo voto
- III passo: viene creata la tabella Risultato con una unica colonna (chiamata AVG(voto)) e un'unica ennupla, contenente il risultato

Operatori aggregati: COUNT

COUNT restituisce il numero di righe della tabella o il numero di valori di un particolare attributo

Esempio: Il numero di figli di Franco:

```
SELECT count(*) as NumFigliDiFranco  
FROM Paternita  
WHERE Padre = 'Franco'
```

Paternità

Padre	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo

L'operatore aggregato (**count**) viene applicato al risultato dell'interrogazione:

```
SELECT *  
FROM Paternita  
WHERE Padre = 'Franco'
```



Paternità

Padre	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo

```
SELECT *  
FROM Paternita  
WHERE Padre = 'Franco'
```



Padre	Figlio
Franco	Andrea
Franco	Aldo

count



NumFigliDiFranco
2

```
SELECT count(*)  
as NumFigliDiFranco  
FROM Paternita  
WHERE Padre = 'Franco'
```

(*), ALL e DISTINCT

- Mediante le specifiche (*), ALL e DISTINCT è possibile contare
- (*): tutte le righe selezionate;
- ALL: tutti i valori non nulli delle righe selezionate;
- DISTINCT: tutti i valori non nulli distinti delle righe selezionate.

EsamiBD

Studente	BD	LBD
012345	27	NULL
032456	25	23
035221	NULL	NULL
033445	28	30
032441	NULL	30

- Contare il numero di studenti iscritti al corso di BD e Laboratorio

```
SELECT count(*) as "NumStud"  
FROM EsamiBD
```

NumStud
5

- Contare il numero di esami di BD superati positivamente

```
SELECT count([ALL] BD) "ContaBD"  
FROM EsamiBD
```

ContaBD
3

- Numero di voti distinti dati all'esame di LBD

```
SELECT count(distinct LBD) "ContDistLBD"  
FROM EsamiBD
```

ContDistLBD
2

AVG

- La funzione **AVG** calcola la media (average) dei valori **non nulli** di una colonna.
- Le specifiche **ALL** e **DISTINCT** servono a calcolare la media fra tutti i valori o tra i valori distinti.
- **Esempio:** Calcolare la media degli stipendi degli impiegati del dipartimento di Produzione e che hanno meno di 30 anni

```
SELECT AVG(stipendio)
FROM Impiegato
WHERE Dipart='Produzione' AND eta<30
```

Impiegato

NOME	ETA	STIPENDIO	DIPART
Andrea	27	2100	Direzione
Aldo	25	1500	Produzione
Maria	55	4200	Distribuzione
Anna	50	3500	Produzione
Filippo	26	3900	Distribuzione
Luigi	50	4000	Amministrazione
Franco	60	2000	Produzione
Olga	30	4100	Produzione
Sergio	85	3500	Amministrazione
Luisa	75	8700	Direzione

Max e Min

- Le funzioni **MAX** e **MIN** calcolano rispettivamente il maggiore e il minore degli elementi di una colonna.
- **Esempi:**
- L'età della persona più anziana nella tabella persone
`SELECT max(eta)
FROM Persone`
- Il reddito più basso della tabella persone
`SELECT min(reddito)
FROM Persone`

Persone

NOME	ETA	REDDITO
Andrea	27	21000
Aldo	25	15000
Maria	55	42000
Anna	50	35000
Filippo	26	29000
Luigi	50	40000
Franco	60	20000
Olga	30	41000
Sergio	85	35000
Luisa	75	87000

Sum

- La funzione SUM calcola la somma dei valori di una colonna.
- Le specifiche ALL e DISTINCT permettono di sommare tutti i valori non nulli o tutti i valori distinti.
- Esempio:
- Calcolare la somma dei redditi mensili delle persone che hanno 50 anni.

```
SELECT SUM (ALL reddito)  
FROM Persone  
WHERE eta=50
```

Persone

NOME	ETA	REDDITO
Andrea	27	21000
Aldo	25	15000
Maria	55	42000
Anna	50	35000
Filippo	26	29000
Luigi	50	40000
Franco	60	20000
Olga	30	41000
Sergio	85	35000
Luisa	75	87000

Funzioni Aggregative e proiezione

- Regola:
 - in una proiezione possono comparire **o solo attributi ordinari**, o **solo funzioni aggregative**
 - altrimenti la semantica non è ben definita
- Esempio
 - “Titolo e numero dei corsi della laurea triennale”

$\pi_{\text{COUNT(cod) AS numcorsi}}(\pi_{\text{titolo, COUNT(cod)}}(\pi_{\text{ciclo='laurea tr.'}}(\text{Corsi})))$

Funzioni Aggregative e lista degli attributi

- Non è possibile utilizzare in una stessa select una proiezione su alcuni attributi della tabella considerata e operatori aggregati sulla stessa tabella.

```
SELECT nome, max(reddito)  
FROM persone
```

Funziona???
Interrogazione
scorretta

- La target list deve essere omogenea.
- E' corretta invece la seguente:

```
SELECT min(eta), avg(reddito)  
FROM persone
```

E se volessimo avere il nome
della persona con il massimo
reddito?

Persone

Nome	Eta	Reddito
Andrea	27	30000
Aldo	25	NULL
Maria	55	36000
Anna	50	36000

Funzione aggregative in espressioni

Esami

<u>studente</u>	<u>corso</u>	voto	lode
111	PR1	27	false
222	ASD	30	true
111	INFT	24	false
77777	PR1	21	false
77777	ASD	20	false
88888	ASD	28	false
88888	PR1	30	false
88888	INFT	30	true

Studenti

<u>matr</u>	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
333	Rossi	Maria	laurea tr.	1	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

- Esempio: media degli esami dello studente Pasquale Bruno in 110mi.

```
SELECT AVG(voto)/30*110
```

```
FROM Studenti JOIN Esami ON cod=studente
```

```
WHERE cognome='Bruno' AND  
nome='Pasquale';
```

RAGGRUPPAMENTI

Clausole GROUP BY e HAVING

- GROUP BY
 - operatore di “raggruppamento”
- Sintassi
 - GROUP BY <attributi di raggruppamento>
- Semantica
 - raggruppamento della tabella
 - divisione in gruppi delle ennuple
 - raggruppamento sulla base dei valori comuni per gli attributi di raggruppamento

Clausole GROUP BY e HAVING

- Esempio: raggruppamento della tabella studenti per ciclo (GROUP BY ciclo)

Studenti

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
333	Rossi	Maria	laurea tr.	1	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

gruppo A
ciclo='laurea tr.'

gruppo B
ciclo='laurea sp.'

Clausole GROUP BY e HAVING

- **Esempio:** raggruppamento della tabella studenti per ciclo e anno
(GROUP BY ciclo, anno)

Studenti

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
333	Rossi	Maria	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

gruppo A
laurea tr., 1

gruppo B
laurea tr., 2

gruppo C
laurea tr., 3

gruppo D
laurea sp., 1

Clausole GROUP BY e HAVING

- **Esempio:** raggruppamento della tabella studenti per matricola
(GROUP BY matr)

Studenti

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
333	Rossi	Maria	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

Una ennupla
per ogni
gruppo

Clausole GROUP BY e HAVING

- Caratteristiche dei gruppi
 - collezioni di ennuple
 - valori comuni per gli attributi di raggruppamento.
- Operazioni interessanti sui gruppi
 - funzioni aggregative
 - analisi della distribuzione di valori tra i gruppi, esempio:
numero di studenti per ciclo o per anno
 - OLAP (“On Line Analytical Processing”)

Clausole GROUP BY e HAVING

- Interrogazioni con raggruppamento:
 - attributi di raggruppamento (nella GROUP BY)
 - proiezioni su attributi di raggruppamento e funzioni aggregative applicate al gruppo (nella SELECT)
 - condizioni sui gruppi (che coinvolgono funzioni aggregative) (nella HAVING)

Clausola GROUP BY

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
333	Rossi	Maria	laurea tr.	1	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

- Esempio: numero di studenti per ciclo

```
SELECT COUNT(*)  
FROM Studenti  
GROUP BY ciclo;
```

COUNT(*)
4
2

Clausola GROUP BY

- Esempio:
numero di studenti per ciclo

```
SELECT ciclo, COUNT(*)  
FROM Studenti  
GROUP BY ciclo;
```

- Semantica
 - viene valutata la clausola FROM
 - viene effettuato il raggruppamento secondo la GROUP BY
 - viene valutata la clausola SELECT per ciascun gruppo (ogni gruppo contribuisce ad UNA sola ennupla del risultato)

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
333	Rossi	Maria	laurea tr.	1	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

ciclo	COUNT(*)
laurea tr.	4
laurea sp.	2

Clausola GROUP BY

- Vincoli sintattici sulla SELECT
 - se c'è una GROUP BY, solo gli attributi di raggruppamento possono comparire nella SELECT

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
333	Rossi	Maria	laurea tr.	1	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

```
SELECT ciclo, count(*)  
FROM Studenti  
GROUP BY ciclo;
```

```
SELECT count(*)  
FROM Studenti  
GROUP BY ciclo;
```

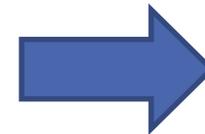
```
SELECT anno, count(*)  
FROM Studenti  
GROUP BY ciclo;
```

Clausola GROUP BY

- Esempio: distribuzione per anno degli studenti della laurea triennale

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
333	Rossi	Maria	laurea tr.	1	null
444	Pinco	Palla	laurea tr.	3	FT
77777	Bruno	Pasquale	laurea sp.	1	FT
88888	Pinco	Pietro	laurea sp.	1	CV

```
SELECT anno, count(*) as numstud  
FROM Studenti  
WHERE ciclo='laurea tr.'  
GROUP BY anno;
```



Clausola GROUP BY

I passo: WHERE ciclo='laurea tr.'

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
333	Rossi	Maria	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
444	Pinco	Palla	laurea tr.	3	FT

Risultato finale:
SELECT anno,
count(*) as numstud

anno	numstud
1	2
2	1
3	1

II passo: GROUP BY anno

matr	cognome	nome	ciclo	anno	relatore
111	Rossi	Mario	laurea tr.	1	null
333	Rossi	Maria	laurea tr.	1	null
222	Neri	Paolo	laurea tr.	2	null
444	Pinco	Palla	laurea tr.	3	FT

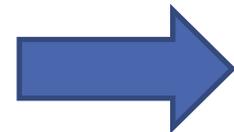
Clausole GROUP BY e HAVING

- Esempio: distribuzioni delle medie (per corso), solo per i corsi con più di 2 esami
 - non è possibile usare la WHERE
 - HAVING: condizioni aggregate su gruppi

Esami

studente	corso	voto	lode
111	PR1	27	false
...

```
SELECT corso, avg(voto) as votomedio  
FROM Esami  
GROUP BY corso  
HAVING count(voto)>2;
```



Clausole GROUP BY e HAVING

Esami

studente	corso	voto	lode
111	PR1	27	false
88888	PR1	30	false
77777	PR1	21	false
111	INFT	24	false
88888	INFT	30	true
222	ASD	30	true
77777	ASD	20	false
88888	ASD	28	false

I passo:
raggruppamento
secondo la
GROUP BY corso

II passo:
selezione dei
gruppi secondo
la HAVING count(voto)>2;

Risultato finale

corso	votomedio
PR1	26
ASD	26

III passo:
proiezione e rid.
secondo la
SELECT corso, avg(voto) as votomedio

Esempio: clausola HAVING

- Esempio: I dipartimenti la cui media degli stipendi degli impiegati è maggiore di 4000 euro.

```
Select dipart, AVG(STIPENDIO)  
FROM Impiegato  
Group by Dipart  
HAVING AVG(STIPENDIO)>4000
```

Dipart	AVG(stipendio)
Distribuzione	4050
Direzione	5400

HAVING AVG(reddito)>4000

Impiegato

NOME	ETA	STIPENDIO	DIPART
Andrea	27	2100	Direzione
Aldo	25	1500	Produzione
Maria	55	4200	Distribuzione
Anna	50	3500	Produzione
Filippo	26	3900	Distribuzione
Luigi	50	4000	Amministratozione
Franco	60	2000	Produzione
Olga	30	4100	Produzione
Sergio	85	3500	Amministratozione
Luisa	75	8700	Direzione

Dipart	AVG(Stipendio)
Amministratozione	3750
Distribuzione	4050
Direzione	5400
Produzione	2775

Where vs Having

- Per il dipartimento «produzione» visualizzare lo **stipendio solo se è maggiore di 2000**.
- Per ogni dipartimento, visualizzare il dipartimento e il massimo degli stipendi se tale **massimo è maggiore di 4000**.

Impiegato

NOME	ETA	STIPENDIO	DIPART
Andrea	27	2100	Direzione
Aldo	25	1500	Produzione
Maria	55	4200	Distribuzione
Anna	50	3500	Produzione
Filippo	26	3900	Distribuzione
Luigi	50	4000	Amministrazione
Franco	60	2000	Produzione
Olga	30	4100	Produzione
Sergio	85	3500	Amministrazione
Luisa	75	8700	Direzione

- Visualizzare la **media dell'età** degli impiegati dei dipartimenti Produzione e Direzione.
- Possibili interpretazioni?
 - calcolare la media degli impiegati che lavorano o in Produzione, o in Direzione o in entrambi
 - calcolare la media degli impiegati che lavorano in Produzione e la media di quelli che lavorano in Direzione

Ambiguità?

Esecuzione della SELECT

- Forma generale della SELECT

SELECT [DISTINCT] <risultato>

FROM <join o prodotti cartesiani>

[WHERE <condizioni>]

[GROUP BY <attributi di raggruppamento>]

[HAVING <condizioni sui gruppi>]

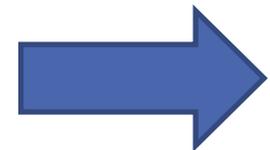
[ORDER BY <attributi di ordinamento>]

Esecuzione della SELECT

- Se la GROUP BY è omessa:
 - tutte le ennuple ottenute dopo la WHERE vengono considerate un unico gruppo
 - in questo caso le funzioni aggregative producono un unico valore e non sono ammessi attributi ordinari nella SELECT
- Nota sulla semantica
 - tutti i valori NULL normalmente vengono raggruppati assieme

Esecuzione della SELECT

- Una semantica operativa
 - viene valutata la clausola FROM
 - join o prodotti cartesiani >> unica tabella
 - viene valutata la clausola WHERE
 - selezione delle ennuple della tabella
 - viene valutata l'eventuale GROUP BY
 - raggruppamento delle ennuple della tabella
 - viene valutata l'eventuale HAVING
 - selezione dei gruppi della tabella



Esecuzione della SELECT

- Una semantica operativa (continua)
 - viene valutata la clausola SELECT
 - proiezioni, espressioni e funzioni aggregative
 - ridenominazioni
 - eventuale eliminazione di duplicati
 - viene valutata la clausola ORDER BY
 - ordinamenti finali

Forma Generale della SELECT

Studenti

matr	cognome	nome	ciclo	relat
111	Rossi	Mario	laurea tr.	null
333	Rossi	Maria	laurea tr.	null
222	Neri	Paolo	laurea tr.	null
444	Pinco	Palla	laurea tr.	FT
77777	Bruno	Pasquale	laurea sp.	FT
88888	Pinco	Pietro	laurea sp.	CV

studente	corso	voto	lode
111	PR1	27	false
222	ASD	30	true
111	INFT	24	false
77777	PR1	21	false
77777	ASD	20	false
88888	ASD	28	false
88888	PR1	30	false
88888	INFT	30	true

- Esempio:
- Visualizzare matr., cognome, nome e medie in ordine decrescente degli studenti della laurea specialistica che hanno sostenuto almeno due esami.

```
SELECT matr, cognome, nome, avg(voto)
FROM Studenti JOIN Esami ON matr=studente
WHERE ciclo='laurea sp.'
GROUP BY matr, cognome, nome
HAVING count(*)>=2
ORDER BY avg(voto) DESC;
```



Esecuzione della SELECT

Passo 1: FROM Studenti JOIN Esami ON matr=studente

matr	cognome	nome	ciclo	relat	studente	corso	voto	lode
111	Rossi	Mario	laurea tr.	null	111	PR1	27	false
111	Rossi	Mario	laurea tr.	null	111	INFT	24	false
222	Neri	Paolo	laurea tr.	null	222	ASD	30	true
77777	Bruno	Pasquale	laurea sp.	FT	77777	PR1	21	false
77777	Bruno	Pasquale	laurea sp.	FT	77777	ASD	20	false
88888	Pinco	Pietro	laurea sp.	VC	88888	ASD	28	false
88888	Pinco	Pietro	laurea sp.	VC	88888	PR1	30	false
88888	Pinco	Pietro	laurea sp.	VC	88888	INFT	30	true

Esecuzione della SELECT

Passo II: WHERE ciclo='laurea sp.'

matr	cognome	nome	ciclo	relat	studente	corso	voto	lode
77777	Bruno	Pasquale	laurea sp.	FT	77777	PR1	21	false
77777	Bruno	Pasquale	laurea sp.	FT	77777	ASD	20	false
88888	Pinco	Pietro	laurea sp.	VC	88888	ASD	28	false
88888	Pinco	Pietro	laurea sp.	VC	88888	PR1	30	false
88888	Pinco	Pietro	laurea sp.	VC	88888	INFT	30	true

Passo III: GROUP BY matr, cognome, nome

matr	cognome	nome	ciclo	relat	studente	corso	voto	lode
77777	Bruno	Pasquale	laurea sp.	FT	77777	PR1	21	false
77777	Bruno	Pasquale	laurea sp.	FT	77777	ASD	20	false
88888	Pinco	Pietro	laurea sp.	VC	88888	ASD	28	false
88888	Pinco	Pietro	laurea sp.	VC	88888	PR1	30	false
88888	Pinco	Pietro	laurea sp.	VC	88888	INFT	30	true

Esecuzione della SELECT

Passo IV: HAVING count(*) >= 2

matr	cognome	nome	ciclo	relat	studente	corso	voto	lode
77777	Bruno	Pasquale	laurea sp.	FT	77777	PR1	21	false
77777	Bruno	Pasquale	laurea sp.	FT	77777	ASD	20	false
88888	Pinco	Pietro	laurea sp.	VC	88888	ASD	28	false
88888	Pinco	Pietro	laurea sp.	VC	88888	PR1	30	false
88888	Pinco	Pietro	laurea sp.	VC	88888	INFT	30	true

Passo V: SELECT matr, cognome, nome, avg(voto)

matr	cognome	nome	avg(voto)
77777	Bruno	Pasquale	20,5
88888	Pinco	Pietro	29,66

Passo VI: ORDER BY avg(voto) DESC

matr	cognome	nome	avg(voto)
88888	Pinco	Pietro	29,66
77777	Bruno	Pasquale	20,5

INTERROGAZIONI NIDIFICATE

Interrogazioni Nidificate

- **SELECT Nidificate**
 - la clausola WHERE di una SELECT contiene un'altra SELECT
- **Due possibili utilizzi**
 - condizioni basate su valori semplici (SELECT che restituiscono un singolo valore)
 - condizioni basate su collezioni (SELECT ordinarie che restituiscono insiemi di ennuple)

Interrogazioni Nidificate

- Condizioni su valori semplici
 - confrontano il valore di un attributo con il risultato di una SELECT “scalare”
 - operatori: >, <, =, >=, <=, <>, LIKE, IS NULL
- SELECT “scalare”
 - SELECT che restituisce un’unica ennupla con un unico attributo
 - tipicamente: funzione aggregativa

Interrogazioni Nidificate

Studenti

- Esempio: lo studente con la matricola più grande

matr	cognome	nome	ciclo	relat
111	Rossi	Mario	laurea tr.	null
333	Rossi	Maria	laurea tr.	null
222	Neri	Paolo	laurea tr.	null
444	Pinco	Palla	laurea tr.	FT
77777	Bruno	Pasquale	laurea sp.	FT
88888	Pinco	Pietro	laurea sp.	CV

```
SELECT matr, cognome, nome
FROM Studenti
WHERE matr = (SELECT max(matr)
              FROM Studenti);
```

max(matr)
88888

per ogni ennupla di Studenti, il valore della matricola viene confrontato con il numero 88888

Esempio visto

- Il nome della persona con il massimo reddito

```
SELECT nome, max(reddito)  
FROM persone
```



Persone

Nome	Eta	Reddito
Andrea	27	30000
Aldo	25	NULL
Maria	55	36000
Anna	50	36000

```
SELECT nome  
FROM Persone  
WHERE reddito = (SELECT max(reddito)  
FROM Persone);
```

Uso di subquery in espressioni di confronto

Veicoli

<u>Targa</u>	Cod_mod	Categoria	Cilindrata	Cod_comb.	cav.Fisc	Velocita	Posti	Imm
--------------	---------	-----------	------------	-----------	----------	----------	-------	-----

- **Esempio:** tutti i veicoli di cilindrata superiore alla media delle cilindrata.

```
Select *
```

```
From Veicoli
```

```
Where Cilindrata > (select AVG(Cilindrata)
```

```
From Veicoli)
```

- **Esempio:** I veicoli di cilindrata massima.

```
Select *
```

```
From Veicoli
```

```
Where Cilindrata = (Select max(cilindrata)
```

```
From Veicoli)
```

Subquery + Join

Veicoli

Targa	Cod_mod	Categoria	Cilindrata	Cod_combust.	cav.Fisc	Velocita	Posti	Imm
-------	---------	-----------	------------	--------------	----------	----------	-------	-----

Modelli

Cod_Mod	Nome_Mod	Cod_Fab	Num_versioni
---------	----------	---------	--------------

Selezionare i modelli che presentano più versioni del numero minimo di versioni dei veicoli a benzina (Cod_combustibile='01').

```
Select *  
From Modelli  
Where num_versioni >
```

```
(Select min (num_versioni)  
From Veicoli, Modelli  
Where Veicoli.cod_mod=Modelli.cod_mod  
and cod_combust='01')
```

Regola della visibilità

Interrogazioni Nidificate

- Condizioni su valori non scalari (collezioni)
 - confrontano il valore di un attributo con il risultato di una SELECT generica (collezione di ennuple)
 - operatori: ordinari combinati con ANY, ALL
- ANY
 - “un elemento qualsiasi della collezione”; es: = ANY, oppure IN
- ALL
 - “tutti gli elementi della collezione”; es: > ALL

Interrogazioni Nidificate

Studenti

- Esempio: lo studente con la matricola **più grande** (senza funzioni aggregative)

matr	cognome	nome	ciclo	relat
111	Rossi	Mario	laurea tr.	null
333	Rossi	Maria	laurea tr.	null
222	Neri	Paolo	laurea tr.	null
444	Pinco	Palla	laurea tr.	FT
77777	Bruno	Pasquale	laurea sp.	FT
88888	Pinco	Pietro	laurea sp.	CV

```
SELECT matr, cognome, nome
FROM Studenti
WHERE matr >= ALL (SELECT matr
                   FROM Studenti);
```

per ogni ennupla di Studenti, il valore della matricola viene confrontato con tutte le matricole

matr
111
222
333
444
77777
88888

Subquery di confronto quantificato, esempio 1

- Selezionare tutti i veicoli con cilindrata **inferiore ad almeno una** delle cilindrature dei veicoli con combustibile 02 (quindi inferiore alla più alta di queste cilindrature).

Veicoli

Targa	Cod_mod	Categoria	Cilindrata	Cod_combust.	cav.Fisc	Velocita	Posti	Imm
-------	---------	-----------	------------	--------------	----------	----------	-------	-----

```
Select * FROM Veicoli
```

```
Where cilindrata < Any (select cilindrata From Veicoli  
Where cod_combust='02')
```

E se volessimo usare la funzione max?

```
Select * From Veicoli
```

```
Where cilindrata < (select max(cilindrata) From Veicoli  
Where cod_combust='02' )
```

Subquery di confronto quantificato, esempio

- Selezionare tutti i veicoli con cilindrata **inferiore a tutte** le cilindrature dei veicoli con combustibile 02 (quindi inferiore alla più bassa di queste cilindrature)

Veicoli

Targa	Cod_mod	Categoria	Cilindrata	Cod_combust.	cav.Fisc	Velocita	Posti	Imm
-------	---------	-----------	------------	--------------	----------	----------	-------	-----

```
Select * FROM Veicoli  
Where cilindrata < All (select cilindrata From Veicoli  
Where cod_combust='02')
```

E se volessimo usare la funzione min?

```
Select * From Veicoli  
Where cilindrata < (select min(cilindrata) From Veicoli  
Where cod_combust='02')
```