SQL: query complesse

VIDEOCONFERENZA 29/04/2020: **OPERATORI AGGREGATI, ORDER BY, GROUP BY, QUERY ANNIDATE**

Docente: CHIARA DAMIANI chiara.damiani@unimib.it

Materiale propedeutico da e-learning

8.6 - operatori aggregati

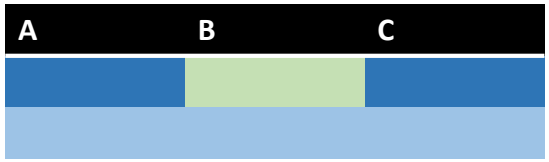
8.7 - operatore di raggruppamento

8.8_clausola having

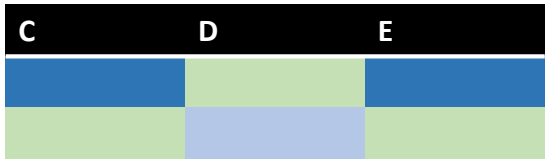
8.9_interrogazioni nidificate_introduzione

L’istruzione SELECT permette di replicare e combinare tutti gli operatori dell’algebra relazionale

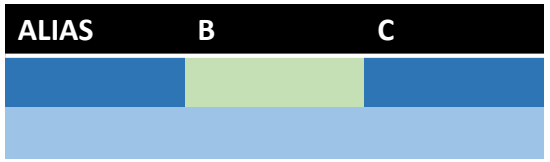
R1



R2



Ridenominazione

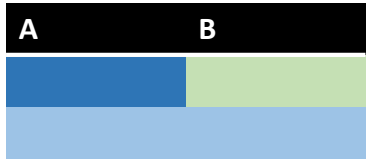


SELECT A ALIAS, B,C
FROM R1
WHERE A>B

Selezione

SELECT *
FROM R1
WHERE A>B

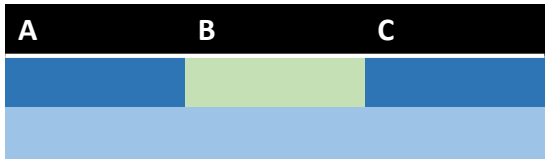
Proiezione



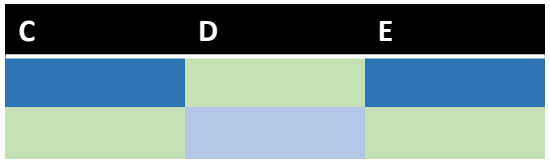
SELECT DISTINCT A,B
FROM R1

L'istruzione SELECT permette di replicare e combinare tutti gli operatori dell'algebra relazionale

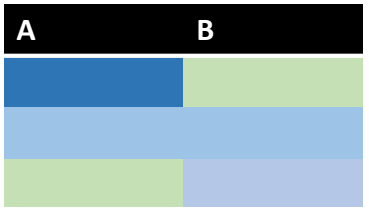
R1



R2



Unione



SELECT A,B **FROM** R1
UNION
SELECT C,D **FROM** R2

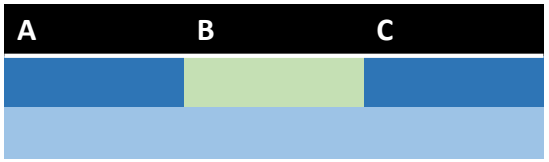
Intersezione

SELECT A,B **FROM** R1
INTERSECT
SELECT C,D **FROM** R2

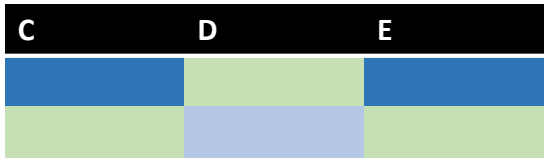
Differenza

SELECT A,B **FROM** R1
MINUS
SELECT C,D **FROM** R2

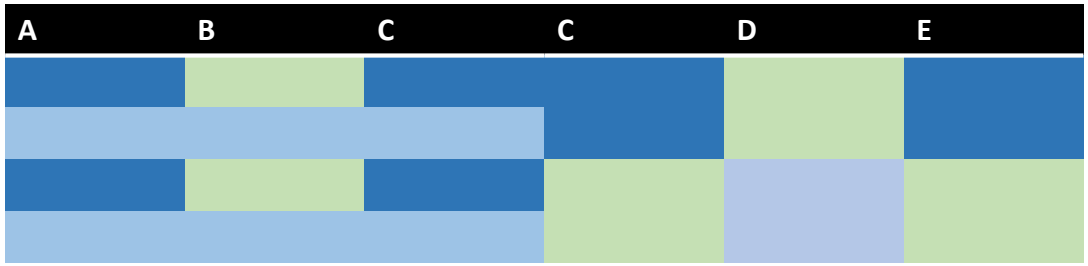
R1



R2



Prodotto cartesiano



SELECT *
FROM R1,R2

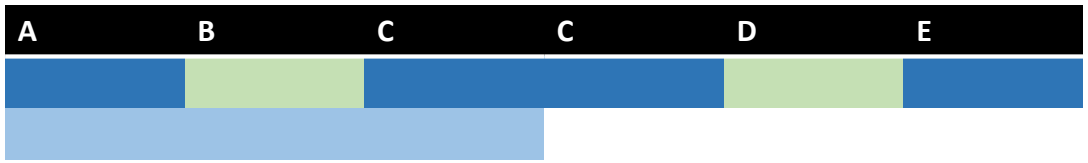
Theta Join

SELECT *
FROM R1,R2
WHERE R1.C =R2.C

Natural Join

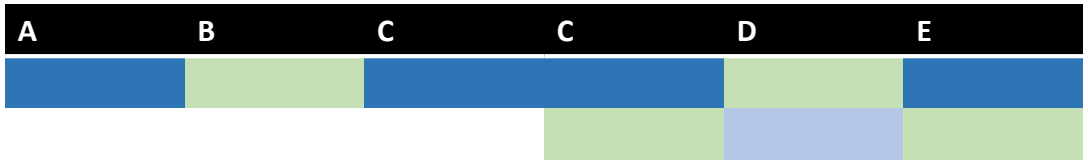
SELECT *
FROM R1 **NATURAL JOIN** R2

Left Join



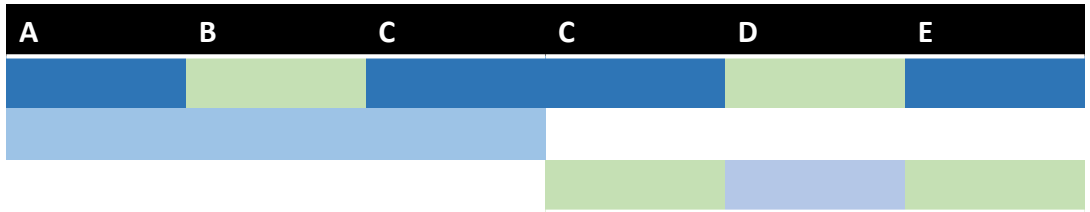
SELECT *
FROM R1 **LEFT JOIN** R2

Right Join



SELECT *
FROM R1 **RIGHT JOIN** R2

Full Join



SELECT *
FROM R1 **FULL JOIN** R2

Algebra relazionale

- Linguaggio prettamente formale che forma la base per linguaggi ‘reali’.
- Linguaggio procedurale: si specifica l’algoritmo con cui ottenere il risultato.
- Istruzioni equivalenti possono differire in termini di efficienza.
- Relazioni intese in senso matematico => Insiemi di tuple, definite su attributi
- **Negli insiemi non ci possono essere elementi uguali.**

SQL

- Linguaggio più usato per basi di dati relazionali.
- Linguaggio (parzialmente) dichiarativo: si specifica il risultato da ottenere senza preoccuparsi di specificare l’algoritmo.
- Istruzioni equivalenti differiscono solo per leggibilità.
- Relazioni intese come tabelle.
- **Possono esserci righe uguali -=> per eliminarle clausola DISTINCT**

Algebra relazionale

- Gli operatori insiemistici si applicano solo a relazioni definite sugli stessi attributi
- L'ordine degli attributi è irrilevante
- Il **theta join** ha senso solo per relazioni che non hanno attributi in comune perché si combinano sempre le tuple che hanno gli stessi valori per gli attributi comuni
- Si possono ridenominare solo attributi

SQL

- Gli operatori insiemistici si applicano a relazioni definite sullo stesso numero di attributi
- L'ordine degli attributi è rilevante
- Il **theta join** ha sempre senso, perché si combinano solo le tuple che hanno lo stesso valore per gli attributi specificati nella condizione di join => occorre disambiguare attributi omonimi con la **clausola punto**
- Si possono ridenominare attributi e tabelle => **ridenominazione tabelle** necessaria in caso di self join

L’istruzione SQL permette di fare più dell’AR

```
SELECT ListaAttributiOEspressioni
FROM ListaTabelle
[ WHERE CondizioneSelezioneTuple]
[ GROUP  BY ListaAttributiDiRaggruppamento ]
[ HAVING CondizioniSelezioneGruppi ]
[ ORDER  BY ListaAttributiDiOrdinamento ]
```

Il predicato può essere il confronto con una tabella ottenuta da una **subquery**

Subquery e predicati

- Gli operatori di confronto semplici **=, <>, <,<=, <=, >, >=** si possono usare solo per **subquery che restiuiscono un singolo valore** che si puo' usare come espressione scalare**:**

- Per usare predicati di confronto con selezioni che possono **restituire più di una riga**, occorre usare le quantificazioni **ALL** oppure **ANY** oppure il predicato **[NOT] IN**
  - Questi predicati possono essere applicati a **liste di attributi** (si usano le parentesi)

- Il predicato **[NOT] EXISTS** permette di verificare se la query **restituisce o meno una tupla**

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
183101	GILARDONI	GIOVANNI	Ordinario	13
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	10
218301	QUAGLIA	LAURA	Ordinario	10
774002	BUZZI	UMBERTO	Ricercatore	3
94302	QUERCINI	PIETRO	Ordinario	5

Esercizio 12: numero dei professori Ordinari

```
SELECT count(*) AS numero_ordinari
FROM Personale_docente
WHERE ruolo = 'Ordinario'
```

Risultato query

Numero_ordinari
3

Per ottenere il risultato dobbiamo utilizzare un operatore aggregato

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
183101	GILARDONI	GIOVANNI	Ordinario	13
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	10
218301	QUAGLIA	LAURA	Ordinario	10
774002	BUZZI	UMBERTO	Ricercatore	3
94302	QUERCINI	PIETRO	Ordinario	5

Esercizio 13: classifica ruolo docenti in base a numerosità

Risultato query

Ruolo	Numero
Ordinario	3
Ricercatore	2
Associato	1

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	10
774002	BUZZI	UMBERTO	Ricercatore	3
94302	QUERCINI	PIETRO	Ordinario	5
218301	QUAGLIA	LAURA	Ordinario	10
183101	GILARDONI	GIOVANNI	Ordinario	13

Esercizio 13: classifica ruolo docenti in base a numerosità

Per ottenere questo risultato dobbiamo prima raggruppare e poi contare

Risultato query

Ruolo	Numero
Ordinario	3
Ricercatore	2
Associato	1

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	10
774002	BUZZI	UMBERTO	Ricercatore	3
94302	QUERCINI	PIETRO	Ordinario	5
218301	QUAGLIA	LAURA	Ordinario	10
183101	GILARDONI	GIOVANNI	Ordinario	13

Esercizio 13: classifica ruolo docenti in base a numerosità

```
SELECT ruolo, count(*) AS numero
FROM Personale_docente
GROUP BY ruolo
ORDER BY numero DESC
```

Risultato query

Ruolo	Numero
Ordinario	3
Ricercatore	2
Associato	1

Per ottenere questo risultato dobbiamo prima raggruppare e poi contare

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	10
774002	BUZZI	UMBERTO	Ricercatore	3
94302	QUERCINI	PIETRO	Ordinario	5
218301	QUAGLIA	LAURA	Ordinario	10
183101	GILARDONI	GIOVANNI	Ordinario	13

Esercizio 13: classifica ruolo docenti in base a numerosità

```
SELECT ruolo, count(*) AS numero
FROM Personale_docente
GROUP BY ruolo
ORDER BY numero DESC
```

Risultato query

Ruolo	Numero
Ordinario	3
Ricercatore	2
Associato	1

Per ottenere questo risultato dobbiamo prima raggruppare e poi contare

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	6
774002	BUZZI	UMBERTO	Ricercatore	3

Stipendio

Classe	Valore
3	30000
5	50000
6	60000
14	140000

Esercizio 14: Matricola docente e valore stipendio del docente con lo stipendio più alto

Poll 3: La seguente soluzione è corretta?

```
SELECT Matricola_d, max(Valore)
FROM  Personale_docente, Stipendio
WHERE Classe_stipendio=Classe
```

La target list non può contenere colonne non aggregate insieme a colonne aggregate

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	6
774002	BUZZI	UMBERTO	Ricercatore	3

Stipendio

Classe	Valore
3	30000
5	50000
6	60000
14	140000

Esercizio 14: Matricola docente e valore stipendio del docente con lo stipendio più alto

Poll 4: La seguente soluzione è corretta?

```
SELECT Matricola_d, Valore
FROM  Personale_docente, Stipendio
WHERE Classe_stipendio=Classe AND Valore=
        (SELECT max(valore)
        FROM Stipendio)
```

NO perché il valore massimo in Stipendio non è necessariamente assegnato ad un docente

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	6
774002	BUZZI	UMBERTO	Ricercatore	3

Stipendio

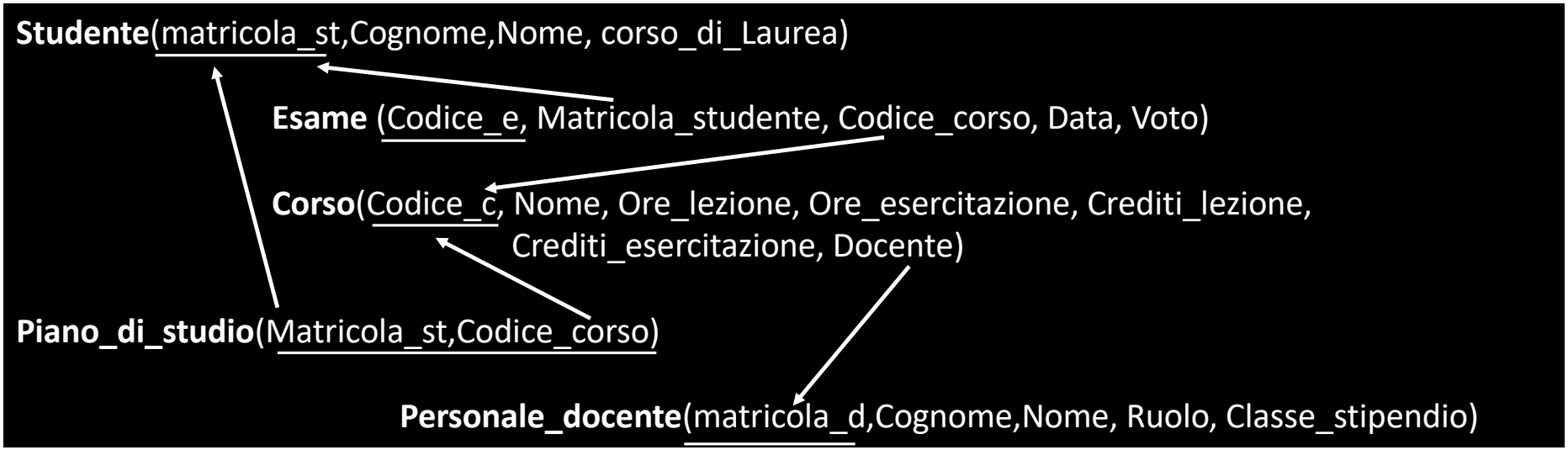
Classe	Valore
3	30000
5	50000
6	60000
14	140000

Esercizio 14: Matricola docente e valore stipendio del docente con lo stipendio più alto

```
SELECT Matricola_d, Valore
FROM  Personale_docente, Stipendio
WHERE Classe_stipendio=Classe AND Valore=
            (SELECT  max(valore)
             FROM  Personale_docente, Stipendio
            WHERE Classe_stipendio=Classe)
```

Soluzione corretta

Esercizio 15: Identificare vincoli di chiave primaria e di integrità referenziale del seguente schema relazionale



Offerta formativa

Ogni corso può essere insegnato da un solo professore, e ogni professore può erogare più corsi.

I **corsi** sono descritti con un codice, un nome, la laurea a cui afferiscono, cioè se triennale o magistrale, il numero di ore di lezione, il numero di ore di esercitazione, il corso di laurea o i corsi di laurea a cui afferiscono).

Carriere degli studenti

I **piani di studio** sono descritti per ogni studente mediante la matricola dello studente e il codice dei corsi che fanno parte del piano.

Gli **esami** sono descritti da un codice corso, la matricola dello studente, il voto e la data, con giorno, mese, anno.

I **corsi** sono descritti mediante un codice, un nome, il corso di laurea cui afferiscono, ad esempio "Informatica", "Teoria e tecnica della comunicazione", ecc.) il numero di crediti di lezioni, il numero di crediti di esercitazioni.

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

Che info ci servono?

- Esami sostenuti
- Voto ottenuto ad ogni esame
- CFU corsi (crediti lezione + crediti esercitazione)

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

Che tabelle ci servono?

- Esame
- Corso

Corso

Codice_c	Nome	Ore_lezione	Ore_esercitazione	Crediti_lezione	Crediti_esercitazione	Docente
F1801Q109	Biologia computazionale	35	18	5	3	462901
F1801Q103	Sistemi Informativi	28	14	4	2	439001
F1801Q106	Intelligenza Artificiale	28	14	4	2	512402
F1801Q108	Bioinformatica	35	18	5	3	159501
F1801Q114	Ubiquitous e Context-Aware Computing	21	11	3	2	764702
F1801Q116	Evoluzione dei Sistemi Software e Reverse Eng	35	18	5	3	494202
F1801Q121	Data warehouse	14	7	2	1	617501
F1801Q122	Teoria dell'Informazione e Crittografia	35	18	5	3	439003
F1801Q130	Architetture del software e dei dati	28	14	4	2	101

Esame

#Codice_e	Matricola_studente	Codice_corso	Data	Voto
5311	1497001	F1801Q108	13/07/13 00:00	19
5312	1497001	F1801Q121	02/06/13 00:00	26
5316	1497001	F1801Q103	28/09/13 00:00	30
5317	1515801	F1801Q114	27/06/13 00:00	30
5318	1515801	F1801Q106	20/03/13 00:00	22
5322	1515801	F1801Q121	15/08/13 00:00	24
5324	1515801	F1801Q103	28/03/13 00:00	29
5325	1515801	F1801Q130	04/08/13 00:00	20
5326	1515801	F1801Q116	25/07/13 00:00	19
5327	1524501	F1801Q122	07/06/13 00:00	19

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

Come vanno correlate le due tabelle?

```
SELECT *
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
```

SELECT * FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c

Codice_e	Matricola_studente	Codice_corso	Data	Voto	Codice_c	Nome	Ore_lezione	Ore_esercitazione	Crediti_lezione	Crediti_esercitazione	Docente
5311	1497001	F1801Q108	13/07/13 00:00	19	F1801Q108	Bioinformatica	35	18	5	3	159501
5312	1497001	F1801Q121	02/06/13 00:00	26	F1801Q121	Data warehouse	14	7	2	1	617501
5313	1497001	F1801Q114	21/10/13 00:00	30	F1801Q114	Ubiquitous e Context-Aware Computing	21	11	3	2	764702
5316	1497001	F1801Q103	28/09/13 00:00	30	F1801Q103	Sistemi Informativi	28	14	4	2	439001
5317	1515801	F1801Q114	27/06/13 00:00	30	F1801Q114	Ubiquitous e Context-Aware Computing	21	11	3	2	764702
5322	1515801	F1801Q121	15/08/13 00:00	24	F1801Q121	Data warehouse	14	7	2	1	617501
5324	1515801	F1801Q103	28/03/13 00:00	29	F1801Q103	Sistemi Informativi	28	14	4	2	439001
5325	1515801	F1801Q130	04/08/13 00:00	20	F1801Q130	Architetture del software e dei dati	28	14	4	2	101
5326	1515801	F1801Q116	25/07/13 00:00	19	F1801Q116	Evoluzione dei Sistemi Software e Reverse Eng	35	18	5	3	494202
5327	1524501	F1801Q122	07/06/13 00:00	19	F1801Q122	Teoria dell'Informazione e Crittografia	35	18	5	3	439003
5330	1524501	F1801Q130	10/11/13 00:00	29	F1801Q130	Architetture del software e dei dati	28	14	4	2	101
5331	1524501	F1801Q108	05/06/13 00:00	31	F1801Q108	Bioinformatica	35	18	5	3	159501
5332	1524501	F1801Q103	03/08/13 00:00	29	F1801Q103	Sistemi Informativi	28	14	4	2	439001

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

Step 1: Trovare voto e CFU totali degli esami sostenuti dagli studenti

```
SELECT Matricola_studente Matr, Voto, Crediti_lezione+Crediti_esercitazione AS CFU
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
```

SELECT *
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c

Codice_e	Matricola_studente	Codice_corso	Data	Voto	Codice_c	Nome	Ore_lezione	Ore_esercitazione	Crediti_lezione	Crediti_esercitazione	Docente
5311	1497001	F1801Q108	13/07/13 00:00	19	F1801Q108	Bioinformatica	35	18	5	3	159501
5312	1497001	F1801Q121	02/06/13 00:00	26	F1801Q121	Data warehouse	14	7	2	1	617501
5313	1497001	F1801Q114	21/10/13 00:00	30	F1801Q114	Ubiquitous e Context-Aware Computing	21	11	3	2	764702
5316	1497001	F1801Q103	28/09/13 00:00	30	F1801Q103	Sistemi Informativi	28	14	4	2	439001
5317	1515801	F1801Q114	27/06/13 00:00	30	F1801Q114	Ubiquitous e Context-Aware Computing	21	11	3	2	764702
5322	1515801	F1801Q121	15/08/13 00:00	24	F1801Q121	Data warehouse	14	7	2	1	617501
5324	1515801	F1801Q103	28/03/13 00:00	29	F1801Q103	Sistemi Informativi	28	14	4	2	439001
5325	1515801	F1801Q130	04/08/13 00:00	20	F1801Q130	Architetture del software e dei dati	28	14	4	2	101
5326	1515801	F1801Q116	25/07/13 00:00	19	F1801Q116	Evoluzione dei Sistemi Software e Reverse Eng	35	18	5	3	494202
5327	1524501	F1801Q122	07/06/13 00:00	19	F1801Q122	Teoria dell'Informazione e Crittografia	35	18	5	3	439003
5330	1524501	F1801Q130	10/11/13 00:00	29	F1801Q130	Architetture del software e dei dati	28	14	4	2	101
5331	1524501	F1801Q108	05/06/13 00:00	31	F1801Q108	Bioinformatica	35	18	5	3	159501
5332	1524501	F1801Q103	03/08/13 00:00	29	F1801Q103	Sistemi Informativi	28	14	4	2	439001

SELECT Matricola_studente as Matr,
Voto,
Crediti_lezione+Crediti_esercitazione AS CFU
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c

Matr	Voto	CFU
1497001	19	8
1497001	26	3
1497001	30	5
1497001	30	6
1515801	30	5
1515801	24	3
1515801	29	6
1515801	20	6
1515801	19	8
1524501	19	8
1524501	29	6
1524501	31	8
1524501	29	6

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

Step 2: calcolare la media ponderata della Matricola 1497001

```
SELECT SUM(Voto*(Crediti_lezione+Crediti_esercitazione))/SUM(Crediti_lezione+Crediti_esercitazione) AS
media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
AND Matricola_studente = 1497001
```

1

```
SELECT Matricola_studente as Matr,
Voto,
Crediti_lezione+Crediti_esercitazione AS
CFU
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
```

Matr	Voto	CFU
1497001	19	8
1497001	26	3
1497001	30	5
1497001	30	6
1515801	30	5
1515801	24	3
1515801	29	6
1515801	20	6
1515801	19	8
1524501	19	8
1524501	29	6
1524501	31	8
1524501	29	6

2

```
SELECT Matricola_studente as Matr, Voto,
Crediti_lezione+Crediti_esercitazione AS CFU
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c and Matr=
1497001
```

Matr	Voto	CFU
1497001	19	8
1497001	26	3
1497001	30	5
1497001	30	6

3

```
SELECT
sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Credit
i_lezione+Crediti_esercitazione) as media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
AND Matricola_studente = 1497001
```

Media_ponderata
23.8974

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

Step 3: calcolare la media ponderata di ciascuno studente

```
SELECT Matricola_studente,
sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Crediti_lezione+Crediti_esercitazione) as
media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
GROUP BY Matricola_studente
```

```
SELECT Matricola_studente as Matr,
Voto,
Crediti_lezione+Crediti_esercitazione AS
CFU
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
```

Matr	Voto	CFU
1497001	19	8
1497001	26	3
1497001	30	5
1497001	30	6
1515801	30	5
1515801	24	3
1515801	29	6
1515801	20	6
1515801	19	8
1524501	19	8
1524501	29	6
1524501	31	8
1524501	29	6

Matricola_studente	media_ponderata
1524501	26.7143
1515801	23.8571
1497001	25.4545

```
SELECT Matricola_studente,
sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Credit
i_lezione+Crediti_esercitazione) as media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
GROUP BY Matricola_studente
```

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

Step 4: trovare gli studenti che hanno la media ponderata maggiore di 25

```
SELECT Matricola_studente,
sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Crediti_lezione+Crediti_esercitazione) as
media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
GROUP BY Matricola_studente
HAVING media_ponderata>25
```

```
SELECT Matricola_studente as Matr,
Voto,
Crediti_lezione+Crediti_esercitazione AS
CFU
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
```

Matr	Voto	CFU
1497001	19	8
1497001	26	3
1497001	30	5
1497001	30	6
1515801	30	5
1515801	24	3
1515801	29	6
1515801	20	6
1515801	19	8
1524501	19	8
1524501	29	6
1524501	31	8
1524501	29	6

Matricola_studente	media_ponderata
1524501	26.7143
1515801	23.8571
1497001	25.4545

```
SELECT Matricola_studente,
sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Credit
i_lezione+Crediti_esercitazione) as media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
GROUP BY Matricola_studente
```

```
SELECT Matricola_studente,
sum(Voto*(Crediti_lezione+Crediti_esercit
azione))/sum(Crediti_lezione+Crediti_eser
citazione) as media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
GROUP BY Matricola_studente
HAVING media_ponderata>25
```

Matricola_studente	media_ponderata
1524501	26.7143
1497001	25.4545

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

Step finale: trovare gli studenti che hanno la media ponderata maggiore delle medie ponderate degli altri studenti

```
SELECT  Matricola_studente,
sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Crediti_lezione+Crediti_esercitazione) AS
media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
GROUP BY Matricola_studente
HAVING media_ponderata >=  ALL (
            SELECT
            sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Crediti_lezione+Crediti_esercitazione) AS
            media_ponderata
            FROM Esame E, Corso C
            WHERE E.Codice_corso=C.Codice_c
            GROUP BY Matricola_studente )
```

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 16: trovare lo studente (matricola) con la media ponderata più alta

```
SELECT  Matricola_studente,
sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Crediti_lezione+Crediti_esercitazione) AS
media_ponderata
FROM Esame E, Corso C
WHERE E.Codice_corso=C.Codice_c
GROUP BY Matricola_studente
HAVING media_ponderata >=  ALL (
            SELECT     Matricola_studente,
            sum(Voto*(Crediti_lezione+Crediti_esercitazione))/sum(Crediti_lezione+Crediti_esercitazione) AS
            media_ponderata
            FROM Esame E, Corso C
            WHERE E.Codice_corso=C.Codice_c
            GROUP BY Matricola_studente )
```

Considerare più di un attributo nella target list della subquery sarebbe in questo caso un errore

Corso

Codice_c	Nome	Ore_lezione	Ore_esercitazione	Crediti_lezione	Crediti_esercitazione	Docente
F1801Q106	Intelligenza Artificiale	28	14	4	2	512402
F1801Q107	Sistemi Complessi: modelli e simulazione	14	7	2	1	399601
F1801Q108	Bioinformatica	35	18	5	3	159501
F1801Q109	Biologia computazionale	35	18	5	3	462901
F1801Q110	Information Retrieval	28	14	4	2	512302
F1801Q111	Gestione della Conoscenza	28	14	4	2	163601

Esercizio 17: trovare i corsi che contengono la parola «bio»

```
SELECT *
FROM Corso
WHERE Nome like '%bio%'
```

- Operatore **like**
- il simbolo **%** rappresenta zero, uno, più caratteri
- Il simbolo **_** rappresenta un singolo carattere

Risultato query

Codice_c	Nome	Ore_lezione	Ore_esercitazione	Crediti_lezione	Crediti_esercitazione	Docente
F1801Q108	Bioinformatica	35	18	5	3	159501
F1801Q109	Biologia computazionale	35	18	5	3	462901

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 18: trovare gli studenti che non hanno mai dato un esame

```
SELECT *
FROM Studente
WHERE Matricola_st NOT IN (
        SELECT Matricola_studente
        FROM Esame E)
```

```
    SELECT *
    FROM Studente
MINUS
    SELECT Matricola_St,Nome, Cognome
    FROM Studente S, Esame E
    WHERE S.Matricola_St=E.Matricola_studente
```

Oppure OUTER JOIN a selezione valori NULLI

Studente

Matricola_st	Cognome	Nome
1446203	VISANI	FRANCESCO
1497001	MOSCHINI	PIETRO
1515801	CASADIO	SILVANO
1524501	GATTI	LUIGI

Esame

# Codice_e	Matricola_studente	Codice_corso	Data	Voto
5311	1497001	F1801Q108	13/07/13 00:00	19
5312	1497001	F1801Q121	02/06/13 00:00	26
5316	1497001	F1801Q103	28/09/13 00:00	30
5317	1515801	F1801Q114	27/06/13 00:00	30
5318	1515801	F1801Q106	20/03/13 00:00	22
5322	1515801	F1801Q121	15/08/13 00:00	24
5324	1515801	F1801Q103	28/03/13 00:00	29
5325	1515801	F1801Q130	04/08/13 00:00	20
5326	1515801	F1801Q116	25/07/13 00:00	19
5327	1524501	F1801Q122	07/06/13 00:00	19

```
SELECT *
FROM Studente
WHERE Matricola_st NOT IN
(SELECT Matricola_studente
FROM Esame E)
```

Matricola_st	Cognome	Nome
1446203	VISANI	FRANCESCO

Studente(matricola_st,Cognome,Nome, corso_di_Laurea)
Esame (Codice_e, Matricola_studente, Codice_corso, Data, Voto)
Piano_di_studio(Matricola_st,Codice_corso)
Corso(Codice_c, Nome, Ore_lezione, Ore_esercitazione, Crediti_lezione, Crediti_esercitazione, Docente)
Personale_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)

Esercizio 19: studenti che hanno dato tutti gli esami previsti dal loro piano di studi

DA FARE A CASA