

27. Il comando select e l'algebra relazionale

parte prima

database

il comando select e l'algebra relazionale

Le operazioni di interrogazione usando il linguaggio SQL, vengono realizzate per mezzo del costrutto **SELECT-FROM-WHERE** la cui forma base generale è la seguente:

SELECT <lista_campi> ← Quali campi deve visualizzare ?
FROM <lista_tabelle> ← Da quali tabelle preleva i dati da visualizzare ?
[**WHERE** <condizione>]; ← In base a quali criteri sceglie le righe da visualizzare ?

Questo comando realizza un'interrogazione (*query*) della base di dati restituendo come risultato una *tabella virtuale* avente come colonne i campi specificati dopo la parola chiave **SELECT**.

il comando select e l'algebra relazionale

Le operazioni di interrogazione usando il linguaggio SQL, vengono realizzate per mezzo del costrutto **SELECT-FROM-WHERE** la cui forma base generale è la seguente:

SELECT <lista_campi> ← Quali campi deve visualizzare ?
FROM <lista_tabelle> ← Da quali tabelle preleva i dati da visualizzare ?
[WHERE <condizione> **];** ← In base a quali criteri sceglie le righe da visualizzare ?

NB: Le parentesi quadre indicano che la clausola `WHERE` è opzionale; inoltre, nella lista dei campi (o nella condizione), quando è necessario identificare il nome di un attributo come appartenente ad una specifica tabella, si deve utilizzare la notazione:

<nome_tabella>.<nome_campo> (separati dal punto).

il comando select e l'algebra relazionale

Gli operatori dell'algebra relazionale di selezione (**RESTRICT**), proiezione (**PROJECT**) e congiunzione (**JOIN**) su una base di dati relazionale visti in precedenza, possono essere realizzati attraverso il comando **SELECT**, secondo le diverse forme consentite dalla sintassi di questo comando.

NB: SQL esprime le interrogazioni in modo *dichiarativo*, ovvero specifica l'obiettivo della query, e non il modo in cui ottenerlo; in ciò SQL si contrappone ai linguaggi di interrogazione *procedurali*, come l'algebra relazionale, in cui l'interrogazione specifica i passi da compiere per estrarre le informazioni dal database.

























Riprendiamo lo schema di relazione che registra gli studenti iscritti in un certo ateneo universitario:

Studente (Matricola, Cognome, Nome, DataNascita, Residenza, CorsoLaurea, DataIscrizione, AnnoCorso);

il comando select e l'algebra relazionale

Possiamo realizzare questa tabella con il seguente comando SQL:

```
CREATE TABLE Studente(  
    Matricola INT PRIMARY KEY AUTO_INCREMENT,  
    Cognome CHAR(24) NOT NULL,  
    Nome CHAR(24) NOT NULL,  
    DataNascita DATE DEFAULT NULL,  
    Residenza VARCHAR(32) DEFAULT '',  
    CorsoLaurea VARCHAR(32) DEFAULT '',  
    DataIscrizione DATE DEFAULT NULL,  
    AnnoCorso INT NOT NULL )
```

← T →			▼	Matricola	Cognome	Nome	DataNascita	Residenza	CorsoLaurea	DataIscrizione	A
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	4498	Bitonti	Piero	1993-10-11	Lamezia Terme	Scienze Infermieristiche	2016-11-05	3
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	5901	Pascale	Irene	1975-11-13	Lamezia Terme	Scienze Infermieristiche	2017-11-09	2
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	6577	Velasco	Ester	1998-12-09	Crotone	Scienze Infermieristiche	2018-10-10	1
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	6990	Vecchi	Andrea	1994-07-14	Cosenza	Fisioterapista	2016-10-31	3
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	7609	Maurito	Franco	1987-09-09	Catanzaro	Logopedista	2018-09-03	1
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	7639	Criniti	Annarita	1990-09-02	Cosenza	Fisioterapista	2017-10-12	2
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	8473	Baldieri	Marco	1995-07-12	Catanzaro	Scienze Infermieristiche	2018-09-02	1
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	8841	Ragusa	Giacomo	1995-05-04	Soverato	Tecnico della Prevenzione	2018-10-04	1
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	8904	Petrocelli	Carmelina	1997-06-03	Crotone	Scienze Infermieristiche	2018-11-12	1
<input type="checkbox"/>	 Modifica	 Copia	 Elimina	9021	Amato	Gianluca	2000-03-01	Cosenza	Tecnico della Prevenzione	2018-10-04	1

Struttura della tabella *Studente*

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Pre
<input type="checkbox"/>	1 Matricola	int(11)			No	Ne
<input type="checkbox"/>	2 Cognome	char(24)	utf8mb4_general_ci		No	Ne
<input type="checkbox"/>	3 Nome	char(24)	utf8mb4_general_ci		No	Ne
<input type="checkbox"/>	4 DataNascita	date			Sì	NU
<input type="checkbox"/>	5 Residenza	varchar(32)	utf8mb4_general_ci		Sì	
<input type="checkbox"/>	6 CorsoLaurea	varchar(32)	utf8mb4_general_ci		Sì	
<input type="checkbox"/>	7 DataIscrizione	decimal(6,2)			Sì	0.0
<input type="checkbox"/>	8 AnnoCorso	int(11)			No	Ne

Dati contenuti nella tabella *Studente*

il comando select e l'algebra relazionale

L'operazione di selezione (**RESTRICT**), che consente di ricavare da una tabella iniziale, un'altra tabella (temporanea) contenente solo le righe che soddisfano ad una certa condizione, viene realizzata in SQL sfruttando la clausola **WHERE** del comando **SELECT**.

Ad es. per ottenere l'elenco con tutti i dati degli studenti iscritti al corso di laurea in "Fisioterapista", possiamo usare il seguente operatore dell'algebra relazionale:

```
RESTRICT Studenti  
  WHERE CorsoLaurea= "Fisioterapista"
```

equivalente al comando SQL:

```
SELECT *  
  FROM Studenti  
  WHERE CorsoLaurea= "Fisioterapista";
```

NB: La lista degli attributi da visualizzare può essere estesa all'intera tabella (quindi a tutti i suoi campi) utilizzando la clausola **SELECT** *****.

il comando select e l'algebra relazionale

Utilizzando *MySQL* è possibile provare immediatamente il funzionamento della query appena descritta:

Query scritta in SQL

Tabella risultato

Esegui la/le query SQL sulla tabella registroesami.studente:

```
1 SELECT *  
2 FROM studente  
3 WHERE CorsoLaurea = "Fisioterapista"
```

☐ Mostra tutti | Numero di righe: 25 | Filtra righe: Cerca nella tabella | Ordina per chiave: Nessuno

+ Opzioni

				Matricola	Cognome	Nome	DataNascita	Residenza	CorsoLaurea	DataIscrizione	AnnoCorso
<input type="checkbox"/>	Modifica	Copia	Elimina	6990	Vecchi	Andrea	1994-07-14	Cosenza	Fisioterapista	2016-10-31	3
<input type="checkbox"/>	Modifica	Copia	Elimina	7639	Criniti	Annarita	1990-09-02	Cosenza	Fisioterapista	2017-10-12	2

il comando select e l'algebra relazionale

L'operazione di proiezione (**PROJECT**), che permette di ottenere una tabella contenente solo alcuni attributi della tabella di partenza, si realizza indicando accanto al comando **SELECT** l'elenco degli attributi richiesti.

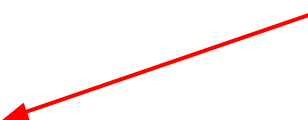
Ad es. per ottenere l'elenco di tutti gli studenti con *Cognome*, *Nome* e *Matricola*, usiamo il seguente operatore dell'algebra relazionale:

PROJECT *Studenti*
OVER *Cognome, Nome, Matricola*

equivalente al comando SQL:

SELECT *Cognome, Nome, Matricola*
FROM *Studenti* ;

Ovviamente, non è necessario che i campi vengano riportati seguendo lo stesso ordine specificato nello schema di relazione.



il comando select e l'algebra relazionale

Anche in questo caso, l'aiuto dell'interfaccia *phpMyAdmin* è utile per la realizzazione dei nostri comandi SQL:

☐ Mostra tutti | Numero di righe: 25 | Filtra righe: Cerca ne

+ Opzioni

				Cognome	Nome	Matricola
<input type="checkbox"/>	Modifica	Copia	Elimina	Bitonti	Piero	4498
<input type="checkbox"/>	Modifica	Copia	Elimina	Pascale	Irene	5901
<input type="checkbox"/>	Modifica	Copia	Elimina	Velasco	Ester	6577
<input type="checkbox"/>	Modifica	Copia	Elimina	Vecchi	Andrea	6990
<input type="checkbox"/>	Modifica	Copia	Elimina	Maurito	Franco	7609
<input type="checkbox"/>	Modifica	Copia	Elimina	Criniti	Annarita	7639
<input type="checkbox"/>	Modifica	Copia	Elimina	Baldieri	Marco	8473
<input type="checkbox"/>	Modifica	Copia	Elimina	Ragusa	Giacomo	8841
<input type="checkbox"/>	Modifica	Copia	Elimina	Petrocelli	Carmelina	8904
<input type="checkbox"/>	Modifica	Copia	Elimina	Amato	Gianluca	9021



Esegui la/le query SQL sulla tabella registroesami studente: ?

```
1 SELECT Cognome, Nome, Matricola
2 FROM studente
```

Tabella risultato

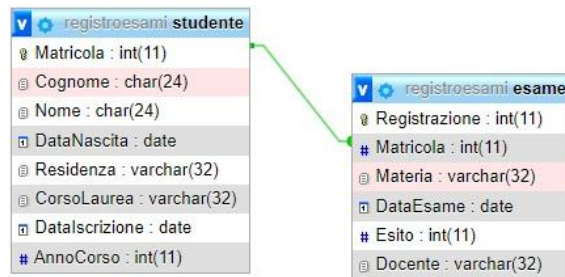
Query scritta in SQL

il comando select e l'algebra relazionale

Aggiungiamo adesso alla tabella *Studenti* anche la tabella seguente:

Esami (Registrazione, Matricola, Materia, DataEsame, Esito, Docente);

```
CREATE TABLE Esame (  
    Registrazione INT PRIMARY KEY AUTO_INCREMENT,  
    Matricola INT NOT NULL,  
    Materia VARCHAR(32) NOT NULL,  
    DataEsame Date NOT NULL,  
    Esito INT NOT NULL,  
    Docente VARCHAR(32),  
    FOREIGN KEY (Matricola) REFERENCES Studente (Matricola)  
);
```



Esiste un'associazione 1:N da *Studenti* a *Esami*, in quanto ogni studente registra sul suo libretto universitario più esami sostenuti (ogni volta che ne supera uno), ma ogni registrazione di esame superato fa riferimento ad un solo studente.

il comando select e l'algebra relazionale

Il comando **SELECT** può operare su più tabelle, indicandone i nomi (separati da virgole) dopo la parola chiave **FROM**.

Riportando poi, dopo la parola chiave **WHERE**, i nomi degli attributi che si corrispondono nelle due tabelle, legati tra loro dal segno "=", si realizza di fatto l'operazione di congiunzione (**JOIN**) delle tabelle secondo l'attributo comune.

```
SELECT Campo1, Campo2, Campo3,...  
FROM Tabella1, Tabella2,...  
WHERE Tabella1.Campo1 = Tabella2.Campo2
```

il comando select e l'algebra relazionale

Nel nostro caso l'attributo comune è *Matricola*; chiave primaria nella tabella *Studenti*, viene riportato come chiave esterna nella tabella *Esami*.

Per ottenere l'elenco di tutti gli studenti con i dati degli esami da essi sostenuti, occorre effettuare la congiunzione delle tabelle sul campo comune *Matricola*; possiamo usare il seguente operatore dell'algebra relazionale:

```
JOIN Studenti WITH Esami  
ON Matricola
```

equivalente al comando SQL:

```
SELECT *  
FROM Studenti, Esami  
WHERE Studenti.Matricola = Esami.Matricola ;
```

il comando select e l'algebra relazionale

Vogliamo adesso un elenco con il cognome, il nome e la città di residenza degli studenti che vivono nel catanzarese (nel nostro esempio, in Catanzaro oppure Soverato).

PROJECT

RESTRICT Studenti **WHERE**

Residenza = "Catanzaro" **OR** Residenza = "Soverato"

OVER

Cognome, Nome, Residenza

Matricola	Cognome	Nome	DataNascita	DataIscrizione	Residenza	CorsoLaurea	AnnoCorso
8473	Baldieri	Marco	12/07/1995	02/09/2018	Catanzaro	Scienze Infermieristiche	1
7639	Criniti	Annarita	02/09/1990	12/10/2017	Cosenza	Fisioterapista	2
6577	Velasco	Ester	09/12/1998	10/10/2018	Crotone	Scienze Infermieristiche	1
4498	Bitonti	Piero	11/10/1993	05/11/2016	Lamezia T.	Scienze Infermieristiche	3
8841	Ragusa	Giacomo	04/05/1995	04/10/2018	Soverato	Tecn. Prevenzione	1
6990	Vecchi	Andrea	14/07/1994	31/10/2016	Cosenza	Fisioterapista	3
8904	Petrocelli	Carmelina	03/06/1997	12/11/2018	Crotone	Scienze Infermieristiche	1
7609	Maurito	Franco	09/09/1987	03/09/2018	Catanzaro	Logopedista	1
5901	Pascale	Irene	13/11/1975	09/11/2017	Lamezia T.	Scienze Infermieristiche	2

Studenti

Matricola	Cognome	Nome	DataNascita	DataIscrizione	Residenza	CorsoLaurea	AnnoCorso
8473	Baldieri	Marco	12/07/1995	02/09/2018	Catanzaro	Scienze Infermieristiche	1
8841	Ragusa	Giacomo	04/05/1995	04/10/2018	Soverato	Tecn. Prevenzione	1
7609	Maurito	Franco	09/09/1987	03/09/2018	Catanzaro	Logopedista	1

DopoLaSelezione

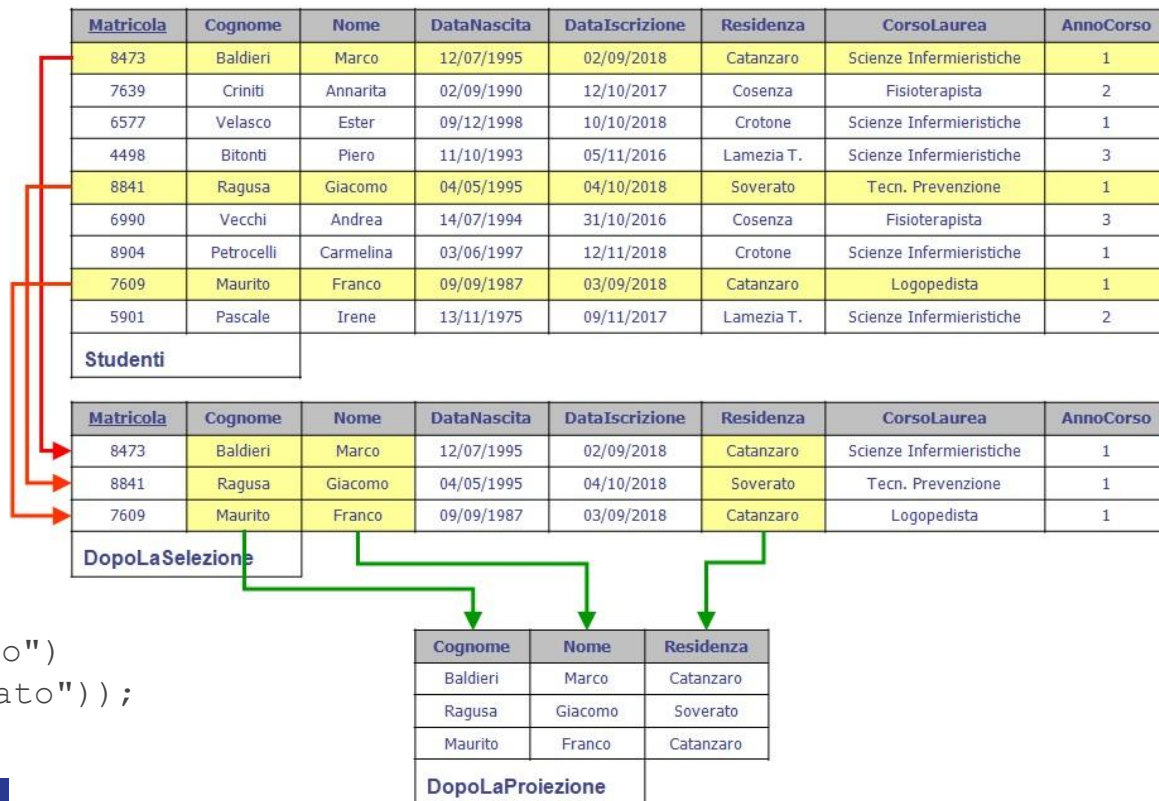
Cognome	Nome	Residenza
Baldieri	Marco	Catanzaro
Ragusa	Giacomo	Soverato
Maurito	Franco	Catanzaro

DopoLaProiezione

il comando select e l'algebra relazionale

Utilizzando i comandi di SQL, abbiamo invece l'istruzione seguente:

```
SELECT Cognome, Nome, Residenza
FROM Studenti
WHERE ((Residenza = "Catanzaro")
      OR (Residenza = "Soverato"));
```



il comando select e l'algebra relazionale

Mettendo a confronto i due comandi, possiamo notare dunque come il nostro **SELECT** corrisponda in termini di algebra relazionale, ad un'operazione di **RESTRICT** (la selezione delle righe di *Studenti* il cui campo *Residenza* assume il valore "Catanzaro" oppure "Soverato") seguita da un **PROJECT** (estrazione dei valori assunti dai campi *Cognome*, *Nome* e *Residenza* dalle righe trovate in precedenza).

PROJECT

RESTRICT *Studenti* **WHERE**

Residenza = "Catanzaro" **OR** *Residenza* = "Soverato"

OVER *Cognome*, *Nome*, *Residenza*

SELECT *Cognome*, *Nome*, *Residenza*

FROM *Studenti*

WHERE *Residenza* = "Catanzaro" **OR** *Residenza* = "Soverato";

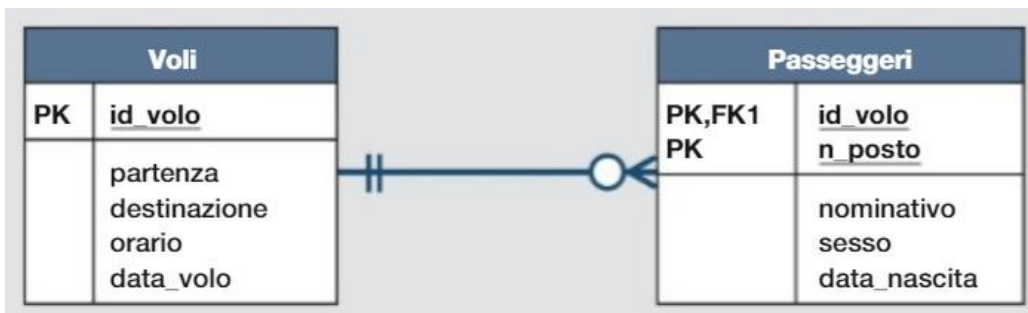
il comando select e l'algebra relazionale

Riprendiamo adesso l'esempio relativo ai voli aerei visto in precedenza per analizzare nel dettaglio le analogie tra algebra relazionale e linguaggio SQL:

Voli (id_volo, partenza, destinazione, orario, data_volo);

Passeggeri (id_volo, n_posto, nominativo, sesso, data_nascita);

il cui diagramma delle tabelle è schematizzato nella seguente figura:




il comando select e l'algebra relazionale

Vogliamo ottenere, per ogni passeggero maschio, il nominativo, il posto occupato e la data del volo di tutti i voli con la sigla 'AZ215'.

```
PROJECT (
  RESTRICT (
    JOIN Voli WITH Passeggeri ON id_volo
  ) WHERE id_volo = 'AZ215' AND sesso = 'M'
) OVER data_volo,nominativo,n_posto
```

Viene eseguita prima la **congiunzione** che mette insieme le due tabelle sulla base del campo in comune *id_volo*, poi la **selezione** delle righe in cui *id_volo*='AZ215' e sesso='M', infine la **proiezione** sui campi *data_volo*, *nominativo* e *n_posto*.



```
SELECT      data_volo,nominativo,n_posto
FROM        Voli,Passeggeri
WHERE id_volo = 'AZ215' AND sesso = 'M'
AND Voli.id_volo = Passeggeri.id_volo
```

il comando select e l'algebra relazionale

Possiamo notare che:

- La <lista campi> (*data_volo, nominativo, n_posto*) che segue la parola chiave **SELECT**, realizza l'operatore **PROJECT** dell'algebra relazionale.

```
PROJECT (  
  RESTRICT (  
    JOIN Voli WITH Passeggeri  
      ON id_volo  
  ) WHERE id_volo = 'AZ215'  
    AND sesso = 'M'  
)  
OVER data_volo, nominativo, n_posto
```

Lista campi

```
SELECT data_volo, nominativo, n_posto  
FROM Voli, Passeggeri  
WHERE id_volo = 'AZ215'  
  AND sesso = 'M'  
  AND Volo.id_volo = Passeggeri.id_volo
```

il comando select e l'algebra relazionale

Possiamo notare che:

- La <lista tabelle> (*Voli, Passeggeri*) che segue la parola chiave **FROM**, indica le tabelle coinvolte nella query e, se queste sono più di una, realizza di fatto l'operatore **JOIN** dell'algebra relazionale;

```
PROJECT (  
  RESTRICT (  
    JOIN Voli WITH Passeggeri  
      ON id_volo  
  ) WHERE id_volo = 'AZ215'  
    AND sesso = 'M'  
)  
OVER data_volo, nominativo, n_posto
```

Lista tabelle

```
SELECT data_volo, nominativo, n_posto  
FROM Voli, Passeggeri  
WHERE id_volo = 'AZ215'  
  AND sesso = 'M'  
  AND Volo.id_volo = Passeggeri.id_volo
```

il comando select e l'algebra relazionale

Possiamo notare che:

- Le singole componenti della condizione nella clausola **WHERE**, rappresentano il criterio di selezione espresso dall'operatore **RESTRICT** dell'algebra relazionale (`id_volo = 'AZ215' AND sesso = 'M'`).

```
PROJECT (  
  RESTRICT (  
    JOIN Voli WITH Passeggeri  
      ON id_volo  
  ) WHERE id_volo = 'AZ215'  
    AND sesso = 'M'  
)  
OVER data_volo, nominativo, n_posto
```

Condizioni
clausola WHERE

```
SELECT data_volo, nominativo, n_posto  
FROM Voli, Passeggeri  
WHERE id_volo = 'AZ215'  
  AND sesso = 'M'  
  AND Volo.id_volo = Passeggeri.id_volo
```

il comando select e l'algebra relazionale

Possiamo notare che:

NB: Se, come in questo caso, le tabelle coinvolte sono più di una, la clausola **WHERE**, rappresenta anche il criterio di *join* sulla base del quale debbono essere legate le righe di tali tabelle (`Volo.id_volo = Passeggeri.id_volo`).

```
PROJECT (  
  RESTRICT (  
    JOIN Voli WITH Passeggeri  
      ON id_volo  
  ) WHERE id_volo = 'AZ215'  
    AND sesso = 'M'  
)  
OVER data_volo, nominativo, n_posto
```

Criterio di JOIN

```
SELECT data_volo, nominativo, n_posto  
FROM Voli, Passeggeri  
WHERE id_volo = 'AZ215'  
AND sesso = 'M'  
AND Volo.id_volo = Passeggeri.id_volo
```

il comando select e l'algebra relazionale

Nella formulazione delle condizioni possono essere utilizzati i classici operatori di relazione =, >, <, >=, <=, <> e gli operatori booleani **AND**, **OR** e **NOT**.

L'ordine di applicazione degli operatori è il seguente: **NOT** viene applicato prima di **AND**, e **AND** prima di **OR**, altrimenti l'uso delle parentesi « (» e «) » consente di modificare la precedenza nella valutazione delle singole condizioni elementari nel contesto della condizione generale.

Le costanti di tipo stringa debbono essere racchiuse tra simboli « ' » (alcuni DBMS accettano anche il simbolo « " »).

il comando select e l'algebra relazionale

NB: La clausola **WHERE** non è obbligatoria; nel caso venga omessa e l'elenco dei campi nella clausola **SELECT** comprenda campi da più tabelle (riportate nella clausola **FROM**), il risultato sarà la proiezione (**PROJECT**) su tali campi del **prodotto cartesiano** delle tabelle coinvolte.

Ad es. la seguente interrogazione:

```
SELECT *  
FROM Voli, Passeggeri;
```

restituisce come risultato il prodotto cartesiano delle tabelle *Vol*i e *Passeggeri*.

il comando select e l'algebra relazionale

Osservazione: Una singola condizione di una clausola **WHERE** può specificare un insieme di valori a cui deve soddisfare uno stesso campo; l'operatore **IN** permette di verificare l'appartenenza del valore del campo rispetto ad uno specifico insieme.

Volendo ad es. conoscere il cognome e la qualifica di tutti i dipendenti dei dipartimenti D2, D4 e D5, si può formulare la seguente interrogazione:

```
SELECT Cognome, Qualifica  
FROM Personale  
WHERE CodDip IN ('D2', 'D4', 'D5');
```

equivalente a:

```
SELECT Cognome, Qualifica  
FROM Personale  
WHERE CodDip='D2' OR CodDip='D4' OR CodDip='D5';
```


il comando select e l'algebra relazionale

Osservazione: Una singola condizione di una clausola **WHERE** può specificare un insieme di valori a cui deve soddisfare uno stesso campo; l'operatore **IN** permette di verificare l'appartenenza del valore del campo rispetto ad uno specifico insieme.

Si noti che premettendo alla parola chiave **IN** l'operatore booleano **NOT** è possibile fare riferimento al complementare dell'insieme specificato.

La seguente interrogazione:

```
SELECT Cognome, Qualifica  
FROM Personale  
WHERE CodDip NOT IN ('D2','D4','D5');
```

recupera il nome e la qualifica di tutti i dipendenti che non appartengono ai dipartimenti D2, D4 e D5.