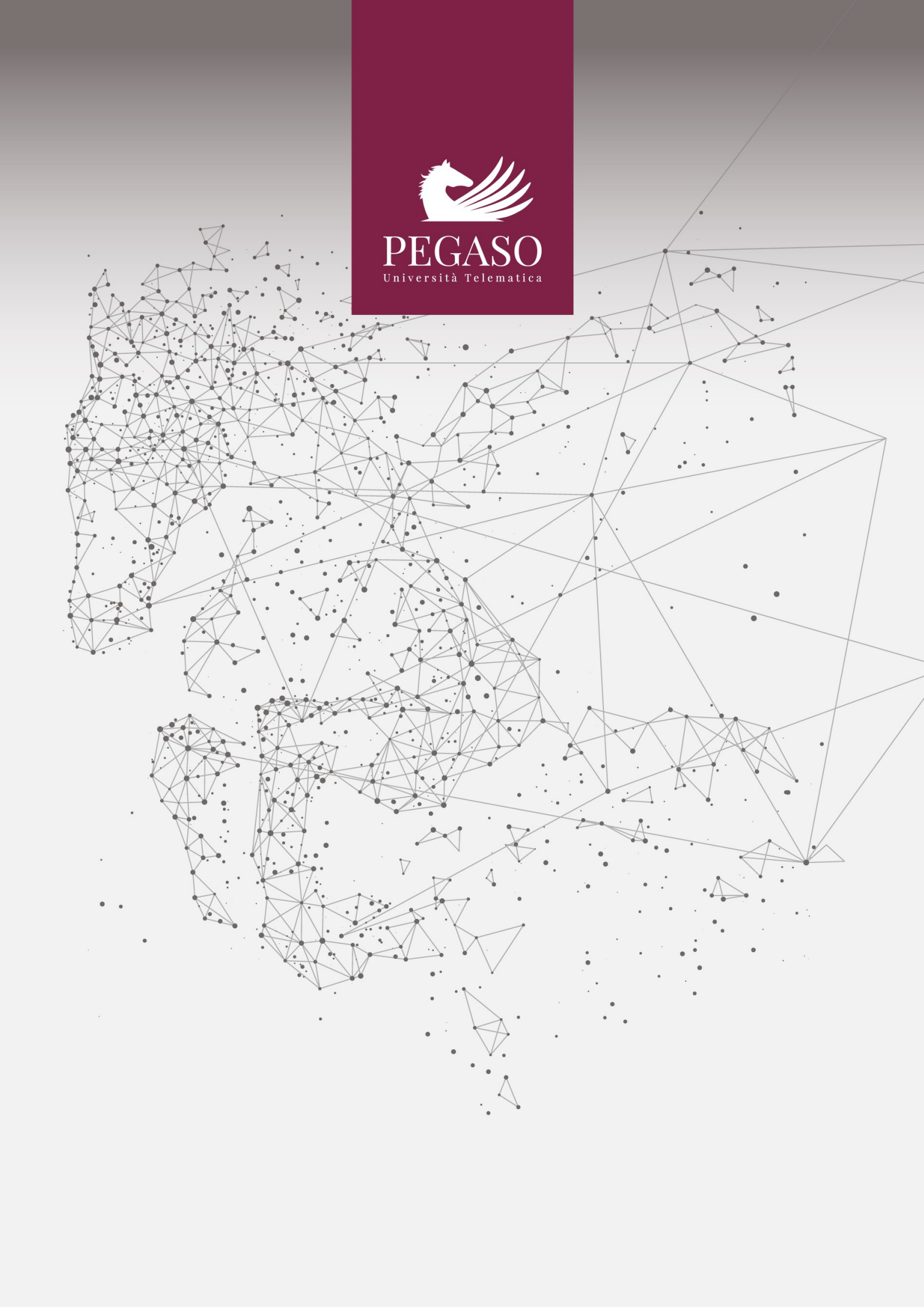
PEGASO
Università Telematica

Indice

1. Introduzione

Un join SQL è un'istruzione Structured Query Language (SQL) per combinare i dati da due serie di dati (ad esempio due tabelle). Come si sa, SQL è un linguaggio di programmazione creato per scopi speciali e progettato per la gestione delle informazioni in un sistema di gestione di database relazionali (RDBMS). La parola relazionale è quindi la chiave; specifica che il sistema di gestione del database è organizzato in modo tale che vi siano chiare relazioni definite tra diversi insiemi di dati.

In questa dispensa sono principalmente approfonditi, con esempi, i concetti riguardanti le varie tipologie di JOIN che possono essere utilizzate nelle queries.

2. Gestione dei duplicati

In questa sezione affrontiamo una situazione che si può presentare, ovvero tabelle che presentano righe duplicate. L'SQL mette a disposizione una serie di soluzioni per affrontare tale problema, al contrario dell'algebra relazionale dove una tabella viene vista come una relazione matematica e quindi come insieme di tuple tutte diverse tra loro. In SQL invece, in una tabella si possono avere più righe uguali tra loro ovvero righe aventi tutti gli attributi uguali (duplicati). È compito di chi scrive la query di gestire i duplicati (in SQL).

2.1 DISTINCT

Parola chiave DISTINCT da inserire subito dopo la clausola FROM. All'interno di una tabella, una colonna spesso contiene molti valori duplicati; e a volte si desidera solo elencare i diversi valori (distinti). La parola chiave DISTINCT permette quindi di recuperare le ennuple che sono tutte distinte tra loro.

Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma
Rossi	Roma

Figura 1: la tabella Impiegati.

Ad esempio, data la tabella di Figura 1, la query:

SELECT **DISTINCT** Cognome, Filiale

FROM Impiegati

La query torna il resource-set di Figura 2.

Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma

Figura 2: il resource-set risultato della query.

Data la tabella Impiegato di Figura 3, La query:

SELECT DISTINCT Dipart

FROM Impiegati

Nome	Cognome	Dipart	Ufficio	Stipendio	Città
Mario	Rossi	Amministrazione	10	45	Milano
Carlo	Bianchi	Produzione	20	36	Torino
Giovanni	Verde	Amministrazione	20	40	Roma
Franco	Neri	Distribuzione	16	45	Napoli
Carlo	Rossi	Direzione	14	80	Milano
Lorenzo	Gialli	Direzione	7	73	Genova
Paola	Rosati	Amministrazione	75	40	Venezia
Marco	Franco	Produzione	20	46	Roma

Figura 3: tabella impiegato.

Dà come risultato la tabella di Figura 4.

Dipart
Amministrazione
Produzione
Distribuzione
Direzione

Figura 4: resource-set risultato delle query sulla tabella Impiegato.

3. Join interni ed esterni

In questo paragrafo si illustrano le varie tipologie di JOIN che possono essere utilizzate nelle queries, fino ad ora utilizzate nella clausola WHERE. Infatti, si possono distinguere, tra le condizioni che compaiono in una interrogazione:

- Quelle che rappresentano condizioni di Join
- Quelle che rappresentano condizioni di selezione sulla riga

La sintassi per utilizzare direttamente il JOIN all'interno della SELECT è:

SELECT *AttrEspr* [[as]Alias],{*AttrEspr* [[as] *Alias*]}

FROM Tabella [[*as]Alias*]

{[*Tipojoin*] **JOIN** *Tabella [[as]Alias]* ***ON*** *Condizionedijoin}*

WHERE {*AltraCondizione*}

Da notare che:

- La condizione di JOIN non compare come argomento della clausola WHERE
- Viene spostata nell'ambito della clausola FROM associata alle tabelle coinvolte nel JOIN

Il parametro *TipoJoin* specifica qual è il tipo di JOIN da usare, tra i seguenti (attenzione però alla versione di SQL che è accettata dal DBMS):

- *Inner* (può essere omesso, è il default)
- *Left outer*
- *Right outer*
- *Full outer*

Esistono quindi quattro tipi base di join SQL: interno, sinistro, destro e completo. Il modo più semplice e intuitivo per spiegare la differenza tra questi quattro tipi è l'utilizzo di un diagramma di Venn, che mostra tutte le possibili relazioni logiche tra insiemi di dati.

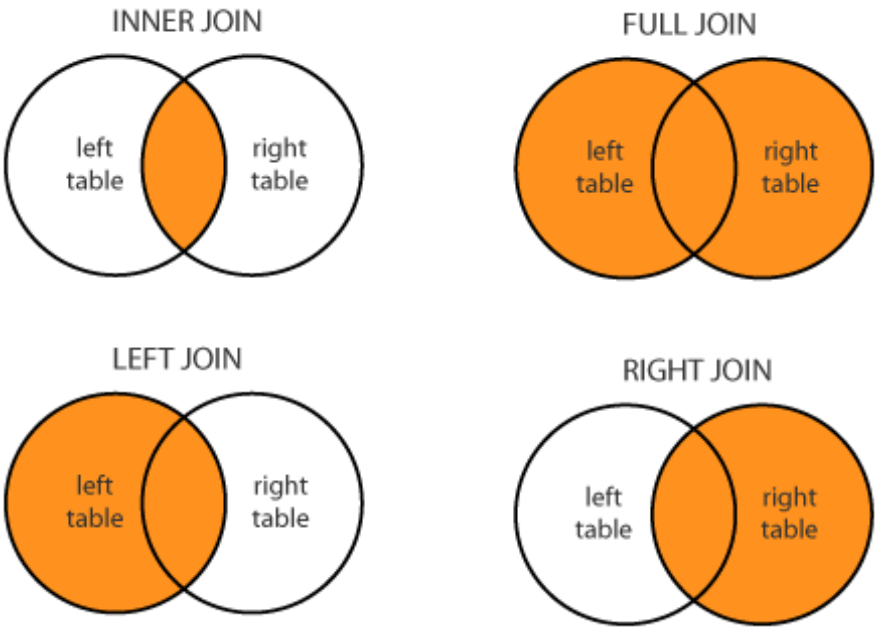

Figura 5: tipologie di Join con gli insiemi.

Il qualificatore *outer* è opzionale nell'istruzione. Per quanto riguarda i suddetti Join, la loro funzionalità è la seguente:

- **Left Join**: fornisce come risultato il join interno esteso con le righe della tabella che compare a sinistra per le quali non esiste una corrispondente riga nella tabella di destra.
- **Right Join**: simmetrico rispetto al left join.
- **Full Join:** restituisce il join interno esteso con le righe escluse di entrambe le tabelle.

In Figura 5 è illustrata la versione insiemistica dei vari JOIN, sfruttando l'approccio di relazione matematica ovvero di prodotto cartesiano.

L'inner JOIN rappresenta il tradizionale Theta JOIN dell'algebra relazionale. Riprendiamo *l'interrogazione 5*: estrarre i nomi degli impiegati e le città dove lavorano dallo schema relazionale di Figura 6 e Figura 7, formato dalle due relazioni Impiegato e Dipartimento.

Nome	Cognome	Dipart	Ufficio	Stipendio	Città
Mario	Rossi	Amministrazione	10	45	Milano
Carlo	Bianchi	Produzione	20	36	Torino
Giovanni	Verde	Amministrazione	20	40	Roma
Franco	Neri	Distribuzione	16	45	Napoli
Carlo	Rossi	Direzione	14	80	Milano
Lorenzo	Gialli	Direzione	7	73	Genova
Paola	Rosati	Amministrazione	75	40	Venezia
Marco	Franco	Produzione	20	46	Roma

Figura 6: tabella Impiegato.

Nome	Indirizzo	Città
Amministrazione	Via Livio 32	Milano
Produzione	Via Lavier 4	Torino
Distribuzione	Via Segre 9	Roma
Direzione	Via Livio 32	Milano
Ricerca	Via Venosa 6	Milano

Figura 7: tabella Dipartimento.

Query SQL (prima versione):

- SELECT Impiegato.Nome, Impiegato.Cognome, Dipartimento.Città
- FROM Impiegato,Dipartimento
- WHERE Impiegato.Dipart=Dipartimento.nome

Il JOIN è specificato indicando in modo esplicito attraverso le condizioni che esprimono il legame tra le diverse tabelle (clausola WHERE).

Impiegato.Nome	Impiegato.Cognome	Dipartimento.Città
Mario	Rossi	Milano
Carlo	Bianchi	Torino
Giovanni	Verde	Milano
Franco	Neri	Roma
Carlo	Rossi	Milano
Lorenzo	Gialli	Milano
Paola	Rosati	Milano
Marco	Franco	Torino

Figura 8: resource-set risultato della interrogazione 5.

Da notare che per gli attributi **Nome** e **Città** i quali presentano ambiguità poiché nomi di attributi uguali si utilizza il «.». Per tale motivo, la suddetta query l'avevamo riscritta nel modo seguente:

```
Select I.Nome, Cognome,D.Città
FROM Impiegato as I, Dipartimento as D
WHERE Dipart=D.Nome
```

Tornando al JOIN, la precedente interrogazione può essere scritta nel modo seguente (Join interno o inner Join):

```
SELECT I.Nome,Cognome, D.Città
FROM Impiegato I JOIN Dipartimento D ON Dipart=D.Nome
```

Con questo tipo di Join:

- Le righe che vengono coinvolte sono un sottoinsieme delle righe di ciascuna tabella
- Alcune righe possono essere eliminate quando non c'è corrispondenza

4. Applicazioni

Consideriamo le due tabelle Guidatore e Automobile di Figura 9 e Figura 10.

Nome	Cognome	NroPatente
Mario	Rossi	VR 2030020Y
Carlo	Bianchi	PZ 1012436B
Marco	Neri	AP 4544442R

Figura 9: tabella guidatore.

Targa	Marca	Modello	NroPatente
KB 574 WW	Fiat	Punto	VR 2030020Y
GA 652 FF	Fiat	Panda	VR 2030020Y
BJ 747 XX	Lancia	Ypsilon	PZ 1012436B
ZB 421 JJ	Fiat	Uno	MI 2020030U

Figura 10: tabella automobile.

Interrogazione 13: estrarre i guidatori con le automobili loro associate, mantenendo nel risultato anche i guidatori senza automobile. La query è la seguente:

- SELECT nome, Cognome, G.NroPatente, Targa, Marca, Modello
- FROM Guidatore G **LEFT JOIN** Automobile A **ON**
- (G.NroPatente=A.NroPatente)

Il risultato è illustrato in Figura 11.

Nome	Cognome	G.NroPatente	Targa	Marca	Modello
Mario	Rossi	VR 2030020Y	KB 574 WW	Fiat	Punto
Mario	Rossi	VR 2030020Y	GA 652 FF	Fiat	Panda
Carlo	Bianchi	PZ 1012436B	BJ 747 XX	Lancia	Ypsilon
Marco	Neri	AP 4544442R	NULL	NULL	NULL

Figura 11: resourse-set interrogazione 13.

Interrogazione 14: estrarre tutti i guidatori e tutte le auto, mostrando tutte le relazioni esistenti tra di esse. La soluzione è la seguente:

```
SELECT nome, Cognome, G.NroPatente, Targa, Marca, Modello
FROM Guidatore G FULL JOIN Automobile A ON
(G.NroPatente=A.NroPatente)
```

Il risultato è illustrato in Figura 12.

Nome	Cognome	G.NroPatente	Targa	Marca	Modello
Mario	Rossi	VR 2030020Y	KB574 WW	Fiat	Punto
Mario	Rossi	VR 2030020Y	GA 652 FF	Fiat	Panda
Carlo	Bianchi	PZ 1012436B	BJ 747 XX	Lancia	Ypsilon
Marco	Neri	AP 4544442R	NULL	NULL	NULL
NULL	NULL	NULL	ZB 421 JJ	Fiat	Uno

Figura 12: resource -set interrogazione 14.

Bibliografia

- Atzeni P., Ceri S., Fraternali P., Paraboschi S., Torlone R. (2018). Basi di Dati. McGraw-Hill Education.
- Batini C., Lenzerini M. (1988). Basi di Dati. In Cioffi G. and Falzone V. (Eds). Calderini. Seconda Edizione.

Sitografia

- http://lnx.poggiodelpapa.com/linguaggi/sql_oracle/combinazioni-tabelle.html
- https://www.edatlas.it/scarica/informatica/info_prog_database_sql/basi-sqlMOL4-1.pdf