

# Basi di dati

Corso di laurea in Ingegneria Informatica  
Scuola di Ingegneria — Università di Pisa

---

**Oracle MySQL**  
A.A. 2017-2018

Ing. Francesco Pistolesi

*Postdoctoral Researcher*  
Data Science and Engineering Lab  
Dipartimento di Ingegneria dell'Informazione  
[francesco.pistolesi@iet.unipi.it](mailto:francesco.pistolesi@iet.unipi.it)

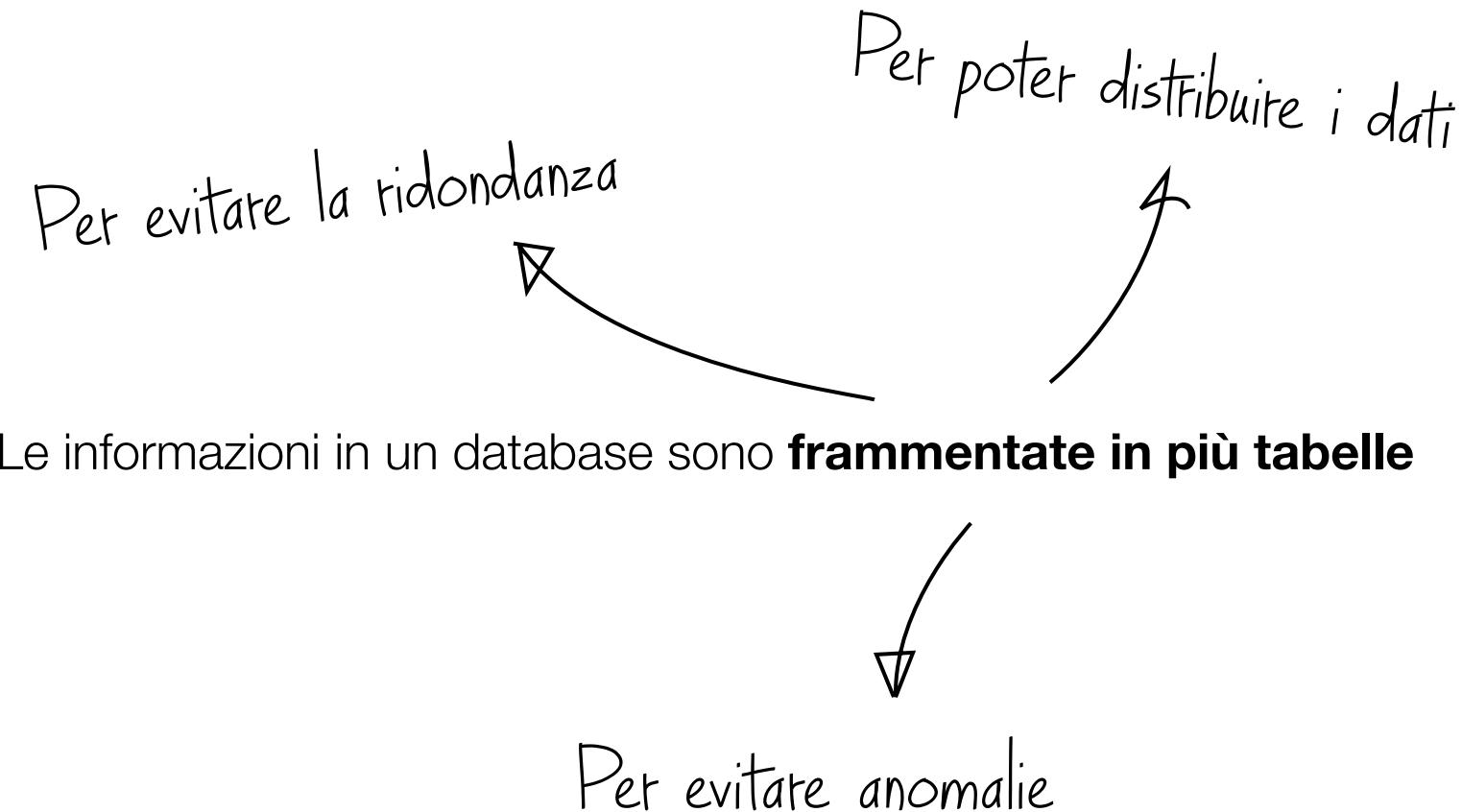
# Multi-table querying

---



# Multi-table querying

---



# Il nostro database, per ora...

**PAZIENTE**

 CodFiscale	Cognome	Nome	Sesso	Citta	DataNascita	Reddito
LPRNTA	Lepre	Antonella	F	Pisa	1958-05-03	1200
GTTRTA	Gatto	Rita	F	Firenze	1983-10-30	1400
MNZMBT	Manzi	Umberto	M	Pisa	1949-07-12	1350
CPRLND	Capra	Leonardo	M	Milano	1967-09-20	1600

**MEDICO**

 Matricola	Cognome	Nome	Specializzazione	Parcella	Citta
18339	Verdi	Paolo	Ortopedia	150	Pisa
35512	Rossi	Marta	Ortopedia	120	Pisa
16220	Gialli	Rita	Cardiologia	135	Roma
29858	Neri	Rino	Dermatologia	110	Siena

**VISITA**

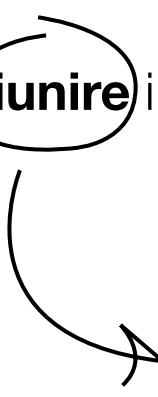
Medico	Paziente		Data		Mutuata
35512	GTTRTA		2013-01-15		1
29858	MNZMBT		2012-11-30		0
18339	CPRLND		2012-10-28		1
35512	GTTRTA		2013-02-20		1
16220	MNZMBT		2013-01-25		0
35512	LPRNTA		2013-03-01		0
18339	CPRLND		2013-01-01		0

Nelle slide seguenti alcune colonne non sono riportate per carenza di spazio

# Query su più tabelle

---

Necessità di **riunire** i pezzi dell'informazione



Le interrogazioni hanno bisogno di manipolare l'informazione completa

# Join

---



# Join: a cosa serve?

---

natural join

inner join

left outer join

Il join dà la possibilità di **combinare i dati di più tavelle**

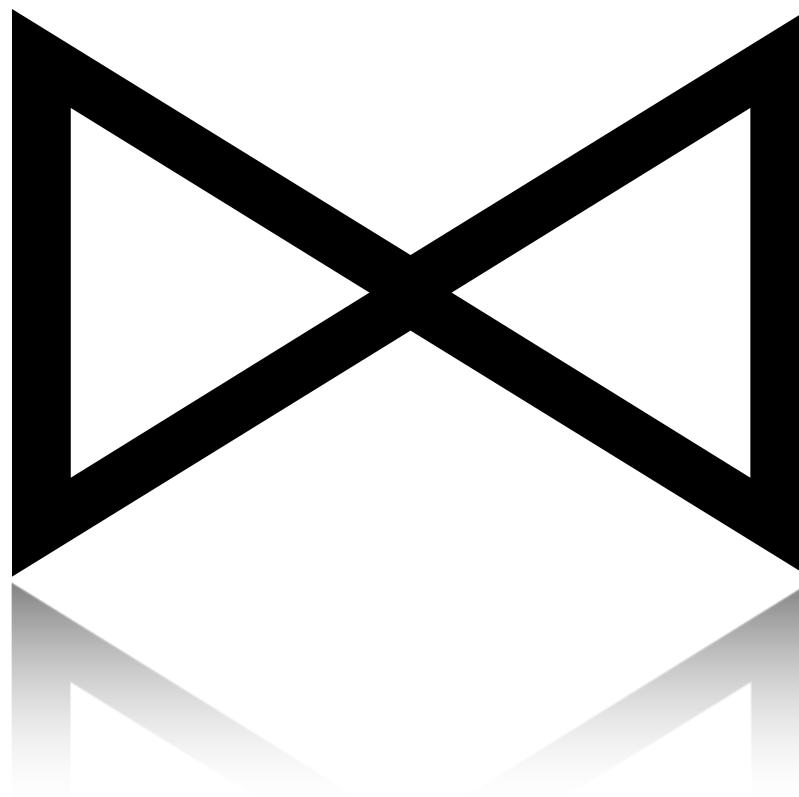
cross join

right outer join

self join

# Inner join

---



# Inner join

---

affianca le due record per formarne uno solo

Date due tabelle, combina ogni record della prima con tutti i record della seconda **che verificano una determinata condizione**

predicato del join

# Inner join: esempio

---

Indicare nome e cognome dei medici che hanno effettuato almeno una visita



# Inner join: esempio

Indicare nome e cognome dei medici che hanno effettuato almeno una visita

**MEDICO**

Matricola	Cognome	Nome	Specializzazione
18339	Verdi	Paolo	Ortopedia
35512	Rossi	Marta	Ortopedia
16220	Gialli	Rita	Cardiologia
29858	Neri	Rino	Dermatologia

**VISITA**

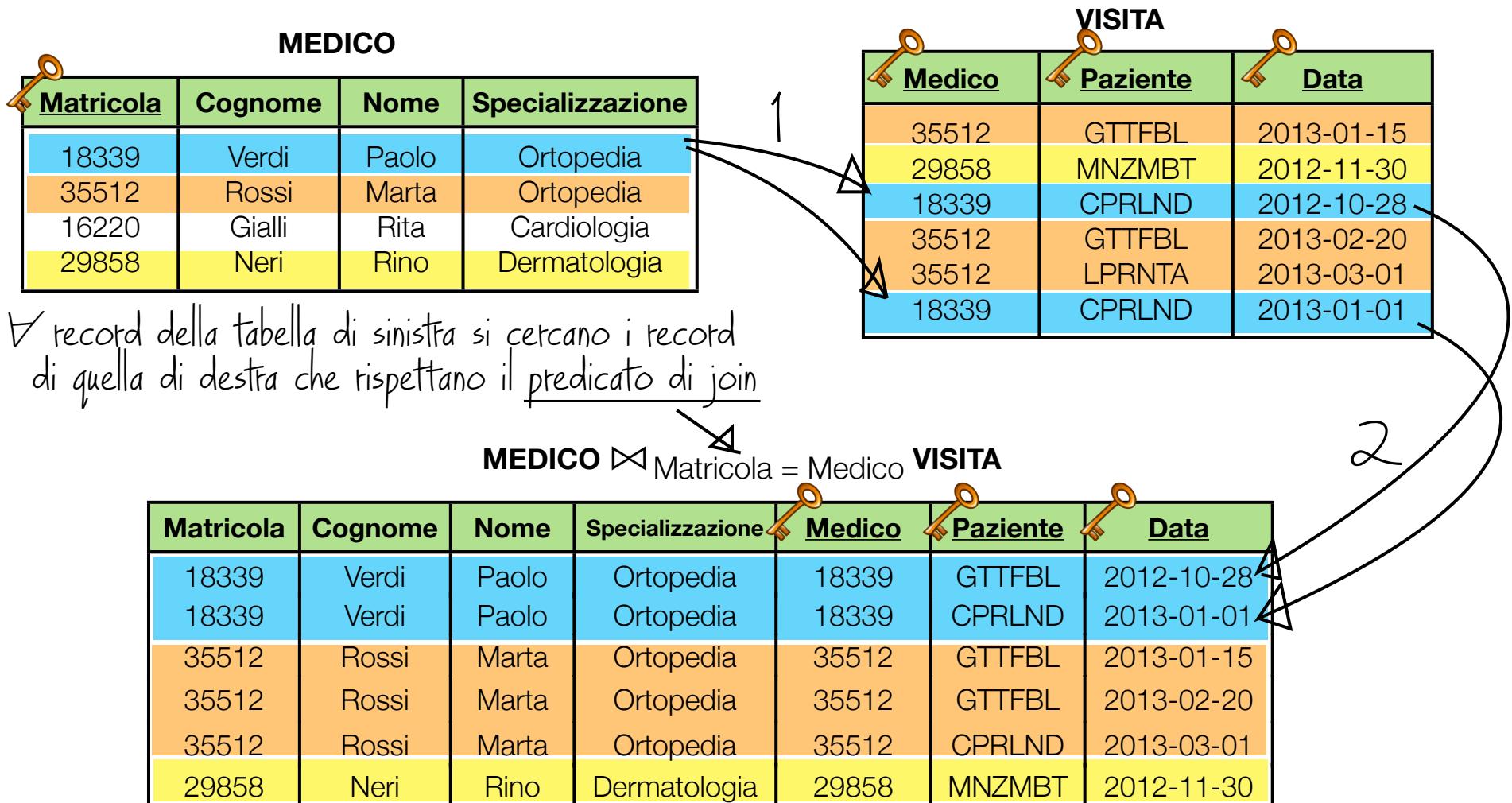
Medico	Paziente	Data
35512	GTTRTA	2013-01-15
29858	MNZMBT	2012-11-30
18339	CPRLND	2012-10-28
35512	GTTRTA	2013-02-20
16220	MNZMBT	2013-01-25
35512	LPRNTA	2013-03-01
18339	CPRLND	2013-01-01



L'informazione deve ricongiungersi...

# Inner join: come funziona

Indicare nome e cognome dei medici che hanno effettuato almeno una visita



# Theta join in MySQL

Indicare nome e cognome dei medici che hanno effettuato almeno una visita

alias per riferire le tabelle

```
SELECT DISTINCT M.Nome, M.Cognome  
FROM Visita V INNER JOIN Medico M ON V.Medico = M.Matricola;
```



V.Medico è la matricola  
del medico nella tabella Visita



predicato di join

# Join naturale

---

Combina i record della prima tabella con i record della seconda tabella  
aventi **valori uguali sugli attributi omonimi.**



# Join naturale: esempio

Indicare nome e cognome dei medici che hanno effettuato almeno una visita

*stesso nome!*

MEDICO			
<u>ID_Medico</u>	Cognome	Nome	Specializzazione
18339	Verdi	Paolo	Ortopedia
35512	Rossi	Marta	Ortopedia
16220	Gialli	Rita	Cardiologia
29858	Neri	Rino	Dermatologia

VISITA		
<u>ID_Medico</u>	Paziente	Data
35512	GTTRTA	2013-01-15
29858	MNZMBT	2012-11-30
18339	CPRLND	2012-10-28
35512	GTTRTA	2013-02-20
16220	MNZMBT	2013-01-25
35512	LPRNTA	2013-03-01
18339	CPRLND	2013-01-01

**MEDICO  $\bowtie$  VISITA**

*compare una sola volta*

<u>ID_Medico</u>	Cognome	Nome	Specializzazione	<u>Paziente</u>	<u>Data</u>
35512	Rossi	Marta	Ortopedia	GTTRTA	2013-01-15
29858	Neri	Rino	Dermatologia	MNZMBT	2012-11-30
18339	Verdi	Paolo	Ortopedia	CPRLND	2012-10-28
35512	Rossi	Marta	Ortopedia	GTTRTA	2013-02-20
35512	Rossi	Marta	Ortopedia	LPRNTA	2013-03-01
18339	Verdi	Paolo	Ortopedia	CPRLND	2013-01-01

# Join naturale in MySQL

---

Indicare nome e cognome dei medici che hanno effettuato almeno una visita

```
SELECT DISTINCT M.Nome, M.Cognome  
FROM Visita V NATURAL JOIN Medico M;
```

il predicato è implicito:  
 $V.ID\_Medico = M.ID\_Medico$



Nel database Clinica questo join non è fattibile, perché  $ID\_Medico$  si chiama "Matricola" nella tabella Medico, mentre nella tabella Visita la matricola del medico che effettua la visita è l'attributo "Medico": non c'è omomimia nei nomi degli attributi, quindi non si può fare un natural join fra Visita e Medico.

# Join naturale: da non dimenticare

---



# Sottoinsiemi di attributi omonimi

---

Attenzione, questo non è  
un join naturale!

MySQL offre la possibilità di imporre l'uguaglianza  
di solamente **alcune coppie di attributi omonimi** tramite **USING**

Ne discuteremo al bisogno...

# Prodotto cartesiano

---

Restituisce **tutte le possibili combinazioni**  
di ciascun record della prima tabella con tutti i record della seconda tabella

il join può essere pensato come un  
prodotto cartesiano con condizione

# Prodotto cartesiano in MySQL

---

Indicare il numero di pazienti aventi nome uguale ad almeno un medico della clinica

**PAZIENTE × MEDICO**

P.CodFiscale	P.Cognome	P.Nome	P. ...	M.Matricola	M.Cognome	M.Nome	M. ...
LPRNTA	Lepre	Antonella		18339	Verdi	Paolo	
LPRNTA	Lepre	Antonella		35512	Rossi	Marta	
LPRNTA	Lepre	Antonella		16220	Gialli	Rita	
LPRNTA	Lepre	Antonella		29858	Neri	Rino	
GTTFB	Gatto	Rita		18339	Verdi	Paolo	
GTTFB	Gatto	Rita		35512	Rossi	Marta	
GTTFB	Gatto	Rita		16220	Gialli	Rita	
GTTFB	Gatto	Rita		29858	Neri	Rino	
MNZMBT	Manzi	Umberto		18339	Verdi	Paolo	
MNZMBT	Manzi	Umberto		35512	Rossi	Marta	
MNZMBT	Manzi	Umberto		16220	Gialli	Rita	
MNZMBT	Manzi	Umberto		29858	Neri	Rino	
CPRLND	Capra	Leonardo		18339	Verdi	Paolo	
CPRLND	Capra	Leonardo		35512	Rossi	Marta	
CPRLND	Capra	Leonardo		16220	Gialli	Rita	
CPRLND	Capra	Leonardo		29858	Neri	Rino	

# Soluzione

---

Indicare il numero di pazienti aventi nome uguale ad almeno un medico della clinica

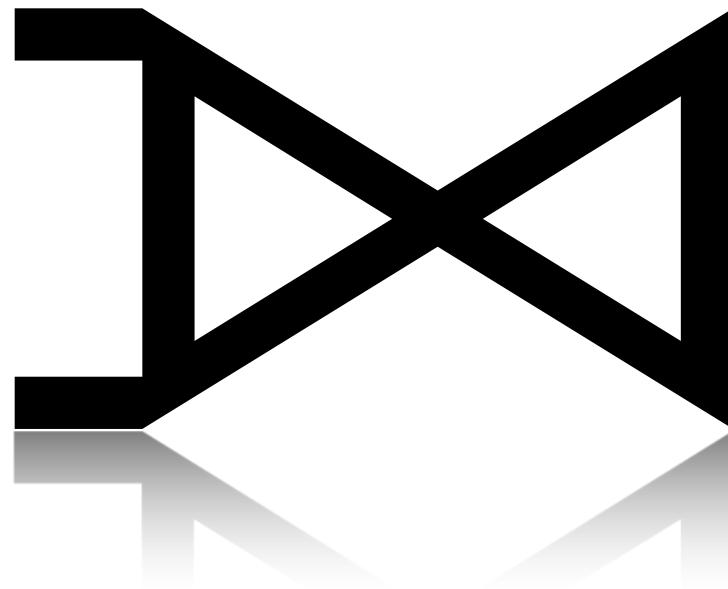
```
SELECT COUNT(DISTINCT P.CodFiscale)  
FROM Paziente P CROSS JOIN Medico M  
WHERE P.Nome = M.Nome;
```



serve `distinct` perché un paziente può avere  
il nome uguale a più di un medico

# Join esterni

---



# Join esterni

---

Date due tabelle, combinano ogni record della prima con tutti i record della seconda che soddisfano una condizione,  
**mantenendo tutti i record di una delle due tabelle**

diverso da inner join che invece  
scarta i record che non fanno join

# Join esterni

---



# Join esterno sinistro

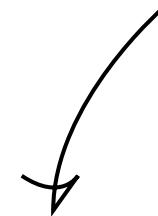
---

Combina ogni record della tabella sinistra con i record della tabella destra che soddisfano una condizione, **mantenendo tutti i record della tabella di sinistra**

# Join esterno sinistro: esempio

---

Indicare le visite effettuate da medici che **non lavorano più** presso la clinica



la loro anagrafica non è presente

# Join esterno sinistro: esempio

VISITA		
Medico	Paziente	Data
22222	GTTRTA	2010-01-15
18339	CPRLND	2012-10-28
35512	GTTFBL	2013-02-20
18339	CPRLND	2013-01-01



MEDICO			
Matricola	Cognome	Nome	Specializzazione
18339	Verdi	Paolo	Ortopedia
35512	Rossi	Marta	Ortopedia
16220	Gialli	Rita	Cardiologia

VISITA  $\bowtie$  Medico = Matricola MEDICO

Medico	Paziente	Data	Matricola	Cognome	Nome	Specializzazione
22222	GTTRTA	2010-01-15	NULL	NULL	NULL	NULL
18339	CPRLND	2012-10-28	18339	Verdi	Paolo	Ortopedia
35512	GTTFBL	2013-02-20	35512	Rossi	Marta	Ortopedia
18339	CPRLND	2013-01-01	18339	Verdi	Paolo	Ortopedia

Siamo i null, veniamo in pace, sempre...

# Join esterno sinistro in MySQL

---

Indicare le visite effettuate da medici che **non lavorano più** presso la clinica

meglio proiettare V.\*



```
SELECT *
FROM Visita V LEFT OUTER JOIN Medico M ON V.Medico = M.Matricola
WHERE M.Matricola IS NULL;
```

Se si utilizza solamente \*, in ogni record si ha la matricola del medico due volte (una deriva da V e l'altra da M) e tutti i valori null per i record non joinabili.

# Join esterno sinistro: significato

---

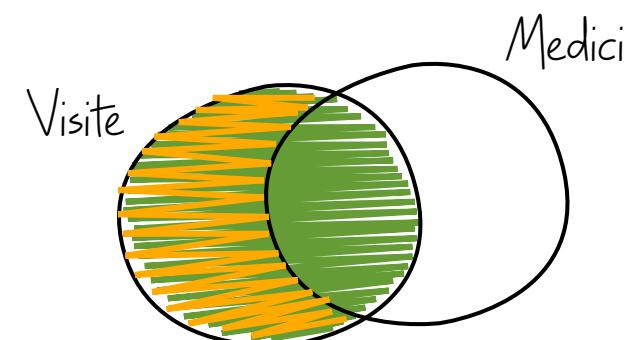
Indicare le visite effettuate da medici che **non lavorano più** presso la clinica

`SELECT V.*`

`FROM Visita V LEFT OUTER JOIN Medico M ON V.Medico = M.Matricola`

`WHERE M.Matricola IS NULL;`

Left outer join ricava la parte verde. Mediante il predicato where è possibile ottenere la parte arancio, cioè i soli record non joinabili



# Join esterno sinistro

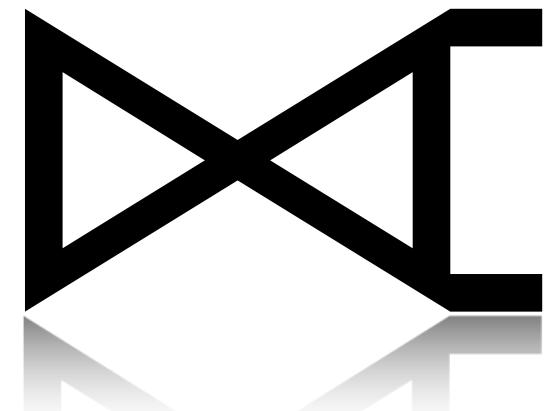
---



# Join esterno destro

---

Combina ogni record della tabella destra con i record della tabella sinistra che soddisfano una condizione, **mantenendo tutti i record della tabella di destra**



# Join esterno destro: esempio

---

Indicare cognome e specializzazione dei medici della clinica che  
**non hanno ancora effettuato visite.**

# Join esterno destro

VISITA		
Medico	Paziente	Data
22222	GTTRTA	2010-01-15
18339	CPRLND	2012-10-28
35512	GTTFBL	2013-02-20
18339	CPRLND	2013-01-01



MEDICO			
Matricola	Cognome	Nome	Specializzazione
18339	Verdi	Paolo	Ortopedia
35512	Rossi	Marta	Ortopedia
16220	Gialli	Rita	Cardiologia

VISITA $\bowtie$ Medico = Matricola MEDICO						
Medico	Paziente	Data	Matricola	Cognome	Nome	Specializzazione
18339	CPRLND	2012-10-28	18339	Verdi	Paolo	Ortopedia
18339	CPRLND	2013-01-01	18339	Verdi	Paolo	Ortopedia
35512	GTTRTA	2013-02-20	35512	Rossi	Marta	Ortopedia
NULL	NULL	NULL	16220	Gialli	Rita	Cardiologia

# Join esterno destro in MySQL

---

Indicare cognome e specializzazione dei medici della clinica che  
**non hanno ancora effettuato visite.**

```
SELECT DISTINCT M.Cognome, M.Specializzazione  
FROM Visita V RIGHT OUTER JOIN Medico M ON V.Medico = M.Matricola  
WHERE V.Medico IS NULL;
```

distinct può essere omesso, anche se in generale  
un duplicato è un altro medico

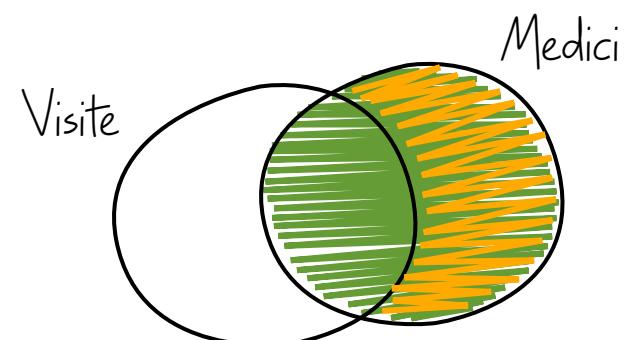
# Join esterno destro: significato

---

Indicare cognome e specializzazione dei medici della clinica che  
**non hanno ancora effettuato visite.**

```
SELECT M.Cognome, M.Specializzazione  
FROM Visita V RIGHT OUTER JOIN Medico M ON V.Medico = M.Matricola  
WHERE V.Medico IS NULL
```

Right outer join ricava la parte verde.  
Mediante il predicato where è possibile ottenere  
la parte arancio, cioè i soli record non joinabili



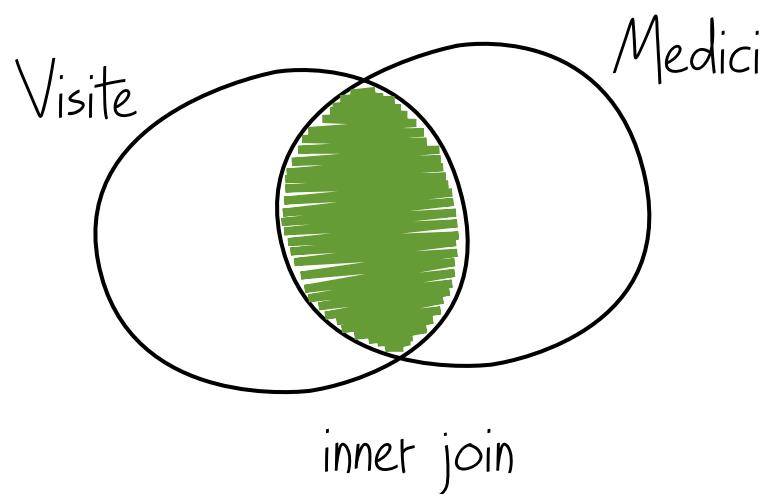
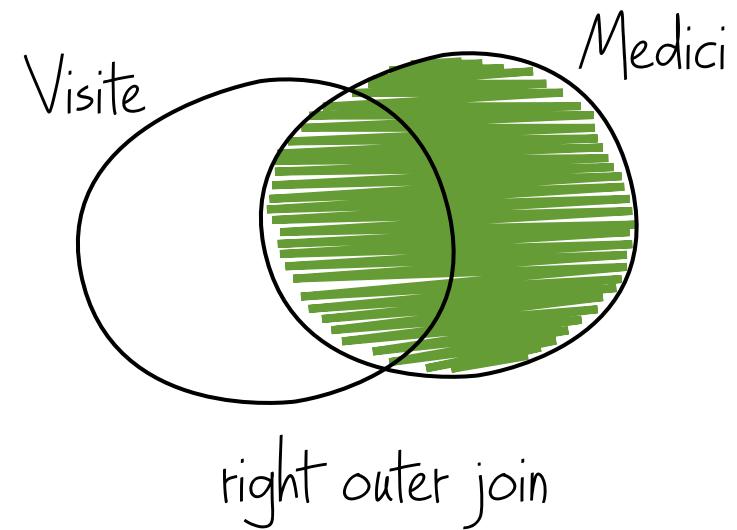
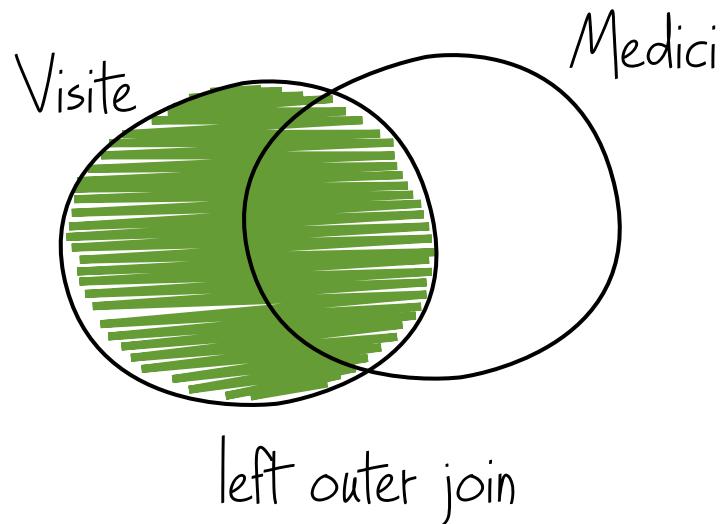
## Join esterno destro

---

Il join esterno DESTRO  
mantiene tutti i record  
della tabella di destra mettendo  
NULL A SINISTRA  
per i record non joinabili

# Ecce scheminum

---



# Query con join e condizioni sui record

---

La clausola WHERE contiene solitamente condizioni su attributi  
che **non compaiono nel predicato di join**

vuol dire che c'è un ulteriore filtraggio dei record  
effettuato DOPPO l'esecuzione del join

# Query con join e condizioni sui record: esempio

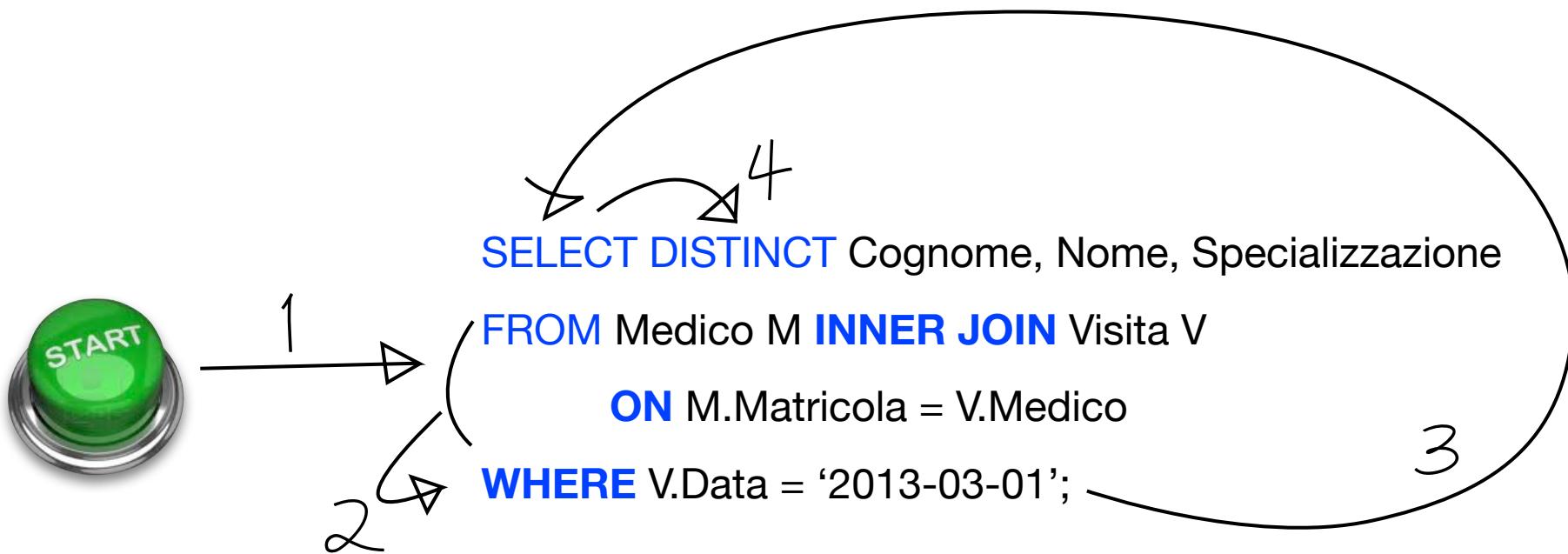
---

Indicare nome, cognome e specializzazione dei medici che hanno visitato almeno un paziente **il giorno 1° Marzo 2013**

```
SELECT DISTINCT M.Nome, M.Cognome, M.Specializzazione  
FROM Medico M INNER JOIN Visita V ON M.Matricola = V.Medico  
WHERE V.Data = '2013-03-01';
```

la data della visita rappresenta un ulteriore filtraggio

# Query con join e condizioni sui record: processazione



per prima cosa si esegue il join

# Query con join e condizioni sui record: processazione



Visita  $\bowtie_{\text{Medico} = \text{Matricola}}$  Medico

Medico	Paziente	Data	Matricola	Cognome	Nome	Specializzazione
35512	GTTRTA	2013-03-01	35512	Rossi	Marta	Ortopedia
29858	MNZMBT	2012-11-30	29858	Neri	Rino	Dermatologia
18339	CPRLND	2012-10-28	18339	Verdi	Paolo	Ortopedia
35512	GTTRTA	2013-02-20	35512	Rossi	Marta	Ortopedia
35512	LPRNTA	2013-03-01	35512	Rossi	Marta	Ortopedia
18339	CPRLND	2013-01-01	18339	Verdi	Paolo	Ortopedia

Data = '2013-03-01'

Medico	Paziente	Data	Matricola	Cognome	Nome	Specializzazione
35512	LPRNTA	2013-03-01	35512	Rossi	Marta	Ortopedia
	GTTRTA	2013-03-01	35512	Rossi	Marta	Ortopedia

Proiezione

Cognome	Nome	Specializzazione
Rossi	Marta	Ortopedia
Rossi	Marta	Ortopedia



Distinct

Cognome	Nome	Specializzazione
Rossi	Marta	Ortopedia

# Join multiplo

---

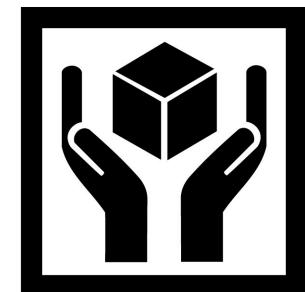


# Join su più di due tabelle

---

Il join può essere fatto su un numero di tabelle **maggiore di due**

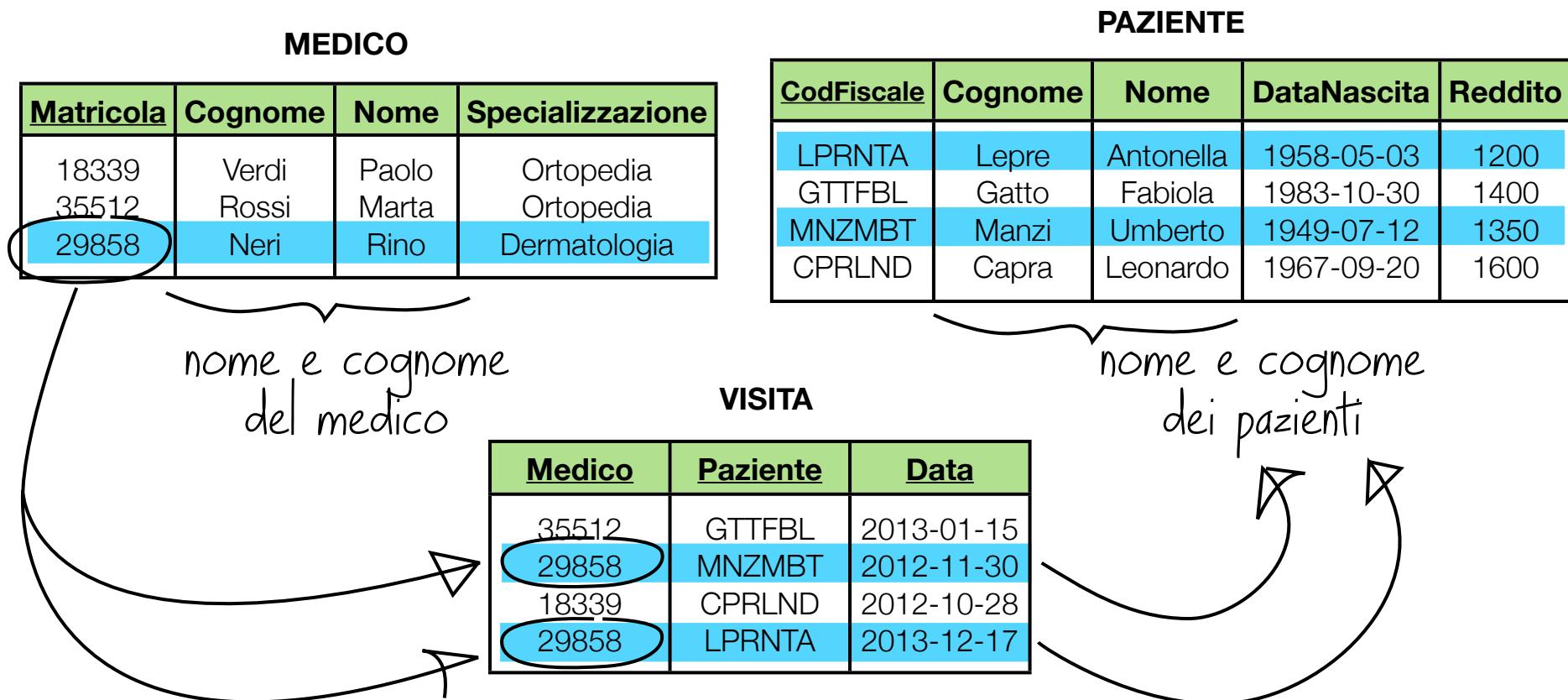
*con attenzione*



**HANDLE  
WITH CARE**

# Join multiplo: esempio

Indicare nome e cognome dei pazienti visitati dal dott. Rino Neri



# Join multiplo in MySQL

---

Indicare nome e cognome dei pazienti visitati dal dott. Rino Neri

```
SELECT DISTINCT P.Nome, P.Cognome  
FROM Paziente P  
      INNER JOIN  
        Visita V ON P.CodFiscale = V.Paziente  
      INNER JOIN  
        Medico M ON V.Medico = M.Matricola  
      WHERE M.Nome = 'Rino'  
            AND M.Cognome = 'Neri';
```

ATTENTION!

se non si usassero le variabili P ed M avremmo ambiguità:  
quali sono il nome e cognome del medico e quali quelli del paziente?

# Ambiguità

---



# Ambiguità

---

Si verifica se nel risultato ci sono **più attributi con lo stesso nome**

si rende necessaria la ridenominazione

# Ambiguità: ridenominazione nella proiezione

---

Indicare nome e cognome dei pazienti visitati nel mese Dicembre 2013,  
e il nome e cognome dei medici che li hanno visitati

```
SELECT DISTINCT P.Nome AS NomePaziente, P.Cognome AS CognomePaziente,  
M.Nome AS NomeMedico, M.Cognome AS CognomeMedico,  
FROM Paziente P INNER JOIN Visita V ON P.CodFiscale = V.Paziente  
INNER JOIN Medico M ON V.Medico = M.Matricola  
WHERE YEAR(V.Data) = 2013 AND MONTH(V.Data) = 12;
```

NomePaziente	CognomePaziente	NomeMedico	CognomeMedico
Antonella	Lepre	Rino	Neri

# Ridenominazione: quali nomi assegnare?

---

NomePaziente

ParcellaMediaOtorini

NumeroCardiologi

Molto apprezzato l'uso dell'**upper camel case**

RedditoMassimoPazienti

NumeroVisiteMaggio



# Ridenominazione: nomi da evitare

---

abcd

pazzienti

quellocheVuole

togliete

contrazzioni

howmanypaz



patcmax

tighebuone

pazvisit3

medOto

# Ambiguità: ridenominazione nel FROM

---

Anche gli **alias** dichiarati nel FROM effettuano una ridenominazione



# Ambiguità: ridenominazione non necessaria

---

Indicare il numero di visite effettuate dalla dottoressa Marta Rossi

```
SELECT COUNT(*)  
FROM Visita INNER JOIN Medico ON Medico = Matricola  
WHERE Nome = 'Marta' AND Cognome = 'Rossi';
```

non c'è ambiguità, ridenominazione nel from non necessaria,  
ma comunque consigliata per leggibilità

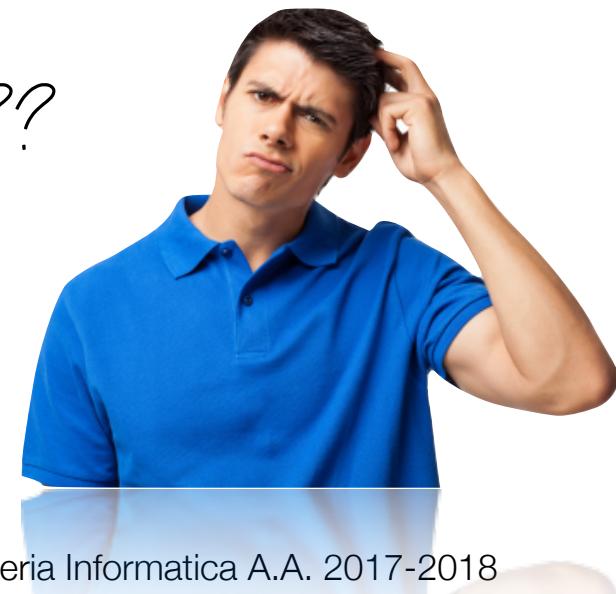
# Ambiguità: ridenominazione necessaria

---

Indicare il numero di volte che la dottoressa Marta Rossi ha visitato il paziente Umberto Manzi

```
SELECT COUNT(*)  
FROM Paziente INNER JOIN Visita ON CodFiscale = Paziente  
    INNER JOIN Medico ON Medico = Matricola  
WHERE Nome = ??? AND Cognome = ???  
    AND Nome = ??? AND Cognome = ???
```

nome e cognome sono ambigui,  
non si riesce a completare la query



# Ambiguità: ridenominazione necessaria

---

Indicare il numero di volte che la dottoressa Marta Rossi ha visitato il paziente Umberto Manzi

```
SELECT COUNT(*)  
FROM Paziente P INNER JOIN Visita V ON P.CodFiscale = V.Paziente  
    INNER JOIN Medico M ON V.Medico = M.Matricola  
WHERE M.Nome = 'Marta' AND M.Cognome = 'Rossi'  
    AND P.Nome = 'Umberto' AND P.Cognome = 'Manzi';
```

con ridenominazione mediante alias

# Ridenominazione: a rule of thumb

---



Per leggibilità, specialmente  
nelle query complesse,  
usate SEMPRE  
la ridenominazione nel from

# Self join

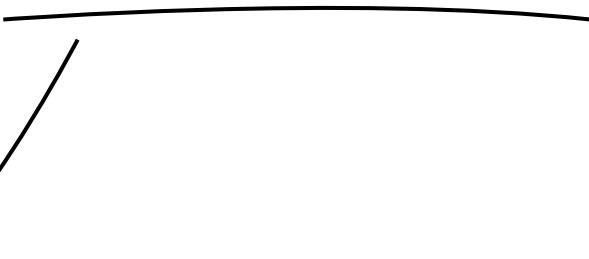
---



# Self join

---

Combina i record di una tabella con i record **della stessa tabella**  
che rispettano una determinata condizione

  
più articolata rispetto a una semplice uguaglianza

# Self join: esempio

---

Indicare il codice fiscale dei pazienti che sono stati visitati  
**più di una volta da almeno un medico** della clinica, nel mese corrente

```
SELECT DISTINCT V1.Paziente
FROM Visita V1 INNER JOIN Visita V2
ON (
    V2.Medico = V1.Medico
    AND V2.Paziente = V1.Paziente
    AND V2.Data <> V1.Data
)
WHERE MONTH(V1.Data) = MONTH(CURRENT_DATE)
    AND MONTH(V2.Data) = MONTH(CURRENT_DATE);
```

Se un paziente è stato visitato più volte nel mese corrente da almeno un medico, allora esiste almeno un medico che ha visitato il paziente come minimo due volte, ovviamente in date diverse perché, nella stessa data, un medico non può visitare lo stesso paziente più volte (vincolo di chiave primaria su Visita). Quindi, se per ogni visita v1 si generano i record ottenuti, ciascuno, affiancando v1 a un'altra visita v2 fatta dallo stesso paziente con lo stesso medico in date diverse, ogni visita v1 che fa join almeno una volta con v2 è relativa a un paziente che è stato nuovamente visitato, in v2, dallo stesso medico.

# Self join: cosa accade (I)

Indicare il codice fiscale dei pazienti che sono stati visitati  
**più di una volta da almeno un medico** della clinica, nel mese corrente

**VISITA V1**

Medico	Paziente	Data	Mutuata
35512	GTTRTA	2018-03-19	1
29858	MNZMBT	2012-11-30	0
18339	CPRLND	2012-10-28	1
35512	GTTRTA	2018-03-02	1
16220	MNZMBT	2013-01-25	0
35512	LPRNTA	2013-03-01	0
18339	CPRLND	2018-03-01	0



**VISITA V2**

Medico	Paziente	Data	Mutuata
35512	GTTRTA	2018-03-19	1
29858	MNZMBT	2012-11-30	0
18339	CPRLND	2012-10-28	1
35512	GTTRTA	2018-03-02	1
16220	MNZMBT	2013-01-25	0
35512	LPRNTA	2013-03-01	0
18339	CPRLND	2018-03-01	0

**VISITA V1**  $\bowtie$  **VISITA V2**  $V1.\text{Medico} = V2.\text{Medico} \wedge V1.\text{Paziente} = V2.\text{Paziente} \wedge V1.\text{Data} < > V2.\text{Data}$

Medico	Paziente	Data	Mutuata	Medico	Paziente	Data	Mutuata
35512	GTTRTA	2018-03-19	1	35512	GTTRTA	2018-03-02	1
18339	CPRLND	2012-10-28	1	18339	CPRLND	2018-03-01	0
35512	GTTRTA	2018-03-02	1	35512	GTTRTA	2018-03-19	1
18339	CPRLND	2018-03-01	0	18339	CPRLND	2012-10-28	1

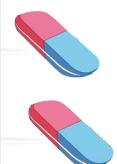
# Self join: cosa accade (II)

Indicare il codice fiscale dei pazienti che sono stati visitati  
**più di una volta da almeno un medico** della clinica, nel mese corrente

**VISITA V1**  $\bowtie$   $V1.\text{Medico} = V2.\text{Medico} \wedge V1.\text{Paziente} = V2.\text{Paziente} \wedge V1.\text{Data} <> V2.\text{Data}$  **VISITA V2**

Medico	Paziente	Data	Mutuata	Medico	Paziente	Data	Mutuata
35512	GTTRTA	2018-03-19	1	35512	GTTRTA	2018-03-02	1
18339	CPRLND	2012-10-28	1	18339	CPRLND	2018-03-01	0
35512	GTTRTA	2018-03-02	1	35512	GTTRTA	2018-03-19	1
18339	CPRLND	2018-03-01	0	18339	CPRLND	2012-10-28	1

Mese corrente  
Marzo 2018



Applicando la condizione nel WHERE, rimangono le visite effettuate  
nel mese corrente (record di V1) per le quali esiste un'altra visita  
effettuata nel mese corrente dallo stesso paziente, con lo stesso medico

Medico	Paziente	Data	Mutuata	Medico	Paziente	Data	Mutuata
35512	GTTRTA	2018-03-19	1	35512	GTTRTA	2018-03-02	1
35512	GTTRTA	2018-03-02	1	35512	GTTRTA	2018-03-19	1



Paziente
GTTRTA

# Esempio con self outer join

---

Indicare la matricola dei medici che, nel mese di Ottobre 2013, hanno visitato **per la prima volta** almeno un paziente

```
SELECT DISTINCT V1.Medico  
FROM Visita V1 LEFT OUTER JOIN Visita V2  
ON (  
    V2.Medico = V1.Medico  
    AND V2.Paziente = V1.Paziente  
    AND V2.Data < V1.Data  
)  
WHERE YEAR(V1.Data) = 2013 AND MONTH(V1.Data) = 10  
    AND V2.Paziente IS NULL;
```

Se in Ottobre 2013 un medico ha visitato per la prima volta un paziente in una visita v1, allora non esiste nella tabella Vista una visita v2 antecedente a v1 in cui il medico e il paziente coinvolti siano gli stessi coinvolti in v1. Consideriamo quindi le visite v1 che non fanno join con visite v2 antecedenti a v1 che coinvolgono lo stesso paziente e lo stesso medico di v1.

# Self outer join: cosa accade (I)

Indicare la matricola dei medici che, nel mese di Ottobre 2013, hanno visitato **per la prima volta** almeno un paziente

**VISITA V1**

Medico	Paziente	Data	Mutuata
35512	GTTRTA	2017-03-19	1
29858	MNZMBT	2012-11-30	0
18339	CPRLND	2013-10-28	1
35512	GTTRTA	2013-10-02	1
16220	MNZMBT	2013-01-25	0
35512	LPRNTA	2013-10-01	0
18339	CPRLND	2017-03-01	0



**VISITA V2**

Medico	Paziente	Data	Mutuata
35512	GTTRTA	2017-03-19	1
29858	MNZMBT	2012-11-30	0
18339	CPRLND	2013-10-28	1
35512	GTTRTA	2013-10-02	1
16220	MNZMBT	2013-01-25	0
35512	LPRNTA	2013-10-01	0
18339	CPRLND	2017-03-01	0

**VISITA V1** V1.Medico = V2.Medico  $\wedge$  V1.Paziente = V2.Paziente  $\wedge$  V2.Data < V1.Data      **VISITA V2**

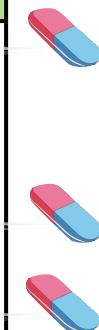
Medico	Paziente	Data	Mutuata	Medico	Paziente	Data	Mutuata
35512	GTTRTA	2017-03-19	1	35512	GTTRTA	2017-03-02	1
29858	MNZMBT	2012-11-30	0	NULL	NULL	NULL	NULL
18339	CPRLND	2013-10-28	1	NULL	NULL	NULL	NULL
35512	GTTRTA	2013-10-02	1	NULL	NULL	NULL	NULL
16220	MNZMBT	2013-01-25	0	NULL	NULL	NULL	NULL
35512	LPRNTA	2013-10-01	0	NULL	NULL	NULL	NULL
18339	CPRLND	2017-03-01	0	18339	CPRLND	2012-10-28	1

# Self outer join: cosa accade (II)

Indicare la matricola dei medici che, nel mese di Ottobre 2013, hanno visitato **per la prima volta** almeno un paziente

**VISITA V1**  **V1.Medico = V2.Medico  $\wedge$  V1.Paziente = V2.Paziente  $\wedge$  V2.Data < V1.Data** **VISITA V2**

Medico	Paziente	Data	Mutuata	Medico	Paziente	Data	Mutuata
35512	GTTRTA	2017-03-19	1	35512	GTTRTA	2017-03-02	1
29858	MNZMBT	2012-11-30	0	NULL	NULL	NULL	NULL
18339	CPRLND	2013-10-28	1	NULL	NULL	NULL	NULL
35512	GTTRTA	2013-10-02	1	NULL	NULL	NULL	NULL
16220	MNZMBT	2013-01-25	0	NULL	NULL	NULL	NULL
35512	LPRNTA	2013-10-01	0	NULL	NULL	NULL	NULL
18339	CPRLND	2017-03-01	0	18339	CPRLND	2012-10-28	1



Applicando la condizione nel WHERE, rimangono le visite effettuate nel mese di Ottobre 2013 (record di V1) per le quali non esiste un'altra visita effettuata in passato dallo stesso paziente, con lo stesso medico.

Medico	Paziente	Data	Mutuata	Medico	Paziente	Data	Mutuata
29858	MNZMBT	2012-11-30	0	NULL	NULL	NULL	NULL
18339	CPRLND	2013-10-28	1	NULL	NULL	NULL	NULL
35512	GTTRTA	2013-10-02	1	NULL	NULL	NULL	NULL
35512	LPRNTA	2013-10-01	0	NULL	NULL	NULL	NULL



Medico
29858
18339
35512

# Esempio con self join e USING

---

Indicare il codice fiscale dei pazienti che sono stati visitati  
**più di una volta** nel mese corrente

```
SELECT DISTINCT V1.Paziente
FROM Visita V1 INNER JOIN Visita V2 USING(Paziente)
WHERE MONTH(V1.Data) = MONTH(CURRENT_DATE)
      AND MONTH(V2.Data) = MONTH(CURRENT_DATE)
      AND (
            V2.Data = V1.Data AND V2.Medico <> V1.Medico)
      OR
            V2.Data <> V1.Data
);
```

using(Paziente) equivale a mettere  
on V1.Paziente=V2.Paziente

# Derived table

---



# Derived table

---

sono immediatamente cancellate alla fine dell'esecuzione



Sono **tabelle volatili** che possono essere incapsulate nel FROM

utili per costruire risultati intermedi

# Derived table: esempio

---

Indicare la matricola dei medici che non hanno mai visitato di giovedì

```
SELECT DISTINCT V1.Medico  
FROM Visita V1 LEFT OUTER JOIN  
(  
    SELECT V2.Medico      Derived table  
    FROM Visita V2  
    WHERE DAYOFWEEK(V2.Data) = 4  
) AS D ←  
    USING(Medico)  
WHERE D.Medico IS NULL;
```



alias obbligatorio

# Subquery

---



# Subquery: cosa sono?

---

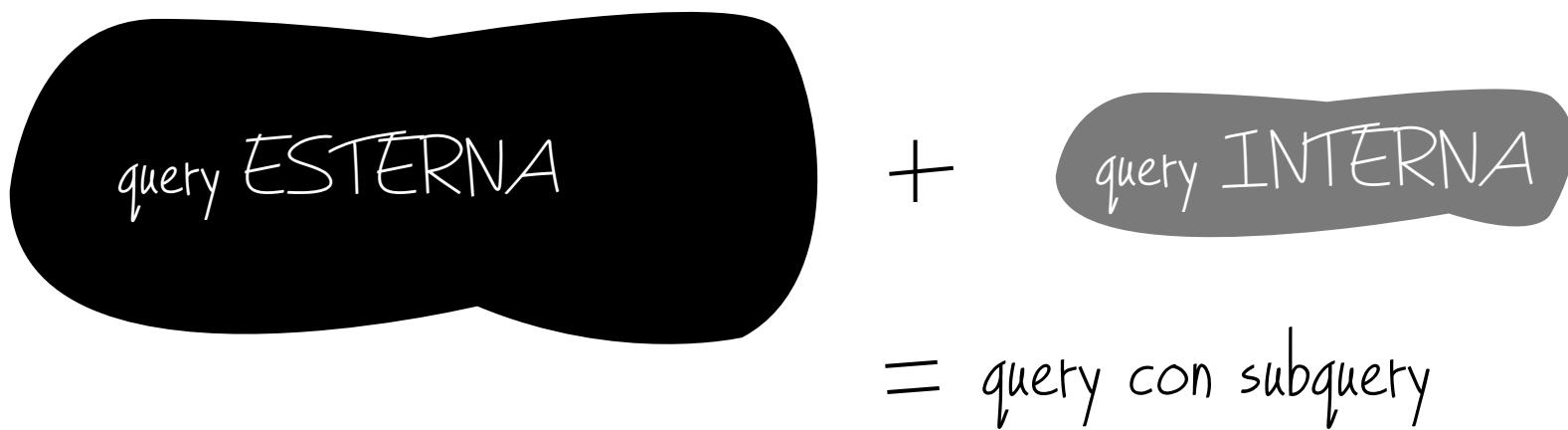
Sono query che possono essere **incapsulate** in un'altra query

*dette subquery*

in due modalità diverse:  
non correlata o correlata

# Subquery: struttura

---



# Subquery: perché?

---

Rappresentano **un'alternativa al join** per query su più tabelle



per alcuni programmatori sono un modo più  
semplice per risolvere query complesse

# Noncorrelated subquery

---



# Noncorrelated subquery: cosa sono

---

eliminati alla fine dell'esecuzione

Si incapsulano nel WHERE per costruire risultati che  
**servono come elementi** per determinare il risultato della query esterna.



i record della subquery non dipendono dalla query esterna

# Noncorrelated subquery: risultato

---

Un record fa parte del risultato se un sottoinsieme  
dei suoi attributi si trova **anche** nel result set della subquery

→ quello che vede l'utente

# Noncorrelated subquery in MySQL

Indicare nome, cognome e parcella degli ortopedici che hanno effettuato almeno una visita nell'anno 2013

`SELECT M.Nome, M.Cognome, M.Parcella  
FROM Medico M  
WHERE M.Specializzazione = 'Ortopedia'  
AND M.Matricola IN (  
 SELECT V.Medico  
 FROM Visita V  
 WHERE YEAR(V.Data) = 2013  
);`

qui si può spezzare

query esterna

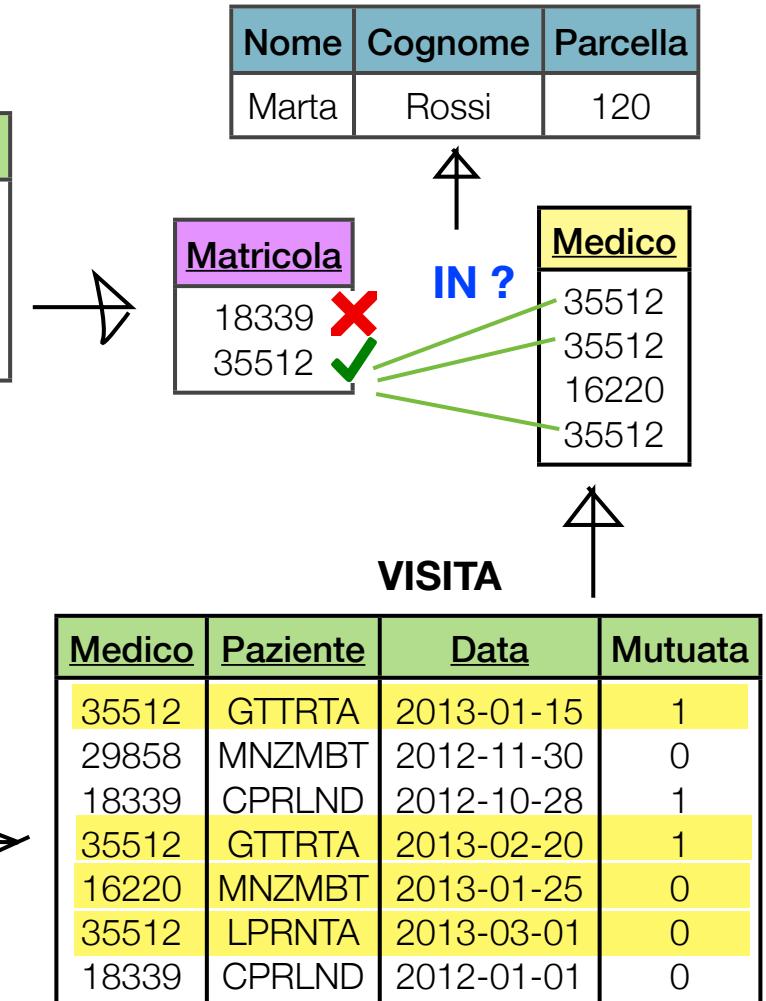
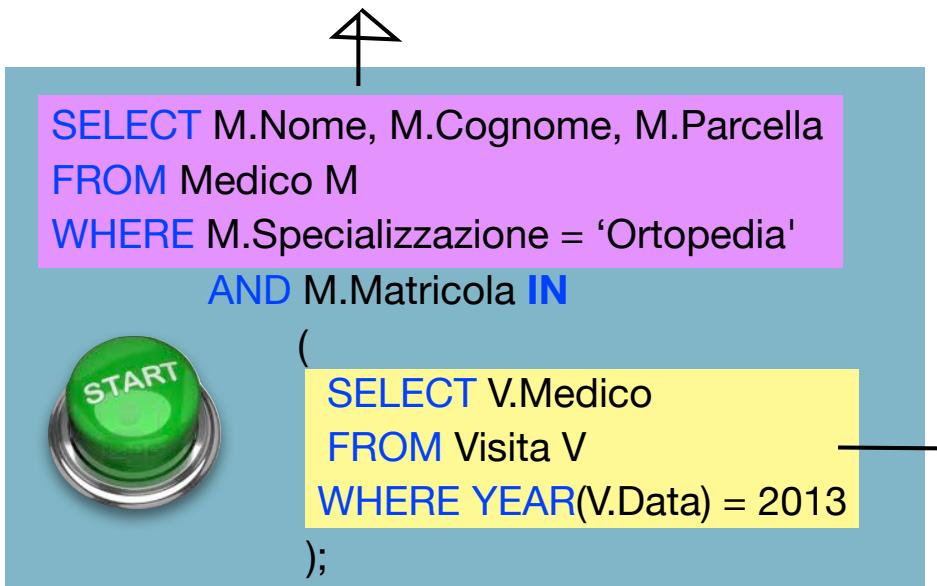
subquery

'IN' permette di controllare la presenza della matricola all'interno del risultato della subquery

# Damn! How does it work?

Indicare nome, cognome e parcella degli ortopedici che hanno effettuato almeno una visita nell'anno 2013

MEDICO					
Matricola	Cognome	Nome	Specializzazione	Parcella	Città
18339	Verdi	Paolo	Ortopedia	150	Pisa
35512	Rossi	Marta	Ortopedia	120	Pisa
16220	Gialli	Rita	Cardiologia	135	Roma
29858	Neri	Rino	Dermatologia	110	Siena



# Noncorrelated subquery: negazione

---

Usando **NOT IN** un record della outer query va nel risultato solo **se non è presente** nel result set della subquery



IN controlla che il record sia anche nel risultato della subquery, mentre NOT IN che non ci sia

# Negazione: esempio

---

Indicare i cognomi dei pazienti che **non appartengono** anche a un medico

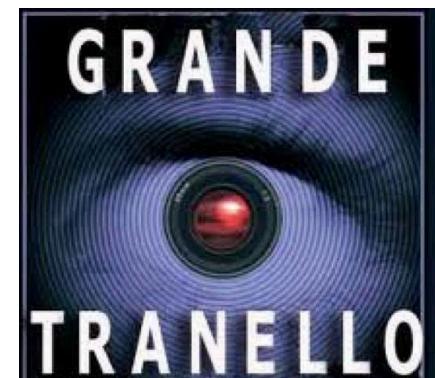
```
SELECT DISTINCT P.Cognome  
FROM Paziente P  
WHERE P.Cognome NOT IN (  
    SELECT M.Cognome  
    FROM Medico M  
);
```

il cognome di un paziente va nel risultato se non c'è  
un medico a cui tale cognome appartiene

# La query insidiosa

---

Indicare nome, cognome e specializzazione dei medici che hanno effettuato visite eccetto che il giorno 1° Marzo 2013



# Soluzione

---

Indicare nome, cognome e specializzazione dei medici **che hanno effettuato visite** eccetto che il giorno 1° Marzo 2013

```
SELECT M.Nome, M.Cognome, Specializzazione  
FROM Medico M  
WHERE Matricola NOT IN (SELECT V.Medico  
                          FROM Visita V  
                          WHERE V.Data = '2013-03-01'  
);
```

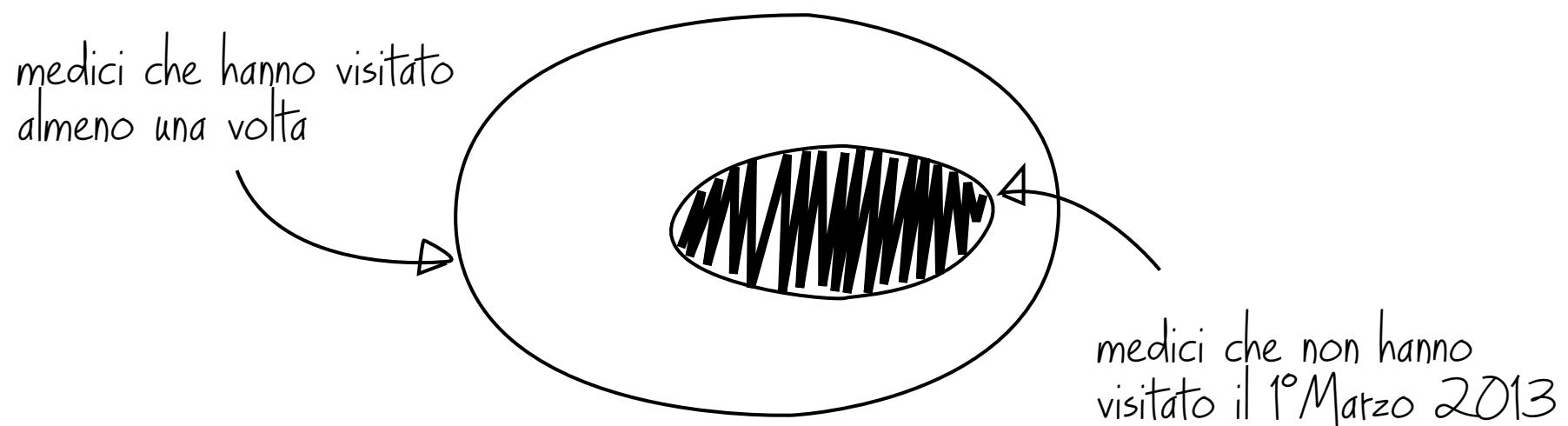


se un medico non ha effettuato alcuna visita va nel risultato!

# Perché è sbagliata?

---

Indicare nome, cognome e specializzazione dei medici che hanno effettuato visite eccetto che il giorno 1° Marzo 2013



# Soluzione corretta

---

Indicare nome, cognome e specializzazione dei medici che hanno effettuato visite eccetto che il giorno 1° Marzo 2013

```
SELECT M.Nome, M.Cognome, M.Specializzazione  
FROM Medico M  
WHERE M.Matricola IN (  
    SELECT V.Medico  
    FROM Visita V  
)  
AND M.Matricola NOT IN (  
    SELECT V.Medico  
    FROM Visita V  
    WHERE V.Data = '2013-03-01'  
);
```



se un medico non ha effettuato alcuna visita NON va nel risultato!

# Equivalenza join-subquery

---

Data una query con subquery è **sempre** possibile passare alla versione basata su join, e viceversa



il query optimizer sfrutta questa equivalenza per riscrivere le query il cui codice avrebbe bassa performance

# Versione join-equivalente (slide 76)

---

Indicare nome, cognome e specializzazione dei medici che hanno effettuato visite eccetto che il giorno 1° Marzo 2013

```
SELECT DISTINCT M.Nome, M.Cognome, M.Specializzazione  
FROM Visita V INNER JOIN Medico M ON V.Medico = M.Matricola  
LEFT OUTER JOIN(  
    SELECT *  
    FROM Visita  
    WHERE Data = '2013-03-01'  
) AS D  
ON V.Medico = D.Medico  
WHERE D.Medico IS NULL;
```



data una query con subquery è sempre possibile passare alla versione basata su join, e viceversa

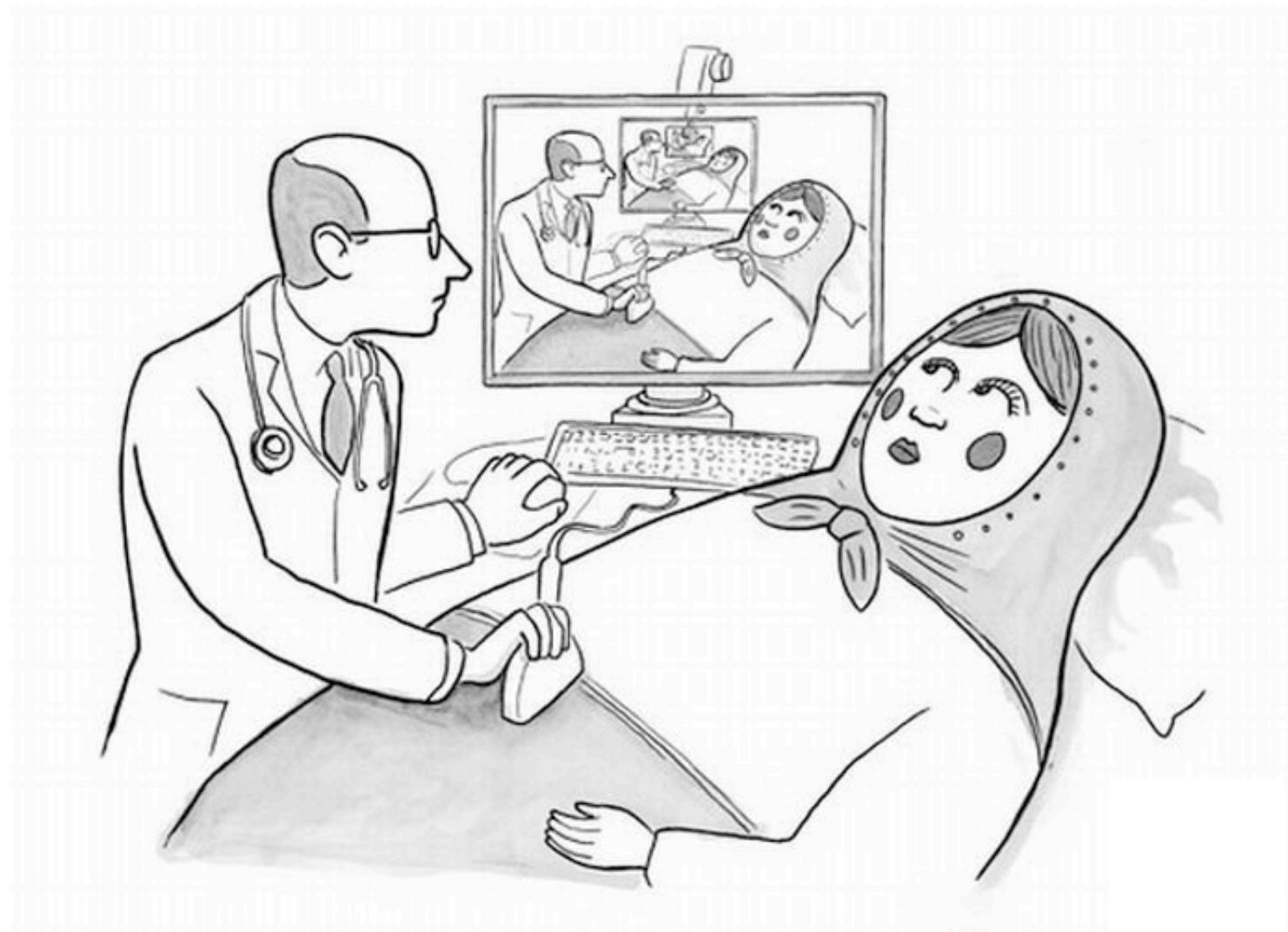
...la domanda sorge spontanea

---



# Quante subquery?

---



# Annidamento multiplo

---

**Non c'è limite** al numero di query che si possono annidare l'una nell'altra

*teoricamente*

# Annidamento multiplo: esempio

---

Indicare il numero di pazienti di Siena, mai visitati da ortopedici  
da Paziente ↗ da Visita ↗ ↘ da Medico

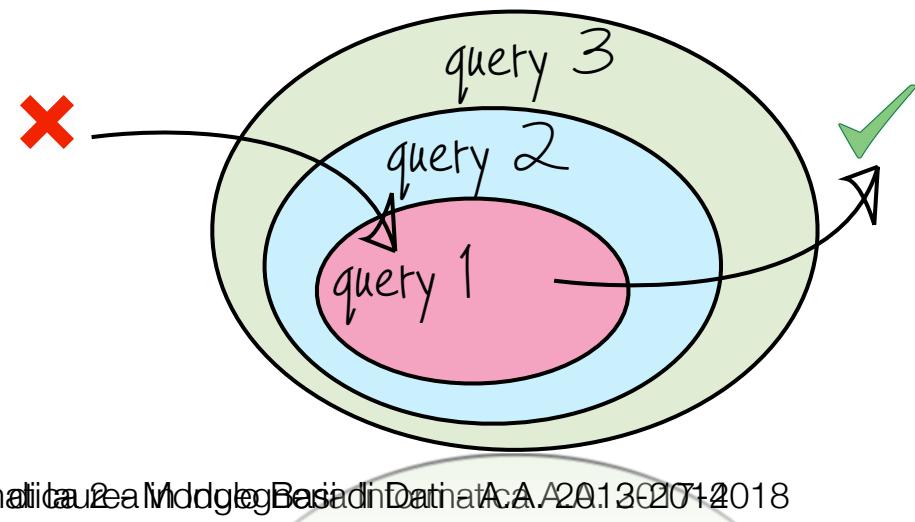
```
SELECT COUNT(*)  
FROM Paziente P  
WHERE P.Citta = 'Siena'  
AND P.CodFiscale NOT IN (  
    SELECT V.Paziente  
    FROM Visita V  
    WHERE V.Medico IN (  
        SELECT M.Matricola  
        FROM Medico M  
        WHERE M.Specializzazione = 'Ortopedia' )  
);
```

# Visibilità

---

In una subquery si possono riferire **tutti gli attributi delle query esterne**, a qualsiasi livello di annidamento esterno essi si trovino

si esce ma non si entra!



# Subquery scalari

---

une seule

solo uno

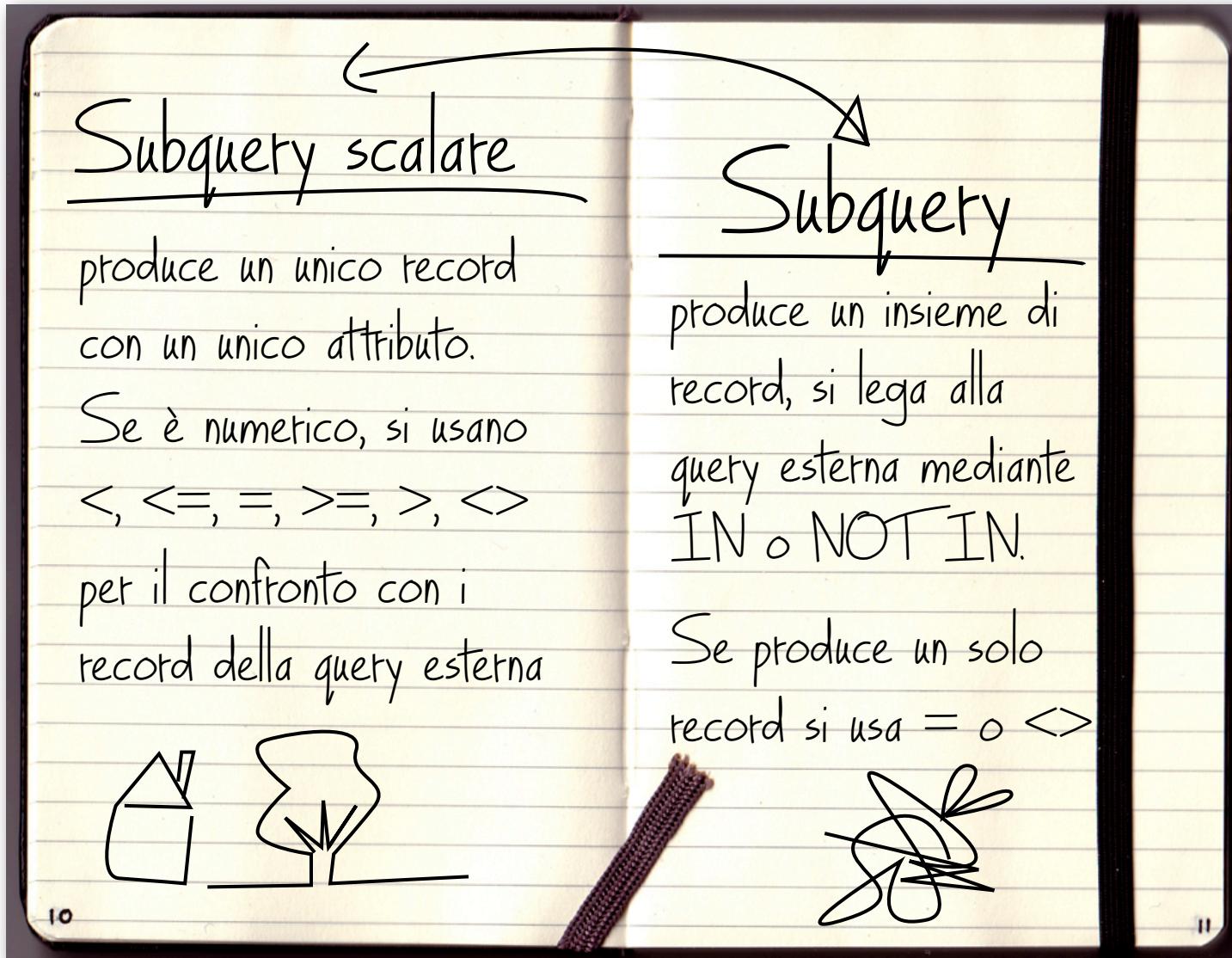
nut ein

Restituiscono **un unico record**, composto da un **unico attributo**

just one

uno e basta!

# Sottoquery vs sottoquery scalari



# Subquery scalare e operatori di confronto

---

Indicare il numero degli otorini **aventi parcella più alta della media** delle parcelle dei medici della loro specializzazione.

# Soluzione

---

Indicare il numero degli otorini **aventi parcella più alta della media** delle parcelle dei medici della loro specializzazione.

```
SELECT COUNT(*)
FROM Medico M1
WHERE M1.Specializzazione = 'Otorinolaringoiatria'
AND M1.Parcella > (
    SELECT AVG(M2.Parcella)
    FROM Medico M2
    WHERE M2.Specializzazione = 'Otorinolaringoiatria'
);
```

la subquery restituisce un numero!!!

# A volte ritornano...

---

Indicare qual è il reddito massimo, **e il nome e cognome di chi lo detiene**

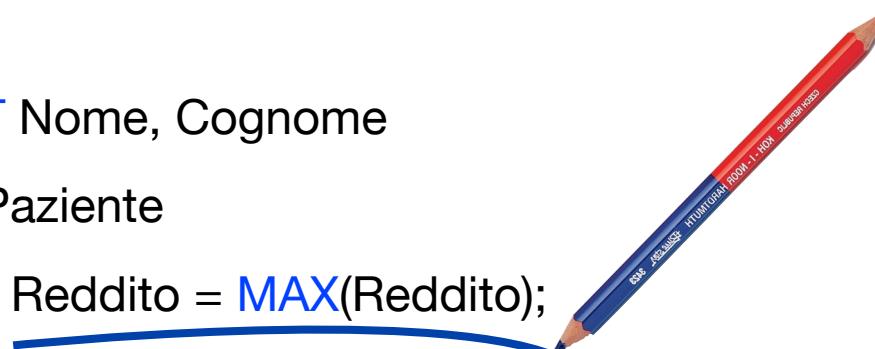


# Gli errori blu, la follia

---



```
SELECT MAX(Reddotto), Nome, Cognome  
FROM Paziente;
```



```
SELECT Nome, Cognome  
FROM Paziente  
WHERE Reddotto = MAX(Reddotto);
```



**COSE**<sub>da</sub>  
**PAZZI**

# Soluzione

---

Indicare qual è il reddito massimo, **e il nome e cognome di chi lo detiene**

```
SELECT Reddito,  
      Nome,  
      Cognome  
  FROM Paziente  
 WHERE Reddito = (  
                   SELECT MAX(Reddito)  
                   FROM Paziente  
                 );
```



è l'unico modo corretto di risolvere il problema del record correlato al valore massimo su un attributo

# Ma il male...

---

Indicare nome e cognome dei pazienti che non sono mai stati visitati dal medico  
avente la parcella più alta fra tutti i medici della clinica

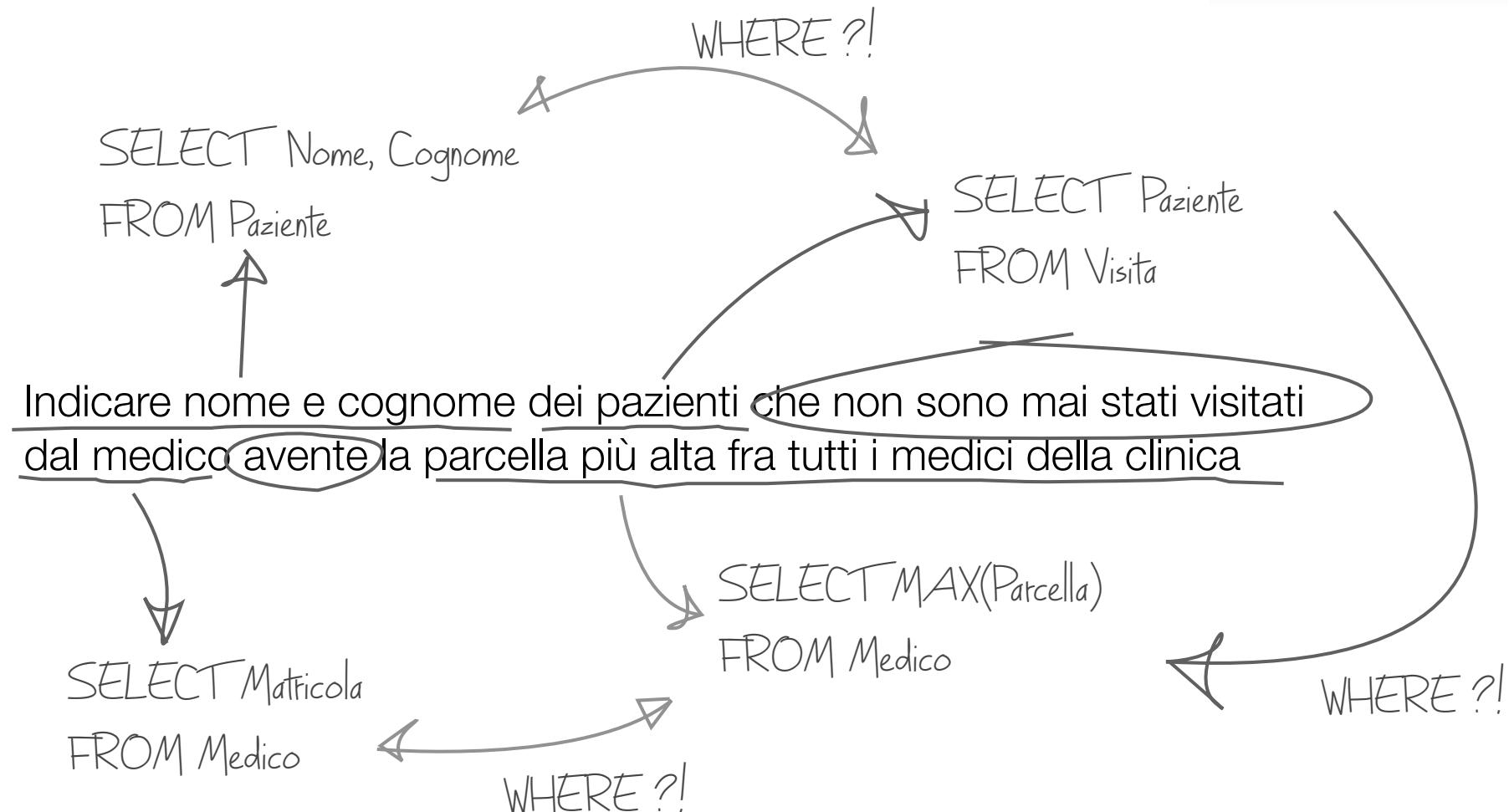


# Sconfiggiamola

---



# Soluzione



# Soluzione

---

Indicare nome e cognome dei pazienti che non sono mai stati visitati dal medico  
avente la parcella più alta fra tutti i medici della clinica

```
SELECT Nome, Cognome
FROM Paziente
WHERE CodFiscale NOT IN (
    SELECT Paziente
    FROM Visita
    WHERE Medico = (
        SELECT Matricola
        FROM Medico
        WHERE Parcella = ( SELECT MAX(Parcella)
                           FROM Medico)
    )
);
```

...ho nascosto un errore, non lo troverete mai!!!!



# Soluzione

---

Indicare nome e cognome dei pazienti che non sono mai stati visitati dal medico  
avente la parcella più alta fra tutti i medici della clinica

```
SELECT Nome, Cognome  
FROM Paziente  
WHERE CodFiscale NOT IN (  
    SELECT Paziente  
    FROM Visita  
    WHERE Medico = (  
        SELECT Matricola  
        FROM Medico  
        WHERE Parcella = ( SELECT MAX(Parcella)  
                           FROM Medico )  
    );  
);
```

questa subquery in generale può restituire più di un record!



più di un medico può avere la parcella massima, si deve usare IN!!!

# Soluzione

---

Indicare nome e cognome dei pazienti che non sono mai stati visitati dal medico  
avente la parcella più alta fra tutti i medici della clinica

```
SELECT Nome, Cognome
FROM Paziente
WHERE CodFiscale NOT IN (
    SELECT Paziente
    FROM Visita
    WHERE Medico IN (
        SELECT Matricola
        FROM Medico
        WHERE Parcella = ( SELECT MAX(Parcella)
                           FROM Medico)
    )
);
```

'IN' ammette che più di un medico abbia la parcella massima

# Risoluzione mista subquery-join

---



Una query può essere risolta anche **combinando** le subquery e join



# Esempio

---

Indicare il numero di pazienti di età superiore a 50 anni visitati dai cardiologi di Pisa aventi parcella inferiore alla media delle parcelle dei cardiologi.

# Bisturi!



SELECT COUNT( ?? )

Indicare il numero di pazienti di età superiore a 50 anni visitati

SELECT CodFiscale  
FROM Paziente P INNER JOIN Visita V  
ON P.CodFiscale=V.Paziente  
WHERE « Età superiore a 50 anni »  
AND V.Medico IN

dai cardiologi di Pisa con parcella inferiore alla media delle parcelle dei cardiologi.

SELECT Matricola  
FROM Medico  
WHERE Specializzazione="Cardiologia"  
AND Citta="Pisa"

SELECT AVG(Parcella)  
FROM Medico  
WHERE Specializzazione="Cardiologia"

AND Parcella <

# ...e adesso ricuciamo

---

Indicare il numero di pazienti di età superiore a 50 anni visitati dai cardiologi di Pisa aventi parcella inferiore alla media delle parcelle dei cardiologi.

```
SELECT COUNT(DISTINCT CodFiscale)
FROM Paziente P INNER JOIN Visita V
    ON P.CodFiscale = V.Paziente
WHERE CURRENT_DATE > P.DataNascita + INTERVAL 50 YEAR
    AND V.Medico IN (
        SELECT M.Matricola
        FROM Medico M
        WHERE M.Specializzazione = 'Cardiologia'
            AND M.Citta = 'Pisa'
            AND M.Parcella < ( SELECT AVG(M2.Parcella)
                FROM Medico M2
                WHERE M2.Specializzazione = 'Cardiologia' )
    );
```



# Avvisi importanti

---

Risolvete gli esercizi che trovate al termine di questo pacchetto di slide **dopo aver studiato gli argomenti della lezione.**

Vi consiglio di provare a risolvere anche gli **esercizi presentati come esempi** nelle slide. Se riuscite a risolverli senza guardare la soluzione non dovreste avere problemi con gli esercizi per casa.

Vi ricordo che questo pacchetto di slide è quello completo che ho progettato durante la lezione. Segnalatemi eventuali errori, può darsi che qualcosa mi sia scappato.  
Su richiesta di uno studente, ho aggiunto alcune slide sul funzionamento del self join e del self outer join, dove non c'erano animazioni. Spero che siano utili.

La lezione di tutoring nella quale correggerò gli esercizi assegnati e faremo il solito dibattito si terrà **giovedì 5 Aprile, alle ore 10:30, in aula F9.**

# Esercizi

---

- Indicare l'incasso totale degli ultimi due anni, realizzato grazie alle visite dei medici cardiologi della clinica.
- Indicare il numero di pazienti di sesso femminile che, nel quindicesimo anno d'età, sono stati visitati, una o più volte, sempre dallo stesso ginecologo.
- Indicare codice fiscale, nome e cognome ed età del paziente più anziano della clinica, e il numero di medici dai quali è stato visitato.
- Indicare nome e cognome dei pazienti che sono stati visitati non meno di due volte dalla dottoressa Gialli Rita.
- Indicare il reddito medio dei pazienti che sono stati visitati solo da medici con parcella superiore a 100 euro, negli ultimi sei mesi. [Risolvere solo con noncorrelated subquery e scrivere la versione join-equivalente].