Linguaggi di Basi di Dati

Analisi dei compiti di simulazione d’esame

Raccomandazioni generali

- Scrivere query AR e SQL al computer
- Per AR, invece delle lettere greche, è più rapido usare le diciture SEL (selezione) PROJ (proiezione) JOIN e REN (ridenominazione)
- Scrivere operatori AR e SQL e clausole SQL in maiuscolo
- Controllare l'uso delle parentesi in AR
- Scrivere il documento in Word, se si usa Power Point per schema ER impostare pagina come A4 verticale per poterla copiare decentemente in Word
- Usare undescore invece degli spazi nei nomi di relazioni e attributi
- Preferire query annidate nella clausola WHERE a meno che non sia effettivamente utile una query annidata nella FROM, ovvero nei pochi casi in cui si vogliono eseguire operatori aggregati a diversi livelli di aggregazione

Algebra Relazionale

Errori da evitare

Errori gravi molto comuni

- Trascurare il fatto che il **Join** in algebra relazionale combina sempre le tuple che hanno valori uguali per gli attributi comuni alle due relazioni
- Utilizzare la clausola **punto** (.) per specificare la relazione di appartenenza
  - il costrutto punto esiste in SQL ma non esiste e non ha senso di esistere AR

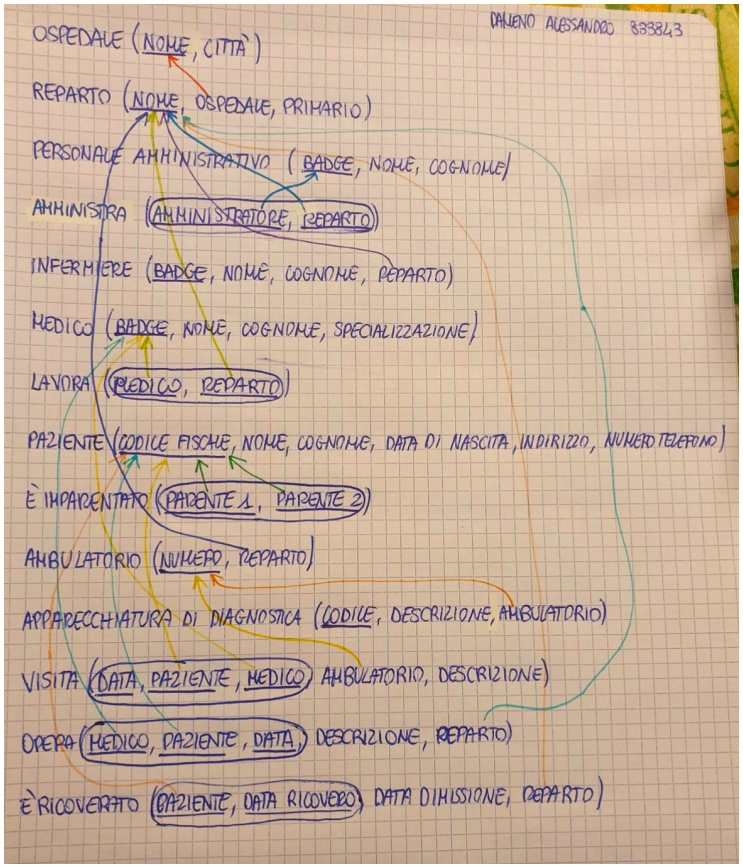
Errori gravi meno comuni

- Predicato di join assente
- Errato uso delle parentesi
- Discordanza tra testo e soluzione query
- Testo query poco chiaro
- Unione di relazioni definite su attributi diversi

Imprecisioni comuni

- Effettuare un Join inutile ai fini del testo della query
- Ridenominazione poco utile

Esempio di Join errato



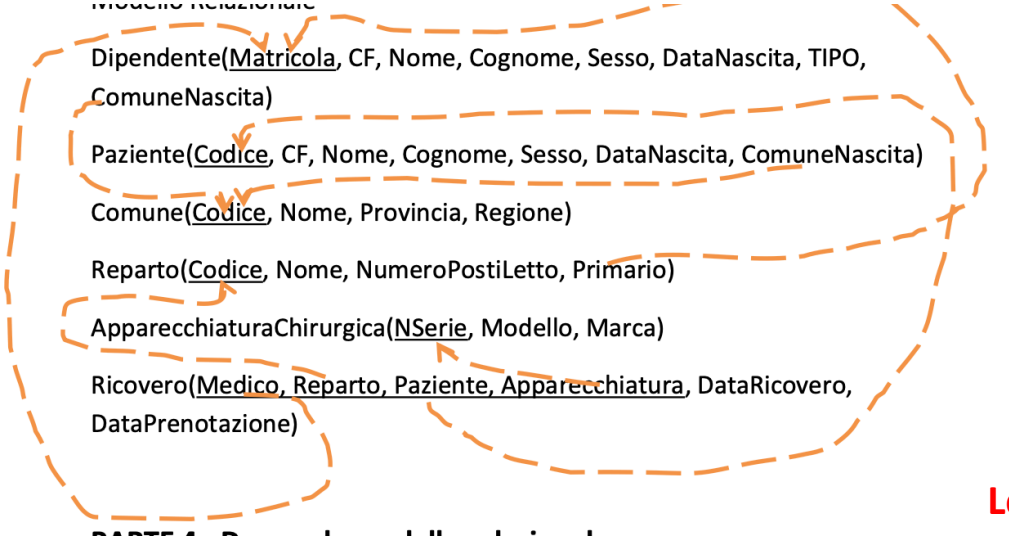
Tutto il personale ospedaliero che non è primario

$PROJ_{Badge,\ Cognome,\ Nome}$(Personale amministrativo)
UNION
$PROJ_{Badge,\ Cognome,\ Nome}$(Infermiere)
UNION
$PROJ_{Badge,\ Cognome,\ Nome}$(Medico)
DIFF
$PROJ_{Badge,\ Cognome,\ Nome}$(Medico $JOIN_{Badge\ =\ Primario}$(Reparto))

Le relazioni Medico e Reparto hanno entrambe un attributo NOME.
Quindi Medico $_{JOINBadge=Primario}$Reparto restituisce solo i medici primari che si chiamano come dei reparti

Questa è un'ottima occasione per utilizzare l'operatore di RIDENOMINAZIONE ad esempio:
Medico $_{JOINBadge=Primario}REN_{Nome\_reparto<-Nome}$(Reparto)

Altro Esempio di Join errato



SEL Nome = "Milano"

(**REN** Codice, NomePersona ←ComuneNascita, Nome

(**PROJ** CF, Nome, Cognome, ComuneNascita (Dipendente)

UNION

PROJ CF, Nome, Cognome, ComuneNascita (Paziente)))

JOIN

(**PROJ** Codice, Nome (Comune))

Le relazioni Paziente e Comune hanno entrambe un attributo NOME.
Saranno selezionati solo i Pazienti che abitano a Milano e che si chiamano Milano

Esempi di utilizzo del costrutto punto impropriamente in AR

5. Query algebra relazionale

1. Selezionare i pazienti a cui è stata diagnosticata una malattia con un tasso di mortalita maggiore del 40%

PROJ paziente.nome,paziente.cognome(SEL malattia.tasso_mortalita>40 (PAZIENTE join paziente.cf=diagnosi.cf DIAGNOSI join diagnosi.id = malattia.id MALATTIA)

2. Selezionare tutti i medici che non hanno effettuato interventi di neurologia

PROJ personale.nome,personale.cognome(SEL (PERSONALE join personale.matricola=operazione_chirurgica.matricola OPERAZIONE_CHRIRUGICA join reparti.id_reparto=operazione_chirurgica.id_reparto REPARTI)

-

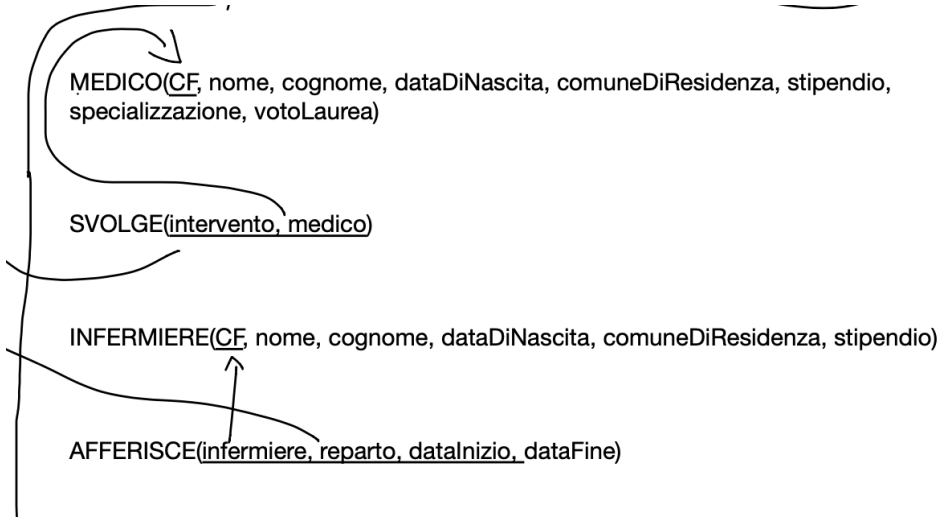
PROJ personale.nome,personale.cognome(SEL reparti.id_reparto="neurologia" (PERSONALE join personale.matricola=operazione_chirurgica.matricola OPERAZIONE_CHRIRUGICA join reparti.id_reparto=operazione_chirurgica.id_reparto REPARTI)

Non occorre indicare il predicato di join quando due attributi hanno lo stesso nome

Basta scrivere Paziente JOIN Diagnosi

Per questo motivo in AR non esiste la claiusola punto (.) per specificare l'appartenenza

Esempio testo poco chiaro

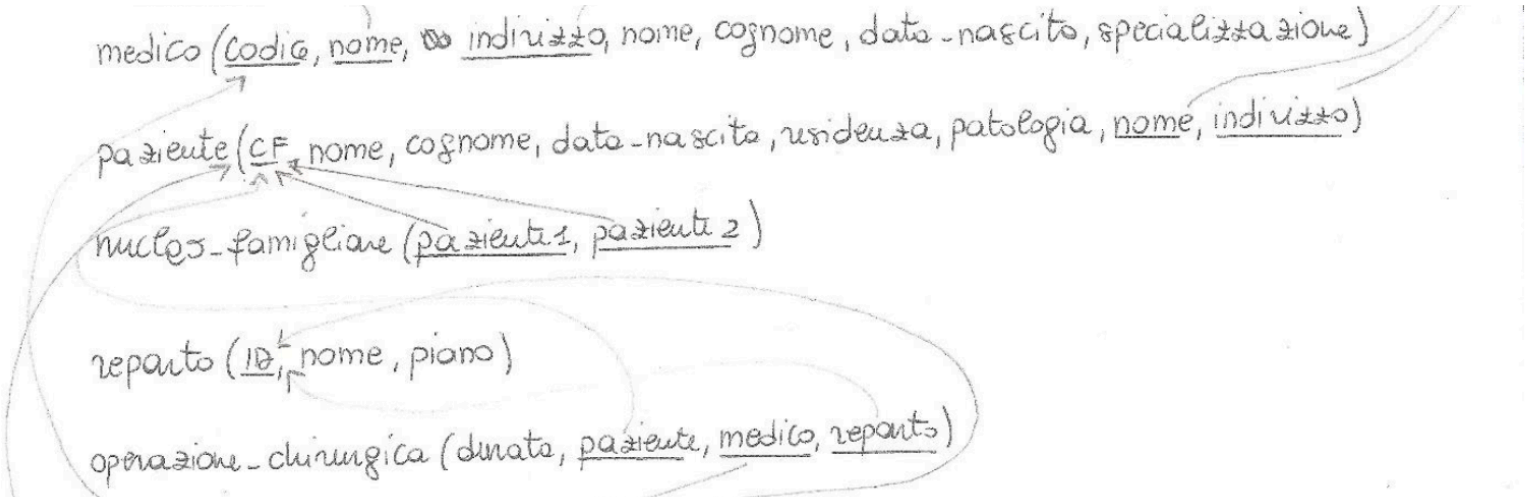


Cosa significa «gli attributi comuni»?

2) Gli attributi comuni di medici e di infermieri lavorano attualmente nel reparto "Cardiologia", e che abbiano uno stipendio maggiore o uguale di 50000

$$\sigma_{stipendio \geq 50K}\Big(\Big(\pi_{CF,nome,cognome,dataD.Nascita,comuneD.Residenza,stipendio}\big(\sigma_{reparto="cardiologia" \text{ AND } DataFine \text{ IS NULL}}(INFERMIERE \bowtie_{CF=Infermiere} AFFERISCE)\big)\Big) \cup \big(\pi_{CF,nome,cognome,dataD.Nascita,comuneD.Residenza,stipendio}(MEDICO)\big)\Big)$$

Esempio predicato di join assente



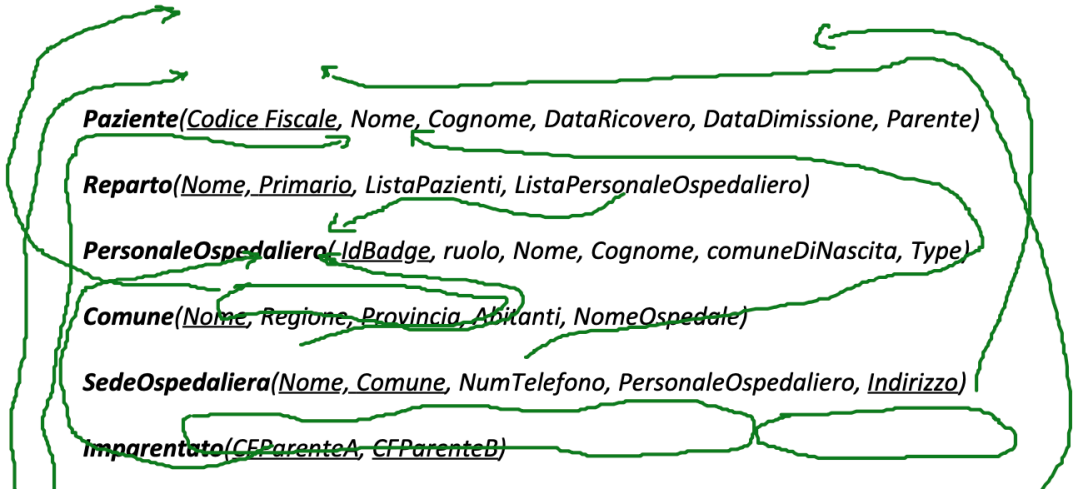
2. Espressione che produce tutti i medici specializzati in cardiochirurgia che hanno operato nel reparto con ID ABC123.

   $SEL_{reparto = ABC123 \text{ AND } specializzazione = \text{“cardiochirurgia”}}$ (operazione_chirurgica JOIN medico)

Il join naturale tra operazione_chirurgica e medico risulta in un prodotto cartesiano perché le due relazioni non hanno attributi comuni

Esempio discordanza tra testo e soluzione (e altri errori)

SCHEMA RELAZIONALE →



TESTO QUERY →

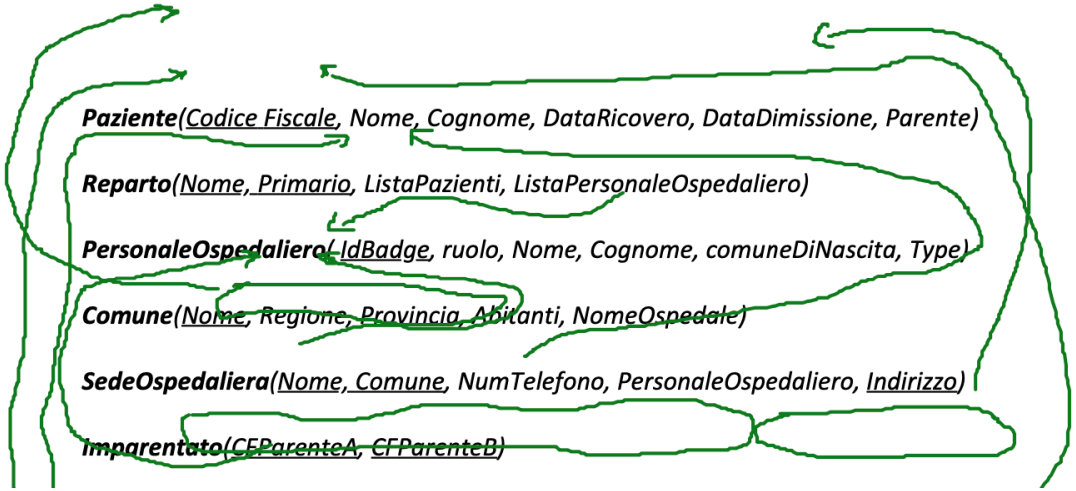
Fare Una query che rappresenti tutte le persone ricoverate a milano:

SOLUZIONE QUERY ↓

$$PROJ_{Nome,cognome}\ (((SEL_{Comune = Milano}(SedeOspedaliera\ \ JOIN_{Comune = Nome}\ Comune))\ JOIN_{Comune = ComuneDiNascita}\ (SEL_{ComuneDiNascita = Milano}\ Paziente)\ )$$

Esempio discordanza tra testo e soluzione (e altri errori)

SCHEMA RELAZIONALE



TESTO QUERY

Fare Una query che rappresenti tutte le persone ricoverate a milano:

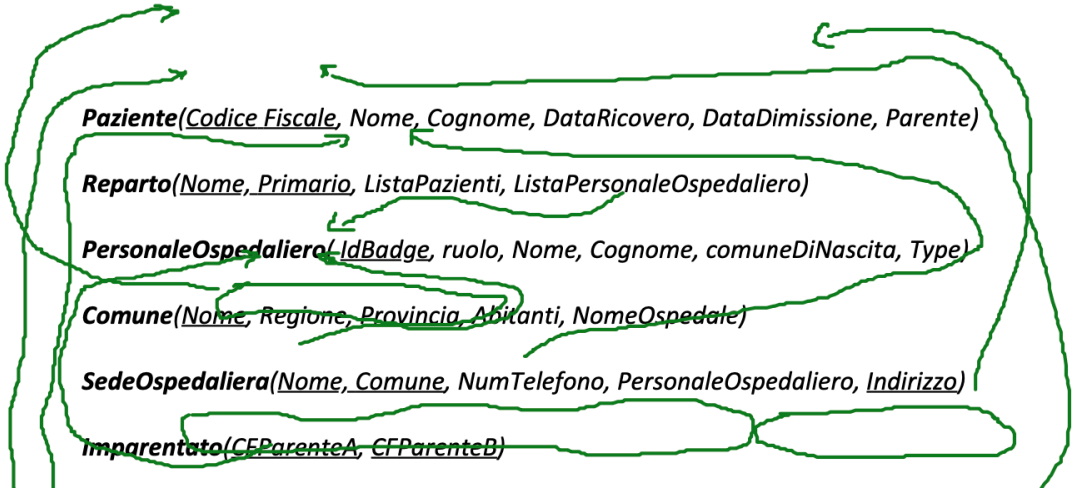
SOLUZIONE QUERY

La query seleziona solo le persone ricoverate a Milano che son anche nate a Milano!

$$PROJ_{Nome,cognome}\ (((SEL_{Comune = Milano}(SedeOspedaliera\ \ JOIN_{Comune = Nome}\ Comune))\ JOIN_{Comune = ComuneDiNascita}\ (SEL_{ComuneDiNascita = Milano}\ Paziente)\ )$$

Esempio discordanza tra testo e soluzione (e altri errori)

SCHEMA RELAZIONALE



TESTO QUERY

Fare Una query che rappresenti tutte le persone ricoverate a milano:

SOLUZIONE QUERY

Ci sono 5 parentesi aperte e 4 chiuse

$$PROJ_{Nome,cognome}\ (((SEL_{Comune = Milano}(SedeOspedaliera\ JOIN_{Comune = Nome}\ Comune))\ JOIN_{Comune = ComuneDiNascita}\ (SEL_{ComuneDiNascita = Milano}\ Paziente)\ )$$

Errato uso delle parentesi

⑤ Pazienti (CF, nome e cognome) che hanno subito una operazione e l'ospedale dove essa è stata eseguita.

$$\left(\Pi_{CF,\ Nome,\ cognome}\ (Paziente)\right.$$

$$\bowtie_{CF = CF\_Paziente}$$

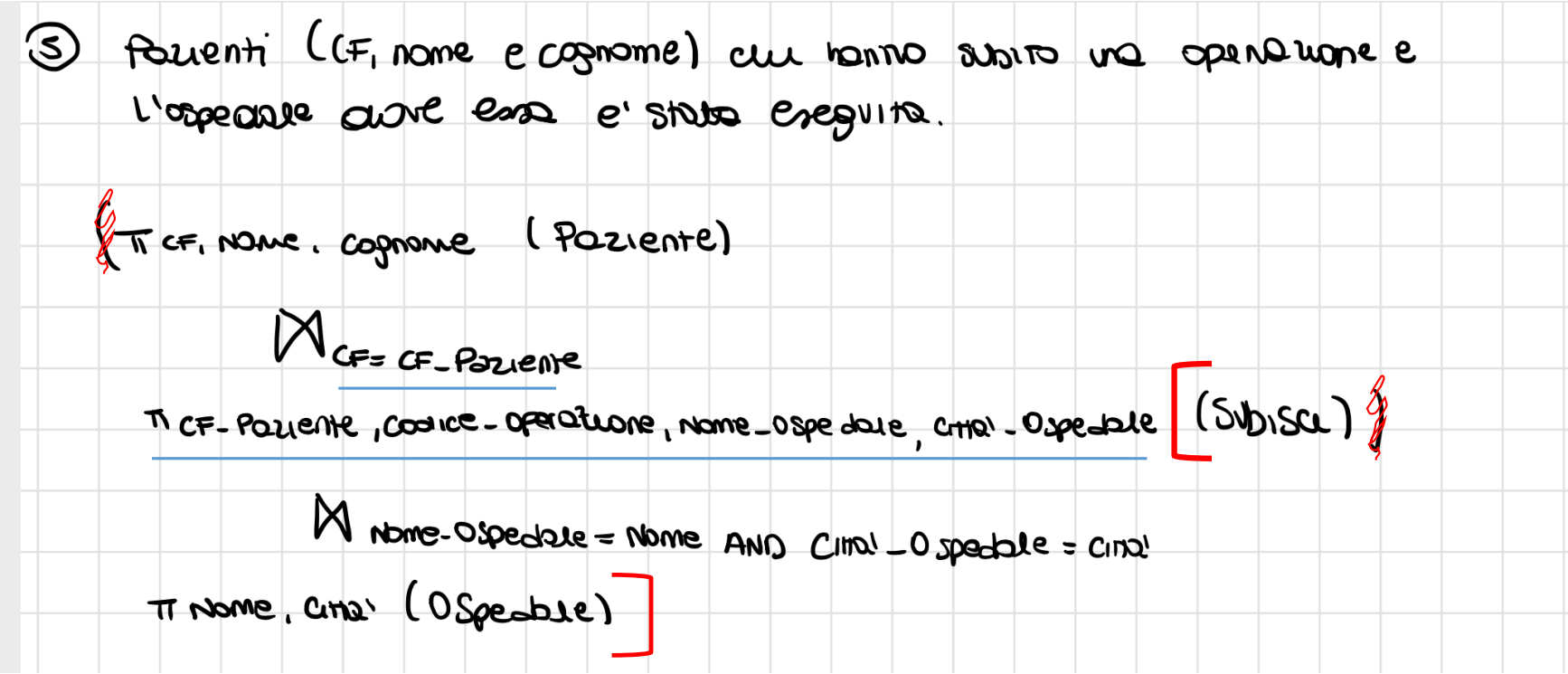
$$\Pi_{CF\_Paziente,\ Codice\_operazione,\ Nome\_Ospedale,\ Citta'\_Ospedale}\ (Subisce)\ \Big)$$

$$\bowtie_{Nome\_Ospedale = Nome\ AND\ Citta'\_Ospedale = Citta'}$$

$$\Pi_{Nome,\ Citta'}\ (Ospedale)$$

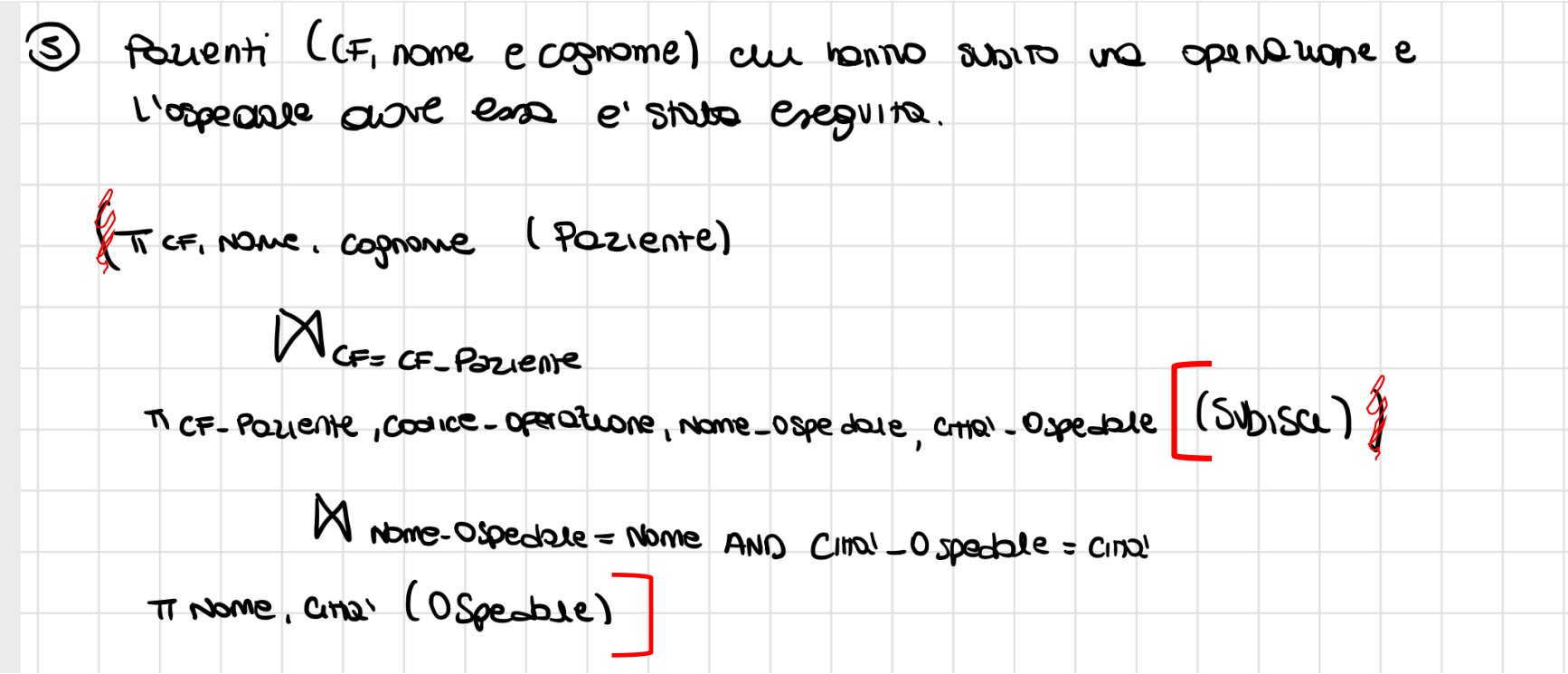
Le parentesi messe in questo modo portano ad un errore nel join tra le relazioni Pazienze, Subisce e Ospedale perché il nome dell'ospedale dovrà essere uguale a quello del paziente

Errato uso delle parentesi



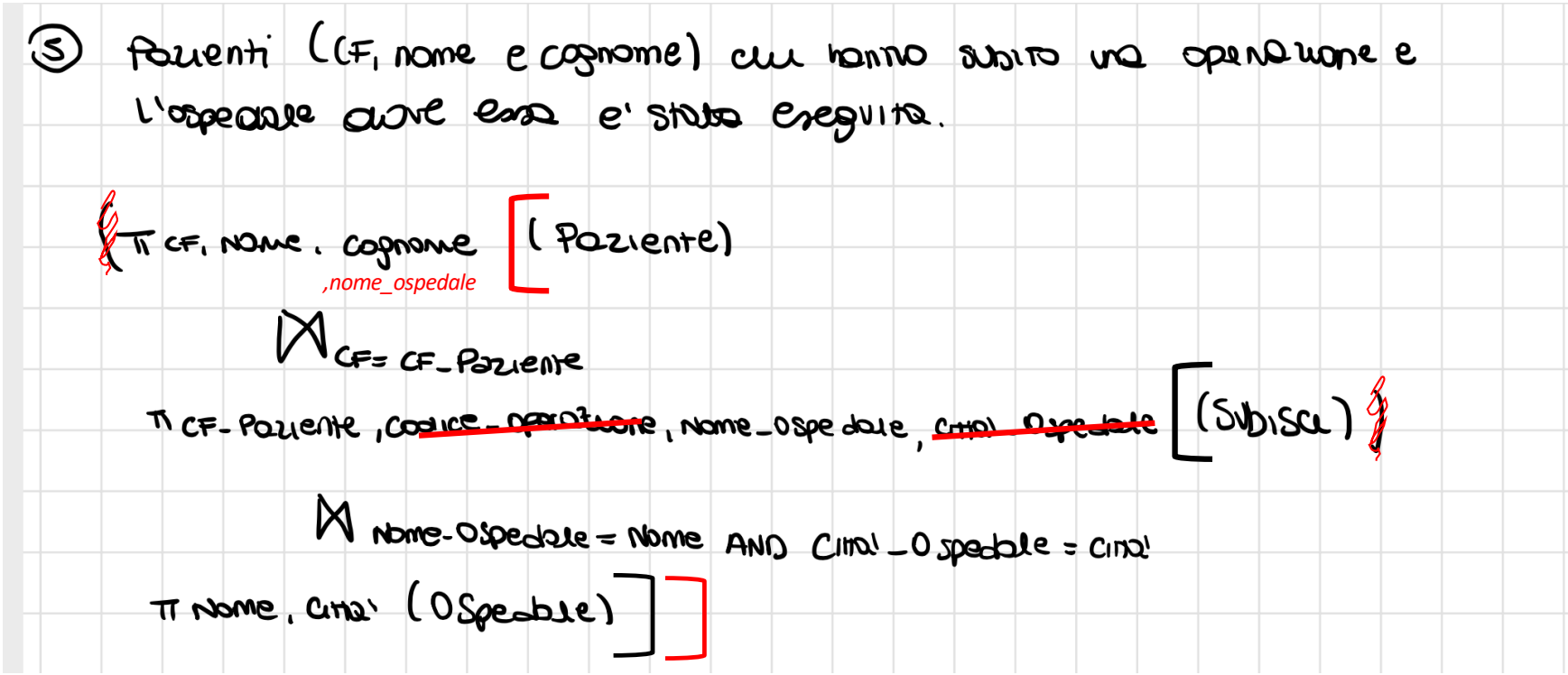
Usando le parentesi in questo modo non ci sarebbe stato l'errore di join perché l'attributo *Nome* in ospedale non è coinvolto nel Join con paziente, grazie alla proiezione (sottolineata in blu)

Errato uso delle parentesi



Tuttavia permane il problema che la query restituisce molti più attributi di quelli richiesti

Errato uso delle parentesi



Errore che possiamo risolvere con nuove parentesi e una proiezione più mirata

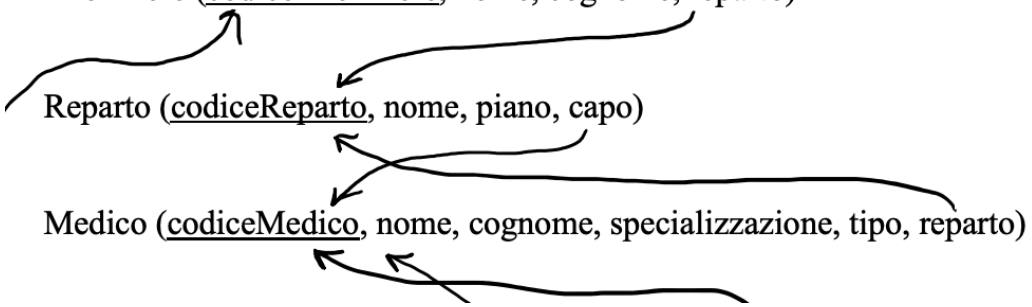
Esempio unione di relazioni definite su attributi diversi

PersonaleAmministrativo (codicePersonale, nome, cognome)

Infermiere (codiceInfermiere, nome, cognome, reparto)

Reparto (codiceReparto, nome, piano, capo)

Medico (codiceMedico, nome, cognome, specializzazione, tipo, reparto)



Query 1:
formulare un'espressione in algebra relazione che produca tutto il personale dell'ospedale

$REN_{codicePersonaleAmministrativo<-codicePersonale}$ (Medico U Infermiere U PersonaleAmministrativo)

Esempio di ridenominazione poco utile

2) Formulare un'espressione in algebra lineare che produca codice fiscale, nome e cognome dei pazienti affetti da polmonite e del personale amministrativo col ruolo di segretario.

$$\left(\Pi_{CF,Nome,Cognome}\left(\rho_{CF \leftarrow Cod\_F}\left(\sigma_{NomeP='Polmonite'}\left(\right.\right.\right.\right.$$

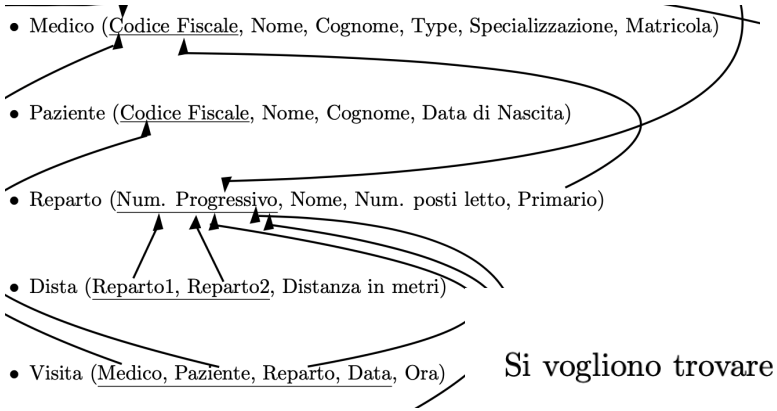
$$\left.\left.\left.\left.PAZIENTE \bowtie_{Patologia=Codice}\left(\rho_{NomeP \leftarrow Nome}(PATOLOGIA)\right)\right)\right)\right)\right)$$

$$\cup$$

$$\left(\Pi_{CF,Nome,Cognome}\left(PERSONALE \bowtie_{ID=Codice\,Personale}\left(\sigma_{Ruolo='segretario'}\right.\right.\right.$$

$$\left.\left.\left.(P\_AMMINISTRATIVO)\right)\right)\right)$$

Esempio ridenominazione utile ad evitare errore nel join



Si vogliono trovare il nome, cognome e codice fiscale dei pazienti visitati dal Dottor Mario Rossi.

$$(\pi_{NomePaziente,\, CognomePaziente,\, CodiceFiscalePaziente}$$

$$(\rho_{NomePaziente,\, CognomePaziente,\, CodiceFiscalePaziente \leftarrow Nome,Cognome,CodiceFiscale}\ (Paziente))$$

$$\bowtie_{CodiceFiscalePaziente=Paziente}\ Visita)$$

$$\bowtie_{Visita.Medico=Medico.CodiceFiscale}\ (\sigma_{Nome=\text{"}Mario\text{"}\ AND\ Cognome=\text{"}Rossi\text{"}}\ (Medico)))$$

Attenzione però che la query scritta in questo modo restituirà anche gli attributi Medico e relativo CodiceFiscale (problema parentesi)
nonché all'uso improprio del pinto (.)

SQL

Errori da evitare

Errori gravi molto comuni

- Assenza predicato di Join

Errori gravi meno comuni

- Errore visibilità variabili
- Errore utilizzo del group by

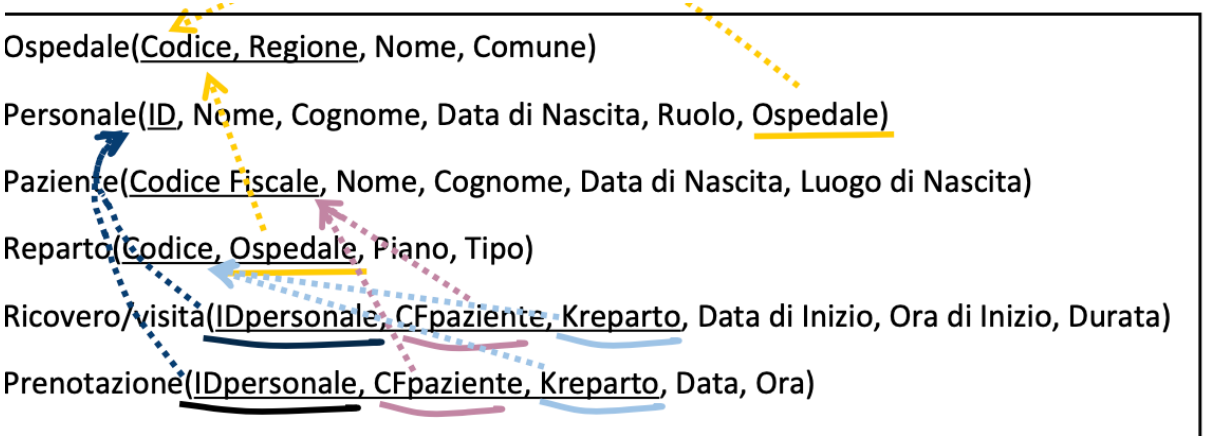
Imprecisioni comuni

- Effettuare un Join inutile ai fini del testo della query
- Utilizzare inutilmente una subquery nella clausola FROM
  - Inclusa subquery in clausola JOIN

Strategie poco premiabili

- Riciclo query
- Testi query poco interessanti

Predicato di Join assente

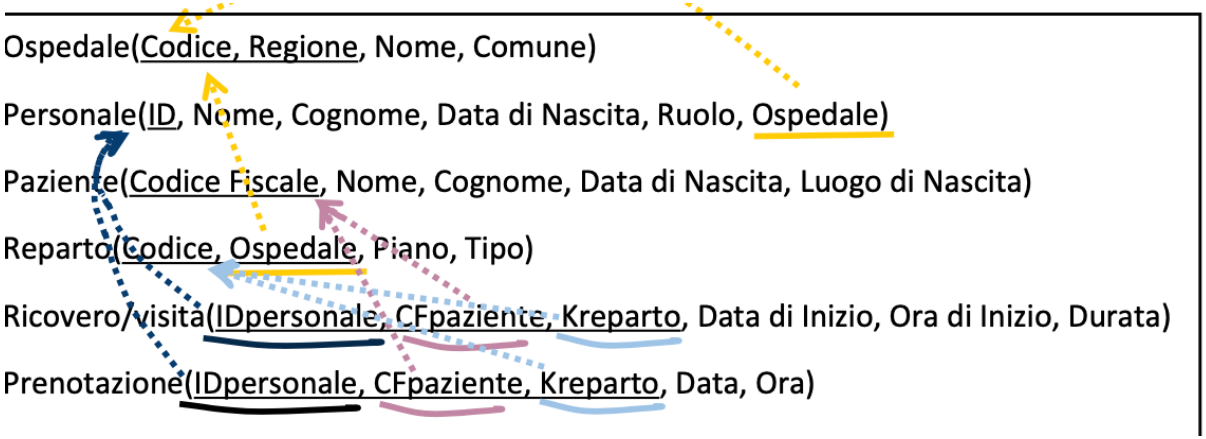


Selezionare i nomi e i cognomi dei medici che hanno una prenotazione per il 5 giugno 2020

```
SELECT Nome, Cognome
FROM Personale, Prenotazione
WHERE Prenotazione.Data = '2020/06/05' AND Personale.Ruolo="Medico"
```

Tutte le prenotazioni in data x verrano combinate con tutto il personale medico indiscriminatamente

Predicato di Join assente



Selezionare i nomi e i cognomi dei medici che hanno una prenotazione per il 5 giugno 2020

```
SELECT Nome, Cognome
FROM Personale, Prenotazione
WHERE Prenotazione.Data = '2020/06/05' AND Personale.Ruolo="Medico"
        AND Prenotazione.Idpersonale=Personale.ID
```

È necessario il predicato di join

Riferimento ad attributi della query interna nella query esterna

```
SELECT CF, NOME, COGNOME, NUMERO
FROM PAZIENTE
WHERE CF IS IN ( SELECT CF_PAZIENTE COUNT (*) AS NUMERO
                 FROM SUBISCE )
GROUP BY CF
```

?

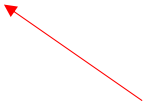
Non visibile da query esterna

Una variabile è visibile solo nella query che l'ha definita o in una sua subquery

Errore nell'uso group by

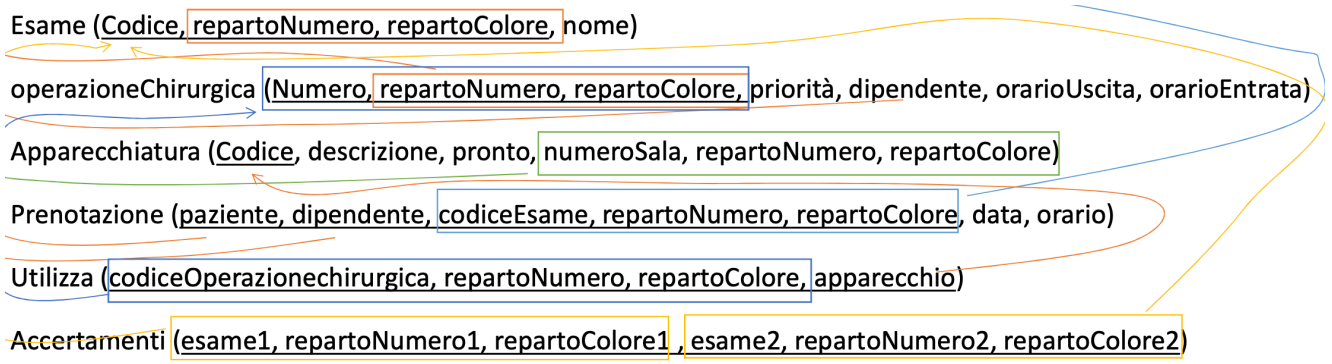
3. Il numero di visite che ha fatto ogni paziente da un medico Oculista e Dermatologo

```
SELECT Paziente.Cognome, Paziente.Nome, count(*) AS Numero
FROM Paziente, Visita
WHERE Paziente.Codice fiscale = Visita.Paziente AND Visita.Medico IN (
        SELECT Badge
        FROM Medico
        WHERE Specializzazione NOT IN ("Oculista", "Dermatologo");)
GROUP BY Paziente.Cognome, Paziente.nome;
```



In questo modo andiamo a raggruppare per cognome e poi contiamo le visite fatte da ciascun paziente con lo stesso cognome ma nome diverso

Esempio (self) Join inutile



- Vedere quali esami possono comportare a ulteriori esami di accertamento

```
SELECT e1.*
FROM esame AS e1, accertamenti AS a, esame AS e2
WHERE e1.codice = a.esame1 AND e1.repartoNumero = a.repartoNumero1 AND e1.repartoColore=a.repartoColore1 AND e2.codice = a.esame2 AND
e2.repartoNumero = a.repartoNumero2 AND e2.repartoColore=a.repartoColore2
```

Query corretta ma che informazione aggiunge la seconda tabella esame visto che proiettiamo solo gli attributi di E1?

Atro esempio di join inutile

```
SELECT med.codice, med.nome, med.cognome
FROM medico AS med INNER JOIN operazione_chirurgica AS oper
WHERE med.codice NOT IN (SELECT medico
                         FROM operazione_chirurgica)
```

Non aggiunge nessuna informazione

Esempio di Join inutile (su schema errato)

Esame(codiceID, data, ora, tipologia, apparecchiatura)
Prenotazione(codice, tipologia)

1) Mostrare le informazioni di un esame, ovvero codiceID, data, ora, tipologia e apparecchiatura, quando la tipologia dell'esame è "TAC".

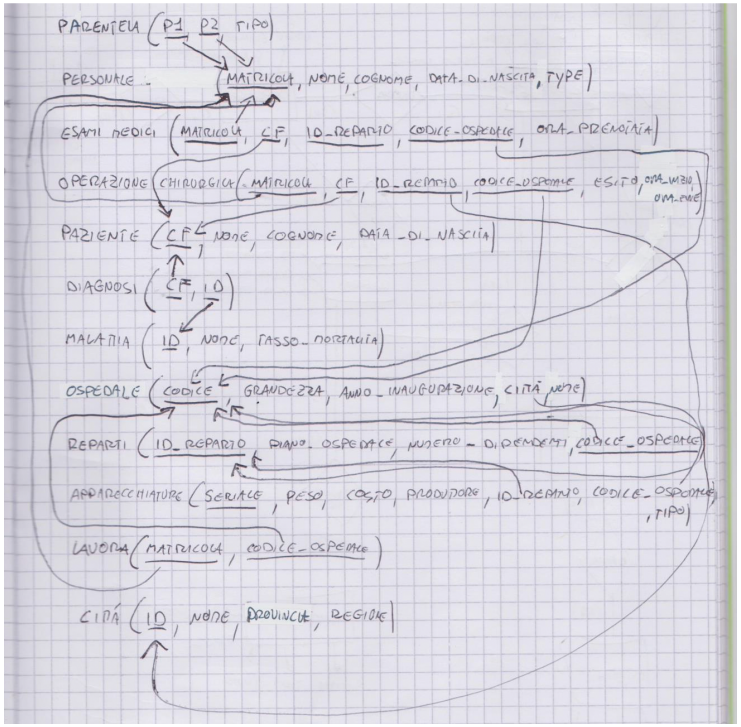
```
SELECT * FROM Esame INNER JOIN Prenotazione ON
Esame.tipologia = Prenotazione.tipologia WHERE
Prenotazione.tipologia = "TAC";
```

Trascurando il problema con lo schema, che informazione aggiunge la tabella Prenotazione in questa query?

Esempio discordanza testo e soluzione

3. Selezionare il numero di dottori per regione con il numero totale di interventi positivi effettuati in ‘cardiologia’ in ospedali inaugurati dopo il 2009

```
SELECT c.regione, count(*)
FROM citta as c
JOIN ospedale as o on o.citta = citta.id
JOIN lavora as l on o.codice = l.codice_ospedale
JOIN personale as p on l.matricola = p.matricola
JOIN operazione_chirurgica as oc on oc.matricola = p.matricola
WHERE p.matricola in (  SELECT matricola
                        FROM operazione_chirurgica
                        WHERE esito = ‘positivo’
                        AND id_reparto = ‘cardiologia’)
AND anno_innaugurazione > 2009
GROUP BY c.regione
```



La query è corretta, ma non fa quello che dice di fare: seleziona il numero di dottori per regione che ha effettuato ALMENO un intervento positivo in cardiologia

Esempio di subquery in clausola FROM inutile

```
select paziente, sum(costo)
from
      (
      select po.paziente, o.costo
      from PrenotaOperazione po
      inner join Paziente p
            on po.paziente = p.codicefiscale
      inner join Operazione o
            on po.codiceOperazione = o.codice
      where po.data >= to_date('04/06/2020','dd/mm/yyyy')
      and p.cognome = 'Rossi'
      and p.nome = 'Mario'
      )
group by paziente
```

A cosa serve la subquery nella clausola FROM quando basta fare i vari Join direttamente nella clausola from?

Esempio di subquery in clausola FROM poco convenzionale

Visualizzare il nume del paziente e data delle prenotazioni che durano 2 (120 minuti) ore dei pazienti con cognome Bianchi.

```
SELECT nome, data

FROM (SELECT * FROM Prenotazione WHERE durata = 120) AS T1, (SELECT * FROM Paziente WHERE
cognome = 'Bianchi') AS T2

WHERE T1.codice_fiscale_FK = T2.codice_fiscale;
```

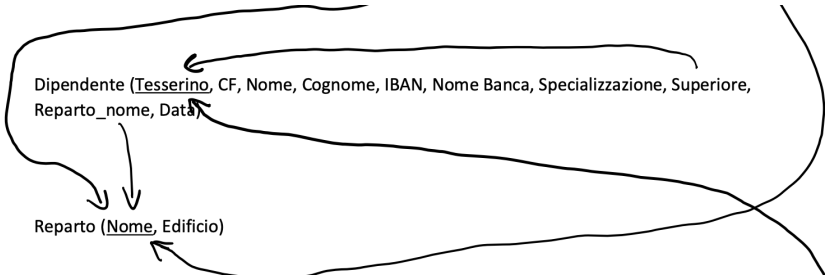
Fare selezione e proiezione prima del JOIN va bene in AR ma non è desiderabile in SQL. SQL è un linguaggio principalmente dichiarativo, nonostante i nuovi standard permettano maggiore «proceduralità».

Esempio di subquery in clausola WHERE utile

2. Selezionare i dati dei dottori nati prima del 1972 che non sono riusciti a salvare la vita ad almeno un paziente durante un'operazione chirurgica

```
SELECT *
FROM personale as p
WHERE p.matricola in (  SELECT matricola
                        FROM operazione_chirurgica
                        WHERE esito = 'negativo')
AND data_di_nascita < '1972-01-01'
```

Esempio di Query complessa ben fatta




```
Trovare il dipendente presente nel reparto Medicina da più tempo e il num
ero dei colleghi di cui è superiore.

SELECT
    D1.Tesserino,
    count(D2.Tesserino) as NumeroSottoposti
FROM
    Dipendente D1 LEFT JOIN
    Dipendente D2 on D1.Tesserino = D2.Superiore
WHERE
    D1.Reparto = 'Medicina' AND
    D1.Data = (
        SELECT MIN(Data)
        FROM Dipendente
        Where Reparto = 'Medicina'
    )
GROUP BY
    D1.Tesserino
```

Esempio di efficiente riciclo delle query

Infermiere(cod_fiscale, nome, cognome, titolo_studio, anno_assunzione, *cod_reparto*)

Medico(cod_fiscale, nome, cognome, specializzazione, anno_assunzione, *cod_reparto*)

Personale_amministrativo(cod_fiscale, nome, cognome, anno_assunzione, mansione)

Reparto(codice, nome)

Esame(codice, tipo, descrizione, *cod_reparto*)

Apparecchiatura_esame(*cod_esame, cod_apparecchiatura*)

Apparecchiatura_diagnostica(codice, nome)

Paziente(cod_fiscale, nome, cognome, anno nascita)

EsamePrenotato(*cod_esame, cod_paziente*, data, ora, *cod_medico*)

EsameSvolto(*cod_esame, cod_paziente*, data, ora, *cod_medico*, esito)

Un'ottima strategia se si ha poco tempo a disposizione, non per chi punta alla Lode!

6

Visualizza gli esami disponibili nel reparto Pneumologia

```
SELECT Esame.*
FROM Reparto, Esame
WHERE Reparto.nome = Pneumologia, Reparto.codice = Esame.cod_reparto
```

Visualizza gli esami svolti relativi a gli esami disponibili nel reparto Pneumologia

```
SELECT EsameSvolto.*
FROM EsameSvolto
WHERE cod_esame = ANY(
        SELECT Esame.codice
        FROM Reparto, Esame
        WHERE Reparto.nome = Pneumologia, Reparto.codice = Esame.cod_reparto)
```

Visualizza il numero degli esamiSvolti nel reparto Pneumologia per ciascun paziente

```
SELECT cod_paziente AS paziente, COUNT(cod_esame)
FROM EsameSvolto
WHERE cod_esame = ANY(
        SELECT Esame.codice
        FROM Reparto, Esame
        WHERE Reparto.nome = Pneumologia, Reparto.codice = Esame.cod_reparto)
GROUP BY cod_paziente
```

Esempi di query non molto interessanti

- Tutti i pazienti che hanno nome di personale che hanno nomi di città
- Contare il numero del personale suddiviso in classe di stipendio che non percepisce lo stipendio massimo