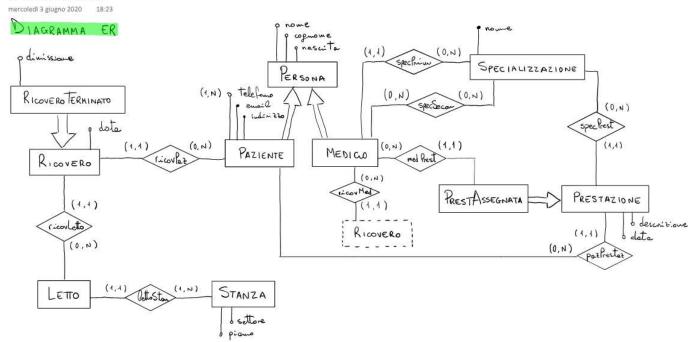
#### P. QuickHospital



## SPECIFICHE DEI DATI

rome stringe telefono (1,N) Telefono
cognome stringe email stringe
uescuta data indirizzo Indorizzo

Entité Stanta Entité Specializzatione E piono interno > 0 nome straingra d settere interno > 0

Entita Prestorione data data descrizione strigia

# V. Ricoveraterminato. domissione. dopo Datahirovera

∀it, 1, 11 Ricovero Terminato (rt) ∧ data (rt, d) ∧ dimissione (rt, dd) -> d ≤ dd

# V. Ricovero. peniodo Stesso Letto

Vr, d, l Ricoveno (r) Λ data (r, d) Λ micovletto (r, l) ->
((∃dl Ricoveno Terminato (r) Λ dimissiona (r. dd) -> ¬∃ rz, dz Ricoveno (rz) Λ data (rz, dz) Λ dz ≥ d Λ
Λ dz ≤ dd Λ ricovletto (rz, l) Λ r + rz) V (¬∃ rz, dz Ricoveno (rz) Λ data (rz, dz) Λ dz ≥ d Λ nicovletto (rz, l) Λ rz+r))

# V. Ricovers. periodo Stesso Paziente

Vr.d, p Ricoveno (r) Λ date (r, d) Λ nicovPez (r, p) —>

( (∃dd Ricoveno Terminato (r) Λ damossione (r, dd) —> ¬∃ rz, dz Rocoverno (rz) Λ date (rz, dz) Λ dz ≥ d Λ

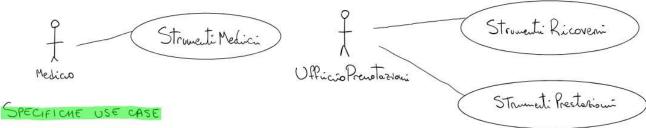
Λ dz ≤ dd Λ ricov Poz (rz, p) Λ r ≠ rz) V (¬∃ rz, dz Ricoveno (rz) Λ date (rz, dz) Λ dz ≥ d Λ nicovPoz (rz, p) Λ rz ≠ r))

# [V. Persama. nascta. primarkicovena] \( \text{e., pe, d (Ricovero(e) \( \) (nicovPar(e, pa) \( \) \( \) NascavMed(e, pe)) \( \) Nascata (pe, d)) \( --> \) \( \) Let diata(e, de) \( \) \( \) Persama. nascta. prima Prestatione \( \text{V. Persama. nascta. prima Prestatione} \) \( \text{V. persama. nascta. prima Prestatione} \) \( \text{V. persona. neutroPariete. el neue Una} \) \( \text{V. Persona. metroPariete. el neue Una} \) \( \text{V. Persona. metroPariete. el neue Una} \) \( \text{V. Persona. (p) \cdots \) \( \text{Persona. (p) \cdots \) \( \text{Medico. Spec. Lisp.} \) \( \text{V. Medico. spec. Lisp.} \) \( \text{M. Medico. (m) \cdots \) (3 s. spechim. (m, s.) \( -> \) \( \) specSecou. (m, s.) \)

V. Parsente. Nicov Prestaz. alueno Umo

∀p Poriente (p) -> ∃e NovPor(p. e) V porPrestor (p. e)

### DIAGRAMMA UML USE- CASE



Use-Case StrumetoRocover

Posti Disposibilii (d: data): intero ≥ 0

Preconditioni: nessume

postionalitioni:

Moderna livello estensionale dei dati: nessuma

Velore di nitorna:

L+ot = {l | ∃ s Stanza(s) ∧ lettoStan(l, s)}

Locc = {l | ∃ r.dr (Riccovero(r) ∧ ¬ RiccoveroTerminato(r) ∧ data(r, dr) ∧ d > dr) ∨

(RiccoveroTerminato(r) ∧ data(r, dr) ∧ d × dr)}

result = | L+o+ | - | Locc |

· registra hicoverno (p: Paz Intermo, l: Letto, m: Medica): Ricovero precondizioni: postiDisponibili (ADESSO) 1 73 r ricov Letto (r, l) post condizioni:

Modifica livello estensande dei dati:

Voncoriour dominio di interpretarione: nuovo elemento a

Modifica livello estensande dei dati:

Vanioriouri dansiurio di interpreterioure: nuovo elemento a

Variationi emple di predicato:

- · Ricoveno (a)
- · date (a, ADESSO)
- · ricov Paz (a, p)
- · ricov Letto (x, l)

· ricov Med (a, m) Velore do ritorno: result = a

· dimissione Ricovens (1: Ricoveno) : Ricoveno terminato precondizioni: ressura

post conditioni:

Modernica dominio de enterpretazione:

Voriarione dominio de atterpretarione: nessura

Verrierieni encuple de predicato:

· Ricovero Terminato (r)

· dimissione (r, ADESSO)

Velore do natorno: result = r

# Use-Case Structo Prestorioni

· medici Idonei (p: Prestatione): Medicao (O,N) precondizioni: - Prest Asseguata (7) post condizations:

Modoficia livello estensionale dei dati: nessume

Valore do natormo:

Pri = { m | ] sp specPrest(p, sp) 1 specPrim(m, sp) }

Sec = { m | 3sp specPrest (p, sp) 1 specSecon (m, sp)}

result = { Pri se | Pri ) 0

## Use-Case Struments Medsai

· itsueranio Visite (m: Medica)

preconditioni: nessuma postcondo sioni:

Modoficia dominio di interpretazione: nessura

Velore do notorno:

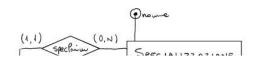
V = { (pi, se) | Ir, l, s nicovMed (m, r) ~ ricovLetto (r, l) ~ letto Stan (l, s) ~ } ~ pieuro (s, pi) ~ settore (s, se) ~ ~ Ricovero Terminato (r)}

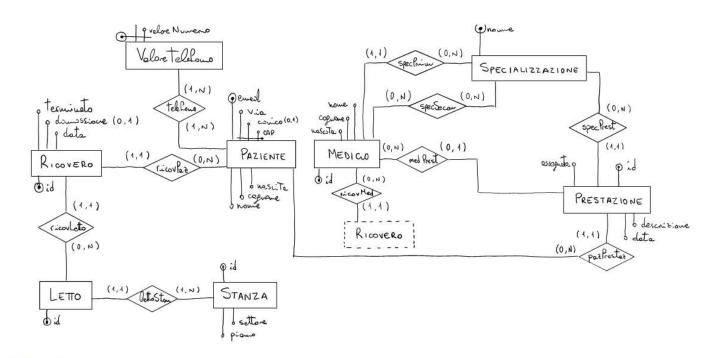
result = V

NOTA: l'insierre V donne; in frase de propettazione, essere ordinets in base al piono, e o positor di para in base al settore.

# RISTRUTURAZIONE ER







#### NUOVI VINCOLI

Vr (Ricoveno(r) n terminato(r, true)) <--> 3d dimissione(1, d)

∀ pr (Prestorioue(pr) Λ assegnato(pr, tive)) < → ) ∃ m med Prest (m, pr)
</p>

#### VINCOLI MODIFICATI

# V. Ricoveno Terminato, dimissione, dopo Detahiroveno

Y, t, 1, 11 Ricovers (rt) ∧ Terminato (rt. true) ∧ data (rt, d) ∧ dimissione (rt, dd) -> d ≤ dd

# V. Ricovero, peniodo Stesso Letto

∀r,d,l Ricoveno (r) A data (r,d) A micovletto (r,l) ->

((∃dd terminato (r, true) n damissione (r, dd) -> ¬∃ rz, dz Racoverno (rz) n data (rz, dz) n dz ≥ d n dz ≤ dd n nicovletto (rz, l) n r + rz) v (¬∃ rz, dz Racoverno (rz) n data (rz, dz) n dz ≥ d n nicovletto (rz, l) n rz+r))

# V. Ricovers. persodo Stesso Pariente

∀r,d,p Ricoveno (r) ∧ data (r,d) ∧ nicovPez (r,p) ->

((∃dd Terminato (r, True) A domissione (r, dd) -> ¬∃ rz, dz Rocoverno (rz) A data (rz, dz) A dz≥d A dz ∠dd A A ricov Pere (rz, p) A r ≠ rz) V (¬∃ rz, dz Ricoverno (rz) A data (rz, dz) A dz≥d A nicov Pere (rz, p) A rz≠r))

## RISTRUTTURAZIONE SPECIFICHE DATI

Damino Strings create domain Strings as virolar (100);

Damino IntegerGZ es integer chem (value > 0);

Downso String M create domain String M es varcher (200);

Daninso Prelissoteletono creste domain Prelissoteletono 25 Varchar (5) cheex (value ~ '^\+ [0-9]+\$');

```
Dominio Numero Tele Formo
                                          Danino CAP
create domain Nonpotelelano as char (10) create domain CAP as char(5)
                                           check (velue ~ '^[0.9][5,5]$');
   chear (value ~ '~ [0,9] (10,10]$');
                                    Entota Ricoveno Entota Letto Entita Stanza
Entota Medico Entota Porvente
                          Strugs
                                                         id integer
        integer
                                      id outger
                                                                      ad integer
 id
                  nome
        STrugS
                  copuome STALYS
                                             date
                                                                        piono Integer GZ
cogname STALYS
                  nascuta date
                                     Terminato boolean
                                                                        settore Integer G Z
nascuta date
                  email Strings
                                     dimissione date
                          STrigeM
                  Via
                  wrice Integeris?
                  CAP
                          CAP
                                               Entota Volore Telefono
                          Eulita Prestatione
Entito Specializzazione
                                               volore Prelisso Prelissotelehono
nome Strings
                                   inleger
                                               valore Nunero Nunero Tele Lono
                                   data
                          data
                          descrizione stringia
                          assegueta baslean
SCHEMA RELAZIONALE
Stanza (id: serial, piano: IntegerGZ, settore: IntegerGZ)
   inclusione: id = Letto(stanza)
Letto (id: serial, stanza: integer)
   Loresper ney: stanta references Stanta (id)
Kiscoverio (id: serial, dota: date, terminato: boolean, divissione*: date,
               Otto: integer, portiente: strings, medica: integer)
   Lorengen key: letto references Letto (id)
   Loreign key: pariete (alerences Pariete (email)
   Lorespu key: medico references Medico (id)
   enuopha: teruminato = true (-) dimissione + NULL
   enupho: dimissione of NULL -> dimissione > disto
Pariente (email: Strings, nome: Strings, cognome: Strings, resulta: date,
   Wa: StringM, covico*: IntegerGZ, CAP: CAP; CAP; inclusione: emeil = Telefone (poriente)
   chiave: (via, civico, CAP)
Telefono (valore Prefussio: Prelisco Telefono, valore Nuneno: Numeno Telefono, pariente: Straing S)
   Foesgu key: passente references Possente (email)
Prestazione (id: serial, descrizvoue: StringM, data: date, asseguete: booken,
            medico*: integer, porsiente: STraing S, specializzazione: "STraing S)
   Porespur key: medico references Medica (2d)
   Corespo key: porverte references Pasiente (email)
  Lorezen key: specializzatione references Specializzatione (name)
  eunopha: medico / NULL (-) assegnata = true
Medico (id: serial, nove: Strings, cognome: Strings, rescuta: date, specknim: Strings)
   forevous key: specPrim references Specializzarione (nome)
Specializzatione (nome: Strings)
```

```
Specializzatione (nome: 51 ming )
specSecon (medico: integer, specializzarione: Strings)
   foreven key: medicio references Medico (id)
   loreign rey: specializzazione references Specializzanione (mome)
PROGETTAZIONE VINCOLI ESTERNI
Vincolo V. Stanza. max Letti
Tripper:
  Gerorion intercettate: inserimente cumplia relatione Letto
  Istante di invocatione: prima operazione intercetteta
     new = emupha dua insendre
        select court (*) < 8 es lette Appi up: bale
        from Stenza s
        where s.id = new. stanta
    IF il velore della colonna letto Ages upilale di a == true
      permetti l'operazione
    ELSE blocca l'operatione
Vincolo V. Melico. Spec. Wsp.
Tripper:
   Operardani sutercettete: inserimento o modifica emupla relazione Medica
  Istante di Duvocortone: prima operazione sutercettate
   Furzione:
      new : enumple de deserte o risultato modifica
      has Error =
        exists (
           select *
           from specSecon ss
           where ss. median = new. id
             and ss. specializzatione = new. spec Prom
     IF hestinor THEN blocce operatione
     ELSE permetti operarione
Tripper:
   Operariani sutercettete: inserimento o modifica emple relazione specSeción
  Istante di Duvocortone: prima operazione intercettata
  fourtione:
      new = encuple da ilisarire o risultato modifica
     has Error =
        exists (
           select *
           from Medica m
          where m.id = new. medicao
             and m. specPrim = new. specializes sione
     IF hestinor THEN blocce operatione
     ELSE permetti operarione
```

```
Viucolo V. Ricoveno. peniodo Stessoletto
Tripper:
   operorioni intercettate: inserimento o modifica emupho relorione hicovero
   Istante DuvocoNove: prima operazione intercettata
   fun Houre:
      new = emuple da inserire o resultato modifica
      has Error = exists (
         select *
         from Ricoveno r
        where new id <> r.id
           and new letto = r. letto
           and ( new. date, new. dimissione) overlops ( r. date, r. dimissione));
      IF has Error THEN blocks operatione
     ELSE permetti operacione
 Viucolo V. Ricoveno. peniodo Stesso Portente
Tripper:
   operorioni intercettate: inserimento o modifica emupho relorione hicovero
   Istante Duvo costone: prima operarione intercettata
      new = emuple da inserire o resultato modifica
      has Error = exists (
         select *
         from Ricoveno r
         where new id <> r. id
           and new. partete = r. partete
           and ( new. data, new. diruissione) overlops (1. data, 1. diruissione));
      IF hes Error THEN blocks operatione
      ELSE permetti operazione
Vincolo V. Persona. Mascita. prima Ricovers
Trugger:
  operazioni intercettate: inservento o modifica emplea relatione kicoverno
  Istante invocatione: prime operatione intercettata
  turiare:
     new = emuple de inserire o risultato modofica
      has Ermor = exists (
        select *
       from Medico m, Porvente p
        where new possente = p. emosl
         and new data < p. nascite
           or new. medico = m. id
          and new date < m. nesote);
     IF hostinar THEN blocce operations
     ELSE permett: openostione
Tropper:
   operazioni intercettete: modificia enupla relatione Medico
```

IF hestinor THEN blocce operations

ELSE permetti operazione

```
Tropper:
   operazieni intercettete: modificia empla relatione Medico
  Istente auxocorone: prima operariano intercettate
     new: euripla willoute modifica
     hestrior = exusts (
        select *
        From Ricovers r
        where r. medico = new. id
          end r. date < new. nescota);
     IF hasterror THEN blocks peroxiene
     ELSE permetts operarione
Trugger:
   Operations atorettete: modifica emplo relatione Posite
  Istate invocarione: prome openarione intercettata
   Fundame:
     vew = emuple visultate modifica
     hastinor = exists (
        select *
        from Rowero r
        where r. partente = new. email
          and r. data < new. hascita );
     IF hostrior THEN blocca openatione
     ELSE permetto operariane
Vaucolo V. Persono. Masoto. prima hesterlane
Tripper eneloghes a quelli in V. Persona, voseta, privua Riconerro sostituendo la relatione Ricovero con la relatione Prestaname
Vincolo V. Pariente, nicov Prestar, alueno Uno
Tripper:
   Option intercettate: inserimento emplo relatione Pariente
   Islante invocazione: dopo operazione intercettata
   toutione:
      new = emupho do inserire
      hastinor = not exists (
        select *
        from Ricovenor, Prestosione p
        where r. pariente = new. email
          or p. pariente = new. emenl );
     IF has Einor THEN blocks operations
     ELSE permetto operazione
```

## SPECIFICHE REALIZZATIVE USE-CASE

Use-cose StometiPrestoriemi

· medicii Idonei (p: integer): Insieme (<id: integer, nome: StringS, cognome: StringS, nascite: date, specPrim: StringS>)

```
medicii Idoneii (p: integer): Iusieme (<id: integer, nome: Strings, cognome: Strings, nascita: date, specPrim: Strings)</li>
res = risultato del comando sal segmente sistituendo a ':p' il volore dell'amonimo parama attuele select m.id, m. nome, m. cognome, m. noscita, m. specPrim from Medico m, Prestatione pr where: p = pr.id and pr. specializzorione = m. specPrim
IF res == NULL THEN
res = risultato del comando sal segmente sistituendo a ':p' il volore del parama attuele p select m.id, m. nome, m. cognome, m. noscita, m. specPrim from Medicio m, Prestatione pr, specSecon ss where: p = pr.id
and pr. specializzorione = ss. specializzorione and pr. specializzorione = m.id
IF res == NULL THEN penera l'emore 'non esistono medici con questo, specializzorione'
```

## Use-case Struct Medici

ELSE restituisi res

· iToueromo Visate (m: integer): Collerione ((piono: IntegerGZ, settore: IntegerGZ))

res = risultato del comuncho SQL seguente sostituendo a ':m' il valore del parametro attuele m

select s. piono, s. settore

from STenza s, Ricaveno r, Letto l

where :m = r. medico

and r. terminato = helse

and r. letto = l. id

and l. sTenza = s. id

order by s. piono ASC, s. sottore ASC;

IF res == NULL THEN penerua l'ernore 'nessuma virsata da mostrare'

ELSE restatuisi res

## Use-case Strumentihocoveni

registraticovero (p: integer, l: integer, m: integer): integer

IF DB, postoDispanibili (CURRENT\_DATE) restituisce un ernore

THEN inoltra l'ernore

ELSE IF DB. postoDispanibili (CURRENT\_DATE) < 1

THEN ternorma con l'ernore 'posto dispanibilii non sufficieto'

res = risultato del comendo SQL segnante sostotuendo a 'p'.':l'. ': m' i valori dei parametri

insert into Riccovero (data, terminato, letto, poriente, medico)

Volues (CURRENT\_DATE, felse, :l, :p, :m)

returning id

IF res impresenta un ermora THEN implane l'ernore

ELSE restrotuscii il valore della colama 'id' dell'unica emupha di res

· domissione Ricavero (r: integer)

update Ricavero set terminato = True, dimossione = CURRENT\_DATE

where id = r;

# Si deliuiscomo le seguenti Purroni nel DBMS:

· DB. posta Duspouritable (d: duste): IntegerGEZ
res: nisultato della query Sal segmente sostituendo a ':d' il valore del parametro attuele d
select x. postitotali - y. posti Occupato as posti Dispouritable

```
res' = nisultato della query SQL sequente sostituendo a ':d' il valore del parametro ettuele d

select x. postitotali - y. postiOccupato as posti Disponibilio

fram (select count(*) as postioccupato

fram Letto ) x,

(select count(*) as postioccupato

fram licaverno;

Where (r. terminato = fralse

and r. dusta \(\perceq :d\))

or (r. terminato = true

and r. dusta \(\perceq :d\))

Qual r. dumissione > :d)) y

[F res == NULL THEN penena l'errore ' non ai sono litto o nicoverni'

ELSE restituisii la coolonna 'posto Disponibili' dell' unica enurpho di res
```