Relazione sul Progetto di Basi di Dati

Ruzzante Alex matricola 1030089,
Vegro Federico matricola 1009448

June 17, 2013

Abstract

Il progetto prevede la gestione dei processi relativi alla manutenzione preventiva su apparecchiature elettromedicali ospedaliere. In particolare, vengono elaborati i rapporti tecnici relativi a ciascun intervento effettuato da un tecnico contenenti i vari test. Inoltre vengono considerati committenti e tipi delle apparecchiature.

1 Descrizione dei requisiti

Il progetto consiste nella realizzazione di una base di dati che modelli le principali classi coinvolte nella gestione dei rapporti tecnici di manutenzione su apparecchiature elettromedicali presenti negli ospedali.
L'entità principale del caso di studio è il Rapporto Tecnico.
Di ognuno interessa l'ubicazione, la data, la tipologia(generico o specifico in base al tipo di apparecchiatura), ed eventuali note. Inoltre interessano il referente della manutenzione, il tecnico effettuante, l'ospedale in cui è stata effettuata e l'apparecchiatura coinvolta.
Un'apparecchiatura compare nel database nel momento in cui viene compilato un rapporto tecnico ad essa relativo, non interessano le apparecchiature presenti in ciascun ospedale sulle quali non è stato effettuato alcun intervento di manutenzione. Di un'apparecchiatura interessano marca, modello e codice.
I principali tipi di apparecchiature sono presenti nell'entità tipi con una breve descrizione testuale.
Su ogni tipo di apparecchiatura vengono effettuati dei test principali che poi andranno a comparire nei rapporti con descrizione del test e relativi esiti.
Un'ulteriore entità significativa modellata sono gli ospedali, di cui vengono memorizzati il nome, la città, l'indirizzo ed il telefono.
Infine le persone coinvolte nei rapporti, di cui interessa nome, cognome, data di nascita e recapito telefonico. Le persone possono essere referenti, con il relativo ruolo all'interno dell'ospedale oppure tecnici qualificati che lavorano presso un'azienda di manutenzione.

2 Progettazione concettuale

2.1 Lista delle classi

RAPPORTI TECNICI: modella un rapporto tecnico.

- Ubicazione: *string*
- Data: *date*
- Tipologia: (*Generico, Specifico*)
- Note: *string*

TEST: modella i Test.

- Descrizione: *string*

TIPI: modella i Tipi.

- Descrizione: *string*

APPARECCHIATURE: modella le Apparecchiature.

- Marca: *string*
- Modello: *string*
- Codice: *string*

OSPEDALI: modella gli Ospedali.

- Nome: *string*

- Città: *string*
- Indirizzo: *string*
- Telefono: *string*

PERSONE: rappresenta le Persone.

- Nome: *string*
- Cognome: *string*
- DataNascita: *date*
- Telefono: *string*

TECNICI: modella i Tecnici.

- Azienda: *string*
- Qualifica: *string*

REFERENTI: modella i Referenti.

- Ruolo: *string*

2.2 Lista delle associazioni

- Rapporti Tecnici-Test: **contiene**
  - Ogni rapporto contiene più test. Ogni test è contenuto in più rapporti.
  - Molteplicità: N:M
  - Totalità: totale in entrambi i versi.

- Test-Tipi: **compatibile**
  - Ogni test è compatibile con più tipi. Ogni tipo è compatibile con più test.
  - Molteplicità: N:M

- Totalità: totale in entrambi i versi.

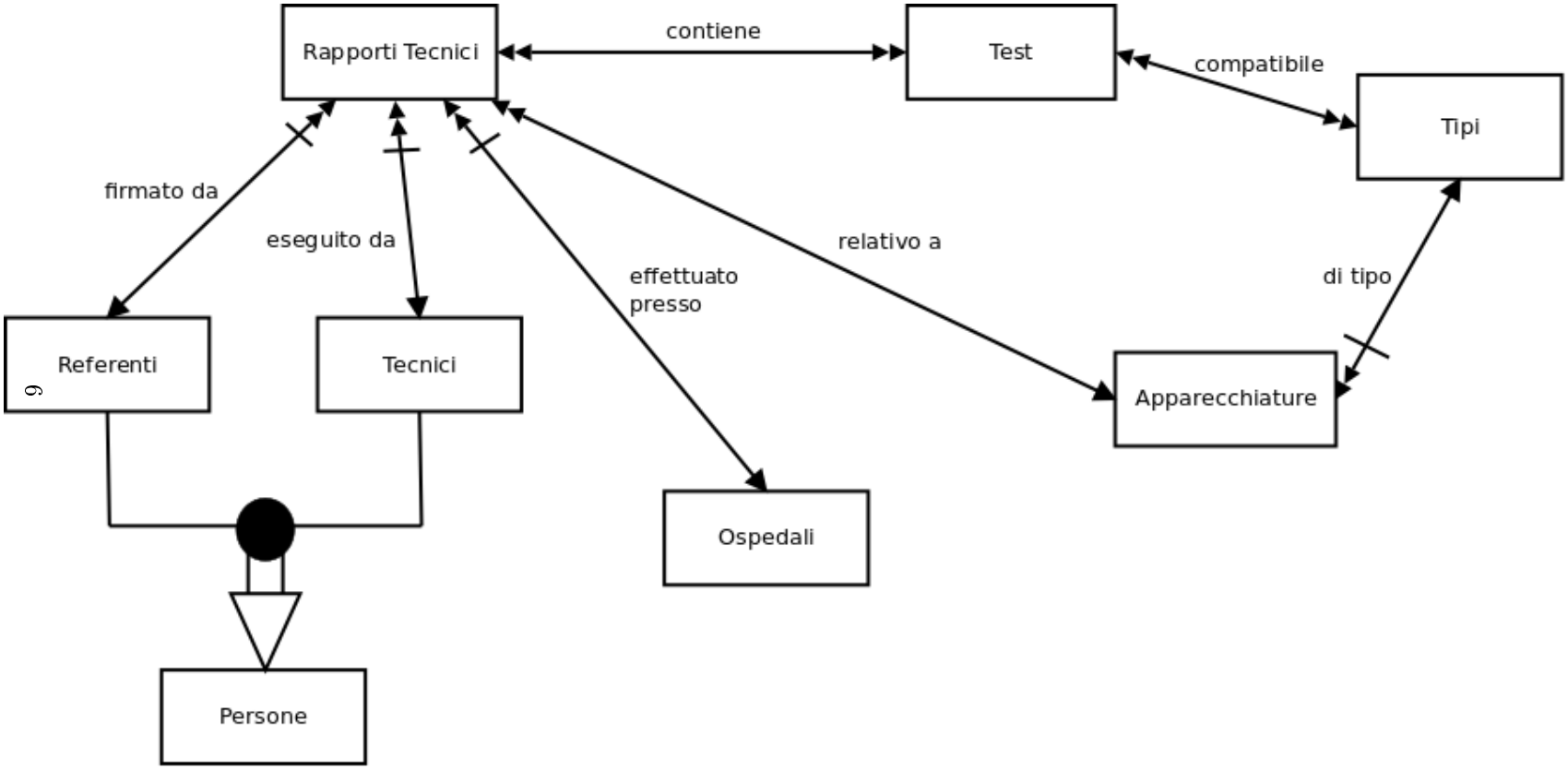
- Apparecchiature-Tipi: **di tipo**
  - Ogni apparecchiatura ha un solo tipo. Ogni tipo può appartenere a più apparecchiature o a nessuna.
  - Molteplicità: N:1
  - Totalità: parziale verso Apparecchiature, totale verso Tipi.

- Rapporti Tecnici-Apparecchiature: **relativo a**
  - Ogni rapporto tecnico è relativo ad una sola apparecchiatura. Ogni apparecchiatura può avere più rapporti tecnici.
  - Molteplicità: N:1
  - Totalità: totale in entrambi i versi.

- Rapporti Tecnici-Ospedali: **effettuato presso**
  - Ogni rapporto tecnico è effettuato presso un determinato ospedale. Ogni ospedale può avere più rapporti tecnici o nessuno.
  - Molteplicità: N:1
  - Totalità: totale verso Ospedali, parziale verso Rapporti Tecnici.

- Rapporti Tecnici-Tecnici: **eseguito da**
  - Ogni rapporto tecnico è eseguito da un tecnico. Ogni tecnico esegue più rapporti tecnici o nessuno.
  - Molteplicità: N:1
  - Totalità: totale verso Tecnici, parziale verso Rapporti Tecnici.

- Rapporti Tecnici-Referenti: **firmato da**
  - Ogni rapporto tecnico è firmato da un referente. Ogni referente firma più rapporti tecnici.
  - Molteplicità: N:1
  - Totalità: totale verso Referenti, parziale verso Rapporti Tecnici.


Rapporti Tecnici
contiene
Test
compatibile
Tipi
firmato da
eseguito da
effettuato
presso
relativo a
di tipo
Referenti
Tecnici
Apparecchiature
Ospedali
Persone

3 Progettazione logica

3.1 Gerarchie

La gerarchia della classe PERSONE è stata implementata con partizionamento verticale poichè le sottoclassi hanno attributi propri. La gerarchia completa è così formata: **PERSONE**: viene aggiunta una chiave primaria intera idPersona.

- idPersona: *int* «PK»
- Nome: *string*
- Cognome: *string*
- DataNascita: *date*
- Telefono: *string*

REFERENTI: idPersona è chiave esterna verso PERSONE.

- idPersona: *int* «FK(Persone)» «PK»
- Ruolo: *string*

TECNICI: idPersona è chiave esterna verso PERSONE.

- idPersona: *int* «FK(Persone)» «PK»
- Azienda: *string*
- Qualifica: *string*

3.2 Chiavi primarie

Sono state aggiunte chiavi primarie intere alle varie classi, si veda la lista completa più avanti.

3.3 Associazioni

- Rapporti Tecnici-Test: **contiene**
  - Ogni rapporto contiene più test. Ogni test è contenuto in più rapporti.
  - Molteplicità: N:M
  - Totalità: totale in entrambi i versi.
  - Nuova tabella **Contiene**, con i seguenti attributi:
    - idRapporto: int «FK(RapportiTecnici)» «PK»
    - idTest: int «FK(Test)» «PK»
    - Esito: (*Si, No, N.A.*)

- Test-Tipi: **compatibile**
  - Ogni test è compatibile con più tipi. Ogni tipo è compatibile con più test.
  - Molteplicità: N:M
  - Totalità: totale in entrambi i versi.
  - Nuova tabella **CompatibileCon**, con i seguenti attributi:
    - idTest: int «FK(Test)» «PK»
    - Civab: string «FK(Tipi)» «PK»

- Apparecchiature-Tipi: **di tipo**
  - Ogni apparecchiatura ha un solo tipo. Ogni tipo può appartenere a più apparecchiature o a nessuna.
  - Molteplicità: N:1
  - Totalità: parziale verso Apparecchiature, totale verso Tipi.
  - Chiave esterna non nulla in Apparecchiature verso Tipi.

- Rapporti Tecnici-Apparecchiature: **relativo a**
  - Ogni rapporto tecnico è relativo ad una sola apparecchiatura. Ogni apparecchiatura può avere più rapporti tecnici.
  - Molteplicità: N:1
  - Totalità: totale in entrambi i versi.
  - Chiave esterna non nulla in Rapporti Tecnici verso Apparecchiature.

- Rapporti Tecnici-Ospedali: **effettuato presso**
  - Ogni rapporto tecnico è effettuato presso un determinato ospedale. Ogni ospedale può avere più rapporti tecnici o nessuno.
  - Molteplicità: N:1
  - Totalità:totale verso Rapporti Tecnici, parziale verso Ospedali.
  - Chiave esterna non nulla in Rapporti Tecnici verso Ospedali.

- Rapporti Tecnici-Tecnici: **eseguito da**
  - Ogni rapporto tecnico è eseguito da un tecnico. Ogni tecnico esegue più rapporti tecnici o nessuno.
  - Molteplicità: N:1
  - Totalità: totale verso Rapporti Tecnici, parziale verso Tecnici.
  - Chiave esterna non nulla in Rapporti Tecnici verso Tecnici.

- Rapporti Tecnici-Referenti: **firmato da**
  - Ogni rapporto tecnico è firmato da un referente. Ogni referente firma più rapporti tecnici.
  - Molteplicità: N:1
  - Totalità: totale verso Rapporti Tecnici, parziale verso Ospedali.
  - Chiave esterna non nulla in Rapporti Tecnici verso Referenti.

3.4 Classi

RAPPORTI TECNICI: modella un rapporto tecnico.

- idRapporto: *int* «PK»
- Ubicazione: *string*
- Data: *date*

- Tipologia: (*Generico, Specifico*)
- Note: *string*
- idTecnico: *int* «FK(Tecnici)» «NOT NULL»
- idReferente: *int* «FK(Referenti)» «NOT NULL»
- idOspedale: *int* «FK(Ospedali)» «NOT NULL»
- serialeApparecchiatura: *string* «FK(Apparecchiature)» «NOT NULL»

CONTIENE: associazione RapportiTecnici - Test.

- idRapporto: int «FK(RapportiTecnici)» «PK»
- idTest: int «FK(Test)» «PK»
- Esito: (*Si, No, N.A.*)

TEST: modella i Test.

- idTest: *int* «PK»
- Descrizione: *string*

COMPATIBILE CON: associazione Test - Tipi.

- idTest: int «FK(Test)» «PK»
- Civab: string «FK(Tipi)» «PK»

TIPI: modella i Tipi.

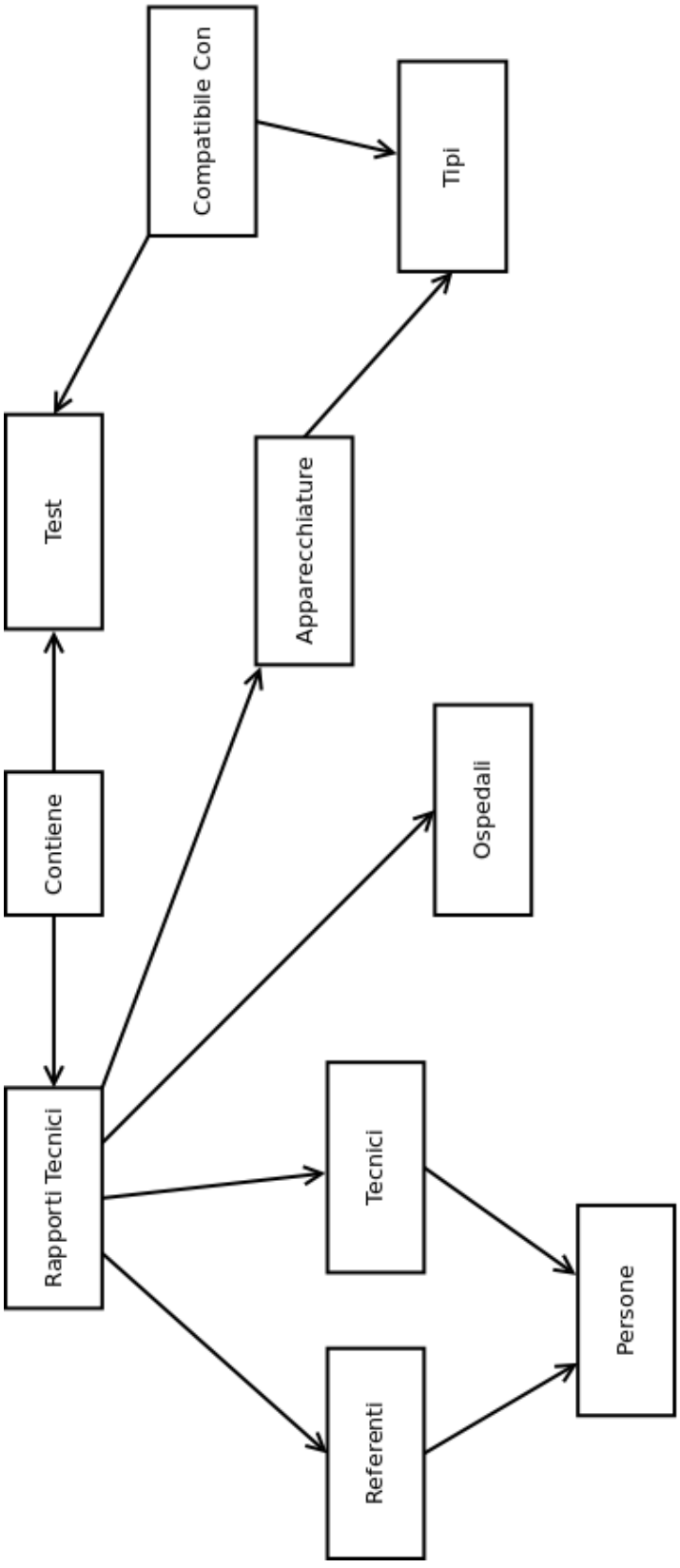
- Civab: *string* «PK»
- Descrizione: *string*

APPARECCHIATURE: modella le Apparecchiature.

- serialeApparecchiatura: *string* «PK»
- Marca: *string*
- Modello: *string*
- Codice: *string*
- Civab: *string* «FK(Tipi)» «NOT NULL»

OSPEDALI: modella gli Ospedali.

- idOspedale: *int* «PK»
- Nome: *string*
- Città: *string*
- Indirizzo: *string*
- Telefono: *string*


Compatibile Con
Tipi
Test
Apparecchiature
Contiene
Ospedali
Rapporti Tecnici
Tecnici
Persone
Referenti

4 Implementazione della base di dati

```
USE aruzzant_PR;

DROP TABLE IF EXISTS CompatibileCon;
DROP TABLE IF EXISTS Contiene;
DROP TABLE IF EXISTS Test;
DROP TABLE IF EXISTS RapportiTecnici;
DROP TABLE IF EXISTS Apparecchiature;
DROP TABLE IF EXISTS Tipi;
DROP TABLE IF EXISTS Ospedali;
DROP TABLE IF EXISTS Tecnici;
DROP TABLE IF EXISTS Referenti;
DROP TABLE IF EXISTS Persone;
DROP TABLE IF EXISTS Utenti;
DROP TRIGGER IF EXISTS CheckApp;
DROP TRIGGER IF EXISTS CheckNote;
DROP FUNCTION IF EXISTS calcoloEta;
DROP FUNCTION IF EXISTS calcoloSpesa;
DROP VIEW IF EXISTS Negativi;
DROP VIEW IF EXISTS SpeseOspedale;


CREATE TABLE Utenti(
        Login       CHAR(8)     PRIMARY KEY,
        Password    CHAR(40)
) ENGINE=InnoDB;

CREATE TABLE Test(
   idTest       INT PRIMARY KEY,
   Descrizione VARCHAR(255)
) ENGINE=InnoDB;

CREATE TABLE Tipi(
   Civab    VARCHAR(5) PRIMARY KEY,
   Descrizione VARCHAR(50)
) ENGINE=InnoDB;

CREATE TABLE Ospedali(
   idOspedale   INT PRIMARY KEY,
   Nome     VARCHAR(50),
   Citta    VARCHAR(50),
   Indirizzo VARCHAR(50),
   Telefono  VARCHAR(15)
```

```
) ENGINE=InnoDB;

CREATE TABLE Persone(
   idPersona INT PRIMARY KEY,
   Nome      VARCHAR(20),
   Cognome   VARCHAR(20),
   DataNascita DATE,
   Telefono  VARCHAR(15)
) ENGINE=InnoDB;

CREATE TABLE CompatibileCon(
   idTest      INT,
   Civab    VARCHAR(5),
  PRIMARY KEY (idTest,Civab),
  FOREIGN KEY (idTest) REFERENCES Test(idTest)
        ON DELETE NO ACTION,
  FOREIGN KEY (Civab) REFERENCES Tipi(Civab)
        ON DELETE NO ACTION
) ENGINE=InnoDB;

CREATE TABLE Apparecchiature(
   serialeApparecchiatura  VARCHAR(50) PRIMARY KEY,
   Marca VARCHAR(50),
   Modello VARCHAR(50),
   Codice  VARCHAR(50),
   Civab VARCHAR(5) NOT NULL,
  FOREIGN KEY (Civab) REFERENCES Tipi(Civab)
        ON DELETE NO ACTION
) ENGINE=InnoDB;

CREATE TABLE Tecnici(
   idPersona INT PRIMARY KEY,
   Azienda   VARCHAR(50),
   Qualifica VARCHAR(50),
  FOREIGN KEY (idPersona) REFERENCES Persone(idPersona)
        ON DELETE NO ACTION
) ENGINE=InnoDB;

CREATE TABLE Referenti(
   idPersona INT PRIMARY KEY,
   Ruolo   VARCHAR(50),
  FOREIGN KEY (idPersona) REFERENCES Persone(idPersona)
        ON DELETE NO ACTION
) ENGINE=InnoDB;

CREATE TABLE RapportiTecnici(
   idRapporto  INT PRIMARY KEY,
   Ubicazione  VARCHAR(50),
   Data     DATE,
   Tipologia ENUM('Generico','Specifico'),
   Note     VARCHAR(120),
   idTecnico INT NOT NULL,
   idReferente INT NOT NULL,
   idOspedale  INT NOT NULL,
   serialeApparecchiatura  VARCHAR(50),
  FOREIGN KEY (idTecnico) REFERENCES Tecnici(idPersona)
        ON DELETE NO ACTION,
  FOREIGN KEY (idReferente) REFERENCES Referenti(idPersona)
        ON DELETE NO ACTION,
  FOREIGN KEY (idOspedale) REFERENCES Ospedali(idOspedale)
        ON DELETE NO ACTION,
  FOREIGN KEY (serialeApparecchiatura) REFERENCES
```

```
                Apparecchiature(serialeApparecchiatura)
        ON DELETE CASCADE
) ENGINE=InnoDB;

CREATE TABLE Contiene(
   idRapporto  INT,
   idTest      INT,
   Esito    ENUM('SI','NO','N.A.'),
   PRIMARY KEY (idRapporto,idTest),
   FOREIGN KEY (idRapporto) REFERENCES
            RapportiTecnici(idRapporto)
        ON DELETE NO ACTION,
   FOREIGN KEY (idTest) REFERENCES Test(idTest)
        ON DELETE NO ACTION
) ENGINE=InnoDB;

DELIMITER $$
CREATE TRIGGER CheckApp
BEFORE INSERT ON Apparecchiature
FOR EACH ROW
BEGIN
     IF
     (new.serialeApparecchiatura IN
     (SELECT serialeApparecchiatura FROM Apparecchiature))
     THEN
     DELETE FROM ERROR ;
END IF ;
END ;

CREATE TRIGGER CheckNote
BEFORE INSERT ON RapportiTecnici
FOR EACH ROW
BEGIN
     IF (new.Note='') THEN
     SET new.Note="Nessuna nota inserita." ;
END IF ;
END ;

CREATE FUNCTION calcoloEta (nascita DATETIME)
RETURNS INT
BEGIN
     DECLARE eta INT ;
     DECLARE oggi DATETIME ;
     SET oggi = CURDATE() ;
     SET eta = YEAR(oggi) - YEAR(nascita) -
            (DATE_FORMAT(oggi, '%m%d') < DATE_FORMAT(nascita, '%m%d')) ;
     RETURN(eta) ;
END ;

CREATE FUNCTION calcoloSpesa (nomeOsp VARCHAR(50), prezzo DOUBLE)
RETURNS INT
BEGIN
     DECLARE spesa INT ;
     SET spesa = (SELECT COUNT(*)
                  FROM  RapportiTecnici rt
                  NATURAL JOIN Ospedali o
                  WHERE o.Nome=nomeOsp) * prezzo;
     RETURN spesa ;
END $$
DELIMITER ;

SOURCE insertTipi.sql;
```

```
SOURCE insertTest.sql;
SOURCE insertCompatibileCon.sql;
SOURCE insertPersone.sql;
SOURCE insertTecnici.sql;
SOURCE insertReferenti.sql;
SOURCE insertOspedali.sql;
SOURCE insertRapportiTecnici.sql;
SOURCE insertContiene.sql;
```

5 Query

1. Trovare Nome, Cognome, Ruolo dei referenti con età superiore a 30 anni per i quali non esistono rapporti tecnici riguardanti ospedali di Parma.

```
SELECT p.Nome,p.Cognome,r.Ruolo
FROM Persone p JOIN Referenti r ON p.idPersona=r.idPersona
WHERE calcoloEta(p.DataNascita)>30 AND
NOT EXISTS
            (SELECT *
             FROM Ospedali o JOIN RapportiTecnici rt
             ON o.idOspedale=rt.idOspedale
             WHERE rt.idReferente=p.idPersona
             AND o.Citta="Parma");
```

Nome	Cognome	Ruolo
Matteo	Stevanella	Caporeparto pediatria
Alessandro	Berton	Medico dirigente del distretto
Gioele	Gurrieri	Dirigente tecnico
Luca	Bozzato	Caporeparto dermatologia
Jacopo	Bazzoli	Caporeparto gastroenterologia
Simone	Luise	Caposala anestesia e rianimazione
Emanuele	Rosteghin	Caposala cardiologia
Federico	Scattolo	Responsabile dermatologia
Davide	Lunardi	Responsabile ematologia
Massimiliano	Andriollo	Responsabile oculistica
Damiano	Pozzan	Responsabile urologia
Maurizio	Sabbadin	Caposala reumatologia

2. Definire una vista Negativi contenente i Rapporti tecnici dove il test "Verifica della presenza e della completezza dei dati di targa." ha avuto esito negativo. Utilizzare la vista Negativi per trovare il tipo delle apparecchiature di marca "Fisiotre".

```
DELIMITER $$
DROP VIEW IF EXISTS Negativi;
CREATE VIEW Negativi AS
SELECT rt.idRapporto,rt.Ubicazione,rt.Data,
       rt.Tipologia,rt.Note,rt.idTecnico,
       rt.idReferente,rt.idOspedale,rt.serialeApparecchiatura
FROM Test t NATURAL JOIN Contiene c
```

```
      NATURAL JOIN RapportiTecnici rt
WHERE t.Descrizione="Verifica della presenza e della
completezza dei dati di targa."
       AND c.Esito="NO";

SELECT tp.Civab,tp.Descrizione
FROM Negativi NATURAL JOIN Apparecchiature a NATURAL JOIN Tipi tp
WHERE a.Marca="Fisiotre" $$
DELIMITER ;
```

Civab	Descrizione
CEM	CICLO PER USI FISIOTERAPICI E/O DIAGNOSTICI

3. Nome e Cognome di Referente e Tecnico per Rapporti effettuati prima del 7 dicembre 1999 e tali che tutte le apparecchiature coinvolte sono Autoclavi.

```
SELECT p.Nome AS NomeReferente,p.Cognome AS CognomeReferente,
       p1.Nome AS NomeTecnico,p1.Cognome AS CognomeTecnico
FROM RapportiTecnici rt JOIN Persone p
      ON rt.idReferente=p.idPersona
      JOIN Persone p1 ON rt.idTecnico=p1.idPersona
WHERE rt.Data<'1999-12-07' AND
NOT EXISTS
   (SELECT *
    FROM Apparecchiature a
    WHERE rt.serialeApparecchiatura=a.serialeApparecchiatura
    AND a.Civab IN
      (SELECT tp.Civab
       FROM Tipi tp
       WHERE tp.Descrizione!="AUTOCLAVE"));
```

NomeReferente	CognomeReferente	NomeTecnico	CognomeTecnico
Omar	Ballarin	Stefano	Morgagni

4. Per ogni ospedale di Milano fornirne il nome e visualizzarne il numero dei rapporti tecnici effettuati da un tecnico con qualifica Ingegnere.

```
SELECT o.Nome, COUNT(*) AS ConteggioRapportiTecnici
FROM Ospedali o NATURAL JOIN RapportiTecnici rt
      NATURAL JOIN Tecnici t
WHERE o.Citta="Milano" AND t.Qualifica LIKE 'Ingegnere%'
GROUP BY o.Nome
ORDER BY o.Nome;
```

Nome	ConteggioRapportiTecnici
Ospedale Niguarda	30
San Raffaele	120

5. Visualizzare tutti i test compatibili con l'apparecchiatura di seriale FDHAJ-245867TG3 tali che ognuno fa parte di un rapporto tecnico di tipologia Generico ed è relativo ad un ospedale romano.

```
SELECT t.Descrizione
FROM Apparecchiature a NATURAL JOIN Tipi tp JOIN CompatibileCon c
     ON c.Civab=tp.Civab JOIN Test t ON t.idTest=c.idTest
WHERE serialeApparecchiatura="FDHAJ-245867TG3" AND
NOT EXISTS (SELECT *
            FROM Contiene ct NATURAL JOIN RapportiTecnici rt
            NATURAL JOIN Ospedali o
            WHERE ct.idTest=t.idTest
            AND rt.Tipologia!="Generico"
            AND o.Citta!="Roma");
```

```
+-----------------------------------------------------+
| Descrizione                                         |
+-----------------------------------------------------+
| Verificare i sistemi di apertura, chiusura e blocco. |
+-----------------------------------------------------+
```

6. Restituire le spese di manutenzione per l'ospedale San Raffaele utilizzando la Funzione 2, la spesa per ogni rapporto tecnico è assunta come standard e specificata anch'essa come parametro.

```
DELIMITER $$
DROP VIEW IF EXISTS SpeseOspedale;
CREATE VIEW SpeseOspedale AS
SELECT calcoloSpesa ("San Raffaele",150);

SELECT * FROM SpeseOspedale $$
DELIMITER ;
```

```
+-----------------------------------+
| calcoloSpesa ("San Raffaele",150) |
+-----------------------------------+
|                              1800 |
+-----------------------------------+
```

6 Trigger e Funzioni

1. Tramite il seguente trigger viene impedito all'utente di inserire una nuova apparecchiatura nel database durante la compilazione di un Rapporto se essa è già presente.

```
DELIMITER $$
CREATE TRIGGER CheckApp
BEFORE INSERT ON Apparecchiature
FOR EACH ROW
BEGIN
      IF
      (new.serialeApparecchiatura IN
      (SELECT serialeApparecchiatura FROM Apparecchiature))
      THEN
      DELETE FROM ERROR ;
END IF ;
END $$
DELIMITER ;
```

2. Trigger che controlla il campo Note all'inserimento di un nuovo Rapporto Tecnico e se esso è vuoto lo setta a "Nessuna nota inserita."

```
DELIMITER $$
CREATE TRIGGER CheckNote
BEFORE INSERT ON RapportiTecnici
FOR EACH ROW
BEGIN
      IF (new.Note='')
      THEN
      SET new.Note="Nessuna nota inserita." ;
END IF ;
END $$
DELIMITER ;
```

3. Funzione che, ricevuto come parametro la data di nascita di una persona, ne calcola l'età.

```
DELIMITER $$
CREATE FUNCTION calcoloEta (nascita DATETIME)
RETURNS INT
```

```
BEGIN
    DECLARE eta INT ;
    DECLARE oggi DATETIME ;
    SET oggi = CURDATE() ;
    SET eta = YEAR(oggi) - YEAR(nascita) -
              (DATE_FORMAT(oggi, '%m%d') <
               DATE_FORMAT(nascita, '%m%d')) ;
    RETURN(eta) ;
END $$
DELIMITER ;

/* L'espressione (DATE_FORMAT(data1, '%m%d') <
DATE_FORMAT(data2, '%m%d')) è vera se data1 è
"precedente nell'anno" a data2 e poichè in mysql true = 1
e false = 0 l'aggiustamento (cioè sottrarre 1 anno all'età
se la persona deve ancora compiere gli anni quest'anno)
consiste nel sottrarre il risultato di quest'espressione. */
```

4. Funzione che calcola la spesa per l'ospedale scelto in base al numero di rapporti tecnici effettuati(a prezzo standard passato come parametro).

```
DELIMITER $$
CREATE FUNCTION calcoloSpesa (nomeOsp VARCHAR(50), prezzo DOUBLE)
RETURNS INT
BEGIN
    DECLARE spesa INT ;
    SET spesa = (SELECT COUNT(*)
                 FROM RapportiTecnici rt
                 NATURAL JOIN Ospedali o
                 WHERE o.Nome=nomeOsp) * prezzo;
    RETURN spesa ;
END $$
DELIMITER ;
```

7 Interfaccia web

L'interfaccia grafica si presenta con uno stile semplice composto da un menu laterale, per l'accesso alle pagine del sito, e un corpo centrale per la visualizzazione della pagina corrente. Per ogni tabella significativa è stata realizzata una pagina php che ne permette la consultazione dei contenuti. Nella visualizzazione dei Rapporti Tecnici sono stati inseriti dei collegamenti ipertestuali in corrispondenza di alcuni campi per visualizzarne i dati completi. Sono state previste delle pagine che permettono la ricerca all'interno di alcune tabelle ad esempio Rapporti Tecnici, Test, Apparecchiature ecc. Per Ospedali, Referenti e Tecnici sono state rese possibili la modifica, l'eliminazione e l'inserimento tramite interfaccia web. I risultati delle query del punto 5 vengono visualizzati in rispettive pagine. Ogni pagina del sito è accessibile solo previa autenticazione tramite form di login dedicato. Per i nuovi utenti c'è la possibilità di registrazione. Conseguentemente al login viene avviata una sessione che dura fino al logout o alla chiusura del browser.

7.1 Librerie

Viene utilizzato un file *library.php* contenente diverse funzioni che implementano la realizzazione delle pagine html e la visualizzazione dei dati, la connessione al database e le funzionalità di autenticazione. Inoltre esso richiama due file esterni *style.css* e *script.js*. Il file *style.css* contiene principalmente alcune funzionalità estetiche riguardanti i link. Il file *script.js* è utilizzato nelle pagine di ricerca, in cui permette la scelta dell'attributo su cui fare la ricerca, inserimento e modifica di record, dove controlla che non vengano lasciati campi vuoti.