Basi di Dati e Conoscenza

Progetto A.A. 2018/2019

PRENOTAZIONE ESAMI ASL

0245513

Danilo Dell’Orco

Indice

1. Descrizione del Minimondo

Si vuole realizzare il sistema informativo di gestione di un sistema di prenotazioni di esami medici all’interno di una Azienda Sanitaria Locale (ASL), tenendo conto delle seguenti informazioni.

Ciascun paziente è identificato da un codice di tessera sanitaria ed è caratterizzato da un nome, un cognome, la data ed il luogo di nascita, un indirizzo di residenza ed un insieme arbitrario di recapiti (e-mail, telefono, cellulare). La gestione dei pazienti è in capo al personale del CUP, che può gestire nella sua interezza l’anagrafica e le prenotazioni degli esami. In fase di prenotazione, è possibile prenotare con un unico codice di prenotazione un numero arbitrario di esami.

Gli esami medici che possono essere eseguiti sono identificati da un codice numerico e sono caratterizzati dalla descrizione di esame medico (ad esempio Radiografia, ecc.). L’insieme degli esami disponibili presso la ASL sono gestiti dagli amministratori del sistema. Ciascun esame è associato ad un insieme di valori numerici, riportanti i risultati dei parametri legati allo specifico esame. Inoltre, per ciascun esame è possibile inserire da parte del personale medico una diagnosi testuale.

Gli ospedali della ASL sono identificati da un codice numerico e sono caratterizzati da un nome, un indirizzo e dal nome di un responsabile. Anche la gestione degli ospedali è in capo all’amministrazione.

I laboratori che eseguono gli esami sono identificati da un codice univoco all’interno di un ospedale della ASL e sono caratterizzati dal nome del laboratorio, dal piano di ubicazione e dal numero di stanza. Anche per i laboratori è prevista la designazione di un responsabile.

Per ogni prenotazione di un esame da parte di un paziente si vuole memorizzare la data e l’ora dell’esame, il laboratorio presso cui è eseguito, il costo del ticket e se tale esame è prescritto con urgenza. Si tenga presente che ogni paziente può effettuare più prenotazioni dello stesso esame in date diverse. Si noti inoltre che lo stesso esame non può essere ripetuto nello stesso giorno dallo stesso paziente.

Ogni ospedale è suddiviso in reparti identificati da un codice numerico univoco all’interno dell’ospedale di appartenenza e caratterizzati dal nome del reparto e da un numero di telefono. Il personale del reparto è identificato attraverso il codice fiscale; sono noti inoltre il

nome, il cognome e l’indirizzo di domicilio. Tra il personale, nel caso dei medici primari del reparto è noto l’elenco delle specializzazioni, mentre per il personale volontario è noto il nome dell’associazione di appartenenza, se disponibile. I responsabili degli ospedali e dei laboratori vanno individuati all’interno del personale medico, che può essere gestito unicamente dall’amministrazione.

Per motivi di storicizzazione, gli amministratori possono generare dei report che mostrano ciascun membro del personale quanti esami (e quali esami) ha svolto, su base mensile e/o annuale. Il personale del CUP, altresì, ha la possibilità di generare dei report che riportano i risultati di un insieme di esami associati ad una prenotazione, e/o mostrare lo storico di tutti gli esami svolti da un determinato paziente dalla sua registrazione nel sistema.

2. Analisi dei Requisiti

Identificazione dei termini ambigui e correzioni possibili

Linea	Termine	Nuovo termine	Motivo correzione
8, 9	Esami	Esami Reali	Ho utilizzato l’instance of per differenziare i concetti di Esami disponibili (ovvero la lista di esami medici che è possibile effettuare) ed Esame Reale, che rappresenta lo svolgimento dell’esame su un paziente, con i relativi risultati. In questo caso un paziente prenota un “Esame Reale”, con relativa data ed ora di svolgimento.
10	Esami Medici	Esami Disponibili	Gli esami medici cui si fa riferimento, sono gli esami che si possono effettuare; nella linea 12 si fa riferimento agli esami disponibili, che rappresentano lo stesso concetto di “Esami Medici” nella linea 10, per cui elimino il sinonimo dando ad entrambi lo stesso nome.
11	descrizione di esame medico	nome dell’esame	Il termine “descrizione di esame medico” rappresenta il nome dell’esame (Radiografia etc..), per cui riporto esplicitamente questo concetto.
14	Esame	Esame Reale	In questo caso non è chiaro se “Esame” si riferisca ad un “Esame Disponibile” o ad un “Esame Reale” Ho quindi esplicitato il riferimento di “Esame” verso un “Esame Reale”, poiché i parametri numerici e l’eventuale diagnosi testuale, sono relative ad un preciso esame, svolto su un certo paziente a seguito di una prenotazione.
14	Personale medico	Medico	Un medico può fornire una diagnosi. Il termine “personale medico” che viene utilizzato è ambiguo e poco chiaro, mentre “medico” esprime esplicitamente il concetto.
22, 23	Esame	Esame Reale	Un paziente prenota un “Esame Reale”, che si svolgerà in una certa data ad una certa ora. Ho esplicitato il riferimento verso “Esame Reale”

25	Esame	Esame Disponibile	Un paziente può prenotare uno stesso esame in due date diverse. In questo caso esame si riferisce ad un “Esame Disponibile”. Ad esempio, un paziente può prenotare due Radiografie per due date diverse. Così facendo il paziente prenota due “Esami Reali” distinti (Radiografia del 15 maggio e Radiografia del 3 Giugno), aventi entrambi lo stesso esame disponibile (Radiografia)
29	Personale del reparto	Medico	Con “Personale del reparto” viene espresso lo stesso concetto di Medico già utilizzato precedentemente. In questa linea, “personale del reparto” indica un medico che lavora in un certo reparto, per cui i due termini sono sinonimi.
30	Tra il Personale	Tra i medici	Il personale medico rappresenta l’insieme dei medici. Rendo quindi esplicito il riferimento verso “Medico”, ed elimino una possibile ambiguità
33	All’interno del Personale Medico	Tra i medici	Nella consegna non è ben chiaro il concetto di “personale medico”, per cui esprimo chiaramente il concetto riferendomi esplicitamente a “medici”
36	Membro del personale	Medico	Un membro del personale altro non è che un medico, per cui utilizzo lo stesso termine utilizzato in precedenza ed elimino il sinonimo.
36	quanti esami	quanti esami reali	Con “quanti esami”, si vuole conoscere quante visite sui pazienti ha effettuato un medico, per cui si riferisce agli esami reali.
36	quali esami	quali esami disponibili	Con “quali esami”, si vuole conoscere il tipo di esame che ha svolto un medico, tra tutti quelli possibili. Quindi si riferisce agli esami disponibili
38,39	Esami	Esami reali	I risultati di un insieme di esami e lo storico di tutti gli esami svolti da un determinato paziente si riferiscono agli esami reali, quindi elimino l’ambiguità

Specifica disambiguata

Si vuole realizzare il sistema informativo di gestione di un sistema di prenotazioni di esami medici all’interno di una Azienda Sanitaria Locale (ASL), tenendo conto delle seguenti informazioni.

Ciascun paziente è identificato da un codice di tessera sanitaria ed è caratterizzato da un nome, un cognome, la data ed il luogo di nascita, un indirizzo di residenza ed un insieme arbitrario di recapiti (e-mail, telefono, cellulare). La gestione dei pazienti è in capo al personale del CUP, che può gestire

nella sua interezza l’anagrafica e le prenotazioni degli esami reali. In fase di prenotazione, è possibile prenotare con un unico codice di prenotazione un numero arbitrario di esami reali.

Gli esami disponibili che possono essere eseguiti sono identificati da un codice numerico e sono caratterizzati dal nome dell’esame (ad esempio Radiografia, ecc.). L’insieme degli esami disponibili presso la ASL sono gestiti dagli amministratori del sistema. Ciascun esame reale è associato ad un insieme di valori numerici, riportanti i risultati dei parametri legati allo specifico esame. Inoltre, per ciascun esame reale è possibile inserire da parte di un medico una diagnosi testuale.

Gli ospedali della ASL sono identificati da un codice numerico e sono caratterizzati da un nome, un indirizzo e dal nome di un responsabile. Anche la gestione degli ospedali è in capo all’amministrazione.

I laboratori che eseguono gli esami reali sono identificati da un codice univoco all’interno di un ospedale della ASL e sono caratterizzati dal nome del laboratorio, dal piano di ubicazione e dal numero di stanza. Anche per i laboratori è prevista la designazione di un responsabile.

Per ogni prenotazione di un esame reale da parte di un paziente si vuole memorizzare la data e l’ora dell’esame reale, il laboratorio presso cui è eseguito, il costo del ticket e se tale esame reale è prescritto con urgenza. Si tenga presente che ogni paziente può effettuare più prenotazioni dello stesso esame disponibile, in date diverse. Si noti inoltre che lo stesso esame disponibile non può essere ripetuto nello stesso giorno dallo stesso paziente.

Ogni ospedale è suddiviso in reparti identificati da un codice numerico univoco all’interno dell’ospedale di appartenenza e caratterizzati dal nome del reparto e da un numero di telefono. Un medico è identificato attraverso il codice fiscale; sono noti inoltre il nome, il cognome e l’indirizzo di domicilio. Tra i medici, nel caso dei medici primari del reparto è noto l’elenco delle specializzazioni, mentre per il personale volontario è noto il nome dell’associazione di appartenenza, se disponibile. I responsabili degli ospedali e dei laboratori vanno individuati tra i medici, che possono essere gestiti unicamente dall’amministrazione.

Per motivi di storicizzazione, gli amministratori possono generare dei report che mostrano ciascun medico quanti esami reali (e quali esami disponibili) ha svolto, su base mensile e/o annuale. Il personale del CUP, altresì, ha la possibilità di generare dei report che riportano i risultati di un insieme di esami reali associati ad una prenotazione, e/o mostrare lo storico di tutti gli esami reali svolti da un determinato paziente dalla sua registrazione nel sistema.

Glossario dei Termini

Termine	Descrizione	Sinonimi	Collegamenti
Paziente	Una persona che prenota una visita presso la ASL.		Prenotazione
Prenotazione	Prenotazione di un certo numero di esami da parte di un paziente. Identificata da un codice univoco.		Paziente, Esame Reale
Esame Reale	Specifico esame eseguito su un paziente in un certo giorno. Caratterizzato dai parametri riscontrati ed eventualmente da una diagnosi di un medico	Esame	Prenotazione, Esame Disponibile, Laboratorio, Personale Medico
Esame Disponibile	Un esame che è possibile effettuare, identificato da un codice numerico, e da un nome.	Esame medico, esame	Esame reale
Laboratorio	Un laboratorio si trova all’interno di un ospedale, ed è il luogo dove viene svolto un esame. Ogni laboratorio ha un responsabile.		Esame reale, ospedale, personale medico
Medico	Lavora all’interno di un ospedale. Ogni medico lavora in un reparto. Ogni medico può essere responsabile di un laboratorio e/o di un ospedale. Un medico può effettuare un esame reale, ed inserire una diagnosi	Personale, personale del reparto, Personale medico, Membro del personale	Laboratorio, ospedale, reparto, esame reale

Primario	Specializzazione di medico. È di interesse conoscere la lista di specializzazioni		Medico
Volontario	Specializzazione di Medico, è di interesse conoscere il nome dell’associazione (se possibile)		Medico
Ospedale	Un ospedale della ASL		Laboratorio, personale medico, reparto
Reparto	Un reparto presente in un ospedale		Ospedale, personale medico
Personale del CUP	Persone che lavorano per il CUP. Possono gestire l’anagrafica, le prenotazioni, generare dei report sugli esami associati ad un paziente o ad una prenotazione		Anagrafica, prenotazione, esame, paziente
Amministratore	Utente che può gestire gli esami reali e gli ospedali. Possono anche generare dei report che mostrano quanti e quali esami ha effettuato un medico su base mensile e/o annuale.	Amministrazione	Esame reale, esame disponibile, ospedale, personale medico

Raggruppamento dei requisiti in insiemi omogenei

Frasi di carattere generale
Si vuole realizzare il sistema informativo di gestione di un sistema di prenotazioni di esami medici all’interno di una Azienda Sanitaria Locale (ASL)
Frasi relative ai pazienti
Ciascun paziente è identificato da un codice di tessera sanitaria ed è caratterizzato da un nome, un cognome, la data ed il luogo di nascita, un indirizzo di residenza ed un insieme arbitrario di recapiti (e-mail, telefono, cellulare).
Frasi relative agli esami reali
Ciascun esame reale è associato ad un insieme di valori numerici, riportanti i risultati dei parametri legati allo specifico esame. Per ogni prenotazione di un esame reale da parte di un paziente si vuole memorizzare la data e l’ora dell’esame reale, il laboratorio presso cui è eseguito, il costo del ticket e se tale esame reale è prescritto con urgenza. Inoltre, per ciascun esame reale è possibile inserire da parte di un medico una diagnosi testuale.
Frasi relative agli esami disponibili
Gli esami disponibili che possono essere eseguiti sono identificati da un codice numerico e sono caratterizzati dal nome dell’esame (ad esempio Radiografia, ecc.). Si noti inoltre che lo stesso esame disponibile non può essere ripetuto nello stesso giorno dallo stesso paziente.
Frasi relative alle prenotazioni
In fase di prenotazione, è possibile prenotare con un unico codice di prenotazione un numero arbitrario di esami reali. Si tenga presente che ogni paziente può effettuare più prenotazioni dello stesso esame disponibile, in date diverse. Si noti inoltre che lo stesso esame disponibile non può essere prenotato nello stesso giorno dallo stesso paziente.
Frasi relative ai laboratori
I laboratori che eseguono gli esami reali sono identificati da un codice univoco all’interno di un ospedale della ASL e sono caratterizzati dal nome del laboratorio, dal piano di ubicazione e dal numero di stanza. Anche per i laboratori è prevista la designazione di un responsabile.
Frasi relative ai medici

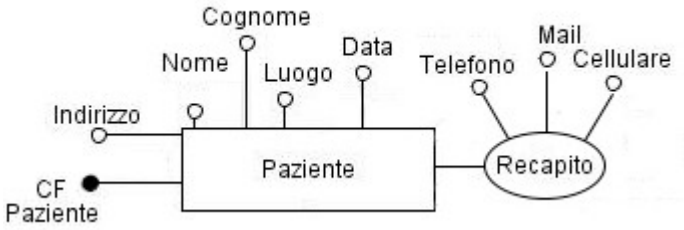
Un medico è identificato attraverso il codice fiscale; sono noti inoltre il nome, il cognome e l’indirizzo di domicilio. Tra i medici, nel caso dei medici primari del reparto è noto l’elenco delle specializzazioni, mentre per il personale volontario è noto il nome dell’associazione di appartenenza, se disponibile. I responsabili degli ospedali e dei laboratori vanno individuati tra i medici, che possono essere gestiti unicamente dall’amministrazione.
Frasi relative agli ospedali
Gli ospedali della ASL sono identificati da un codice numerico e sono caratterizzati da un nome, un indirizzo e dal nome di un responsabile.
Frasi relative ai reparti
Ogni ospedale è suddiviso in reparti identificati da un codice numerico univoco all’interno dell’ospedale di appartenenza e caratterizzati dal nome del reparto e da un numero di telefono.
Frasi relative agli amministratori
L’insieme degli esami disponibili presso la ASL sono gestiti dagli amministratori del sistema. Anche la gestione degli ospedali è in capo all’amministrazione. I responsabili degli ospedali e dei laboratori vanno individuati tra i medici, che possono essere gestiti unicamente dall’amministrazione. Per motivi di storicizzazione, gli amministratori possono generare dei report che mostrano ciascun medico quanti esami reali (e quali esami disponibili) ha svolto, su base mensile e/o annuale
Frasi relative al personale del CUP
La gestione dei pazienti è in capo al personale del CUP, che può gestire nella sua interezza l’anagrafica e le prenotazioni degli esami reali. Il personale del CUP, altresì, ha la possibilità di generare dei report che riportano i risultati di un insieme di esami reali associati ad una prenotazione, e/o mostrare lo storico di tutti gli esami reali svolti da un determinato paziente dalla sua registrazione nel sistema.

3. Progettazione concettuale

Costruzione dello schema E-R

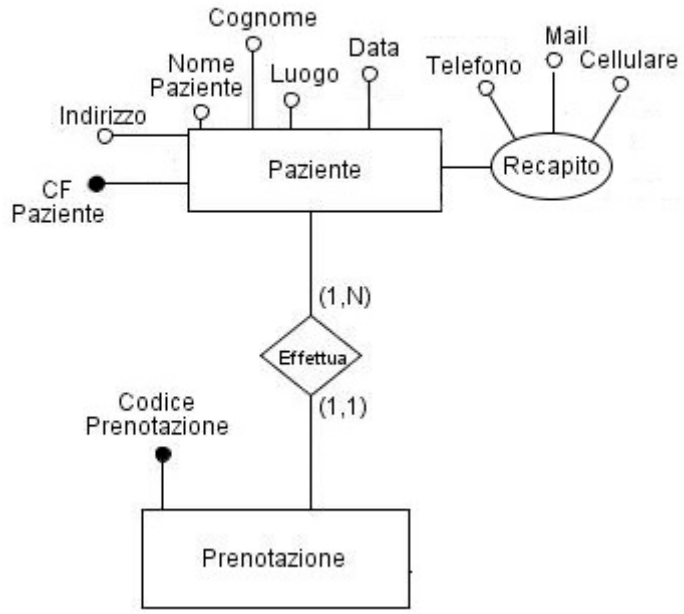
Per la costruzione dello schema E-R ho utilizzato una strategia inside-out, partendo da concetti elementari ed espandendoli progressivamente.

Sono partito dal concetto di "Paziente". Ho creato quindi l'entità "Paziente", identificata dal "Codice di tessera sanitaria", con i relativi attributi.

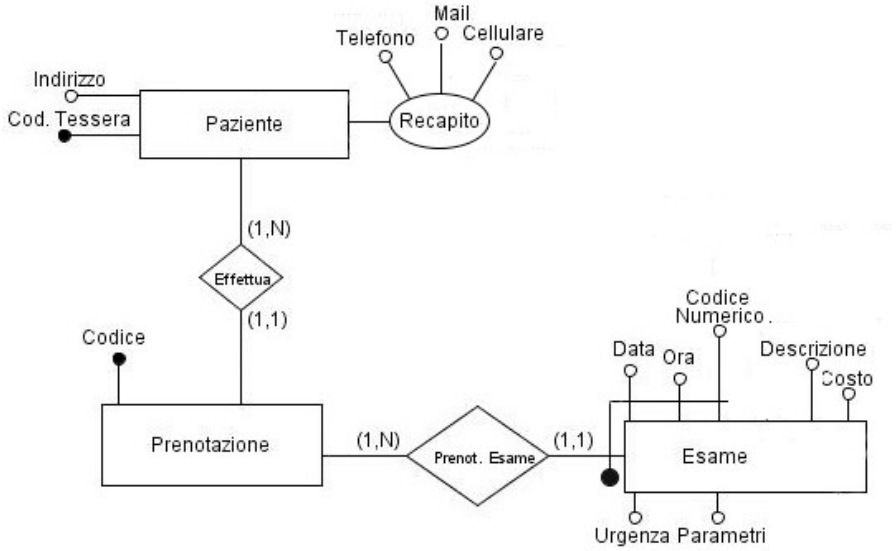


Un paziente prenota uno o più esami presso la ASL.

Ogni prenotazione ha un codice univoco, a cui sono associati uno o più esami prenotati. Quindi creo un'entità "Prenotazione" identificata univocamente dal "Codice", legata con un'associazione uno a molti con "Paziente". Infatti, una prenotazione è relativa ad un solo paziente (due pazienti non possono avere lo stesso codice di prenotazione), e un paziente può effettuare più prenotazioni presso la ASL



Con un unico codice di prenotazione è possibile prenotare un certo numero di esami. Creo quindi l'entità "Esame", identificata dal codice relativo alla prenotazione, la data, l'ora, ed il codice numerico che indica il tipo di esame (Radiografia, TAC…)

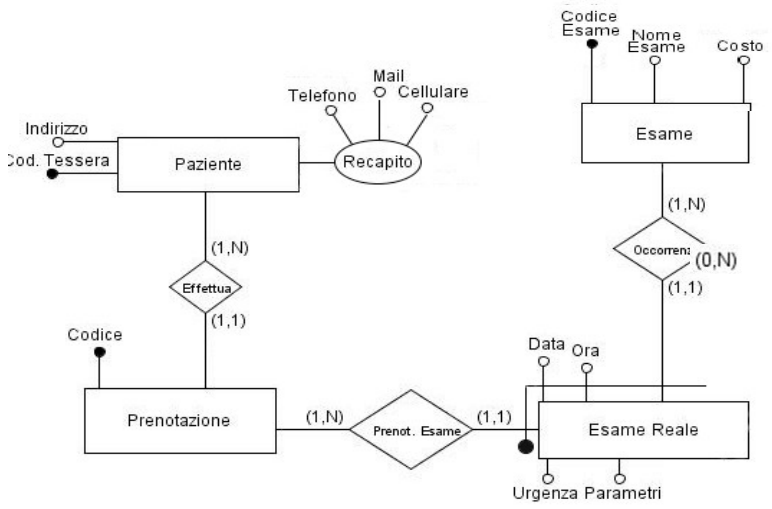


Utilizzo "instance-of" per differenziare un esame che è possibile prenotare, da un esame effettivamente svolto da un paziente.

"Esame" indica un esame, tra quelli disponibili presso la ASL. In questo senso un esame è caratterizzato dal "Codice Numerico" (unico) che ne identifica il tipo, il nome dell'esame medico, ed il costo del ticket.

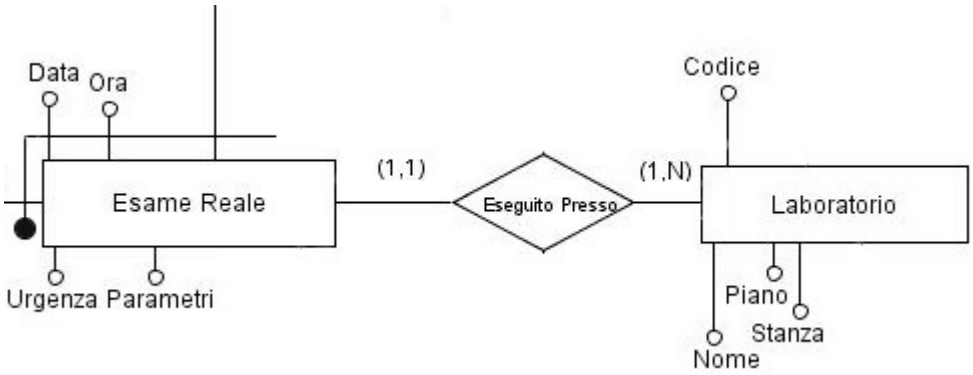
"Esame Reale" indica invece un effettivo esame svolto da un paziente, con relativa diagnosi ed urgenza.
È un'entità debole e viene identificata dal codice di "prenotazione" che, in quanto unico per ogni cliente, associa "Esame Reale" ad una specifica prenotazione di un preciso cliente. Dal codice numerico di "Esame", per specificare quale esame effettua il paziente. Da "data" ed "ora", poiché un paziente potrebbe effettuare lo stesso esame (es. radiografia) in due esami reali differenti.

L'associazione è uno a molti: un esame reale è relativo ad un solo esame astratto. Un esame può essere presente in diversi esami reali. Esame verso esame reale è (0,N) perché potrebbero esserci esami disponibili presenti nel database, ma che non sono associati a nessun esame reale



Un esame reale viene eseguito presso un Laboratorio, caratterizzato da un codice, un nome, il piano, ed il numero di stanza.

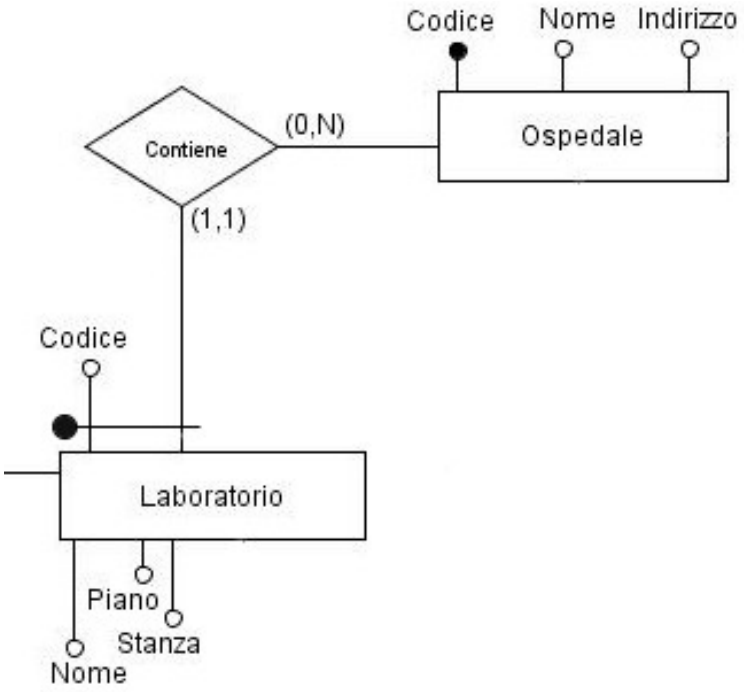
L'associazione è uno a molti poiché un esame viene svolto in un solo laboratorio, ma in un laboratorio vengono svolti diversi esami



Un laboratorio viene identificato all'interno di un ospedale, quindi è un'entità debole con chiavi "codice laboratorio" e identificatore esterno "codice ospedale".

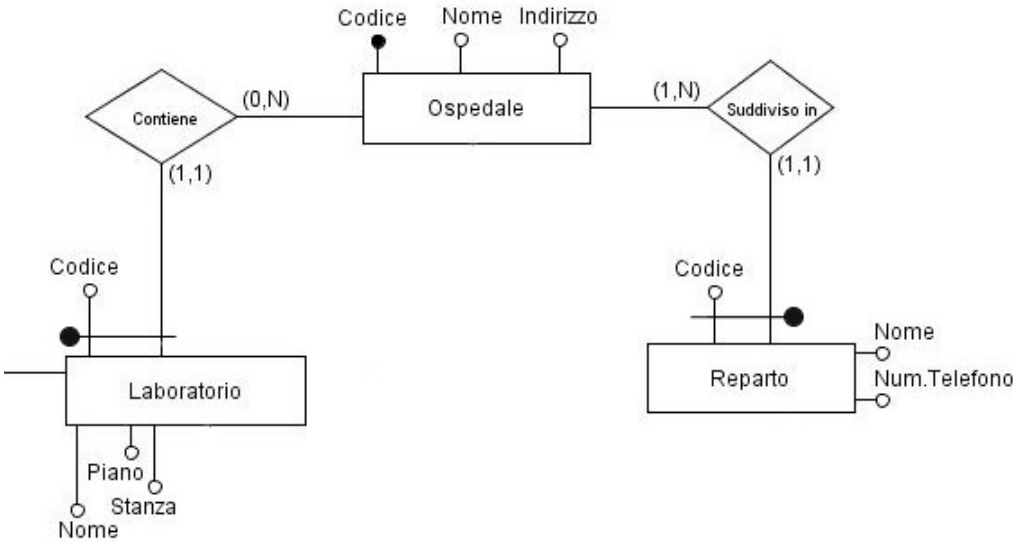
Un ospedale è identificato da un codice univoco, un nome, e un indirizzo.

L'associazione è uno a molti poiché il laboratorio è specifico di un ospedale, mentre l'ospedale può contenere zero o più laboratori.



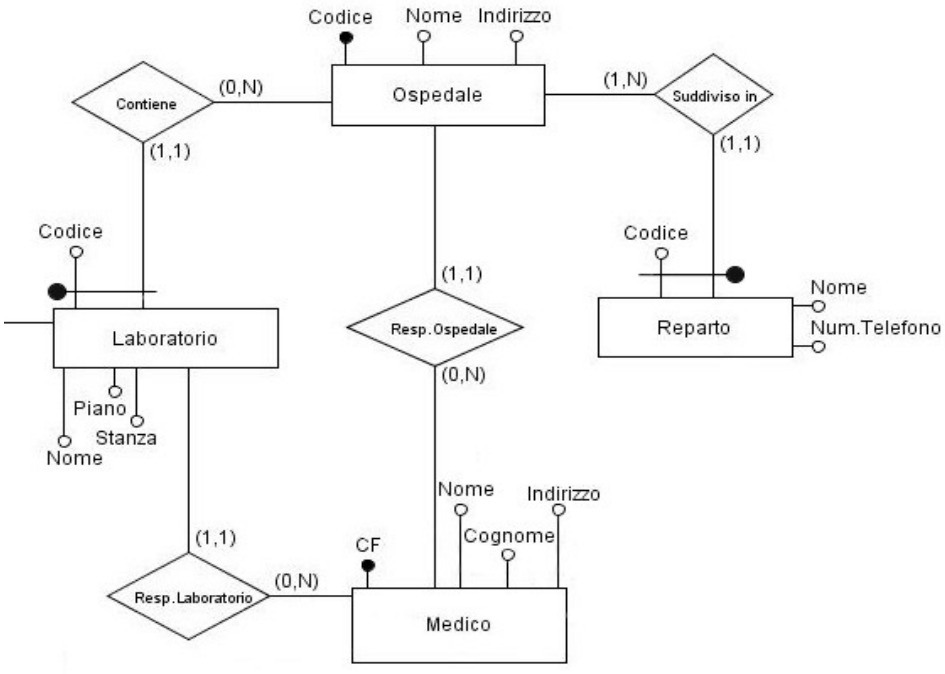
Un ospedale è suddiviso in reparti. "Reparto" è un'entità debole poiché è identificato all'interno di uno specifico ospedale, e ha come chiavi il codice del reparto ed il codice dell'ospedale, che è un identificatore esterno. Il reparto inoltre ha come attributi il nome, ed il numero di telefono.

L'associazione è uno a molti, poiché un reparto è relativo ad un solo ospedale, mentre un ospedale è suddiviso in più reparti



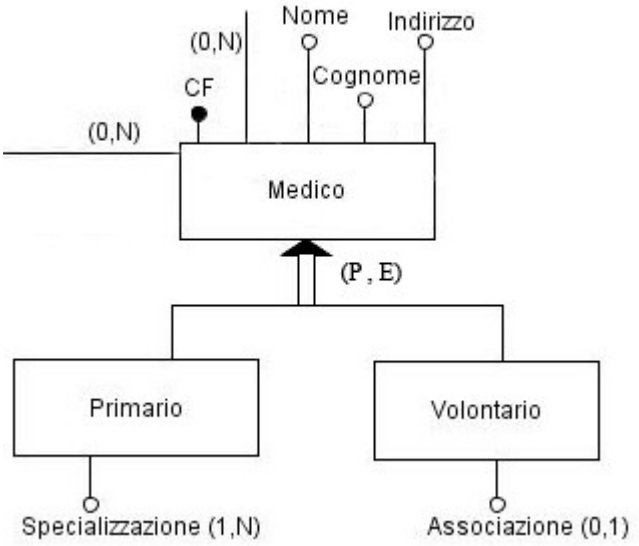
Per i laboratori e gli ospedali va specificato il nome di un responsabile. I responsabili vanno identificati tra i medici, per cui creo l'entità "Medico".

Un medico è caratterizzato dal Codice Fiscale (che lo identifica), dal nome, cognome, indirizzo di residenza. Un medico potrebbe non essere responsabile di nessun laboratorio, oppure essere responsabile di più laboratori; "Medico" verso "Laboratorio" ha quindi cardinalità (0,N). Per un laboratorio è invece necessario specificare uno e un solo responsabile (che è un medico), per cui "Laboratorio" verso "Medico" ha cardinalità (1,1). Il ragionamento è analogo per gli ospedali: "Ospedale" verso "Medico" (1,1) e "Medico" verso "Ospedale" (0,N).

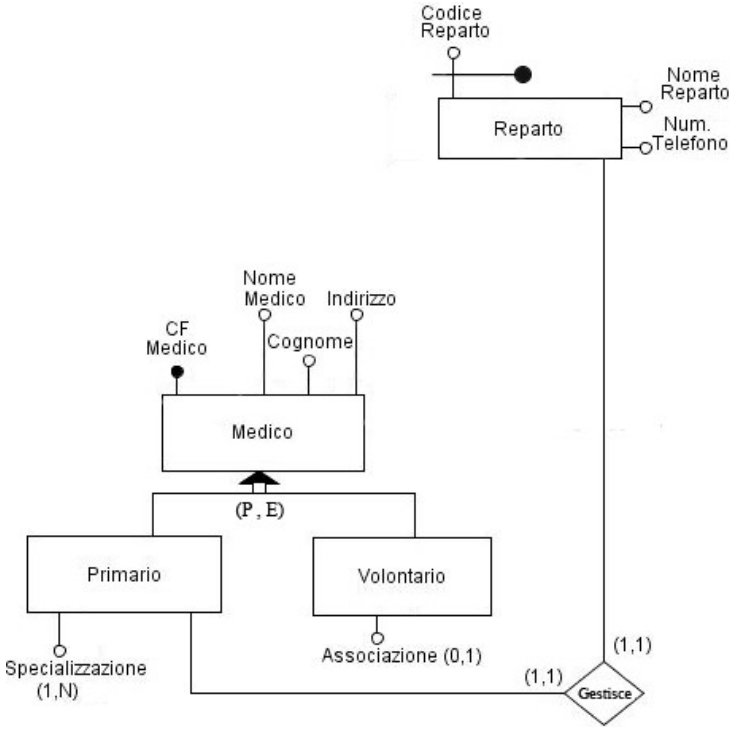


Tra i medici bisogna specificare, nel caso dei medici primari l'elenco delle specializzazioni, mentre per il personale volontario il nome dell'associazione di appartenenza, se disponibile. Creo quindi una generalizzazione con due entità figlie "Primario" e "Volontario". La generalizzazione è parziale (ci sono medici che non sono né primari né volontari) ed esclusiva (un medico primario non può essere un volontario, e viceversa).

Per un primario utilizzo un attributo multivalore "specializzazione" con cardinalità (1,N). Per un volontario utilizzo un attributo opzionale "associazione" con cardinalità (0,1)



Un primario è un medico posto a capo di un reparto ospedaliero, quindi creo un'associazione "gestisce" per associare il primario con il relativo reparto di cui è responsabile. Reparto verso primario ha cardinalità (1,1) poiché un reparto ha un solo primario. Primario verso reparto ha cardinalità (1,1) poiché un medico può essere primario di un solo reparto.

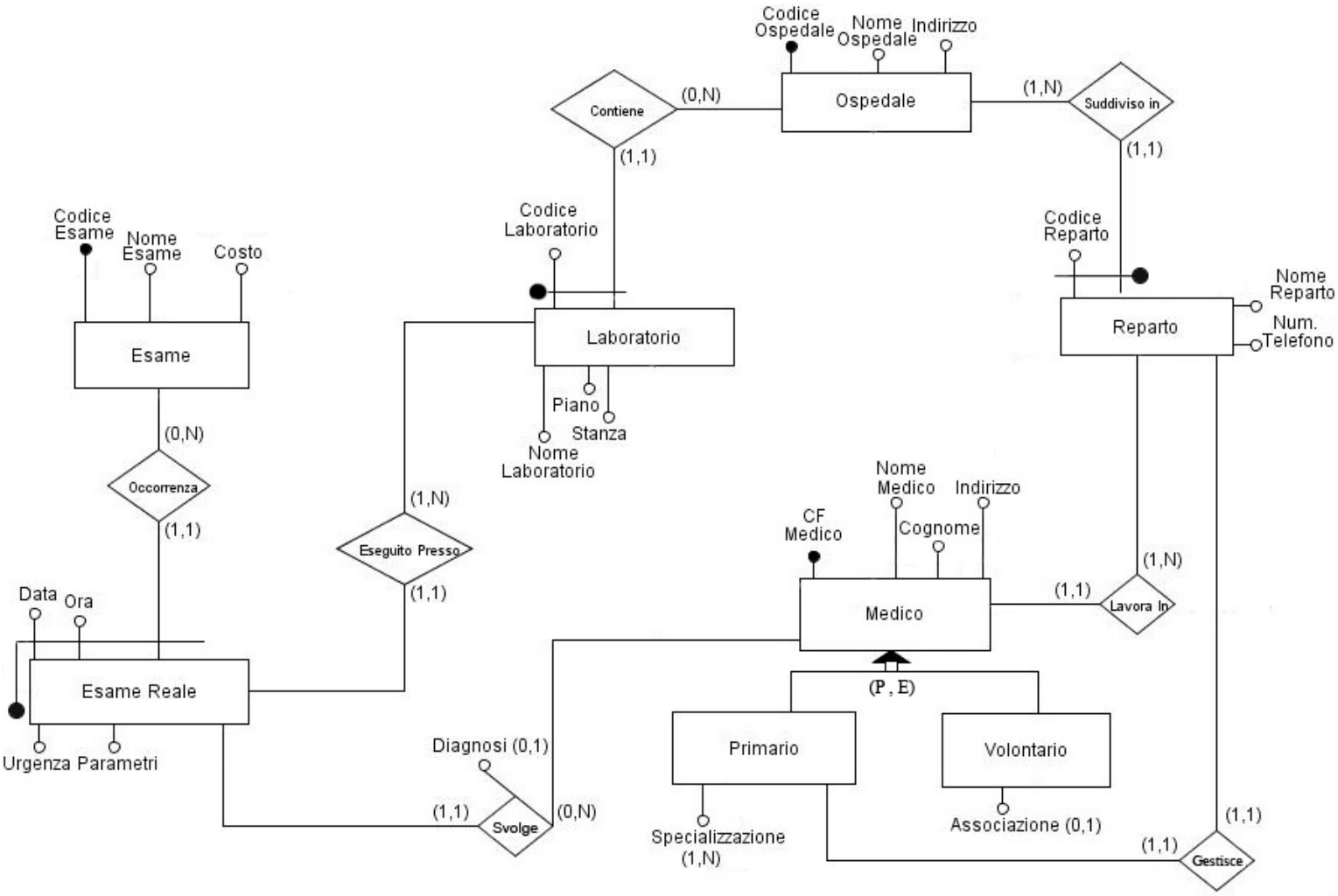


Un medico generico lavora presso uno e un solo reparto. Creo quindi un'associazione "Lavora in" tra "reparto" e "medico". L'associazione è uno a molti poiché in un reparto lavorano più medici, mentre un medico lavora in un solo reparto.

Inoltre, un medico può aggiungere una diagnosi testuale di un esame reale. Poiché è un medico ad effettuare l'esame, e di questo esame può fornire o meno una diagnosi, creo un'associazione "svolge" tra "medico" ed "esame reale".

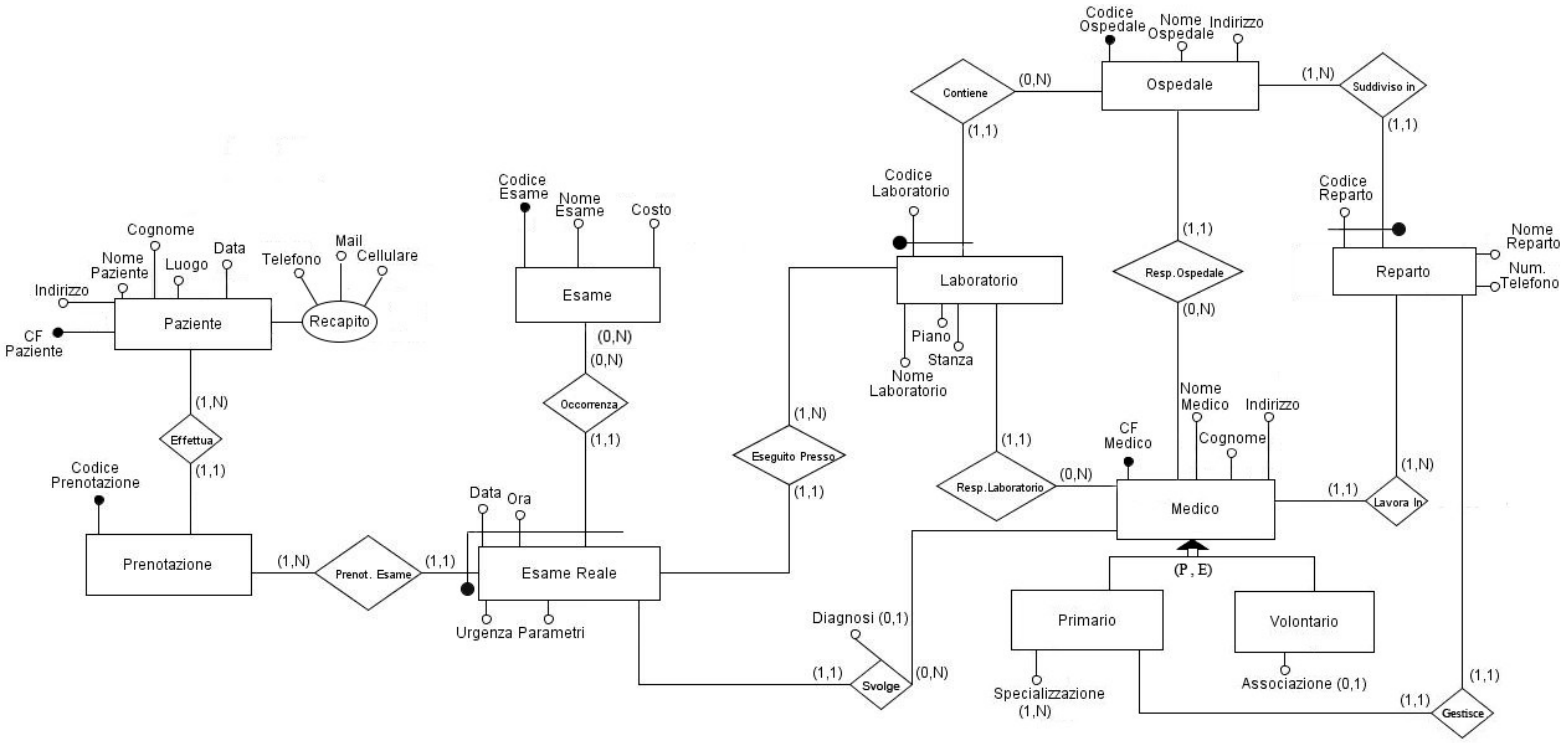
Medico verso esame ha cardinalità (0,N) poiché un medico potrebbe non effettuare esami, o effettuarne più di uno. Esame verso medico ha cardinalità (1,1) poiché un esame reale è necessariamente effettuato da uno e un solo medico.

L'associazione "svolge" ha attributo opzionale (0,1) "diagnosi", in modo tale che nello schema relazionale venga espresso il fatto che sia il medico ad aggiungere (eventualmente) una diagnosi su un esame reale.



Integrazione finale

Nello schema sono presenti molti attributi con nome "Codice" e "Nome", ma relativi ad entità differenti. Per ognuno di questi ho espresso esplicitamente a quale entità si riferisce, per evitare possibili conflitti. Ad esempio, per il codice relativo alle prenotazioni ho indicato "Codice prenotazione", o per il nome dei pazienti e dei medici "Nome paziente" e "Nome medico"



Regole aziendali

Lo stesso esame disponibile NON PUO' essere ripetuto nello stesso giorno dallo stesso paziente.

Un medico NON PUO' effettuare due esami reali contemporaneamente.

In un laboratorio NON POSSONO essere effettuati due esami reali contemporaneamente.

Dizionario dei dati

Entità	Descrizione	Attributi	Identificatori
Paziente	Effettua una prenotazione di uno o più esami presso la ASL	CF paziente, Nome paziente, Cognome, Luogo di nascita, Data di nascita indirizzo, Telefono, Mail, Cellulare	CF paziente
Prenotazione	Prenotazione di uno o più esami.	Codice prenotazione	Codice prenotazione
Esame Reale	Esame, inteso come una visita svolta da un paziente.	Codice prenotazione, Codice Esame, Data, Ora, Urgenza, Parametri	Codice prenotazione, Codice Esame, Data, Ora
Esame	Esame tra quelli che è possibile prenotare presso la ASL	Codice esame, Nome esame, Costo	Codice esame
Ospedale	Un ospedale della ASL	Codice ospedale, Nome ospedale, Indirizzo	Codice ospedale
Laboratorio	Laboratorio di un ospedale, nel quale vengono effettuati degli esami.	Codice laboratorio, Codice ospedale, Nome laboratorio, Piano, Stanza	Codice Laboratorio, Codice Ospedale
Reparto	Reparto di un ospedale, nel quale lavorano uno o più medici	Codice reparto, Codice ospedale, Nome reparto, Numero telefono	Codice reparto, Codice ospedale
Medico	Medico che lavora in un reparto di un ospedale. Può essere responsabile di un laboratorio e/o di un ospedale. Può effettuare degli esami	CF medico, Nome medico, Cognome, indirizzo	CF medico
Effettua	Associa un paziente con la prenotazione che quel cliente ha effettuato.		

Prenotazione Esame	Associa una prenotazione con gli esami reali relativi a quella prenotazione. Serve per identificare un esame reale		
Occorrenza	Relazione utilizzata per l’instance of. Associa un esame reale con il relativo tipo di esame. Serve per identificare un esame reale		
Eseguito presso	Associa un esame reale con il laboratorio presso cui è eseguito		
Contiene	Associa un laboratorio con l’ospedale di appartenenza. Serve per identificare un laboratorio.		
Suddiviso in	Associa un reparto con il relativo ospedale. Serve per identificare un reparto.		
Responsabile laboratorio	Associa un laboratorio con il medico che lo gestisce		
Responsabile ospedale	Associa un ospedale con il medico che lo gestisce		
Lavora in	Associa un medico con il reparto presso cui lavora		
Gestisce	Associa un medico primario con il reparto di cui è a capo		
Svolge	Associa un esame reale con il medico che ha svolto l’esame	Diagnosi (0,1)	

4. Progettazione logica

Volume dei dati

Concetto nello schema	Tipo[1]	Volume atteso
Paziente	E	20 000
Prenotazione	E	200 000
Esame	E	50
Esame Reale	E	600 000
Laboratorio	E	25
Ospedale	E	5
Reparto	E	100
Medico	E	1000
Primario	E	100
Volontario	E	250
Effettua	R	200 000
Prenotazione Esame	R	600 000
Occorrenza	R	600 000
Eseguito Presso	R	600 000
Contiene	R	25
Suddiviso in	R	100
Responsabile Laboratorio	R	25
Responsabile Ospedale	R	5
Lavora in	R	1000
Gestisce	R	100
Svolge	R	600 000

1Indicare con E le entità, con R le relazioni

Tavola delle operazioni

Cod.	Descrizione	Frequenza attesa
1	Aggiungi un paziente	5000 al mese
2	Modifica/Cancella un paziente	1000 al mese
3	Aggiungi prenotazione relativa ad un paziente	15 000 al mese
4	Modifica/Cancella prenotazione	3000 al mese
5	Aggiungi un esame reale	45 000 al mese
6	Modifica/Cancella un esame reale	15 000 al mese
7	Aggiungi un esame	10 al mese
8	Modifica/Cancella un esame	5 al mese
9	Aggiungi un laboratorio	15 al mese
10	Modifica/Cancella un Laboratorio	5 al mese
11	Aggiungi un ospedale	2 al mese
12	Modifica/Cancella un ospedale	4 al mese
13	Aggiungi un reparto	10 al mese
14	Modifica/Cancella un reparto	2 al mese
15	Mostra il numero di esami svolti da un medico	100 al mese
16	Mostra quali esami ha svolto un medico	200 al mese
17	Mostra i risultati degli esami associati ad una prenotazione	10 000 al mese
18	Mostra gli esami reali svolti da un paziente	3500 al mese

Costo delle operazioni

Operazione 1: Aggiungi Paziente

Per aggiungere un paziente effettuo solo una scrittura. Costo = 1(S)*5000/mese = 2*5000 = 10000

Operazione 2: Modifica/Cancella Paziente

Per modificare o cancellare un paziente effettuo prima una lettura, e poi una scrittura per la modifica. Costo = (1(L) + 1(S)) *3000/mese = 3*1000 = 3000

Operazione 3: Aggiungi una prenotazione

Per aggiungere una prenotazione effettuo solo un accesso in scrittura.

Costo = 1(S)*15000/mese = 2*15000 = 30000

Operazione 4: Modifica/Cancella una prenotazione

Per cancellare o modificare una prenotazione effettuo prima una lettura della prenotazione, e poi la modifico effettuando una scrittura. Costo = (1(L) + 1(S))*3000/mese = 3*3000 = 9000

Operazione 5: Aggiungi un esame reale

Per aggiungere un esame devo leggere il codice di prenotazione, l’esame(disponibile), e il medico che effettua la visita. Poi effettuo una scrittura.

Costo = (3(L) + 1(S))*45000/mese = (4+2*1)*45000 = 6*45000 = 225000

Operazione 6: Modifica/Cancella un esame reale

Effettuo un accesso in lettura per leggere l’esame, e poi una scrittura per cancellarlo o modificarlo.

Costo = (1(L) + 1(S))*15000/mese = 3*15000 = 45000

Operazione 7: Aggiungi un esame

Per aggiungere un esame alla lista di esami disponibili effettuo solo una scrittura.

Costo = 1(S)*10/mese = 2*10 = 20

Operazione 8: Modifica/Cancella un esame

Per cancellare un esame effettuo una lettura ed una scrittura.

Costo = (1(L) + 1(S))*5/mese = 3*5 = 15

Operazione 9: Aggiungi un laboratorio

Per aggiungere un laboratorio devo leggere l’ospedale di appartenenza ed il medico responsabile. Poi effettuo una scrittura. Costo = (2(L) + 1(S))*15/mese = 4*15 = 60

Operazione 10: Modifica/Cancella un laboratorio

Per cancellare un laboratorio effettuo una lettura del laboratorio da cancellare, e poi una scrittura. Costo = (1(L) + 1(S))*5/mese = 3*5 = 15

Operazione 11: Aggiungi un ospedale

Per aggiungere un ospedale dobbiamo leggere il medico responsabile dell’ospedale, e poi effettuare una scrittura. Costo = (1(L) + 1(S))*2/mese = 3*2 = 6

Operazione 12: Modifica/Cancella un ospedale

Per modificare o cancellare un ospedale effettuo una lettura per leggere l’ospedale da modificare. Poi effettuo una scrittura. Costo = (1(L) + 1(S))*4/mese = 3*4 = 12

Operazione 13: Aggiungi un reparto

Per aggiungere un reparto devo leggere l'ospedale di appartenenza e poi effettuare una scrittura. Costo = (1(L) + 1(S))*10/mese = 3*10 = 30

Operazione 14: Modifica/Cancella un reparto

Per modificare o cancellare un reparto dobbiamo effettuare una lettura del reparto in questione e poi una scrittura per cancellarlo. Costo = (1(L) + 1(S))*2/mese = 3*2 = 15

Operazione 15: Mostra il numero di esami svolti da un medico

Ho un volume di 600 000 esami reali e di 1000 medici. In media quindi ogni medico svolge 600 esami. Per contare il numero di esami svolti da un medico devo effettuare un conteggio sul numero di occorrenze del medico cercato nella tabella "Esame reale". Effettuo quindi un accesso in lettura per leggere il medico. Poi effettuo 600 accessi in lettura per leggere nella tabella degli esami reali, quelli che sono stati svolti dal medico che stiamo considerando.

Costo = (1(L) + 600(L))*100/mese = 601*100 = 60100

Operazione 16: Mostra quali esami ha svolto un medico

Per mostrare gli esami svolti da un medico devo mostrare a schermo tutte le tuple di "Esami reali" che contengono il medico cercato. Per fare questo effettuo un accesso in lettura per leggere il medico. Poi effettuo 600 accessi in lettura per verificare nella tabella degli esami reali quali sono stati svolti dal medico che stiamo considerando.

Costo = (1(L) + 600(L))*200/mese = 601*200 = 120200

Operazione 17: Mostra i risultati degli esami associati ad una prenotazione.

Ho un volume di 600 000 Esami Reali e di 200 000 prenotazioni. In media quindi ho 3 esami per ogni codice di prenotazione. Effettuo quindi un accesso in lettura per leggere il codice di prenotazione. Poi effettuo 3 accessi in lettura per leggere l'esame relativo al codice di prenotazione che stiamo considerando.

Costo = (1(L) + 3(L))*10000/mese = 4*10000 = 40000

Operazione 18: Mostra tutti gli esami svolti da un paziente

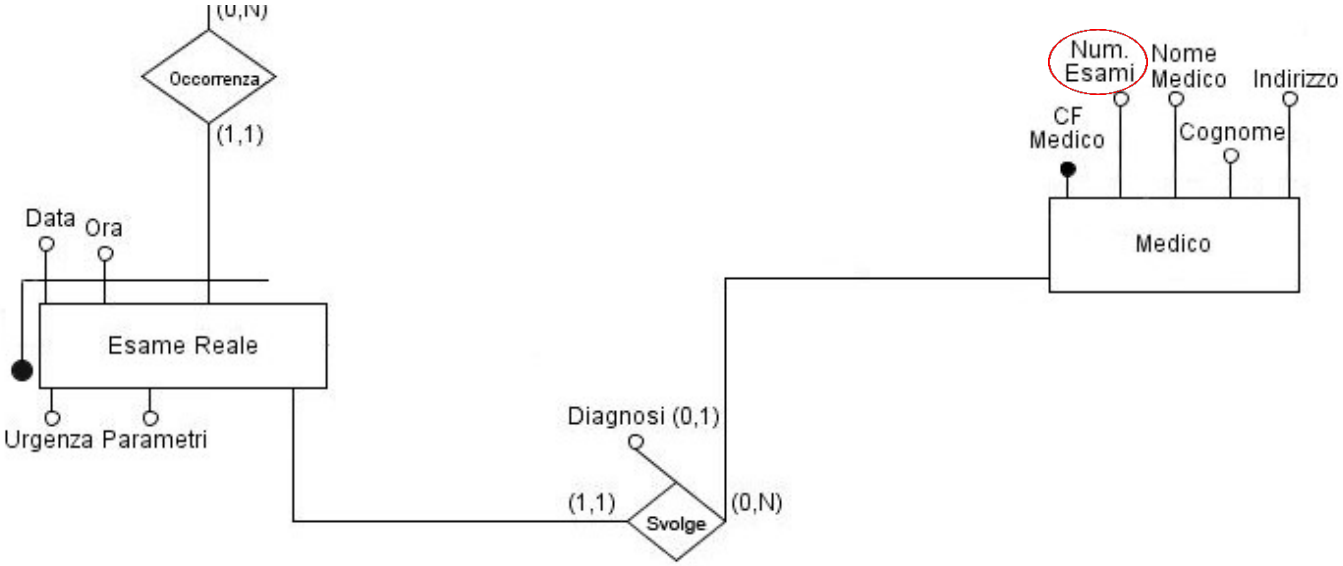
Ho un volume di 20 000 pazienti, 200 000 prenotazioni, 600 000 esami reali. In media ogni paziente effettua 10 prenotazioni, e ad ogni prenotazione corrispondono 3 esami (600000/20000); ogni paziente effettua quindi in media 30 esami. Per mostrare gli esami svolti da un paziente, effettuo un accesso in lettura per leggere il codice fiscale del paziente. Poi effettuo 10 accessi in lettura per verificare quali prenotazioni sono relative al paziente considerato, e per ognuno di questi effettuo 3 accessi in lettura per leggere tutti gli esami relativi alla prenotazione considerata. Effettuiamo in totale quindi 30 accessi in lettura per leggere gli esami associati ad un paziente.

Costo = (1(L) + 30(L))*3500/mese = 31*3500 = 108500

Ristrutturazione dello schema E-R

1. Analisi delle ridondanze

Nell'operazione 16 calcoliamo il numero di esami svolti da un medico con un costo pari a 120 200. Per effettuare questa operazione con costo unitario, potremmo aggiungere l'attributo derivabile "Numero di Esami" relativo all'entità "Medico".



In questo caso bisogna valutare il costo delle operazioni 6 e 7 che aggiungono o cancellano un esame reale.

Se aggiungiamo un esame reale, che viene svolto da un certo medico, dobbiamo incrementare di 1 il valore dell'attributo "Numero di Esami" relativo a quel medico.

Se invece cancelliamo un esame reale, dobbiamo diminuire di 1 il valore dell'attributo "Numero di Esami".

<u>Operazione 5 con attributo derivabile: aggiungi un esame reale</u>

In questo caso effettuiamo una lettura per conoscere il codice di prenotazione, una lettura per conoscere l'esame (disponibile) da svolgere, e una scrittura per aggiungere l'esame. Inoltre, per aggiornare l'attributo "Numero di Esami" effettuiamo un accesso in lettura per conoscere il precedente valore dell'attributo, e successivamente un accesso in scrittura per incrementarne il valore. Costo = (3(L) + 2(S))*45000/mese = (3+2*2)*45000 = 7*45000 = 315 000

Operazione 6 con attributo derivabile: Cancella un esame reale

In questo caso effettuiamo una lettura per leggere l’esame reale da cancellare, poi effettuiamo una scrittura per cancellarlo. Inoltre, per aggiornare l’attributo “Numero di Esami” effettuiamo un accesso in lettura per conoscere il valore precedente dell’attributo, e successivamente un accesso in scrittura per diminuirne il valore.

Costo = (2(L) + 2(S))*10000/mese = (2+2*2)*10000 = 6*10000 = 60 000

Operazione 16 con attributo derivabile: Mostra il numero di esami svolti da un medico

Dobbiamo solo leggere il valore dell’attributo “Numero di Esami” e mostrarlo a schermo

Costo = 1(L)*100/mese = 1*100 = 100

Costo totale con attributo derivabile

100 (Operazione 16) + 315000 (Operazione 6) + 60000 (Operazione 7) = 375 100

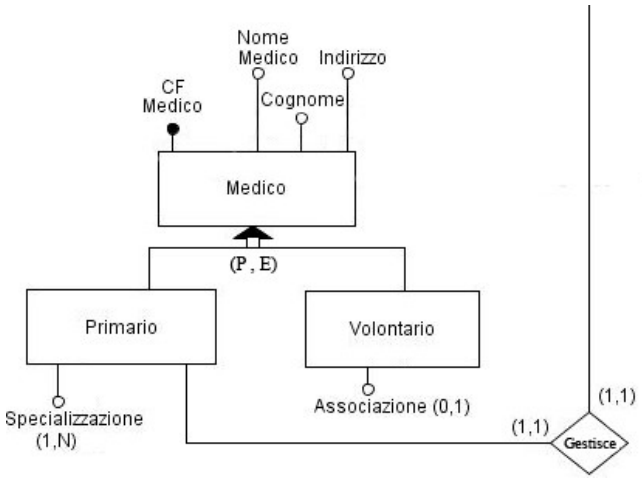
Costo totale senza attributo derivabile

60100 (Operazione 16) + 225000 (Operazione 6) + 45000 (Operazione 7) = 330 100

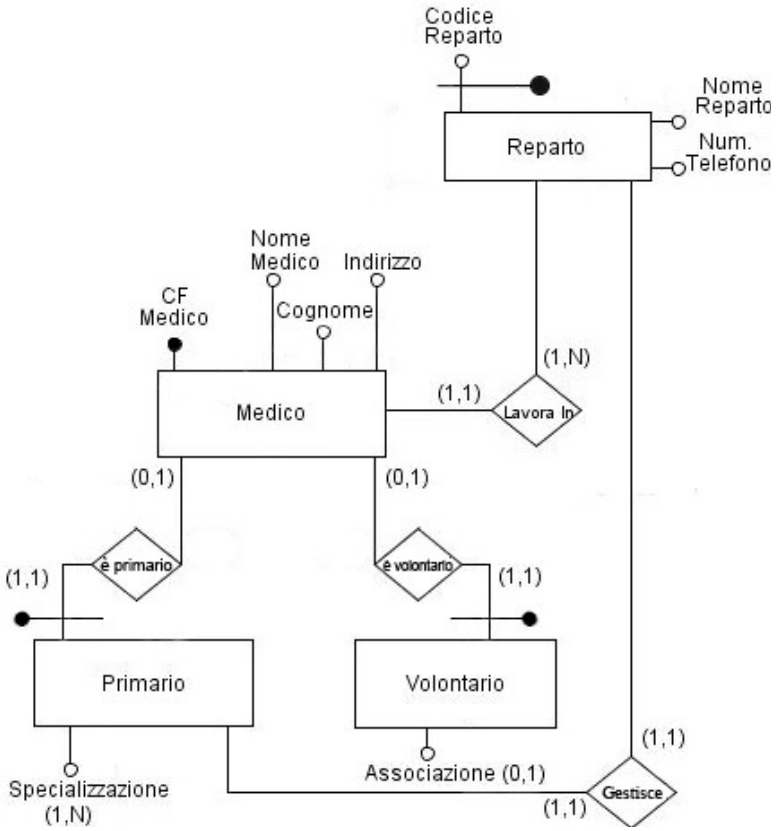
Le operazioni hanno un costo totale minore senza l’attributo “Numero di Esami”, per cui non introduco l’attributo derivabile e lascio invariato lo schema E-R.

2. Eliminazione delle generalizzazioni

L'unica generalizzazione presente nello schema E-R è la seguente:



La generalizzazione è parziale, ed inoltre un medico primario ha un ruolo differente rispetto ad un altro medico o ad un volontario, poiché gestisce un reparto. Sostituisco quindi la generalizzazione con due associazioni uno a uno: l'associazione "è primario" tra medico e primario ha cardinalità (0,1) da medico verso primario, e (1,1) da primario verso medico. L'associazione "è volontario" tra medico e volontario ha cardinalità (0,1) da medico verso volontario, e (1,1) da volontario verso medico. "Primario" e "Volontario" sono entità deboli, identificate univocamente dal codice fiscale del medico.



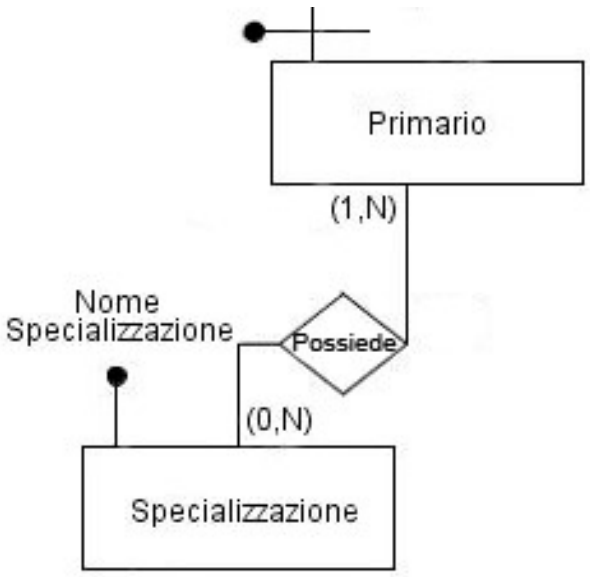
3. **Scelta degli identificatori primari**

Gli identificatori primari scelti per ogni entità sono:

- Paziente: CF Paziente
- Prenotazione: Codice Prenotazione
- Esame Reale: Codice Prenotazione, Codice Esame, Data, Ora
- Esame: Codice Esame
- Laboratorio: Codice Laboratorio, Codice Ospedale
- Ospedale: Codice Ospedale
- Reparto: Codice Reparto, Codice Ospedale
- Medico: CF Medico
- Primario: CF Medico
- Volontario: CF Medico

Trasformazione di attributi e identificatori

Devo eliminare l’attributo multivalore “Specializzazione” dell’entità “Primario”. Creo quindi l’entità “Specializzazione” identificata univocamente dal proprio nome, e la collego tramite un’associazione “possiede” con l’entità “Primario”. Primario verso specializzazione ha cardinalità (1,N), mentre specializzazione verso primario ha cardinalità (0,N) poiché potremmo avere una specializzazione salvata nel nostro database, a cui però non è associato nessun primario.

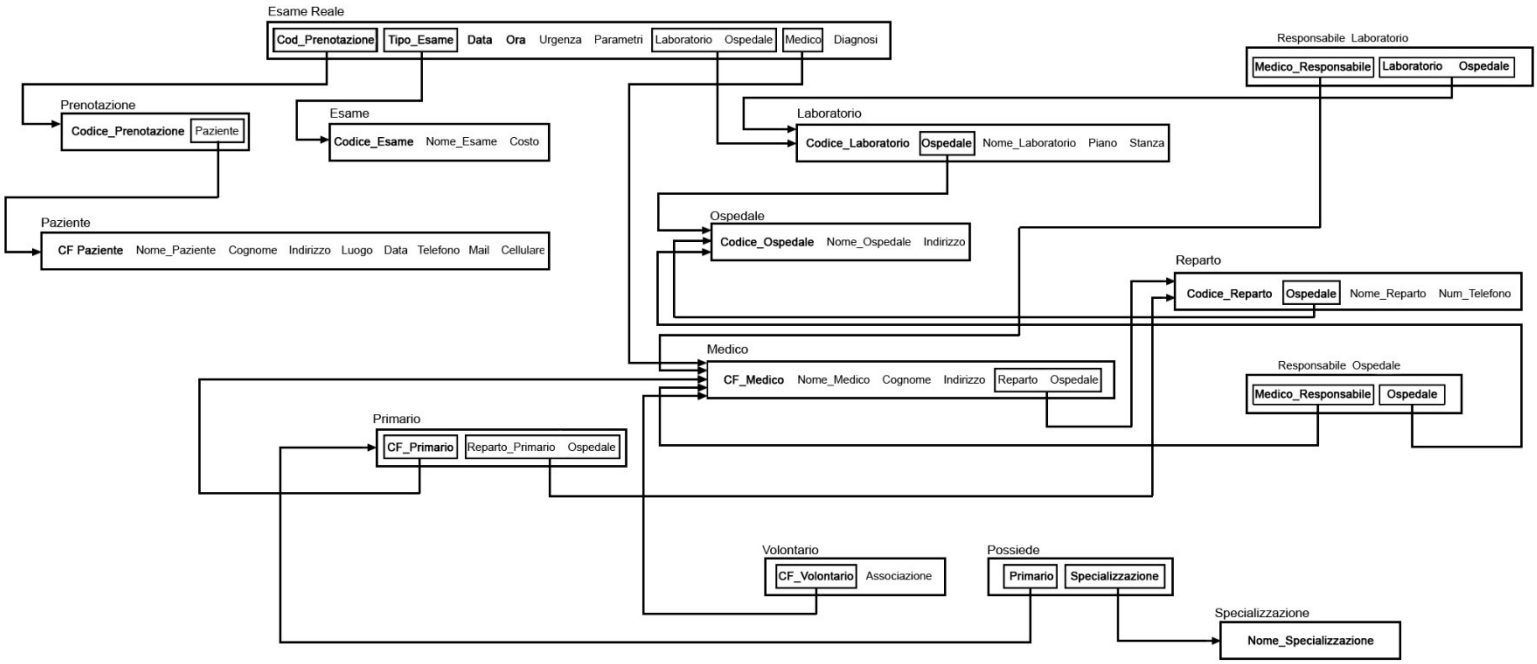


Traduzione di entità e associazioni

- **Paziente** (CF_Paziente, Nome_Paziente, Cognome, Indirizzo, Luogo, Data, Telefono, Mail, Cellulare)
- **Prenotazione** (Codice_Prenotazione, Paziente) con vincolo di integrità referenziale tra Codice_Prenotazione e PAZIENTE
- **Esame** (Codice_Esame, Nome_Esame, Costo)
- **Esame Reale** (Cod_Prenotazione, Tipo_Esame, Data, Ora, Urgenza, Parametri, Laboratorio, Ospedale, Medico, Diagnosi) con vincoli di integrità referenziale tra:
    - Cod_Prenotazione e PRENOTAZIONE
    - Tipo_Esame e ESAME
    - (Laboratorio,Ospedale) e LABORATORIO
    - Medico e MEDICO
- **Laboratorio** (Codice_Laboratorio, Ospedale, Nome_Laboratorio, Piano, Stanza) con vincolo di integrità referenziale tra Ospedale e OSPEDALE
- **Ospedale** (Codice_Ospedale, Nome_Ospedale, Indirizzo)
- **Reparto** (Codice_Reparto, Ospedale, Nome_Reparto, Num_Telefono) con vincolo di integrità referenziale tra Ospedale e OSPEDALE
- **Medico** (CF_Medico, Nome_Medico, Cognome, Indirizzo, Reparto, Ospedale) con vincoli di integrità referenziale tra (Reparto,Ospedale) e REPARTO
- **Primario** (CF_Primario, Reparto_Primario, Ospedale) con vincoli di integrità referenziale tra:
    - CF_Primario e MEDICO,
    - (Reparto_Primario,Ospedale) e REPARTO
- **Volontario** (CF_Volontario, Associazione) con vincolo di integrità referenziale tra CF_Volontario e MEDICO
- **Specializzazione** (Nome_Specializzazione)
- **Possiede** (Primario, Specializzazione) con vincoli di integrità referenziale tra:
    - Primario e PRIMARIO
    - Specializzazione e SPECIALIZZAZIONE
- **Responsabile Laboratorio** (Medico_Responsabile, Laboratorio, Ospedale) con vincoli di integrità referenziale tra:
    - Medico_Responsabile e MEDICO
    - (Laboratorio, Ospedale) e LABORATORIO

- **Responsabile Ospedale** (Medico_Responsabile, Ospedale) con vincoli di integrità referenziale tra:
    - Medico_Responsabile e MEDICO
    - Ospedale e OSPEDALE

Rappresentazione grafica



Normalizzazione del modello relazionale

Le dipendenze funzionali presenti nel nostro modello relazionale sono:

- CF Paziente → Nome Paziente, Cognome, Indirizzo, Luogo, Data
- CF Medico → Nome Medico, Cognome

1NF

Il modello relazionale descritto precedentemente si trova già in forma 1NF. Infatti, per ogni relazione non abbiamo attributi ripetuti, ogni attributo è definito su un dominio con valori atomici, ed ogni relazione ha una chiave che lo identifica.

2NF

Nella forma 2NF dobbiamo eliminare tutte le dipendenze funzionali parziali, e far sì che i valori degli attributi dipendano dalle chiavi minime. Nel nostro modello relazionale non abbiamo dipendenze parziali.

In particolare:

- CF Paziente → Nome Paziente, Cognome, Indirizzo, Luogo, Data
  - Gli attributi Nome Paziente, Cognome, Indirizzo, Luogo e Data dipendono solo dall'attribuito CF Paziente. Poiché CF Paziente è una chiave minimale per la relazione 'Paziente', non abbiamo una dipendenza funzionale parziale.
- CF Medico → Nome Medico, Cognome
  - Gli attributi Nome Medico e Cognome dipendono solo dall'attribuito CF Medico. Poiché CF Medico è una chiave minimale per la relazione 'Medico', non abbiamo una dipendenza funzionale parziale.

3NF

Nella forma 3NF non devono esserci dipendenze transitive tra le relazioni. Per rispettare questa proprietà bisogna eliminare tutte le dipendenze tra due attributi non-chiave.

Nel nostro modello relazionale questa condizione è già rispettata, poiché le dipendenze sono del tipo chiave → attributi non-chiave:

- CF Paziente → Nome Paziente, Cognome, Indirizzo, Luogo, Data
  - Gli attributi Nome Paziente, Cognome, Indirizzo, Luogo e Data dipendono solamente dall'attribuito CF Paziente. Poiché CF Paziente è una chiave, la dipendenza è tra un attributo chiave e un gruppo di attributi non-chiave, che rispetta la forma 3NF

- CF Medico → Nome Medico, Cognome
    - Gli attributi Nome Medico e Cognome dipendono solamente dall'attribuito CF Medico. Poiché CF Medico è una chiave non abbiamo una dipendenza funzionale del tipo non-chiave→non-chiave. Quindi la forma 3NF viene rispettata.

5. Progettazione fisica

Utenti e privilegi

- **Utente 'personale_CUP'**
  - Questo utente può gestire nella sua interezza l'anagrafica dei pazienti e le prenotazioni degli esami. Pertanto, ho fornito all'utente 'personale_CUP' i seguenti privilegi:
    - Tutti i privilegi sulle tabelle 'paziente', 'esamereale', 'prenotazione', poiché deve poter gestire i dati riguardanti queste tabelle nella sua interezza.
    - Privilegi read-only sulle tabelle 'esame', 'medico' e 'laboratorio', poiché per aggiungere la prenotazione di un esame, è necessario conoscerne il codice, il medico che lo ha svolto, ed il laboratorio presso il quale viene effettuato. Per fare ciò ho garantito all'utente 'personale_CUP' il permesso di eseguire delle procedure che permettono di leggere i dati di queste tabelle, ma non di alterarli.
  - Questo utente può inoltre generare dei report che mostrano i risultati degli esami associati ad una prenotazione, e/o mostrare tutti gli esami svolti da un paziente. Pertanto, ho fornito solamente all'utente 'personale_CUP' il permesso di esecuzione per le procedure che generano questi report.

- **Utente 'amministratore'**
  - Questo utente può gestire gli ospedali, e l'insieme degli esami prenotabili presso la ASL. Pertanto, ho fornito all'utente 'amministratore' i seguenti privilegi:
    - Tutti i privilegi sulle tabelle 'esame', 'ospedale', 'laboratorio', 'reparto', 'responsabileospedale' e 'responsabilelaboratorio', poiché un amministratore per gestire gli ospedali nella loro interezza, deve poter modificare anche i reparti ed i laboratori al suo interno e/o modificarne i responsabili.
    - Privilegi di lettura/scrittura per la tabella 'medico'. Infatti, per nominare un responsabile di un laboratorio e/o di un ospedale, è utile che un medico possa avere accesso ad una lista di tutti i medici, o cercarne uno specifico. Inoltre un amministratore deve gestire nella loro interezza l'insieme dei medici, per cui ho concesso all'utente 'amministratore' i permessi per eseguire le stored procedures riguardanti la tabella 'Medico'.

- Gli amministratori possono inoltre generare dei report che mostrano per ciascun medico quanti esami, e quali esami ha svolto. Pertanto, ho fornito al solo utente 'amministratore' il permesso per eseguire le procedure che generano questi report.

Strutture di memorizzazione

Tabella Paziente		
Attributo	**Tipo di dato**	**Attributi**[2]
CF_Paziente	CHAR (16)	PK
Nome_Paziente	VARCHAR (15)	NN
Cognome_Paziente	VARCHAR (15)	NN
Indirizzo	VARCHAR (30)	
Luogo	VARCHAR (30)	
Data	DATE	
Telefono	VARCHAR (10)	
Mail	VARCHAR (30)	
Cellulare	VARCHAR (10)	

Tabella Esame		
Codice_Esame	CHAR (5)	PK
Nome_Esame	VARCHAR (30)	NN
Costo	FLOAT	NN

Tabella Prenotazione		
Codice_Prenotazione	CHAR (8)	PK
Paziente	CHAR (16)	NN

2PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna.

Tabella Ospedale		
Codice_Ospedale	CHAR (4)	PK
Nome_Ospedale	VARCHAR (20)	NN
Indirizzo	VARCHAR (30)	

Tabella Reparto		
Codice_Reparto	CHAR (5)	
Ospedale	CHAR (4)	
Nome_Reparto	VARCHAR (20)	NN
Num_Telefono	CHAR (10)	
Codice_Reparto,Ospedale		PK

Tabella Medico		
CF_Medico	CHAR (16)	PK
Nome_Medico	VARCHAR (15)	NN
Cognome	VARCHAR (15)	NN
Indirizzo	VARCHAR (30)	
Reparto	CHAR (5)	NN
Ospedale	CHAR (4)	NN

Tabella Laboratorio		
Codice_Laboratorio	CHAR (5)	
Ospedale	CHAR (4)	
Nome_Laboratorio	VARCHAR (25)	NN
Piano	INT	
Stanza	INT	
Codice_Laboratorio, Ospedale		PK

Tabella EsameReale		
Cod_Prenotazione	CHAR (8)	
Tipo_Esame	CHAR (5)	
Data	DATE	
Ora	TIME	
Urgenza	VARCHAR (15)	
Parametri	VARCHAR (30)	
Laboratorio	CHAR (5)	NN
Ospedale	CHAR (4)	NN
Medico	CHAR (16)	NN
Diagnosi	VARCHAR (50)	
Cod_Prenotazione, Tipo_Esame, Data, Ora		PK
Cod_Prenotazione, Tipo_Esame, Data		UQ
Data,Ora,Medico		UQ
Data, Ora, Laboratorio, Ospedale		UQ

Tabella Primario		
CF_Primario	CHAR (16)	PK
Reparto_Primario	CHAR (5)	NN
Ospedale	CHAR (4)	NN
Reparto_Primario,Ospedale		UQ

Tabella Volontario		
CF_Volontario	CHAR (16)	PK
Associazione	VARCHAR (50)	

Tabella Specializzazione		
Nome_Specializzazione	VARCHAR (50)	PK

Tabella Possiede		
Primario	CHAR (16)	
Specializzazione	VARCHAR (50)	
Primario, Specializzazione		PK

Tabella ResponsabileLaboratorio		
Medico_Responsabile	CHAR (16)	
Specializzazione	VARCHAR (50)	
Ospedale	CHAR (4)	
Medico_Responsabile, Laboratorio, Ospedale		PK
Laboratorio, Ospedale		UQ

Tabella ResponsabileOspedale		
Medico_Responsabile	CHAR (16)	
Ospedale	CHAR (4)	UQ
Medico_Responsabile, Ospedale		PK

Indici

Tabella Paziente	
Indice PRIMARY	**Tipo[3]:**
CF_Paziente	PR

3IDX = index, UQ = unique, FT = full text, PR = primary.

Tabella Esame	
Indice PRIMARY	**Tipo:**
Codice_Esame	PR

Tabella Prenotazione	
Indice PRIMARY	**Tipo:**
Codice_Prenotazione	PR
Indice Paziente	**Tipo:**
Paziente	IDX

Tabella Ospedale	
Indice PRIMARY	**Tipo:**
Codice_Ospedale	PR

Tabella Reparto	
Indice PRIMARY	**Tipo:**
Codice_Reparto,Ospedale	PR
Indice Ospedale	**Tipo:**
Ospedale	IDX

Tabella Medico	
Indice PRIMARY	**Tipo:**
CF_Medico	PR
Indice Reparto	**Tipo:**
Reparto,Ospedale	IDX

Tabella Laboratorio	
Indice PRIMARY	**Tipo:**
Codice_Laboratorio,Ospedale	PR
Indice Ospedale	**Tipo:**
Ospedale	IDX

Tabella Esamereale	
Indice PRIMARY	**Tipo:**
Cod_Prenotazione, Tipo_Esame, Data, Ora	PR
Indice cod_prenotazione	**Tipo:**
cod_prenotazione, tipo_esame, data	UQ
Indice Data	**Tipo:**
Data, Ora, Medico	UQ
Indice Data_2	**Tipo:**
Data, Ora, Laboratorio, Ospedale	UQ
Indice Laboratorio	**Tipo:**
Laboratorio, Ospedale	IDX
Indice Medico	**Tipo:**
Medico	IDX

Tabella Primario	
Indice PRIMARY	**Tipo:**
CF_Primario	PR
Indice Reparto_Primario	**Tipo:**
Reparto_Primario, Ospedale	UQ

Tabella Volontario	
Indice PRIMARY	**Tipo:**
CF_Volontario	PR

Tabella Specializzazione	
Indice PRIMARY	**Tipo:**
Nome_Specializzazione	PR

Tabella Possiede	
Indice PRIMARY	**Tipo:**
Primario,Specializzazione	PR
Indice Specializzazione	**Tipo:**
Specializzazione	IDX

Tabella ResponsabileLaboratorio	
Indice PRIMARY	**Tipo:**
Medico_Responsabile, Laboratorio, Ospedale	PR
Indice Laboratorio	**Tipo:**
Laboratorio, Ospedale	UQ

Tabella ResponsabileOspedale	
Indice PRIMARY	**Tipo:**
Medico_Responsabile, Ospedale	PR
Indice Ospedale	**Tipo:**
Ospedale	UQ

Trigger

Per la progettazione del Database non ho creato nessun Trigger, poiché non necessari per implementare le funzionalità richieste. Non ho utilizzato, inoltre, neanche CHECK e/o ASSERZIONI. Infatti, ho implementato le regole aziendali descritte precedentemente

1) Lo stesso esame disponibile NON PUO’ essere ripetuto nello stesso giorno dallo stesso paziente.
2) Un medico NON PUO’ effettuare due esami reali contemporaneamente.

```
3) In un laboratorio NON POSSONO essere effettuati due esami reali contemporaneamente.
```

utilizzando il costrutto UNIQUE durante il CREATE TABLE. La regola 1 è garantita imponendo come UNIQUE la tupla (Cod_Prenotazione,Tipo_Esame,Data). La regola 2 è garantita imponendo come UNIQUE la tupla (Medico,Data,Ora). La regola 3 è garantita imponendo come UNIQUE la tupla (Laboratorio,Ospedale,Data,Ora).

Eventi

Per la realizzazione del database non ho creato Eventi.

Viste

Per la realizzazione del database non ho fatto utilizzo delle Viste.

Stored Procedures e transazioni

- **paziente_cerca(CF).**
  - Procedura che prende in input un codice fiscale, che tramite una select restituisce tutti i dati del paziente cercato.

```
CREATE PROCEDURE ASL.paziente_cerca (IN CodiceFiscale CHAR(16))
SELECT *
FROM ASL.paziente
WHERE CF_Paziente = CodiceFiscale;
```

- **mostra_pazienti()**
  - Procedura che mostra tutti i pazienti presenti nel database, con i loro relativi dati

```
CREATE PROCEDURE ASL.mostra_pazienti ()
SELECT *
FROM ASL.paziente;
```

- **paziente_aggiungi (CF,Nome,Cognome,Indirizzo,Luogo,Data,Telefono,Mail,Cellulare)**
  - Procedura che prende in input tutti i dati relativi ad una persona, e la registra all'interno del database come un paziente.

```
CREATE PROCEDURE ASL.paziente_aggiungi
(
IN CF CHAR(16),
IN nome VARCHAR(15),
IN cognome VARCHAR(15),
IN indirizzo VARCHAR(30),
IN Luogo VARCHAR(30),
IN data DATE,
IN Telefono VARCHAR(10),
IN Mail VARCHAR(30),
IN Cellulare VARCHAR(10)
)
INSERT INTO ASL.Paziente
VALUES (CF,nome,cognome,indirizzo,luogo,data,telefono,mail,cellulare);
```

- **paziente_cancella (CF)**
  - Procedura che prende in input un codice fiscale, e cancella dalla tabella 'paziente 'la persona avente quel codice fiscale.

```
CREATE PROCEDURE ASL.paziente_cancella(IN CF CHAR(16))
DELETE FROM ASL.paziente
WHERE CF_paziente = CF;
```

- **Paziente_modifica (CF,indirizzo,telefono,mail,cellulare)**
  - Procedura che prende in input il codice fiscale del paziente da modificare, ed i nuovi valori per i campi indirizzo,telefono,mail,cellulare.
  - Ho ritenuto utile poter modificare solo questi dati riguardo un paziente, poiché modificare i dati anagrafici (Nome, cognome, luogo), comporterebbe dover modificare anche il Codice Fiscale. In questo caso è conveniente cancellare il paziente in questione tramite *paziente_cancella(CF)* e successivamente aggiungerlo, con i dati aggiornati e coerenti, tramite *paziente_aggiungi(Dati)*

```
CREATE PROCEDURE ASL.paziente_modifica (IN CF CHAR(16),IN n_indirizzo
VARCHAR(30), IN n_Telefono VARCHAR(10),IN n_Mail VARCHAR(30),IN n_Cellulare
VARCHAR(10))
UPDATE ASL.paziente
SET Indirizzo = n_indirizzo,
    Telefono = n_Telefono,
    Mail = n_Mail,
    Cellulare = n_cellulare
WHERE CF = CF_Paziente;
```

- **prenotazione_aggiungi (CFpaziente,Cod_Prenot)**
  - Procedura che apre una nuova prenotazione, identificata da *Cod_Prenot* e relativa al paziente indicato in input.

```
CREATE PROCEDURE ASL.prenotazione_aggiungi (IN cod_prenotazione CHAR(8),IN
cf_paziente CHAR(16))
INSERT INTO ASL.prenotazione
VALUES (cod_prenotazione,cf_paziente);
```

- **prenotazione_cancella (Codice)**
  - Procedura che prendendo in input un codice di prenotazione, lo cerca all'interno della tabella 'prenotazione', e lo cancella.

```
CREATE PROCEDURE ASL.prenotazione_cancella (IN cod_prenot CHAR(8))
DELETE FROM ASL.prenotazione
WHERE Codice_Prenotazione=cod_prenot;
```

- **mostra_prenotazioni ()**
  - Procedura che mostra tutte le prenotazioni registrate nel sistema.

```
CREATE PROCEDURE ASL.mostra_prenotazioni ()
SELECT *
FROM ASL.prenotazione ORDER BY Codice_Prenotazione;
```

- **esamereale_aggiungi** (cod_pren,esame,data,ora,urgenza,parametri,laborat,ospedale,medico)
    - Procedura che prende in input tutti i dati relativi ad un certo esame prenotato, e lo aggiunge alla tabella 'esamereale'

```
CREATE PROCEDURE ASL.esamereale_aggiungi
(
IN cod_prenot CHAR(8),
cod_esame CHAR (5),
data DATE,
ora TIME,
urgenza VARCHAR(15),
Parametri VARCHAR(30),
Laboratorio CHAR(5),
Ospedale CHAR(4),
Medico CHAR(16),
diagnosi VARCHAR(50)
)
INSERT INTO ASL.esamereale
VALUES
(cod_prenot,cod_esame,data,ora,urgenza,parametri,laboratorio,ospedale,medico,diag
nosi);
```

- **esamereale_cancella (Cod_pren,cod_es,data,ora)**
    - Procedura che prende in input i dati necessari ad identificare uno specifico esame reale, e lo cancella dalla tabella 'esamereale'

```
CREATE PROCEDURE ASL.esamereale_cancella (IN cod_prenot CHAR(8), cod_esame CHAR
(5), data_ DATE, ora_ TIME)
DELETE FROM ASL.esamereale
WHERE (cod_prenot,cod_esame,data_,ora_) = (Cod_Prenotazione,tipo_esame,data,ora);
```

- **esamereale_modifica_data (cod_prenot, nuova_data, nuova_ora)**
    - Procedura che permette di modificare la data e l'ora di un esame prenotato o svolto, modificando i valori data e ora, con quelli inseriti in input.

```
CREATE PROCEDURE ASL.esamereale_modifica_data (IN cod_prenot CHAR(8), IN esame
CHAR(5),IN data DATE,IN ora TIME, nuova_data DATE, nuova_ora TIME)
UPDATE ASL.esamereale
SET data = nuova_data,
      ora = nuova_ora
WHERE (Cod_Prenotazione,Tipo_Esame,Data,Ora) = (cod_prenot,esame,data,ora);
```

- **esamereale_inserisci_diagnosi (cod_prenot, cod_esame, data, ora, n_param, n_diagn)**
    - Procedura che permette di inserire e/o modificare i valori 'parametri' e 'diagnosi' relativi ad uno specifico esame reale, identificato tramite i valori inseriti in input *cod_prenot,cod_esame,data,ora*

```
CREATE PROCEDURE ASL.esamereale_inserisci_diagnosi (IN cod_prenot CHAR(8), cod_esame CHAR (5), data DATE, ora TIME, IN nuovi_parametri VARCHAR(30), IN nuova_diagnosi VARCHAR(50))
UPDATE ASL.esamereale
SET parametri = nuovi_parametri,
      diagnosi = nuova_diagnosi
WHERE (Cod_Prenotazione,Tipo_Esame,Data,Ora) = (cod_prenot,cod_esame,data,ora);
```

- **esamereale_cerca (cod_prenot, cod_esame, data, ora)**
    - Procedura che permette di cercare, uno specifico esame reale tramite i valori della chiave inseriti in input. Se la ricerca ha successo, vengono mostrati tutti i dati relativi all'esame reale cercato.

```
CREATE PROCEDURE ASL.esamereale_cerca (IN cod_prenot CHAR(8), cod_esame CHAR (5), data_ DATE, ora_ TIME)
SELECT Codice_Prenotazione AS
Codice,Paziente,Tipo_Esame,Data,Ora,Urgenza,Parametri,Laboratorio,Ospedale,Medico
,Diagnosi
FROM ASL.esamereale JOIN ASL.prenotazione ON esamereale.Cod_Prenotazione =
prenotazione.Codice_Prenotazione
WHERE (cod_prenot,cod_esame,data,ora) = (Cod_Prenotazione,Tipo_Esame,Data,Ora);
```

- **mostra_esamireali ()**
    - Procedura che mostra tutti i dati relativi ad ogni esame reale registrato all'interno del database.

```
CREATE PROCEDURE ASL.mostra_esamireali ()
SELECT Codice_Prenotazione AS
Codice,Paziente,Tipo_Esame,Data,Ora,Urgenza,Parametri,Laboratorio,Ospedale,Medico
,Diagnosi
FROM ASL.esamereale JOIN ASL.prenotazione ON esamereale.Cod_Prenotazione =
prenotazione.Codice_Prenotazione;
```

- **esame_aggiungi (codice,nome,costo)**
  - Procedura che prendendo in input un codice, un nome, ed un costo, aggiunge al database un esame disponibile con i dati inseriti.

```
CREATE PROCEDURE ASL.esame_aggiungi (IN cod_esame CHAR(5), IN nome_esame
VARCHAR(30), costo FLOAT)
INSERT INTO ASL.esame
VALUES (cod_esame,nome_esame,costo);
```

- **esame_cancella (codice)**
  - Procedura che cancella un esame, avente il codice inserito in input.

```
CREATE PROCEDURE ASL.esame_aggiungi (IN cod_esame CHAR(5), IN nome_esame
VARCHAR(30), costo FLOAT)
INSERT INTO ASL.esame
VALUES (cod_esame,nome_esame,costo);
```

- **esame_modifica (codice,nuovo_codice,nuovo_nome,nuovo_costo)**
  - Procedura che cerca uno specifico esame tramite il suo codice, e ne cambia tutti i valori con quelli inseriti in input

```
CREATE PROCEDURE ASL.esame_modifica (IN cod_esame CHAR(5), IN nuovo_codice
CHAR(5), IN nuovo_nome VARCHAR(30), IN nuovo_costo FLOAT)
UPDATE ASL.esame
SET codice_esame = nuovo_codice,nome_esame = nuovo_nome,costo = nuovo_costo
WHERE Codice_Esame = cod_esame;
```

- **esame_cerca (codice)**
  - Procedura che cerca uno specifico esame tramite il codice inserito in input, e ne mostra tutti i dati.

```
CREATE PROCEDURE ASL.esame_cerca (IN cod_esame CHAR(5))
SELECT *
FROM ASL.esame
WHERE cod_esame = Codice_Esame;
```

- **mostra_esami ()**
  - Procedura che mostra tutti i dati degli esami disponibili registrati nel database.

```
CREATE PROCEDURE ASL.mostra_esami ()
SELECT *
FROM ASL.esame;
```

- **ospedale_aggiungi (codice,nome,indirizzo)**
  - Procedura che prende in input tutti i dati necessari per descrivere un ospedale, e li registra all'interno del database.

```
CREATE PROCEDURE ASL.ospedale_aggiungi (IN cod_osp CHAR(4), IN nome_osp
VARCHAR(20), IN indirizzo VARCHAR(30))
INSERT INTO ASL.ospedale
VALUES (cod_osp,nome_osp,indirizzo);
```

- **ospedale_cancella (codice)**
  - Procedura che prende in input il codice di un ospedale, e lo cancella dal database.

```
CREATE PROCEDURE ASL.ospedale_cancella (IN cod_osp CHAR(4))
DELETE FROM ASL.ospedale
WHERE Codice_Ospedale = cod_osp;
```

- **ospedale_cerca (codice)**
  - Procedura che prende in input il codice di un ospedale, lo cerca all'interno della tabella 'ospedale', e ne restituisce tutti i dati relativi.

```
CREATE PROCEDURE ASL.ospedale_cerca (IN cod_osp CHAR(4))
SELECT *
FROM ASL.ospedale
WHERE Codice_Ospedale = cod_osp;
```

- **mostra_ospedali ()**
  - Procedura che mostra tutti i dati relativi agli ospedali registrati nel database**,** ed il medico responsabile di quell'ospedale.

```
CREATE PROCEDURE ASL.mostra_ospedali()
SELECT Codice_Ospedale AS Codice,Nome_Ospedale AS
Nome,Indirizzo,Medico_Responsabile AS Responsabile
FROM ASL.ospedale
JOIN ASL.responsabileospedale ON ospedale.Codice_Ospedale =
responsabileospedale.Ospedale;
```

- **ospedale_modifica(codice,n_codice,n_nome,n_indirizzo)**
  - Procedura che prende in input il codice di un ospedale presente nel database, e ne modifica i valori con **,***n_codice,n_nome,n_indirizzo*.

```
CREATE PROCEDURE ASL.ospedale_modifica (IN cod_osp CHAR(4), IN nuovo_codice CHAR(4),IN nuovo_nome VARCHAR(20), IN nuovo_indirizzo VARCHAR(30))
UPDATE ASL.ospedale
SET Codice_Ospedale = nuovo_codice, Nome_Ospedale = nuovo_nome, Indirizzo = nuovo_indirizzo
WHERE Codice_Ospedale = cod_osp;
```

- **laboratorio_aggiungi(codice,ospedale,nome,piano,stanza)**
  - Procedura che pende in input i dati necessari per descrivere un laboratorio, e li registra all'interno del database.

```
CREATE PROCEDURE ASL.laboratorio_aggiungi (IN cod_lab CHAR(5), IN cod_osp CHAR(4), IN nome VARCHAR(25), IN piano INT, IN stanza INT)
INSERT INTO ASL.laboratorio
VALUES (cod_lab,cod_osp,nome,piano,stanza);
```

- **laboratorio_cancella(laboratorio,ospedale)**
  - Procedura che prende in input il codice di un laboratorio e dell'ospedale di appartenenza, e lo cancella dal database.

```
CREATE PROCEDURE ASL.laboratorio_cancella (IN cod_lab CHAR(5), IN cod_osp CHAR(4))
DELETE FROM ASL.laboratorio
WHERE (Codice_Laboratorio,Ospedale) = (cod_lab,cod_osp);
```

- **laboratorio_cerca(laboratorio,ospedale)**
  - Procedura che prende in input il codice di un laboratorio e dell'ospedale di appartenenza, cerca i valori all'interno di 'laboratorio', e ne mostra i relativi dati.

```
CREATE PROCEDURE ASL.laboratorio_cerca (IN cod_lab CHAR(5), IN cod_osp CHAR(4))
SELECT *
FROM ASL.laboratorio
WHERE (Codice_Laboratorio,Ospedale) = (cod_lab,cod_osp);
```

- **mostra_laboratori()**
  - Procedura che mostra tutti i dati relativi ai laboratori presenti nel database, ed il medico responsabile di quel laboratorio.

```
CREATE PROCEDURE ASL.mostra_laboratori()
SELECT Codice_Laboratorio AS Laboratorio,laboratorio.Ospedale AS
Ospedale,Nome_Laboratorio AS Nome,Piano,Stanza,Medico_Responsabile AS
Responsabile
FROM ASL.laboratorio
JOIN ASL.responsabilelaboratorio ON laboratorio.Codice_Laboratorio =
responsabilelaboratorio.Laboratorio
AND laboratorio.Ospedale = responsabilelaboratorio.Ospedale;
```

- **laboratorio_modifica(lab,osp,n_lab,n_osp,n_nome,n_piano,n_stanza)**
  - Procedura che cerca uno specifico laboratorio tramite la sua chiave, e ne modifica i valori, sostituendoli con quelli forniti in input.

```
CREATE PROCEDURE ASL.laboratorio_modifica (IN cod_lab CHAR(5), IN cod_osp
CHAR(4), IN nuovo_lab CHAR(5), IN nuovo_osp CHAR(4), IN nuovo_nome VARCHAR(25),
IN nuovo_piano INT, IN nuova_stanza INT)
UPDATE ASL.laboratorio
SET Codice_Laboratorio = nuovo_lab,
      Ospedale = nuovo_osp,
    Nome_Laboratorio = nuovo_nome,
    Piano = nuovo_piano,
    Stanza = nuova_stanza
WHERE (Codice_Laboratorio,Ospedale) = (cod_lab,cod_osp);
```

- **medico_aggiungi(CF,nome,cognome,indirizzo,reparto,ospedale)**
  - Procedura che permette di registrare un nuovo medico nel database

```
CREATE PROCEDURE ASL.medico_aggiungi (IN cod_medico CHAR(16), nome_medico
VARCHAR(15), cognome_medico VARCHAR(15), indirizzo VARCHAR(30), IN reparto
CHAR(5),IN ospedale CHAR(4))
INSERT INTO ASL.medico
VALUES (cod_medico,nome_medico,cognome_medico,indirizzo,reparto,ospedale);
```

- **medico_cancella(CF)**
  - Procedura che permette di cancellare un medico dal database, tramite il codice fiscale fornito in input.

```
CREATE PROCEDURE ASL.medico_cancella (IN cod_medico CHAR(16))
DELETE FROM ASL.medico
WHERE CF_Medico = cod_medico;
```

- **medico_modifica(CF,n_indirizzo,n_reparto,n_ospedale)**
  - Procedura che permette di modificare indirizzo,reparto,ospedale, relativi al medico identificato dal codice fiscale fornito in input

```
CREATE PROCEDURE ASL.medico_modifica (IN cod_medico CHAR(16), nuovo_indirizzo
VARCHAR(30), IN nuovo_reparto CHAR(5),IN nuovo_ospedale CHAR(4))
UPDATE ASL.medico
SET Indirizzo = nuovo_indirizzo,
    Reparto = nuovo_reparto,
    Ospedale = nuovo_ospedale
WHERE CF_Medico = cod_medico;
```

- **nomina_resp_osp(CF,Ospedale)**
  - Procedura che permette di assegnare ad un ospedale il medico che ne è responsabile, inserendo il CF del medico e il codice dell'ospedale forniti in input, nella tabella 'responsabileospedale'

```
CREATE PROCEDURE ASL.nomina_resp_osp (IN cod_medico CHAR(16),IN ospedale CHAR(4))
INSERT INTO ASL.responsabileospedale
VALUES (cod_medico,ospedale);
```

- **nomina_resp_lab(CF,Laboratorio,Ospedale)**
  - Procedura che permette di assegnare ad un laboratorio il medico che ne è responsabile, inserendo il CF del medico, il codice del laboratorio ed il codice dell'ospedale forniti in input, nella tabella 'responsabilelaboratorio'

```
CREATE PROCEDURE ASL.nomina_resp_lab (IN cod_medico CHAR(16), IN laboratorio
CHAR(5), IN ospedale CHAR(4))
INSERT INTO ASL.responsabilelaboratorio
VALUES (cod_medico,laboratorio,ospedale);
```

- **medico_cerca (CF)**
  - Procedura che mostra tutti i dati relativi ad uno specifico medico, identificato dal codice fiscale inserito in input.

```
CREATE PROCEDURE ASL.medico_cerca (IN cod_medico CHAR(16))
SELECT *
FROM ASL.medico
WHERE CF_Medico = cod_medico;
```

- **mostra_medici()**
  - Procedura che mostra tutti i dati dei medici registrati nel database

```
CREATE PROCEDURE ASL.mostra_medici ()
SELECT *
FROM ASL.medico;
```

- **nomina_primario (CF)**
  - Procedura che nomina un medico come primario del reparto in cui lavora, inserendo nella tabella 'primario' il codice fiscale del medico fornito in input, il reparto in cui lavora e l'ospedale. Il reparto e l'ospedale in cui lavora il medico vengono ricavati tramite una SELECT sulla tabella medico.

```
CREATE PROCEDURE ASL.nomina_primario (IN cod_medico CHAR(16))
INSERT INTO ASL.primario (CF_Primario,Reparto_Primario,Ospedale)
SELECT CF_Medico,Reparto,Ospedale
FROM ASL.medico
WHERE CF_Medico = cod_medico;
```

- **rimuovi_primario(CF)**
  - Procedura che rimuove un primario dalla tabella 'primario'.

```
CREATE PROCEDURE ASL.rimuovi_primario (IN cod_primario CHAR(16))
DELETE FROM ASL.primario
WHERE cod_primario = CF_Primario;
```

- **primario_agg_spec(Primario,Specializzazione)**
  - Procedura che permette di aggiungere una specializzazione ad un medico primario, inserendo i valori *primario* e *specializzazione* nella tabella 'possiede'

```
CREATE PROCEDURE ASL.primario_agg_spec(IN CF_primario CHAR(16),IN spec
VARCHAR(50))
INSERT INTO ASL.possiede
VALUES (CF_primario,spec);
```

- **primario_rim_spec(Primario,Specializzazione)**
  - Procedura che permette di rimuovere una specializzazione di un medico primario.

```
CREATE PROCEDURE ASL.primario_rim_spec(IN CF_primario CHAR(16),IN spec
VARCHAR(50))
DELETE FROM ASL.possiede
WHERE (CF_primario,spec) = (Primario,Specializzazione);
```

- **primario_cerca(CF)**
  - Procedura che cerca un primario tramite il codice fiscale in input, e ne mostra i relativi dati.

```
CREATE PROCEDURE ASL.primario_cerca (IN CodiceFiscale CHAR(16))
SELECT CF_Primario,Nome_Medico,Cognome,Indirizzo,medico.Reparto,medico.Ospedale
FROM ASL.primario
      JOIN ASL.medico ON primario.CF_Primario = medico.CF_Medico
WHERE CF_Primario = CodiceFiscale;
```

- **mostra_primari()**
  - Procedura che permette di mostrare i dati di tutti i primari registrati nel database

```
CREATE PROCEDURE ASL.mostra_primari ()
SELECT CF_Primario,Nome_Medico,Cognome,Indirizzo,medico.Reparto,medico.Ospedale
FROM ASL.primario
      JOIN ASL.medico ON primario.CF_Primario = medico.CF_Medico;
```

- **primario_mostra_spec(CF)**
  - Procedura che mostra tutte le specializzazioni che possiede un certo medico primario, identificato dal codice fiscale fornito in input.

```
CREATE PROCEDURE ASL.primario_mostra_spec (IN CF CHAR(16))
SELECT *
FROM ASL.possiede
WHERE Primario = CF;
```

- **nomina_volontario (CF,Associazione)**
  - Procedura che specifica un medico come medico volontario, aggiungendo alla tabella 'volontario' il codice fiscale e l'eventuale associazione di appartenenza passate come parametri.

```
CREATE PROCEDURE ASL.nomina_volontario (IN cod_medico CHAR(16),IN associazione
VARCHAR(50))
INSERT INTO ASL.volontario
VALUES (cod_medico,associazione);
```

- **rimuovi_volontario (CF)**
  - Cerca uno specifico volontario e lo rimuove dal database

```
CREATE PROCEDURE ASL.rimuovi_volontario (IN cod_volontario CHAR(16))
DELETE FROM ASL.volontario
WHERE CF_Volontario = cod_volontario;
```

- **volontario_cerca (CF)**
  - Cerca un volontario nel database e ne mostra i relativi dati.

```
CREATE PROCEDURE ASL.volontario_cerca (IN CodiceFiscale CHAR(16))
SELECT CF_Volontario AS CF,Nome_Medico AS
Nome,Cognome,Indirizzo,medico.Reparto,medico.Ospedale,Associazione
FROM ASL.volontario
      JOIN ASL.medico ON volontario.CF_Volontario = medico.CF_Medico
WHERE CF_Volontario = CodiceFiscale;
```

- **mostra_volontari()**
  - Procedura che mostra tutti i dati dei volontari registrati nel database.

```
CREATE PROCEDURE ASL.mostra_volontari ()
SELECT CF_Volontario AS CF,Nome_Medico AS
Nome,Cognome,Indirizzo,medico.Reparto,medico.Ospedale,Associazione
FROM ASL.volontario
      JOIN ASL.medico ON volontario.CF_Volontario = medico.CF_Medico;
```

- **reparto_aggiungi (cod_reparto,cod_ospedale,nome,numero)**
  - Procedura che inserisce un nuovo reparto nel database, inserendo nella tabella 'reparto' i valori passati in input

```
CREATE PROCEDURE ASL.reparto_aggiungi (IN cod_reparto CHAR(5), IN cod_ospedale
CHAR(4), IN nome VARCHAR(20), IN numero_telefono CHAR(10))
INSERT INTO ASL.reparto
VALUES (cod_reparto,cod_ospedale,nome,numero_telefono);
```

- **reparto_modifica (reparto,ospedale,n_rep,n_osp,n_nome,n_numero)**
  - Procedura che aggiorna i dati di un reparto identificato dalla coppia *reparto,ospedale* passata in input, inserendo i nuovi valori *n_rep,n_osp,n_nome,n_numero*

```
CREATE PROCEDURE ASL.reparto_modifica (IN cod_reparto CHAR(5), IN cod_ospedale
CHAR(4),IN nuovo_codice CHAR(5),IN nuovo_ospedale CHAR(4), IN nuovo_nome
VARCHAR(20), IN nuovo_numero CHAR(10))
UPDATE ASL.reparto
SET Codice_Reparto = nuovo_codice,
      Ospedale = nuovo_ospedale,
    Nome_Reparto = nuovo_nome,
    Num_Telefono = nuovo_numero
WHERE (Codice_Reparto,Ospedale) = (cod_reparto,cod_ospedale);
```

- **reparto_cancella (cod_rep,cod_osp)**
  - Procedura che cancella uno specifico reparto identificato dai valori passati in input

```
CREATE PROCEDURE ASL.reparto_cancella (IN cod_reparto CHAR(5),IN cod_ospedale
CHAR(4))
DELETE FROM ASL.reparto
WHERE (Codice_Reparto,Ospedale) = (cod_reparto,cod_ospedale);
```

- **reparto_cerca (cod_rep,cod_osp)**
  - Procedura che cerca all'interno del database un certo reparto tramite i valori della chiave in input, e ne mostra tutti i dati associati

```
CREATE PROCEDURE ASL.reparto_cerca (IN cod_reparto CHAR(5), IN cod_ospedale
CHAR(4))
SELECT *
FROM ASL.reparto
WHERE (Codice_Reparto,Ospedale) = (cod_reparto,cod_ospedale);
```

- **mostra_reparti()**
  - Mostra i dati di tutti i reparti salvati all'interno del database.

```
CREATE PROCEDURE ASL.mostra_reparti ()
SELECT *
FROM ASL.reparto;
```

- **report_medici_esami(data_inizio,data_fine)**
  - Report generabile dall'amministratore di sistema, che mostra, per ciascun medico registrato nel database, tutti gli esami svolti durante un certo periodo.
  - Il periodo sul quale viene generato il report, viene espresso tramite i due parametri della procedura *data_inizio* e *data_fine.* La procedura utilizza il right join tra Esamereale e medico, così da avere, per ogni medico, tutti gli esami svolti nel periodo indicato. Utilizzando il right join includiamo nel report anche i medici che non hanno svolto esami, riportandone solo CF,nome,cognome, mentre tutti gli attributi relativi all'esame avranno valore *null*
  - Il join con la tabella esame, serve per esprimere, oltre al codice dell'esame effettuato, anche il nome di quest'ultimo, poiché molto più esplicativo e pratico all'interno del report.

```
CREATE PROCEDURE ASL.report_medici_esami (IN data_inizio DATE,IN data_fine DATE)
SELECT CF_Medico AS Codice_Medico,Nome_Medico,Cognome AS
Cognome_Medico,Tipo_Esame,Nome_Esame,Data,Ora
FROM ASL.esamereale
      JOIN ASL.esame ON esamereale.Tipo_Esame = esame.Codice_Esame
      RIGHT JOIN ASL.medico ON esamereale.Medico = medico.CF_Medico
WHERE Data>=data_inizio AND Data<=data_fine OR Data is NULL
ORDER BY Cognome_Medico;
```

- **report_numero_esami (data_inizio,data_fine)**
  - Report generabile dall’amministratore di sistema, che mostra per ciascun medico il numero di esami che ha svolto durante un certo periodo.
  - Il periodo viene definito tramite i parametri della procedura *data_inizio* e *data_fine*. La procedura utilizza il right join tra Esamereale e medico, così da includere nel report anche i medici che non hanno svolto esami, riportando ‘0’ sotto la colonna *num_esami*, che indica il numero di esami svolti da ogni medico.
  - Il join con la tabella esame, serve per esprimere, oltre al codice dell’esame effettuato, anche il nome di quest’ultimo, poiché molto più esplicativo e pratico all’interno del report.

```
CREATE PROCEDURE ASL.report_numero_esami (data_inizio DATE, data_fine DATE)
SELECT CF_Medico AS Codice,Nome_Medico AS Nome ,Cognome, COUNT(Medico) AS
num_esami
FROM ASL.esamereale RIGHT JOIN ASL.medico ON esamereale.Medico = medico.CF_Medico
WHERE Data is null or Data>=data_inizio AND Data<=data_fine
GROUP BY CF_Medico
ORDER BY num_esami DESC;
```

- **report_esami_paziente (CodiceFiscale)**
  - Questa procedura, eseguibile dal personale del CUP, restituisce un report contenente tutti gli esami svolti da un paziente, identificato dal suo codice fiscale fornito in input.
  - Nella procedura utilizzo 3 JOIN:
    - Il join tra Paziente e Prenotazione permette di ottenere tutte le prenotazioni effettuate da un paziente
    - Il successivo join con Esamereale permette di ottenere tutti gli esami effettuati da un paziente, poiché prendo oltre tutte le prenotazioni del paziente, anche tutti gli esami associati ad ogni prenotazione.
    - Il successivo join con Esame permette di ottenere oltre al codice dell’esame, anche il suo nome, rendendo così più esplicito il report.

```
CREATE PROCEDURE ASL.report_esami_paziente (IN CF CHAR(16))
SELECT
CF_Paziente AS CodiceFiscale,
 Nome_Paziente AS Nome,
 Cognome_Paziente AS Cognome,
 Tipo_Esame AS cod_Esame,
 Nome_Esame AS Esame,
 esamereale.Data AS Data,
 esamereale.Ora AS Ora,
 Diagnosi
FROM ASL.paziente
      JOIN ASL.prenotazione ON paziente.CF_Paziente = prenotazione.Paziente
    JOIN ASL.esamereale ON prenotazione.Codice_Prenotazione =
esamereale.Cod_Prenotazione
    JOIN ASL.esame ON esamereale.Tipo_Esame = esame.Codice_Esame
WHERE CF_Paziente = CF;
```

- **report_risultati_prenotazione (codice_prenotazione)**
  - Questa procedura, eseguibile dal personale del CUP, genera un report contenente tutti i risultati degli esami associati ad una prenotazione, identificata dal codice fornito in input.
  - Nella procedura utilizzo un JOIN tra esamereale ed esame, così da fornire nel report, oltre al codice dell’esame, anche il suo nome.
  - Per selezionare tutti gli esami associati ad una prenotazione impongo, nella clausola WHERE, che i valori relativi alla colonna *cod_prenotazione* della tabella ‘esamereale’ debbano essere uguali al valore *codice_prenotazione* passato come parametro della procedura. Non sono necessari join con altre tabelle in quanto ‘esamereale’ contiene tutti gli esami svolti, ed in particolare quindi, anche quelli relativi alla prenotazione cercata.
  - Eseguo il join tra esamereale ed esame, per mostrare nel report il nome dell’esame, oltre al suo codice.

```
CREATE PROCEDURE ASL.report_risultati_prenotazione(IN prenotazione CHAR(8))
SELECT Tipo_Esame AS Codice,Nome_Esame,Data,Ora,Parametri,Diagnosi
FROM ASL.esamereale JOIN ASL.esame ON esamereale.Tipo_Esame = esame.Codice_Esame
WHERE prenotazione = Cod_Prenotazione;
```

Appendice: Implementazione

Codice SQL per instanziare il database

```
# ==============================TABLES==============================#


CREATE SCHEMA ASL;

CREATE TABLE ASL.paziente(
   CF_Paziente CHAR(16) PRIMARY KEY,
   Nome_Paziente VARCHAR (15) NOT NULL,
   Cognome_Paziente VARCHAR (15) NOT NULL,
   Indirizzo VARCHAR (30),
   Luogo VARCHAR (30),
   Data DATE,
   Telefono VARCHAR(10),
   Mail VARCHAR(30),
   Cellulare VARCHAR(10)
   );

CREATE TABLE ASL.esame(
   Codice_Esame CHAR(5) PRIMARY KEY,
   Nome_Esame VARCHAR (30) NOT NULL,
   Costo FLOAT NOT NULL
   );

CREATE TABLE ASL.prenotazione(
   Codice_Prenotazione CHAR(8) PRIMARY KEY,
   Paziente CHAR(16) NOT NULL,
     FOREIGN KEY (Paziente) REFERENCES ASL.paziente(CF_Paziente) ON DELETE CASCADE ON UPDATE
CASCADE
   );

CREATE TABLE ASL.ospedale(
   Codice_Ospedale CHAR(4) PRIMARY KEY,
   Nome_Ospedale VARCHAR(20) NOT NULL,
   Indirizzo VARCHAR(30)
   );

CREATE TABLE ASL.reparto(
   Codice_Reparto CHAR(5),
   Ospedale CHAR(4),
   Nome_Reparto VARCHAR(20) NOT NULL,
   Num_Telefono CHAR(10),
   PRIMARY KEY (Codice_Reparto, Ospedale),
      FOREIGN KEY (Ospedale) REFERENCES ASL.ospedale(Codice_Ospedale) ON DELETE CASCADE ON
UPDATE CASCADE
   );

CREATE TABLE ASL.medico(
   CF_Medico CHAR(16) PRIMARY KEY,
   Nome_Medico VARCHAR(15) NOT NULL,
   Cognome VARCHAR(15) NOT NULL,
   Indirizzo VARCHAR(30),
   Reparto CHAR(5) NOT NULL,
   Ospedale CHAR(4) NOT NULL,
```

```
      FOREIGN KEY (Reparto,Ospedale) REFERENCES ASL.reparto(Codice_Reparto,Ospedale) ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE TABLE ASL.laboratorio(
  Codice_Laboratorio CHAR(5),
  Ospedale CHAR(4),
  Nome_Laboratorio VARCHAR(25) NOT NULL,
  Piano INT,
  Stanza INT,
  PRIMARY KEY (Codice_Laboratorio, Ospedale),
     FOREIGN KEY (Ospedale) REFERENCES ASL.ospedale(Codice_Ospedale) ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE TABLE ASL.esamereale(
  Cod_Prenotazione CHAR (8),
  Tipo_Esame CHAR(5),
  Data Date,
  Ora TIME,
  Urgenza VARCHAR(15),
  Parametri VARCHAR (30),
  Laboratorio CHAR (5) NOT NULL,
  Ospedale CHAR(4) NOT NULL,
  Medico CHAR (16) NOT NULL,
  Diagnosi VARCHAR (50),
   UNIQUE (Cod_Prenotazione,Tipo_Esame,Data), #un esame non può essere ripetuto nello stesso giorno dallo stesso paziente.
  UNIQUE (Data,Ora,Medico), #un medico non può effettuare due esami contemporaneamente.
  UNIQUE (Data,Ora,Laboratorio,Ospedale), #in un laboratorio non posso effettuare due esami contemporaneamente
  PRIMARY KEY (Cod_Prenotazione, Tipo_Esame, Data, Ora),
     FOREIGN KEY (Cod_Prenotazione) REFERENCES ASL.prenotazione(Codice_Prenotazione) ON DELETE CASCADE ON UPDATE CASCADE,
      FOREIGN KEY (Laboratorio,Ospedale) REFERENCES ASL.laboratorio(Codice_Laboratorio,Ospedale) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (Medico) REFERENCES ASL.medico(CF_Medico) ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE TABLE ASL.primario(
  CF_Primario CHAR(16) PRIMARY KEY,
  Reparto_Primario CHAR(5) NOT NULL,
  Ospedale CHAR(4) NOT NULL,
  UNIQUE (Reparto_Primario,Ospedale), #un reparto ospedaliero non può avere due o più primari
   FOREIGN KEY (CF_Primario) REFERENCES ASL.medico(CF_Medico) ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (Reparto_Primario,Ospedale) REFERENCES ASL.reparto(Codice_Reparto,Ospedale) ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE TABLE ASL.volontario(
  CF_Volontario CHAR(16) PRIMARY KEY,
  Associazione VARCHAR(50),
   FOREIGN KEY (CF_Volontario) REFERENCES ASL.medico(CF_Medico) ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE TABLE ASL.specializzazione(
  Nome_Specializzazione VARCHAR(50) PRIMARY KEY
  );
```

```
CREATE TABLE ASL.possiede(
   Primario CHAR(16),
   Specializzazione VARCHAR(50),
   PRIMARY KEY (Primario, Specializzazione),
     FOREIGN KEY (Primario) REFERENCES ASL.primario(CF_Primario) ON DELETE CASCADE ON UPDATE
CASCADE,
      FOREIGN KEY (Specializzazione) REFERENCES ASL.specializzazione(Nome_Specializzazione) ON DELETE
CASCADE ON UPDATE CASCADE
   );

CREATE TABLE ASL.responsabilelaboratorio(
   Medico_Responsabile CHAR(16),
   Laboratorio CHAR(5),
   Ospedale CHAR (4),
   UNIQUE (Laboratorio,Ospedale), #un laboratorio deve avere un solo responsabile
   PRIMARY KEY (Medico_Responsabile,Laboratorio,Ospedale),
     FOREIGN KEY (Medico_Responsabile) REFERENCES ASL.medico(CF_Medico) ON DELETE CASCADE ON
UPDATE CASCADE,
       FOREIGN KEY (Laboratorio,Ospedale) REFERENCES ASL.laboratorio(Codice_Laboratorio, Ospedale) ON
DELETE CASCADE ON UPDATE CASCADE
   );

CREATE TABLE ASL.responsabileospedale(
   Medico_Responsabile CHAR(16),
   Ospedale CHAR (4) UNIQUE, #un ospedale deve avere un solo responsabile
   PRIMARY KEY (Medico_Responsabile,Ospedale),
     FOREIGN KEY (Medico_Responsabile) REFERENCES ASL.medico(CF_Medico) ON DELETE CASCADE ON
UPDATE CASCADE,
       FOREIGN KEY (Ospedale) REFERENCES ASL.ospedale(Codice_Ospedale) ON DELETE CASCADE ON
UPDATE CASCADE
   );









   # ================================POPULATE================================#


INSERT INTO ASL.paziente
       VALUES ("MNTVTR97T57I493M","Vittoria","Mount","Via scopigliette,21","SCAMPITELLA (AV)","1997-12-
17","0775408976","mntvtr97t57i493m@mailstop.it","3311456789"),
        ("RVSNZT92T60I089W","Nunziata","Rivas","Via roma,13","SAN PIETRO AL TANAGRO (SA)","1992-12-
20","0311345678","cuchurivas@libero,it","3546756129"),
                  ("ZNGSBN98P28I561G","Sabino","Zinghini","via pellicano,23","SECUGNAGO (LO)","1998-09-
28","0775678956","sabizinghi@hmail.com","3318978654"),
      ("DNGGDE79R21G943W","Egidio","Donegaglia","via della questura,2","POVE DEL GRAPPA (VI)","1979-10-
21","019876546","egidonaga@outlook.com","3771234987"),
           ("DNIDTT79B42G893V","Diletta","Dione","via accorciatoia,27","PORTA CARRATICA (PT)","1979-02-
02","0987456734","dnidtt79b42g893v@mailstop.it","3456787432"),
   ("SNNSNN96E52F557A","Osanna","Sannullo","via libertà,13","SANT'ARCANGELO TRIMONTE (BN)","1996-05-
12","0812897654","sannullolib@libero.it","3335678342"),
            ("GNSRFL92B09I022L","Raffaele","Giansetto","viale ippocrate,32","SAN MASSIMO (VR)","1992-02-
09","0986452314","rafygian@tiscali.it","3456767687"),
```

```
("CRRCDD69B62G024R","Candida","Carrabotta","via mastruccia,2","OLGIA (NO)","1969-02-22","0956234567","candida69@libero.it","3125676543"),
("PTRMRZ85E69H447Q","Marzia","Pietrantoni","via cavour,8","ROCCASPARVERA (CN)","1985-05-29","0775678765","stonemarz@gmail.com","3456123453"),
("TRMSNT87B64H597L","Santa","Taramasco","via aldo moro,43","ROVATE (VA)","1987-02-24","0912345323","terasar44@libero.it","3456576879"),
("LMBGVF60H44F273C","Genoveffa","Lambagi","via marittima,12","MOIANA (CO)","1960-06-04","0986090807","geno1518@hotmail.it","3314565876"),
("DGNDNI78R45G864Y","Diana","Digangi","via aristotele,12","PONTIDA (BG)","1978/10/05","0543564578","digaghana@gmail.com","3459876123"),
("MLPLNZ70L24G144J","Lorenzo","Melponti","viale augusto,8","ORTOVERO (SV)","1970-07-24","0555345610","meldru7@tor.it","3934565456"),
("SNGLSN48L20E252P","Alessandro","Asnago","via lucciola,2","GUASILA (CA)","1948-07-20","01564532","alemanas1@alice.it","3214593457"),
("BRGVLR81B23H064W","Valerio","Borgianini","via tornabene,5","PRIMANO (FM)","1981-02-23","024567865","borgvale@tim.it","3435678123");

INSERT INTO ASL.esame
VALUES ("01543","Radiografia",30.45), ("76453","TAC",50.12), ("46763","Elettrocardiogramma",65.87), ("34871","Elettroencefalogramma",120.30),
("46098","Analisi del Sangue",15.09), ("01009","MOC",45.98), ("02031","PAP-Test",34.65), ("99654","Vaccino",15.14), ("33456","Ecografia",21.65),
("30009","Ecodoppler",65.87), ("77001","Gastroscopia",25.65), ("77002","Colonscopia",35.54), ("00123","Analisi Tossicologica",65.23),
("21234","PHmetria",33.43), ("88001","Esame Prostata",12.90), ("12010","Emocromo",7.90), ("33990","Biopsia",72.94);

INSERT INTO ASL.ospedale
VALUES ("0423","San Raffaele","Via Polledrara,32"),("8675","San Francesco","Via della ricerca,15"), ("9090","Spaziani","Via Armando Fabi,12"),
("9231","San Benedetto","Via Chiappitto,32"),("1276","San Giovanni","Via dei tassi,15"),("5510","Umberto I","Via santa cecilia,2"),
("0992","San Martino","Via della libertà,5");

INSERT INTO ASL.reparto
VALUES ("00145","0423","Radiologia","0775403124"),("00146","0423","Pronto Soccorso","0775406578"), ("00147","0423","Pediatria","0775405690"),
("00148","0423","Ortopedia","0775408855"),("00149","0423","Oncologia","0775401143"), ("11001","8675","Oculistica","0334608976"),("11002","8675","Cardiologia","0334608811"),
("11003","8675","Ortopedia","0334606523"),("11004","8675","Radiologia","0334609965"), ("10650","9090","Cardiologia","0265783456"),("10651","9090","Pediatria","0265785690"),
("10652","9090","Radiologia","0265785512"),("10653","9090","Gastroenterologia","0265789012"), ("55101","9231","Radiologia","0711325678"),("55102","9231","Pediatria","0711325671"),
("55103","9231","Oncologia","0711321195"),("55104","9231","Pronto Soccorso","0711322376"), ("55105","9231","Oculistica","0711326677"),("55106","9231","Neurologia","0711325512"),
("55107","9231","Urologia","0711326610"),("21300","5510","Pediatria","0345651277"), ("21301","5510","Radiologia","0345652388"),("21302","5510","Oncologia","0345651092"),
("21303","5510","Neurologia","0345655472"),("21504","5510","Oculistica","0345651145"), ("21505","5510","Ortopedia","0345654123"),("21506","5510","Odontoiatria","0345652134"),
("13130","0992","Radiologia","0980801277"),("13131","0992","Ortopedia","0980807263"), ("13132","0992","Oncologia","0980809812"),("13133","0992","Odontoiatria","0980801265"),
("13134","0992","Neurologia","0980801265"),("13135","0992","Oculistica","0980801380"), ("02200","1276","Pediatria","0140985680"),("02201","1276","Cardiologia","0140986523");

INSERT INTO ASL.laboratorio
VALUES ("05132","0423","Laboratorio di Radiologia","2","8"),("05133","0423","Centro Analisi","3","18"), ("05134","0423","Centro Esami","0","4"),
("77100","8675","Centro Radiologia","0","7"),("77101","8675","Analisi","2","16"),("77102","8675","Laboratorio Generico","0","4"),
```

```
("13005","9090","Analisi","3","15"),("13006","9090","Raggi X","2","10"),("13007","9090","Medicina Generale","0","3"),
("22880","9231","Lab. Analisi 1","4","40"),("22881","9231","Lab. Radiologia","2","23"),("22882","9231","Lab. Generico 1","0","4"),
("22883","9231","Lab. Analisi 2","4","41"),("22884","9231","Lab. Generico 2","0","5"),("03001","1276","Centro Analisi","0","9"),
("03002","1276","Laboratorio di Radiologia","2","16"),("03003","1276","Laboratorio Generale","1","12"),("99000","5510","Laboratorio Analisi","2","23"),
("99001","5510","Laboratorio di Radiologia","2","24"),("99002","5510","Laboratorio Esami","2","25"),("22110","0992","Analisi","1","15"),
("22111","0992","Raggi X","3","32"),("22112","0992","Centro Esami","0","4");

INSERT INTO ASL.medico
VALUES ("PCVNDM80B23G969Z","Nicodemo","Pacuvio","Via della libertà,15","00145","0423"),("MTTNSI69L54G051I","Ines","Matteozzi","Viale della repubblica,1","00145","0423"),
("RSTNST84L44H358G","Onesta","Rustea","Via polledrara,18","00145","0423"),("CTCDCE73S02G430T","Decio","Coticchia","via scozzarella,7","00146","0423"),
("RGNNVS81D57M279N","Nives","Reganati","viale ippocrate,2","00146","0423"),("PCCMRT74R63G515U","Marta","Picicco","via marittima,29","00146","0423"),
("FFADGS67A07F839N","Adalgiso","Alafa","via carnevale,16","00147","0423"),("RNZLCA70M15G145C","Alceo","Aranzi","piazza san marco,13","00147","0423"),
("PRTPTR68S16F994T","Pietro","Pratesi","Via sant'anna","00147","0423"),("GRSFRC84R56H397U","Federica","Grossole","via braga,11","00147","0423"),
("FRCLVE67D48F859G","Elvia","Fracelio","via della carta,18","00148","0423"),("FSCGZE64B41F556A","Egizia","Fasce","via cavour,3","00148","0423"),
("SNGMHL76R30G693W","Michele","Sangemi","via mastruccia,12","00149","0423"),("SGMCRI67T01F919I","Ciro","Sigimbosco","via aldo moro,2","00149","0423"),
("RCTNLL72T64G352T","Novella","Ricetti","via della cultura,19","11001","8675"),("ZPPVVN68A30F934V","Viviano","Zappellini","Via cicogna,4","11001","8675"),
("MRRLDI68D43F946Z","Ilda","Marrancone","viale cesare,7","11002","8675"),("CHNLEO68H12F966D","Leo","Cheni","via eduardo de filippo,5","11002","8675"),
("PPCLNU81M60H100X","Luna","Papucci","via vittorio de sica,4","11003","8675"),("SCRBFC64L18F607O","Bonifacio","Scarpante","via santa cecilia,16","11003","8675"),
("CNTBGI80M24H016Y","Biagio","Caonetto","via danimarca,11","11004","8675"),("NLLGTN69P53G067H","Agostina","Anellucci","via roma,4","11004","8675"),
("SBESDR66B07F761A","Sandro","Seba","via sanremo,18","10650","9090"),("MTTTLN59B44F156R","Teolinda","Mattacheo","viale maramao,5","10650","9090"),
("ZNZZNE65M28F722O","Zeno","Zuanazzi","via preziosi,5","10651","9090"),("SNDBRT62M63F626P","Berta","Sandor","via santa colomba,5","10651","9090"),
("NNFLNI78A48G794T","Lina","Nunfris","via fogli,11","10651","9090"),("MMNSVR69C28G026N","Severo","Ammanato","via lucciola,6","10652","9090"),
("LVRGNI62E22F488L","Igino","Levorin","via vortice,5","10652","9090"),("FRDFDL67S18F915Y","Fedele","Fredo","via del cane,13","10653","9090"),
("LCRDLM68A19F932J","Adelmo","Alcuri","via santa colomba,7","55101","9231"),("GRSFRZ74M52G496O","Fabrizia","Gorospe","via armando fabi,7","55101","9231"),
("GLLRSL80A70G963I","Ersilia","Gellini","Via della scienza,5","55102","9231"),("MRTLNI68L68F976G","Ilenia","Martusciello","via colle traiano","55102","9231"),
("NCLMZU77T20G789S","Muzio","Nocellese","viale dello sport,6","55102","9231"),("FRBRMS83D18H250U","Ermes","Farabbi","via della ricerca,9","55103","9231"),
("MBRLVC78S50G869P","Ludovica","Ombroneschi","via del mare,8","55103","9231"),("TLLTDR74L17G489D","Teodoro","Tellona","via marco polo,19","55104","9231"),
("TRZSST54M42E769N","Sebastiana","Terzic","via moratti,15","55104","9231"),("BNCMSA76S12G697A","Amos","Bencresciuto","via novella,20","55104","9231"),
("FRNLSE66A61F757T","Elisa","Frinu","via popolare,11","55105","9231"),("DPTDDR58L70F111Y","Desideria","De Pietro","via platone,4","55105","9231"),
("NPTMHL76P24G684I","Michele","Nepitali","via tornabene,9","55106","9231"),("MRCGNZ67M54F890X","Ignazia","Marcino","via della cultura,9","55106","9231"),
("CRNCDD65D46F686N","Candida","Carniselli","via milano,6","55107","9231"),("RBLPRM56L04E932J","Primo","Rabelli","via dei tassi,6","21300","5510"),
```

```
("PLCMNC69R60G076R","Monica","Pulcinari","via delle segrete,7","21300","5510"),
("TRVDBR57A41E977F","Debora","Trevisoli","via accorciatoia,21","21300","5510"),
("RPMNCC59C64F169F","Nuccia","Ripamonti","via maremma,14","21301","5510"),
("LCCMLE83E44H253V","Emilia","Lucci","via porta nuova,12","21301","5510"),
("MRMGNS69E44G036M","Agnese","Amrami","viale italia,14","21302","5510"),
("GRNFRZ74M49G495J","Fabrizia","Gorner","via siracusa,4","21302","5510"),
("MSTTEA58R55F129X","Tea","Mastrobiso","via paciotti,8","21303","5510"),
("ZNDVGL67R56F906Z","Virgilia","Zandigiacomo","via gaucci,8","21303","5510"),
("VRCLVO83R66H296Y","Olivia","Vercellono","Via Chiappitto,9","21504","5510"),
("RBLQNT71T23G263D","Quinto","Rebellati","via pepe,31","21504","5510"),
("RPPMSM74E23G475G","Massimiliano","Roppo","via conte,6","21505","5510"),
("GLLFRM69L09G050N","Fermo","Gallichi","via salina,21","21505","5510"),
("CVGCMR66S50F823U","Casimira","Cavigliani","via cristianesimo,5","21506","5510"),
("MNTMRA73S24G434F","Mario","Monterenzi","via padova,7","21506","5510"),
("LTRFTT55H63E841O","Fioretta","Ielitro","via imperiale,14","13130","0992"),
("LBZGIO63D65F487U","Gioia","Lebez","via del mare,17","13130","0992"),
("GLNGTN65L13F710F","Gastone","Gualina","Via montemario,8","13131","0992"),
("MNDGLL66M51F802C","Guglielma","Mandalino","viale del mercato,12","13131","0992"),
("PRGPTR68R16F990I","Pietro","Paglioli","viale degli ulivi,16","13132","0992"),
("SCHLRN79E59G906K","Lorena","Schoss","via della pace, 1","13132","0992"),
("CHRCTN69D10G030P","Costante","Chirigoni","Via akihabara,12","13133","0992"),
("BDSDNT60E16F267D","Donetta","Baldassarre","via coinciliazione,12","13133","0992"),
("PGLLRC69T48G089B","Alberica","Pagliucci","via del martire,1","13134","0992"),
("BRBTTL63B12F464K","Attilio","Breber","via mastruccia,19","13134","0992"),
("NZZLTZ76H61G663Q","Letizia","Nazzarelli","viale innocenzo,17","13135","0992"),
("LNTLDE84A66H317A","Elda","Lanteri","via borsellino,32","13135","0992"),
("GDDNZE45R41E006H","Enza","Gadda","viale ippocrate,17","02200","1276"),
("PRRMMM68H50F964B","Mimma","Porriello","via bitta,16","02201","1276");

INSERT INTO ASL.specializzazione
VALUES ("Logopedia"),("Chirurgia Vascolare"),("Medicina legale"),("Ortopedia"),("Dermatologia"),
("Epidemiologia"),("Odontoiatria"),("Neuroscienze"),("Traumatologia"),
("Cardiologia"),("Cardiochirurgia"),("Medicina sportiva"),("Pediatria"),("Infettivologia"),("Dietetica"),("Igiene e medicina preventiva"),("Neuroriabilitazione"),
("Endocrinologia"),("Reumatologia");

INSERT INTO ASL.prenotazione
VALUES ("00000000","MNTVTR97T57I493M"),("00000001","RVSNZT92T60I089W"),
("00000002","ZNGSBN98P28I561G"),("00000003","DNGGDE79R21G943W"),
("00000004","DNIDTT79B42G893V"),("00000005","SNNSNN96E52F557A"),
("00000006","GNSRFL92B09I022L"),("00000007","CRRCDD69B62G024R"),("00000008","PTRMRZ85E69H447Q"),
("00000010","TRMSNT87B64H597L"),("00000011","LMBGVF60H44F273C"),
("00000012","DGNDNI78R45G864Y"),("00000013","MLPLNZ70L24G144J"),("00000014","SNGLSN48L20E252P"),
("00000016","CRRCDD69B62G024R"),("00000017","BRGVLR81B23H064W"),
("00000018","DNIDTT79B42G893V"),("00000019","SNGLSN48L20E252P"),("00000020","MNTVTR97T57I493M"),
("00000022","BRGVLR81B23H064W"),("00000023","GNSRFL92B09I022L"),
("00000024","TRMSNT87B64H597L"),("00000025","RVSNZT92T60I089W"),("00000026","GNSRFL92B09I022L"),
("00000009","BRGVLR81B23H064W"),("00000015","RVSNZT92T60I089W"),
("00000021","DNGGDE79R21G943W"),("00000027","LMBGVF60H44F273C"),
("00000028","TRMSNT87B64H597L");

INSERT INTO ASL.esamereale
VALUES ("00000000","77001","2019-03-21","15:30","codice bianco","a5iu8f34gh7g","05132","0423","NCLMZU77T20G789S","parametri ok"),
("00000001","33990","2019-05-12","12:00","codice verde","s6jh32jv4g9","22883","9231","RSTNST84L44H358G","NULL"),
("00000002","46098","2018-12-06","14:00","codice giallo","kdne7skn3","22883","9231","GLNGTN65L13F710F","riscontrata patologia, necessari ulteriori esami"),
("00000003","01543","2019-10-08","09:30","codice verde","dj238fjcsad","13007","9090","FFADGS67A07F839N","parametri ok"),
```

```
("00000004","76453","2019-03-01","11:30","codice rosso","d39jc8hf8a0","99002","5510","LVRGNI62E22F488L","NULL"),
("00000005","21234","2018-02-14","09:45","codice verde","a34ng83nfas","13007","9090","NCLMZU77T20G789S","nessun problema riscontrato"),
("00000006","34871","2018-01-05","10:15","codice verde","c9wnf73nfda9","22883","9231","LVRGNI62E22F488L","necessario un intervento specifico"),
("00000007","76453","2017-12-01","16:05","codice giallo","c83hdfaijf93","77102","8675","RSTNST84L44H358G","NULL"),
("00000008","02031","2018-06-13","18:00","codice verde","f39hdfa8hf3oi","13007","9090","GLNGTN65L13F710F","riscontrato un problema,necessario intervento"),
("00000009","33990","2016-05-09","12:10","codice giallo","gfh83ha9f0k5","22883","9231","NCLMZU77T20G789S","nessun problema rilevato"),
("00000010","02031","2019-07-10","08:15","codice giallo","h38hv8as9","03003","1276","FFADGS67A07F839N","NULL"),
("00000011","33456","2018-02-06","07:30","codice rosso","cn983hfda93","22883","9231","FRCLVE67D48F859G","nessun problema,ma consigliata ulteriore visita"),
("00000012","01543","2015-04-09","14:30","codice verde","dfh3jfaw303","13005","9090","SBESDR66B07F761A","NULL"),
("00000013","77001","2015-12-14","12:45","codice bianco","fh39fj93jfi","03003","1276","CRNCDD65D46F686N","necessaria una radiografia"),
("00000014","46098","2018-02-14","15:00","codice giallo","df3fjh9hj3jg","77100","8675","MSTTEA58R55F129X","NULL"),
("00000015","46763","2019-03-23","08:15","codice bianco","djh3jv9sedj3","22112","0992","MSTTEA58R55F129X","NULL"),
("00000016","21234","2019-05-07","12:10","codice rosso","ffh3jhf9as3jg9","05134","0423","MRMGNS69E44G036M","NULL"),
("00000017","02031","2010-09-14","15:30","codice giallo","fn3nfiasdjf33","22112","0992","SBESDR66B07F761A","NULL"),
("00000018","33456","2018-02-28","08:15","codice giallo","mbnf239f94hf8","77102","8675","PRTPTR68S16F994T","riscontrati valori anomali"),
("00000019","46763","2012-06-09","18:00","codice bianco","mdfb83fa9034f","13005","9090","MSTTEA58R55F129X","valori nella norma"),
("00000020","77001","2011-12-31","12:00","codice rosso","bnndsfj308gth","03003","1276","MRMGNS69E44G036M","NULL"),
("00000021","00123","2016-09-16","09:30","codice bianco","bnwsnf93ah8hg","13006","9090","MSTTEA58R55F129X","consigliata cura riabilitativa"),
("00000022","01543","2018-02-09","15:30","codice bianco","bmfn98h38hfaf","05133","0423","NCLMZU77T20G789S","NULL"),
("00000023","46098","2013-05-08","09:30","codice rosso","mfan398nfs9u","13005","9090","LCRDLM68A19F932J","condigliato ulteriore controllo medico"),
("00000024","33456","2016-07-11","09:30","codice giallo","fn8sadherw8h","05134","0423","MSTTEA58R55F129X","parametri regolari"),
("00000025","77001","2015-10-15","12:10","codice bianco","jg9j943j9ejf9j","77102","8675","FRCLVE67D48F859G","nessuna patologia rilevata"),
("00000026","30009","2019-05-07","12:00","codice bianco","gnwehr8jas3fg","05132","0423","SBESDR66B07F761A","NULL"),
("00000027","34871","2016-05-09","18:00","codice verde","dwenmr3jn8fdyg","77100","8675","CRNCDD65D46F686N","controllo ok"),
("00000028","30009","2016-05-09","15:30","codice verde","dingbnb49df84j","22112","0992","MSTTEA58R55F129X","NULL"),
("00000012","01009","2018-02-11","12:10","codice rosso","bbba3h8fh834","05133","0423","PLCMNC69R60G076R","nessun problema rilevato"),
("00000009","88001","2019-05-11","08:15","codice verde","mjqa938gydfg34","77101","8675","PRTPTR68S16F994T","NULL"),
("00000019","01009","2019-05-07","12:00","codice verde","z934j988fj84hg","22883","9231","MBRLVC78S50G869P","NULL"),
("00000007","30009","2018-02-18","18:00","codice verde","ogjioj49fds873","77101","8675","LCRDLM68A19F932J","NULL"),
("00000028","34871","2019-05-01","15:30","codice giallo","m876fdhf934fj9","05132","0423","PLCMNC69R60G076R","NULL"),
```

```
                                                ("00000021","88001","2018-08-01","09:30","codice
verde","j834hfj93fdsf0e","77100","8675","PRTPTR68S16F994T","parametri regolari"),
                                                ("00000020","33456","2017-03-04","10:15","codice
giallo","564fg34rtg6hj6","22112","0992","FRCLVE67D48F859G","consigliata una ulteriore visita"),
                                                ("00000000","02031","2016-09-18","15:00","codice
bianco","d3dasfg45ytgs3","03003","1276","GLNGTN65L13F710F","nessun problema"),
                                                ("00000000","46098","2015-05-21","12:00","codice
bianco","bv4trws34sdcg4","13007","9090","CHRCTN69D10G030P","diagnosticata patologia"),
                                                ("00000001","33456","2014-12-31","08:15","codice
giallo","dcv43w5tsg546y","99002","5510","PRTPTR68S16F994T","NULL"),
                                                ("00000008","46763","2013-08-09","08:15","codice
verde","cv4fgt32rfatgy4","77100","8675","GLLRSL80A70G963I","necessario ulteriore controllo"),
                                                ("00000008","46098","2012-09-11","12:00","codice
rosso","bvrty4w5wsr3234f","03003","1276","PRTPTR68S16F994T","rilevato problema"),
                                                ("00000009","02031","2011-01-10","09:30","codice
giallo","cvvgh432wfr3234","13005","9090","FRCLVE67D48F859G","NULL"),
                                                ("00000015","88001","2010-04-12","08:15","codice
bianco","dgf3raf4rteytsdg","22112","0992","GLLRSL80A70G963I","nessun problema rilevato"),
                                                ("00000012","46763","2018-10-13","09:30","codice
verde","f3fadf45w56dsft4","13007","9090","RCTNLL72T64G352T","necessario ulteriore controllo");

INSERT INTO ASL.primario
                        VALUES      ("MTTNSI69L54G051I","00145","0423"),("RGNNVS81D57M279N","00146","0423"),
("GRSFRC84R56H397U","00147","0423"),("FSCGZE64B41F556A","00148","0423"),
                                ("SNGMHL76R30G693W","00149","0423"),("ZPPVVN68A30F934V","11001","8675"),
("MRRLDI68D43F946Z","11002","8675"),("PPCLNU81M60H100X","11003","8675"),
                                 ("NLLGTN69P53G067H","11004","8675"),("SBESDR66B07F761A","10650","9090"),
("NNFLNI78A48G794T","10651","9090"),("MMNSVR69C28G026N","10652","9090"),
                                 ("FRDFDL67S18F915Y","10653","9090"),("LCRDLM68A19F932J","55101","9231"),
("MRTLNI68L68F976G","55102","9231"),("FRBRMS83D18H250U","55103","9231"),
                                 ("TRZSST54M42E769N","55104","9231"),("FRNLSE66A61F757T","55105","9231"),
("NPTMHL76P24G684I","55106","9231"),("CRNCDD65D46F686N","55107","9231"),
                                ("PLCMNC69R60G076R","21300","5510"),("LCCMLE83E44H253V","21301","5510"),
("GRNFRZ74M49G495J","21302","5510"),("MSTTEA58R55F129X","21303","5510"),
                                ("VRCLVO83R66H296Y","21504","5510"),("RPPMSM74E23G475G","21505","5510"),
("MNTMRA73S24G434F","21506","5510"),("LBZGIO63D65F487U","13130","0992"),
                                 ("MNDGLL66M51F802C","13131","0992"),("PRGPTR68R16F990I","13132","0992"),
("CHRCTN69D10G030P","13133","0992"),("PGLLRC69T48G089B","13134","0992"),
                                ("NZZLTZ76H61G663Q","13135","0992"),("GDDNZE45R41E006H","02200","1276"),
("PRRMMM68H50F964B","02201","1276");

INSERT INTO ASL.possiede
                        VALUES      ("MTTNSI69L54G051I","Cardiochirurgia"),("MTTNSI69L54G051I","Cardiologia"),
("RGNNVS81D57M279N","Odontoiatria"),("GRSFRC84R56H397U","Ortopedia"),
                            ("FSCGZE64B41F556A","Reumatologia"),("FSCGZE64B41F556A","Medicina Sportiva"),
("SNGMHL76R30G693W","Neuroscienze"),("ZPPVVN68A30F934V","Dietetica"),
                        ("MRRLDI68D43F946Z","Dermatologia"),("MRRLDI68D43F946Z","Chirurgia Vascolare"),
("PPCLNU81M60H100X","Cardiologia"),("NLLGTN69P53G067H","Odontoiatria"),
                                ("SBESDR66B07F761A","Cardiochirurgia"),("NNFLNI78A48G794T","Dermatologia"),
("NNFLNI78A48G794T","Igiene e medicina preventiva"),("MMNSVR69C28G026N","Pediatria"),
                                 ("FRDFDL67S18F915Y","Reumatologia"),("LCRDLM68A19F932J","Infettivologia"),
("MRTLNI68L68F976G","Cardiologia"),("FRBRMS83D18H250U","Logopedia"),
                                    ("TRZSST54M42E769N","Dietetica"),("FRNLSE66A61F757T","Traumatologia"),
("NPTMHL76P24G684I","Logopedia"),("CRNCDD65D46F686N","Infettivologia"),
                            ("PLCMNC69R60G076R","Reumatologia"),("LCCMLE83E44H253V","Medicina Sportiva"),
("LCCMLE83E44H253V","Cardiologia"),("GRNFRZ74M49G495J","Dermatologia"),
                                    ("MSTTEA58R55F129X","Logopedia"),("VRCLVO83R66H296Y","Pediatria"),
("RPPMSM74E23G475G","Chirurgia Vascolare"),("MNTMRA73S24G434F","Dietetica"),
                                    ("MNTMRA73S24G434F","Cardiologia"),("LBZGIO63D65F487U","Pediatria"),
("MNDGLL66M51F802C","Reumatologia"),("PRGPTR68R16F990I","Medicina Sportiva"),
```

```
                                ("CHRCTN69D10G030P","Dermatologia"),("PGLLRC69T48G089B","Reumatologia"),
("PGLLRC69T48G089B","Dermatologia"),("NZZLTZ76H61G663Q","Logopedia"),
  ("GDDNZE45R41E006H","Pediatria"),("PRRMMM68H50F964B","Cardiologia");

INSERT INTO ASL.volontario
            VALUES ("SNDBRT62M63F626P","Croce Rossa Italiana"),("PCCMRT74R63G515U","AVIS"),
("LVRGNI62E22F488L","ANPAS"),("GLLRSL80A70G963I","AVER"),
  ("RPMNCC59C64F169F","Associazione Italiana Volontari"),("BRBTTL63B12F464K","SILOE");

INSERT INTO ASL.responsabilelaboratorio
                VALUES ("RSTNST84L44H358G","05132","0423"),("CTCDCE73S02G430T","05133","0423"),
("RNZLCA70M15G145C","05134","0423"),("FRCLVE67D48F859G","77100","8675"),
                ("SGMCRI67T01F919I","77101","8675"),("PPCLNU81M60H100X","77102","8675"),
("SBESDR66B07F761A","13005","9090"),("ZNZZNE65M28F722O","13006","9090"),
                ("MMNSVR69C28G026N","13007","9090"),("GLLRSL80A70G963I","22880","9231"),
("FRBRMS83D18H250U","22881","9231"),("TLLTDR74L17G489D","22882","9231"),
                ("FRNLSE66A61F757T","22883","9231"),("NPTMHL76P24G684I","22884","9231"),
("RPPMSM74E23G475G","03001","1276"),("LBZGIO63D65F487U","03002","1276"),
                ("CHRCTN69D10G030P","03003","1276"),("RBLQNT71T23G263D","99000","5510"),
("TRVDBR57A41E977F","99001","5510"),("NPTMHL76P24G684I","99002","5510"),
                ("TLLTDR74L17G489D","22110","0992"),("MMNSVR69C28G026N","22111","0992"),
("CTCDCE73S02G430T","22112","0992");

INSERT INTO ASL.responsabileospedale
    VALUES ("FRCLVE67D48F859G","0423"),("MRRLDI68D43F946Z","8675"),("FRDFDL67S18F915Y","9090"),
("MRTLNI68L68F976G","9231"),
  ("PRRMMM68H50F964B","1276"),("GRNFRZ74M49G495J","5510"),("CHRCTN69D10G030P","0992");









  # ============================STORED PROCEDURES============================#


# Operazioni sui Pazienti

CREATE PROCEDURE ASL.paziente_cerca (IN CodiceFiscale CHAR(16))
SELECT *
FROM ASL.paziente
WHERE CF_Paziente = CodiceFiscale;

CREATE PROCEDURE ASL.mostra_pazienti ()
SELECT *
FROM ASL.paziente;

CREATE PROCEDURE ASL.paziente_aggiungi (IN CF CHAR(16), IN nome VARCHAR(15), IN cognome
VARCHAR(15),IN indirizzo VARCHAR(30),IN Luogo VARCHAR(30),IN data DATE, IN Telefono VARCHAR(10),IN
Mail VARCHAR(30),IN Cellulare VARCHAR(10)) #aggiunge un paziente al DB
INSERT INTO ASL.paziente
VALUES (CF,nome,cognome,indirizzo,luogo,data,telefono,mail,cellulare);

CREATE PROCEDURE ASL.paziente_cancella(IN CF CHAR(16)) #cancella un paziente dal DB
DELETE FROM ASL.paziente
WHERE CF_paziente = CF;
```

```
CREATE PROCEDURE ASL.paziente_modifica (IN CF CHAR(16),IN n_indirizzo VARCHAR(30), IN n_Telefono VARCHAR(10),IN n_Mail VARCHAR(30),IN n_Cellulare VARCHAR(10))
UPDATE ASL.paziente
SET Indirizzo = n_indirizzo,
   Telefono = n_Telefono,
   Mail = n_Mail,
   Cellulare = n_cellulare
WHERE CF = CF_Paziente;
```

```
# Operazioni sulle Prenotazioni degli esami

CREATE PROCEDURE ASL.prenotazione_aggiungi (IN cod_prenotazione CHAR(8),IN cf_paziente CHAR(16))
INSERT INTO ASL.prenotazione
VALUES (cod_prenotazione,cf_paziente);

CREATE PROCEDURE ASL.prenotazione_cancella (IN cod_prenot CHAR(8))
DELETE FROM ASL.prenotazione
WHERE Codice_Prenotazione=cod_prenot;

CREATE PROCEDURE ASL.mostra_prenotazioni ()
SELECT *
FROM ASL.prenotazione ORDER BY Codice_Prenotazione;

CREATE PROCEDURE ASL.esamereale_aggiungi (IN cod_prenot CHAR(8), cod_esame CHAR (5), data DATE, ora TIME, urgenza VARCHAR(15), parametri VARCHAR(30), laboratorio CHAR(5), ospedale CHAR(4), medico CHAR(16), diagnosi VARCHAR(50)) #aggiunge  un esame reale relativo ad una prenotazione
INSERT INTO ASL.esamereale
VALUES (cod_prenot,cod_esame,data,ora,urgenza,parametri,laboratorio,ospedale,medico,diagnosi);

CREATE PROCEDURE ASL.esamereale_cancella (IN cod_prenot CHAR(8), cod_esame CHAR (5), data_ DATE, ora_ TIME)
DELETE FROM ASL.esamereale
WHERE (cod_prenot,cod_esame,data_,ora_) = (Cod_Prenotazione,tipo_esame,data,ora);

CREATE PROCEDURE ASL.esamereale_modifica_data (IN cod_prenot CHAR(8), IN esame CHAR(5),IN data DATE,IN ora TIME, nuova_data DATE, nuova_ora TIME)
UPDATE ASL.esamereale
SET data = nuova_data,
   ora = nuova_ora
WHERE (Cod_Prenotazione,Tipo_Esame,Data,Ora) = (cod_prenot,esame,data,ora);

CREATE PROCEDURE ASL.esamereale_inserisci_diagnosi (IN cod_prenot CHAR(8), cod_esame CHAR (5), data DATE, ora TIME, IN nuovi_parametri VARCHAR(30), IN nuova_diagnosi VARCHAR(50))
UPDATE ASL.esamereale
SET parametri = nuovi_parametri,
   diagnosi = nuova_diagnosi
WHERE (Cod_Prenotazione,Tipo_Esame,Data,Ora) = (cod_prenot,cod_esame,data,ora);

CREATE PROCEDURE ASL.esamereale_cerca (IN cod_prenot CHAR(8), cod_esame CHAR (5), data_ DATE, ora_ TIME)
SELECT Codice_Prenotazione AS Codice,Paziente,Tipo_Esame,Data,Ora,Urgenza,Parametri,Laboratorio,Ospedale,Medico,Diagnosi
FROM ASL.esamereale JOIN ASL.prenotazione ON esamereale.Cod_Prenotazione = prenotazione.Codice_Prenotazione
WHERE (cod_prenot,cod_esame,data,ora) = (Cod_Prenotazione,Tipo_Esame,Data,Ora);
```

```
CREATE PROCEDURE ASL.mostra_esamireali ()
SELECT Codice_Prenotazione AS Codice,Paziente,Tipo_Esame,Data,Ora,Urgenza,Parametri,Laboratorio,Ospedale,Medico,Diagnosi
FROM ASL.esamereale JOIN ASL.prenotazione ON esamereale.Cod_Prenotazione = prenotazione.Codice_Prenotazione;
```

Operazioni sugli Esami disponibili

```
CREATE PROCEDURE ASL.esame_aggiungi (IN cod_esame CHAR(5), IN nome_esame VARCHAR(30), costo FLOAT)
INSERT INTO ASL.esame
VALUES (cod_esame,nome_esame,costo);

CREATE PROCEDURE ASL.esame_cancella (IN cod_esame CHAR(5))
DELETE FROM ASL.esame
WHERE Codice_Esame = cod_esame;

CREATE PROCEDURE ASL.esame_modifica (IN cod_esame CHAR(5), IN nuovo_codice CHAR(5), IN nuovo_nome VARCHAR(30), IN nuovo_costo FLOAT)
UPDATE ASL.esame
SET codice_esame = nuovo_codice,nome_esame = nuovo_nome,costo = nuovo_costo
WHERE Codice_Esame = cod_esame;

CREATE PROCEDURE ASL.esame_cerca (IN cod_esame CHAR(5))
SELECT *
FROM ASL.esame
WHERE cod_esame = Codice_Esame;

CREATE PROCEDURE ASL.mostra_esami ()
SELECT *
FROM ASL.esame;
```

Operazioni sugli Ospedali

```
CREATE PROCEDURE ASL.ospedale_aggiungi (IN cod_osp CHAR(4), IN nome_osp VARCHAR(20), IN indirizzo VARCHAR(30))
INSERT INTO ASL.ospedale
VALUES (cod_osp,nome_osp,indirizzo);

CREATE PROCEDURE ASL.ospedale_cancella (IN cod_osp CHAR(4))
DELETE FROM ASL.ospedale
WHERE Codice_Ospedale = cod_osp;

CREATE PROCEDURE ASL.ospedale_cerca (IN cod_osp CHAR(4))
SELECT *
FROM ASL.ospedale
WHERE Codice_Ospedale = cod_osp;

CREATE PROCEDURE ASL.mostra_ospedali()
SELECT Codice_Ospedale AS Codice,Nome_Ospedale AS Nome,Indirizzo,Medico_Responsabile AS Responsabile
FROM ASL.ospedale
```

```
JOIN ASL.responsabileospedale ON ospedale.Codice_Ospedale = responsabileospedale.Ospedale;

CREATE PROCEDURE ASL.ospedale_modifica (IN cod_osp CHAR(4), IN nuovo_codice CHAR(4),IN nuovo_nome VARCHAR(20), IN nuovo_indirizzo VARCHAR(30))
UPDATE ASL.ospedale
SET Codice_Ospedale = nuovo_codice, Nome_Ospedale = nuovo_nome, Indirizzo = nuovo_indirizzo
WHERE Codice_Ospedale = cod_osp;




# Operazioni sui Laboratori

CREATE PROCEDURE ASL.laboratorio_aggiungi (IN cod_lab CHAR(5), IN cod_osp CHAR(4), IN nome VARCHAR(25), IN piano INT, IN stanza INT)
INSERT INTO ASL.laboratorio
VALUES (cod_lab,cod_osp,nome,piano,stanza);

CREATE PROCEDURE ASL.laboratorio_cancella (IN cod_lab CHAR(5), IN cod_osp CHAR(4))
DELETE FROM ASL.laboratorio
WHERE (Codice_Laboratorio,Ospedale) = (cod_lab,cod_osp);

CREATE PROCEDURE ASL.laboratorio_cerca (IN cod_lab CHAR(5), IN cod_osp CHAR(4))
SELECT *
FROM ASL.laboratorio
WHERE (Codice_Laboratorio,Ospedale) = (cod_lab,cod_osp);

CREATE PROCEDURE ASL.mostra_laboratori()
SELECT Codice_Laboratorio AS Laboratorio,laboratorio.Ospedale AS Ospedale,Nome_Laboratorio AS Nome,Piano,Stanza,Medico_Responsabile AS Responsabile
FROM ASL.laboratorio
JOIN ASL.responsabilelaboratorio ON laboratorio.Codice_Laboratorio = responsabilelaboratorio.Laboratorio
AND laboratorio.Ospedale = responsabilelaboratorio.Ospedale;

CREATE PROCEDURE ASL.laboratorio_modifica (IN cod_lab CHAR(5), IN cod_osp CHAR(4), IN nuovo_lab CHAR(5), IN nuovo_osp CHAR(4), IN nuovo_nome VARCHAR(25), IN nuovo_piano INT, IN nuova_stanza INT)
UPDATE ASL.laboratorio
SET Codice_Laboratorio = nuovo_lab,
   Ospedale = nuovo_osp,
   Nome_Laboratorio = nuovo_nome,
   Piano = nuovo_piano,
   Stanza = nuova_stanza
WHERE (Codice_Laboratorio,Ospedale) = (cod_lab,cod_osp);




# Operazioni sui Medici

CREATE PROCEDURE ASL.medico_aggiungi (IN cod_medico CHAR(16), nome_medico VARCHAR(15), cognome_medico VARCHAR(15), indirizzo VARCHAR(30), IN reparto CHAR(5),IN ospedale CHAR(4))
INSERT INTO ASL.medico
VALUES (cod_medico,nome_medico,cognome_medico,indirizzo,reparto,ospedale);

CREATE PROCEDURE ASL.medico_cancella (IN cod_medico CHAR(16))
DELETE FROM ASL.medico
WHERE CF_Medico = cod_medico;
```

```
CREATE PROCEDURE ASL.medico_modifica (IN cod_medico CHAR(16), nuovo_indirizzo VARCHAR(30), IN nuovo_reparto CHAR(5),IN nuovo_ospedale CHAR(4))
UPDATE ASL.medico
SET Indirizzo = nuovo_indirizzo,
   Reparto = nuovo_reparto,
   Ospedale = nuovo_ospedale
WHERE CF_Medico = cod_medico;

CREATE PROCEDURE ASL.nomina_resp_osp (IN cod_medico CHAR(16),IN ospedale CHAR(4))
INSERT INTO ASL.responsabileospedale
VALUES (cod_medico,ospedale);

CREATE PROCEDURE ASL.nomina_resp_lab (IN cod_medico CHAR(16), IN laboratorio CHAR(5), IN ospedale CHAR(4))
INSERT INTO ASL.responsabilelaboratorio
VALUES (cod_medico,laboratorio,ospedale);

CREATE PROCEDURE ASL.medico_cerca (IN cod_medico CHAR(16))
SELECT *
FROM ASL.medico
WHERE CF_Medico = cod_medico;

CREATE PROCEDURE ASL.mostra_medici ()
SELECT *
FROM ASL.medico;




# Operazioni sui Primari

CREATE PROCEDURE ASL.nomina_primario (IN cod_medico CHAR(16))
INSERT INTO ASL.primario (CF_Primario,Reparto_Primario,Ospedale)
SELECT CF_Medico,Reparto,Ospedale
FROM ASL.medico
WHERE CF_Medico = cod_medico;

CREATE PROCEDURE ASL.rimuovi_primario (IN cod_primario CHAR(16))
DELETE FROM ASL.primario
WHERE cod_primario = CF_Primario;

CREATE PROCEDURE ASL.nuova_specializzazione (IN nome VARCHAR(50))
INSERT INTO ASL.specializzazione (Nome_Specializzazione)
VALUES (nome);

CREATE PROCEDURE ASL.mostra_specializzazioni ()
SELECT *
FROM ASL.specializzazione;

CREATE PROCEDURE ASL.primario_agg_spec(IN CF_primario CHAR(16),IN spec VARCHAR(50))
INSERT INTO ASL.possiede
VALUES (CF_primario,spec);

CREATE PROCEDURE ASL.primario_rim_spec(IN CF_primario CHAR(16),IN spec VARCHAR(50))
DELETE FROM ASL.possiede
WHERE (CF_primario,spec) = (Primario,Specializzazione);

CREATE PROCEDURE ASL.primario_cerca (IN CodiceFiscale CHAR(16))
SELECT CF_Primario,Nome_Medico,Cognome,Indirizzo,medico.Reparto,medico.Ospedale
```

```
FROM ASL.primario
   JOIN ASL.medico ON primario.CF_Primario = medico.CF_Medico
WHERE CodiceFiscale = CF_Primario;

CREATE PROCEDURE ASL.mostra_primari ()
SELECT CF_Primario,Nome_Medico,Cognome,Indirizzo,medico.Reparto,medico.Ospedale
FROM ASL.primario
   JOIN ASL.medico ON primario.CF_Primario = medico.CF_Medico;

CREATE PROCEDURE ASL.primario_mostra_spec (IN CF CHAR(16))
SELECT *
FROM ASL.possiede
WHERE Primario = CF;




# Operazioni sui Volontari

CREATE PROCEDURE ASL.nomina_volontario (IN cod_medico CHAR(16),IN associazione VARCHAR(50))
INSERT INTO ASL.volontario
VALUES (cod_medico,associazione);

CREATE PROCEDURE ASL.rimuovi_volontario (IN cod_volontario CHAR(16))
DELETE FROM ASL.volontario
WHERE CF_Volontario = cod_volontario;

CREATE PROCEDURE ASL.volontario_cerca (IN CodiceFiscale CHAR(16))
SELECT CF_Volontario AS CF,Nome_Medico AS
Nome,Cognome,Indirizzo,medico.Reparto,medico.Ospedale,Associazione
FROM ASL.volontario
   JOIN ASL.medico ON volontario.CF_Volontario = medico.CF_Medico
WHERE CF_Volontario = CodiceFiscale;

CREATE PROCEDURE ASL.mostra_volontari ()
SELECT CF_Volontario AS CF,Nome_Medico AS
Nome,Cognome,Indirizzo,medico.Reparto,medico.Ospedale,Associazione
FROM ASL.volontario
   JOIN ASL.medico ON volontario.CF_Volontario = medico.CF_Medico;




# Operazioni sui Reparti

CREATE PROCEDURE ASL.reparto_aggiungi (IN cod_reparto CHAR(5), IN cod_ospedale CHAR(4), IN nome
VARCHAR(20), IN numero_telefono CHAR(10))
INSERT INTO ASL.reparto
VALUES (cod_reparto,cod_ospedale,nome,numero_telefono);

CREATE PROCEDURE ASL.reparto_modifica (IN cod_reparto CHAR(5), IN cod_ospedale CHAR(4),IN nuovo_codice
CHAR(5),IN nuovo_ospedale CHAR(4), IN nuovo_nome VARCHAR(20), IN nuovo_numero CHAR(10))
UPDATE ASL.reparto
SET Codice_Reparto = nuovo_codice,
   Ospedale = nuovo_ospedale,
   Nome_Reparto = nuovo_nome,
   Num_Telefono = nuovo_numero
WHERE (Codice_Reparto,Ospedale) = (cod_reparto,cod_ospedale);
```

```
CREATE PROCEDURE ASL.reparto_cancella (IN cod_reparto CHAR(5),IN cod_ospedale CHAR(4))
DELETE FROM ASL.reparto
WHERE (Codice_Reparto,Ospedale) = (cod_reparto,cod_ospedale);

CREATE PROCEDURE ASL.reparto_cerca (IN cod_reparto CHAR(5), IN cod_ospedale CHAR(4))
SELECT *
FROM ASL.reparto
WHERE (Codice_Reparto,Ospedale) = (cod_reparto,cod_ospedale);

CREATE PROCEDURE ASL.mostra_reparti ()
SELECT *
FROM ASL.reparto;
```

Operazioni degli Amministratori

```
CREATE PROCEDURE ASL.report_medici_esami (IN data_inizio DATE,IN data_fine DATE)
SELECT CF_Medico AS Codice_Medico,Nome_Medico,Cognome AS
Cognome_Medico,Tipo_Esame,Nome_Esame,Data,Ora
FROM ASL.esamereale
   JOIN ASL.esame ON esamereale.Tipo_Esame = esame.Codice_Esame
   RIGHT JOIN ASL.medico ON esamereale.Medico = medico.CF_Medico
WHERE Data>=data_inizio AND Data<=data_fine OR Data is NULL
ORDER BY Cognome_Medico;

CREATE PROCEDURE ASL.report_numero_esami (data_inizio DATE, data_fine DATE)
SELECT CF_Medico AS Codice,Nome_Medico AS Nome ,Cognome, COUNT(Medico) AS num_esami
FROM ASL.esamereale RIGHT JOIN ASL.medico ON esamereale.Medico = medico.CF_Medico
WHERE Data is null or Data>=data_inizio AND Data<=data_fine
GROUP BY CF_Medico
ORDER BY num_esami DESC;
```

Operazioni del Personale CUP

```
CREATE PROCEDURE ASL.report_esami_paziente (IN CF CHAR(16))
SELECT
CF_Paziente AS CodiceFiscale,
 Nome_Paziente AS Nome,
 Cognome_Paziente AS Cognome,
 Tipo_Esame AS cod_Esame,
 Nome_Esame AS Esame,
 esamereale.Data AS Data,
 esamereale.Ora AS Ora,
 Diagnosi
FROM ASL.paziente
   JOIN ASL.prenotazione ON paziente.CF_Paziente = prenotazione.Paziente
   JOIN ASL.esamereale ON prenotazione.Codice_Prenotazione = esamereale.Cod_Prenotazione
   JOIN ASL.esame ON esamereale.Tipo_Esame = esame.Codice_Esame
WHERE CF_Paziente = CF;

CREATE PROCEDURE ASL.report_risultati_prenotazione(IN prenotazione CHAR(8))
SELECT Tipo_Esame AS Codice,Nome_Esame,Data,Ora,Parametri,Diagnosi
```

```
FROM ASL.esamereale JOIN ASL.esame ON esamereale.Tipo_Esame = esame.Codice_Esame
WHERE prenotazione = Cod_Prenotazione;









# ================================USERS================================#


# Creo gli utenti e garantisco i privilegi sulle tabelle che devono gestire

CREATE USER 'personale_CUP' IDENTIFIED BY 'personalecup';
CREATE USER 'amministratore' IDENTIFIED BY 'admin';
GRANT ALL PRIVILEGES ON ASL.paziente TO 'personale_CUP';
GRANT ALL PRIVILEGES ON ASL.esamereale TO 'personale_CUP';
GRANT ALL PRIVILEGES ON ASL.prenotazione TO 'personale_CUP';
GRANT ALL PRIVILEGES ON ASL.esame TO 'amministratore';
GRANT ALL PRIVILEGES ON ASL.ospedale TO 'amministratore';
GRANT ALL PRIVILEGES ON ASL.laboratorio TO 'amministratore';
GRANT ALL PRIVILEGES ON ASL.reparto TO 'amministratore';
GRANT ALL PRIVILEGES ON ASL.medici TO 'amministratore';
GRANT ALL PRIVILEGES ON ASL.responsabileospedale TO 'amministratore';
GRANT ALL PRIVILEGES ON ASL.responsabilelaboratorio TO 'amministratore';


#grant procedure Personale CUP

GRANT EXECUTE ON PROCEDURE ASL.esamereale_aggiungi TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.esamereale_cancella TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.esamereale_cerca TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.esamereale_inserisci_diagnosi TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.esamereale_modifica_data TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.paziente_aggiungi TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.paziente_cancella TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.paziente_cerca TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.paziente_modifica TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.report_esami_paziente TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.report_risultati_prenotazione TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.mostra_esami TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.mostra_esamireali TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.mostra_pazienti TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.esame_cerca TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.medico_cerca TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.mostra_medici TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.laboratorio_cerca TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.mostra_laboratori TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.prenotazione_aggiungi TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.prenotazione_cancella TO 'personale_CUP';
GRANT EXECUTE ON PROCEDURE ASL.mostra_prenotazioni TO 'personale_CUP';


#grant procedure Amministratore

GRANT EXECUTE ON PROCEDURE ASL.esame_aggiungi TO 'amministratore';
```

```
GRANT EXECUTE ON PROCEDURE ASL.esame_cancella TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.esame_cerca TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.esame_modifica TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.mostra_esami TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.ospedale_aggiungi TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.ospedale_cancella TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.ospedale_cerca TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.mostra_ospedali TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.ospedale_modifica TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.laboratorio_aggiungi TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.laboratorio_cancella TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.laboratorio_cerca TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.laboratorio_modifica TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.mostra_laboratori TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.reparto_aggiungi TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.reparto_cancella TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.reparto_cerca TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.reparto_modifica TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.mostra_reparti TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.mostra_medici TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.medico_cerca TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.medico_aggiungi TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.medico_modifica TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.medico_cancella TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.nomina_resp_osp TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.nomina_resp_lab TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.nomina_primario TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.rimuovi_primario TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.nuova_specializzazione TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.mostra_specializzazioni TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.primario_agg_spec TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.primario_rim_spec TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.primario_cerca TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.mostra_primari TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.primario_mostra_spec TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.nomina_volontario TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.rimuovi_volontario TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.volontario_cerca  TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.mostra_volontari TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.report_numero_esami TO 'amministratore';
GRANT EXECUTE ON PROCEDURE ASL.report_medici_esami TO 'amministratore';
```

Codice del Front-End

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <mysql.h>
#include <unistd.h>
#define fflush(stdin) while ((getchar()) != '\n')

MYSQL *conn;
MYSQL *login;
char u[255];
char p[255];
char query[255];
char c;
```

```
int cmd1 = 0;
int cmd2 = 0;
int num_fields;
MYSQL_RES *result;
MYSQL_ROW row;
MYSQL_FIELD *field;

static void finish_with_error(MYSQL *con, char *err)
{
	fprintf(stderr, "%s error: %s\n", err, mysql_error(con));
	mysql_close(con);
	exit(1);
}

void input_wait()
{
	fflush(stdin);
	char c;
	printf("\n> Premi invio per continuare: \n");
	while (c = getchar() != '\n'){}
}

void print_simple_query(char *query)
{
	printf("\n===== %s =====\n",query);
	mysql_query (conn,query);
	result = mysql_store_result(conn);

	if (result == NULL)
	{
		finish_with_error(conn, "errore");
	}

	num_fields = mysql_num_fields(result);
	printf ("\n");
	while ((row = mysql_fetch_row(result)))
	{



		for(int i = 0; i < num_fields; i++)
		{
			if (i == 0)
			{
				while(field = mysql_fetch_field(result)) //include il nome della colonna nella stampa
				{
					printf( "|  %s  ", field->name);
				}
```

```
                                        printf ("\n");
                                }
                        printf(" %s ", row[i] ? row[i] : "NULL");
                        }
                }
                mysql_free_result(result);
                mysql_next_result(conn);
                input_wait();
}

void run_query (char *query)
{
                printf("\n===== %s =====\n",query);
                if(mysql_query(conn,query))
                {
                        finish_with_error(conn, "Query");
                }
                else
                {
                        printf ("> Operazione completata.");
                        input_wait();
                }
}

void do_logout ()
{               printf ("\n================================\n");
                printf ("> Logout eseguito.");
                printf ("\n================================\n\n\n");
                mysql_close (conn);
                strcpy (u,"");
                strcpy (p,"");
}

void admin_logged()
{
                printf("\n_____\n");
                printf ("\n > Accesso eseguito come amministratore del sistema\n\n");
                while (true)
                {
                        printf ("\n\n===========Lista Funzioni eseguibili dal personale del CUP==========\n\n");
                        printf ("  1) Mostra una lista degli esami prenotabili\n");
                        printf ("  2) Mostra una lista degli ospedali\n");
                        printf ("  3) Mostra una lista dei laboratori\n");
                        printf ("  4) Mostra una lista dei reparti ospedalieri\n");
                        printf ("  5) Mostra una lista dei medici registrati\n\n");
                        printf ("  6) Registra un nuovo esame prenotabile\n");
                        printf ("  7) Modifica un esame prenotabile\n");
                        printf ("  8) Cancella un esame prenotabile\n");
```

```
printf ("  9) Cerca un esame prenotabile\n\n");
printf (" 10) Registra un nuovo ospedale\n");
printf (" 11) Modifica un ospedale\n");
printf (" 12) Cancella un ospedale\n");
printf (" 13) Cerca un ospedale\n\n");
printf (" 14) Registra un nuovo laboratorio\n");
printf (" 15) Modifica un laboratorio\n");
printf (" 16) Cancella un laboratorio\n");
printf (" 17) Cerca un laboratorio\n\n");
printf (" 18) Registra un nuovo reparto\n");
printf (" 19) Modifica un reparto\n");
printf (" 20) Cancella un reparto\n");
printf (" 21) Cerca un reparto\n\n");
printf (" 22) Registra un nuovo Medico\n");
printf (" 23) Modifica un Medico registrato\n");
printf (" 24) Cancella un Medico registrato\n");
printf (" 25) Cerca un medico tra quelli registrati\n\n");
printf (" 26) Nomina il responsabile di un Ospedale\n");
printf (" 27) Nomina il responsabile di un Laboratorio\n");
printf (" 28) Nomina il Primario di un reparto\n");
printf (" 29) Rimuovi il Primario di un reparto\n");
printf (" 30) Registra una nuova specializzazione nel sistema\n");
printf (" 31) Mostra una lista delle specializzazioni registrate\n");
printf (" 32) Assegna una nuova specializzazione ad un Primario\n");
printf (" 33) Cancella una specializzazione da un Primario\n");
printf (" 34) Cerca un Primario tra quelli registrati\n");
printf (" 35) Mostra tutti i Primari registrati\n");
printf (" 36) Mostra le specializzazioni di un Primario\n\n");
printf (" 37) Specifica un medico registrato come volontario\n");
printf (" 38) Rimuovi un medico dalla lista dei volontari\n");
printf (" 39) Cerca un volontario tra quelli registrati\n");
printf (" 40) Mostra tutti i volontari registrati\n\n");
printf (" 41) Genera un report sugli esami svolti in un certo periodo dai medici registrati \n");
printf (" 42) Mostra il numero di esami svolti in un certo periodo dai medici registrati \n\n");
printf (" 00) Logout \n");

printf ("\nInserisci un comando: ");
scanf ("%i",&cmd2);
printf ("\n\n\n");

if (cmd2 == 1)
{
        print_simple_query("call mostra_esami");
}
else if (cmd2 == 2)
{
        print_simple_query("call mostra_ospedali");
}
```

```
                else if (cmd2 == 3)
                {
                        print_simple_query("call mostra_laboratori");
                }
                else if (cmd2 == 4)
                {
                        print_simple_query("call mostra_reparti");
                }
                else if (cmd2 == 5)
                {
                        print_simple_query("call mostra_medici");
                }
                else if (cmd2 == 6) //esame_aggiungi (IN cod_esame CHAR(5), IN nome_esame VARCHAR(30),
costo FLOAT)
                {
                        char cod[20],nome[255],costo[255];
                        printf ("\nInserisci il Codice dell'esame: ");
                        scanf ("%s",cod);
                        fflush(stdin);
                        printf ("\nInserisci il Nome dell'esame: ");
                        scanf("%[^\n]",nome);
                        printf ("\nInserisci il Costo dell'esame: [xx.xx]: ");
                        scanf ("%s",costo);
                        snprintf(query, 1000, "call esame_aggiungi('%s','%s',%s);",cod,nome,costo);
                        run_query(query);
                }
                else if (cmd2 == 7) //esame_modifica (IN cod_esame CHAR(5), IN nuovo_codice CHAR(5), IN
nuovo_nome VARCHAR(30), IN nuovo_costo FLOAT)
                {
                        char cod[20],n_cod[20],n_nome[255],n_costo[255];
                        printf ("\nInserisci il Codice dell'esame da modificare: ");
                        scanf ("%s",cod);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo codice dell'esame: ");
                        scanf("%s",n_cod);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo nome dell'esame: ");
                        scanf("%[^\n]",n_nome);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo costo dell'esame: [xx.xx]: ");
                        scanf ("%s",n_costo);
                        snprintf(query, 1000, "call esame_modifica('%s','%s','%s',%s);",cod,n_cod,n_nome,n_costo);
                        run_query(query);
                }
                else if (cmd2 == 8) //esame_cancella (IN cod_esame CHAR(5))
                {
                        char cod[20];
                        printf ("\nInserisci il Codice dell'esame da modificare: ");
                        scanf("%s",cod);
```

```
                        snprintf(query, 1000, "call esame_cancella('%s');",cod);
                        run_query(query);
                }
                else if (cmd2 == 9)
                {
                        char cod[20];
                        printf ("\nInserisci il Codice dell'esame da cercare: ");
                        scanf("%s",cod);
                        snprintf(query, 1000, "call esame_cerca('%s');",cod);
                        print_simple_query(query);
                }
                else if (cmd2 == 10) //ospedale_aggiungi (IN cod_osp CHAR(4), IN nome_osp VARCHAR(20), IN
indirizzo VARCHAR(30))
                {
                        char cod[20],nome[255],indirizzo[255];
                        printf ("\nInserisci il Codice dell'ospedale: ");
                        scanf ("%s",cod);
                        fflush(stdin);
                        printf ("\nInserisci il Nome dell'ospedale: ");
                        scanf("%[^\n]",nome);
                        fflush(stdin);
                        printf ("\nInserisci l'indirizzo dell'ospedale: ");
                        scanf("%[^\n]",indirizzo);
                        snprintf(query, 1000, "call ospedale_aggiungi('%s','%s','%s')",cod,nome,indirizzo);
                        run_query(query);
                }
                else if (cmd2 == 11) //ospedale_modifica (IN cod_osp CHAR(4), IN nuovo_codice CHAR(4),IN
nuovo_nome VARCHAR(20), IN nuovo_indirizzo VARCHAR(30))
                {
                        char cod[20],n_cod[20],n_nome[255],n_indirizzo[255];
                        printf ("\nInserisci il Codice dell'ospedale da modificare: ");
                        scanf ("%s",cod);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo codice dell'ospedale: ");
                        scanf("%s",n_cod);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo nome dell'ospedale: ");
                        scanf("%[^\n]",n_nome);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo indirizzo dell'ospedale: ");
                        scanf("%[^\n]",n_indirizzo);
                        snprintf(query, 1000, "call
ospedale_modifica('%s','%s','%s','%s');",cod,n_cod,n_nome,n_indirizzo);
                        run_query(query);
                }
                else if (cmd2 == 12) //ospedale_cancella (IN cod_osp CHAR(4))
                {
                        char cod[20];
                        printf ("\nInserisci il Codice dell'ospedale da Cancellare: ");
```

```
                        scanf ("%s",cod);
                        fflush(stdin);
                        snprintf(query, 1000, "call ospedale_cancella('%s')",cod);
                        run_query(query);
                }
                else if (cmd2 == 13) //ospedale_cerca (IN cod_osp CHAR(4))
                {
                        char cod[20];
                        printf ("\nInserisci il Codice dell'ospedale da cercare: ");
                        scanf ("%s",cod);
                        fflush(stdin);
                        snprintf(query, 1000, "call ospedale_cerca('%s')",cod);
                        print_simple_query(query);
                }
                else if (cmd2 == 14) //laboratorio_aggiungi (IN cod_lab CHAR(5), IN cod_osp CHAR(4), IN nome
VARCHAR(25), IN piano INT, IN stanza INT)
                {
                        char cod_lab[20],cod_osp[20],nome[255],piano[20],stanza[20];
                        printf ("\nInserisci il Codice del laboratorio: ");
                        scanf ("%s",cod_lab);
                        fflush(stdin);
                        printf ("\nInserisci il Codice dell'ospedale: ");
                        scanf ("%s",cod_osp);
                        fflush(stdin);
                        printf ("\nInserisci il Nome del laboratorio: ");
                        scanf("%[^\n]",nome);
                        fflush(stdin);
                        printf ("\nInserisci il Piano del laboratorio: ");
                        scanf ("%s",piano);
                        fflush(stdin);
                        printf ("\nInserisci la Stanza del laboratorio: ");
                        scanf ("%s",stanza);
                        snprintf(query, 1000, "call laboratorio_aggiungi('%s','%s','%s',%s,
%s);",cod_lab,cod_osp,nome,piano,stanza);
                        run_query(query);
                }

                else if (cmd2 == 15) //laboratorio_modifica (IN cod_lab CHAR(5), IN cod_osp CHAR(4), IN
nuovo_lab CHAR(5), IN nuovo_osp CHAR(4), IN nuovo_nome VARCHAR(25), IN nuovo_piano INT, IN nuova_stanza
INT)
                {
                        char cod_lab[6],cod_osp[5],n_lab[6],n_osp[5],n_nome[255],n_piano[20],n_stanza[20];
                        printf ("\nInserisci il Codice del Laboratorio da modificare: ");
                        scanf ("%s",cod_lab);
                        fflush(stdin);
                        printf ("\nInserisci il Codice dell'Ospedale di appartenenza del laboratorio: ");
                        scanf ("%s",cod_osp);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Codice del laboratorio: ");
```

```
                        scanf ("%s",n_lab);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Codice dell'ospedale: ");
                        scanf ("%s",n_osp);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Nome del laboratorio: ");
                        scanf("%[^\n]",n_nome);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Piano del laboratorio: ");
                        scanf ("%s",n_piano);
                        fflush(stdin);
                        printf ("\nInserisci la nuova Stanza del laboratorio: ");
                        scanf ("%s",n_stanza);
                        snprintf(query, 1000, "call laboratorio_modifica('%s','%s','%s','%s','%s',%s,
%s);",cod_lab,cod_osp,n_lab,n_osp,n_nome,n_piano,n_stanza);
                        run_query(query);
                }
                else if (cmd2 == 16) //laboratorio_cancella (IN cod_lab CHAR(5), IN cod_osp CHAR(4))
                {
                        char cod_lab[6],cod_osp[5];
                        printf ("\nInserisci il Codice del Laboratorio da cancellare: ");
                        scanf ("%s",cod_lab);
                        fflush(stdin);
                        printf ("\nInserisci il Codice dell'Ospedale di appartenenza del laboratorio: ");
                        scanf ("%s",cod_osp);
                        snprintf(query, 1000, "call laboratorio_cancella('%s','%s');",cod_lab,cod_osp);
                        run_query(query);
                }
                else if (cmd2 == 17) //laboratorio_cerca (IN cod_lab CHAR(5), IN cod_osp CHAR(4))
                {
                        char cod_lab[6],cod_osp[5];
                        printf ("\nInserisci il Codice del Laboratorio da cercare: ");
                        scanf ("%s",cod_lab);
                        fflush(stdin);
                        printf ("\nInserisci il Codice dell'Ospedale di appartenenza del laboratorio: ");
                        scanf ("%s",cod_osp);
                        snprintf(query, 1000, "call laboratorio_cerca('%s','%s');",cod_lab,cod_osp);
                        print_simple_query(query);
                }
                else if (cmd2 == 18) //reparto_aggiungi (IN cod_reparto CHAR(5), IN cod_ospedale CHAR(4), IN
nome VARCHAR(20), IN numero_telefono CHAR(10))
                {
                        char cod_rep[6],cod_osp[5],nome[255],numero[11];
                        printf ("\nInserisci il Codice del reparto: ");
                        scanf ("%s",cod_rep);
                        fflush(stdin);
                        printf ("\nInserisci il Codice dell'ospedale di appartenenza del reparto: ");
                        scanf ("%s",cod_osp);
                        fflush(stdin);
```

```
                        printf ("\nInserisci il Nome del reparto: ");
                        scanf("%[^\n]",nome);
                        fflush(stdin);
                        printf ("\nInserisci il Numero di telefono del reparto: ");
                        scanf ("%s",numero);
                        snprintf(query, 1000, "call
reparto_aggiungi('%s','%s','%s','%s');",cod_rep,cod_osp,nome,numero);
                        run_query(query);

                }
                else if (cmd2 == 19) //reparto_modifica (IN cod_reparto CHAR(5), IN cod_ospedale
CHAR(4),IN nuovo_codice CHAR(5),IN nuovo_ospedale CHAR(4), IN nuovo_nome VARCHAR(20), IN
nuovo_numero CHAR(10))
                {
                        char
cod_rep[6],cod_osp[5],n_rep[6],n_osp[5],n_nome[255],n_piano[20],n_numero[11];
                        printf ("\nInserisci il Codice del Reparto da modificare: ");
                        scanf ("%s",cod_rep);
                        fflush(stdin);
                        printf ("\nInserisci il Codice dell'Ospedale di appartenenza del laboratorio: ");
                        scanf ("%s",cod_osp);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Codice del reparto: ");
                        scanf ("%s",n_rep);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Codice dell'ospedale: ");
                        scanf ("%s",n_osp);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Nome del reparto: ");
                        scanf("%[^\n]",n_nome);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Numero del reparto: ");
                        scanf ("%s",n_numero);
                        snprintf(query, 1000, "call
reparto_modifica('%s','%s','%s','%s','%s','%s');",cod_rep,cod_osp,n_rep,n_osp,n_nome,n_numero);
                        run_query(query);
                }
                else if (cmd2 == 20) //reparto_cancella (IN cod_reparto CHAR(5),IN cod_ospedale
CHAR(4))
                {
                        char cod_rep[6],cod_osp[5];
                        printf ("\nInserisci il Codice del reparto da cancellare: ");
                        scanf ("%s",cod_rep);
                        fflush(stdin);
                        printf ("\nInserisci il Codice dell'ospedale di appartenenza del reparto: ");
                        scanf ("%s",cod_osp);
                        snprintf(query, 1000, "call reparto_cancella('%s','%s');",cod_rep,cod_osp);
                        run_query(query);
                }
                else if (cmd2 == 21) //reparto_cerca (IN cod_reparto CHAR(5), IN cod_ospedale CHAR(4))
                {
                        char cod_rep[6],cod_osp[5];
                        printf ("\nInserisci il Codice del reparto da cercare: ");
                        scanf ("%s",cod_rep);
```

```
                        fflush(stdin);
                        printf ("\nInserisci il Codice dell'ospedale di appartenenza del reparto: ");
                        scanf ("%s",cod_osp);
                        snprintf(query, 1000, "call reparto_cerca('%s','%s');",cod_rep,cod_osp);
                        print_simple_query(query);
                }
                else if (cmd2 == 22) //medico_aggiungi (IN cod_medico CHAR(16), nome_medico
VARCHAR(15), cognome_medico VARCHAR(15), indirizzo VARCHAR(30), IN reparto CHAR(5),IN
ospedale CHAR(4))
                {
                        char CF[17],nome[16],cognome[16],indirizzo[31],reparto[6],ospedale[5];
                        printf ("\nInserisci il Codice Fiscale del medico: ");
                        scanf ("%s",CF);
                        fflush(stdin);
                        printf ("\nInserisci il Nome del medico: ");
                        scanf("%[^\n]",nome);
                        fflush(stdin);
                        printf ("\nInserisci il Cognome del medico: ");
                        scanf("%[^\n]",cognome);
                        fflush(stdin);
                        printf ("\nInserisci l'Indirizzo di residenza del medico: ");
                        scanf("%[^\n]",indirizzo);
                        fflush(stdin);
                        printf ("\nInserisci l'Ospedale del medico: ");
                        scanf ("%s",ospedale);
                        fflush(stdin);
                        printf ("\nInserisci il Reparto dove lavora il medico: ");
                        scanf ("%s",reparto);
                        snprintf(query, 1000, "call
medico_aggiungi('%s','%s','%s','%s','%s','%s');",CF,nome,cognome,indirizzo,reparto,ospedale);
                        run_query(query);
                }
                else if (cmd2 == 23) //medico_modifica (IN cod_medico CHAR(16), nuovo_indirizzo
VARCHAR(30), IN nuovo_reparto CHAR(5),IN nuovo_ospedale CHAR(4))
                {
                        char CF[17],indirizzo[31],reparto[6],ospedale[5];
                        printf ("\nInserisci il Codice Fiscale del medico da modificare: ");
                        scanf ("%s",CF);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo indirizzo del medico: ");
                        scanf("%[^\n]",indirizzo);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Ospedale del medico: ");
                        scanf ("%s",ospedale);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Reparto dove lavora il medico: ");
                        scanf ("%s",reparto);
                        snprintf(query, 1000, "call
medico_modifica('%s','%s','%s','%s');",CF,indirizzo,reparto,ospedale);
                        run_query(query);
                }
                else if (cmd2 == 24) //medico_cancella (IN cod_medico CHAR(16))
                {
                        char CF[17];
                        printf("\nInserisci il Codice Fiscale del medico da canellare: ");
```

```
                        scanf ("%s",CF);
                        snprintf(query, 1000, "call medico_cancella('%s');",CF);
                        print_simple_query(query);
                }
                else if (cmd2 == 25) //medico_cerca (IN cod_medico CHAR(16))
                {
                        char CF[17];
                        printf("\nInserisci il Codice Fiscale del medico da cercare: ");
                        scanf ("%s",CF);
                        snprintf(query, 1000, "call medico_cerca('%s');",CF);
                        print_simple_query(query);
                }
                else if (cmd2 == 26) //nomina_resp_osp (IN cod_medico CHAR(16),IN ospedale CHAR(4))
                {
                        char CF[17],cod_osp[5];
                        printf("\nInserisci il Codice dell'ospedale: ");
                        scanf ("%s",cod_osp);
                        fflush(stdin);
                        printf("\nInserisci il Codice Fiscale del medico responsabile dell'ospedale: ");
                        scanf ("%s",CF);
                        snprintf(query, 1000, "call nomina_resp_osp('%s','%s');",CF,cod_osp);
                        run_query(query);
                }
                else if (cmd2 == 27) //nomina_resp_lab (IN cod_medico CHAR(16), IN laboratorio
CHAR(5), IN ospedale CHAR(4))
                {
                        char CF[17],cod_lab[6],cod_osp[5];
                        printf("\nInserisci il Codice del laboratorio: ");
                        scanf ("%s",cod_lab);
                        printf("\nInserisci il Codice dell'ospedale di appartenenza del laboratorio: ");
                        scanf ("%s",cod_osp);
                        fflush(stdin);
                        printf("\nInserisci il Codice Fiscale del medico responsabile del laboratorio: ");
                        scanf ("%s",CF);
                        snprintf(query, 1000, "call nomina_resp_lab('%s','%s','%s');",CF,cod_lab,cod_osp);
                        run_query(query);
                }
                else if (cmd2 == 28) //nomina_primario (IN cod_medico CHAR(16))
                {
                        char CF[17];
                        printf("\nInserisci il Codice Fiscale del medico da nominare primario: ");
                        scanf("%s",CF);
                        snprintf(query, 1000, "call nomina_primario('%s');",CF);
                        run_query(query);
                }
                else if (cmd2 == 29) //rimuovi_primario (IN cod_primario CHAR(16))
                {
                        char CF[17];
                        printf("\nInserisci il Codice Fiscale del primario da rimuovere: ");
                        scanf("%s",CF);
                        snprintf(query, 1000, "call rimuovi_primario('%s');",CF);
                        run_query(query);
                }
                else if (cmd2 == 30) //nuova_specializzazione (IN nome VARCHAR(50))
                {
```

```
                        char spec[51];
                        printf("\nInserisci il nome della nuova Specializzazione Medica: ");
                        fflush(stdin);
                        scanf("%[^\n]",spec);
                        snprintf(query, 1000, "call nuova_specializzazione('%s');",spec);
                        run_query(query);
                }
                else if (cmd2 == 31) //mostra_specializzazioni ()
                {
                        print_simple_query("call mostra_specializzazioni");
                }
                else if (cmd2 == 32) //primario_agg_spec(IN CF_primario CHAR(16),IN specializzazione
VARCHAR(50))
                {
                        char CF[17],spec[51];
                        printf("\nInserisci il Codice Fiscale del Primario: ");
                        scanf("%s",CF);
                        fflush(stdin);
                        printf("\nInserisci il nome della Specializzazione da aggiungere: ");
                        scanf("%[^\n]",spec);
                        snprintf(query, 1000, "call primario_agg_spec('%s','%s');",CF,spec);
                        run_query(query);
                }
                else if (cmd2 == 33) //primario_rim_spec(IN CF_primario CHAR(16),IN specializzazione
VARCHAR(50))
                {
                        char CF[17],spec[51];
                        printf("\nInserisci il Codice Fiscale del Primario: ");
                        scanf("%s",CF);
                        fflush(stdin);
                        printf("\nInserisci il nome della Specializzazione da rimuovere: ");
                        scanf("%[^\n]",spec);
                        snprintf(query, 1000, "call primario_rim_spec('%s','%s');",CF,spec);
                        run_query(query);
                }
                else if (cmd2 == 34) //primario_cerca (IN CodiceFiscale CHAR(16))
                {
                        char CF[17];
                        printf("\nInserisci il Codice Fiscale del Primario da cercare: ");
                        scanf("%s",CF);
                        snprintf(query, 1000, "call primario_cerca('%s');",CF);
                        print_simple_query(query);
                }
                else if (cmd2 == 35) //mostra_primari ()
                {
                        print_simple_query("call mostra_primari");
                }
                else if (cmd2 == 36) //primario_mostra_spec (IN CF CHAR(16))
                {
                        char CF[17];
                        printf("\nInserisci il Codice Fiscale del Primario: ");
                        scanf("%s",CF);
                        snprintf(query, 1000, "call primario_mostra_spec('%s');",CF);
                        print_simple_query(query);
                }
```

```
		else if (cmd2 == 37) //nomina_volontario (IN cod_medico CHAR(16),IN associazione
VARCHAR(50))
		{
			char CF[17],assoc[51];
			printf("\nInserisci il Codice Fiscale del medico da nominare volontario: ");
			scanf("%s",CF);
			fflush(stdin);
			printf("\nInserisci il nome dell'Associazione di Volontariato: ");
			scanf("%[^\n]",assoc);
			snprintf(query, 1000, "call nomina_volontario('%s','%s');",CF,assoc);
			run_query(query);
		}
		else if (cmd2 == 38) //rimuovi_volontario (IN cod_volontario CHAR(16))
		{
			char CF[6];
			printf("\nInserisci il Codice Fiscale del medico da rimuovere tra i volontari: ");
			scanf("%s",CF);
			snprintf(query, 1000, "call rimuovi_volontario('%s');",CF);
			run_query(query);
		}
		else if (cmd2 == 39) //volontario_cerca (IN CodiceFiscale CHAR(16))
		{
			char CF[6];
			printf("\nInserisci il Codice Fiscale del volontario da cercare: ");
			scanf("%s",CF);
			snprintf(query, 1000, "call volontario_cerca('%s');",CF);
			print_simple_query(query);
		}
		else if (cmd2 == 40) //mostra_volontari ()
		{
			print_simple_query("call mostra_volontari");
		}
		else if (cmd2 == 41) //report_medici_esami (IN data_inizio DATE,IN data_fine DATE)
AAAA-MM-GG
		{
			char inizio[11],fine[11];
			printf("\nInserisci la data di inizio del report [AAAA-MM-GG]: ");
			scanf ("%s",inizio);
			fflush(stdin);
			printf("\nInserisci la data di fine del report [AAAA-MM-GG]: ");
			scanf ("%s",fine);
			snprintf(query, 1000, "call report_medici_esami('%s','%s');",inizio,fine);
			print_simple_query(query);
		}
		else if (cmd2 == 42) //report_numero_esami (data_inizio DATE, data_fine DATE)
		{
			char inizio[11],fine[11];
			printf("\nInserisci la data di inizio del report [AAAA-MM-GG]: ");
			scanf ("%s",inizio);
			fflush(stdin);
			printf("\nInserisci la data di fine del report [AAAA-MM-GG]: ");
			scanf ("%s",fine);
			snprintf(query, 1000, "call report_numero_esami('%s','%s');",inizio,fine);
			print_simple_query(query);
		}
```

```
                else if (cmd2 == 00)
                {
                        do_logout();
                        break;
                }
        }
}

void cup_logged()
{
        printf("\n_____\n");
        printf ("\n > Accesso eseguito come personale del CUP\n\n");

        while (true)
        {
                printf ("\n\n===========Lista Funzioni eseguibili dal personale del CUP==========\n\
n");
                printf ("  1) Mostra una lista degli esami prenotabili\n");
                printf ("  2) Mostra una lista dei laboratori\n");
                printf ("  3) Mostra una lista dei pazienti registrati\n");
                printf ("  4) Mostra una lista dei medici registrati\n\n");
                printf ("  5) Cerca un esame tra quelli prenotabili\n");
                printf ("  6) Cerca un laboratorio tra quelli registrati\n");
                printf ("  7) Cerca un paziente registrato nel sistema\n");
                printf ("  8) Cerca un medico tra quelli registrati\n\n");
                printf ("  9) Apri una nuova prenotazione relativa ad un paziente\n");
                printf (" 10) Cancella una prenotazione aperta\n");
                printf (" 11) Mostra una lista delle prenotazioni registrate\n\n");
                printf (" 12) Registra un nuovo esame svolto\n");
                printf (" 13) Cancella un esame tra quelli svolti\n");
                printf (" 14) Inserisci una diagnosi ad un esame svolto\n");
                printf (" 15) Modifica la data di un esame svolto\n");
                printf (" 16) Cerca un esame tra quelli svolti\n");
                printf (" 17) Mostra una lista degli esami svolti, registrati nel sistema\n\n");
                printf (" 18) Registra un nuovo paziente nel sistema\n");
                printf (" 19) Cancella un paziente dal sistema\n");
                printf (" 20) Modifica un paziente registrato nel sistema\n");
                printf (" 21) Cerca un paziente registrato nel sistema\n\n");
                printf (" 22) Genera un report con i risultati degli esami svolti relativi ad una prenotazione\n");
                printf (" 23) Genera un report con i risultati di tutti gli esami svolti da un paziente registrato\n\
n");
                printf (" 00) Logout \n");
                printf ("\nInserisci un comando: ");
                scanf ("%i",&cmd2);
                printf ("\n\n\n");
                if (cmd2 == 1)
                {
                        print_simple_query("call mostra_esami");
                }
                else if (cmd2 == 2)
                {
                        print_simple_query("call mostra_laboratori");
                }
                else if (cmd2 == 3)
                {
```

```
                print_simple_query("call mostra_pazienti");
        }
        else if (cmd2 == 4)
        {
                print_simple_query("call mostra_medici");
        }
        else if (cmd2 == 5)
        {
                char cod[20];
                printf ("\nInserisci il Codice dell'esame da cercare: ");
                scanf("%s",cod);
                snprintf(query, 1000, "call esame_cerca('%s');",cod);
                print_simple_query(query);
        }
        else if (cmd2 == 6)
        {
                char cod_lab[6],cod_osp[5];
                printf ("\nInserisci il Codice del Laboratorio da cercare: ");
                scanf ("%s",cod_lab);
                fflush(stdin);
                printf ("\nInserisci il Codice dell'Ospedale di appartenenza del laboratorio: ");
                scanf ("%s",cod_osp);
                snprintf(query, 1000, "call laboratorio_cerca('%s','%s');",cod_lab,cod_osp);
                print_simple_query(query);
        }
        else if (cmd2 == 7)
        {
                char CF[17];
                printf ("\nInserisci il Codice Fiscale del paziente da cercare: ");
                scanf("%s",CF);
                snprintf(query, 1000, "call esame_cerca('%s');",CF);
                print_simple_query(query);
        }
        else if (cmd2 == 8)
        {
                char CF[17];
                printf("Inserisci il Codice Fiscale del medico da cercare: ");
                scanf ("%s",CF);
                snprintf(query, 1000, "call medico_cerca('%s');",CF);
                print_simple_query(query);
        }
        else if (cmd2 == 9) //prenotazione_aggiungi (IN cod_prenotazione CHAR(8),IN cf_paziente
CHAR(16))
        {
                char pren[9],CF[17];
                printf ("\nInserisci il Codice Fiscale del paziente che vuole aprire la prenotazione: ");
                scanf ("%s",CF);
                printf ("\nInserisci il Codice numerico da associare alla prenotazione: ");
                scanf ("%s",pren);
                snprintf(query, 1000, "call prenotazione_aggiungi('%s','%s');",pren,CF);
                run_query(query);
        }
        else if (cmd2 == 10) //prenotazione_cancella (IN cod_prenot CHAR(8))
        {
                char pren[9];
```

```
                        printf ("\nInserisci il Codice della prenotazione da cancellare: ");
                        scanf ("%s",pren);
                        snprintf(query, 1000, "call prenotazione_cancella('%s');",pren);
                        run_query(query);
                }
                else if (cmd2 == 11)
                {
                        print_simple_query("call mostra_prenotazioni");
                }
                else if (cmd2 == 12) //esamereale_aggiungi (IN cod_prenot CHAR(8), cod_esame CHAR (5),
data DATE, ora TIME, urgenza VARCHAR(15), parametri VARCHAR(30), laboratorio CHAR(5), ospedale
CHAR(4), medico CHAR(16), diagnosi VARCHAR(50))
                {
                        char
pren[9],esame[6],data[11],ora[6],urg[16],param[31],lab[6],osp[5],med[17],diagn[51];
                        printf ("\nInserisci il codice della prenotazione: ");
                        scanf ("%s",pren);
                        fflush(stdin);
                        printf ("\nInserisci il codice dell'esame: ");
                        scanf ("%s",esame);
                        fflush(stdin);
                        printf ("\nInserisci la data di svolgimento dell'esame [AAAA-MM-GG]: ");
                        scanf ("%s",data);
                        fflush(stdin);
                        printf ("\nInserisci l'ora di svolgimento dell'esame [hh:mm]: ");
                        scanf ("%s",ora);
                        fflush(stdin);
                        printf ("\nInserisci l'urgenza dell'esame: ");
                        scanf("%[^\n]",urg);
                        fflush(stdin);
                        printf ("\nInserisci i parametri riscontrati: ");
                        scanf("%[^\n]",param);
                        fflush(stdin);
                        printf ("\nInserisci l'ospedale: ");
                        scanf ("%s",osp);
                        fflush(stdin);
                        printf ("\nInserisci il laboratorio: ");
                        scanf ("%s",lab);
                        fflush(stdin);
                        printf ("\nInserisci il medico che svolge l'esame: ");
                        scanf ("%s",med);
                        fflush(stdin);
                        printf ("\nInserisci la diagnosi relativa all'esame: ");
                        scanf("%[^\n]",diagn);
                        snprintf(query, 1000, "call
esamereale_aggiungi('%s','%s','%s','%s','%s','%s','%s','%s','%s','%s');",pren,esame,data,ora,urg,param,lab,osp,
med,diagn);
                        run_query(query);
                }
                else if (cmd2 == 13) //esamereale_cancella (IN cod_prenot CHAR(8), cod_esame CHAR (5),
data_ DATE, ora_ TIME)
                {
                        char pren[9],esame[6],data[11],ora[6];
                        printf ("\nInserisci il codice di prenotazione dell'esame da cancellare: ");
                        scanf ("%s",pren);
```

```
                        fflush(stdin);
                        printf ("\nInserisci il codice dell'esame: ");
                        scanf ("%s",esame);
                        fflush(stdin);
                        printf ("\nInserisci la data di svolgimento dell'esame da cancellare [AAAA-MM-GG]:
");
                        scanf ("%s",data);
                        fflush(stdin);
                        printf ("\nInserisci l'ora di svolgimento dell'esame da cancellare [hh:mm]: ");
                        scanf ("%s",ora);
                        snprintf(query, 1000, "call
esamereale_cancella('%s','%s','%s','%s');",pren,esame,data,ora);
                        run_query(query);
                }
                else if (cmd2 == 14) //esamereale_inserisci_diagnosi (IN cod_prenot CHAR(8), cod_esame
CHAR (5), data DATE, ora TIME, IN nuovi_parametri VARCHAR(30), IN nuova_diagnosi VARCHAR(50))
                {
                        char pren[9],esame[6],data[11],ora[6],param[31],diagn[51];
                        printf ("\nInserisci il codice della prenotazione: ");
                        scanf ("%s",pren);
                        fflush(stdin);
                        printf ("\nInserisci il codice dell'esame: ");
                        scanf ("%s",esame);
                        fflush(stdin);
                        printf ("\nInserisci la data di svolgimento dell'esame [AAAA-MM-GG]: ");
                        scanf ("%s",data);
                        fflush(stdin);
                        printf ("\nInserisci l'ora di svolgimento dell'esame [hh:mm]: ");
                        scanf ("%s",ora);
                        fflush(stdin);
                        printf ("\nInserisci i parametri riscontrati: ");
                        scanf("%[^\n]",param);
                        fflush(stdin);
                        printf ("\nInserisci la diagnosi relativa all'esame: ");
                        scanf("%[^\n]",diagn);
                        snprintf(query, 1000, "call
esamereale_inserisci_diagnosi('%s','%s','%s','%s','%s','%s');",pren,esame,data,ora,param,diagn);
                        run_query(query);
                }
                else if (cmd2 == 15) //esamereale_modifica_data (IN cod_prenot CHAR(8), IN esame
CHAR(5),IN data DATE,IN ora TIME, nuova_data DATE, nuova_ora TIME)
                {
                        char pren[9],esame[6],data[11],ora[6],n_data[11],n_ora[6];
                        printf ("\nInserisci il codice della prenotazione: ");
                        scanf ("%s",pren);
                        fflush(stdin);
                        printf ("\nInserisci il codice dell'esame: ");
                        scanf ("%s",esame);
                        fflush(stdin);
                        printf ("\nInserisci la data di svolgimento dell'esame [AAAA-MM-GG]: ");
                        scanf ("%s",data);
                        fflush(stdin);
                        printf ("\nInserisci l'ora di svolgimento dell'esame [hh:mm]: ");
                        scanf ("%s",ora);
                        fflush(stdin);
```

```
                        printf ("\nInserisci la nuova data [AAAA-MM-GG]: ");
                        scanf ("%s",n_data);
                        fflush(stdin);
                        printf ("\nInserisci la nuova ora [hh:mm]: ");
                        scanf ("%s",n_ora);
                        snprintf(query, 1000, "call
esamereale_modifica_data('%s','%s','%s','%s','%s','%s');",pren,esame,data,ora,n_data,n_ora);
                        run_query(query);
                }
                else if (cmd2 == 16) //esamereale_cerca (IN cod_prenot CHAR(8), cod_esame CHAR (5),
data_ DATE, ora_ TIME)
                {
                        char pren[9],esame[6],data[11],ora[6];
                        printf ("\nInserisci il codice di prenotazione dell'esame da cercare: ");
                        scanf ("%s",pren);
                        fflush(stdin);
                        printf ("\nInserisci il codice dell'esame: ");
                        scanf ("%s",esame);
                        fflush(stdin);
                        printf ("\nInserisci la data di svolgimento dell'esame da cercare [AAAA-MM-GG]: ");
                        scanf ("%s",data);
                        fflush(stdin);
                        printf ("\nInserisci l'ora di svolgimento dell'esame da cercare [hh:mm]: ");
                        scanf ("%s",ora);
                        snprintf(query, 1000, "call
esamereale_cerca('%s','%s','%s','%s');",pren,esame,data,ora);
                        print_simple_query(query);
                }
                else if (cmd2 == 17) //mostra_esamireali ()
                {
                        print_simple_query("call mostra_esamireali");
                }
                else if (cmd2 == 18) //paziente_aggiungi (IN CF CHAR(16), IN nome VARCHAR(15), IN
cognome VARCHAR(15),IN indirizzo VARCHAR(30),IN Luogo VARCHAR(30),IN data DATE, IN Telefono
VARCHAR(10),IN Mail VARCHAR(30),IN Cellulare VARCHAR(10))
                {
                        char
CF[17],nome[16],cognome[16],indirizzo[31],luogo[31],data[11],tel[11],mail[31],cell[11];
                        printf ("\nInserisci il Codice Fiscale del Paziente: ");
                        scanf ("%s",CF);
                        fflush(stdin);
                        printf ("\nInserisci il Nome del Paziente: ");
                        scanf("%[^\n]",nome);
                        fflush(stdin);
                        printf ("\nInserisci il Cognome del Paziente: ");
                        scanf("%[^\n]",cognome);
                        fflush(stdin);
                        printf ("\nInserisci l'Indirizzo del Paziente: ");
                        scanf("%[^\n]",indirizzo);
                        fflush(stdin);
                        printf ("\nInserisci il Luogo di Nascita del Paziente: ");
                        scanf("%[^\n]",luogo);
                        fflush(stdin);
                        printf ("\nInserisci la Data di Nascita del Paziente: ");
                        scanf ("%s",data);
```

```
                        fflush(stdin);
                        printf ("\nInserisci il Numero di Telefono del Paziente: ");
                        scanf ("%s",tel);
                        fflush(stdin);
                        printf ("\nInserisci l'Indirizzo e-mail del Paziente: ");
                        scanf ("%s",mail);
                        fflush(stdin);
                        printf ("\nInserisci il Numero di Cellulare del Paziente: ");
                        scanf ("%s",cell);
                        snprintf(query, 1000, "call
paziente_aggiungi('%s','%s','%s','%s','%s','%s','%s','%s','%s');",CF,nome,cognome,indirizzo,luogo,data,tel,mail
,cell);
                        run_query(query);
                }
                else if (cmd2 == 19) //paziente_cancella(IN CF CHAR(16))
                {
                        char CF[17];
                        printf ("\nInserisci il Codice Fiscale del Paziente da cancellare: ");
                        scanf ("%s",CF);
                        snprintf(query, 1000, "call paziente_cancella('%s');",CF);
                        run_query(query);
                }
                else if (cmd2 == 20) //paziente_modifica (IN CF CHAR(16),IN n_indirizzo VARCHAR(30),
IN n_Telefono VARCHAR(10),IN n_Mail VARCHAR(30),IN n_Cellulare VARCHAR(10))
                {
                        char CF[17],indirizzo[31],tel[11],mail[31],cell[11];
                        printf ("\nInserisci il Codice Fiscale del Paziente da modificare: ");
                        scanf ("%s",CF);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Indirizzo: ");
                        scanf("%[^\n]",indirizzo);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Numero di Telefono: ");
                        scanf ("%s",tel);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Indirizzo e-mail: ");
                        scanf ("%s",mail);
                        fflush(stdin);
                        printf ("\nInserisci il nuovo Numero di Cellulare: ");
                        scanf ("%s",cell);
                        snprintf(query, 1000, "call
paziente_modifica('%s','%s','%s','%s','%s');",CF,indirizzo,tel,mail,cell);
                        run_query(query);

                }
                else if (cmd2 == 21) //paziente_cerca (IN CodiceFiscale CHAR(16))
                {
                        char CF[17];
                        printf ("\nInserisci il Codice Fiscale del Paziente da cercare: ");
                        scanf ("%s",CF);
                        snprintf(query, 1000, "call paziente_cerca('%s');",CF);
                        print_simple_query(query);
                }
                else if (cmd2 == 22) //report_risultati_prenotazione(IN prenotazione CHAR(8))
                {
```

```
                        char cod[9];
                        printf ("\nInserisci il codice della prenotazione su cui generare il report: ");
                        scanf ("%s",cod);
                        snprintf(query, 1000, "call report_risultati_prenotazione('%s');",cod);
                        print_simple_query(query);
                }
                else if (cmd2 == 23) //report_esami_paziente (IN CF CHAR(16))
                {
                        char CF[17];
                        printf ("\nInserisci il Codice Fiscale del paziente su cui generare il report: ");
                        scanf ("%s",CF);
                        snprintf(query, 1000, "call report_esami_paziente('%s');",CF);
                        print_simple_query(query);
                }
                else if (cmd2 == 00)
                {
                        do_logout();
                        break;
                }
        }
}

void do_login ()
{
        printf("\n========Login=======\n\n");
        printf("Inserisci l'username: ");
        scanf("%s",u);
        printf("Inserisci la password: ");
        scanf("%s",p);
        conn = mysql_init (NULL);
        login = mysql_real_connect(conn, "localhost",u,p, "ASL", 3306, NULL, 0);

        if (login == NULL)
        {
                fprintf(stderr, "%s\n", mysql_error(conn));
                mysql_close(conn);
                exit(1);
        }

        else
        {
                printf ("\nConnessione riuscita\n");
                if (strcmp(u, "personale_CUP") == 0)
                {
                printf("Loggato come membro del personale CUP\n\n");
                cup_logged();

                }

                else if (strcmp(u, "amministratore") == 0)
                {
                        printf ("Loggato come amministratore");
                        admin_logged();
                }
        }
```

```
}

int main (int argc, char *argv[])
{
	while (true)
	{
		printf("\n==========Programma Gestione ASL==========\n\nInserisci un
Comando:\n");
		printf ("1) Login\n2) Termina Sessione\n");
		scanf ("%i",&cmd1);
		if (cmd1 == 1)
		{
			do_login();
		}

		else
		{
			printf("==================================\nProgramma Gestione ASL
terminato.\n==================================\n\n");
			exit(0);
		}
	}
}
```