Università degli Studi di Napoli Federico II Scuola Politecnica e delle Scienze di Base

Dipartimento di Ingegneria Elettrica e Tecnologie dell’Informazione

Corso di Laurea in Informatica - Insegnamento di Basi di Dati I

Anno Accademico 2022/2023

Progettazione e Sviluppo di una Base di Dati Relazionale per la Gestione di un Laboratorio Scientifico

Antonio Abbatiello - N86003037

Leonardo Colamarino - N86003586

21/02/2023

Indice

1 Descrizione del Progetto6

1.1 Analisi del problema6

2 Progettazione Concettuale7

2.1 Introduzione7

2.2 Convenzione nell’uso di UML7

2.3 Class Diagram8

2.4 Ristrutturazione9

2.4.1 Analisi delle ridondanze9

2.4.2 Analisi degli identificativi9

2.4.3 Rimozione degli attributi multipli9

2.4.4 Rimozione degli attributi strutturati10

2.4.5 Rimozione delle classi di associazione10

2.4.6 Rimozione delle gerarchie di specializzazione10

2.4.7 Rimozione delle composizioni ed aggregazioni10

2.5 Class Diagram Ristrutturato11

2.6 Dizionario delle classi12

2.7 Dizionario delle associazioni15

2.8 Dizionario dei vincoli17

3 Progettazione Logica21

3.1 Introduzione21

3.2 Schema logico21

4 Progettazione Fisica23

4.1 Introduzione23

4.2 Tabelle e Sequence24

4.2.1 Sequence24

4.2.2 Tabella: Laboratorio25

4.2.3 Tabella: Sede25

4.2.4 Tabella: Postazione25

4.2.5 Tabella: Responsabile26

4.2.6 Tabella: Tecnico27

4.2.7 Tabella: Strumento28

4.2.8 Tabella: Prenotazione28

4.2.9 Tabella: Utente29

4.3 Viste30

4.3.1 Viste per il riepilogo mensile sulle prenotazioni di strumento 30

4.3.2 Viste per il riepilogo annuale sulle prenotazioni di strumento 32

4.4 Funzioni e Procedure34

4.4.1 Funzione per l’ encrypt di una password di un utente 34

4.4.2 Funzione per il decrypt di una password di un utente 34

4.4.3 Funzione per il controllo delle credenziali di login 35

4.4.4 Funzione per il controllo della validità di una password 35

4.5 Trigger36

4.5.1 Trigger NoOverlapPren 36

4.5.2 Trigger ModifyOrDeletePren 37

4.5.3 Trigger InserisciPw 38

4.5.4 Trigger NoDoublePren 39

5 Manuale d’uso40

5.1 Popolamento del database40

5.1.1 Popolamento della tabella Laboratorio 40

5.1.2 Popolamento della tabella Sede 40

5.1.3 Popolamento della tabella Postazione 41

5.1.4 Popolamento della tabella Responsabile 42

5.1.5 Popolamento della tabella Tecnico 43

5.1.6 Popolamento della tabella Strumento 44

5.1.7 Popolamento della tabella Utente 46

5.1.8 Popolamento della tabella Prenotazione 47

Capitolo 1

Descrizione del Progetto

* 1. Analisi del problema

L’obiettivo principale del progetto sarà l’implementazione di una base di dati relazionale come supporto per la gestione di prenotazioni di strumenti in sedi di laboratori scientifici.

La base di dati conterrà informazioni su laboratori scientifici e le sedi presenti su tutto il territorio, oltre ad informazioni personali sui dipendenti (tecnici e responsabili) in sedi di laboratorio. Ogni sede avrà un determinato numero di postazioni in cui saranno disponibili strumenti per lo studio in ambito scientifico.

Saranno memorizzate credenziali su utenti registrati, i quali potranno effettuare prenotazioni su strumenti di laboratorio. Le prenotazioni per uno strumento potranno essere effettuate in base a:

* Tipo e descrizione
* Sede di appartenenza

Nel caso in cui lo strumento da prenotare in un certo orario sia già associato ad una prenotazione nello stesso giorno con un orario che si sovrappone, allora la prenotazione verrà abortita. Infine, agli utenti sarà data la possibilità di gestire le proprie prenotazioni, tramite modifica o cancellazione.

Per ogni strumento sarà visibile un riepilogo, su base mensile ed annuale, di statistiche pertinenti alla frequenza di utilizzo ed agli utenti che lo hanno utilizzato.

Capitolo 2

Progettazione concettuale

2.1 Introduzione

Il seguente capitolo presenta la progettazione del sistema di gestione di laboratori al livello più astratto, che verra rappresentato usando un Class Diagram UML.

Saranno descritte le classi più rilevanti ai requisiti richiesti dal progetto e le associazioni tra esse, oltre ad includere eventuali vincoli.

2.2 Convenzioni nell’uso di UML

Si è scelto di adottare le seguenti convenzioni per rendere la lettura del diagramma UML più leggibile e comprensibile al lettore:

* Tutti gli attributi, ad eccezioni di quelli in cui specificata, hanno molteplicità pari ad [1].
* Le note esterne a classi ed associazioni sono colorate in rosso.
* Tutti i nomi di associazioni sono stati omessi per leggibilità. Saranno definiti all’interno del dizionario di associazioni.

2.3 Class Diagram

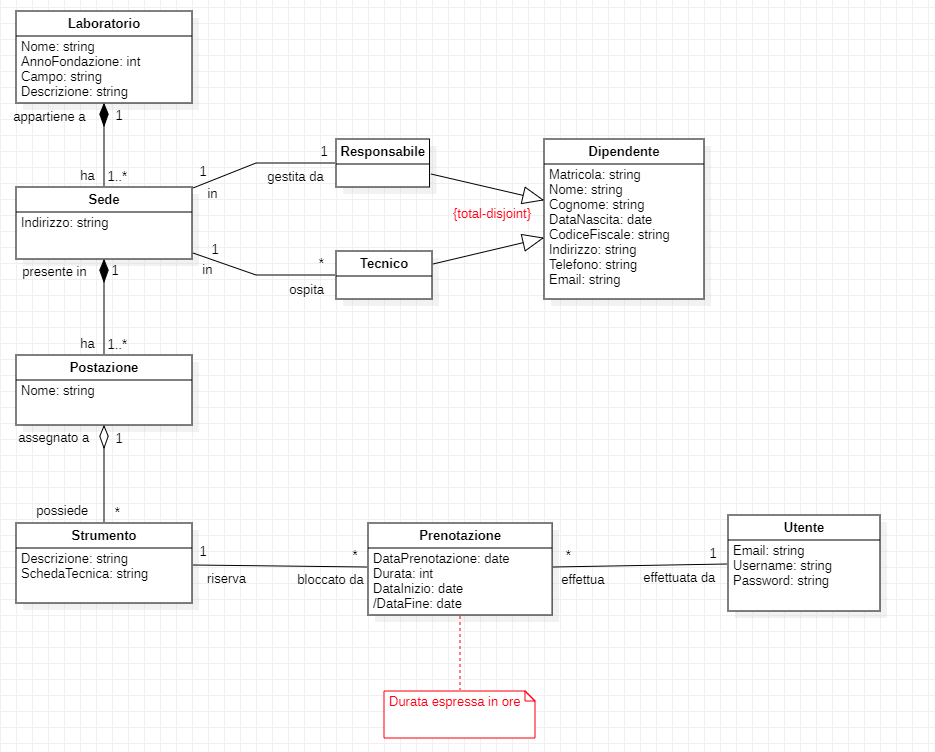


Figura 1: -Class Diagram

2.4 Ristrutturazione

Per tradurre lo schema concettuale in schema relazionale si procederà alla ristrutturazione del Class Diagram precedentemente mostrato. La ristrutturazione sarà necessaria per ricostruire il quadro concettuale in modo da ridurre il livello di astrazione e semplificare il passaggio dalla progettazione concettuale a quella logica e fisica.

Partiremo con l’analisi di ridondanze e di identificativi di classe, seguite dall’eventuale rimozione di attributi strutturati, attributi multipli, composizioni, associazioni e gerarchie di specializzazione.

2.4.1 Analisi delle ridondanze

L’attributo **DataFine** della classe **Prenotazione** è ridondante, siccome è possibile calcolare la data e ora finale della prenotazione tramite gli attributi **DataInizio** e **Durata**. Nei casi in cui sia necessario ottenere la data finale di prenotazione sarà possibile ricavarne il valore aggiungendo la durata(in ore) alla data iniziale. Pertanto, nel processo di ristrutturazione l’attributo sarà rimosso.

2.4.2 Analisi degli identificativi

Per ogni entità ad eccezione di **Utente** e **Dipendente** verrà aggiunta una chiave surrogata di tipo numerico per identificare univocamente ogni istanza della stessa. Per convenzione avrà la sigla ID\_.

2.4.3 Rimozione degli attributi multipli

L’attributo **Telefono** di **Dipendente** è un attributo multiplo nel caso in cui un dipendente voglia indicare più di un numero telefonico all’interno della sezione dei dati anagrafici. Di conseguenza nella ristrutturazione l’attributo sarà diviso in due attributi singoli: **Telefono1** e **Telefono2**(facoltativo). Inoltre l’attributo **Indirizzo** presente in **Sede** e **Dipendente** sarà trattato come unica stringa di testo.

2.4.4 Rimozione degli attributi strutturati

L’attributo **Indirizzo** presente in **Sede** e **Dipendente** sarà trattato come unica stringa di testo.

2.4.5 Rimozione delle classi di associazione

Non sono presenti classi di associazione.

2.4.6 Rimozione delle gerarchie di specializzazione

La classe generale **Dipendente** sarà compressa sulle due classi specializzate **Responsabile** e **Tecnico**. Le due classi sono strutturalmente simili ma la scelta della divisione di classi porterà ad una gestione più semplice del processo di progettazione fisica.

2.4.7 Rimozione delle composizioni ed aggregazioni

Le composizioni nel Class Diagram sono tra **Laboratorio** e **Sede**, e tra **Sede** e **Postazione.** Saranno sostituite con delle semplici associazioni, tuttavia l’eliminazione di un’istanza della classe totale causerà l’eliminazione di tutte le istanze della classe parte.

L’unica aggregazione è tra le classi **Postazione** e **Strumento**. Uno strumento è contenuto in una postazione ma, non essendo parte fondante di essa, l’eliminazione della postazione non comporterà l’eliminazione degli strumenti associati.

2.5 Class Diagram Ristrutturato

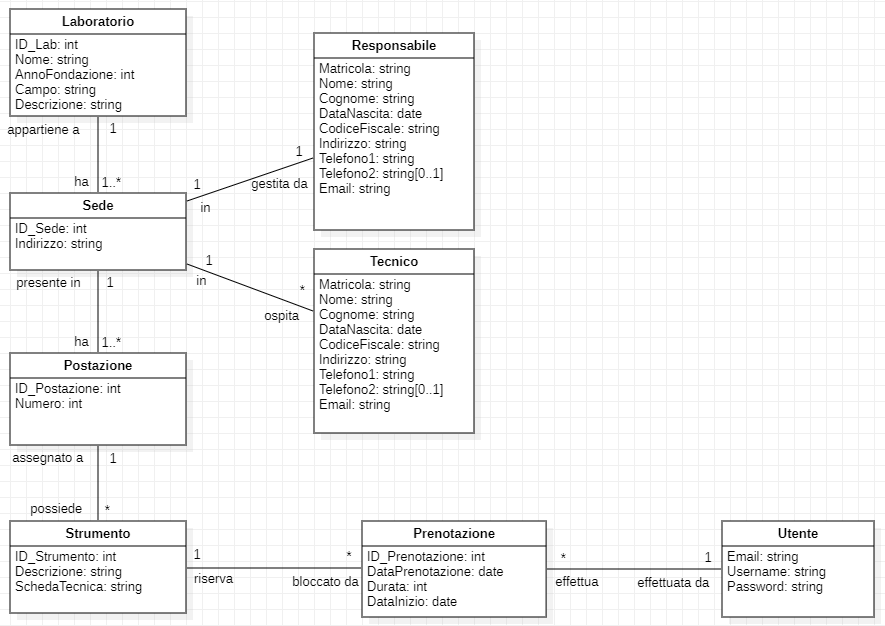


Figura 2: -Class Diagram ristrutturato

2.6 Dizionario delle Classi

|  |  |  |
| --- | --- | --- |
| Classe | Descrizione | Attributi |
| **Laboratorio** | Contiene i dati del laboratorio | **ID\_Lab**(*int*): Identificativo univoco del laboratorio.  **Nome**(*string*):Nome del laboratorio.  **AnnoFondazione**(*int*): Anno in cui è stata fondato il laboratorio.  **Campo**(*string*): Ambito di studi scientifici del laboratorio (es. chimica, biologia).  **Descrizione**(*string*): Descrizione del laboratorio. |
| **Sede** | Contiene i dati sulla sede di un laboratorio | **ID\_Sede**(*int*): Identificativo univoco della sede.  **Indirizzo**(*string*):Indirizzo in cui è situata la sede. |
| **Postazione** | Contiene i dati sulla postazione di una sede | **ID\_Postazione**(*int*): Identificativo univoco della postazione.  **Nome**(*string*): Nominativo associato al luogo fisico della postazione all’interno della sede. |
| **Strumento** | Contiene i dati sullo strumento in una postazione | **ID\_Strumento**(*int*): identificativo univoco dello strumento.  **Descrizione**(*string*): Descrizione dello strumento che ne specifica il nome e le caratteristiche principali.  **SchedaTecnica**(*string*): Scheda tecnica dello strumento. |
| **Responsabile** | Contiene i dati sul responsabile di una sede | **Matricola**(*string*): Numero di matricola del responsabile. Composta dal prefisso “R” seguito da una sequenza di 9 cifre numeriche.  **Nome**(*string*): Nome del responsabile.  **Cognome**(*string*): Cognome del responsabile.  **DataNascita**(*date*): Data di nascita del responsabile.  **CodiceFiscale**(*string*): Codice fiscale del responsabile.  **Indirizzo**(*string*): Indirizzo di residenza del responsabile.  **Telefono1**(*string*): Numero di telefono principale del responsabile.  **Telefono2**(*string*): Numero di telefono secondario del responsabile (opzionale).  **Email**(*string*): Indirizzo email del responsabile. |
| **Tecnico** | Contiene i dati sul tecnico di una sede | **Matricola**(*string*): Numero di matricola del tecnico. Composta dal prefisso “T” seguito da una sequenza di 9 cifre numeriche.  **Nome**(*string*): Nome del tecnico.  **Cognome**(*string*): Cognome del tecnico.  **DataNascita**(*date*): Data di nascita del tecnico.  **CodiceFiscale**(*string*): Codice fiscale del tecnico.  **Indirizzo**(*string*): Indirizzo di residenza del tecnico.  **Telefono1**(*string*): Numero di telefono principale del tecnico.  **Telefono2**(*string*): Numero di telefono secondario del tecnico (opzionale).  **Email**(*string*): Indirizzo email del tecnico. |
| **Prenotazione** | Contiene i dati sulla prenotazione fatta da un utente di uno strumento | **ID\_Prenotazione**(*int*): Identificativo univoco della prenotazione.  **DataPrenotazione**(*date*): Data in cui è stata fatta la prenotazione.  **Durata**(*int*): Durata dell’utilizzo dello strumento prenotato, espressa in ore.  **DataInizio**(*date*): Data in cui si intende utilizzare lo strumento. |
| **Utente** | Contiene i dati di un utente registrato | **Username**(*string*): Nome univoco dell’utente.  **Email**(*string*): Indirizzo email dell’utente.  **Password**(*string*): Password dell’utente. |

2.7 Dizionario delle associazioni

|  |  |  |
| --- | --- | --- |
| Associazione | Descrizione | Classi coinvolte |
| **Appartenenza** | Esprime l’appartenenza di una o più sedi ad un laboratorio | **Laboratorio [1]** ruolo **appartiene a**: indica il laboratorio di cui fa parte la sede.  **Sede [1..\*]** ruolo **ha**: indica le sedi che ha il laboratorio. |
| **Presenza** | Esprime la presenza di una o più postazioni ad una sede | **Sede [1]** ruolo **presente in**: indica la sede di cui fa parte la postazione.  **Postazione [1..\*]** ruolo **ha**: indica le postazioni che ha la sede. |
| **Assegnazione** | Esprime l’assegnazione di uno o più strumenti ad una postazione | **Postazione [1]** ruolo **assegnato a**: indica la postazione a cui è assegnato lo strumento.  **Strumento [1..\*]** ruolo **possiede**: indica gli strumenti che possiede la postazione. |
| **Gestione** | Esprime il responsabile che gestisce la sede | **Responsabile [1]** ruolo **gestita da**: indica il responsabile che gestisce la sede.  **Sede [1]** ruolo **in**: indica la sede che gestisce il responsabile. |
| **Manutenzione** | Esprime l’assegnazione di uno o più tecnici ad una sede | **Tecnico [0..\*]** ruolo **ospita**: indica i tecnici che fanno manutenzione nella sede.  **Sede [1]** ruolo **in**: indica la sede manutenuta dal tecnico. |
| **Riserva** | Esprime le prenotazioni che riservano lo strumento | **Strumento [1]** ruolo **riserva**: indica quale strumento blocca la prenotazione.  **Prenotazione [\*]** ruolo **bloccato da**: indica quale prenotazione blocca lo strumento. |
| **Prenota** | Esprime l’ utente che effettua la prenotazione | **Utente [1]** ruolo **effettuata da**: indica l’utente che fa la prenotazione  **Prenotazione [\*]** ruolo **effettua**: indica le prenotazioni effettuate dall’utente. |

2.8 Dizionario dei vincoli

|  |  |  |
| --- | --- | --- |
| Vincolo | Tipo | Descrizione |
| **compositLab** | Interrelazionale | L’eliminazione di un laboratorio comporterà l’eliminazione di tutte le sedi associate |
| **compositSede** | Interrelazionale | L’eliminazione di una sede comporterà l’eliminazione di tutte le postazioni associate |
| **uniqPostNome** | Intrarelazionale | Il nominativo di postazione sarà unico per la sede |
| **validPostNome** | Dominio | Il nome di postazione dovra’ essere composto da un carattere ‘A-Z’ seguito da una cifra ‘1-9’ |
| **validTel** | Dominio | Non saranno accettati numeri di telefono che non hanno esattamente 10 cifre |
| **validMatr** | Dominio | La matricola dovrà avere esattamente 10 caratteri, con il primo carattere ‘T’(Tecnico) o ‘R’(Responsabile) ed il resto cifre comprese tra ‘0-9’ |
| **validCf** | Dominio | Il codice fiscale dovrà rispettare il formato ufficiale |
| **deleteOrModifyPren** | Interrelazionale | Le prenotazioni potranno essere modificate o cancellate solo se DataInizio è maggiore della data attuale. |
| **validPrenInizio** | Interrelazionale | Le prenotazioni potranno essere effettuate solo se la DataInizio è maggiore della DataPrenotazione |
| **validPrenDurata** | Dominio | Le prenotazioni potranno essere effettuate solo se la Durata è compresa tra 1 ora e 24 ore |
| **noOverlapPren** | Intrarelazionale | Non potranno esistere due prenotazioni per lo stesso strumento con tempi che si sovrappongono |
| **validEmail** | Dominio | Non saranno accettate email che non rispettano la forma: \_@\_.\_ |
| **uniqEmail** | Intrarelazionale | Ogni E-mail sarà univocamente assegnata ad un utente |
| **validUsername** | Dominio | Il nome utente sarà compreso tra 8 e 16 caratteri e potrà contenere solo caratteri contenuti in ‘a-z’, ‘A-Z’e ‘0-9’ |
| **validPw** | Dominio | La password sarà compresa tra 8 e 16 caratteri e dovrà contenere almeno una cifra |
| **noDoublePren** | Intrarelazionale | Non possono esistere due prenotazioni dello stesso utente per strumenti diversi con tempi che si sovrappongono |

Capitolo 3

Progettazione logica

3.1 Introduzione

In questo capitolo sarà trattata la fase successiva della progettazione della base di dati

scendendo ad un livello di astrazione più basso rispetto alla precedente.

Si tradurrà lo schema concettuale (già predisposto in seguito alla ristrutturazione) in

uno schema logico, dipendente dal tipo di struttura dei dati prescelto cioè quello relazionale puro.

Le PK saranno evidenziate da una singola sottolineatura mentre le FK saranno evidenziate da una doppia sottolineatura

3.2 Schema logico

|  |  |
| --- | --- |
| Laboratorio | (ID\_Lab, Nome, AnnoFondazione, Campo, Descrizione) |
| Sede | (ID\_Sede, Indirizzo, ID\_Lab)  ID\_Lab→Laboratorio.ID\_Lab |
| Postazione | (ID\_Postazione, Nome, ID\_Sede)  ID\_Sede→Sede.ID\_Sede |
| Strumento | (ID\_Strumento, Descrizione, SchedaTecnica, ID\_Postazione)  ID\_Postazione→Postazione.ID\_Postazione |
| Responsabile | (Matricola, Nome, Cognome, DataNascita, CodiceFiscale, Indirizzo, Telefono1, Telefono2, Email, ID\_Sede)  ID\_Sede→Sede.ID\_Sede |
| Tecnico | (Matricola, Nome, Cognome, DataNascita, CodiceFiscale, Indirizzo, Telefono1, Telefono2, Email, ID\_Sede)  ID\_Sede→Sede.ID\_Sede |
| Prenotazione | (ID\_Prenotazione, DataPrenotazione, Durata, DataInizio, ID\_Strumento, Username)  ID\_Strumento→Strumento.ID\_Strumento  Username→Utente.Username |
| Utente | (Email, Username, Password) |

Capitolo 4

Progettazione fisica

4.1 Introduzione

In questo paragrafo si forniranno le definizioni scritte in linguaggio SQL delle tabelle indicate dallo schema logico, inoltre saranno implementati i vincoli piu’ rilevanti, alcune funzioni e procedure utili al funzionamento del sistema ed infine alcune viste per delle tabelle di riepilogo di prenotazioni.

Per motivi di semplificazione per la gestione ed il popolamento dello schema del database, saranno implementati alcuni vincoli aggiuntivi non presenti nella sezione di progettazione concettuale della documentazione.

Inoltre, le funzioni scritte in linguaggio procedurale PL/SQL avranno come principale scopo la gestione per un sistema di criptaggio/decriptaggio di password per la tabella Utente.

4.2 Tabelle e Sequence

4.2.1 Sequence

--SEQUENCE: GENERA ID PER LABORATORIO

CREATE SEQUENCE genera\_id\_lab

  START WITH 1

  INCREMENT BY 1

  MINVALUE 1

  NOCACHE;

--SEQUENCE: GENERA ID PER SEDE

CREATE SEQUENCE genera\_id\_sede

  START WITH 1

  INCREMENT BY 1

  MINVALUE 1

  NOCACHE;

--SEQUENCE: GENERA ID PER POSTAZIONE

CREATE SEQUENCE genera\_id\_post

  START WITH 1

  INCREMENT BY 1

  MINVALUE 1

  NOCACHE;

--SEQUENCE: GENERA ID PER STRUMENTO

CREATE SEQUENCE genera\_id\_strum

  START WITH 1

  INCREMENT BY 1

  MINVALUE 1

  NOCACHE;

--SEQUENCE: GENERA ID PER PRENOTAZIONE

CREATE SEQUENCE genera\_id\_pren

  START WITH 1

  INCREMENT BY 1

  MINVALUE 1

  NOCACHE;

4.2.2 Tabella: Laboratorio

--TABLE: LABORATORIO

CREATE TABLE LABORATORIO(

    ID\_LAB INTEGER DEFAULT ON NULL genera\_id\_lab.NEXTVAL PRIMARY KEY,

    NOME VARCHAR2(50) NOT NULL,

    ANNOFONDAZIONE NUMBER(4, 0) NOT NULL,

    CAMPO VARCHAR2(50) DEFAULT ON NULL 'Sconosciuto' NOT NULL,

    DESCRIZIONE VARCHAR2(250) DEFAULT ON NULL 'Descrizione non presente' NOT NULL

);

4.2.3 Tabella: Sede

--TABLE: SEDE

CREATE TABLE SEDE(

    ID\_SEDE INTEGER DEFAULT ON NULL genera\_id\_sede.NEXTVAL PRIMARY KEY,

    INDIRIZZO VARCHAR2(100) NOT NULL,

    ID\_LAB INTEGER NOT NULL,

    --FOREIGN KEY

    CONSTRAINT sede\_fk FOREIGN KEY (ID\_LAB) REFERENCES LABORATORIO(ID\_LAB) ON DELETE CASCADE

);

4.2.4 Tabella: Postazione

--TABLE: POSTAZIONE

CREATE TABLE POSTAZIONE(

    ID\_POSTAZIONE INTEGER DEFAULT ON NULL genera\_id\_post.NEXTVAL PRIMARY KEY,

    NOME CHAR(2) NOT NULL,

    ID\_SEDE INTEGER NOT NULL,

    --FOREIGN KEY

    CONSTRAINT postazione\_fk FOREIGN KEY (ID\_SEDE) REFERENCES SEDE(ID\_SEDE) ON DELETE CASCADE,

    --VINCOLO UNIQ\_POST\_NOME

    CONSTRAINT uniq\_post\_nome UNIQUE (NOME, ID\_SEDE),

    --VINCOLO VALID\_POST\_NOME

    CONSTRAINT valid\_post\_nome CHECK (REGEXP\_LIKE(NOME, '^[A-Z][1-9]$'))

);

4.2.5 Tabella: Tecnico

--TABLE: TECNICO

CREATE TABLE TECNICO(

    MATRICOLA CHAR(10) PRIMARY KEY,

    NOME VARCHAR2(50) NOT NULL,

    COGNOME VARCHAR2(50) NOT NULL,

    DATANASCITA DATE NOT NULL,

    CODICEFISCALE CHAR(16)NOT NULL,

    INDIRIZZO VARCHAR(100) NOT NULL,

    TELEFONO1 CHAR(10) NOT NULL,

    TELEFONO2 CHAR(10),

    EMAIL VARCHAR(100) NOT NULL,

    ID\_SEDE INTEGER NOT NULL,

    --FOREIGN KEY

    CONSTRAINT tecnico\_fk FOREIGN KEY (ID\_SEDE) REFERENCES SEDE(ID\_SEDE) ON DELETE CASCADE,

    --VINCOLO VALID\_MATR

    CONSTRAINT valid\_matr\_t CHECK (REGEXP\_LIKE(MATRICOLA, '^T[0-9]{9}$')),

    --VINCOLO VALID\_TEL

    CONSTRAINT valid\_tel1\_t CHECK (REGEXP\_LIKE(TELEFONO1, '^[0-9]{10}$')),

    CONSTRAINT valid\_tel2\_t CHECK (REGEXP\_LIKE(TELEFONO2, '^[0-9]{10}$')),

    --VINCOLO VALID\_CF

    CONSTRAINT valid\_cf\_t CHECK (REGEXP\_LIKE(CODICEFISCALE, '^[A-Z]{6}[0-9]{2}[A-Z][0-9]{2}[A-Z][0-9]{3}[A-Z]$')),

    --VINCOLO VALID\_EMAIL

    CONSTRAINT valid\_mail\_t CHECK (EMAIL LIKE '\_%@\_%.\_\_%')

);

4.2.6 Tabella: Responsabile

--TABLE: RESPONSABILE

CREATE TABLE RESPONSABILE(

    MATRICOLA CHAR(10) PRIMARY KEY,

    NOME VARCHAR2(50) NOT NULL,

    COGNOME VARCHAR2(50) NOT NULL,

    DATANASCITA DATE NOT NULL,

    CODICEFISCALE CHAR(16) NOT NULL,

    INDIRIZZO VARCHAR(100) NOT NULL,

    TELEFONO1 CHAR(10) NOT NULL,

    TELEFONO2 CHAR(10),

    EMAIL VARCHAR(100) NOT NULL,

    ID\_SEDE INTEGER NOT NULL UNIQUE,

    --FOREIGN KEY

    CONSTRAINT responsabile\_fk FOREIGN KEY (ID\_SEDE) REFERENCES SEDE(ID\_SEDE) ON DELETE CASCADE,

    --VINCOLO VALID\_MATR

    CONSTRAINT valid\_matr\_r CHECK (REGEXP\_LIKE(MATRICOLA, '^R[0-9]{9}$')),

    --VINCOLO VALID\_TEL

    CONSTRAINT valid\_tel1\_r CHECK (REGEXP\_LIKE(TELEFONO1, '^[0-9]{10}$')),

    CONSTRAINT valid\_tel2\_r CHECK (REGEXP\_LIKE(TELEFONO2, '^[0-9]{10}$')),

    --VINCOLO VALID\_CF

    CONSTRAINT valid\_cf\_r CHECK (REGEXP\_LIKE(CODICEFISCALE, '^[A-Z]{6}[0-9]{2}[A-Z][0-9]{2}[A-Z][0-9]{3}[A-Z]$')),

    --VINCOLO VALID\_EMAIL

    CONSTRAINT valid\_email\_r CHECK (EMAIL LIKE '\_%@\_%.\_\_%')

);

4.2.7 Tabella: Strumento

--TABLE: STRUMENTO

CREATE TABLE STRUMENTO(

    ID\_STRUMENTO INTEGER DEFAULT ON NULL genera\_id\_strum.NEXTVAL PRIMARY KEY,

    DESCRIZIONE VARCHAR2(75) NOT NULL,

    SCHEDATECNICA VARCHAR2(250) DEFAULT ON NULL'Scheda tecnica non presente' NOT NULL,

    ID\_POSTAZIONE INTEGER,

    --FOREIGN KEY

    CONSTRAINT strumento\_fk FOREIGN KEY (ID\_POSTAZIONE) REFERENCES POSTAZIONE(ID\_POSTAZIONE) ON DELETE SET NULL

);

4.2.8 Tabella: Prenotazione

--TABLE: PRENOTAZIONE

CREATE TABLE PRENOTAZIONE(

    ID\_PRENOTAZIONE INTEGER DEFAULT ON NULL genera\_id\_pren.NEXTVAL PRIMARY KEY,

    DATAINIZIO DATE NOT NULL,

    DURATA INTEGER NOT NULL,

    DATAPRENOTAZIONE DATE DEFAULT ON NULL SYSDATE NOT NULL,

    ID\_STRUMENTO INTEGER,

    USERNAME VARCHAR(18),

    --FOREIGN KEY

    CONSTRAINT prenotazione\_fk\_strumento FOREIGN KEY (ID\_STRUMENTO) REFERENCES STRUMENTO(ID\_STRUMENTO) ON DELETE SET NULL,

    CONSTRAINT prenotazione\_fk\_utente FOREIGN KEY (USERNAME) REFERENCES UTENTE(USERNAME) ON DELETE SET NULL,

    --VINCOLO VALID\_PREN\_DURATA

    CONSTRAINT valid\_pren\_durata CHECK(DURATA BETWEEN 1 AND 24),

    --VINCOLO VALID\_PREN\_INIZIO

    CONSTRAINT valid\_pren\_inizio CHECK(DATAINIZIO > DATAPRENOTAZIONE)

);

4.2.9 Tabella: Utente

--TABLE: UTENTE

CREATE TABLE UTENTE(

    USERNAME VARCHAR(18) PRIMARY KEY,

    USR\_PASSWORD RAW(200) NOT NULL,

    EMAIL VARCHAR(100) NOT NULL,

--VINCOLO DI CHIAVE PRIMARIA

    CONSTRAINT utente\_pk PRIMARY KEY (USERNAME),

    --VINCOLO VALID\_EMAIL

    CONSTRAINT valid\_email\_usr CHECK (EMAIL LIKE '\_%@\_%.\_\_%'),

    --VINCOLO VALID\_USERNAME

    CONSTRAINT valid\_username CHECK (REGEXP\_LIKE(USERNAME, '[a-zA-Z0-9]{6}.\*')),

    --VINCOLO UNNIQ\_EMAIL

    CONSTRAINT uniq\_email UNIQUE(EMAIL)

);

4.3 Viste

4.3.1 Viste per il riepilogo mensile sulle prenotazioni di strumento

--VIEW: MOSTRA LE ORE DI UTILIZZO DI UNO STRUMENTO IN UN DETERMINATO MESE DA UN DETERMINATO UTENTE

CREATE OR REPLACE VIEW CONT\_UTILIZZO\_PER\_STRUMENTO\_MESE\_UTENTE(ID\_STRUMENTO, MESE, USERNAME, ORE\_UTILIZZO)

AS SELECT id\_strumento, mese, username, SUM(durata) AS ore\_utilizzo

FROM (

    SELECT id\_prenotazione, TO\_DATE(to\_char(datainizio, 'YYYY/MM'), 'YYYY/MM') AS mese, id\_strumento, username, durata

    FROM prenotazione

)

GROUP BY id\_strumento, mese, username;

--VIEW: MOSTRA L'UTENTE CHE HA UTILIZZATO PER PIU' ORE UN DETERMINATO STRUMENTO IN UN DETERMINATO MESE

CREATE OR REPLACE VIEW TOP\_UTILIZZATORE\_PER\_STRUMENTO\_MESE(ID\_STRUMENTO, MESE, USERNAME, ORE\_UTILIZZO)

AS SELECT p2.id\_strumento, p2.mese, p2.username, ore\_utilizzo

FROM(

    SELECT id\_strumento, mese, MAX(ore\_utilizzo) AS max\_ore\_utilizzo

    FROM cont\_utilizzo\_per\_strumento\_mese\_utente

    GROUP BY id\_strumento, mese

) p1

JOIN (

    SELECT id\_strumento, mese, username, ore\_utilizzo

    FROM cont\_utilizzo\_per\_strumento\_mese\_utente

) p2

ON p1.id\_strumento = p2.id\_strumento

AND p1.mese = p2.mese

WHERE p1.max\_ore\_utilizzo = p2.ore\_utilizzo;

--VIEW: MOSTRA LE ORE DI UTILIZZO TOTALI PER UNO STRUMENTO IN UN DETERMINATO MESE

CREATE OR REPLACE VIEW CONT\_UTILIZZO\_PER\_STRUMENTO\_MESE (ID\_STRUMENTO, MESE, ORE\_UTILIZZO)

AS SELECT id\_strumento, mese, SUM(ore\_utilizzo)

FROM cont\_utilizzo\_per\_strumento\_mese\_utente

GROUP BY id\_strumento, mese;

--VIEW: MOSTRA IL RIEPILOGO MENSILE DEL TEMPO DI UTILIZZO DI UNO STRUMENTO (IL NUMERO TOTALE DI ORE E L'UTENTE CON PIU'

--ORE DI UTILIZZO PER UNO STRUMENTO IN UN DETERMINATO MESE)

CREATE OR REPLACE VIEW RIEPILOGO\_UTILIZZO\_STRUMENTO\_MESE (ID\_STRUMENTO, MESE, TOP\_UTENTE, ORE\_UTILIZZO\_TOP\_UTENTE, ORE\_UTILIZZO\_TOTALI)

AS SELECT DISTINCT p1.id\_strumento, p1.mese, p2.username, p2.ore\_utilizzo, p1.ore\_utilizzo

FROM cont\_utilizzo\_per\_strumento\_mese p1

JOIN cont\_utilizzo\_per\_strumento\_mese\_utente p2

ON p1.id\_strumento = p2.id\_strumento

AND p1.mese = p2.mese

WHERE p2.username IN (

    SELECT username

    FROM top\_utilizzatore\_per\_strumento\_mese

    WHERE id\_strumento = p1.id\_strumento

    AND mese = p1.mese

);

4.3.2 Viste per il riepilogo annuale sulle prenotazioni di strumento

--VIEW: MOSTRA LE ORE DI UTILIZZO DI UNO STRUMENTO IN UN DETERMINATO ANNO DA UN DETERMINATO UTENTE

CREATE OR REPLACE VIEW CONT\_UTILIZZO\_PER\_STRUMENTO\_ANNO\_UTENTE( ID\_STRUMENTO, ANNO, USERNAME, ORE\_UTILIZZO)

AS SELECT id\_strumento, anno, username, SUM(durata) AS ore\_utilizzo

FROM(

    SELECT id\_prenotazione, TO\_DATE(to\_char(datainizio, 'YYYY'), 'YYYY') AS anno, id\_strumento, username, durata

    FROM prenotazione

)

GROUP BY id\_strumento, anno, username;

--VIEW: MOSTRA L'UTENTE CHE HA UTILIZZATO PER PIU' ORE UN DETERMINATO STRUMENTO IN UN DETERMINATO ANNO

CREATE OR REPLACE VIEW TOP\_UTILIZZATORE\_PER\_STRUMENTO\_ANNO(ID\_STRUMENTO, ANNO, USERNAME, ORE\_UTILIZZO)

AS SELECT p2.id\_strumento, p2.anno, p2.username, ore\_utilizzo

FROM(

    SELECT id\_strumento, anno, MAX(ore\_utilizzo) AS max\_ore\_utilizzo

    FROM cont\_utilizzo\_per\_strumento\_anno\_utente

    GROUP BY id\_strumento, anno

) p1

JOIN (

    SELECT id\_strumento, anno, username, ore\_utilizzo

    FROM cont\_utilizzo\_per\_strumento\_anno\_utente

) p2

ON p1.id\_strumento = p2.id\_strumento

AND p1.anno = p2.anno

WHERE p1.max\_ore\_utilizzo = p2.ore\_utilizzo;

--VIEW: MOSTRA LE ORE DI UTILIZZO TOTALI PER UNO STRUMENTO IN UN DETERMINATO ANNO

CREATE OR REPLACE VIEW CONT\_UTILIZZO\_PER\_STRUMENTO\_ANNO (ID\_STRUMENTO, ANNO, ORE\_UTILIZZO)

AS SELECT id\_strumento, anno, SUM(ore\_utilizzo)

FROM cont\_utilizzo\_per\_strumento\_anno\_utente

GROUP BY id\_strumento, anno;

--VIEW: MOSTRA IL RIEPILOGO ANNUALE DEL TEMPO DI UTILIZZO DI UNO STRUMENTO (IL NUMERO TOTALE DI ORE E L'UTENTE CON PIU'

--ORE DI UTILIZZO PER UNO STRUMENTO IN UN DETERMINATO ANNO)

CREATE OR REPLACE VIEW RIEPILOGO\_UTILIZZO\_STRUMENTO\_ANNO (ID\_STRUMENTO, ANNO, TOP\_UTENTE, ORE\_UTILIZZO\_TOP\_UTENTE, ORE\_UTILIZZO\_TOTALI)

AS SELECT DISTINCT p1.id\_strumento, p1.anno, p2.username, p2.ore\_utilizzo, p1.ore\_utilizzo

FROM cont\_utilizzo\_per\_strumento\_anno p1

JOIN cont\_utilizzo\_per\_strumento\_anno\_utente p2

ON p1.id\_strumento = p2.id\_strumento

AND p1.anno = p2.anno

WHERE p2.username IN (

    SELECT username

    FROM top\_utilizzatore\_per\_strumento\_anno

    WHERE id\_strumento = p1.id\_strumento

    AND anno = p1.anno

);

4.4 Funzioni e Procedure

4.4.1 Funzione per l’encrypt di una password di un utente

--FUNCTION: CRITTOGRAFA LA PASSWORD UTILIZZANDO L'ALGORITMO AES128

CREATE OR REPLACE FUNCTION encrypt\_pwd(pwd VARCHAR2)

RETURN RAW AS

chiave RAW(16) := '1234567890123456';

modulo NUMBER := DBMS\_CRYPTO.ENCRYPT\_AES128 + DBMS\_CRYPTO.CHAIN\_CBC + DBMS\_CRYPTO.PAD\_PKCS5;

encrypted\_raw RAW(2000);

l\_return RAW(2000);

BEGIN

    encrypted\_raw := DBMS\_CRYPTO.ENCRYPT(UTL\_I18N.STRING\_TO\_RAW(pwd, 'AL32UTF8'), modulo, UTL\_I18N.STRING\_TO\_RAW(chiave, 'AL32UTF8'));

    RETURN encrypted\_raw;

END encrypt\_pwd;

/

4.4.2 Funzione per il decrypt di una password di un utente

--FUNCTION: DECRIPTA LA PASSWORD

CREATE OR REPLACE FUNCTION decrypt\_pwd(pwd RAW)

RETURN VARCHAR2 AS

chiave RAW(16) := '1234567890123456';

modulo NUMBER := DBMS\_CRYPTO.ENCRYPT\_AES128 + DBMS\_CRYPTO.CHAIN\_CBC + DBMS\_CRYPTO.PAD\_PKCS5;

decrypted\_raw RAW(2000);

decrypted\_pwd VARCHAR2(2000);

BEGIN

    decrypted\_raw := DBMS\_CRYPTO.DECRYPT (pwd, modulo, UTL\_I18N.STRING\_TO\_RAW(chiave, 'AL32UTF8'));

    decrypted\_pwd := UTL\_RAW.CAST\_TO\_VARCHAR2(decrypted\_raw);

    RETURN decrypted\_pwd;

END decrypt\_pwd;

/

4.4.3 Funzione per il controllo delle credenziali di login

--FUNCTION: CONTROLLA LE CREDENZIALI DURANTE IL LOGIN

CREATE OR REPLACE FUNCTION autenticazione(usrname UTENTE.USERNAME%TYPE, pwd VARCHAR2)

RETURN BOOLEAN AS

tmp UTENTE.USERNAME%TYPE;

BEGIN

SELECT USERNAME INTO tmp

FROM UTENTE

WHERE usrname = UTENTE.USERNAME AND decrypt\_pwd(UTENTE.USR\_PASSWORD) = LTRIM(RTRIM(pwd));

RETURN TRUE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

END autenticazione;

/

4.4.4 Funzione per il controllo della validita’ di una password

--FUNCTION: CONTROLLA SE LA PASSWORD RISPETTA IL VINCOLO VALID\_PW

CREATE OR REPLACE FUNCTION controlla\_pw (pw VARCHAR2)

RETURN BOOLEAN AS

BEGIN

    IF(REGEXP\_INSTR(pw, '[[:digit:]]')>=1 AND REGEXP\_LIKE(pw, '\w{5,}') AND NOT REGEXP\_LIKE(pw, '[[:blank:]]'))

    THEN

    RETURN TRUE;

    ELSE

    RETURN FALSE;

    END IF;

END controlla\_pw;

/

4.5 Trigger

4.5.1 Trigger NoOverlapPren

--TRIGGER: IMPLEMENTA IL VINCOLO NO\_OVERLAP\_PREN

CREATE OR REPLACE TRIGGER no\_overlap\_pren

BEFORE INSERT OR UPDATE ON PRENOTAZIONE

FOR EACH ROW

DECLARE

    strumento\_prenotato EXCEPTION;

    CURSOR curr IS

    SELECT P.DATAINIZIO, P.DURATA

    FROM PRENOTAZIONE P

    WHERE P.ID\_STRUMENTO = :NEW.ID\_STRUMENTO

    AND P.ID\_PRENOTAZIONE != :NEW.ID\_PRENOTAZIONE

    ORDER BY DATAINIZIO ASC;

    c curr%ROWTYPE;

    PRAGMA AUTONOMOUS\_TRANSACTION;

BEGIN

    OPEN curr;

    LOOP

        FETCH curr INTO c;

        EXIT WHEN curr%NOTFOUND;

        IF((:NEW.DATAINIZIO >= c.DATAINIZIO AND :NEW.DATAINIZIO < c.DATAINIZIO+(1/24\*c.DURATA))

            OR

           (:NEW.DATAINIZIO+(1/24\*:NEW.DURATA) > c.DATAINIZIO AND :NEW.DATAINIZIO+(1/24\*:NEW.DURATA) <= c.DATAINIZIO+(1/24\*c.DURATA)))

        THEN

            RAISE strumento\_prenotato;

        END IF;

    END LOOP;

    CLOSE curr;

    COMMIT;

EXCEPTION

    WHEN strumento\_prenotato THEN

        RAISE\_APPLICATION\_ERROR(-20004, 'Strumento gia'' prenotato');

END no\_overlap\_pren;

/

4.5.2 Trigger DeleteOrModifyPren

--TRIGGER: IMPLEMENTA IL VINCOLO DELETE\_OR\_MODIFY\_PREN

CREATE OR REPLACE TRIGGER delete\_or\_modify\_pren

BEFORE DELETE OR UPDATE ON PRENOTAZIONE

FOR EACH ROW

DECLARE

    curr\_date DATE;

    modifica\_non\_valida EXCEPTION;

    elim\_non\_valida EXCEPTION;

BEGIN

    curr\_date := SYSDATE;

    IF(:OLD.DATAINIZIO <= curr\_date) THEN

        IF DELETING THEN

            RAISE elim\_non\_valida;

        END IF;

        IF UPDATING THEN

IF :NEW.USERNAME IS NOT NULL THEN

            RAISE modifica\_non\_valida;

END IF;

        END IF;

    END IF;

EXCEPTION

    WHEN modifica\_non\_valida THEN

        RAISE\_APPLICATION\_ERROR(-20001, 'Impossibile modificare prenotazione passata');

    WHEN elim\_non\_valida THEN

        RAISE\_APPLICATION\_ERROR(-20002, 'Impossibile eliminare prenotazione passata');

END delete\_or\_modify\_pren;

/

4.5.3 Trigger InserisciPw

--TRIGGER: IMPLEMENTA IL VINCOLO VALID\_PW ED ESEGUE LA CRITTOGRAFIA DELLA PASSWORD

CREATE OR REPLACE TRIGGER inserisci\_pw

BEFORE INSERT OR UPDATE ON UTENTE

FOR EACH ROW

DECLARE

    password\_non\_valida EXCEPTION;

    psswd VARCHAR2(1000);

    PRAGMA AUTONOMOUS\_TRANSACTION;

BEGIN

    psswd := UTL\_RAW.CAST\_TO\_VARCHAR2(:NEW.USR\_PASSWORD);

    IF(controlla\_pw(psswd))

    THEN

    :NEW.USR\_PASSWORD := encrypt\_pwd(psswd);

    ELSE

    RAISE password\_non\_valida;

    END IF;

    EXCEPTION

    WHEN password\_non\_valida THEN

    RAISE\_APPLICATION\_ERROR(-20003, 'La password non rispetta i criteri');

    COMMIT;

END inserisci\_pw;

/

4.5.4 Trigger NoDoublePren

--TRIGGER: IMPLEMENTA IL VINCOLO NO\_DOUBLE\_PREN

CREATE OR REPLACE TRIGGER no\_double\_pren

BEFORE INSERT OR UPDATE ON PRENOTAZIONE

FOR EACH ROW

FOLLOWS delete\_or\_modify\_pren

DECLARE

    doppia\_prenotazione EXCEPTION;

    CURSOR curr IS

    SELECT \*

    FROM PRENOTAZIONE P

    WHERE UPPER(P.USERNAME) = UPPER(:NEW.USERNAME)

    AND P.ID\_PRENOTAZIONE != :NEW.ID\_PRENOTAZIONE;

    c curr%ROWTYPE;

    PRAGMA AUTONOMOUS\_TRANSACTION;

BEGIN

    OPEN curr;

    LOOP

        FETCH curr INTO c;

        EXIT WHEN curr%NOTFOUND;

        IF((:NEW.DATAINIZIO >= c.DATAINIZIO AND :NEW.DATAINIZIO < c.DATAINIZIO+(1/24\*c.DURATA))

            OR

           (:NEW.DATAINIZIO+(1/24\*:NEW.DURATA) > c.DATAINIZIO AND :NEW.DATAINIZIO+(1/24\*:NEW.DURATA) <= c.DATAINIZIO+(1/24\*c.DURATA))

           AND :NEW.ID\_STRUMENTO<>c.ID\_STRUMENTO)

            THEN RAISE doppia\_prenotazione;

        END IF;

    END LOOP;

    CLOSE curr;

    COMMIT;

EXCEPTION

    WHEN doppia\_prenotazione THEN

        RAISE\_APPLICATION\_ERROR(-20005, 'Non e'' possibile prenotare due strumenti diversi alla stessa data e ora');

END no\_double\_pren;

/

Capitolo 5

Manuale d’uso

* 1. Popolamento del database

5.1.1 Popolamento della tabella Laboratorio

INSERT INTO LABORATORIO (NOME, DESCRIZIONE, CAMPO, ANNOFONDAZIONE) VALUES ('Laboratorio Newton', 'Il Laboratorio Newton progetta, sviluppa, produce e distribuisce dispositivi medici certificati, con particolare riferimento ad accessi vascolari e dispositivi medici monouso.', 'Bioingegneria', 1850);

INSERT INTO LABORATORIO (NOME, DESCRIZIONE, CAMPO, ANNOFONDAZIONE) VALUES ('Laboratorio Tesla', 'Nel Laboratorio Nazionale Tesla, moltissimi scienziati utilizzano l''esclusivo acceleratore di particelle, per sondare gli elementi costitutivi piu'' basilari della materia, aiutandoci a comprendere meglio queste particelle e le forze che le legano.', 'Chimica', 1923);

5.1.2 Popolamento della tabella Sede

INSERT INTO SEDE (INDIRIZZO, ID\_LAB)

VALUES ('Via Duomo 23 80138 Napoli', (SELECT ID\_LAB FROM LABORATORIO WHERE UPPER(LABORATORIO.NOME) LIKE ('%NEWTON%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO SEDE (INDIRIZZO, ID\_LAB)

VALUES ('Via Scarlatti 77 80131 Napoli', (SELECT ID\_LAB FROM LABORATORIO WHERE UPPER(LABORATORIO.NOME) LIKE ('%TESLA%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO SEDE (INDIRIZZO, ID\_LAB)

VALUES ('Via dei mille 43 80132 Napoli', (SELECT ID\_LAB FROM LABORATORIO WHERE UPPER(LABORATORIO.NOME) LIKE ('%NEWTON%') FETCH NEXT 1 ROWS ONLY));

5.1.3 Popolamento della tabella Postazione

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('A1', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('A2', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('A3', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('B1', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('B2', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('B3', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('A1', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('A2', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('B1', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('B2', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('A1', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DEI MILLE%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO POSTAZIONE (NOME, ID\_SEDE)

VALUES ('B1', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DEI MILLE%') FETCH NEXT 1 ROWS ONLY));

5.1.4 Popolamento della tabella Responsabile

INSERT INTO RESPONSABILE (MATRICOLA,NOME,COGNOME,DATANASCITA,CODICEFISCALE,INDIRIZZO,TELEFONO1,EMAIL,ID\_SEDE)

VALUES ('R123456789', 'Roberto', 'Rossi', DATE '1969-11-24', 'RSSRRT69E01G234I', 'Via Roma 11', '3341234567', 'rob.rossi@email.it', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO RESPONSABILE (MATRICOLA,NOME,COGNOME,DATANASCITA,CODICEFISCALE,INDIRIZZO,TELEFONO1,EMAIL,ID\_SEDE)

VALUES ('R987654321', 'Michele', 'Verdi', DATE '1982-06-11', 'VRDMHL82E01G455X', 'Vico nuovo 33', '3345678901', 'mitch.green@email.it', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO RESPONSABILE (MATRICOLA,NOME,COGNOME,DATANASCITA,CODICEFISCALE,INDIRIZZO,TELEFONO1,EMAIL,ID\_SEDE)

VALUES ('R010156789', 'Massimo', 'De Angelis', DATE '1974-02-15', 'DNGMSM72D01H581V', 'Piazza san Marco 11', '3678967890', 'max.ang@email.it', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DEI MILLE%') FETCH NEXT 1 ROWS ONLY));

* + 1. Popolamento della tabella Tecnico

INSERT INTO TECNICO (MATRICOLA,NOME,COGNOME,DATANASCITA,CODICEFISCALE,INDIRIZZO,TELEFONO1,TELEFONO2,EMAIL,ID\_SEDE)

VALUES ('T000000001', 'Nicola', 'Esposito', DATE '1994-01-15', 'SPSNCL94D10H581A', 'Via Marina 10', '3448567810','0812345678', 'nick.exp@email.it', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO TECNICO (MATRICOLA,NOME,COGNOME,DATANASCITA,CODICEFISCALE,INDIRIZZO,TELEFONO1,EMAIL,ID\_SEDE)

VALUES ('T000000002', 'Maria', 'Gentile', DATE '1992-04-18', 'GNTMRA92D58D947F', 'Via Armando Diaz 3', '3348261110', 'mary.kind@email.it', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO TECNICO (MATRICOLA,NOME,COGNOME,DATANASCITA,CODICEFISCALE,INDIRIZZO,TELEFONO1,EMAIL,ID\_SEDE)

VALUES ('T000000003', 'Lucia', 'Cecchi', DATE '1988-12-15', 'CCCLCU88T55M185K', 'Via Medina 17', '3311564010', 'lucy.lu@email.it', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO TECNICO (MATRICOLA,NOME,COGNOME,DATANASCITA,CODICEFISCALE,INDIRIZZO,TELEFONO1,EMAIL,ID\_SEDE)

VALUES ('T000000004', 'Marco', 'De Luca', DATE '1987-11-15', 'DLCMRC87S15I082P', 'Via Giulio Cesare 2', '3771824344', 'mark.gp@email.it', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO TECNICO (MATRICOLA,NOME,COGNOME,DATANASCITA,CODICEFISCALE,INDIRIZZO,TELEFONO1,EMAIL,ID\_SEDE)

VALUES ('T000000005', 'Giulio', 'Mattei', DATE '1990-05-02', 'MTTGLI90E02I082W', 'Piazza medaglie d''oro 22', '3452276344', 'jul.best@email.it', (SELECT ID\_SEDE FROM SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DEI MILLE%') FETCH NEXT 1 ROWS ONLY));

5.1.6 Popolamento della tabella Strumento

--SCHEDA TECNICA -> 'Marca: , Dimensioni: cm x cm x cm, Voltaggio: V, Potenza: W'

--SOTTOSEZIONE SEDE 'VIA DUOMO'

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Microscopio Elettronico', 'Marca: ArgonTech, Dimensioni: 60cm x 50cm x 20cm, Peso: 3.0Kg, Voltaggio: 100V, Potenza: 10W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') AND NOME='A1'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Centrifuga', 'Marca: KITSeft, Dimensioni: 30cm x 30m x 50cm, Peso: 2.5Kg, Voltaggio: 220V, Potenza: 25W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') AND  NOME='A1'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Miscelatore', 'Marca: KITSeft, Dimensioni: 20cm x 20cm x 15cm, Peso: 3.7Kg, Voltaggio: 12V, Potenza: 3W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') AND  NOME='A2'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Bilancia di precisione', 'Marca: ArgonTech, Dimensioni: 20cm x 20cm x 5cm, Peso: 6.0Kg, Voltaggio: 12V, Potenza: 3W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') AND  NOME='A2'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Agitatore magnetico', 'Marca: ArgonTech, Dimensioni: 23cm x 23cm x 10cm, Peso: 4.7Kg, Voltaggio: 120V, Potenza: 20W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') AND  NOME='A2'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Cappa chimica', 'Marca: LabProto, Dimensioni: 60cm x 180cm x 1cm, Peso: 0.5Kg', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') AND  NOME='A3'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Glove box', 'Marca: LabProto, Dimensioni: 20cm x 10cm x 10cm, Peso: 0.5Kg', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') AND  NOME='B1'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Pompa per vuoto', 'Marca: KITSeft, Dimensioni: 37cm x 17cm x 28cm, Peso: 12.0Kg, Voltaggio: 220V, Potenza: 50W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DUOMO%') AND  NOME='B2'));

--SOTTOSEZIONE SEDE 'VIA SCARLATTI'

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Oscilloscopio', 'Marca: KITSeft, Dimensioni: 15cm x 10cm x 5cm, Peso: 1.0Kg, Voltaggio: 30V, Potenza: 5W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') AND  NOME='A1'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Alimentatore da banco', 'Marca: LabProto, Dimensioni: 30cm x 20cm x 5cm, Peso: 1.3Kg, Voltaggio: 120V, Potenza: 12W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') AND  NOME='A1'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Generatore di forme d''onda', 'Marca: LabProto, Dimensioni: 20cm x 10cm x 5cm, Peso: 0.8Kg, Voltaggio: 12V, Potenza: 3W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') AND  NOME='A2'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Saldatore di precisione', 'Marca: ArgonTech, Dimensioni: 10cm x 1cm x 1cm, Peso: 0.7Kg, Voltaggio: 12V, Potenza: 3W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA SCARLATTI%') AND  NOME='A2'));

--SOTTOSEZIONE SEDE 'VIA DEI MILLE'

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Microscopio Elettronico', 'Marca: ArgonTech, Dimensioni: 40cm x 30cm x 70cm, Peso: 3.0Kg, Voltaggio: 220V, Potenza: 12W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DEI MILLE%') AND  NOME='A1'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Piastra riscaldante', 'Marca: KITSeft, Dimensioni: 30cm x 30cm x 5cm, Peso: 2.0Kg, Voltaggio: 120V, Potenza: 5W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DEI MILLE%') AND  NOME='B1'));

INSERT INTO STRUMENTO (DESCRIZIONE, SCHEDATECNICA, ID\_POSTAZIONE)

VALUES ('Incubatore', 'Marca: ArgonTech, Dimensioni: 50cm x 50cm x 50cm, Peso: 6.0Kg, Voltaggio: 220V, Potenza: 20W', (SELECT ID\_POSTAZIONE FROM POSTAZIONE JOIN SEDE ON POSTAZIONE.ID\_SEDE=SEDE.ID\_SEDE WHERE UPPER(SEDE.INDIRIZZO) LIKE ('%VIA DEI MILLE%') AND  NOME='B1'));

5.1.7 Popolamento della tabella Utente

INSERT INTO UTENTE(USERNAME, USR\_PASSWORD, EMAIL)

VALUES ('cicciogamer89', (SELECT UTL\_I18N.STRING\_TO\_RAW(LTRIM(RTRIM('paguro89')), 'AL32UTF8') FROM DUAL), 'cicciogamer89@paguro.it');

INSERT INTO UTENTE(USERNAME, USR\_PASSWORD, EMAIL)

VALUES ('MadScientist', (SELECT UTL\_I18N.STRING\_TO\_RAW(LTRIM(RTRIM('Einstein111')), 'AL32UTF8') FROM DUAL), 'albertEinstein22@mail.it');

5.1.8 Popolamento della tabella Prenotazione

INSERT INTO PRENOTAZIONE(DATAINIZIO, DURATA, USERNAME, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/05/10 17:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 3,'cicciogamer89', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%OSCILLOSCOPIO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DATAINIZIO, DURATA, USERNAME, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/07/10 08:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 4, 'cicciogamer89', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%MICROSCOPIO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DATAINIZIO, DURATA, USERNAME, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/10/10 23:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 2, 'cicciogamer89', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%PIASTRA%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DATAINIZIO, DURATA, USERNAME, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/03/10 13:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 3, 'cicciogamer89', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%SALDATORE%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DATAINIZIO, DURATA, USERNAME, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/03/10 02:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 5, 'MadScientist', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%CAPPA%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DATAINIZIO, DURATA, USERNAME, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/01/10 12:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 2, 'MadScientist', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%AGITATORE%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DATAINIZIO, DURATA, USERNAME, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/12/10 11:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 1, 'MadScientist', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%BILANCIA%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DATAINIZIO, DURATA, USERNAME, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/08/10 09:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 3, 'MadScientist', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%OSCILLOSCOPIO%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DataInizio, Durata, Username, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/09/10 07:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 3, 'MadScientist', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%PIASTRA%') FETCH NEXT 1 ROWS ONLY));

INSERT INTO PRENOTAZIONE(DataInizio, Durata, Username, ID\_STRUMENTO)

VALUES ((SELECT TO\_DATE('2025/07/12 07:30', 'YYYY/MM/DD HH24:MI') FROM DUAL), 5, 'MadScientist', (SELECT ID\_STRUMENTO FROM STRUMENTO WHERE UPPER(STRUMENTO.DESCRIZIONE) LIKE ('%MICROSCOPIO%') FETCH NEXT 1 ROWS ONLY));