Documentazione Database

1. Introduzione

Nelle seguenti pagine si riporta la documentazione relativa alla progettazione ed implementazione di una base di dati che sarà da appoggio ad un’applicazione web per la gestione di interventi di manutenzione in più siti. Tale applicazione terrà conto di molti fattori, tra cui le attività ordinarie e straordinarie, i materiali utilizzati, il tipo di manutenzione e il luogo dove avverrà, i manutentori e le skill necessarie per intraprendere un’attività. La lista completa delle feature richieste può essere consultata nel documento SRS relativo al progetto, quella delle feature implementate è invece disponibile nella documentazione dell’applicazione. Il database implementato è di tipo relazionale ed è stato sviluppato tramite PostgreSQL v12.

1. Progettazione del Database

Lo sviluppo del database è stato effettuato tenendo conto dei requisiti messi a disposizione dal documento SRS. Una volta identificati i dati da memorizzare e focalizzata l’attenzione sul significato dei dati stessi, senza scendere nei dettagli implementativi, è stato redatto il modello concettuale dei dati, in particolare si è scelto di utilizzare un diagramma ER, ossia di tipo entità-relazione. Si è passato quindi ad identificare le entità e le relazioni tra essi. Una volta redatto lo schema E-R, ristrutturato ed effettuate le analisi delle ridondanze, si è passato alla traduzione in schema logico.

1. Lo schema logico

Segue la lista delle tabelle di cui è composto il database con una breve descrizione dei contenuti.

La tabella **System\_user** identifica l’utente della piattaforma che ha un ruolo (planner, maintener, DBloader), un’email, nome, cognome, codice fiscale, username (assegnato in modo automatico alla registrazione) e password.

L’accesso alla piattaforma è registrato nella tabella **Access\_log** che tiene traccia degli accessi alla piattaforma da parte di tutti gli utenti registrandone l’orario d’accesso.

La tabella **Log\_access** mette in relazione Access\_log e System\_user.

La tabella **Days\_of\_week** identifica il giorno lavorativo, contiene infatti le varie fasce orarie, il giorno, la settimana e l’utente a cui è associata. Gli slot contengono i minuti di lavoro assegnati in quello slot.

La tabella **Competences** identifica le competenze che possono avere gli utenti e che possono essere associate ad una determinata attività.

La tabella **Ownership** mette in relazione gli utenti e le competenze, rappresentando le competenze appartenenti agli utenti.

La tabella **Request** mette in relazione le attività di manutenzione e le competenze, rappresentando le competenze richieste per svolgerla.

La tabella **Assignment\_table** mette in relazione gli utenti e le attività di manutenzione, rappresentando l’assegnazione dell’attività.

La tabella **Maintenance\_activity** rappresenta l’attività di manutenzione e contiene la sua descrizione, il suo ID, il tempo stimato in minuti, la settimana in cui deve essere eseguita, se è un’attività straordinaria, di tipo extra, se è interrompibile, il pdf della procedura standard di esecuzione.

La tabella **Maintenance\_type** rappresenta i vari tipi di manutenzione che possono essere associati ad una attività di manutenzione.

La tabella **Ticket** rappresenta i dati relativi al ticket di una attività di manutenzione. Contiene i dati relativi allo status dell’esecuzione.

La tabella **Materials** rappresenta i materiali che possono essere utilizzati in un’attività di manutenzione, classificati tramite nome e quantità.

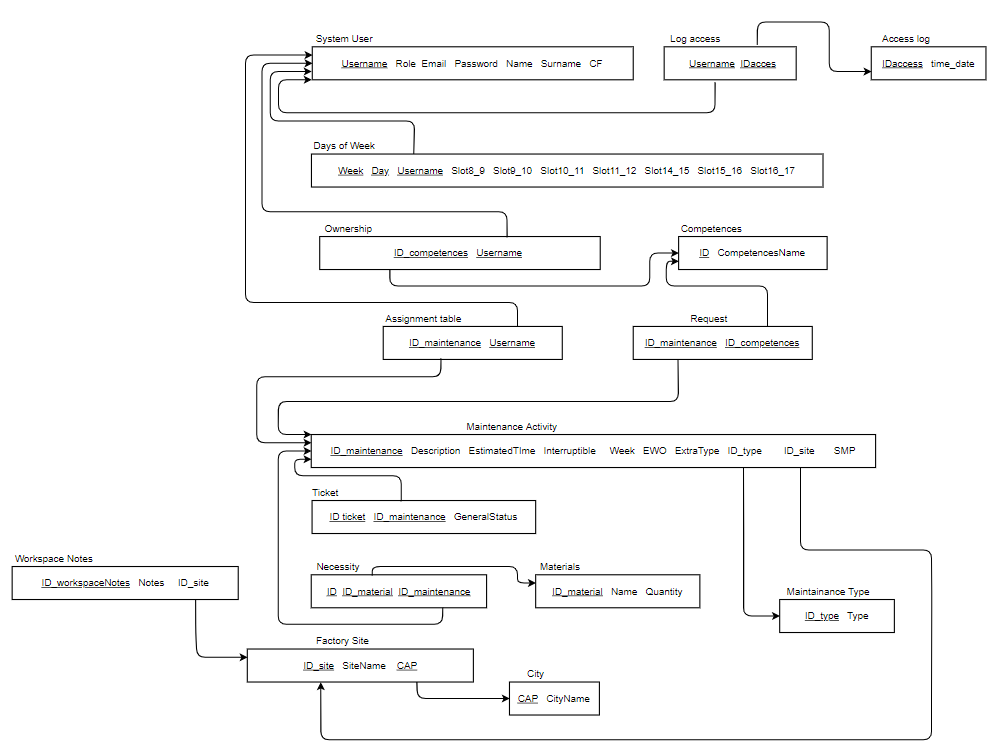
La tabella **Necessity** mette in relazione l’attività di manutenzione e i materiali, rendendo associabili più materiali a più manutenzioni.

La tabella **City** contiene una lista delle città presenti nell’applicazione.

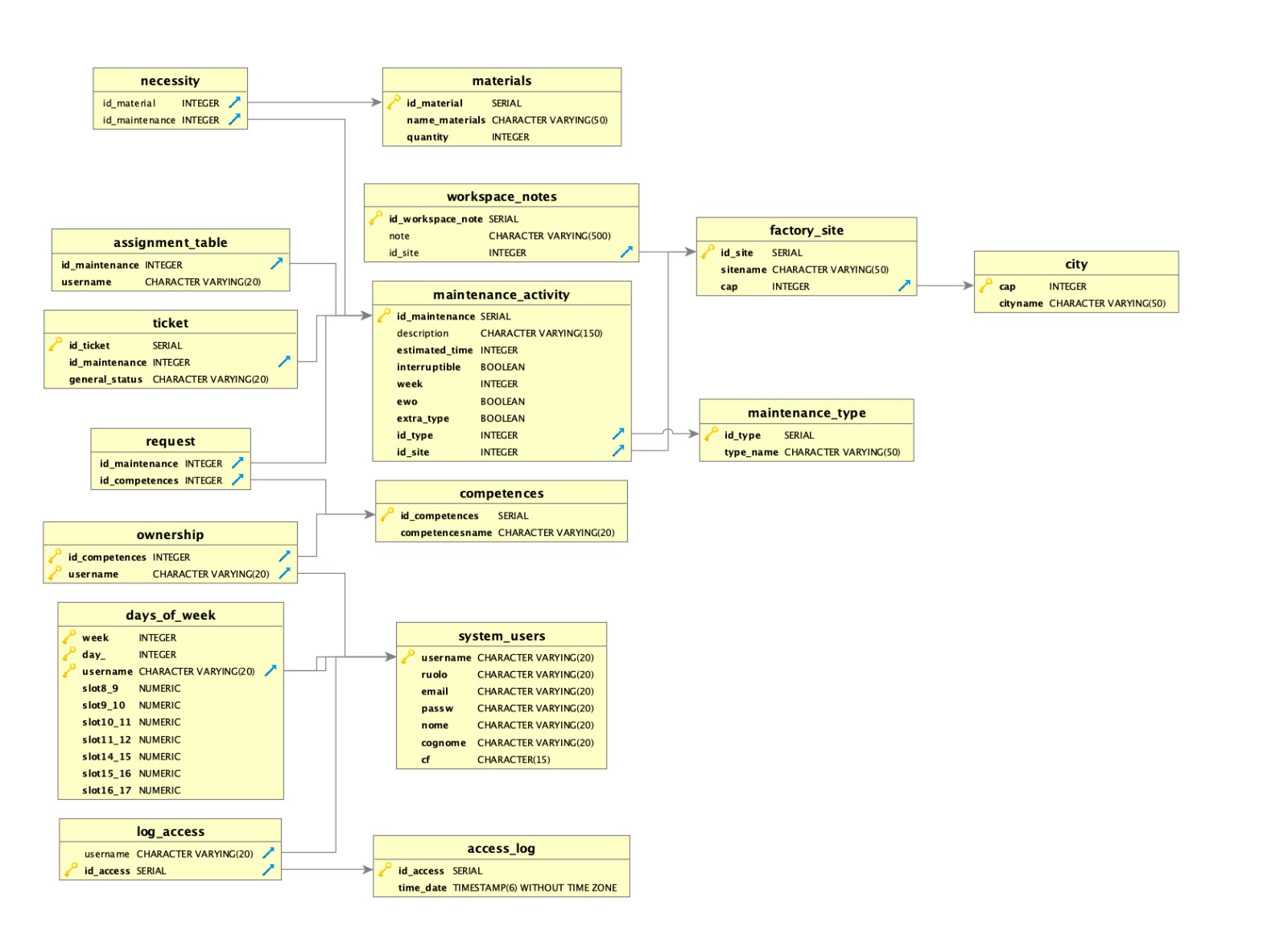
La tabella **Factory\_site** contiene il nome del sito presente nella determinata città e viene associato all’attività di manutenzione.

La tabella **Workspace\_notes** contiene le note associate ad un determinato sito.

Di seguito è riportato lo schema logico del Database.



*(il disegno è stato eseguito su draw.io)*



*(Il disegno è stato prodotto tramite DB visualizer)*

1. Script di creazione del database

Di seguito è riportato lo script di creazione del database. È presente anche la creazione di un utente chiamato System\_admin tramite il quale ci interfacceremo al database lato client.

--------------SYSTEM USERS-------------------

DROP TABLE IF EXISTS system\_users CASCADE;

CREATE TABLE system\_users(

username varchar(20) PRIMARY KEY,

ruolo varchar(20) NOT NULL,

email varchar(20) NOT NULL,

passw varchar(20) NOT NULL,

nome varchar(20) NOT NULL,

cognome varchar(20) NOT NULL,

cf char(16) NOT NULL,

CONSTRAINT ck\_ruoli

CHECK (ruolo = 'Maintainer' or ruolo = 'Planner')

);

--------------ACCESS LOG-------------------

DROP SEQUENCE IF EXISTS id\_access\_log\_seq CASCADE;

CREATE SEQUENCE id\_access\_log\_seq;

DROP TABLE IF EXISTS access\_log CASCADE;

CREATE TABLE access\_log(

id\_access SERIAL PRIMARY KEY,

time\_date TIMESTAMP NOT NULL

);

--------------LOG ACCESS-------------------

DROP TABLE IF EXISTS log\_access CASCADE;

CREATE TABLE log\_access(

username varchar(20),

id\_access SERIAL,

CONSTRAINT pk\_log\_access PRIMARY KEY (id\_access),

CONSTRAINT unique\_log\_access UNIQUE (id\_access, username),

CONSTRAINT fk\_user\_sysus

FOREIGN KEY (username)

REFERENCES system\_users(username)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT fk\_id\_access\_access\_log

FOREIGN KEY (id\_access)

REFERENCES access\_log(id\_access)

ON DELETE RESTRICT

ON UPDATE CASCADE

);

--------------DAYS OF WEEK-------------------

DROP TABLE IF EXISTS days\_of\_week CASCADE;

CREATE TABLE days\_of\_Week(

week INT NOT NULL,

day\_ INT NOT NULL,

username Varchar(20) REFERENCES system\_users(username),

Slot8\_9 DECIMAL NOT NULL DEFAULT 0,

Slot9\_10 DECIMAL NOT NULL DEFAULT 0,

Slot10\_11 DECIMAL NOT NULL DEFAULT 0,

Slot11\_12 DECIMAL NOT NULL DEFAULT 0,

Slot14\_15 DECIMAL NOT NULL DEFAULT 0,

Slot15\_16 DECIMAL NOT NULL DEFAULT 0,

Slot16\_17 DECIMAL NOT NULL DEFAULT 0,

CONSTRAINT pk\_DaysOfWeek PRIMARY KEY (week, day\_, username),

CONSTRAINT fk\_user\_sysus

FOREIGN KEY (username)

REFERENCES system\_users(username)

);

--------------COMPETENCES-------------------

DROP SEQUENCE IF EXISTS id\_competences\_seq CASCADE;

CREATE SEQUENCE id\_competences\_seq;

DROP TABLE IF EXISTS competences CASCADE;

CREATE TABLE competences(

id\_competences SERIAL,

competencesName Varchar(20) NOT NULL,

CONSTRAINT pk\_competences PRIMARY KEY (id\_competences),

CONSTRAINT unique\_competences UNIQUE (id\_competences, competencesName)

);

--------------OWNERSHIP-------------------

DROP TABLE IF EXISTS ownership CASCADE;

CREATE TABLE ownership(

id\_competences INT,

username Varchar(20),

CONSTRAINT pk\_Ownership PRIMARY KEY (id\_competences, username),

CONSTRAINT fk\_id\_com\_compentences

FOREIGN KEY (id\_competences)

REFERENCES competences(id\_competences),

CONSTRAINT fk\_user\_sysus

FOREIGN KEY (username)

REFERENCES system\_users(username)

);

--------------MAINTENANCE TYPE-------------------

DROP SEQUENCE IF EXISTS id\_maintenance\_type\_seq CASCADE;

CREATE SEQUENCE id\_maintenance\_type\_seq;

DROP TABLE IF EXISTS maintenance\_type CASCADE;

CREATE TABLE maintenance\_type

(

id\_type SERIAL,

type\_name VARCHAR(50) NOT NULL,

CONSTRAINT pk\_Maintenance\_Type PRIMARY KEY(id\_type),

CONSTRAINT unique\_Maintenance\_Type\_Name UNIQUE(id\_type, type\_name)

);

DROP TABLE IF EXISTS city CASCADE;

CREATE TABLE city

(

cap INT,

cityname VARCHAR(50) NOT NULL,

CONSTRAINT pk\_CAP PRIMARY KEY(cap)

);

--------------FACTORY SITE-------------------

DROP SEQUENCE IF EXISTS id\_factory\_site\_seq CASCADE;

CREATE SEQUENCE id\_factory\_site\_seq;

DROP TABLE IF EXISTS factory\_Site CASCADE;

CREATE TABLE factory\_site

(

id\_site SERIAL,

sitename VARCHAR(50) NOT NULL,

cap INT NOT NULL,

CONSTRAINT pk\_Site PRIMARY KEY(id\_site),

CONSTRAINT unique\_Site UNIQUE(id\_site, sitename, cap),

CONSTRAINT fk\_CAP

FOREIGN KEY (cap)

REFERENCES city(cap)

ON DELETE RESTRICT

ON UPDATE CASCADE

);

--------------MAINTENANCE\_ACTIVITY-------------------

DROP SEQUENCE IF EXISTS id\_maintenance\_seq CASCADE;

CREATE SEQUENCE id\_maintenance\_seq;

DROP TABLE IF EXISTS maintenance\_activity CASCADE;

CREATE TABLE maintenance\_activity

(

id\_maintenance SERIAL,

description VARCHAR(150),

estimated\_time INT NOT NULL,

interruptible BOOLEAN NOT NULL,

week INT NOT NULL,

EWO BOOLEAN NOT NULL,

extra\_type BOOLEAN NOT NULL,

id\_type INT NOT NULL,

id\_site INT NOT NULL,

SMP bytea,

CONSTRAINT pk\_maintenance\_activity PRIMARY KEY (id\_maintenance),

CONSTRAINT unique\_maintenances UNIQUE (id\_maintenance,description,estimated\_time,interruptible,week,EWO,extra\_type,id\_type,id\_site),

CONSTRAINT fk\_id\_type

FOREIGN KEY (id\_type)

REFERENCES maintenance\_type(id\_type)

ON UPDATE cascade

ON DELETE restrict,

CONSTRAINT fk\_id\_site

FOREIGN KEY (id\_site)

REFERENCES factory\_site(id\_site)

ON UPDATE cascade

ON DELETE restrict,

CONSTRAINT ck\_extra\_type

CHECK ((EWO = 'true' and extra\_type = 'false') or (EWO = 'true' and extra\_type = 'true') or (EWO = 'false' and extra\_type = 'false'))

);

--------------ASSIGNMENT TABLE-------------------

DROP TABLE IF EXISTS assignment\_table CASCADE;

CREATE TABLE assignment\_table

(

id\_maintenance INT NOT NULL,

username VARCHAR(20) NOT NULL,

CONSTRAINT unique\_assignment\_table UNIQUE (id\_maintenance, username),

CONSTRAINT fk\_id\_maintenance

FOREIGN KEY (id\_maintenance)

REFERENCES maintenance\_activity(id\_maintenance)

ON UPDATE CASCADE

ON DELETE RESTRICT

);

--------------REQUEST-------------------

DROP TABLE IF EXISTS request CASCADE;

CREATE TABLE request

(

id\_maintenance INT NOT NULL,

id\_competences INT NOT NULL,

CONSTRAINT unique\_request UNIQUE (id\_maintenance, id\_competences),

CONSTRAINT fk\_id\_maintenance

FOREIGN KEY (id\_maintenance)

REFERENCES maintenance\_activity(id\_maintenance)

ON UPDATE CASCADE

ON DELETE RESTRICT,

CONSTRAINT fk\_id\_competences

FOREIGN KEY (id\_competences)

REFERENCES competences(id\_competences)

ON UPDATE CASCADE

ON DELETE RESTRICT

);

--------------TICKET-------------------

DROP SEQUENCE IF EXISTS id\_ticket\_seq CASCADE;

CREATE SEQUENCE id\_ticket\_seq;

DROP TABLE IF EXISTS ticket CASCADE;

CREATE TABLE Ticket

(

id\_Ticket SERIAL,

id\_maintenance INT NOT NULL,

general\_status VARCHAR(20) NOT NULL,

CONSTRAINT pk\_ticket PRIMARY KEY (id\_ticket),

CONSTRAINT unique\_maintenance UNIQUE (id\_ticket,id\_maintenance, general\_status),

CONSTRAINT fk\_id\_maintenance

FOREIGN KEY (id\_maintenance)

REFERENCES maintenance\_activity(id\_maintenance)

ON UPDATE cascade

ON DELETE restrict,

CONSTRAINT ck\_extra\_type

CHECK (general\_status = 'not started' or general\_status = 'in progress' or general\_status = 'closed')

);

--------------MATERIALS-------------------

DROP SEQUENCE IF EXISTS id\_materials\_seq CASCADE;

CREATE SEQUENCE id\_materials\_seq;

DROP TABLE IF EXISTS materials CASCADE;

CREATE TABLE materials

(

id\_material SERIAL,

name\_materials VARCHAR(50) NOT NULL,

quantity INT NOT NULL,

CONSTRAINT pk\_Materials PRIMARY KEY(id\_material),

CONSTRAINT unique\_Material UNIQUE (id\_material, name\_materials, quantity)

);

--------------NECESSITY-------------------

DROP TABLE IF EXISTS necessity CASCADE;

CREATE TABLE necessity

(

id\_material INT,

id\_maintenance INT,

CONSTRAINT unique\_Necessity UNIQUE (id\_material, id\_maintenance),

CONSTRAINT fk\_ID\_Material

FOREIGN KEY (id\_material)

REFERENCES materials(id\_material)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT fk\_ID\_Maintenance

FOREIGN KEY (id\_maintenance)

REFERENCES maintenance\_activity(id\_maintenance)

ON DELETE RESTRICT

ON UPDATE CASCADE

);

--------------WORKSPACE NOTES-------------------

DROP SEQUENCE IF EXISTS id\_workspace\_seq CASCADE;

CREATE SEQUENCE id\_workspace\_seq;

DROP TABLE IF EXISTS workspace\_notes CASCADE;

CREATE TABLE workspace\_notes

(

id\_workspace\_note SERIAL,

note VARCHAR(500),

id\_site INT,

CONSTRAINT pk\_workspace\_notes PRIMARY KEY(id\_workspace\_note),

CONSTRAINT unique\_workspace\_notes UNIQUE(id\_workspace\_note, note, id\_site),

CONSTRAINT fk\_ID\_Site

FOREIGN KEY (id\_site)

REFERENCES factory\_site(id\_site)

ON DELETE RESTRICT

ON UPDATE CASCADE

);

DROP USER IF EXISTS System\_Admin;

CREATE USER System\_Admin WITH PASSWORD 'password';

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO System\_Admin;

GRANT USAGE,SELECT ON ALL SEQUENCES IN SCHEMA PUBLIC TO System\_Admin

Sono state create anche due viste d’appoggio che ci saranno utili per l’esecuzione di determinate query. Segue il codice.

CREATE VIEW n\_competenze\_richieste as

SELECT ma.id\_maintenance, count(competencesname) as n\_competenze\_richieste

FROM maintenance\_activity ma, request r, competences comp

WHERE ma.id\_maintenance = r.id\_maintenance

and r.id\_competences = comp.id\_competences

GROUP BY(ma.id\_maintenance)

ORDER BY ma.id\_maintenance;

Che restituisce il numero di competenze richieste per le varie attività

CREATE VIEW lista\_competenze\_richieste as

SELECT ma.id\_maintenance, comp.competencesname

FROM maintenance\_activity ma, request r, competences comp

WHERE ma.id\_maintenance = r.id\_maintenance and comp.id\_competences = r.id\_competences;

Che restituisce la lista di competenze necessarie per le varie attività

Segue inoltre un piccolo script di popolamento della base di dati utilizzato durante le fasi di progettazione e test dell’applicazione.

------------System Users-------------------------------------------------------

INSERT INTO system\_users(Username, ruolo, email, passw, nome, cognome, cf)

VALUES ('GiuseppeRossi1', 'Maintainer', 'Giuseppe1@libero.it', 'password', 'Giuseppe', 'Rossi', 'GSPRSS23R34CAA1');

INSERT INTO system\_users(Username, ruolo, email, passw, nome, cognome, cf)

VALUES ('MarcoVerdi2', 'Planner', 'Marco1@gmail.com', 'password', 'Marco', 'Verdi', 'MRCVRD22T32CRE5');

INSERT INTO system\_users(Username, ruolo, email, passw, nome, cognome, cf)

VALUES ('MarcoRossi3', 'Planner', 'Marco2@outlook.com', 'password', 'Marco', 'Rossi', 'MRCRSS25T31CRE3');

INSERT INTO system\_users(Username, ruolo, email, passw, nome, cognome, cf)

VALUES ('MarioNeri2', 'Planner', 'Mario1@gmail.com', 'password', 'Mario', 'Neri', 'MRINRD12T67CDA8');

------------Days of week-------------------------------------------------------

INSERT INTO days\_of\_Week(Week, day\_, username)

VALUES (22, 3, 'MarcoVerdi2' );

INSERT INTO days\_of\_week(Week, day\_, username)

VALUES (22, 4, 'MarcoVerdi2' );

INSERT INTO days\_of\_week(Week, day\_, username)

VALUES (22, 5, 'MarcoRossi3' );

INSERT INTO days\_of\_week(Week, day\_, username)

VALUES (22, 6, 'GiuseppeRossi1' );

INSERT INTO days\_of\_week(Week, day\_, username)

VALUES (22, 7, 'MarioNeri2' );

INSERT INTO days\_of\_week(Week, day\_, username)

VALUES (22, 7, 'MarcoVerdi2' );

------------Competences-------------------------------------------------------

INSERT INTO Competences(id\_competences,CompetencesName)

VALUES (1, 'Electronic Skills');

INSERT INTO Competences(CompetencesName)

VALUES ('Plumbing skills');

INSERT INTO Competences(CompetencesName)

VALUES ('Electric skills');

INSERT INTO Competences(CompetencesName)

VALUES ('C programming skills');

INSERT INTO Competences(CompetencesName)

VALUES ('SE skills');

------------Ownership-------------------------------------------------------

INSERT INTO Ownership(ID\_Competences, Username)

VALUES (1,'GiuseppeRossi1');

INSERT INTO Ownership(ID\_Competences, Username)

VALUES (3,'GiuseppeRossi1');

INSERT INTO Ownership(ID\_Competences, Username)

VALUES (2,'GiuseppeRossi1');

INSERT INTO Ownership(ID\_Competences, Username)

VALUES (2,'MarcoVerdi2');

INSERT INTO Ownership(ID\_Competences, Username)

VALUES (3,'MarcoRossi3');

INSERT INTO Ownership(ID\_Competences, Username)

VALUES (5,'MarcoRossi3');

INSERT INTO Ownership(ID\_Competences, Username)

VALUES (4,'MarioNeri2');

----------- Maintenance type --------------------------------------------------------------------

INSERT INTO maintenance\_type(id\_type, type\_name)

VALUES (1,'Electric');

INSERT INTO maintenance\_type(type\_name)

VALUES ('Hydraulic');

INSERT INTO maintenance\_type(type\_name)

VALUES ('Electronics');

INSERT INTO maintenance\_type(type\_name)

VALUES ('Mechanical');

----------- Città --------------------------------------------------------------------

INSERT INTO city (cap, cityname)

VALUES (80053, 'Castellammare di Stabia');

INSERT INTO city (cap, cityname)

VALUES (84084, 'Fisciano');

INSERT INTO city (cap, cityname)

VALUES (83040, 'Morra');

----------- Factory Sites --------------------------------------------------------------------

INSERT INTO factory\_site(id\_site, sitename, cap)

VALUES (1,'Molding', '84084');

INSERT INTO factory\_site(sitename, cap)

VALUES ('Carpentry', '83040');

INSERT INTO factory\_site(sitename, cap)

VALUES ('Painting', '80053');

INSERT INTO factory\_site(sitename, cap)

VALUES ('Molding', '80053');

----------- Maintenance Activity --------------------------------------------------------------------

INSERT INTO maintenance\_activity(id\_maintenance, description, estimated\_time, interruptible, week, ewo, extra\_type, id\_type, id\_site)

VALUES (1,'Bisogna riparare la centralina elettrica', 70, false, 23, true, false, 1, 2);

INSERT INTO maintenance\_activity(description, estimated\_time, interruptible, week, ewo, extra\_type, id\_type, id\_site)

VALUES ('Bisogna riparare il condotto idraulico principale, risultano piccole perdite', 130, true, 22, true, false, 2, 1);

------------Assignment Table-------------------------------------------------------

INSERT INTO assignment\_table(id\_maintenance, username)

VALUES (1,'MarioNeri2');

INSERT INTO assignment\_table(id\_maintenance, username)

VALUES (2,'MarcoRossi3');

------------Request-------------------------------------------------------

INSERT INTO request(id\_maintenance, ID\_Competences)

VALUES (1,1);

INSERT INTO request(id\_maintenance, ID\_Competences)

VALUES (2,2);

------------Ticket-------------------------------------------------------

INSERT INTO ticket(id\_ticket,id\_maintenance, general\_status)

VALUES (1, 1, 'closed');

INSERT INTO ticket(id\_maintenance, general\_status)

VALUES (2, 'not started');

----------- Materials --------------------------------------------------------------------

INSERT INTO materials(id\_material, name\_materials, quantity)

VALUES (1,'ferro', 400);

INSERT INTO materials(name\_materials, quantity)

VALUES ('acciaio', 250);

INSERT INTO materials(name\_materials, quantity)

VALUES ('rame', 300);

INSERT INTO materials(name\_materials, quantity)

VALUES ('legno', 100);

INSERT INTO materials(name\_materials, quantity)

VALUES ('ottone', 70);

INSERT INTO materials(name\_materials, quantity)

VALUES ('tungsteno', 20);

----------- Necessity --------------------------------------------------------------------

INSERT INTO necessity(id\_material, id\_maintenance)

VALUES (1, 2);

INSERT INTO necessity(id\_material, id\_maintenance)

VALUES (4, 1);

----------- Workspaces Notes --------------------------------------------------------------------

INSERT INTO workspace\_notes(id\_workspace\_note, note, id\_site)

VALUES (1,'L impianto chiude alle 17:30', 2);

INSERT INTO workspace\_notes(note, id\_site)

VALUES ('Limpianto apre alle 8:00',1 );

INSERT INTO workspace\_notes( note, id\_site)

VALUES ( '', 4);

GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA public TO system\_admin;

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO System\_Admin;

GRANT ALL ON SCHEMA public TO system\_admin