UNIVERSITÀ DEGLI STUDI DI SALERNO

Atzeni, Ceri, Fraternali, Paraboschi, Torlone
Basi di dati *Quinta edizione*
McGraw-Hill Education, 2018

1

BASI DI DATI

Esercitazione
Concetti Base SQL-DDL – sesta parte

DIEM
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE ED ELETTRICA E MATEMATICA APPLICATA

Matteo Gaeta
Full Professor – Senior Member IEEE

Ognun vede quel che tu pari. Pochi sentono quel che tu sei.
(Niccolò Machiavelli)

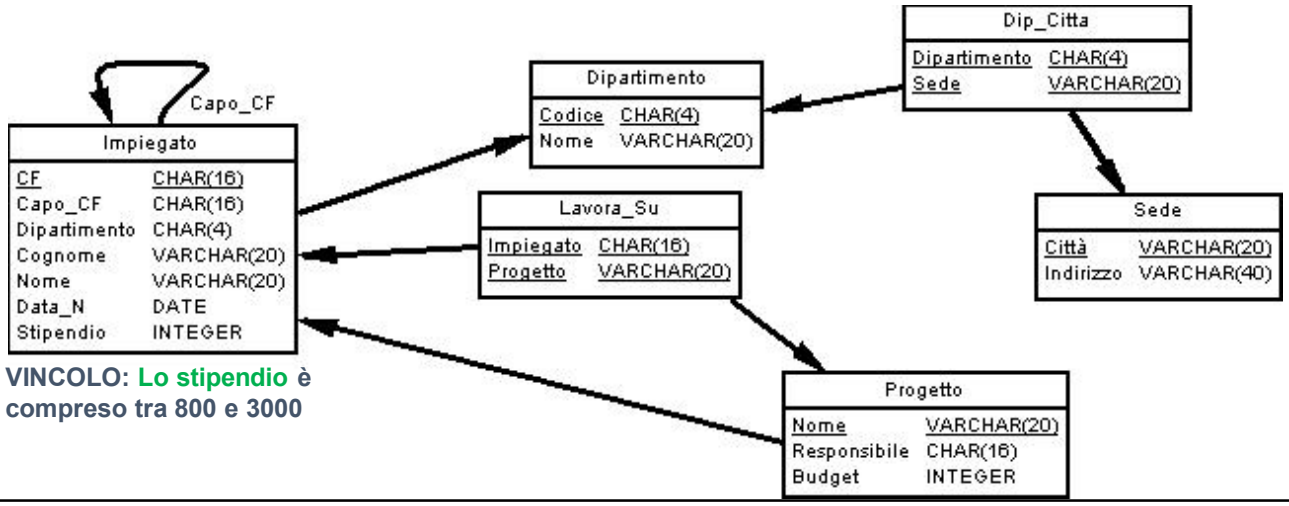
BASI DI DATI

Concetti base – SQL

2

SCHEMA UML DI RIFERIMENTO

Implementare in SQL lo schema del DataBase rappresentato in UML, avendo cura di trattare i vincoli di integrità di colonna e di tabella; i vincoli di integrità referenziale (espressi nel diagramma UML dalle freccia con verso dalla tabella referenziante verso quella refenziata) e delle limitazioni e Policy ragionevoli.



VINCOLO: Lo stipendio è compreso tra 800 e 3000

Ognun vede quel che tu pari. Pochi sentono quel che tu sei.
(Niccolò Machiavelli)

BASI DI DATI

Concetti base – SQL

TEST SCHEMA UML DI RIFERIMENTO

Dopo aver implementato lo schema con dei dati di prova testare se l'implementazione realizzata risponde alle specifiche di cui al Diagramma UML, con particolare riferimento ai vincoli di integrità referenziale, limitazioni e policy definiti e implementati in DDL.
A tal fine utilizzare i seguenti comandi per caricare, cancellare e modificare le istanze delle tabelle:

- **INSERIMENTO DI UNA TUPLA.**
  INSERT INTO *table_name* (*column1*, *column2*, *column3*, ...) VALUES (*value1*, *value2*, *value3*, ...)
  Ad esempio
  INSERT INTO Customers (CustomerName, ContactName, PostalCode)
  VALUES ('Cardinal', 'Tom B. Erichsen', '4006');
- **CANCELLAZIONE DI UNA TUPLA**
  DELETE FROM *table_name* WHERE *condition;*
  ad esempio
  DELETE FROM Customers WHERE CustomerName='Cardinal' ;
- **MODIFICA DI UNA TUPLA**
  UPDATE table_name SET column1 = value1, column2 = value2, ... WHERE condition;
  Ad esempio
  UPDATE Customers SET Postal code = '4008' WHERE *CustomerName='Cardinal'*

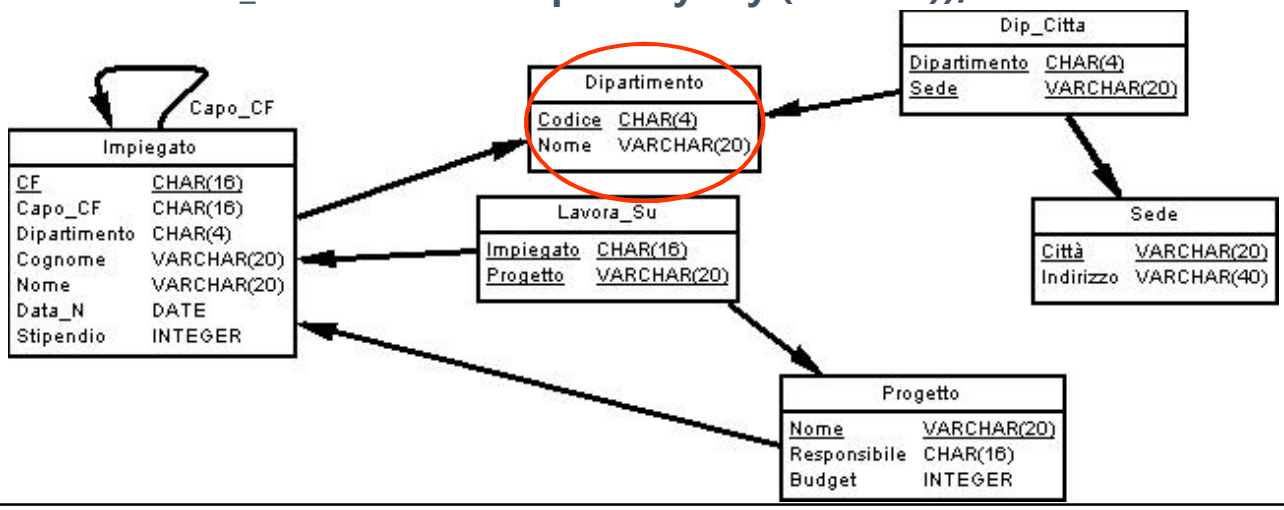
Dove c'è una grande volontà non possono esserci grandi difficoltà.
(Niccolò Machiavelli)

BASI DI DATI

Concetti base – SQL

TABELLA DIPARTIMENTO

create table DIPARTIMENTO (CODICE CHAR(4) not null,
NOME VARCHAR(20) null,
constraint PK_DIPARTIMENTO primary key (CODICE));



L'offesa che si fa all'uomo deve essere tanto grande da non temere la vendetta. (Niccolò Machiavelli)

BASI DI DATI

Concetti base – SQL

TABELLA DIPARTIMENTO

➢ **Codice SQL in Editor**

```
/* Creazione di una tabella
DIPARTIMENTO */

drop table   DIPARTIMENTO;

create table DIPARTIMENTO (
CODICE       CHAR(4) not null,
NOME         VARCHAR(20) null,
constraint PK_DIPARTIMENTO
        primary key (CODICE)
);
```

➢ **Codice SQL generato in PostgreSQL**

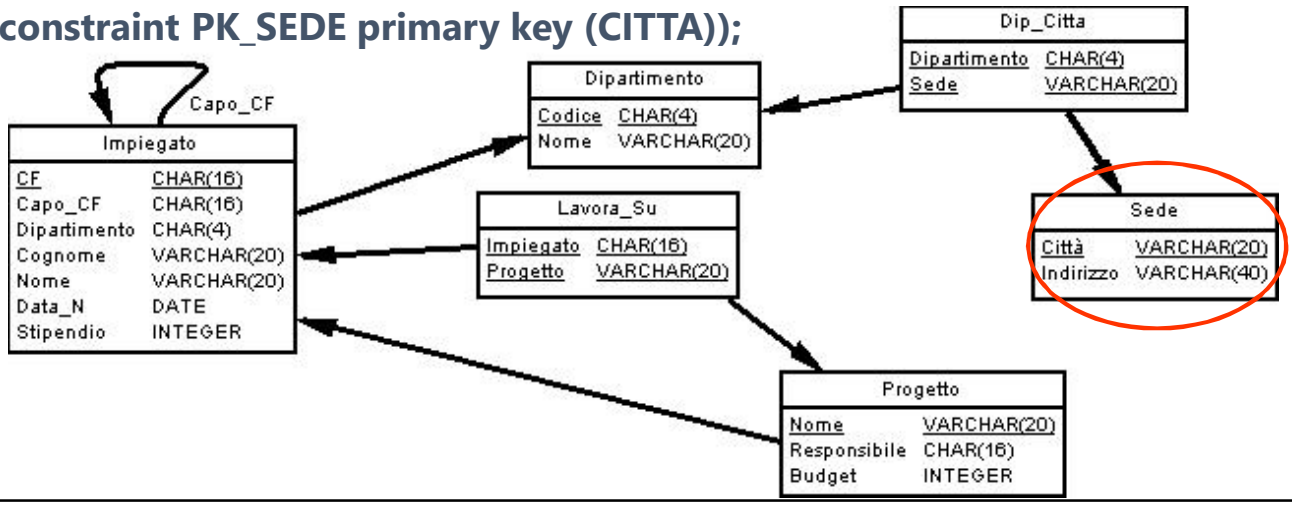
```
-- Table: public.dipartimento
-- DROP TABLE public.dipartimento;
CREATE TABLE public.dipartimento (
codice character(4) COLLATE
pg_catalog."default" NOT NULL, nome
character varying(20) COLLATE
pg_catalog."default", CONSTRAINT
pk_dipartimento PRIMARY KEY (codice)
)
TABLESPACE pg_default;
ALTER TABLE public.dipartimento OWNER
to postgres;
```

BASI DI DATI

Concetti base – SQL

TABELLA SEDE

```
create table SEDE (
CITTA        VARCHAR(20) not null,
INDIRIZZO VARCHAR(40) null,
constraint PK_SEDE primary key (CITTA));
```



BASI DI DATI

Concetti base – SQL

TABELLA SEDE

➢ **Codice SQL in Editor**

```
/* Creazione di una tabella SEDE */

drop table   SEDE;

create table SEDE  (
CITTA         VARCHAR(20) not null,
INDIRIZZO  VARCHAR(40) null,
constraint PK_SEDE primary key
(CITTA)
);
```

➢ **Codice SQL generato in PostgreSQL**

```
-- Table: public.sede
-- DROP TABLE public.sede;
CREATE TABLE public.sede (
citta character varying(20) COLLATE
pg_catalog."default" NOT NULL, indirizzo
character varying(40) COLLATE
pg_catalog."default", CONSTRAINT
pk_sede PRIMARY KEY (citta)
)
TABLESPACE pg_default;
ALTER TABLE public.sede OWNER to
postgres;
```

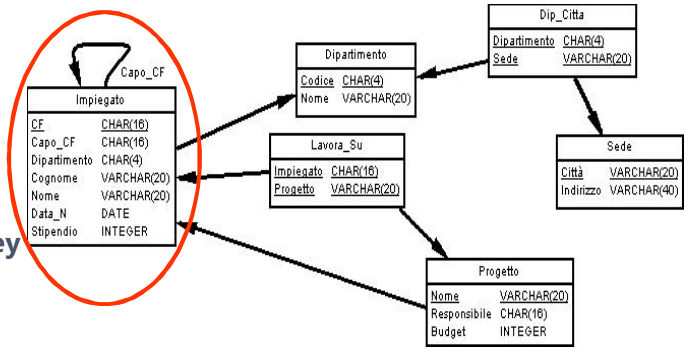
BASI DI DATI

Concetti base – SQL

TABELLA IMPIEGATO

VINCOLO: Lo stipendio è compreso tra 800 e 3000

```
Create table IMPIEGATO (CF char(16),
        CAPO_CF char(16) null,
        DIPARTIMENTO char(4) null,
        COGNOME varchar(20) null,
        NOME varchar(20) null,
        DATA_N date null,
        STIPENDIO integer not null
                constraint CHECK_STIPEDNIO
check(STIPENDIO >= 800 AND STIPENDIO
<=3000),
constraint PK_IMPIEGATO primary key (CF),
constraint FK_CAPO_IS_IMPIEGATO foreign key
(CAPO_CF) references IMPIEGATO (CF)
        on delete restrict on update restrict,
constraint FK_IMPIEGATO_DIPARTIMENTO
foreign key (DIPARTIMENTO) references
                DIPARTIMENTO(CODICE)
        on delete restrict on update restrict
)
```



BASI DI DATI

Concetti base – SQL

TABELLA IMPIEGATO

➢ **Codice SQL in Editor**

```
/* Creazione di una tabella IMPIEGATO */
drop table IMPIEGATO;
create table IMPIEGATO (
CF char(16),
CAPO_CF char(16) null,
DIPARTIMENTO char(4) null,
COGNOME varchar(20) null,
NOME          varchar(20) null,
DATA_N date null,
STIPENDIO integer not null
constraint CHECK_STIPEDNIO check(STIPENDIO >= 800 AND STIPENDIO <=3000),
constraint PK_IMPIEGATO primary key (CF),
constraint FK_CAPO_IS_IMPIEGATO foreign key (CAPO_CF) references IMPIEGATO (CF)
on delete restrict on update restrict,
constraint FK_IMPIEGATO_DIPARTIMENTO foreign key (DIPARTIMENTO) references
DIPARTIMENTO(CODICE) on delete restrict on update restrict );
```

BASI DI DATI

Concetti base – SQL

TABELLA IMPIEGATO

➢ **Codice SQL generato in PostgreSQL**

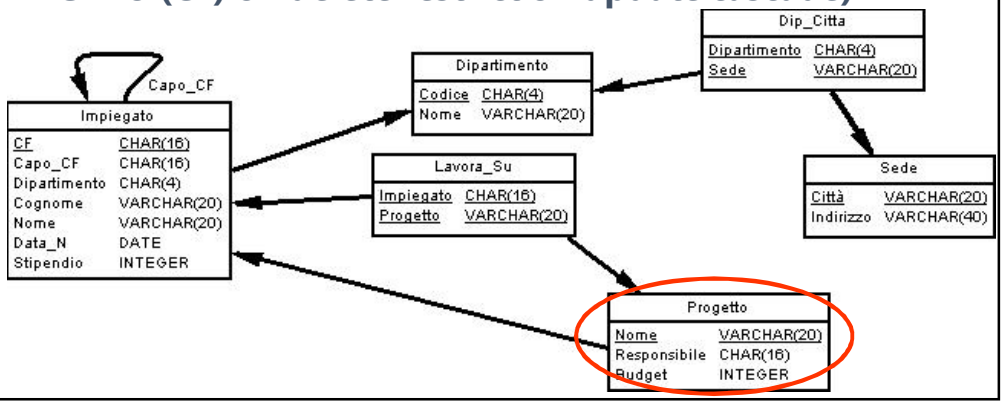
```
-- Table: public.impiegato
-- DROP TABLE public.impiegato;
CREATE TABLE public.impiegato (
cf character(16) COLLATE pg_catalog."default" NOT NULL, capo_cf character(16) COLLATE
pg_catalog."default", dipartimento character(4) COLLATE pg_catalog."default", cognome
character varying(20) COLLATE pg_catalog."default",nome character varying(20) COLLATE
pg_catalog."default", data_n date, stipendio integer NOT NULL,
CONSTRAINT pk_impiegato PRIMARY KEY (cf),
CONSTRAINT fk_capo_is_impiegato FOREIGN KEY (capo_cf) REFERENCES public.impiegato
(cf) MATCH SIMPLE ON UPDATE RESTRICT ON DELETE RESTRICT,
CONSTRAINT fk_impiegato_dipartimento FOREIGN KEY (dipartimento) REFERENCES
public.dipartimento (codice) MATCH SIMPLE ON UPDATE RESTRICT ON DELETE RESTRICT,
CONSTRAINT check_stipednio CHECK (stipendio >= 800 AND stipendio <= 3000))
TABLESPACE pg_default;
ALTER TABLE public.impiegato OWNER to postgres;
```

BASI DI DATI

Concetti base – SQL

TABELLA PROGETTO

```
create table PROGETTO (
   NOME VARCHAR(20) not null,
   RESPONSIBILE CHAR(16) null,
   BUDGET INTEGER null,
   constraint PK_PROGETTO primary key (NOME),
   constraint FK_PROGETTO_RESP_IMPIEGATO foreign key (RESPONSIBILE)
         references IMPIEGATO (CF) on delete restrict on update cascade)
```



BASI DI DATI

Concetti base – SQL

TABELLA PROGETTO

➢ **Codice SQL in Editor**

```
/* Creazione di una tabella PROGETTO */
drop table PROGETTO;
create table PROGETTO (
NOME VARCHAR(20) not null,
RESPONSIBILE CHAR(16) null,
BUDGET INTEGER null,
constraint    PK_PROGETTO primary key (NOME),
Constraint FK_PROGETTO_RESP_IMPIEGATO foreign key (RESPONSIBILE) references IMPIEGATO (CF) on delete restrict on update cascade
);
```

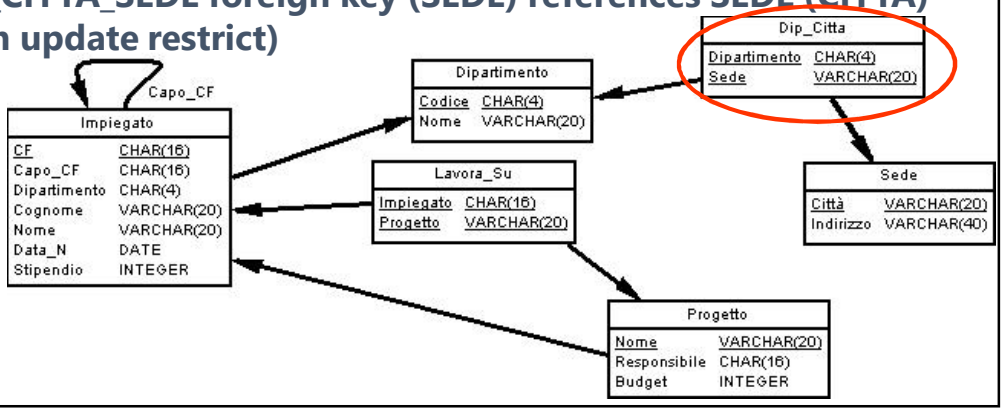
➢ **Codice SQL generato in PostgreSQL**

```
-- Table: public.progetto
-- DROP TABLE public.progetto;
CREATE TABLE public.progetto(
nome character varying(20) COLLATE pg_catalog."default" NOT NULL, responsibile character(16) COLLATE pg_catalog."default",
budget integer,
CONSTRAINT pk_progetto PRIMARY KEY (nome),
CONSTRAINT fk_progetto_resp_impiegato FOREIGN KEY (responsibile) REFERENCES public.impiegato (cf) MATCH SIMPLE ON UPDATE CASCADE ON DELETE RESTRICT)
TABLESPACE pg_default;
ALTER TABLE public.progetto OWNER to postgres;
```

BASI DI DATI

TABELLA DIP_CITTA

```
create table DIP_CITTA (
  DIPARTIMENTO  CHAR(4)   not null,
  SEDE  VARCHAR(20) not null,
  constraint PK_DIP_CITTA primary key (DIPARTIMENTO, SEDE),
  constraint FK_DIP_CITTA_DIPARTIMENTO foreign key (DIPARTIMENTO)
references DIPARTIMENTO (CODICE) on delete restrict on update restrict,
  constraint FK_DIP_CITTA_SEDE foreign key (SEDE) references SEDE (CITTA)
on delete restrict on update restrict)
```



BASI DI DATI

TABELLA DIP_CITTA

➢ **Codice SQL in Editor**

```
/* Creazione di una tabella DIP_CITTA */
drop table DIP_CITTA;
create table DIP_CITTA (
DIPARTIMENTO CHAR(4) not null,
SEDE VARCHAR(20) not null,
constraint PK_DIP_CITTA primary key
(DIPARTIMENTO, SEDE),
ConstraintFK_DIP_CITTA_DIPARTIMENTO
foreign key (DIPARTIMENTO) references
DIPARTIMENTO (CODICE) on delete
restrict on update restrict, constraint
FK_DIP_CITTA_SEDE foreign key (SEDE)
references SEDE (CITTA)  on delete
restrict on update restrict);
```

➢ **Codice SQL generato in PostgreSQL**

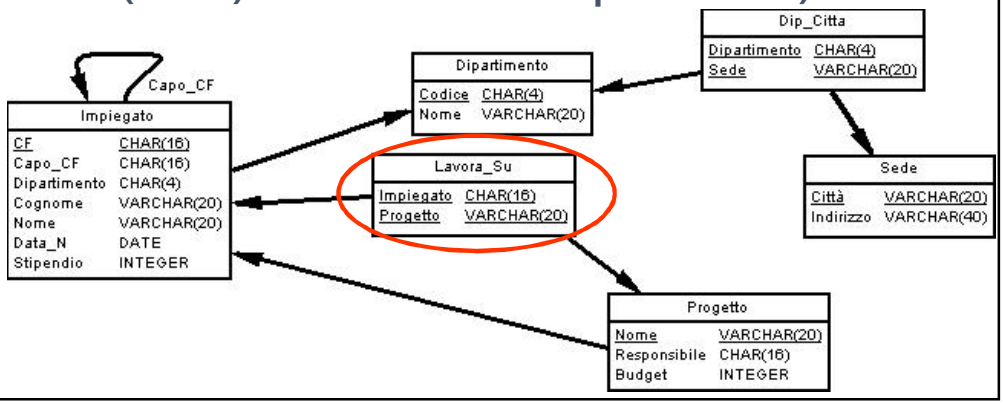
```
-- Table: public.dip_citta
-- DROP TABLE public.dip_citta;
CREATE TABLE public.dip_citta (
dipartimento character(4) COLLATE
pg_catalog."default" NOT NULL, sede character
varying(20) COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT pk_dip_citta PRIMARY KEY
(dipartimento, sede),
CONSTRAINT fk_dip_citta_dipartimento FOREIGN KEY
(dipartimento) REFERENCES public.dipartimento
(codice) MATCH SIMPLE ON UPDATE RESTRICT ON
DELETE RESTRICT, CONSTRAINT fk_dip_citta_sede
FOREIGN KEY (sede) REFERENCES public.sede (citta)
MATCH SIMPLE ON UPDATE RESTRICT ON DELETE
RESTRICT)
TABLESPACE pg_default;
ALTER TABLE public.dip_citta OWNER to postgres;
```

BASI DI DATI

Concetti base – SQL

TABELLA LAVORA_SU

```
create table LAVORA_SU (
  IMPIEGATO CHAR(16) not null,
  PROGETTO VARCHAR(20) not null,
   constraint PK_LAVORA_SU primary key (IMPIEGATO, PROGETTO),
   constraint FK_LAVORA_S_REFERENCE_IMPIEGAT foreign key (IMPIEGATO)
         references IMPIEGATO (CF) on delete restrict on update restrict,
    constraint FK_LAVORA_S_REFERENCE_PROGETTO foreign key (PROGETTO)
         references PROGETTO (NOME) on delete restrict on update restrict)
```



BASI DI DATI

Concetti base – SQL

TABELLA LAVORA_SU

➢ **Codice SQL in Editor**

```
/* Creazione di una tabella LAVORA_SU */
drop table LAVORA_SU;
create table LAVORA_SU  (
IMPIEGATO    CHAR(16) not null,
PROGETTO    VARCHAR(20) not null,
constraint PK_LAVORA_SU primary key
(IMPIEGATO, PROGETTO),
Constraint
FK_LAVORA_S_REFERENCE_IMPIEGAT foreign
key (IMPIEGATO) references IMPIEGATO (CF)
on delete restrict on update restrict,
Constraint
FK_LAVORA_S_REFERENCE_PROGETTO foreign
key (PROGETTO) references PROGETTO
(NOME)  on delete restrict on update restrict)
```

➢ **Codice SQL generato in PostgreSQL**

```
-- Table: public.lavora_su
-- DROP TABLE public.lavora_su;
CREATE TABLE public.lavora_su (
impiegato character(16) COLLATE pg_catalog."default"
NOT NULL, progetto character varying(20) COLLATE
pg_catalog."default" NOT NULL,
CONSTRAINT pk_lavora_su PRIMARY KEY (impiegato,
progetto),
CONSTRAINT fk_lavora_s_reference_impiegat FOREIGN
KEY (impiegato) REFERENCES public.impiegato (cf)
MATCH SIMPLE ON UPDATE RESTRICT ON DELETE
RESTRICT,
CONSTRAINT fk_lavora_s_reference_progetto FOREIGN
KEY (progetto) REFERENCES public.progetto (nome)
MATCH SIMPLE ON UPDATE RESTRICT ON DELETE
RESTRICT)
TABLESPACE pg_default;
ALTER TABLE public.lavora_su OWNER to postgres;
```

BASI DI DATI

Materiale utilizzato e bibliografia

- ***Le slide utilizzate dai docenti per le attività frontali sono in gran parte riconducibili e riprese dalle slide originali (con alcuni spunti parziali ripresi dai libri indicati) realizzate da:***
  - ✓ *autori del libro Basi di Dati (Atzeni e altri) testo di riferimento del corso Basi di Dati e sono reperibili su internet su molteplici link oltre che laddove indicato dagli stessi autori del libro;*
  - ✓ *Prof.ssa Tiziana Catarci e dal dott. Ing. Francesco Leotta – corso di Basi di Dati dell'Università degli Studi La Sapienza di Roma al seguente link ed altri: http://www.dis.uniroma1.it/~catarci/basidatGEST.html (molto Interessanti anche le lezioni su YouTube).*
  - ✓ *Proff. Luca Allulli e Umberto Nanni, Libro Fondamenti di basi di dati, editore HOEPLI (testo di facile lettura ed efficace).*
- ***Diverse slide su specifici argomenti utilizzate dai docenti per le attività frontali sono anche in parte riconducibili e riprese dalle slide originali – facilmente reperibili e accessibili su internet - realizzate da:***

Prof.ssa Roberta Aiello – corso Basi di Dati dell'Università di Salerno
Prof. Dario Maio - corso Basi di Dati dell'Università di Bologna al seguente link ed altri: http://bias.csr.unibo.it/maio
Prof. Marco Di Felice - corso Basi di Dati dell'Università di Bologna al seguente link ed altri: http://www.cs.unibo.it/difelice/dbsi/
Prof Marco Maggini e prof Franco Scarselli - corso Basi di Dati dell'Università di Siena ai seguenti link ed altri: http://staff.icar.cnr.it/pontieri/didattica/LabSI/lezioni/_preliminari-DB1%20(Maggini).pdf
Prof.ssa Raffaella Gentilini - corso Basi di Dati dell'Università di Perugia al seguente link ed altri: http://www.dmi.unipg.it/raffaella.gentilini/BD.htm
Prof. Enrico Giunchiglia - corso Basi di Dati dell'Università di Genova al seguente link ed altri: http://www.star.dist.unige.it/~enrico/BasiDiDati/
Prof. Maurizio Lenzerini - corso Basi di Dati dell'Università degli Studi La Sapienza di Roma al seguente link ed altri http://didatticainfo.altervista.org/Quinta/Database2.pdf

- The PostgreSQL Global Development Group - PostgreSQL nn.xx Documentation
- PostgreSQL (appendice - scaricabile dal sito del libro (area studenti) e www.postgresql.org